SQL Pattern Design, Development & Evaluation of its Efficacy

Huda Al-Shuaily

Submitted in fulfilment of the requirements for the Degree

of Doctor of Philosophy

School of Computing Science

College of Science and Engineering

University of Glasgow

September 2013

Declaration

I declare that this thesis was composed by myself and that the work contained therein is my own, except where explicitly stated otherwise in the text.

(Huda Al-Shuaily)

Abstract

Databases provide the foundation of most software systems. This means that system developers will inevitably need to write code to query these databases. The de facto language for querying is SQL and this, consequently, is the language primarily taught by higher education institutions. There is some evidence that learners find it hard to master SQL.

These issues and concerns were confirmed by reviewing the literature and establishing the scope and context. The literature review allowed extraction of the common issues in impacting SQL acquisition. The identified issues were confirmed and justified by empirical evidence as reported here. A model of SQL learning was derived. This framework or model involves SQL learning taxonomy, a model of SQL problem solving and incorporates cross-cutting factors.

The framework is used as map to the design of a proposed instructional design. The design employed pattern concepts and the related research to structure SQL knowledge as SQL patterns. Also presented are details on how SQL patterns could be organized and presented. A strong theoretical background (checklist, component-level design) was employed to organize, present and facilitated SQL pattern collection.

The evaluation of the SQL patterns yielded new insight such as novice problem solving strategies and the types of errors students made in attempting to solve SQL problems.

SQL patterns, as proposed as a result of this research, yielded statistically significant important in novice performance in writing SQL queries.

A longitudinal field study with a large number of learners in a flexible environment should be conducted to confirm the findings of this research.

Acknowledgements

All thanks and praises go to Allah, the Almighty and the Merciful, for granting me the wisdom and the health to complete this work. I bear witness that there is no true guidance but the guidance from Allah. Alhamdulillah.

I would like to acknowledge the help and the supervision provided to me by Dr. Karen Renaud. Her direct involvement and encouragement throughout the study influenced the positive progress and made me a stronger and better researcher.

I would like to thank all those who have shared their knowledge and provide me with a constructive feedback at some stage of this research. Professor Ray Welland, Dr. Richard Cooper, and Dr. Quintin Cutts.

I would like to thank my sisters who took care of my children while I was diving in this research. Samyia, Tahani , Asma, Fatma, and Thuriay. It is a long journey and I am grateful to have had a supportive family.

To the Ministry of Manpower, I would like to express my gratitude for my scholarship, without this PHD would have been possible.

Dedication

To my mother and father for their sincere prayers and patience;
To my best friend, my husband khalfan for his continuous love and support;
To my future, my heart, my lovely children: Osama, Reenad, Qusay and Sama for the life they bring to our life.

Table of Contents

Abstract 3
1. Chapter 1: Introduction 21
1.2 Applying patterns to enhance SQL learnability 22
1.3 Thesis Statement 23
1.3.1 Objective 1: Identifying impediment that imped SQL novice Learning Performance 24
1.3.2 Objective 2: SQL Patterns Design and Development 26
1.3.3 Objective 3: The impact of SQL Patterns on Learner's Performance 27
1.4 Research Contribution 28
1.4.1 A model of SQL Learning 28
1.4.2 Set of efficacious patterns 28
1.5 Dissertation Structure 29
2. Chapter 2: SQL Learnability 32
2.1 Introduction 32
2.2 How Students Learn: A Learning Taxonomy 34
2.2.1 Computer Science Learning Taxonomy 36
2.2.2 Learning Taxonomy in SQL Teaching and Learning 38
2.3 How Students Learn: A Cognitive Theory in Learning SQL 42
2.3.1 Overview of Cognitive Model of Instruction 42
2.3.2 Cognitive Models in Learning SQL 42
2.3.3 Summary 48
2.4 SQL Instructional Materials Review 49
2.4.1 SQL Teaching Texts 51
2.4.2 SQL Teaching Tools 57
2.4.3 Summary 63
2.5 SQL Content Review 64
2.5.1 Comparison between SQL and other Query Language 66
2.5.2 Comparison between SQL and Natural Language 71
2.5.3 Augmented Use of One Query Language and Database Structure ... 75
2.5.4 Review Summary and Discussion 77

2.6 Instructional Theory Review 79
2.6.1 General Principles of Instructional Design 80
2.6.2 Theory and Research Related to Computer Science Learning 83
2.7 Chapter Summary 85
3. Chapter 3: A Review of the Literature on Patterns' Design, Organization and Usability 89
3.1 Introduction 89
3.2 Patterns Definition 90
3.3 Patterns History 92
3.3.1 Patterns in Architecture Design 92
3.3.2 Patterns in Software Engineering 93
3.3.3 Patterns in HCI 94
3.3.4 Patterns in Education 94
3.3.5 Anti-patterns 95
3.4 Patterns Structure and Format 96
3.4.1 Alexander's Patterns Structure "Alexandrian form" 97
3.4.2 Software Engineering Patterns Structure 99
3.4.3 Human Computer Interaction (HCI) Patterns Structure 99
3.4.4 Pedagogical Patterns Structure 101
3.4.5 Summary 102
3.5 Patterns Collection and Organization 104
3.5.1 Patterns Collection in Architecture 104
3.5.2 Patterns Collection in Software 105
3.5.3 Patterns Collection in HCI 105
3.5.4 Patterns Collection in Education 108
3.5.5 Summary 109
3.6 Patterns Usability 110
3.7 Patterns Efficacy in Education 111
3.7.1 The Review Design 112
3.7.2 Past Experimental Work 115
3.7.3 Review Summary 125
3.8 Patterns and the Cognitive Load Theory 127
3.9 Chapter Summary 128

4. Chapter 4: Research Methodology and Approach 131
4.1 Introduction 131
4.2 Research Setting 132
4.3 Research Strategy 133
4.4 Research Design 134
4.4.1 Identify SQL impediments that get in the way of Learning Performance 136
4.4.2 Development of SQL patterns 136
4.4.3 Assess the efficacy of the designed SQL patterns 138
4.5 Objective 1: Methodology Design 138
4.5.1 Research Questions 138
4.5.2 Tasks Definitions and Measurement 140
4.5.3 Tasks Development 141
4.5.4 Results Analysis 147
4.6 Objective 2: Methodology Design 148
4.6.1 Research Questions 149
4.6.2 Tasks Definitions and Measurement 149
4.6.3 Task Developments 151
4.6.4 Results Analysis 161
4.7 Objective 3: Methodology Design 161
4.7.1 Research Questions 162
4.7.2 Tasks Definitions and Measurement 163
4.7.3 Experiment's Variable and their Measurement 164
4.7.4 Task Development 165
4.7.5 Experiment Setups (environments and materials) 172
4.7.6 Results Analysis 174
4.8 Chapter Summary 175
5. Chapter 5: Analysis of SQL Learning Performance Objectives 177
5.1 Introduction 177
5.2 Data Collection 178
5.2.1 Semi-structured Interview 180
5.2.2 Ability to Solve SQL Problems 183
5.2.3 Educator's Perspective 184

5.2.4 Students' Questionnaire 189
5.2.5 SQL Comprehension 191
5.3 Analysis Strategy 194
5.4 The Analysis of Learner's Characteristics 195
5.4.1 Learner's Personal Attitude towards Learning SQL 195
5.4.2 Learners' Previous Knowledge and Experience 198
5.4.3 Learner's Problem Solving Abilities 201
5.4.4 Summary of the Characteristic of Novice Learners 207
5.5 The Analysis of SQL Language Features 209
5.5.1 SQL Nature 209
5.5.2 SQL Syntax 210
5.5.3 SQL Content 213
5.5.4 Summary of SQL Features and their Relation to SQL Learning 214
5.6 Analysing the Influence of SQL Learning Methods and Approaches 215
5.6.1 SQL Curriculum Design 217
5.6.2 SQL Query Comprehension 219
5.6.3 SQL Query Formulation and Translation 220
5.6.4 SQL Query Writing 222
5.6.5 Summary 224
5.7 Discussion and Interpretation 225
5.8 A Model of SQL Learning 229
5.8.1 Cognition and Mental Models 231
5.8.2 SQL Learning Taxonomy 234
5.8.3 Cross-Cutting Factors 236
5.9 Chapter Summary 236
6. Chapter 6: SQL Patterns Design and Development 239
6.1 Introduction 239
6.2 The Motivation for Using Patterns as an Instructional Material 241
6.3 SQL Patterns Identification Process 243
6.3.1 Problem Solving Strategy Identification via Mining 245
6.3.2 Problem Solving Strategy Identification through Observation 248
6.3.3 SQL Patterns Identification through Expert Observation 251
6.3.4 Summary 259

6.4 SQL Patterns Structuring Process.....261
6.4.1 SQL Patterns Format (Phase 1) 263
6.4.2 SQL Patterns Format (Phase 2) 269
6.4.3 SQL Patterns Format Phase 3 273
6.5 SQL Patterns Organization Process.....277
6.5.1 Checklist Approach 278
6.5.2 Patterns' Graphical Representation 279
6.5.3 Discussion 286
6.5.4 Summary 287
6.6 SQL Patterns' Evaluation Process 288
6.6.1 Evaluation of Phase 1 of SQL Patterns Design 288
6.6.2 Evaluation of Phase 2 of SQL Patterns Design 291
6.6.3 Evaluation of Phase 3 of SQL Patterns Design 292
6.7 Chapter Summary.....292
Chapter 7:.....295
7. The impact of SQL Patterns on Learner Performance 295
7.1 Introduction 295
7.2 Experimental Setups 297
7.3 The impact of SQL Patterns in Knowledge Acquisition.....299
7.3.1 Students' Understanding of SQL Tested Concepts Prior to the Experiment.....299
7.3.2 The Impact of Patterns in Students' Understanding of SQL Concepts in Response to the Experiment 300
7.4 The Impact of SQL Patterns in Problem Solving Skills.....303
7.4.1 Query formulation and translation.....304
7.5 The Impact of SQL Patterns in Intermediate Attempts 307
7.5.1 Decision tree 307
7.5.2 Overview of learners' strategy 309
7.5.3 Participant Attempts: Collection, Classification and source Identification 315
7.6 The impact of SQL pattern in Participants' Performance at Query Writing 323
7.7 SQL Pattern Usability 328

7.8 Results Analysis 332
7.8.1 The Impact of SQL pattern on knowledge Acquisition 332
7.8.2 The impact of SQL pattern in problem solving strategy 335
7.8.3 The Impact of SQL Patterns in Intermediate Attempts 338
7.8.4 The impact of SQL patterns in Query writing 344
7.9 Discussion and Recommendation 347
7.9.1 SQL Acquisition 348
7.9.2 Problem solving skills 349
7.9.3 Error analysis 353
7.9.4 SQL Patterns Usability 360
7.10 Summary 361
8. Chapter 8: Conclusion 364
8.1 Research Contribution 364
- A model of SQL learning 364
- Set of efficacious SQL patterns 364
8.2 Achievement of Thesis Statement's Objectives 365
8.3 Future Work 367
- Teaching Practice 367
- Pattern Design and Development 368
List of References 370
Appendix A: The Semi-structured Interview 401
Appendix B: Problem Analysis and Synthesis Task 403
Appendix C: Online questionnaire- Academic 404
Appendix D: Learning SQL – Questionnaire 406
Appendix E: Comprehension Task 407
Appendix F: Expert Observation Task 408
Appendix G: SQL Patterns Evaluation Task one 409
Appendix H: SQL Patterns Evaluation Task – Pre-test 411
Appendix I: SQL Patterns Evaluation Task – Post-test 413
Appendix J: SQL Patterns Evaluation Task –Questions 417
Appendix K: SQL Patterns Evaluation Task – Tutorial 421
Appendix L: SQL Patterns Evaluation Task – consent Form 425
Appendix M: SQL Patterns Evaluation Task – SPSS Form 426
Appendix N:SQL Patterns Evaluation-Usability Questionnaire 430
Appendix O: SQL Patterns Evaluation –Ethical clearance 432
Appendix p: SQL Patterns 433

List of Tables

Table 1.1: Research Question 1.1 24
Table 1.2: Research Question 1.2 25
Table 1.3: Research Question 1.3 25
Table 1.4: Research Question 2 26
Table1.5: Research Questions, Question 3 27
Table 2.1: Form Used in The Review 52
Table 2.2: Tools Review Rating (1-5) 59
Table 2.3: List of Literature Resources for Comparisons Between SQL and Other Query Language 66
Table 2.4: Research Conducted to Compare SQL and Natural Languages 71
Table 2.5: Summary of the Research Conducted to Augment the Use of Query Language and Database Structure. 75
Table 3.1: Patterns Element 96
Table 3.2: Pedagogical Patterns Structures [122] 101
Table 3.3 : The Published Experimental Research in The Field of Teaching HCI Patterns 115
Table 4.1: Research Questions Align with RQ 1 139
Table 4.2: Research Methods used for Research Question 1 142
Table 4.3: Main Components of Semi-Structured Interview 143
Table 4.4: Student's Questionnaire 147
Table 4.5: Research Questions Align with RQ 2– SQL Patterns Design Process . 149
Table 4.6: Research Methods Used for Research Question 2 153
Table 4.7: Time Spent with Novice SQL Learners 155
Table 4.8: The Questions Used to Direct the Unstructured Observation. 156
Table 4.9: Research Questions – The Effect of SQL Patterns in Learners' Performance 162
Table 4.10: Research Methods Used for Research Question 3 165
Table 4.11: The Tests Used in The Experiment 167
Table 5.1: Research Methods Employed 179
Table 5.2: Result of the Semi-Structured Interviews, SQL Misconception 182

Table 5.3: Result of the Online Questionnaire, SQL Misconception.....189
Table 5.4: Response to Question 5 in student's Questionnaire.....191
Table 5.5: Participant's Task Analysis Results **.Correlation Significant at the 0.01 Level (2- Tailed).....193
Table 5.6: Participants' Response Towards SQL Experience.....198
Table 5.7: Participants' Response Towards Problem Solving 203
Table 5.8: Correlation Between Solving Simple SQL Problems and Complex SQL Writing 203
Table 5.9: Participant's Response to SQL Nature.....210
Table 5.10: Participant's Response to SQL Syntax 212
Table 5.11: Participant's Response to SQL Content.....213
Table 5.12: Results on the Agreement on SQL Concepts Difficulties 214
Table 5.13: Participant's Response to SQL Learning Experience.....215
Table 5.14: Participant's Response in Relation to SQL Query Formulation and Translation 220
Table 5.15: Participant's Response in Relation to SQL Query Writing.....223
Table 6.1: Time Spent with Novice SQL Learners 248
Table 6.2: Example Illustrating Problem Based Learning Steps.....264
Table 6.3: Patterns Structure in Phase 1 265
Table 6.4: An Example of a Pattern at Phase-1 Development.....267
Table 6.5: An Example of a Pattern at Phase 2 Development 272
Table 6.6: Pattern Structure at Phase 3 Development 274
Table 6.7: Patterns' Structure at Phase 3. 276
Table 6.8: Checkilist and Related Patterns-Example 1 284
Table 6.9: Checkilist and Related Patterns-Example 2 285
Table 7.1 : Research Question 3 296
Table 7.2: Chapter Structure 296
Table 7.3 : The numbers of students who participated in the tree tests in both groups 298
Table 7.4 : Participants Results in the Pre-test.....299

Table 7.5: Percentage of Students' Response in the Pre-test and Summary of Independent-Sample t-test (control group N=48, Experimental group N= 35 note: * not significant at 0.05) 300
Table 7.6: Participants' Results in the Post-test 300
Table 7.7: : Percentage of Students' Response in the Post-test and Summary of Independent-Sample t-test 301
Table 7.8: Paired-Sample Test 302
Table 7.9: Marking Schema for Sample Answer to Problem Formulation and Translation 305
Table 7.10: Problem Solving Task 305
Table 7.11: Number of Participants Submitted doc for Analysis and Synthesise 305
Table 7.12: The Average of each Question 306
Table 7.13: Participant Performance Matrix 310
Table 7.14: Example of the Number of Attempts Per Question 312
Table 7.15: Number of Attempts per Question-Control Group 312
Table 7.16: Number of Attempts per Question-Patterns Group 314
Table 7.17: The Error Classification Compare to other Research 318
Table 7.18: Example of Error Source Identification 320
Table 7.19: Source of Error 321
Table 7.20: Completeness Rubric 324
Table 7.21: The Classification of the Correctness of the Final Attempt. 324
Table 7.22: Query Correctness Rubric 325
Table 7.23: Examples of Error Weight 325
Table 7.24: Participants Problem Solving Performance Scores 326
Table 7.25: The Mean Score of Satisfaction 329
Table 7.26: Mean, Median & Mode of Variables 330
Table 7.27: SQL concepts examined in the experiment 333
Table 7.28: Example of Control Group Problem Solving Strategy 340
Table 7.29: Error Frequency 341
Table 7.30: Example of the Most Common Errors 343
Table 7.31: Overall Participants' Performance 345
Table 7.32: Example of Errors 355

List of Figures

Figure 1.1: Research Objectives 23
Figure 1.2: Dissertation Structure 29
Figure 2.1: Learning Taxonomies 35
Figure 2.2: Taxonomy of Task Types in Computing (Adapted from [46]) 37
Figure 2.3: CS Learning Taxonomy 38
Figure 2.4: Shneiderman Five Tasks and The Related Task in Learning Taxonomy 39
Figure 2.5: Spiral Model for Learning through Construction 40
Figure 2.6: Query Writing Model Adapted Reisner [57] 43
Figure 2.7: Query Writing Model Adapted from Mannino's [58] 43
Figure 2.8: Three-Stage Cognitive Model Adapted from Ogden [59] 43
Figure 2.9: SQL Cognitive Model 45
Figure 2.10: The Practice Stage of SQL Learning Taxonomy 46
Figure2.11: The Level of Knowledge Rate (0-5) within the Reviewed Book 57
Figure 2.12: Example of Comprehension Knowledge 54
Figure 2.13:Example of Debugging Task [68] 55
Figure 2.14: Example of "why" Knowledge [68] 56
Figure 2.15: Tools review Result 63
Figure 2.16: ADDIE Core Elements [103] 84
Figure 2.17: Instruction Design Model [104] 81
Figure 2.18: Employing Instructional Design Model [104] in This Dissertation ... 82
Figure 2.19: Problem Solving Approach Adapted from Quilici J.H., & Mayer R. E. [107] 84
Figure 3.1 :Pattern's Definition [115] 90
Figure 3.2: A Sample of a Pattern by Alexander et al. [116] 101
Figure 3.3: Patterns Common Elements 106
Figure 3.4: An Example Class Structure of an Ontology for Usability [181] 108
Figure 3.5: Teaching UI, Pattern Language (TUI) [171] 109
Figure 3.6: The Review Form 114
Figure 3.7: Applied Educational Research Adapted from Pears et al. [207] ... 135

Figure 4.2: A Framework for CRM Designed to Facilitate Teaching [206].....134
Figure 4.3: A Spiral CRM Framework for Research Methods Designed to Achieve the Intended Goals Adapted from [206] 135
Figure 4.4: Sequence of the Research Methods (Research Question 1) 145
Figure 4.5: Question Used as Part of Task Analysis.....145
Figure 4.6: SQL Knowledge Identification Process.....150
Figure 4.7: SQL Patterns Design Process 151
Figure 4.8: SQL Pattern's Design Phases 152
Figure 4.9: Expert Observation Task.....161
Figure 4.10: Instructional Materials Elements.....159
Figure 4.11: Research Instruments Used for Research Question 3 166
Figure 4.12: Experiment Steps 176
Figure 4.13: Experiment Program.....174
Figure 5.1: Research Methods Sequences 184
Figure 5.2: learners' Knowledge Rating.....180
Figure 5.3: Students Rating of the Difficulties of SQL Concepts 182
Figure 5.4: Query Formulation Related Task 183
Figure 5.5: Teacher's Level of Experience with SQL 184
Figure 5.6: Do you believe a Solid Grounding in Set Theory Helps Students Understand Database Concepts? 185
Figure 5.7: Participant's Level of Knowledge and Experience with SQL 190
Figure 5.8: Analysis of Results Relating to SQL Query Comprehension 192
Figure 5.9: Task Analysis Results.....193
Figure 5.10: CS Learning Taxonomy (right) and its Related Kind of Knowledge (left).....205
Figure 5.11: Participant's Response to SQL Syntax 215
Figure 5.12: CS Learning Taxonomy.....220
Figure 5.13: Cognitive Model of Query Learning 228
Figure 5.14: SQL Problem Solving – The Practice Stage of SQL Learning Taxonomy 228

Figure 5.15: A Model of SQL Learning 231
Figure 5.16: The Role of Schemata in Problem Solving (left) and a Trial and Error Approach (right) 232
Figure 5.17: Transfer Approach 233
Figure 5.18: SQL Learning Taxonomy.....235
Figure 6.1: A Model of SQL Learning.....240
Figure 6.2: SQL Patterns Design Process 241
Figure 6.3: IRPLane Identification Process by Wania [310].....244
Figure 6.4: SQL Iterative Pattern Design Phases.....245
Figure 6.5: Mining Process 246
Figure 6.6: Patterns Identification Through Novice Observation 253
Figure 6.7: Novice Strategies in Problem Solving.....250
Figure 6.8: Expert Observation Process 252
Figure 6.9: Expert Observation Task.....252
Figure 6.10: Snapshot of the Cognitive Activities Performed by Experts 254
Figure 6.11: Expert Cognitive Activities 255
Figure 6.12: Expert Problem Solving [318] (left) and SQL Acquisition on Expert Model (right) 256
Figure 6.13: Typical Expert Actions (left) and Novice Actions (right) 258
Figure 6.14: Knowledge Within Pattern 259
Figure 6.1: SQL Learning Taxonomy..... 272
Figure 6.16: : Essential Pattern Section and Their Writing Order [322].....269
Figure 6.17: Expert Problem Solving [318] (left), A SQL Learning Model (right)..... 277
Figure 6.18: Graphical Representation..... 283
Figure 6.19: Level 0 SQL Patterns' Representation 281
Figure 6.20: Level 1 SQL Patterns Representation 281
Figure 6.21: Level 2 of Patterns Presentation 282
Figure 6.22: Level 0 SQL Pattern Representation.....283
Figure 6.23: Level 1 of Pattern Presentation.....283
Figure 6.24: Level 2 of Pattern Presentation..... 288

Figure 6.25: Deploying Patterns.....285
Figure 6.26: Deploying Patterns..... 289
Figure 6.27: Problem Solving Using the Pattern Collection (adapted from Luseau's design Model [327]) 287
Figure 6.28: ExperimentProcedures.....302
Figure 7.2: Novice Problem Solving Task.....303
Figure 7.3: Analysis & Synthesise Stages.....304
Figure 7.4: The Average of PSS-Control (left), The Average of PSS-experiment (right) 306
Figure 7.5: A sample of a decision Tree.....309
Figure 7.6: The Number of Questions Solved by Students..... 316
Figure 7.7: Number of Students among the Question Type 311
Figure 7.8: Students Strategy in Solving the Questions.....312
Figure 7.9: The Number of Questions Solved by Students.....313
Figure 7.10: Number of Students among the Question Type.....313
Figure 7.11: Students Strategy in Solving the Questions 314
Figure 7.12: A Snapshot from the Tool.....316
Figure 7.13: A Snapshot of the Collected Queries from the Tool 316
Figure 7.14: Classification of Human Errors adapted from Reason [337] 317
Figure 7.15: Error Classification Model.....318
Figure 7.16: Error Classification Process.....319
Figure 7.17: SQL Errors from SQL Nature and Cognition Perspective.....326
Figure 7.18: Average of Question Completion-Control.....326
Figure 7.19: Average of Question Correctness-Control 327
Figure 7.20: Average of Question Completion- Experiment.....327
Figure 7.21: Average of Question Correctness- Experiment 328
Figure 7.22: Usability Measurements.....328
Figure 7.23: Time Required Understanding Patterns.....330
Figure 7.24: Which Part was More Helpful in problem solving?.....331
Figure 7.25: Which Part was More Helpful in SQL Acquisition? 332
Figure 7.26: Paired Samples Test (Sig.2 tailed),.....333
Figure 7.27: Both Groups Performance at Query Writing.....334

Figure 7.28: Example of Error in Query Formulation.....337
Figure 7.29: Sample Answer for Question 4 337
Figure 7.30: Number of Student who Specify the Correct Pattern.....337
Figure 7.31: Problem Solving Process 343
Figure 7.32: Both Groups Performance at Query Writing.....345
Figure 7.33 : Time Allocation per Question 346
Figure 7.34: Time Distribution Pattern.....347
Figure 7.35: The Effect of Patterns in SQL Acquisition 348
Figure 7.36: Participant Analysis Skill 349
Figure 7.37 : The Role of Schemata in Problem Solving in Expert and Novice ..350
Figure 7.38: problem solving model(top), teaching problem solving oppsitly (bottom) 351
Figure 7.39: Trial and Error Strategy 354
Figure 7.40: Two Error: Rule-Based & Syntax.....361
Figure 7.41: Multiple errors.....357
Figure 7.42: SQL syntax.....358
Figure 7.43: Participant Comments.....360
Figure 7.44: Pattern Content in the Learning Taxonomy.....360

Associated Publications

As a result of this work, there were nine associated peer-reviewed publications. These were as follows:

[1] H. Al-Shuaily, K. Renaud, " *Exploring and Explaining SQL Errors*" (submitted for review).

[2] H. Al-Shuaily, K. Renaud, "*SQL pattern Organization and Presentation*" accepted in the 1st workshop on Patterns Promotion and Anti-patterns Prevention (PPAP), Collocated with the 17th European Conference on Software Maintenance and Reengineering (CSMR 2013), Genova Italy (2013).

[3] H. Al-Shuaily, K. Renaud, "*SQL Pattern Design and Development*". " (submitted for review).

[4] H. Al-Shuaily, K. Renaud, "*A Model of SQL Learning*" accepted in 10th World Conference on Computers in Education(WCCE),Torunm, Poland (2013).

[5] H. Al-Shuaily, "*Analyzing The Influence Of SQL Teaching and Learning Methods and Approaches?* In 10th workshop on Teaching, Learning and Assessment in Database, UK, HEA (2012).

[6] H. Al-Shuaily, "*SQL Patterns* " Euro PLOP, Irsee, Germany (2011).

[7] H. Al-Shuaily, K. Renaud."*A REVIEW ON THE USE OF PATTERNS IN COMPUTER SCIENCE EDUCATION*" . The International Conference of Education, Research and Innovation, Madrid, Spain iCERi 2010.

[8] H. Al-Shuaily, K. Renaud, "*SQL Patterns- A New Approach for Teaching SQL* " In 8th workshop on Teaching, Learning and Assessment in Database, Dundee, UK, HEA (2010).

[9] K. Renaud, H. Al-Shuaily, R. Cooper, "*Facilitating Efficacious Transfer of Database Knowledge and Skills*" Proc. of The 26th British National Conference on Databases BNCOD 2009, pp.25-38.

Chapter 1: Introduction

It is important for information technology practitioners to be able to query databases, since databases can be found under the hood of just about every major computer application, providing access to essential corporate information [1]. Querying is achieved by writing SQL, in the vast majority of cases. If this is done poorly it affects performance across the entire application.

Query and database manipulation were listed among the set of core database skills that students need to master. Database knowledge and skills are vital to organizations and companies. Some European surveys found that this is the skill that companies consider to be most lacking in new IT graduate recruits [2].

SQL is taught at most universities. Yet novices tend to be rather poor at writing SQL. It is worth understanding why. There are various views about SQL learnability that were explored in the literature. Mitrovic [3] points out that although SQL is simple and highly structured, students still have difficulties learning it. Researchers have attempted to identify the factors that affect SQL learning and use. Some of these factors can be termed human factors [4-6], while others can be related to the physical teaching environment and the type of task [6]. The impact of query language features was investigated in terms of learning and using the language [4, 5, 7-9]. Many studies attribute these difficulties to the nature of SQL as a declarative language, arguing that it is fundamentally different from the other programming languages that students have to learn [3, 10-12]. The effect of the teaching method was also studied by Schlager *et al*. [13].

While these different aspects undoubtedly contribute to the difficulties students experience with SQL, there is no agreement, so far, on how to go about remedying the situation.

This dissertation embarks on a journey to resolve the issues associated with SQL learnability. The literature suggests that the symptoms of this problem revolve around: the characteristics of the learner, the features of the language and the methods utilized in transferring the knowledge. The dissertation will build on the literature in order to develop a more accurate understanding of the problem, and will present a remedy designed in the light of this new understanding, and evaluated in real classrooms.

1.2 Applying patterns to enhance SQL learnability

Patterns are a widely accepted mechanism for supporting knowledge transfer. Patterns were first adopted in education to teach architecture students about aspects of urban design [14]. In Software Engineering [15, 16] both recommended using patterns to teach novices. Astrachan *et al*. [16] argued that patterns should form an essential part of the undergraduate Computer Science curricula.

Patterns traditionally structure knowledge in such a way that they can transfer best practice from experts to novices. Schlager and Ogden [13] found that incorporating a cognitive model in the form of expert user knowledge into novice instruction enhances learning, and this is essentially the rationale for patterns of any kind. This research therefore sets out to examine whether the use of SQL patterns during instruction could help novice SQL learners.

The term "SQL patterns" was coined by Faroult and Robson in [17]. They stated that the SQL patterns in their book were specifically produced for professional SQL developers who need to solve complex problems using common SQL idioms. However, the novice learner cannot utilize these particular patterns because of their limited knowledge and experience in writing SQL.

It cannot be assumed that SQL patterns can be designed and developed in exactly the same way as other more well-established patterns, so there is a need to carefully align SQL pattern design and development with what has been learnt

about the characteristics of the novice learner, the features of the SQL language and the methods utilized in transferring SQL knowledge.

1.3 Thesis Statement

SQL learners encounter well-documented difficulties that impair the SQL acquisition process. The purpose of this research is to determine whether SQL patterns can play a role in improving SQL acquisition by novices. Hence the thesis statement is:

It is possible to create SQL patterns which improve SQL learning by novices.

The thesis statement is broken down into three objectives as follows, each of which is addressed in an interrelated manner in this thesis.

1. **To identify SQL impediments that that imped SQL novice learning performance**
2. **To design and develop SQL patterns as informed by these research findings**.
3. **To assess the efficacy of the designed SQL patterns**.

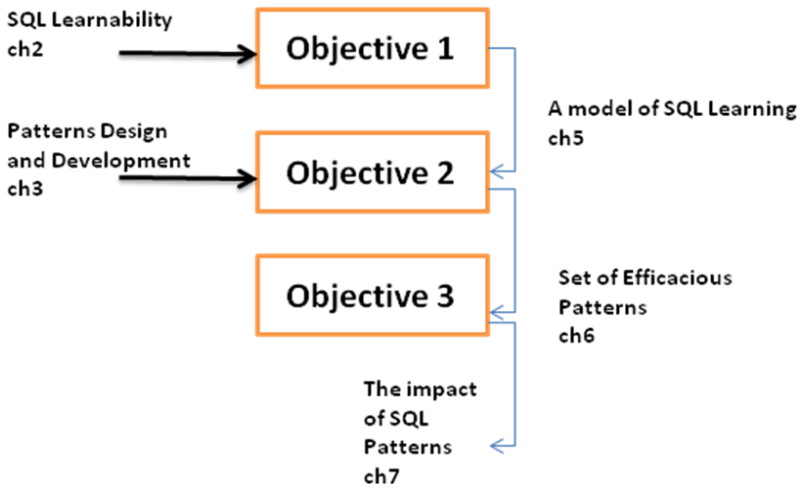


Figure 1.1: Research Objectives

The interrelation between these objectives is shown in Figure 1.1. Each objective has an input and an output. The output is used as input to the following objective.

The thesis statement will be proven if the patterns designed do indeed address the difficulties that are experienced by students, and this will be measured by the experiment and evaluation. The following subsections explore these objectives in details.

1.3.1 Objective 1: Identifying impediment that imped SQL novice Learning Performance

Objective 1 aims to investigate the issues in learning SQL reported in the literature. These issues will be extended and corroborated by surveys with teachers and students, conducted as part of this research. The results will then be interpreted in the context of learning and cognitive theories and reviews of problem solving. These analyses will enable a separation of reported learning challenges specific to SQL from more generic challenges. There are three research questions that are related to this objective (see Table 1.1, Table 1.2 and Table 1.3).

Research Question 1.1: What are the effects of the following novice SQL learner characteristics?		Results and analysis	Research methods
1	Personal attitude toward learning SQL	Section 5.4.1	Semi-structured interviews Student- questionnaire's Online questionnaire Section 4.5.3.
2	Previous knowledge and experience	Section 5.4.2	
3	Problem solving skills	Section 5.4.3	

Table 1.1: Research Question 1.1

Research Question 1.2: What are the effects of the following aspects of SQL language?		Results and analysis	Research methods
1	The declarative nature of SQL	Section 5.5.1	Literature review in chapter 2
2	The syntax of SQL	Section 5.5.2	Semi-structured, interviews and online questionnaire online questionnaire all in section 4.5.3
3	The content of SQL	Section 5.5.3	

Table 1.2: Research Question 1.2

Research Question 1.3: What is the impact of the current teaching methods and approaches in the following aspects of learning SQL?		Results and analysis	Research methods
1	Novices' ability in reading and comprehension of SQL queries (query comprehension)	Section 5.6.2	Comprehension task in Section 4.5.3
2	Novices' ability to understand the given scenario (query formulation)	Section 5.6.3	Cognitive task in Section 4.5.3
3	Novices' ability to translate the given problem (query translation)		
4	Novices' ability to write non-trivial query (query writing), which is the application of their knowledge	Section 5.6.4	

Table 1.3: Research Question 1.3

Research Outcome

The main outcome of objective 1 is "A model of SQL learning". This model presents the performance objectives to be used as a map to facilitate the Instructional Design objective which will be described in section 5.8.

This model is based on a new interpretation of SQL acquisition as being influenced by cross-cutting human factors, the nature of SQL itself, learning theory (SQL learning taxonomy) and cognitive science (development of mental model throughout the learning process).

1.3.2 Objective 2: SQL Patterns Design and Development

Objective 2 of the research aims to identify the design of a new instructional material, building on the results of objective 1, "A model of SQL learning". The model was ideal as a launching pad for the investigation into potential SQL patterns. Moreover, patterns concepts and related research, covered in chapter 3, are employed to structure and organize SQL patterns. The following are the related research questions:

No	Research Question 2 SQL patterns design and development process	Results and analysis	Research methods
1	How should SQL patterns be defined and what should they contain?	section 6.2	Literature review in chapter 3
2	How should SQL patterns be identified?	Section 6.3	Text mining observation : novices & experts Section 4.6.3
3	How should SQL patterns be structured?	Section 6.4	Literature review in chapter 3 shepherding process
4	How can SQL patterns be organized?	Section 6.5	Literature review in chapter 3

Table 1.4: Research Question 2

Research Outcome

Objective 2 of the research contributes toward formulating the strategy of the design and the development of SQL patterns as instructional material that employs both pattern knowledge and the understanding of all the different factors that influence SQL learnability. Thus, SQL designed pattern and development has provided the guidance to inform pattern content, which should ultimately serve as the link between the task requirement and the generic pattern.

1.3.3 Objective 3: The impact of SQL Patterns on Learner's Performance

Objective 3 aims to carry out an experiment with novices to determine whether SQL patterns help them in mastering SQL skills. The impact of SQL patterns on SQL knowledge acquisition is examined and the efficacy of SQL patterns assessed in terms of how well it supports SQL problem solving.

Research Question 3: What is impact of SQL patterns in learners' performance?			Results and analysis	Research methods
1	Do SQL patterns improve SQL knowledge acquisition?		Result in section 7.3 Analysis in section 7.8.1	Pre-test Post-test Section 4.7.4
2	Do SQL patterns improve the following aspects of novices' performance?			Problem solving test Query writing test Section 4.7.4
	A	Problem solving	Result in section 7.4 Analysis in section 7.8.2	
	B	Intermediate attempts	Result in section 7.5 Analysis in section 7.8.3	
	C	Query writing	Result in section 7.6 Analysis in section 7.8.4	
3	How have participants felt about the efficacy of the patterns?		Result in section 7.7 Analysis in section 7.8.5	Questionnaire

Table1.5: Research Questions, Question 3

Research Outcome

The research general outcome at this stage is to determine how effective SQL patterns can be when compared to the traditional way of teaching SQL. There are five specific contributions of this study:

- Evaluating and confirming SQL misconceptions.
- Understanding of learners' strategy during problem solving.
- The impact of the SQL pattern in query correctness.
- Attempts analysis which is employed to understand the reasons behind the errors, which learners commit during problem solving.
- Evaluate the patterns usability.

1.4 Research Contribution

This research will enhance the understanding of the problem encountered in teaching SQL as well as the effects of using SQL patterns in education, and will help to develop interactive methods for using them during knowledge transfer. The following are the main research contributions:

1.4.1 A model of SQL Learning

The research will contribute to the theoretical knowledge of the problems encountered when learning to express queries in SQL and will provide an empirical evidence of the stated issues. The research contribution is extended in how SQL knowledge can be identified, recorded, reviewed and used, especially in novice education.

1.4.2 Set of efficacious patterns

The research employed patterns concepts and the related research to structure SQL knowledge and called it SQL patterns. The research aims to contribute by formulating an approach defining the design of SQL patterns; focusing on maximizing the efficacy of SQL patterns in transferring experts knowledge, especially for the novice learner, arguably the most important target audience. In addition to the development of completely new SQL patterns, there were further contributions in this respect, precisely the following:

1 SQL patterns design strategy: The research employed pattern concepts and other related research to structure SQL knowledge.
2 SQL patterns organization and presentation model: the collection of SQL patterns use a method based on the concept of checklist and component-level design, adapted from the field of software development.
3 SQL patterns evaluation: the research contributed by determining how effective SQL patterns compared to the normal teaching of SQL. The

interaction with students in their approach to SQL patterns has yielded some knowledge that has not been documented until now such as problem solving strategy and the type of errors that students attempts during solving the task.

Some of these contributions have resulted in publications in peer-reviewed conference proceedings, namely [18-22].

1.5 Dissertation Structure

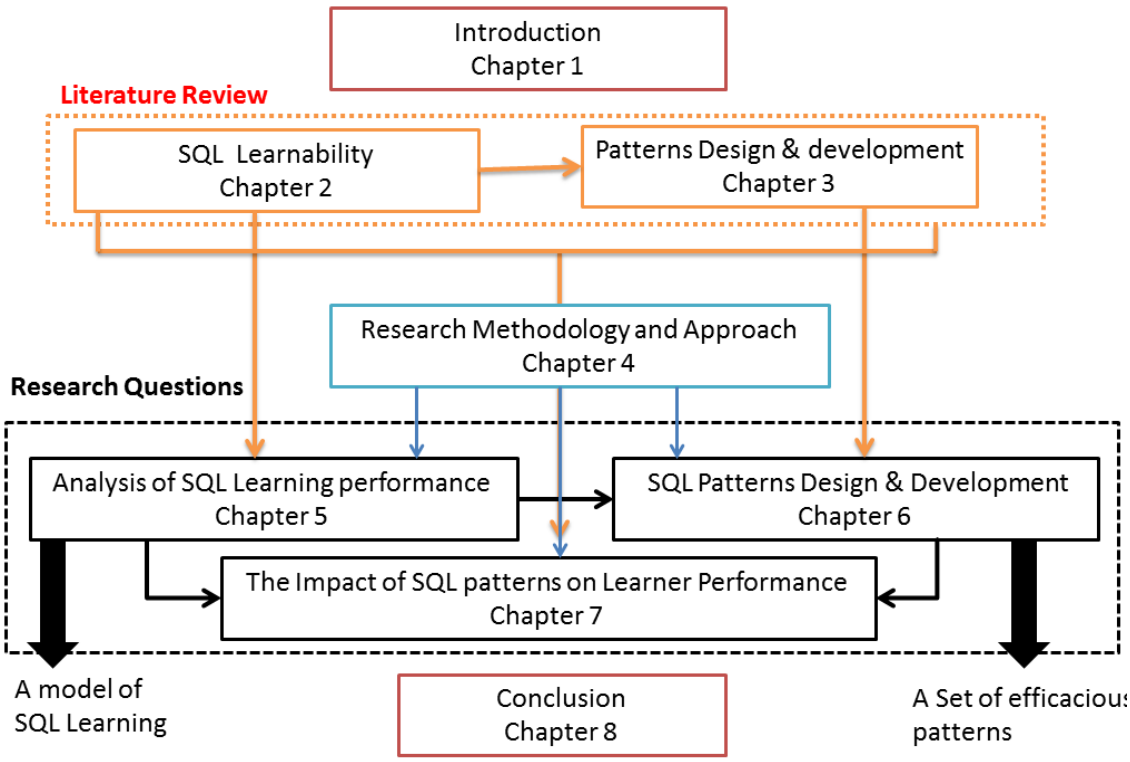


Figure 1.2: Dissertation Structure

Chapter 2 - ***SQL Learnability***: is dedicated to the literature in learning difficulties associated with SQL. This is initiated with a thorough literature review on teaching database courses in general and SQL in specific. Then it is followed by an analysis of how students solve problems. The chapter also covers a review on the empirical studies evaluating the ease-of-use of SQL compared with other query languages and natural languages.

Chapter 3 - ***A Review of the Literature on Patterns' Design, Organization and Usability***: covers patterns history, structure, organization, and usability as available in the literature. The chapter also covers a review on the empirical studies on using patterns in education.

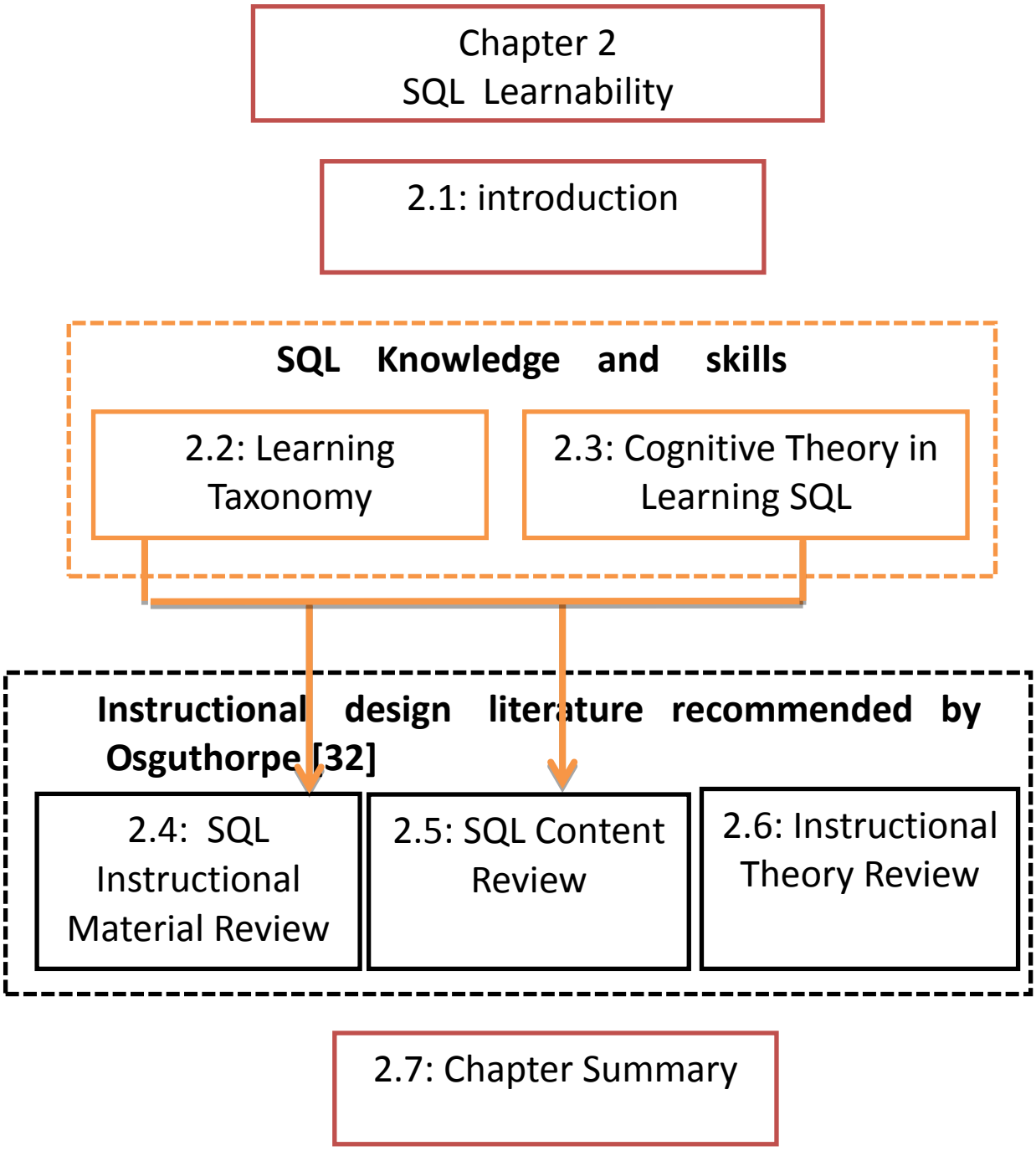
Chapter 4 - ***Research Methodology and Approach***: this chapter includes a description of the research structure and how this research has been conducted. Moreover, the chapter provides insight into the structure of the research and the research framework and the use of combined research approaches (quantitative and qualitative) and different tools.

Chapter 5 – ***Analysis of SQL Learning Performance Objectives:*** it explores the cross-cutting factors that might influence entry-level undergraduate student's performance in learning SQL. It covers the different diagnostic tasks which were used in this research to explore novice's attitude and cognitive factors.

Chapter 6 - ***SQL Patterns Design and Development***: It covers the processes that were involved in SQL patterns identification. It provides a review of different methods on patterns identification and elaborates the used research methods in SQL patterns recognition. This chapter also covers the wisdom behind SQL patterns format and organization approaches.

Chapter 7 – ***The Impact of SQL Patterns on Learner Performance***: this chapter reports the results of the research methods used in this research based on the experiment used to evaluate the effect of using SQL patterns in learning SQL concepts.

Chapter 8 - ***Conclusion***: this final chapter concludes the study by giving an overview of all chapters discussed in this thesis. It also summarizes the main results and lists future research paths.


Chapter 2
SQL Learnability
2.1: introduction
SQL Knowledge and skills
2.2: Learning Taxonomy
2.3: Cognitive Theory in Learning SQL
Instructional design literature recommended by Osguthorpe [32]
2.4: SQL Instructional Material Review
2.5: SQL Content Review
2.6: Instructional Theory Review
2.7: Chapter Summary

Chapter 2: SQL Learnability

This chapter reviews how students learn by discussing the literature on learning taxonomies, and the constructive and cognitive theory of learning. This is done by elaborating three different kinds of literature searches namely: SQL instructional materials review; SQL content review; and instructional theory review.

2.1 Introduction

Database theory and query languages have been taught for a long time and can be considered an established area. Therefore, the basic concepts that a novice should master are well established. There are many good and widely used textbooks on the subject [23-27]. There have been several congresses and publications in which Database teaching is addressed [28] [29, 30] and the annual international workshop TLAD (2003-2012). Most of these publications address the question of which aspects should be covered, or the methodologies that can be used by both the educators and the researcher.

It is only in the last few years that some publications have appeared in which the objectives of Database courses are presented as a set of skills [2]. Computing Curricula by Shackelford *et al*. [31] summarizes a list of skills that Computing professionals should acquire. The EUCIP report (EUCIP 2007) describes professional profiles in Computing as a set of skills. Both include a list of specific skills related to Databases, under the Database Management profile. The list of the skills is a summary considering not only the skills collected from EUCIP [2, 31] but also those that emerge from the author's personal experience and the interpretation of existing literature in Database teaching. Database knowledge and skills are vital to organizations and companies. Some European surveys found

that this is the skill that companies consider to be most lacking in new IT graduate recruits [2].

Many researchers have attempted to identify the factors that influence SQL learning and use. They often conclude with general statements about the ease-of-use. Human factors have been identified as one of the important sources of information to determine the predict success in learning and using SQL [32]. However, according to Yen and Scamell [33]:

> "Few researchers have formally acknowledged the importance of the learning process for a query language either in its language level or in its user interface. As a consequence, another direction for future research is to address the question: how much instruction is required in different languages in order to achieve the same level of competence?" (P.406)

The research reported in this dissertation answers Yen and Scamell [33] call. To do that, a review of the literature related to SQL teaching and learning is presented. Moreover, the chapter evaluates the type of instruction commonly used to deliver SQL contents and skills. In addition, it investigates the type of instruction that is required in SQL in order to achieve a level of competence as required by Yen and Scamell.

Before discussing the research in SQL teaching and learning, it is important to look at the teaching, learning and instruction general terms. According to Mayer [34]:

> "Teaching and learning are inevitably connected processes that involve the fostering of change within the learner" (p.8).

Mayer argues that all learning involves connecting new information to previous knowledge; therefore, it is also important to help learners develop knowledge structures that can support the acquisition of this new knowledge. This support

might be achieved with well-designed teaching instruction. *Instruction* can be defined as something that educators design and implement to promote learning [34]. Examples are: lectures, educational games, text-books, or web-based presentations. The systematic design process of instruction is called Instructional Design (ID) [35].

Instructional design starts first by identifying the learners performance problems, as identified from the literature and the research methodology applied As stated by Morrison *et al*. [35]. There are three types of literature searches that can inform an instructional design process according to Osguthorpe [36]:

- An instructional materials review,
- A content review, and
- An instructional theory review.

However, before exploring these three areas, a general review on teaching and learning theory and processes involved in learning SQL is conducted first. This chapter continues by reviewing how students learn by discussing the literature on learning taxonomies (section 2.2) and the constructive and cognitive theory of learning SQL (section 2.3). This is followed by elaborating the three types of literature searches identified by Osguthorpe [36]; namely: SQL instructional materials review (section 2.4); SQL content review (section 2.5); and instructional theory review (section 2.6). The chapter conclude a summary (section 2.7).

2.2 How Students Learn: A Learning Taxonomy

Learning taxonomies presents a model of a set of levels of cognitive engagement with material being learned. These taxonomies are used in the course design and assessment to ensure that teaching and assessment strike the right balance between the knowledge (course content) and skills (such as syntheses and evaluation).

Bloom [37] proposed a taxonomy which classified forms of learning. He identified six levels of learning, and argued that upper levels should not be attempted before lower levels had been mastered. His taxonomy is shown in Figure 2.1.a.

Anderson *et al.*, [38] proposed an updated version of Bloom's taxonomy to correspond with the way learning objectives are typically described as cognitive activities, as shown in Figure 2.1.b. They removed the synthesis level, and added a new "creating" level at the top of the pyramid. They also emphasized the activity-based nature of each phase or the cognitive aspect of each stage by changing from nouns to verbs. For example, replacing the term "knowledge" with its related cognitive task "Remembering". Gorman [39] proposed a simplified taxonomy, as shown in Figure 2.1.c, with just four levels.

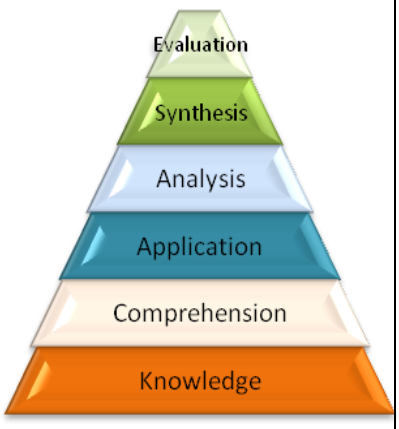


Figure 2.1a: Bloom's Taxonomy

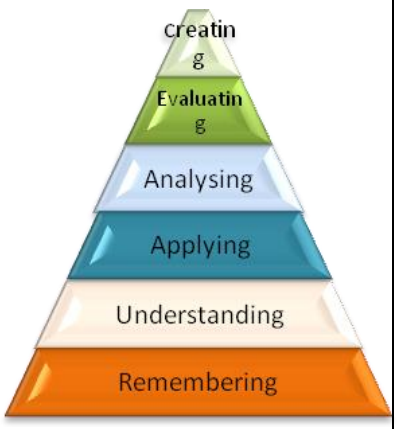


Figure 2.1b: Anderson and Krathwhol's Taxonomy

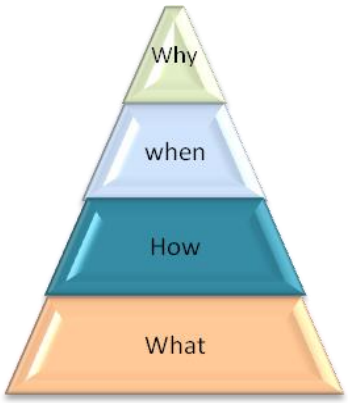


Figure 2.1c: Gorman's Taxonomy

Figure 2.1: Learning Taxonomies

When one examines these three taxonomies, a number of similarities emerge. Gorman's "What" aligns with Bloom's "Knowledge" and Andersen & Krathwhol's "Remembering" levels. It is the declarative knowledge that refers to memory for facts or events.

Gorman's "How" aligns with the "Comprehension" of the other taxonomies. It is the procedural knowledge that is encoded declaratively first, then translated into procedures [40]. Gorman's "When" or Judgment can imply "Application", and "Analysis". "Evaluation of the task judgment includes recognizing that a problem has similar features to one whose solution path is known and knowing when to apply a particular procedure [39].

Gorman's top level is "Why" or Wisdom, which aligns Anderson and Krathwhol's "Creating" levels. According to Gorman [39], "wisdom is the ability to reflect on what someone is doing, and, if required, to come up with a new course of action".

This part provides an abstract source for the other parts of the research when exploring the different kinds of knowledge that SQL learners must have. These different types of cognitive activities or knowledge type are discussed and referred to throughout different parts of this dissertation.

2.2.1 Computer Science Learning Taxonomy

Computer Science educators applied these taxonomies in the same ways as other fields. According to Cutts *et al*.[41]:

> "Learning taxonomies are important for computing education because they give the community a vocabulary to use when discussing student understanding and learning – and curriculum supporting it" (p. 65)

The applicability of various learning taxonomies to Computer Science (CS) has been explored by researchers [42-45]. Lahtinen [45], in particular, investigated whether a subject-specific taxonomy would be of more use to CS instructors than the existing generic ones. He reported that Bloom's cognitive activities were indeed applicable to computing generally.

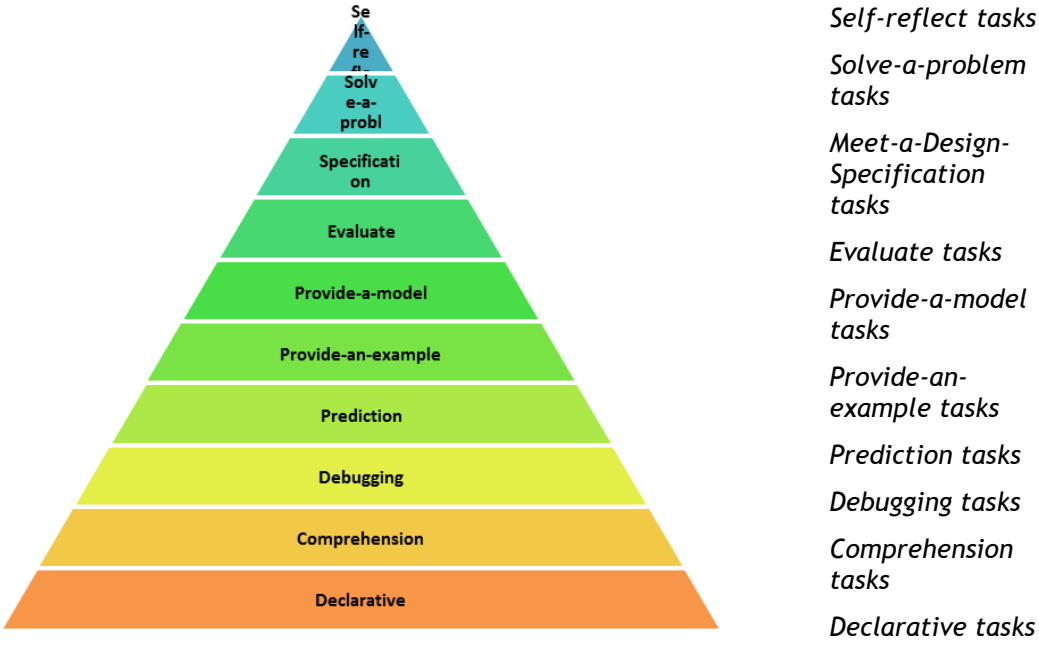


Figure 2.2: Taxonomy of Task Types in Computing (Adapted from [46])

Bower *et al.* [46] proposed a taxonomy of task types in computing, as shown in Figure 2.2 above. It focuses on process-based rather than content-based learning. They argue that it is important that students at an early stage of their education are encouraged to perform tasks that foster higher order thinking.

This led the way towards a more abstract approach, since Computing is essentially a skill-based subject. The three stages of Bloom, which constitute application of principles, are particularly important. Thus, it is possible to argue that several other characteristics apply specifically to CS as a discipline. Learners in CS learn by doing; problem solving is the essence of CS. Therefore, any proposed taxonomy in CS education must highlight problem solving skills at its core. Many researchers recommend incorporating problem solving as a primary learning activity [47-51].

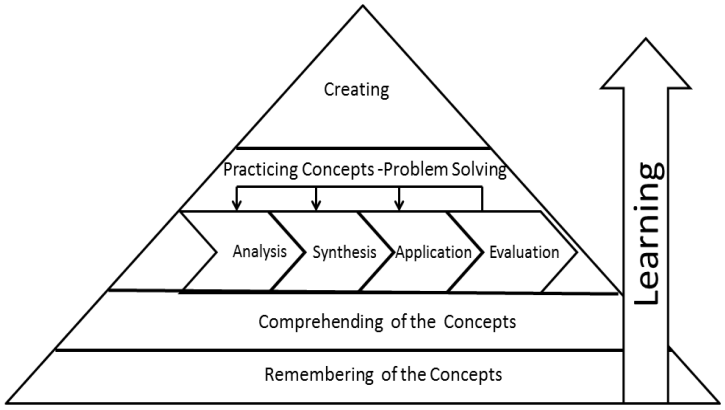


Figure 2.3: CS Learning Taxonomy

Figure 2.3 shows the learning taxonomy that presents CS problem solving process as a core concept of learning. In addition, the three highest Bloom categories (higher thinking processes) are not ordered hierarchically as suggested by Niemierko [52]. Here, it is suggested that analysis, synthesis, application, and evaluation abilities are achieved iteratively during problem solving process within a context specific application of underlining principles. This is supported by Taxonomy of task type in computing (see Figure 2.2) that focuses on process-based. The highest-level "Creating" as proposed by Gorman is the ability to abstract the knowledge and come up with a new course of action such as solving a novel problem or unfamiliar scenario.

2.2.2 Learning Taxonomy in SQL Teaching and Learning

Shneiderman [7] highlighted five tasks that one can apply to provide a query to retrieve information for the database. The following are the five tasks:

1. Learning the syntax and semantics of the function specification. He argues that a typical goal at this stage is to reduce the time of learning.
2. Composition of the syntax required to perform the required function. Composition includes writing a query or formulating a natural language query.
3. Comprehension of function syntax composed by someone else. It is often necessary to read syntax composed by others for learning or other

purposes. Easily composed syntax may not be easy to comprehend. Comprehension is often a component of others.

4. Debugging of syntax or semantics written by others or by the users. The main purpose of the debugging is to correct errors. Shneiderman [7] said that debugging requires comprehension and composition ability but includes other complex cognitive skills. He suggested that query language debugging will require novel debugging strategies. The central problem will be to provide users with feedback to help them determine whether the semantics of the function they invoke correspond with their intentions.
5. Modification of a query written by oneself or others. Existing database queries will often be the basis of new queries. This task requires composition and comprehension skills as well.

Shneiderman's [7] focus was on the human factors aspects of database interactions and how to facilitate the use of query language. However, in this research, the focus is on how to facilitate the learning of SQL. Looking at the five tasks mentioned, it was possible to relate them to learning theory and organize them into a level of learning taxonomy (see Figure 2.4 a. and Figure 2.4 b).

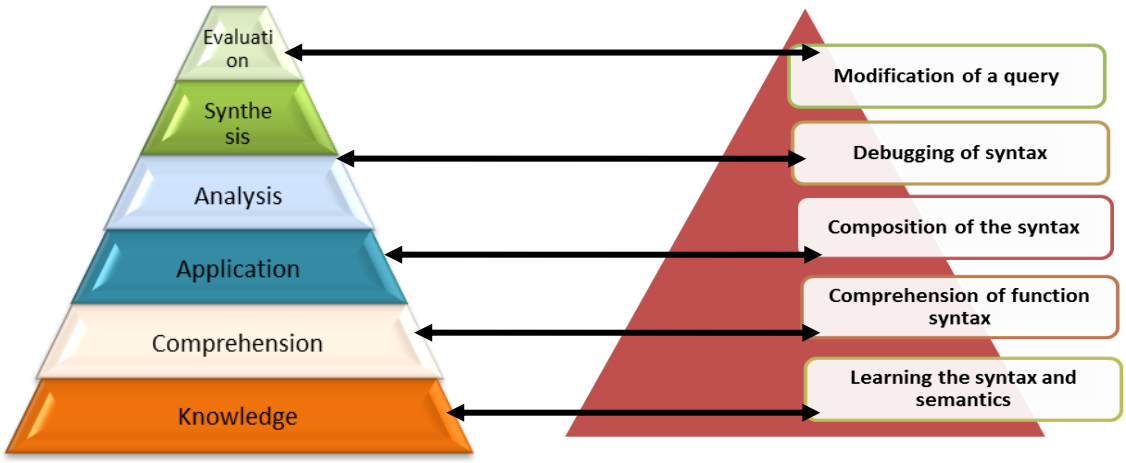


Figure 2.4a: Bloom Taxonomy **Figure 2.4 b: Shneiderman five tasks**
Figure 2.4: Shneiderman Five Tasks and The Related Task in Learning Taxonomy

Renaud et al. [53], on the other hand, examined theories of learning such as those of Bloom and Gorman and noted that the reason students sometimes have

difficulty applying database skills, such as SQL, is that they do not master the required knowledge, or understand the basic concepts correctly. The authors argue that it is thus very important to convey core knowledge first so that students can construct skills (such as SQL) up on top of that core knowledge [53]. They proposed a pedagogical pattern called "Teaching SQL based on Gorman's Taxonomy".

Renaud's [53] approach might be criticized in one aspect, that the suggested waterfall approach in delivering the course material might incorporate possibility of feeling bored by a lengthy period of learning concepts before the application of these concepts. In addition, they need to practice the learnt concepts within the skills-oriented teaching.

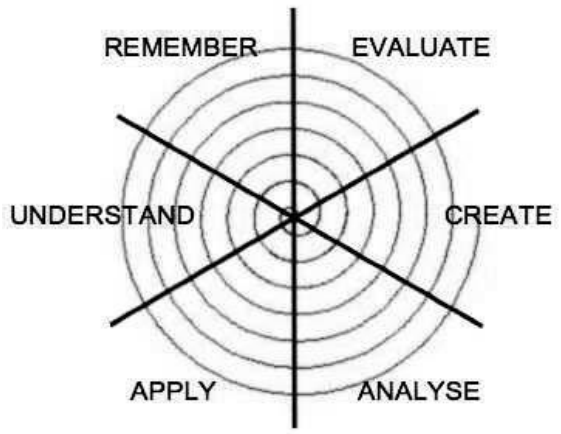


Figure 2.5 a: Bloom Taxonomy as a Spiral Taxonomy [54] (left),

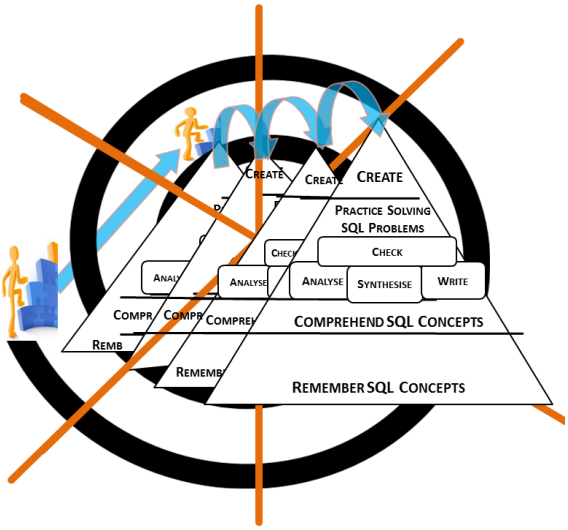


Figure 2.5 b: Learning Through Knowledge Construction (right)

Figure 2.5: Spiral Model for Learning through Construction

Pollock *et al.* [55] describe an approach based on cognitive principles that teaches students information in isolated portions when teaching all concepts together would make it too complex to understand in its entirety. This approach

is often used for material, which is complex, and they argue that one can reduce the complexity and cognitive load by teaching concepts in an isolated fashion with maximum interactivity. This approach appears to be particularly suitable for SQL teaching since students need to have internalised a number of inter-related concepts in order to embark on and master SQL skills.

One can argue that, it is possible to enhance the proposed approach by Renaud *et al.*[53] through changing it to a spiral taxonomy as suggested by Fuller *et al.* [54] arguing that learning is knowledge construction. Hence, learning should not only go directly from bottom to top (what to why), but by seeing each round as thoroughly learning some new pieces of information, which is then used as a basis for the next round topic (see Figure 2.5, right). For example, teaching part of SQL knowledge, then examining the learner‘s understanding of the taught concepts, the student learns how to apply it for different concepts. Later on, learner’s knowledge can be evaluated through involvement in a bigger scenario where many concepts need to be applied.

This research aims to provide a comprehensive emphasis on the practical part of learning taxonomy that is done through problem solving. From the personal observation, students consume a lot of unnecessary time and effort solving each query problem depending on its complexity or occasionally not solving it at all. Thus, it is crucial to understand why students are consuming unnecessarily more time and effort, and why they sometimes give up or end up solving it incorrectly. Are there any deficiencies in transferring the knowledge or skills in one of these levels in the taxonomy, or is it a lack in the problem solving strategy knowledge? These questions might be answered by relating them to cognitive psychology literature. The next section discusses the cognitive aspects in SQL teaching and learning. It presents the different research and models in cognitive activities in solving SQL queries.

2.3 How Students Learn: A Cognitive Theory in Learning SQL

If learning is knowledge construction, it is essential to understand the kind of knowledge that learners constructs. Educational and cognitive psychologists generally distinguish between a number of different types of knowledge, including facts, concepts, procedures, strategies, and beliefs [38, 56].

This section presents a review on aspects of cognitive science and educational psychology. In addition, it highlights the related research in SQL teaching and learning. Thus, this part could provide a conceptual basis for use in the other parts of the research when discussing the related cognitive research and instructional design of SQL education for novices.

2.3.1 Overview of Cognitive Model of Instruction

Significant learning happens when learners engage in correct cognitive processing during learning such as mentally organizing relevant information into a coherent structure, and integrating representation with each other and with prior knowledge retrieved [34]. The focus is on the cognitive aspects in learning SQL, as discussed next.

2.3.2 Cognitive Models in Learning SQL

Some studies provide a cognitive perspective on how the data model and query language influences learners' query performance. Reisner [57] proposed a process where a user will generate a set of lexical items and also generate a query template, followed by the merging of the lexical items with the template to generate the final query (see Figure 2.6).

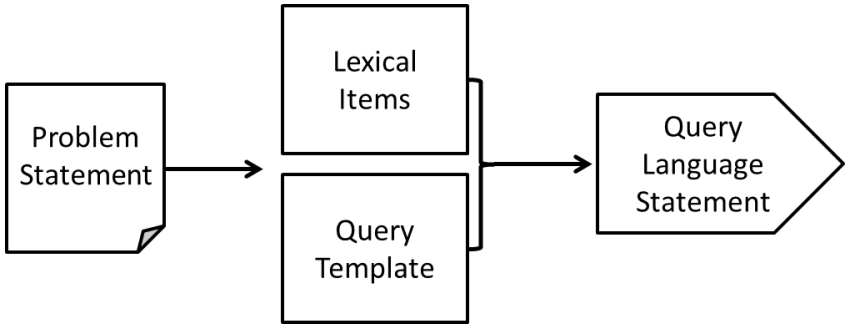


Figure 2.6: Query Writing Model Adapted Reisner [57]

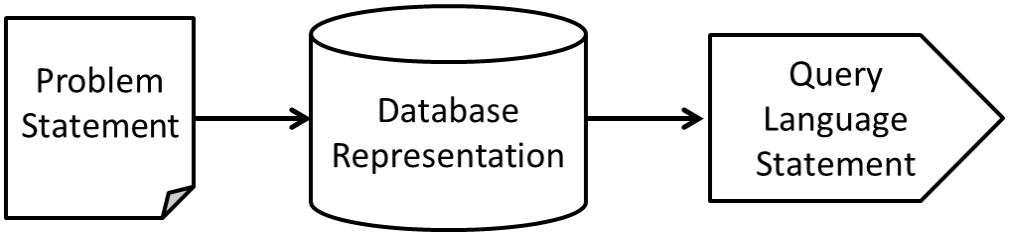


Figure 2.7: Query Writing Model Adapted from Mannino's [58]

Mannino [58] proposes a two-step model: from problem statement to database representation, and from the database representation into a database query language statement, as illustrated in Figure 2.7.

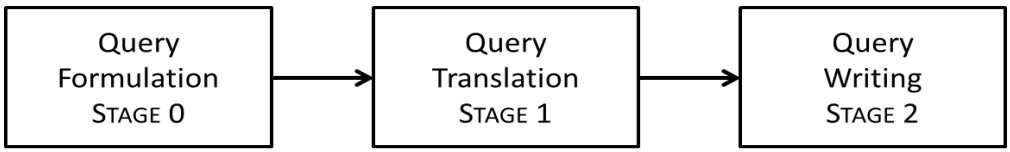


Figure 2.8: Three-Stage Cognitive Model Adapted from Ogden [59]

The model in Figure 2.8 presents an alternative three-stage cognitive model of database query proposed by Ogden [59]:

- Query formulation (stage 0): decide what data they need to solve the problem. One example is: "I need to know the average salary of employees who work in the sales department." This stage relies on knowledge of the application domain.
- Query translation (stage 1): use the output from stage 0 as input, and decide what elements of the data model are relevant, and what the necessary operations are. One example of the output of this stage is: "The employee relation is needed, the column salary is to be selected, and the average to be calculated and a restriction of working in the sales department must be specified on column department. The output of this stage usually retained mentally by experts but written down by novices.
- Query writing (stage 2): write the query in SQL. For the example in the previous stage, to translate into SQL, would be: "select AVG (salary) from employee where…" This stage is heavily dependent on the particular query language syntax and semantics.

Through studying and analyzing these models, it is possible to say that, as individual model, they do not particularly mirror learner cognition and learning stages. They only show the abstract tasks that one can be involved in. Commonly, SQL novices seem to lack a deep understanding of the language construct and the way in which such constructs are used to solve problems [60], which suggests that Mannino's model [58] might more accurately depict an expert's processes than that of a learner. Novices often lack strategic knowledge - i.e. the ability to apply syntactic and semantic knowledge to solve novel problems [61]. Strategic knowledge supports stage 0 and stage 1 of the model in Figure 2.8, and without it, a novice might very well go straight to stage 2, to the detriment of learning and the query quality.

By comparing relevant elements from these models, it is possible to propose a new model that combines elements from Mannino [58] and Ogden [59] models and includes the cognitive science representation of solving problem. In addition, the model highlights the presence of instructional materials.

Cognitive psychologists think of a problem as consisting of an initial state and a goal state, and to solve a problem a person must perform some action (operators) to move from initial state to goal state [62]. Therefore, it is possible to consider the" problem statements" in Mannino model as the initial state and the "query language statements" as the goal state. In addition, the three cognitive processes in Ogden [59] model can be used as the set of operators to move from initial to goal state. Figure 2.9 shows this model to solve SQL problems.

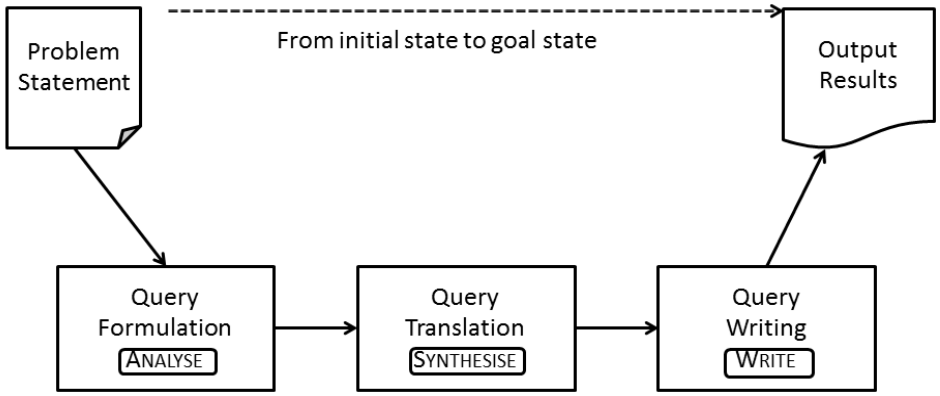


Figure 2.9: SQL Cognitive Model

This leads to investigate other skills, knowledge or tools that need to be available to learners during the process of SQL acquisition. Moreover, the action or process (operators) that happened between exposing students to the problem and presenting the final query is missing. Ogden [59] model presents those as actions or tasks. Integrating these models helps depict how students solve SQL problems.

As a result, the proposed model in Figure 2.9 was enhanced by adding another cognitive process called Evaluation. One could say any problem solver should evaluate the work that has been carried at different stages.

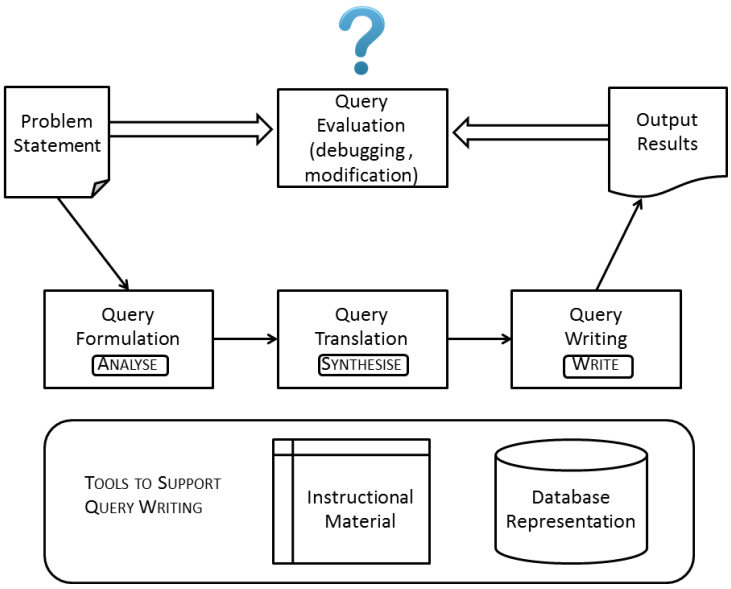


Figure 2.10: The Practice Stage of SQL Learning Taxonomy

Moreover, during problem solving students need tools, knowledge to support solving the problem (such as instructional materials) and database representations, which could be part of any instructional material. This model is illustrated in Figure 2.10, which presents the different stages and tools learners employ during problem solving process that consists of the following:

Problem or task environment: this consists of the "Query problem" statement and the context in which a problem is encountered. Students' understanding of the problem is based on experience of the major variables or facts that are relevant to the problem. Thus, at this stage, learners need to have some pre-existing knowledge about both the context of the given problem and the problem itself. This might be achieved during the lecture or the tutorial by exposing students to some examples or cases that have similar characteristics. As a result, problem interpretation might be straight forward.

It can be concluded that the initial representation of query problems is crucial in helping students in deciding and identifying the initial state of the problem. It influences their decisions about the goals of the problem and the related operations that need to be performed.

Query Formulation stage: this might be called analysis task or operator used stage. At this stage, students try to define the major variables and figure out the required knowledge and skills. This stage is affected by students' skills and knowledge level. Problem solving skills are essential at this stage. Learners are required to divide the problem into small problems, identify the facts and the required knowledge. Moreover, knowledge of underlining database presentation and concepts is crucial at this stage.

Query Translation stage: this can be called task syntheses. In addition to the database knowledge, learners require SQL knowledge. Thus, they might be able to decide about the different required data. For example: tables, columns, relations, keys.

Query writing stage: or the application stage. At this stage, the learner is assumed able to write the related SQL query. Thus, SQL knowledge and syntax is important. This stage leads to the final element in the model, which is output result.

Output result: this presents the output of the query.

Evaluation stage: this is the last destination, where students need to reflect on the results of the previous stage and make a decision about whether the goal of the initial state was achieved or not. If not, then learners need to check the decision taken at one of the above stages. For example, they might need to check their understanding of the problem or their formulation or translation attempts.

Supported materials: such as tools, Database structure, and instructional materials.

Looking at CS problem solving – the practice stage of "CS Learning Taxonomy" (Figure 2.3), it is possible to conclude that to help students to become reasonably experts with query problem solving, it is essential to know in some

details the stages they pass through on their mental process from novice to SQL mastery. To do this, course designers, researchers or educators need to expose both novices and experts to a query problem and observe everything they do.

Gathered data could be analysed in light of some questions, such as: how do participants engage in the problem solving process? Do certain instructional processes help subjects acquire these processes effectively? This is explored in more details in chapter 6.

2.3.3 Summary

The primary message is that learners needs first to have an understanding of the underlying facts and concepts of SQL before one can embark on learning how to write SQL when solving problems. CS Learning Taxonomy (Figure 2.3) suggests that learners need to construct the basic knowledge of SQL first. Then, they build comprehension knowledge. Therefore, they can understand how SQL concepts are applied and interrelate to each other. After that, students should be exposed to problem solving procedures.

Solving problems and producing an effective and efficient solution is the core activity of the CS practitioner, as discussed in section 2.1.1. CS, at its core, involves modelling the real world, representing domains of the most varied nature and complexity, representing knowledge in general and dealing with processes and solutions to problems in such domains. Therefore, any proposed taxonomy should have, at its core, problem solving, to be engaged in after the basic knowledge is delivered and comprehended.

Here, SQL problem solving model is proposed. It illustrates the cognitive tasks learners should follow in terms of solving SQL problems. Learners need to learn how to construct the problem by formulating the scenario. To do that, they should divide the problem into parts and should understand the context of the problem. That requires their previous knowledge and understanding of SQL concepts and skills in solving problems. Then, problem analysis is required.

Learners need to interpret the problem by matching SQL concepts to different parts. Later, they apply the correct SQL syntax to the problem. Only once one understands how to apply this knowledge can one understand when, and in which particular situation, one needs to apply different techniques to problems with specific characteristics. This is what was called SQL evaluation. Only after that, students can expect to understand why this is done in a particular way, and make a contribution to the field.

Throughout problem solving steps, learners need to be supported with an effective instructional material that presents the required knowledge and guide them into a proper model in solving SQL problems.

Therefore, this research is focusing on the instructional methods used to teach SQL. As a result, it is crucial to examine some of the used instructional materials, such as those recommended by Osguthorpe [36]. The next section presents a review on the related research on different SQL instructional materials, such as textbooks and tools.

2.4 SQL Instructional Materials Review

This section discusses and evaluates some of the SQL instructional methods, such as teaching materials, and tools.

The SQL problem solving model that was proposed in section 2.3.2 (Figure 2.10) highlights that instructional materials need to support different stages of problem solving. Therefore, it is vital to gather information concerning the characteristics of existing instructional materials and approaches being used by educators in delivering SQL knowledge and skills. According to Merrill [63]:

> "The greatest impact on learning results from the representation and organization of the knowledge to be learned. Knowledge structure refers to the interrelationships among knowledge components"

Bruner's *Theory of Constructivism* advocates that learners construct new ideas or concepts based upon existing knowledge [64]. Bruner [64] states that a *Theory of Instruction* considers four major facets:

1. Predisposition toward learning.
2. The way in which a body of knowledge can be structured so that it can be most readily grasped by learners.
3. The most effective sequence in which to present material.
4. The nature and pacing of reward and punishment.

Mayer [34], on the other hand, proposes a cognitive model of instruction that consists of six factors in the teaching and learning processes which are:

1. Instructional manipulation
2. Learner characteristics
3. Learning context
4. Learning process
5. Learning outcome
6. Outcome performance

This research focuses on the aspects that relate to the design of instruction. Firstly, concerning the second aspect in Bruner [64] which is the importance of "The way in which a body of knowledge can be structured…" or the instructional manipulation in Mayer model's. Secondly, concerning the third aspect about "The most effective sequence in which to present material" which can be related to the learning process as this will relate to the way for selecting, organizing and integrating the SQL knowledge. However, this research is not focusing on any kind of learning assessments. Therefore, any factors or theories related to assessments are not discussed here.

Renaud *et al*. [53] highlighted two categories of the current problems in teaching database concepts: the first one is related to the teaching methods or approaches that had been used to deliver the knowledge to the learners; the

second was attributed to the tools used by student, to practice their learnt skills. They explored the first category and provided different reasons for such problems:

- Lack in the students' declarative knowledge, because underlying concepts were not taught correctly, key concepts not covered or not fully understood. Then, when the lecturer moves on to subsequent concepts, the student has no chance of progressing up the pyramid to being a skilled database designer and user. Students often do not know that they do not understand something correctly.
- Skills take time to learn; so sometimes expectations are unrealistic if assumptions of a quick and easy mastery are made, i.e. if they think this is just knowledge.
- Students' motivation: studying for exams and not to master concepts.

To be able to design an effective instruction material, one could say that it is crucial to first conduct a review of the current instructional materials that are used to deliver SQL concepts and skills so that their weaknesses and strengths could be highlighted. The following subsections present a review on the current materials that are used to teach SQL knowledge and skills.

2.4.1 SQL Teaching Texts

Many textbooks are used to teach SQL. Some of them are mainly teaching SQL while others are teaching SQL as a part of database textbooks. Conklin and Heinrichs [65] reviewed thirteen database textbooks. The aim of their review was to establish a profile of database texts by examining the content of those suitable for teaching upper-level database courses.

In this particular research, a review on Database textbooks was also conducted. Figure 2.11 shows the list of reviewed database textbooks, which were available in the university library. The aim of this review is based on the CS learning

taxonomy that was discussed in section 2.2 (see Figure 2.3) and the ability to help learners to engage in the tasks highlighted by Shneiderman [66]:

- Focusing on the part that explains SQL syntax and semantics, learning tasks.
- Procedural knowledge or the comprehension tasks.
- Examining the material structure in helping the learner to solve a problem in SQL and writing correct queries. Such teaching problem solving strategy or guiding students in ways to solve problem through examples, worked out examples, or case-based projects (CBP), or tutorials. This is similar to the composition and debugging tasks suggested by Shneiderman [66].
- Examining the material in helping the learner to transfer knowledge and skills through explaining the knowledge of "why" [39] or the engagement in "Modification" tasks proposed by Shneiderman [66].

Book name	Declarative Knowledge Learning	Procedural knowledge Comprehension	Practice skills composition	Debugging	Creating skills Modification
	Description	Description	Description	Description	Description
	Rate :	Rate :	Rate :	Rate :	Rate :

Table 2.1: Form Used in The Review

Each category of knowledge was rated from (0-5) using the following categories: not available, difficult to learn, awkward, simple, informative, effective to learn. The review form is presented in Table 2.1.

What appears (see Figure 2.11) is that most of the available textbooks deliver SQL declarative knowledge such as SQL syntax. However, this does not necessary ensure that learners can apply them correctly. In some of the above texts, there is a lack in the procedural or comprehension knowledge such as comprehension

of function syntax composed for a certain scenario. It is often necessary to read syntax composed by others for learning how a query is executed and how different elements are integrated to achieve a special purpose.

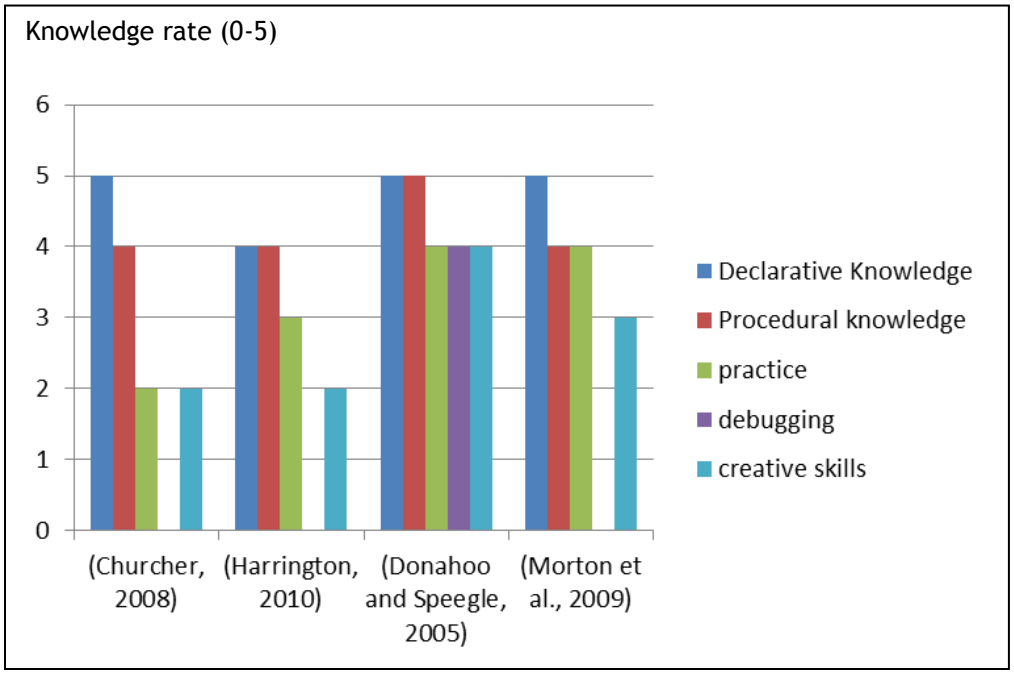


Figure 2.11: The Level of Knowledge Rate (0-5) within the Reviewed Book

Comprehension knowledge can be achieved through engaging students in a task that involves reading query, shown how it works through explanation, and generating the query desired output. For example, explaining in natural language the purpose of the query and the results that might be produced (see Figure 2.12). Furthermore, the "composition" knowledge (see Figure 2.4b) is not available in the reviewed textbook. Textbooks should facilitate problem solving skills through well-designed tutorials. Churchers' book [67] shows in one chapter only (chapter 10: How to approach SQL). This is introduced after all SQL concepts were explained. It is essential to test learners' understanding immediately after each concept.

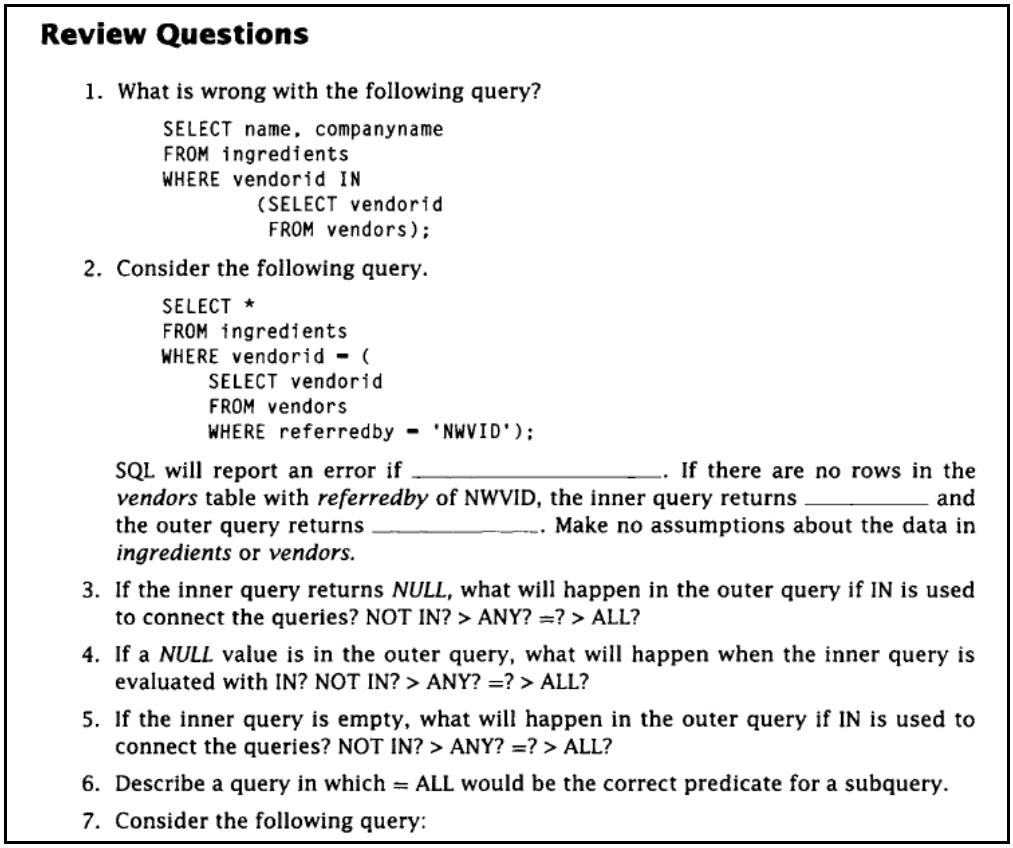
Listing 4-5. *A Nested Query to Find All Entries in Open Tournaments*

```
SELECT e.MemberID
FROM Entry e
WHERE e.TourID IN
      (SELECT t.TourID
      FROM Tournament t
      WHERE t.TourType = 'Open')
```

You can understand a nested query by reading it from the "inside out." The inside SELECT statement retrieves the set of required tournament IDs from the `Tournament` table, and then the outside SELECT finds us all the entries from the `Entry` table for tournaments IN that set. To work correctly with the IN keyword, the nested part of the query must return a list of single values.

Figure 2.12: Example of Comprehension Knowledge

Engaging learners in debugging tasks [66] which involves debugging of syntax or semantics written for a certain context is important. The main purpose of the debugging is to correct errors. According to Shneiderman [7], debugging requires comprehension and composition ability (see Figure 2.4). Only one of the reviewed textbooks, Donahoos' book [68] presents example of debugging task (see Figure 2.13).



Review Questions

1. What is wrong with the following query?

```
SELECT name, companyname
FROM ingredients
WHERE vendorid IN
        (SELECT vendorid
         FROM vendors);
```

2. Consider the following query.

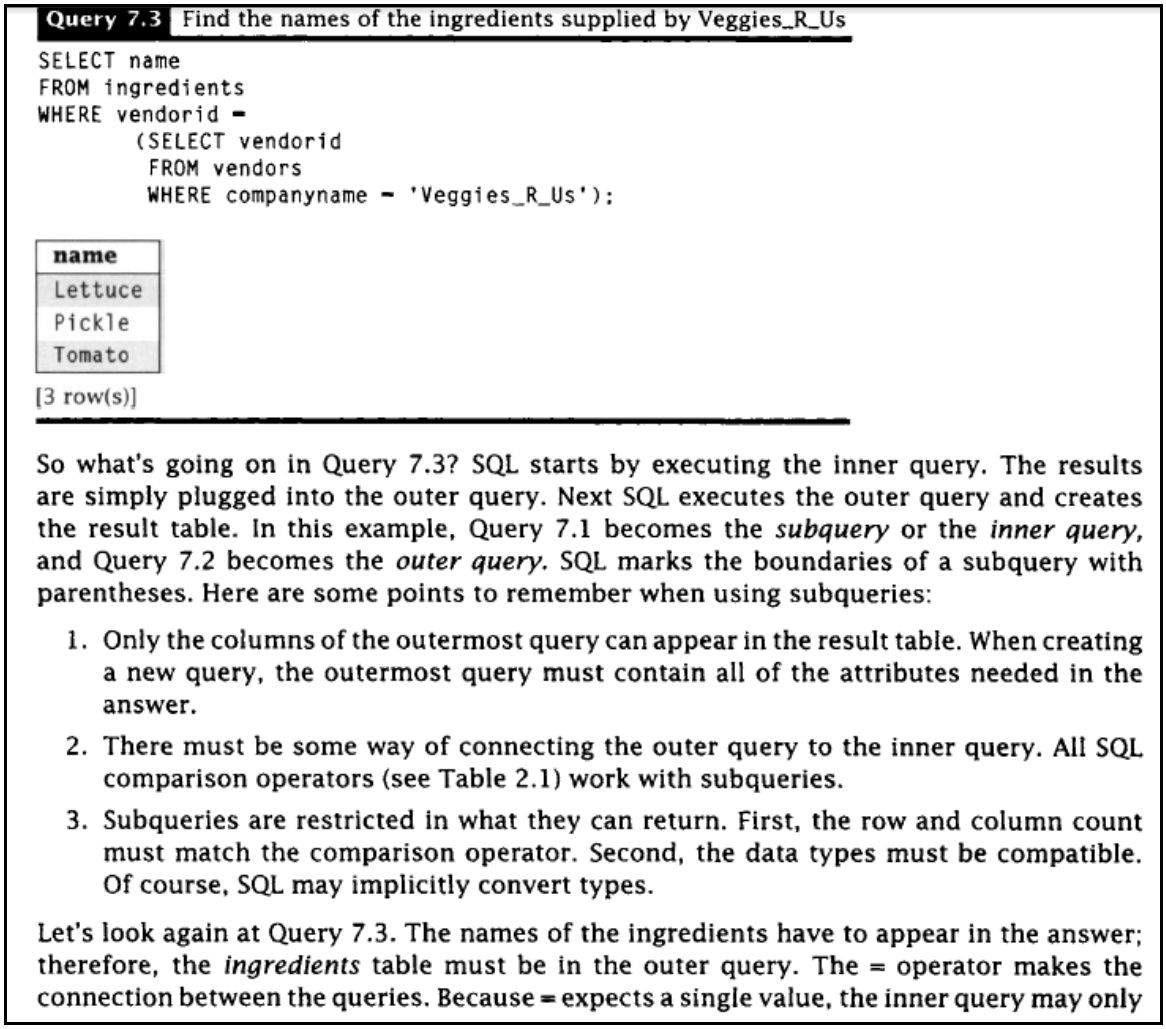
```
SELECT *
FROM ingredients
WHERE vendorid = (
    SELECT vendorid
    FROM vendors
    WHERE referredby = 'NWVID');
```

SQL will report an error if _____. If there are no rows in the *vendors* table with *referredby* of NWVID, the inner query returns _____ and the outer query returns _____. Make no assumptions about the data in *ingredients* or *vendors.*

3. If the inner query returns *NULL*, what will happen in the outer query if IN is used to connect the queries? NOT IN? > ANY? =? > ALL?
4. If a *NULL* value is in the outer query, what will happen when the inner query is evaluated with IN? NOT IN? > ANY? =? > ALL?
5. If the inner query is empty, what will happen in the outer query if IN is used to connect the queries? NOT IN? > ANY? =? > ALL?
6. Describe a query in which = ALL would be the correct predicate for a subquery.
7. Consider the following query:

Figure 2.13:Example of Debugging Task [68]

Few of the textbooks that were reviewed emphasize on facilitating high order skills such as analysis and creating [38]. This can be achieved in many ways. For example, engaging learners in "Modification" tasks [66] which involve modification of a query written by others to fit someone's required need. Furthermore, textbooks need to explain the knowledge of "why". For example, providing a knowledge that explains why this query, or this function and not others. Some of the reviewed texts attempted giving explanations about the wisdom behind the applied queries in a certain context in one chapter [67] (Chapter 9: Efficiency consideration) which might, in some cases, omit such knowledge. Other textbooks provide such knowledge within the same context; i.e. after each concept has been introduced and illustrated with examples (see Figure 2.14).



Query 7.3 Find the names of the ingredients supplied by Veggies_R_Us

```
SELECT name
FROM ingredients
WHERE vendorid =
        (SELECT vendorid
         FROM vendors
         WHERE companyname = 'Veggies_R_Us');
```

name
Lettuce
Pickle
Tomato

[3 row(s)]

So what's going on in Query 7.3? SQL starts by executing the inner query. The results are simply plugged into the outer query. Next SQL executes the outer query and creates the result table. In this example, Query 7.1 becomes the *subquery* or the *inner query,* and Query 7.2 becomes the *outer query.* SQL marks the boundaries of a subquery with parentheses. Here are some points to remember when using subqueries:

1. Only the columns of the outermost query can appear in the result table. When creating a new query, the outermost query must contain all of the attributes needed in the answer.
2. There must be some way of connecting the outer query to the inner query. All SQL comparison operators (see Table 2.1) work with subqueries.
3. Subqueries are restricted in what they can return. First, the row and column count must match the comparison operator. Second, the data types must be compatible. Of course, SQL may implicitly convert types.

Let's look again at Query 7.3. The names of the ingredients have to appear in the answer; therefore, the *ingredients* table must be in the outer query. The = operator makes the connection between the queries. Because = expects a single value, the inner query may only

Figure 2.14: Example of "why" Knowledge [68]

In summary, one could say that knowledge at the level of the text base, however, does not necessarily ensure that the learner understands the intended concepts at a deeper level. McNamara *et al.* [69] argue that the knowledge demand in the scientific text required more understanding than just the ability to reproduce the text itself. One could argue that a strong inference linking the text with the reader's knowledge must exist. This can be called the situation model which might be related to situated learning [70, 71].

Studying the available information or data in the textbooks (see Figure 2.11), it is possible to say that these books in themselves are not sufficient to transfer the knowledge. In addition, there is not enough support to transfer problem solving skills. Most of these texts do not offer tutorials that shows step-by-step

approaching query. They focus on delivering the declarative knowledge of SQL but not on how to apply it. Therefore, these materials cannot help in developing expertise among novices by themselves. The next section examines the tools used to teach or train students SQL.

2.4.2 SQL Teaching Tools

There is various software packages available that were developed specifically for supporting novices learning in CS and few are focused on practicing SQL query skills.

According to Brusilovsky *et al*. [72], SQL tools can be roughly classified into two categories: tools that support students learning of basic SQL concepts and tools that support learning-by-doing.

This section presents a cross-disciplinary review of these tools. The systems or tools evaluation was conducted from learning taxonomy discussed in section 2.2 (Figure 2.3), and from practice perspective (Figure 2.10). The following are the list of some of the tools that have been used for students learning and assessment on SQL skills:

- winRDBI [73]
- *eSQL* [10]
- *SQL-Tutor* [3]
- AsseSQL[74, 75]
- *SQLator* [12],
- Automated tutor for a database skills training environment [76]
- SQLify [77]
- SQL Exploratorium [72]

In 2006, Raadt *et al*. conducted a review for some of the above tools. The main purpose of their review was to evaluate the tools that were used in both SQL

teaching and assessment. Therefore, the review conducted by de Raadt *et al*. [77] focused on the following tools: *SQLator [12]* and *AsseSQL [74, 75]*.

This research, on the other hand, focuses on SQL learning rather than SQL assessments. The main elements that need to be highlighted are:

- Materials support learning SQL knowledge: the rate of presenting declarative knowledge.
- Materials or examples show how query is applied. The procedural knowledge rate is measured.
- Tutorials that guide students toward solving problems: the rate of presenting "practice".
- Other source of information such as feedbacks and guides that aim to help students to build wisdom or creativity knowledge. For example, providing learners with an explanation about the error they get while solving problems and guide them to solve it.

This particular review focused on the following tools: *SQL-Tutor* [3], *eSQL* [10], *Automated tutor for a database skills training environment [76], AsseSQL* [74, 75], *SQLator* [12], *SQLify [77]* and *SQL Exploratorium [72]* as presented in Figure 2.15. Other tools are excluded because they are not relevant to this research review purpose.

Each tool is reviewed by the researcher in terms of the knowledge delivered by the system based on CS learning taxonomy (Figure 2.3). The summary of the review is presented in Table 2.2 below.

	Tool name	Declarative Knowledge "Learning" Rate	Procedural knowledge "comprehension" Rate	Practice skills "composition" Rate	Building skills debugging Rate	Modification Rate
1	eSQL	4	5	0	0	None
2	SQL-Tutor	2	3	4	3	
3	AsseSQL	4	5	5	5	
4	SQLator	2	1	5	4	
5	Automated tutor	0	5	5	4	
6	SQLify	0	0	4	4	
7	SQL Exploratorium	0	0	4	5	

Table 2.2: Tools Review Rating (1-5)

Note: 1 is poor knowledge, 5 is effective knowledge), zero values means are not available.

The tools were rated using Likert Scale from poor knowledge (1) to effective knowledge (5). Each tool was examined in the four type of knowledge (Figure 2.3): declarative (SQL syntax and semantic knowledge), procedural (comprehension), practice (composition, debugging) and creating. In addition, the practice knowledge is examined based on the problem solving model (Figure 2.10). For example, to what extent does the tool support problem solving stages? The results of the tool analysis are reported in Figure 2.15.

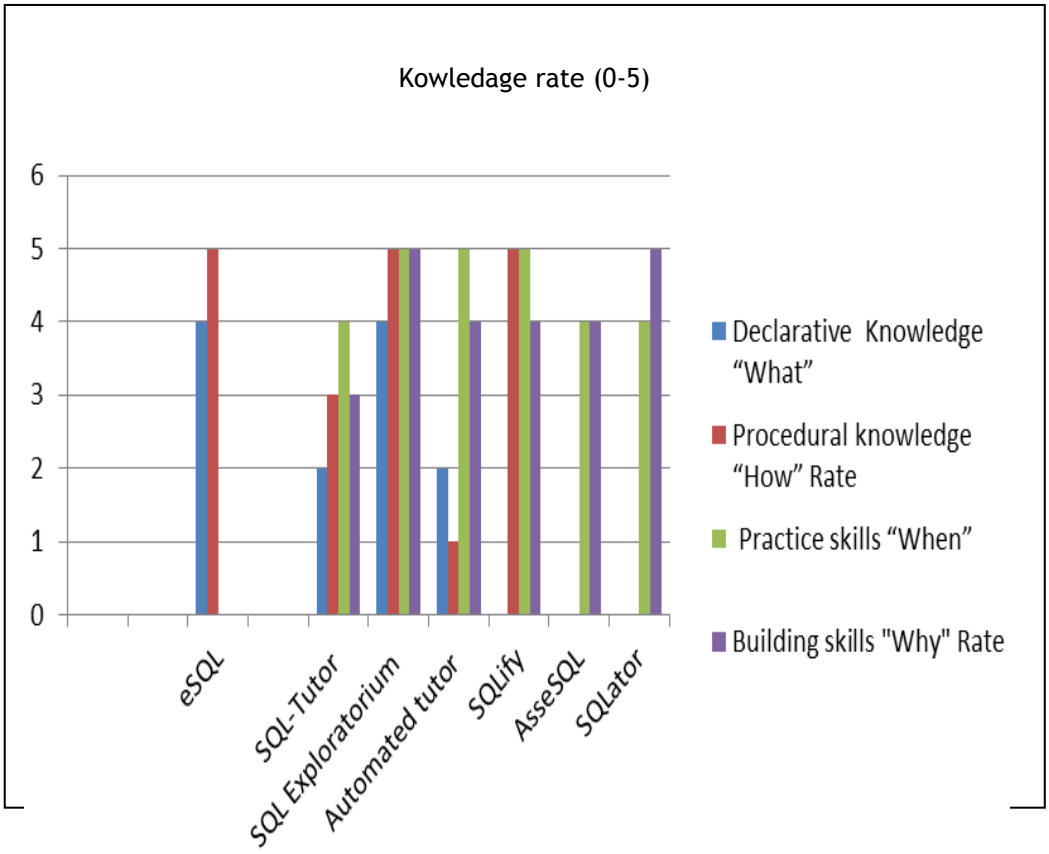


Figure 2.15: Tools review Result

The following is a detailed summer of the review:

The tool	Features	Limitation
eSQL [10]	1-provide procedural knowledge (How) 2-Aid students in understanding SQL queries 3-A Help Mode feature is developed	1-eSQL database tables were designed in assumption that never exceeds thirty or forty rows. 2-eSQL executes a limited number of SQL statements: CREATE DROP, ALTER, DELETE, INSERT, and SELECT.
SQL-Tutor [3]	1-Provides declarative knowledge. 2-Focuses on "Practice" skills through problem solving and meta-learning 3-Supports building skills "why" or meta-learning by supporting self-explanation on the basis of error messages and correct solutions.	1-SQL-Tutor visualizes the database schema only and does not visualize the way a query is executed as eSQL does. Therefore, it does not provide learner with the procedural knowledge. 2-The system is limited to SELECT statement of SQL and other SQL concepts are not covered.

AsseSQL [75, 76]	The tool provides students with a selection of SQL problems and model answers. Thus, procedural knowledge can be achieved through the engagement with the tool. it motivates students to practice using a mock test.	Provides feedback, but this is limited to the correctness of the solution provided by the student. No comments or suggestions for improvement are provided. The tool focus is on the assessments rather than learning.
SQLator [12]	1-Provides "practice" skills through a number of sample databases. 2- Each database defines a business scenario and contains hundreds of English statements describing the query requirements. 3-Focuses on building skills through providing intelligent feedback	The tool does not provide the declarative or procedural knowledge
Automated tutor for a database skills training environment [77]	1-Focus is on training rather than knowledge acquisition. Thus, it provides students with "Practice" skills. 2-Supports building skills "why" through feedback and guidance elements.	1-Focuses on training and development of skills rather than on knowledge. 2- Its execution is limited to SQL SELECT statement.
SQLify [78]	1-Provides a procedural knowledge through the visualization of query processing 2-Provides students with lists of questions to solve; hence, it supports "practice" or problem solving skills. 3- supports building skills through providing feedback to students in an automated and semi-automated fashion. 4-Employs peer-review to enhance learning outcomes for students.	Previous knowledge about SQL facts such as syntax is required before using the tool. It does not help students to learn about SQL. It lacks SQL declarative knowledge.

7- SQL Exploratorium [73]	1-offering centralized access to all three kinds of learning content: WebEx interactive examples, SQL-Knot problems, and SQL-Lab. 2- provides procedural knowledge from WebEx interactive examples tool and supports "Practice" through the use of SQL-Knot problems tool. **3-**High skills "why" is possible to be achieved from using SQL-Knot tool that generates questions that require a student to write an SQL query for a sample database, evaluates the correctness of the student's answer, and provides the student with feedback.	The tool does not guide students or recommend relevant examples or readings after a failure attempt to solve a problem.

From the above discussion and the results shown in Figure 2.15, it is clear that the tools have different purpose and structure. Some of the tools focus on delivering SQL concepts and learning how the related queries are executed through examples and tutorials. They support students learning through interactive examples, demonstrating the basic concepts of SQL. These examples are often created based on multimedia technology [72].The other type of tools support learning-by-doing: offering students SQL problems and evaluating their solutions [3, 11, 12, 76]. Recently, new tools were developed to support both learning and practicing such as SQL Exploratorium [72]. One can conclude that, in terms of the knowledge provided by the tools, different tools provide different knowledge. However, for a tool that aims to help learning SQL, the tool should provide learners with all kinds of knowledge. Students skills and abilities to solve query need to be delivered to students efficiently. Some of the tools such as AsseSQL and SQLator use heuristic methods to evaluate queries entered by students. This involves executing the submitted query on a test database, and comparing the output with that of the query included in the definition of the problem. It is possible for students to sheat by creating simple queries that produce the desired output for the given database instance, which cannot be generalized to all instances of the database. Thus, learners cannot build abstract knowledge and therefore knowledge transfer is not possible to achieve. Moreover, many of these tools do not guide students to solve questions

through the desired stage (Figure 2.15). Therefore, a problem solving strategy is not supported by the tools and students may solve the question by trial and error. Feedback on students' errors is essential to improve students understanding and ability to solve other questions in similar contexts and then later in different contexts. Thus, students high skills "why" can be achieved. Unfortunately, some of these tools do not provide adequate feedback on students' errors nor guidance in how the question should be solved.

2.4.3 Summary

To reflect on the above SQL instructional materials, it can be confirmed that teaching materials must be considered an important factor affecting students learning SQL concepts and skills. The teaching approach and material should aim to help learners to apply different types of knowledge such as: declarative knowledge and procedural knowledge [78]; conditional knowledge [79]; and the syntactic, conceptual and strategic knowledge [60, 80]. Any instructional material should support students in performing all the task types (Figure 2.2) as suggested by Bower *et al*. [42]. Unfortunately, some of the tools, such as those introduced in [76] and [3], were intended as a practice environment and assumes that students have previously been exposed to the concepts of database management in lectures.

The objective of any instructional materials should support students through all problem solving tasks (from problem statements formulation to query output evaluation) in learning and solving SQL query as was illustrated in SQL problem solving model (Figure 2.10). One could argue that the above reviewed teaching or learning materials are not designed from learner's perspective. One example is how to keep students motivated while learning or using the tool. In addition, they do not consider the related cognitive activities that learners need to perform to achieve the intended learning goals. Furthermore, a search into the nature of the learnt materials need to be considered and to ensure that a clear

understanding of the SQL learning aspects from different dimensions. Chapter 5 discussed these dimensions.

It is possible to say that proposing new instructional material should aim to help learners in developing knowledge, skills, and expertise so that learning performance can be enhanced. Hence, the newly designed materials need to consider the highlighted issues discussed in this section and the cognitive aspects that were discussed in section 2.2. The following aspects emerge from the discussion as being important:

1- Emphasize both teaching process and content.
2- Focus on developing problem solving skills.
3- Ensuring the ability of transfer so it might avoid trial and error strategy in writing SQL queries.
4- Facilitate searching for the required knowledge by organizing the material using scaffolding techniques.

These four features can be achieved through a well-designed instructional design that takes into consideration the above findings, the characteristics of learners and the nature of the SQL language. To do that, it is essential first to look at other research that examines the usability or the learnability of SQL. Many studies look at the human and cognitive factors that affect using or learning SQL. The next section gives an overview of review studies that involved the consideration of the causes that affect learning and using SQL.

2.5 SQL Content Review

In this section, both practical and theoretical studies on the teaching and using of SQL and similar subjects are reviewed. This involves the examination of the factors that affect the use of SQL.

Many researchers attempted to identify the factors that affect language learning and use, often called human factors. Welty and Stemple [81] pinpointed two

reasons for focusing on human factors in the field of computer language acquisition:

1- Determining whether a language is learnable, by arguing that failure of this test may predict a language's demise.
2- Eliminating minor difficulties in a language.

Although a number of comparative studies of SQL language and other query language have been conducted [5, 7, 33, 81], the most popular methods of studying SQL has been to teach the language and then examine the participants' ability to use it effectively. Query Language Success based on [5, 6, 33, 57, 82] are identified as:

- Easy to learn by the intended population,
- Easy to comprehend, and
- Satisfaction (user friendly).

Studies used either online tool or paper and pencil [5] or both [33]. In general, most of the studies used some or all of the following tasks: query comprehension, query writing, memorization, and problem solving. Different tests were employed to carry out those tasks. The most common tests were midterms or final exams given after the examined query language had been taught or used for a period of time. Some used quizzes or mini exercises during the teaching to cover the knowledge that were taught up to that time and provide formative feedback.

Thomas and Gould [6] determined how tasks affected the learning of query languages. The effect of the method of teaching query language was studied by Schlager *et.al* [13]. The impact of query language features on learning and using the language is discussed in several publications [4, 5, 9, 33, 81, 83, 84]. These studies can be classified either as

- Comparisons between SQL and other query language;

- Comparisons between SQL and natural language;
- Augmented use of one or more of query language and database base structure (ER, network, relational, or hierarchical) model;
- Studies of the usability of certain features within a language type.

The following sections review examples of these studies in details. The review is conducted in terms of participants, teaching or training procedures, how data is evaluated and the reported results in term of SQL learnability or usability.

2.5.1 Comparison between SQL and other Query Language

There are some human factors' studies that have directly compared the performance of users using SQL and other query language such as QBE, SQUARE, KOL or a procedural language such as TABLET. Many researchers, such as Welty [84], summarizes the issues and the experiments that have been involved in SQL. In this section, an evaluation of the SQL in terms of usability compared to other query, results of human factors studies are surveyed (Table 2.3).

	Research	Query language	Results
1	Reisner et. al [5]	SQL vs. SQUARE	For novices SQL is better than SQUARE. Users with programming experience performed better than with less programming.
2	Boyle, Bury, & Evey [85]	SQL vs. QBE	No performance measures SQL required less time to learn
3	Yen & Scamell [33]	SQL vs. QBE	In paper-pencil, QBE user performed better
4	Hvorecky, Drlik, & Munk [86]	SQL vs. QBE	The more difficult the task is, the more time is required for solving. Time required for OBQ is less than for SQL. User satisfaction decreases with task complexity.
5	Welty & Stemple [81]	SQL vs. TABLET	For complex query, TABLET user performed better.

Table 2.3: List of Literature Resources for Comparisons Between SQL and Other Query Language

The following is a detailed summary of the above studies in terms of their participants, teaching methods, evaluation and results.

1. Reisner *et al*.

The aim of this study was to examine the ease-of-use of SQL compared to SQUARE.

Subjects: The participants were 64 students.

Teaching/Training: The students were taught over a period of two weeks (12 hours-14 hours).

Evaluation: Three stage of evaluation were used in this study: during the teaching, after the teaching and one week after the final assessment.

Results: Previous experience with programming language has impacted the participants' performance.

Results: The results reported in this study showed that, for participants with no programming knowledge, SQUARE was easier than SQL; and for participants with programming experience, SQL was learnt fast and more complete. This means that students with such knowledge and experience might perform better in learning SQL. In addition, the results conveyed that both type of participants could not use either language with reasonable proficiency after 12-14 academic hours of teaching. Thus, it is possible to say that SQL takes time to be mastered.

2. Boyle *et al*.

The aim of this study was to gain information about learning, problem solving, and subjective reactions to QBE and SQL.

Subjects : The subjects were divided into two studies: study 1 was conducted using twelve upper-division students and study 2 was conducted using eight experienced secretaries.

Teaching/Training: Specific training material was designed for these studies in printed form. There were a total of five lessons per language, and each lesson was completed in a single session.

Evaluation: For each study (1 and 2), half of the test participants learned QBE first and then SQL; the other half learned the languages in reverse order. A post-test Questionnaire/Interview was employed to collect participant's performance and feedback about the task and query languages used.

Results: The study results showed that query complexity is an important variable in evaluating user's performance. Both languages were learned equally well for the lessons on simple queries and queries involving comparisons and logic.

- SQL was learned faster and was more often preferred than QBE.
- Problem solution times for simple queries, comparisons, and logical operations: there were isolated cases where statistically significant difference between QBE and SQL were found.

3. Yen & Scamell:

The aim of this study was to evaluate and compare user performance and user satisfaction with QBE and SQL in a controlled research laboratory experiment where comparable participants not only interacted with the same DBMS in an on-line environment but also learned and utilized both query languages in a different order.

Subjects: 65 students participated in this study.

Teaching / Training : The experimental environment was a usual classroom type situation. In order to facilitate subject's learning of the language in a short period of time, each language manual contained only those features required to perform selection and extraction operations.

Evaluation: The experimental tasks consisted of two types of tests for each query language. The first was a paper and pencil test. Subjects were given time

to study. Then, they were given the test, which contained English statements to be translated into queries in the particular query language that they had just learned. For each query, the subjects were asked to record the time they started and the time they finished. Further, they were not permitted to consult notes, manuals, or each other during the test. One week after the on-line training, the second test was given as an on-line test; the subjects were tested on a one-on-one basis with the researchers. The questions were presented as English statements, and the subjects were required to enter an appropriate query and run it. If the query contained one or more errors, then a new version of the query was created and run again until the subject was satisfied with the results.

Results: The experiment's results indicated that query complexity is an important variable in evaluating user performance and in developing user training programs, and emphasize the importance of the actual use of a query language in user training. Both query language type and the order of exposure to different query languages can lead to a difference in user performance and user satisfaction. Thus, it is possible to say that task complexity and students' previous experience might impact students' performance in SQL.

4. Hvorecky *et al*:

The aim of this study was to examine the ease-of-use of SQL compared to QBE.

Subjects: 59 students in QBE group, and 57 students in SQL group.

Evaluation: The experiment was completed during 24 hours of lectures in one semester. It has the form of a pedagogical experiment with pre-test, post-test and two groups – experimental (tested) group and control group.

Results: QBE graphic interface allows faster and more comfortable writing of low and medium difficulty tasks compared to the SQL text-based environment. This indicates the impact of the task complexity in students' performance.

5. Welty & Stemple

The aim of this study was to examine the ease-of-use of SQL compared to TABLET.

Subjects :72 undergraduate students, mostly business majors, were divided into two groups: one group (35) learning SQL, the other (37) learning TABLET.

Teaching / Training : Both languages were taught using instructions read outside class; each contained 12 lessons. These instructions, one presenting SQL and the other presenting TABLET, contained identical examples and problems presented in the same sequence.

Evaluation: Two final exams (an open book exam) were given immediately after the course. A retention test was given three weeks after the final. This test was of the same format to the final.

Results: The following were the finding forms the experiments subjects. The subjects using the more procedural language wrote difficult queries better than the subjects using the less procedural language. The results of the experiments are also used to compare corresponding constructs in the two languages and to recommend improvements for these constructs.

To summarize the above reviewed study, it is possible to conclude that learning and using SQL is affected by different reasons, namely:

- Users or learner's knowledge and experience background, such as programming languages which is an important factor when learning query language [5].
- In learning about query language, it is easy to learn the basic concepts but more care need to be taken when the complex query is used [33].
- Query complexity is an important variable in evaluating user performance and in developing user training programs [33]. Thus, the level of task complexity is important to take into consideration while learning SQL. s need to design a task that is compatible with students level of knowledge.

The next section investigates the difference between using SQL compared to a restricted natural language in learning and using.

2.5.2 Comparison between SQL and Natural Language

Natural language (NL) systems for querying a database have shown technical feasibility and promise in terms of practical use, as evidenced by a large number of experimental systems [9, 87-89]. Some examples are shown in Table 2.4, which compares a restricted natural language with SQL.

Researcher Reference	Results
Shneiderman [8]	Both languages were equal in valid query
Vassiliou *et al*.[90]	Natural language less verbose
Turner *et al*. [91]	Both language were equal in error rate

Table 2.4: Research Conducted to Compare SQL and Natural Languages

1- Shneiderman

The aim of this study was to compare the use of Structured Query Language (SQL) and English in formulating valid database queries.

Subjects: 22 students participated in this study.

Teaching / Training: Students were enrolled in an undergraduate Cobol programming and information systems course.

Evaluation: Three types of evaluation tests were employed in this study:

- Comprehension questions involving three SEQUEL samples that students were to execute against the given database, and four English queries that had to be translated into SEQUEL.
- Situation Problem (SEQUEL)
- Situation Problem (English)

Results: The number of valid English and valid SQL queries had no significant differences. However, the number of invalid queries for English was significantly more than for SQL. In addition, natural language usage would be extremely

difficult without user knowledge of the application domain. There were no restrictions on the complexity of queries for natural language subjects. Under these circumstances, the SQL subjects might tend to write easy and simple questions to avoid syntax errors.

2- Vassiliou *et al*.

The aim of the study is to compare performance between subjects using SQL and subjects using the prototype natural language system, USL (User Specialty Languages).

Subjects: 61 students were divided into three groups:

1. Group 1 (10 students): USL with application training.
2. Group 2 (34 students): USL with application and language training.
3. Group 3 (17 students): SQL with application and language training.

Teaching / Training : The three groups were trained for two hours in the application domain. Moreover, the second and third groups were trained in their respective languages (SQL or USL) for three and one half hours. Subjects in the first group were given a ten minute introduction to the interaction philosophy of USL.

Evaluation: All groups were given the same paper-and-pencil test consisting of fifteen questions. Students were asked to write the required queries to answer the questions in their assigned language. They were also asked to indicate on a five point scale the extent of their understanding of the question, how certain they were of a solution strategy, and how complex they believed the questions to be.

Results: The reported results showed that there was no difference in subject's performance found on the basis of language type. In addition, the finding of a longer answer time for SQL subjects is consistent with the finding that SQL subjects had an average query length that was substantially larger than the USL

average length. Moreover, the fact that USL subjects did not perform better than SQL subjects might be related to the time required for training to use natural language query systems which are quite demanding in restrictions

3- Turner *et al.*

The aim of this study was to test the performance differences between SQL and USL subjects.

Subjects: 8 paid students participated in this study.

Teaching / Training: Training consisted of a 1.5 hour classroom session covering the application domain (date definitions, codes, structures, organization, key actors, etc.), and two 1.5 hour classroom instruction sessions in the respective language followed by a paper and pencil test. Both treatment groups (i.e. SQL and USL) were then given six 1.5 hour hands-on practice sessions with the system using requests modelled after actual user requests. An additional 1.5 hour classroom session was then given in each language followed by another six 1.5 hour practice sessions.

Evaluation: Two tests were conducted as paper and pencil tests. The second test was constructed with questions that described problem situations in the application domain. Students were then asked to write the related queries that would generate the information needed to answer the question.

Results: There was no significant difference in terms of performance, but the standard deviation for the USL subject scores was almost twice that of the SQL subjects, suggesting more variation in USL subject performance. The results were based on data from only eight subjects, so one must be careful in interpreting the results.

Before interpreting the above reviewed studies, it is important to understand the nature of SQL and the natural language. SQL statements' nature, syntax and content are different than other natural languages that novice express in their

daily life. The rigid SQL syntax compared to the inexact and loose nature of NL, results in many students not being able to successfully write SQL as some of the above studies and others, such as Reisner [4, 57], reported as well.

Although many of the studies conclude that there is no significant difference between NL and SQL, it is possible to argue that NL may not be the best for use in executing a complex query or producing technical data. There are many limitations of natural language [66], such as:

- Users or learners may not be aware of the contents and semantics of the database. Therefore, they attempt to request information that is not available in the database.
- Using natural language without sufficient training allows the ambiguities of English syntax to pollute the query process.
- Many users can be aware of English syntax, but failed to understand the semantic of database.
- The efforts of creating and maintaining a natural language interface might be more than for a concise query language.

It is possible to argue, that computer science students are expecting to join the industry. Therefore, they should be able to process different type of query with level of complexity. Thus, teaching SQL within database course should be preferred among academics. One could think that introducing a restricted NL might be a good option to be introduced in level one or at school (i.e. before joining the university or at the foundation level in university). In other words, NL might be an alternative option for novice with no mathematical background, relational algebra, or computer concepts.

The next section investigates the influence of augmented use of query language and database structure.

2.5.3 Augmented Use of One Query Language and Database Structure

Data-model/query language is one of the factors that the performance of a database user is influenced by [4, 92, 93]. Some examples, as shown in Table 2.5, that Augmented between one query language and database base structure (ER, network, relational, or hierarchical) model. One of these examples is explored in detail, Chan's study [92]. The reason is that Chan's study looked at the cognitive activities that this research focused on, as discussed in section 2.3.

Researcher	Experimental nature	Results
Lochovsky & Tsichritzis [94]	Query witting with ER or relational models	No difference in the number of sematic errors. ER user were faster to complete the task
Chan [93]	Query witting with ER model and KOL or relational model with SQL	Users in ER\KOL perform better(time and query correctness), more confident than in ER\SQL user
Leitheiser & March [95]	Query evaluation & witting with ER or relational models	Query learning and using is easier with relational than ER
Chan [92]	query performance with the relational model and SQL was measured at two query stages: the query translation and query writing stages	SQL query difficulties (which are all based on the query writing stage). Exploratory analysis of query difficulties show surprises. For example, operations generally perceived to be difficult (such as joins, group count and repeated relations) are not difficult at the query translation stage, i.e. the difficulties are not because of the relational model, but because of SQL.

Table 2.5: Summary of the Research Conducted to Augment the Use of Query Language and Database Structure.

Chan's study

The aim of this study was to conduct an experiment that measures query performance at both the query translation stage and the query writing stage which was discussed by Ogden [59]

Subjects: 20 first year undergraduate students participated in this study.

Teaching / Training: Students were trained by an administrator before they took the query test. A training manual was used during the study to provide a brief overview of both relational data model and the query language. To improve learning, feedback on query accuracy was given before proceeding to the next example. The period of training was about one hour.

Evaluation: The task in this study is set at two stages. Query translation is the stage that tests subject's understanding of the data value representation of the relational model. The second stage requires the users to write down the query syntax. At this stage, the researchers test whether participants can specify the query operations with SQL syntax. Both stages cover the same query questions. Each subject performed seven queries for both stages. The queries covered a comprehensive range from the very simple to the very difficult. The seven chosen queries covered the following semantic specifications: single entity, two entities (of different types) connected by a relationship, attribute condition, two instances of the same type, counting of relationships, quantifiers for WHERE, EXIST and not EXIST.

Results: The study result showed that:

- It is possible to understand the relational model for many operations, but it is difficult to express these operations in SQL.
- Confirm the findings in the literature about applying SQL operation difficulties.
- Before using relational database systems, it is recommended that one need more training on the particular difficulties of the query language, and also the operations that are even difficult at the model level (e.g. sub query with not exist).
- Knowing more about the difficulties in expressing operations in SQL allows educators focus on these aspects of SQL that cause problems for users and thus allow a more focused training for SQL users.

Chan's study helped this research to investigate more about the learner's ability to perform tasks in the three cognitive activities that were identified by Ogden [59] as was discussed in section 2.3.2. In addition, it motivates this research to investigate the SQL misconception or the most difficult concepts in SQL. Many studies reported that students experience many problems when learning SQL as was discussed earlier. Some of these problems arise from misconceptions in the student's understanding of the elements of SQL and the relational data model in general. For example, students find that join conditions, grouping and restricting grouping are the hardest concepts to understand [96]. The difference between aggregate and scalar functions is another common source of confusion [3]. In addition, Lu *et al*. [96] carried out a survey on the kinds of SQL statements used by 149 SQL writers from 41 companies. They found an even spread from very simple to very complex queries, with just over a quarter of queries involving 5 or more conditions, and 20% of the queries being classified as complex (involving 5 or more relations, or having more than 6 attributes or more than 5 conditions. Moreover, Lu et al. cite research [57, 97] which shows that the rate of incorrect SQL queries ranges around 75% mark, which is as astonishing as it is unacceptable.

Clearly a failure to develop SQL skills is not merely an academic issue: it has wide-ranging effects and there is a need to find a better way of helping students to really grasp the nuances of SQL. Chapter 5 investigates the factors that might affect SQL learnability from both learner and educators perspectives. It also examines the learners' skills in solving query problems. The next section gives a summary of the reviewed studies.

2.5.4 Review Summary and Discussion

Although SQL is simple and highly structured, students still have difficulties learning it [3]. The purpose of the above human factors studies review is to highlight the factors that might influence SQL learnability.

From the above, it is possible to say that a variety of query languages (SQL, OBQ, SQUARE, etc.) might be used or taught for different purposes. In other words, certain features of a language might be more difficult to teach or use than others. For example, users show better performance during problem solving of complex tasks when using procedural (TABLET) method than when using a declarative (SQL) and learn QBE faster and solve a hard query in less time than when using SQL. Therefore the structure of a language seems to be an influential factor that needs to be considered when selecting the language. This might lead to further discussion, such as: to what extent does the nature of SQL as a declarative language affect students' performance in SQL courses? Is there any possible way to teach SQL to overcome the issue of being structured and declarative? More detailed discussion about this will be given in chapter 5 and 7.

Moreover, the learners' previous knowledge and experience could influence their performance. For example, some of the results of the above studies showed that participant with programming knowledge perform better using SQL. This could cause other investigation of what other skills or knowledge affecting learner's performance in SQL. Since SQL is a formal language that is based on relational algebra, then such knowledge might be important to take into consideration. Furthermore, learners' problem solving skills and their ability to apply different strategy to write SQL can affect their performance as well (Figure 2.10).

In some studies, users perform better with the relational model than with other models like ER and network data model. Some studies suggest that the ER/SQL combination is the most appropriate for a low level of task complexity when users are novices, while ER/OBE is better to be used when solving complex tasks. To summarize the above studies, the following factors emerge that might influence SQL learners.

- The user previous knowledge and experience,
- The level of the task complexity,
- The influence of the query language syntax and semantics,
- The influence of the kind of training or teaching used, and

- The effects of a query language when it is used with different database model such Entity Relation (ER), network, relational, or hierarchical model.

Undoubtedly a failure to develop SQL skills and build a strong mental model is not purely related to the above aspects. Renaud *et al.* [53] highlighted other issues such as the nature of SQL mastery. They argue that SQL is a skill, and this is as true as for any other skill, it takes time to master. What is also true is that a skill may easily look effortless when one observes an expert at work. When a database lecturer is demonstrating SQL queries to students it makes a great deal of sense to them, especially if explained properly. An expert always makes things look easy. However, when the student attempts to write his or her own queries, difficulties arise because it is not as easy as it looks. If the student has not laid down the basic skill set, it is almost impossible to master complex SQL queries. Another factor that might affect SQL learnability is the type of instructional materials that has been used to introduce SQL to the students [53, 72], as discussed in section 2.4.

It is possible to say, to learn SQL effectively, it is essential to use a well-designed instruction that considers the above factors. To design a new instructional material that considers the above discussion, it is important to first look at the general literature of instructional theory then focus on the aspect of presenting SQL knowledge and how to master the related SQL skills. The next section presents the review on the instructional design theory. The purpose of this review is to form a basic knowledge about the related knowledge in instructional design and then refer to in other chapters.

2.6 Instructional Theory Review

The purpose of instructional theory is to provide perspective advice to the course designer. According to Osguthorpe [36], three types of literature should be reviewed in conducting instructional design:

- General principles of instructional design,
- Theory and research related to a particular category of learning, and
- Principles associated with a specific delivering context

The principles associated with this research instructional are explored in chapter 3 and chapter 6. The review of theory and research related to a particular category of learning is following in section 2.6.2. The review of general principles of instruction design is discussed in the next section.

2.6.1 General Principles of Instructional Design

Instruction, as was defined in the introduction section, is something that educators design and implement to promote learning. The design of practice, the organization and presentation of information are the domains of instructional designers [98]. The principles of Instruction Design (ID) were discussed in many research [35, 99-102].

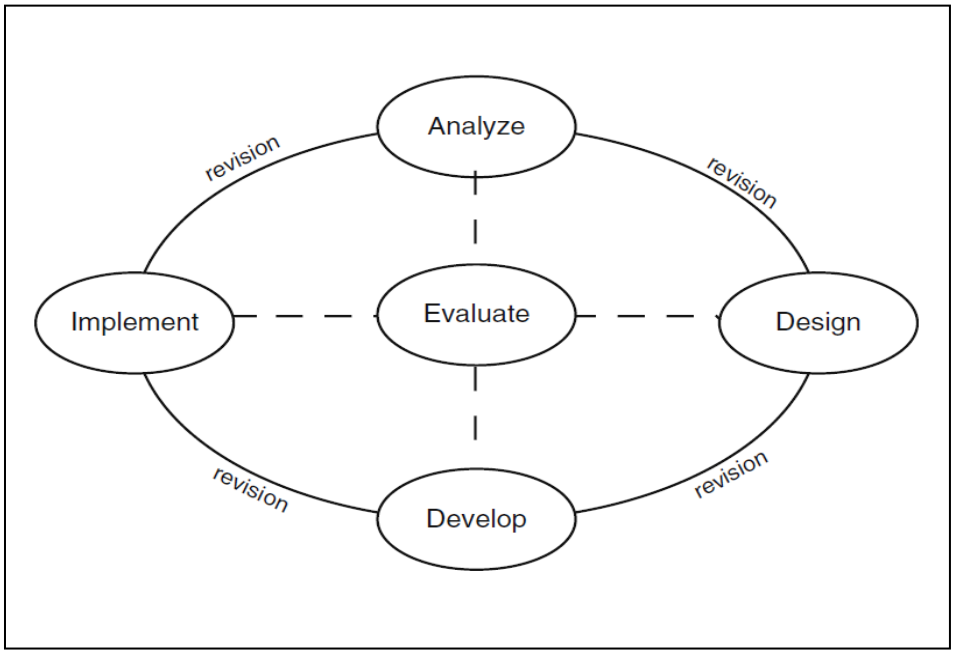


Figure 2.16: ADDIE Core Elements [103]

Different systematic instructional design process have been described [103]. Most of them have included the core elements of Analysis, Design, Development, Implementation, and Evaluation (ADDIE) as illustrated in figure 2.16. The ADDIE model illustrates the conceptual core component of instruction design; however, it does not explain the instruction design process involved. Instruction design process model have been discussed in many research [104].

Since SQL is a language, it might be possible to consider Ellis [105] research. Ellis research aims to answer the following question: How can instruction best ensure successful language learning? By arguing that there is no clear-cut answer. Ellis undertakings to reflect on different research, and then to identify a number of general principles that can provide guidance for instruction design. Rackliffe [106] used the Dick and Carey Systems Approach Model [104] (Figure 2.17) for the design and the development of SQL Tips which he considered as an instructional design.

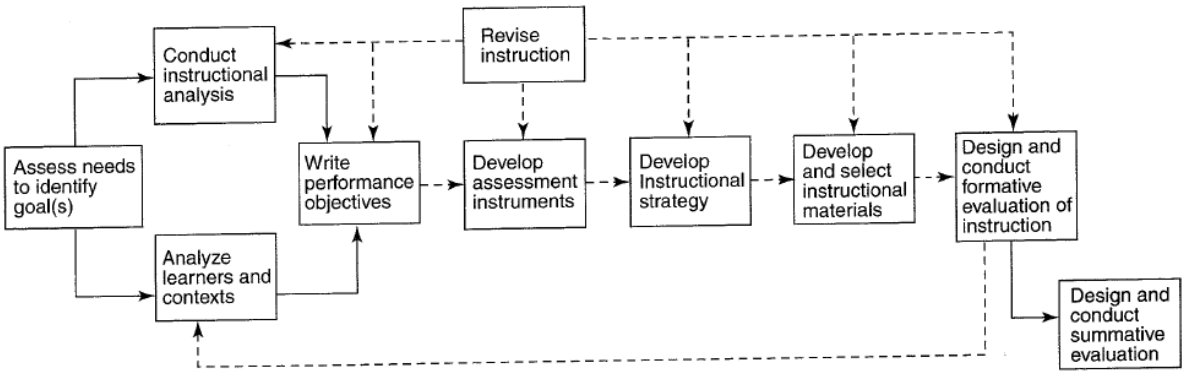


Figure 2.17: Instruction Design Model [104]

Applying the Instruction Design process can reduce reliance on trial and error planning [35]. Dick and Carey [104] and Morrison [35] models are considered here for many reasons:

- The analysis of the characteristic of learner and context.
- The focus on objectives, what the learner should learn and be able to do.

- The focus on the context (Instructional strategy), the choice of the methods, which describes the best way in which content and skills is learnt based directly on the specific learner outcomes.
- The focus on evaluation procedures. They support empiricism through the process of data collection and analysis to show the efficacy of the instruction and, based on the analysis, the instructional material is modified and improved.

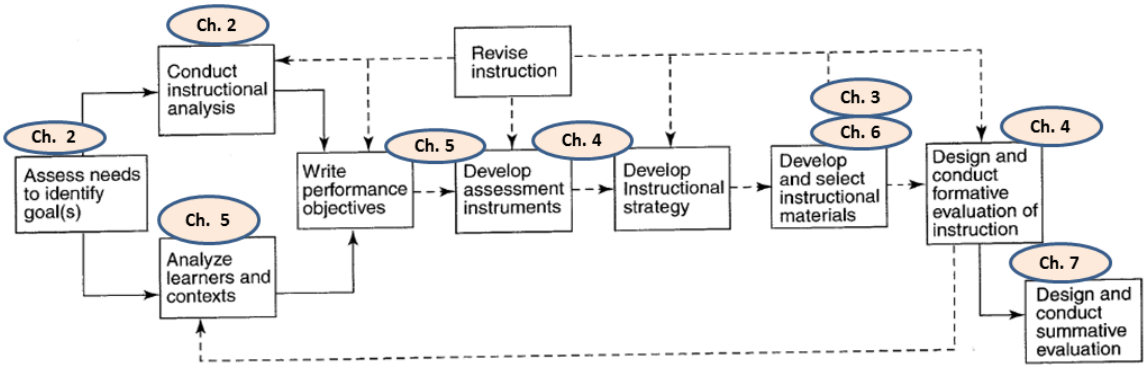


Figure 2.18: Employing Instructional Design Model [104] in This Dissertation

Figure 2.18 shows the component in Dick and Carey's model and where they have been achieved within this study i.e. the related chapters. In this dissertation, chapter 2 and 5 examine and identify the characteristics of students such as their pervious knowledge and skills. In addition, the analysis of the content, which is SQL, is identified in these two chapters. The results of the literature conducted in chapter 2 and the research method in chapter 5 analysed toward determining the "performance objective" [104]. This analysis of the objective is to define what knowledge and procedure need to be included. For example: what should be included to help learners improve their knowledge and skills? This is reported at the end of chapter 5 and structured as a framework called "SQL Framework Model". The objective should provide a map for designing the instruction and for developing the means to assess learner's performance [35]. The SQL framework model is used as a map to facilitate the ID objective.

Instructional strategy involves designing creative and innovative presentation of different knowledge to facilitate learning [35]. The design of this strategy is discussed in chapter 4. Chapter 6 presents the applied instructional strategy that employed patterns concepts to presents SQL different knowledge. Patterns fostering learning efficiency and the reuse of expert knowledge or expertise were discussed in chapter 3. In addition, the development strategy is evaluated in chapter 7.

Patterns concepts seem useful to transfer best practice and expertise. Patterns are captured in a specific structure to convey a context, problem, an example, and a reference to related patterns. More details about patterns and their application are provided in the next chapter (chapter 3).

Here, a general principle of Instruction Design (ID) and its application to this research was discussed. In the next section the theory and research related to CS learning is discussed.

2.6.2 Theory and Research Related to Computer Science Learning

It is essential to understand the nature of CS learning. CS, at its core, involves modelling the real world, representing domains of the most varied nature and complexity, representing knowledge in general and dealing with processes and solutions to problems in such domains. Therefore, any proposed instruction design should have, at its core, problem solving, to be engaged in after the basic knowledge is delivered and comprehended as was proposed early (see Figure 2.3).

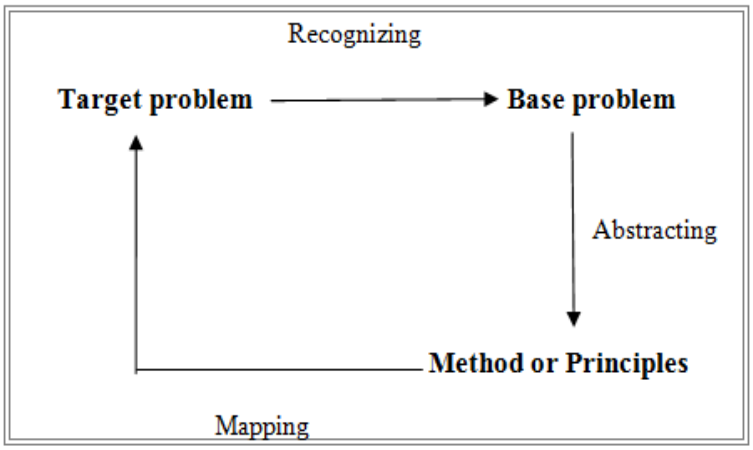


Figure 2.19: Problem Solving Approach Adapted from Quilici J.H., & Mayer R. E. [107]

As discussed in section 2.2, educators of SQL should use problem solving teaching methods and should not limit the focus on SQL content but emphasize the process of applying it. Bloom and Broder [108] and Mayer [34] suggest that problem-solving teaching methods should focus on the modelling of the "process" rather than the "product" and give students practice in comparing their strategies to those of models. This can be related to Qulici and Mayer [107] model, shown in Figure 2.19. They found that students required guidance to learn how to categorize static world problems on the basis of structure rather than surface features.

Analogical problem solving is a process of comparison using the learner's prior knowledge and applying it to the new problem or scenario [109]. This process depends on three cognitive processes:

i. *Recognizing*: in which the learner recognizes that an analogical connection exists between the base problem that has the same characteristic of the target problem or can be categorized in the same context.
ii. *Abstracting*: in which the learner abstracts the principle or the knowledge from the base problem.
iii. *Mapping*: corresponding elements of the problems link onto each other, and knowing how to apply the mapping to generate a solution to a variety of target problems.

Consequently, solutions are derived from applying the instructions learned from the base problem solving experiences [110]. One might say that using analogical reasoning in problem-solving enhances students' performance. However, it was not an automatic or spontaneous process. Rather, they found that prompting to use the analogue increased successful performance from 20% to 75%. This suggests the main problem in novice's problem solving depends on two factors. The first is the failure to recognize domain similarities between the target and base problem. The second is the learners' failure to retrieve the required knowledge. This is supported by Keane [111] who found that domain similarity between the source and target facilitated retrieval of the knowledge. As domain knowledge is a characteristic of an expert, Keane's findings could explain why experts are likely to use analogical reasoning. This leads to this question: How to teach a novice to think like an expert? What is missing in the current teaching approach? This is what this research aims to explore.

The researcher believes that learning SQL requires learning from new instructional designs that consider the *theory of analogical problem solving* in which learners solve the target problem by using a base problem that is similar to the target problem that has to be solved. There is no clear definition of the context of the "base problem", but in this research it is referred to as learners' schemata. More details are given in section 5.7.

2.7 Chapter Summary

This chapter explores learning processes and theory associated with SQL as well as aspects of the instructional design process that impact on this. It reviews educational theory, cognitive science theory and instructional design related research. The drive of this review on aspects of cognitive science and educational psychology is to provide a basic conceptual framework for use in the rest of the thesis when discussing related research and instructional design of SQL education for novices. In addition, this chapter aims to relate theses general theory to SQL teaching and learning and to provide an analysis of the SQL learner

performance problem from the related literature. This was initiated with a thorough literature review on learning taxonomy in general and its applicability in CS education. It concludes with a suggested taxonomy for CS education.

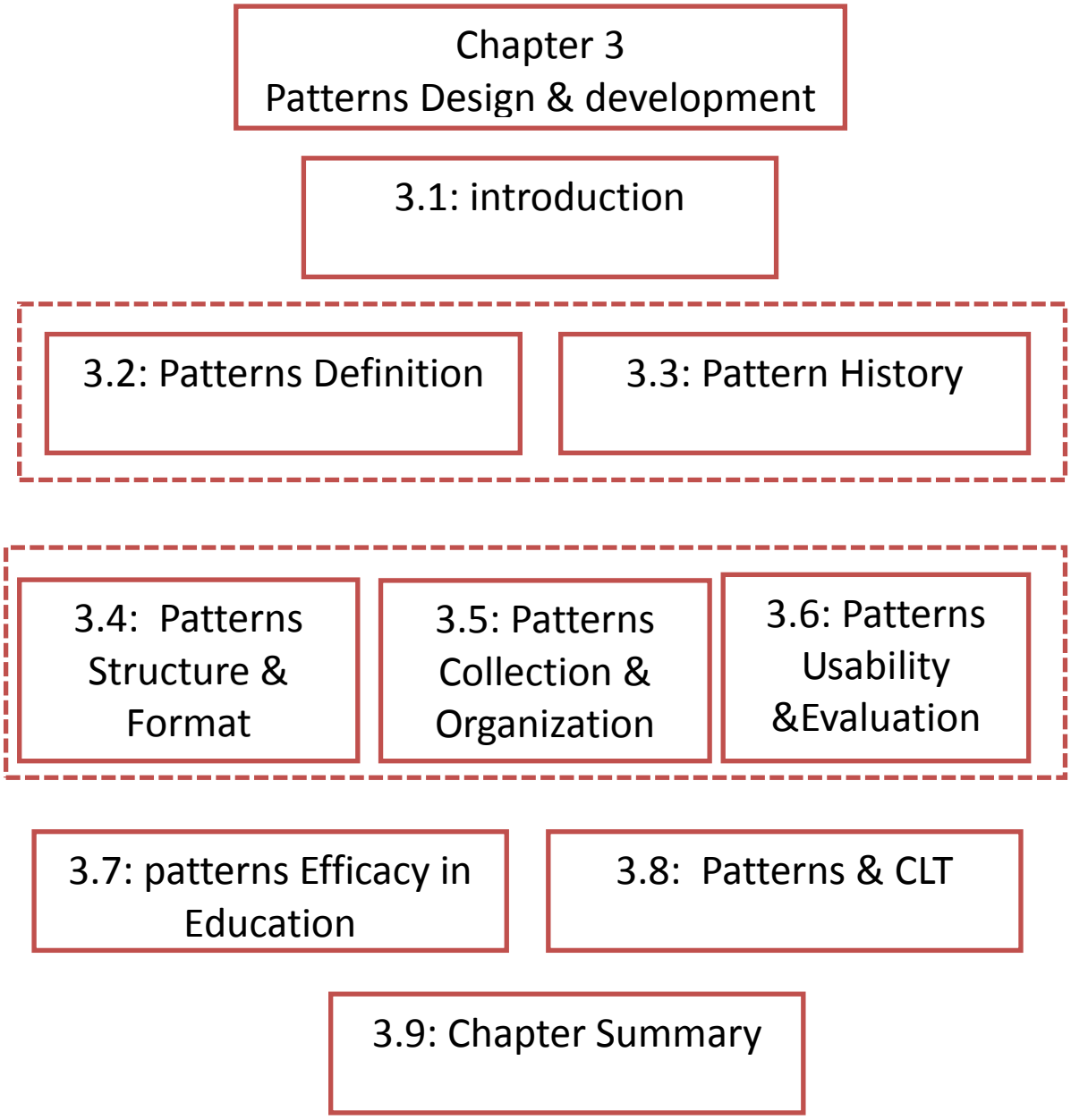
Teaching database courses in general and SQL in particular and the related cognitive research was discussed. It was followed by an analysis of how students solve problems using a thorough literature review on learn-ability and productivity of SQL. Analysing the relevant elements from these models, guide the research to propose a new model "SQL problem solving" model that it illustrates the steps learners should follow in term of solving SQL problems.

A review of the current teaching instruction (textbook and tools) was conducted. It presents the rate in term of its effectiveness on delivering SQL content and the problem solving process. The teaching approach and material should aim to help learners to apply different types of knowledge. It was conclude that the evaluated materials were not designed based on the instructional design principles and models. There were no considerations to the learners' characteristics or the development of their mental model. Moreover, the related objectives did not meet the essential learning tasks (CS learning taxonomy). For example, some of the tools focused on declarative only while others focused on developing problem solving skills. It was determine that there is a need to design a new instructional material.

To understand SQL usability and learnability and their related human factors, a review on the empirical studies used to evaluate the ease-of-use of SQL compared with other query languages and natural languages. In addition, the studies that evaluated the influence of using different database structures along with query language were discussed. In summary, it appears that students can be taught to use a variety of query languages (SQL, OBQ, SQUARE, etc.). Some studies showed that certain features of a language might be more difficult to teach than others. For example, the structure (or procedurally) of a language seems to be an aid in use and retention. Moreover, the learner's previous

knowledge and experience, and the extent of prior programming experience influence a user's performance.

The next chapter, 3, is aimed at investigating the possibility of applying patterns in design, development and evaluation of ID. The use of patterns in teaching and learning has been widely researched and debated.


Chapter 3
Patterns Design & development
3.1: introduction
3.2: Patterns Definition
3.3: Pattern History
3.4: Patterns Structure & Format
3.5: Patterns Collection & Organization
3.6: Patterns Usability &Evaluation
3.7: patterns Efficacy in Education
3.8: Patterns & CLT
3.9: Chapter Summary

Chapter 3: A Review of the Literature on Patterns' Design, Organization and Usability

This chapter presents the literature review on patterns design and development. It highlights the common knowledge available in the field of patterns, especially regarding the history, development, and usability.

3.1 Introduction

There is a growing interest in the possibility of using patterns in teaching design, development and evaluation. Patterns emerged from the idea of the architect Christopher Alexander [112]. Patterns are used to systematize main principles and pragmatics in the architectural fields. Those ideas have inspired many other fields like IT and education. Today, different types of patterns appear, such as Software Engineering Patterns, HCI patterns and pedagogical patterns. The use of patterns in teaching and learning has been widely researched and debated [113], but apparently there is a lack of empirical research onto the efficacy of patterns in education. Moreover, the lack of standard structural format and collection management [114] which adds to the doubt about patterns usability and their contribution in education.

This chapter covers the background and literature review on pattern design and development. The research highlights certain areas where there is a gap in knowledge as far as the efficacy of patterns and their usability is concerned, especially in education.

Section 3.2 presents a list of definitions for various types and uses of patterns. This is followed by patterns and anti-patterns history in section 3.3. Section 3.4 covers their structure and format with direct reference to different methodologies utilized by various researchers. Section 3.5 pinpoints the

important issues regarding patterns collection, which is followed, in section 3.6, by the usability of patterns. Section 3.7 reflects on the use of patterns in education. Section 3.8 reports the cognitive implications of the use of design patterns to transfer knowledge. Section 3.9 concludes by summarizing the problems that require further research and provides a link between the literature review and the research objectives.

3.2 Patterns Definition

A "pattern" is a phrase that has different meanings, uses, definitions and forms as demonstrated in Figure 3.1 by Wania [115].

Name	Source	Definition
Pattern	Merriam-Webster Online Dictionary	"1: a form or model proposed for imitation: exemplar 2: something designed or used as a model for making things <a dressmaker's pattern> 3: an artistic, musical, literary, or mechanical design or form 4: a natural or chance configuration"
Pattern	Dictionary.com	"1. a decorative design, as for wallpaper, china, or textile fabrics, etc. 2. decoration or ornament having such a design. 3. a natural or chance marking, configuration, or design: *patterns of frost on the window.* 4. a distinctive style, model, or form: *a new pattern of army helmet.* 5. a combination of qualities, acts, tendencies, etc., forming a consistent or characteristic arrangement: *the behavior patterns of teenagers.* 6. an original or model considered for or deserving of imitation: 7. anything fashioned or designed to serve as a model or guide for something to be made: *a paper pattern for a dress.*
Pattern	Alexander, 1979	"describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice" (p. x).
Pattern	Fowler, 1997	"an idea that has been useful in one practical context and will probably be useful in others" (p. 8)
Design pattern	Gamma et al., 1995	"are descriptions of communicating objects and classes that are customized to solve a general design problem in a particular context" (p. 3, italics in original) "design patterns capture solutions that have been developed and evolved over time. (p. xi)
Design pattern	Holzner, 2006	"a tested solution to a standard programming problem" (p. 8)
Design pattern	Borchers, 2001	"a structured textual and graphical description of a proven solution to a recurring design problem" (p. 7)

Figure 3.1 :Pattern's Definition [115]

Alexander's definition is a classic one.

> "Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice." [116], as quoted by Gamma *et al*. [117].

Patterns in Software Engineering (SE) context are defined as geared on the way to solving problems in software design [118]. Others gave more precise definition:

> "A piece of literature that describes a design problem and a general solution for the problem in a particular context" [119].

> "Are description of communicating objects and classes that are customized to solve a general design problem in a particular context" [120].

Schach [121], on the other hand, describes SE patterns as "a solution to a common design problem in the form of a set of interacting classes that have to be customized to create a particular design".

Patterns in HCI are somehow different from patterns in Software Engineering. Dearden and Finlay [113] defined HCI patterns as a "structured description of an invariant proven solution for a common user interface or usability problem that occurs in a particular context".

Pedagogical pattern can be define as a process of understanding the critical factors or principles in what makes good "Teaching and Learning Activity" design, and how they can be fore-grounded in a formal representation.

Ljubojevic and Laurillard [122] defined pedagogical pattern as a "structured set of core properties of a learning design (LD) that are critical to facilitating the student in achieving the intended learning outcome. Capture expert knowledge of the practice of teaching and learning".

Anti-patterns, on the other hand, are also defined as a "solution that looks good but it backfires badly when applied" [123]. To best describe anti-patterns, Crawford and Kaplan opted for the following approach in their book *J2EE Design Patterns* [124]:

> "Anti-patterns are to patterns what the falling skier is to the successful one: recurring, sometimes spectacular mistakes that developers make when faced with a common problem". (P.259)

3.3 Patterns History

The review of the history of patterns starts with the classical approach that was developed by the architect, Christopher Alexander. This was followed by development of patterns in the two fields relevant to Computer Science Software Engineering and HCI. Anti-patterns history, on the other hand, will also be explained and highlighted. The following sections outline milestones in patterns history.

3.3.1 Patterns in Architecture Design

The idea of pattern-like structures in architecture was first developed by the architect Christopher Alexander in his PhD thesis, which was summarized in "Note on the Synthesis of Form" [112].

Alexander refers to these hierarchically organized pattern-like structures as a way to represent design problems which makes the problems easier to solve by reducing the gap between the designer's knowledge and the design task. At that time, Alexander illustrated a relationship between the fundamental elements in

a pattern, the problem, the form (solution), the context, and the goodness of fit between the form and the context.

Then, Alexander and his colleagues introduced the architecture and urban design patterns [116, 125-128]. Alexander proposed a systematic approach which involves analytical decomposition of the architectural design problem into sub problems. This approach is built on the concept of pattern language. It is described in a series of books, namely the *Timeless Way of Buildings [125]*, *A Pattern Language* [116], *The Oregon Experiment* [128], *The Linz Café/Das Kafe Linz* [129], *The Production of Houses* [126], and *A New Theory of Urban Design* [127]. Then, Alexander's work fascinated other fields such as business, management, IT and education. Next, the history of Software Engineering patterns is discussed.

3.3.2 Patterns in Software Engineering

The first software patterns experiment was presented in 1987 at the OOPSLA conference on Object Orientation [130]. Later, different forms of patterns appeared. For example,

- The Formal Specification reusability to be reused for a family of products [131];
- Code reusability [132]
- System Architecture reusability [133]
- Design patterns [134] [117] and the "Gang of Four" [120]

Riehle and Züllighoven [135] present a patterns language for their "tools and materials" design metaphor that involves tools, materials, aspects, and environment. Many patterns-focused conferences have been held since 1994. An example is the "Pattern Languages of Programming" (PLoP) that was first held in August 1994. PLoP conferences have been held annually since then. Other conference series investigating pattern languages in Software Engineering have also been established (e.g., EuroPLoP in Europe, ChiliPLoP in Arizona, and

KoalaPLoP in Australasia, Mensore PLOP in Japan). The next section presents HCI patterns' history.

3.3.3 Patterns in HCI

The earliest HCI oriented reference to Alexander's patterns ideas was on user-centred system design [136]. The issue of HCI patterns languages was addressed more intensively by the beginning of 1997 at HCI'97, the annual HCI conference. Then, the use of pattern languages became very attractive after the *Promises of pattern languages* [137]. Different HCI patterns collections can be found in both electronic and hard copy formats. To name a few: user-interface patterns in the "Common Ground" Tidwell, J. [138], UI patterns and techniques [139], Designing interfaces [140], A pattern approach to interaction design [141], A pattern language of statecharts [142], Van Welie [143].

3.3.4 Patterns in Education

It is possible to say that the patterns in education took different forms. Pedagogical patterns, as defined in section 3.2, aims to record good Teaching and Learning Activity to help educators in transferring their experience in teaching. The other way of using patterns in education is to use different sets of patterns in a particular field such as HCI or SE as a teaching method or tool [15, 144].

In the mid-1990s, the pedagogical pattern project [145] started by collecting many types of patterns that can help teachers and students. Then Pedagogical patterns were presented in several patterns collection projects [146, 147] such as ICOPER, TELL, Learning Designs, and PLANET. These collections focused on teaching practice, evaluate/theoretically analyse, and describe the patterns using the patterns collection-specific template (pedagogical patterns). Several different pattern languages have been developed. The pattern languages presented at the pedagogical patterns projects website are organized in many

different ways: some are organized according to a given activity (e.g. Feedback, n.d.), others according to pedagogical values (e.g. Active Learning).

Pedagogical patterns are considered as suitable tools to document the successes of pedagogical activities frameworks in order to enable their reuse [148] and to develop educational frameworks based on conceptual solutions of published pedagogical patterns. Many approaches have been suggested to incorporate patterns into the classroom activities, aimed at teaching computer science concepts and enhancing computer science problem solving [148-151]. Bergin developed a collection of fourteen pedagogical patterns for teaching CS, which formed the basis for a pattern language for CS course development [152].

This research is not focusing on pedagogical patterns but in the teaching of a particular concept through the use of relevant patterns. Barfield *et al*. [153] represented the earliest usage of patterns in education when they used the patterns approach in the interaction design curriculum at Utrecht School of Arts. Many researchers use patterns and teach with patterns; for example [15, 113, 144, 154]). These kinds of patterns are discussed throughout this chapter in terms of their format, organization and usability.

3.3.5 Anti-patterns

Anti-patterns, as a concept is not new; they are common-place in society, and they have been around since software's inception – for example spaghetti code [155]. The idea of anti-patterns was promoted after the pattern term has been published. In 1995, Koenig published a short article using the term for the first time in the Journal of Object-Oriented Programming [156]. Brown [123] published the first book on anti-patterns, where the term was expanded to include Architectural, Design, and Management anti-patterns. Eventually, anti-patterns scope was expanded further by Laplante and Neill in their book *Anti-patterns - identification, refactoring and management* [157]. They included broader management and leadership aspects and introduced other types of anti-patterns called cultural or environmental anti-patterns. The anti-patterns term

was applied to their use in education by [158, 159]. Laplante and Neill [157] mentioned neglect, malice, and ignorance as other reasons for the existence of anti-patterns. In this research, Anti-SQL patterns are not considered since they raise an alarm of anti-counter to the novice learners.

3.4 Patterns Structure and Format

It is easy to observe phenomena in the world but much more difficult to use these observations to develop an explicit good design [160]. If the patterns are not written in a precise way, they are going to cause difficulty and ambiguity for users, especially novices. After the pattern has been discovered, it needs to be formulated at a graduate level of abstraction [161]. This is because a too abstract or too detailed pattern will not be practical for encouraging efficient design use. This aspect was addressed earlier by Alexander [125].

Common components of a Pattern	
Name(s)	
Problem Context Real-world example	Describes the problem and context and shows where this pattern would be used and the conditions that must be met before this pattern is used
Solution Design/structure Implementation	It presents a description of the elements that make up the design patterns and shows their relationships, responsibilities and collaborations.
Consequences	Discuss the pros and cons of using the patterns and the impacts on reusability, portability, extensibility enumerated.
Variations, known uses	Other names or phrases

Table 3.1: Patterns Element

Indeed, one can argue that patterns need to be structured carefully in order to be effective and usable tools for both expert and novice designers. Before discussing the structure of a new set of patterns in chapter 6, a quick review is conducted for the common characteristics and the difference between pattern structure in its original field, Architecture, and in Software Engineering, in HCI

and possibly in pedagogical patterns. Table 3.1 shows the patterns' common elements of those, which all authors agree on.

3.4.1 Alexander's Patterns Structure "Alexandrian form"

Alexander's patterns consist of the following components:

- The name of the pattern
- A ranking of its validity
- A picture as an example of its application
- The context in which it is to be used
- A short problem statement
- A more detailed problem description with empirical background
- The central solution of the problem
- A diagram illustrating the solution
- A reference to smaller patterns

Alexander uses a special text layout to distinguish different parts of his patterns. For example, the problem statement and solution statement are printed in bold font, as shown in Figure 3.2 below.

3 CITY COUNTRY FINGERS

Every city dweller would have access to the countryside; the open country would be a half-hour bicycle ride from downtown.

Therefore:

Keep interlocking fingers of farmland and urban land, even at the center of the metropolis. The urban fingers should never be more than 1 mile wide, while the farmland fingers should never be less than 1 mile wide.

✣ ✣ ✣

Whenever land is hilly, keep the country fingers in the valleys and the city fingers on the upper slopes of hillsides—AGRICULTURAL VALLEYS (4). Break the city fingers into hundreds of distinct self-governing subcultures—MOSAIC OF SUBCULTURES (8), and run the major roads and railways down the middle of these city fingers—WEB OF PUBLIC TRANSPORTATION (16), RING ROADS (17). . . .

Figure 3.2: A Sample of a Pattern by Alexander *et al.* [116]

3.4.2 Software Engineering Patterns Structure

In Software Engineering, a range of alternative formats appear in [120, 134, 162, 163]. In all, the overall format of a pattern has not changed very much from Alexander et al. [116] to, for example, Gamma *et al*. [120]. Patterns template in Gang Of Four (GoF) consists of the following:

- **Pattern Name and Classification**: it is the name for the pattern and the pattern's type
- **Intent:** it is a statement about what the pattern does
- **Also Known As**: alternative names for the pattern
- **Motivation:** A scenario that shows where the pattern would be useful
- **Applicability:** where the pattern can be used
- **Structure:** A graphical representation of the pattern
- **Participants:** The classes and objects participating in the pattern
- **Collaborations:** How do the participants interact to carry out their responsibilities
- **Consequences:** The pros and cons of applying the pattern
- **Implementation:** shows the techniques for implementing the pattern
- **Sample Code:** Code fragments for a sample implementation
- **Known Uses:** Examples of the pattern in real systems
- **Related Patterns:** other existing patterns that are related to the pattern

As it is clear, GOF patterns' structure still has similar elements to Alexander patterns although it was called differently. The next section presents an overview about HCI patterns structure.

3.4.3 Human Computer Interaction (HCI) Patterns Structure

HCI patterns' structure has changed gradually. Tidwell [138-140], Borchers [130, 164], [165] have all described for each pattern, in general, the following structure:

- the name of the given pattern;
- the usability of the pattern;
- the context in which the pattern should be applied or when to use;
- the force that influence the user;
- the design solution;
- examples where the pattern has been applied;
- the usability impact; and
- the rationale behind the pattern

Sometimes some sections are called differently. It is clear that the structure of user interface design patterns has changed during the last 10 years. It has moved from the Alexander format (problem, context, solution and forces) to the (what, when, how and why) format that appears in Tidwell [140] and Van Welie [166]. However, some patterns structures are simply a description of the issues augmented with examples, as in Sally [167].

It is appropriate to highlight the study by [114] regarding HCI patterns format; two issues have been pinpointed:

1. Knowledge of the activity that involved creating and integrating HCI design patterns are seldom identified.
2. Engineers experience difficulties formulating problem statements with the end-user in mind.

In addition, another two points were highlighted by Specker and Wentzlaff [168]

1. HCI design patterns were represented by graphics and a corresponding text passage containing their natural language description.
2. There are synonym patterns in diverse collection, although the pattern authors used different names and described them in different ways.

To reflect on the above highlighted issues, the patterns author, in order to write an effective pattern, must rely on both end users' characteristics and patterns'

content specifications and nature. Patterns in their current collection do not distinguish between novice and experts. All are designed for one homogenous user. For example, when the created set of patterns is intended to be used by students, then patterns' writers need to take into consideration learner's characteristics, learnt subject specifications, and the different methods and approaches that have been used in teaching the subject content. The structure of the patterns must be corresponding to the learner's theoretical and practical understanding of the task, which will require the application of used patterns. In addition, the process of patterns identification should be documented clearly. Different methods and approaches that have been used in teaching SQL need to be studied. The next section presents the nature of pedagogical patterns structure.

3.4.4 Pedagogical Patterns Structure

Ljubojevic and Laurillard [122] argue that pedagogical format's structure should not be not too rigid, so that practitioners can still use their own language and labels to denote the activities and processes, but these are slotted inside the formalised structural whole. Table 3.2 presents pattern structure in different collections.

ICOPER	***Planet***	***TELL***
Author & Copyright		Credits
Summary/ Thumbnail		Context
Rationale	*Rationale*	*Rationale*
Subject/ Discipline	Context	
Learning outcomes	*Problem*	*Problem*
		Forces
Group size		
Duration (part)		
Learner Characteristics		Audience
Sequence of Activities	*Solution*	Diagram
		Solution
Roles		
Type of Assessment		

Table 3.2: Pedagogical Patterns Structures [122]

Ljubojevic and Laurillard [122] defined template (Table 3.2) that reflects strict focus on the core pedagogical properties. The following are the elements in their template:

- **Title:** the name for the pattern
- **Summary:** of the following: "To what End by What Means"; this will potentially be used by the search engine to make inferences about the functional orientation and character of the pattern
- **Rationale:** for providing learning theory justification that links learning outcome with the pedagogical method
- **Learning outcomes:** presents the Higher Cognitive Skill learning outcome(s), most commonly of the following form: "To Be Able To Perform/Apply/Resolve" etc
- **Sequence of Activities:** ordered and timed sequence of Teaching and Learning Activities, each interpreted for the type of Conversational Framework activity it represents
- **Type of Assessment:** How can we prove that the learning outcome is achieved?
- **Time:** Duration of the TLAs sequence that executes this pattern

Ljubojevic and Laurillard [122] collection omitted the context of a learning design; they argued that generic patterns need to be adapted to its local context. However, they included more details on the way the learner's and teacher's time is spent on the learning activities.

3.4.5 Summary

A quick analysis of the common characteristics and the difference between pattern structure in its original field, Architecture, and in Software Engineering, in HCI and possibly in pedagogical patterns was highlighted.

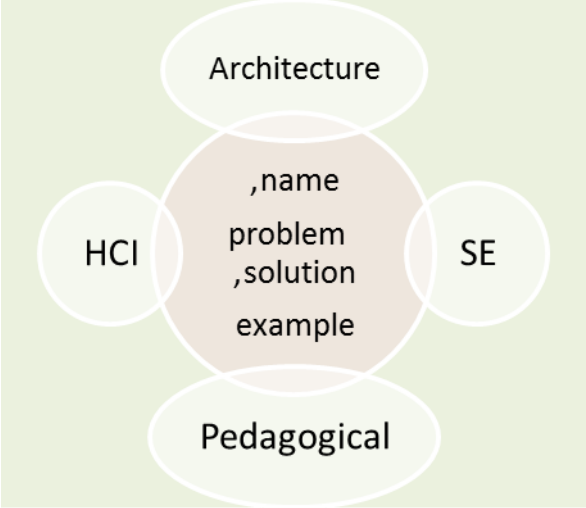


Figure 3.3: Patterns Common Elements

In most patterns collections, patterns have similar content purpose. Each pattern has a name, the sort of problem that needs to be solved and the related solution suggested by the pattern. There are few issues that have been discussed in [114, 168] about patterns structure. Each pattern is supported with one or more example. In terms of individual patterns, the pattern name in some cases was inconsistent and difficult to learn or recall; some patterns appear with the same name with different content in two collections. Patterns sometimes exist with the same name but with different content; two patterns may contain the same problem statement; pattern content may embed other patterns; lack of standardization of pattern format/structure is sometimes confusing. To design a new set of pattern that aims to help students understand a specific knowledge and skill, the following elements must exist:

1. Exploring and documenting the processes by which patterns are recognized, identified or discovered and recorded. Through employing cognitive psychology, literature in how such content is learnt (educational theory).
2. Patterns need to be constructed in a way to provide knowledge required for problem solving.
3. Finding ways to format patterns, through employing some strategy aiming to facilitate knowledge transfer.

After an individual pattern is structured and formatted, then the set of patterns need to be organized effectively. The following discussion is about the different ways that have been used to organize patterns.

3.5 Patterns Collection and Organization

A pattern collection is any set of patterns that might include other subsets. Granlund *et al.* [169] state that:

> "Patterns must also be part of a language of interrelated patterns, participating in and supporting each other, in order to be truly useful". (p.2)

Many researchers highlight the importance of organizing patterns and suggested one or more organizing principles. According to Salingaros [170]:

> “A loose collection of patterns is not a system, because it lacks connections”. (p.154)

More recently Todd *et al.* [171] confirmed this issue saying that:

> “Unless people can fully understand the organization of the language, they find it difficult to select appropriate patterns". (p.33)

This section elaborates different pattern’ collections in Architecture, Software and HCI and their technique in organizing and structuring patterns in patterns language. In addition, this section explores pattern’s collection in the three fields: Architecture, Software and HCI.

3.5.1 Patterns Collection in Architecture

Alexander [125] describes patterns collection as:

> "The structure of the language is created by the network of connections among individual patterns: and the language lives, or not, as a totality, to the degree these patterns form a whole". (p. 305)

Alexander describes a 250-pattern multi-layered pattern language. Alexander's patterns collection is considered the golden standard for a pattern language as a result of its completeness and richness. The pattern, within Alexander's pattern language, are hieratically connected to one another, in the way that higher level patterns are made up of lower level patterns, and these relationships are made explicit within the patterns.

3.5.2 Patterns Collection in Software

The first Software Engineering pattern collection was by the GOF in the publication of '*Design Patterns Elements of Reusable Object-Oriented Software'*. Gamma et al. [120] classifies design patterns by two criteria, purpose and scope. The purpose criterion reflects what a pattern does: creational, structural, or behavioural. The second criterion, pattern scope, specifies whether the pattern applies primarily to classes or to objects. There were 24 patterns in the "Gang of four" collection. Although the "Gang of Four" was regarded as the archetype of a software pattern book, the collection and the linkage between the individual patterns is not complete enough to constitute a language [130].

3.5.3 Patterns Collection in HCI

It is possible to organize patterns according to more than one appropriate establishing principle for pattern languages within HCI [172]. The first substantial set of user-interface patterns was the "Common Ground" Tidwell, J. [138], UI patterns and techniques [139], Designing interfaces [140], A pattern approach to interaction design [141], A pattern language of statecharts [142], and Van Welie [143] who extended the collection followed it.

User Interface (UI) patterns users have identified the organization of a pattern collection as a major issue when using patterns to guide UI development [115, 173-177]. Recently, Seffah [178] argues that pattern languages need to define the relationship between individual pattern clearly.

Fincher and Windsor [179] suggest some requirements for an organizing set of patterns. They point out that pattern languages:

- Should help users to find patterns easily
- Should enable users to find related patterns
- Should allow users to evaluate the problems from multiple viewpoints
- Should allow users to build new solutions

The following are the summarizing points that cover this debate:

1. HCI design patterns have no universally accepted standard for describing them and a way of organizing and categorizing them is still lacking and needs further research.

2. HCI design patterns are represented in screenshots and a corresponding text passage using natural language.

3. HCI design patterns name was in some cases inconsistent and difficult to learn or recall during the design.

4. HCI design patterns synonym appears with different name or described in different ways in diverse collection.

5. HCI design patterns collections are limited and none of them covered all the cases. They are focusing on different levels of the design process, such as task or representation.

6. There is no tool support for usability patterns engineering.

7. HCI design patterns creation, integration and usability are not identified explicitly. In another example of pattern collection management, Gaffar *et al.* [180] used "The Seven C's Methodology" which aims to centralize patterns into one repository. This methodology includes seven steps:

Collect: all different patterns in one Central Data Repository
Cleanup: change from different format into One Style
Certify: define a clear terminology for the collection
Contribute: receive input from the community
Categorize: define clear category for the collection
Connect: establish a semantic relationship between patterns in a Relationship Model
Control: machine-readable format for future tools

Different techniques of organizing patterns were proposed by van Welie and van der Veer [143]. They suggest that patterns be organized by function, by problem similarity, by user task, and by user type. They present a hierarchical partial pattern language for web design, which contains a number of different levels including posture level, experience level, task level, and action level. Henninger and Ashokkumar [181], on the other hand, proposed an ontology-based structure for organizing pattern languages(Figure 3.4).

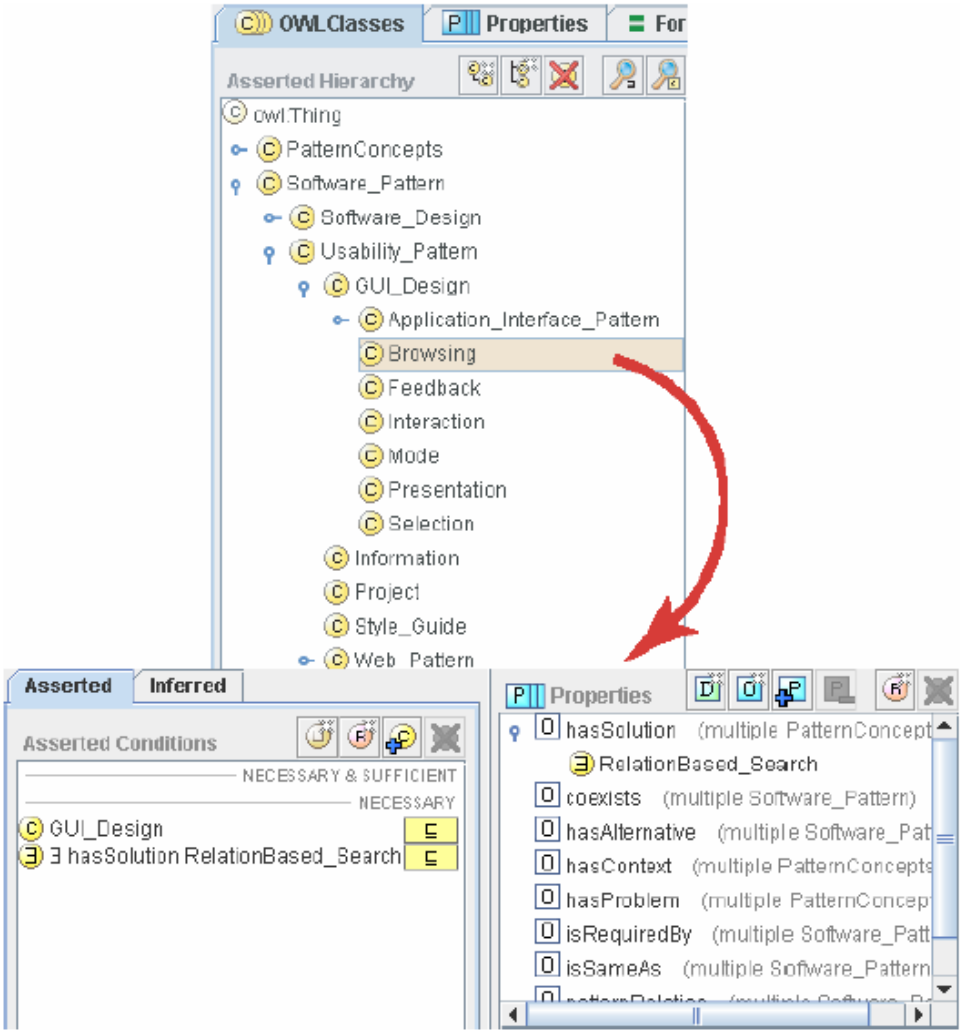


Figure 3.4: An Example Class Structure of an Ontology for Usability [181]

They suggest a connection strategy between patterns, which consist of the following: "contains", "is equivalent", "is an alternative", "is specialization", "is to be used in combination with", and "is disjoint with". The next section gives an overview about patterns collection in education.

3.5.4 Patterns Collection in Education

Creating a pattern language specifically for education use was found to be beneficial for students in the web design field [182] and in the teaching of SE [15, 16]. In terms of specific-purpose patterns collection, Todd *et al.* [171] presents two versions of Teaching UI, pattern language (TUI) (see Figure 3.5)

which is a pattern language that specifically aimed at teaching students about UI design.

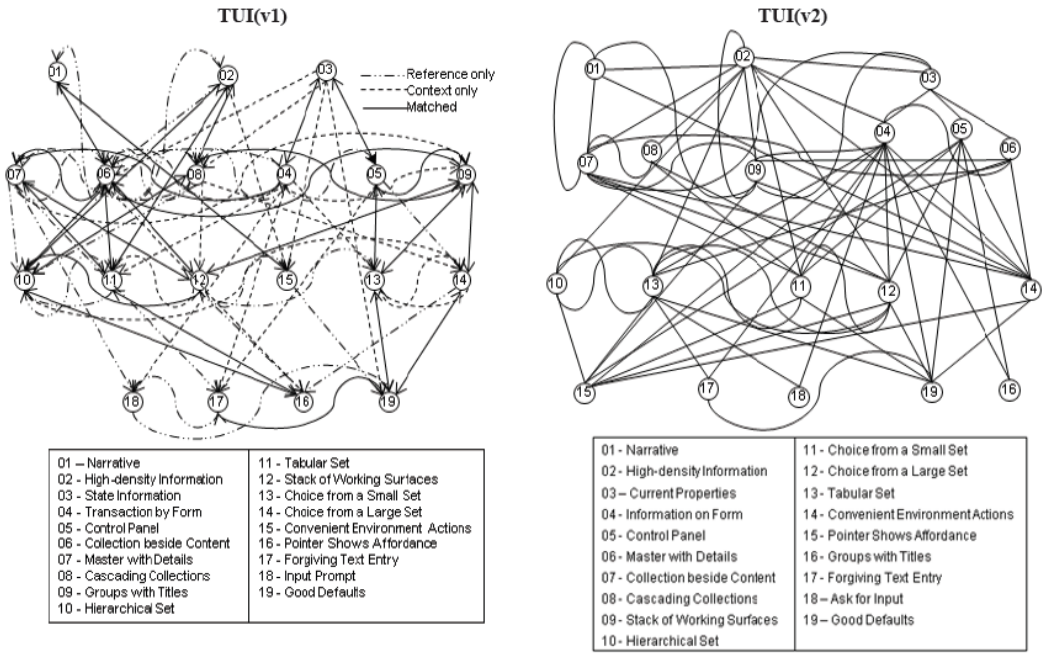


Figure 3.5: Teaching UI, Pattern Language (TUI) [171]

Research have been carried out to evaluate pattern usability in education. However, these efforts were limited to a specific collection of patterns. Furthermore, some of this research has been carried over a restricted period (for example, one or two lectures) and may not necessary reflect long-term usage. Therefore, many researchers call for further study into the use of patterns in education. Hence, this is another justification of the scope of this research and its contribution to filling a gap in the knowledge.

3.5.5 Summary

Different techniques were applied to collect and organize patterns, to facilitate their use. However, in terms of pattern collections, the following issues are apparent: the limitation of the existing pattern collections; each pattern collection have different ways of organizing and classifying patterns; the way of

writing patterns is different from one collection to another; and finally each pattern collection is intended for different reasons and purposes. Therefore, there is a need to conduct a further research in how to maximize the efficacy of patterns through a well-designed organization method. To organize patterns, especially if the aim is to improve novice performance, it is essential to employ some educational strategy such as checklist, or scaffolding techniques.

Todd et al. [171] proposed UI pattern language Maturity Model (UMM) that provides a method for rating the maturity status of UI pattern languages. This approach is a good way to evaluate other pattern language collections. The next section explores the use of patterns in different fields.

3.6 Patterns Usability

This section explores the use of patterns in different fields. To determine whether this assertion is correct According to Alexander [125], anyone can use a pattern language to design buildings and emphasizes that pattern languages is not exclusive to building designers. Alexander's use patterns to support participatory design by a value system that treats localised control and contextual sensitivity in design as essential. The Linz Café [129] and "A New Theory of Urban Design" [127] highlights the value of making decisions on the actual construction site, and taking into account the surrounding context. In The Oregon Experiment and The Production of Houses, Alexander et al. [126, 128] emphasised the use of patterns by a community to design for itself. According to Alexander [125]:

> "In this same way, groups of people can conceive their larger public buildings, on the ground, by following a common pattern language, almost as if they had a single mind". (p. 427)

Many studies have acknowledged the benefits of pattern application in SE [15, 117, 183] which can be summarized in the following points:

1- Patterns create a common vocabulary for communicating designs and support reuse at the design phase. Thus, design patterns could direct the entire process and community.
2- Patterns improve software maintainability and most importantly making the design more flexible to cater for future changes. As a results of using design as a documentation tool to classify the fragments of a design.
3- Patterns could provide valuable assistance to less experienced designers in producing better designs.
4- Design patterns can be used to build robust designs with design-level parts that have well understood trade-off.

The usability of HCI design patterns, on the other hand, has been studied by many researchers; for example [114, 168, 176, 178, 184].

Erickson [185] points to the two most often cited reasons for the use of pattern languages: quality and reuse. Granlund *et al.* [169] highlight pattern reuse as one reason why there has been increasing interest in patterns and pattern languages in HCI. They also acknowledge many of the reasons, mentioned earlier, for interest in the topic including the fact that patterns may offer a way to capture and transfer knowledge, patterns may provide a lingua franca, and patterns may support both analysis and design.

3.7 Patterns Efficacy in Education

This section focus on the use of patterns in education; i.e. both by teaching pattern languages and teaching concepts such as Architecture, SE, and UI throughout patterns languages. Architecture patterns used to teach students about aspect of urban design [14]. In Software Engineering, [15, 16] both recommended using patterns to teach novices. Patterns can provide a list of things to look for during a design review and a list of things that must be taught in a course on OO design [15]. Astrachan et al. [16] argued that patterns should form an essential part of the undergraduate Computer Science curricula.

Borchers [144], on the other hand, suggested two ways of using patterns within the curriculum: as a tool to present HCI design knowledge to students and as a methodology to support design.

Other studies evaluate the effectiveness of pattern to support novice learning [186]. pattern languages suggested as a good tool for teaching [16]. Jalil and Noah [187] conducted an exploratory study to explore the actual difficulties of using design patterns among novices. The factors that influence learning and use of design patterns in education were highlighted by Weir [188].

Some researchers suggest further research into the efficacy of patterns in education. Dearden and Finlay [113] suggest that a significant effort is now required to examine the use of patterns in education to demonstrate what benefits might be gained from a patterns approach.

Since the focus of this research is on the efficacy of patterns in education, a review study on the empirical studies in using patterns in teaching and learning must be conducted, and this is what follows in the next subsections.

3.7.1 The Review Design

This review is addressed to researchers and practitioners in Computer Science education. Consequently, the primary focus is on the use of patterns in teaching and learning process that is relevant to the efficacy of patterns. There are, however, a large number of patterns from other domains (e.g., Software Engineering and HCI) that may have been evaluated in term of its efficacy. To avoid extending the scope of the review beyond practical limits, it was limited to HCI patterns which are considered among the most cited resources in CS education.

The evaluation of pattern usability in education should focus on the following:

- Participants: the type and the number of participants; Participants should be students with a known level of expertise in the tested field.
- Environment: it should be similar to a classroom or a lab.
- Teaching and training: needs to discuss in detail components of the study. Some of the studies lack adequate periods of training (hours) which could be considered rather minimal.
- Methodology: needs to focus on the important aspect of learning and the cognitive aspects as well.
- Results: should focus on the value of patterns in terms of both the students' performance and satisfaction.

To highlight the issues with the empirical work conducted by various researchers in using patterns to teach or assess students understanding of different computing concepts, the following aspects need to be examined:

- How strong was the experimental design?
- Did they use experts or novice students?
- Do they document the background of participants and therefore address the issue of expertise when assessing the usability and efficacy of patterns?
- How was the pattern design evaluated?
- Were the findings based on students saying the patterns helped them or was there a more objective way of determining the effect of the patterns?
- Did the studies really prove the value of the patterns? If they did, how was it proved?
- Was there any attempt to follow-up the study by contacting the student's later in their careers to see whether they made use of the patterns later?
- Were the exercises "rich" enough to support pattern usage?

Although some of the above questions are addressed, others studies do not tackle or report on some of these issues. The above aspects are used to critique

the experiments that are included in this review. The review form, in Figure 3.6, was used to review most of the published related research.

<experiment resource\investigator>	
Experiment Design	
Aims	
Patterns type\name	
Control experiments	
Task rich enough to support pattern usage	
Hypothesis	
Variables	
Participants	
Participant's general background	
Participant's experience with patterns	
Participant's knowledge with patterns	
Number of participants	
Results	
Evaluation methods	
Result analysis	
Finding	
Learning effect	
Further researched addressed	

Figure 3.6: The Review Form

The review data was organized into four parts:

- Participants: Number and type of participants.
- Results interpretation: shows the research methodology and how the results were analyzed.
- Critique: evaluate the experiments in term of the five aspects that were discussed earlier in this section.
- Patterns value: should focus on the value of patterns in terms of both the students' performance and satisfaction.

3.7.2 Past Experimental Work

Using HCI design patterns are considered to be more effective in transferring knowledge compared with the use of guidelines [189]. Many studies examine the use of HCI patterns. This review here considered HCI patterns because it is possible to argue that the evaluation of HCI patterns is more widely cited. Table 3.3 lists the experiments that report using HCI patterns. Two were excluded from the analysis because the subjects were not students, which is the main focus of this review.

<experiment resource\investigator>	Included	Reasons for exclusion
Borchers [144]	Yes	
Dearden et al. [174]	No	Subjects were not students
Dearden et al. [175]	No	Subjects were not students
Chung et al. [190]	Yes	
Saponas et al. [191]	No	Subjects were not students
Cowley and Wesson [192]	Yes	
Kotzé et al. [193]	Yes	
Koukouletsos et al. (Koukouletsos, Khazaei et al. 2009)	Yes	
Kolfschoten et al. [194]	Yes	
Todd et al. [171]	yes	
Bernhaupt et al" USER INTERFACE PATTERNS:A FIELD STUDY EVALUATION"[195]	No	Subjects were not students

Table 3.3 : The Published Experimental Research in The Field of Teaching HCI Patterns

The following is a detailed discussion of some of the included studies above-mentioned research:

[1] Borchers (2002)

This study reported the results of using patterns format to teach HCI basics to computer science students. The researcher published work related to two studies using patterns to teach two HCI design courses.

In the first course, the researcher gave a 90 minutes lecture to introduce the HCI pattern idea, history and usability. Then all the students were given 15 minutes to study Tidwell's Common Ground HCI patterns collection and to find the patterns that were related to the exercise that they were working on (designing a user interface prototype). After two weeks, the students were evaluated in terms of patterns retention, the usefulness of patterns in terms of understanding and remembering user interface design concepts, matching patterns to their own design projects and using patterns in future design projects.

Participants- study1

In the first course, 32 students participated to evaluate the use of patterns. 26 answered the questions about patterns. Results and interpretation:

- ➔ Remembering patterns with average 1.37;
- ➔ The usefulness of patterns in terms of understanding and remembering of user interface design concepts with average 1.96;
- ➔ 2.23 average in matching pattern's to their own design project; and
- ➔ Using patterns in future design project with average 1.94.

Critique

The observations here are focused first on the lack of teaching patterns. A 90 minutes lecture could be considered rather minimal. Second, the author did not consider the number of patterns that students used. Thirdly, no evaluation or mapping of the solution to exercises using patterns was carried out.

In the second course, the author spent 8 weeks, 110 minutes lectures teaching patterns in interaction design. At a different stage of the course, the students were asked to write their own patterns at the stage where students had not been introduced to the patterns concept in full. The students were also asked to create HCI pattern language by identifying patterns from the application domain, HCI design patterns and the software engineering area of any project

they had recently worked on. At the end of the course, the students were asked to explore, in essay, some of the advanced topics such as success and failure stories of actual pattern use. All the assignments were reviewed and discussed during in-class writer workshops in groups of four.

Participants- study 2

18 students participated (8 undergraduate students, 5 CS master students, 3 MA students, 1 psychology PhD student and 1 post-doctoral student).

Results and interpretation

Two measurements were recorded in this study. A direct evaluation, based on the quality of the set of patterns which students created then reviewed and re-wrote to indicate how the patterns concepts were conceived. The study revealed that students did not have any problem in understanding or applying the pattern format to their own contributions, although students had a problem finding the right level of granularity and abstraction in their patterns. The indirect evaluation was carried out by means of the course evaluation questionnaire.

Critique

Students were not a homogenous group; some have had considerably more knowledge and experience than others. The positive side of this work is that students were engaged into a realistic course spread over a full semester of teaching.

Pattern values

HCI design patterns can be used to teach basic HCI design principles and can lead to above-average retention of design principles and to a quick adoption of the pattern vocabulary, even amongst first year undergraduates. Students consider the pattern format useful in formulating their own design experiences.

[2] Cowley and Wesson (2003)

This empirical study involved a heuristic evaluation of a website, the redesign of an existing website and the design of a new website using patterns, as compared to using guidelines. The participants were divided into two groups. One group used a set of patterns while the other used a set of guidelines. Porcupine Ceramics website was chosen as a suitable E-commerce website for evaluation and redesign. Conclusions were based on the initial analysis of the students' ratings of their opinion about the use of patterns compared with the use of guidelines.

Participants

The participants were 33 Computer Science Masters and Honors students who were registered for E-commerce course.

Results and interpretation

A post-test questionnaire was used to record quantitative and qualitative data about the participants' attitudes towards the design aids. The study data was analyzed in terms of 5 categories: evaluation, redesign, new design, format or content and general experience. For each category, three properties were identified: efficiency, effectiveness and satisfaction. Mean, standard deviation and the number of items were calculated for each property.

Critique:

The first concern of this study relates to the lack of explicitly stated hypotheses or experimental variables. Secondly, no details were provided about the task type. Lastly, the study results were not interpreted or analyzed by the authors.

Pattern values:

The study concluded the following: those designers consider patterns to be an efficient and effective aid for evaluation, redesign and new design. Patterns structure and content is found useful. Finally, designers consider patterns to be a personal design language.

[3] Kotzé et al. (2006)

This study used a selected set of patterns from the Van Welie [166] collection and a corresponding set of guidelines. These were used to teach participants, in an optional HCI module, Usability Principles and Web Interface Design. One group used the pattern and the other group used the guidelines. Two one-hour tutorials were followed by a one hour post experiment test that includes an evaluation and design task.

Participants:

Eleven second year undergraduate students participated in this study.

Results and interpretation

The researchers concluded that:

- ➔ The guidelines were easier to teach than patterns and also easier for students to comprehend and to remember.
- ➔ Patterns, being longer in format than guidelines, must be analyzed in greater detail before use.
- ➔ Patterns seem to require more careful and thoughtful teaching approaches.
- ➔ Links between patterns need further attention in order to be appreciated by students.

- ➔ The names of patterns and guidelines carry a significant weight.
- ➔ The examples presented with each pattern, or guideline, are probably the part that captured the attention and interests students. This was indicated by the majority of the students during the last teaching session. These examples help users to comprehend better the context and the intention of the pattern or guideline and provide an easy guide for the application.

Critique

This study critique focuses on three points. First, this was a pilot study and not a structured experiment. There is a lack of explicitly stated hypotheses and variables. Secondly, the number of subjects was very small. Thirdly, there was a clear lack of training, which could be considered rather minimal.

Pattern values

Pattern approach was not valued in this study.

[4] Koukouletsos et al. (2009)

This study assesses the effectiveness of patterns and guidelines as aids to teaching web interaction design. Two groups of students were recruited and taught web design from scratch using a widely used authoring tool. Each group learned about usability principles using either a set of patterns or a set of guidelines. The students were then engaged in two activities: designing and evaluating tasks. The evaluation of the students' designs was conducted by independent evaluator according to a predefined set of metrics.

Participants

45 final year students of the Automation Department, who had not previously studied web design, participated in this study. The students engaged in a course included more than 25 hours of lectures and seminars about web design, usability design principles and evaluation techniques.

Results and interpretation

An independent sample t-test (2-tailed) was conducted to test the hypothesis. The patterns group (M=128.97, SD=20.16) performed better than the guidelines group (M=116.25, SD=13.66), T (37)=2.317, p=0.0261. The null hypothesis was rejected because the P-value was less than 0.05.

Critique

The experiment was a well-designed study. This experiment has much strength: First, the experimental hypothesis was clearly stated. Secondly, the course where students were engaged was not part of any regular formal academic course. Therefore, students were not affected by concerns about marks. All participants had the same chance to learn about designing principles.

Pattern values

The experiment results indicate that the use of patterns can lead to a better performance for novice designers as compared to guidelines. Furthermore, patterns can have a strong impact on the students, provided they address issues close to the level of their experience.

[5] Kolfschoten et al. (2010)

This study asked the participants to design collaboration processes using a thinkLet library. They were divided into three experience categories based on the number of hours they got and their experience in facilitation. They participated in a full day workshop in which they had to design 3 collaboration processes based on a case description.

Participants

The participants were twelve undergraduate students.

Results and interpretation

Both the design quality and the time that was spent on each design were measured. The quality of the design was measured on a 1 to 10 scale by 2 experienced facilitators who were teachers of facilitation classes. The study highlighted the effects of design patterns on the cognitive load (the effort made by a person to understand and perform the task) of the design and modelling theory in terms of the different ways that experts and novices develop their designs. Interestingly, expert students complained that they had to find the design patterns that offered them the tools and methods they were used to apply. However, novices, in the end provided better models with the use of the design patterns than the experienced users. The study argued that, in addition to the benefits described by the previous research, there is a specific added value for the use of design patterns by novices to acquire design skills and domain knowledge.

Critique*:*

The main observations regarding this study are that it examines very interesting aspects of design patterns which is the cognitive learning efficiency in teaching.

On the other hand, the number of participants was small with a possible effect on the results.

Pattern values:

Design patterns affect the efficiency of the design effort of novices as well as their learning efficiency in gaining design skills and enhancing the quality of the design.

[6] Todd et al. (2009)

The purpose of this study was to investigate three issues: firstly, whether UI patterns are an acceptable medium for presenting information to students; secondly, what is the best way to present and organize UI pattern content to augment student's understanding of pattern content?; thirdly, whether a method designed to guide students in creating a UI patterns model aided student's understanding of UI patterns and patterns' language structure. The subjects in the experiment were introduced to the concept of UI patterns in a lecture. Then, they learned how to build a UI-pattern model. After that, they were divided into two groups and were asked to produce a UI-pattern model for two given interfaces. They were given two versions of patterns: an illustrated set and a narrative set. One group used the illustrated version for exercise one and narrative version for exercise two with the situation reversed for the other group.

Participants

The participants were fourteen students who were studying in a third year HCI course.

Results and interpretation

The data was collected through the following methods: observation, analysis of the solution to exercises, and questionnaire to investigate student's opinion on using patterns, pattern content, UI-pattern modelling and whether the patterns helped the discussion amongst students. Two types of observation data were collected: observation which was made by the researcher and digital photographs. Seven mini case-studies were created from the photographs to find how student pairs used the patterns and methods followed to generate their UI pattern models. Students were observed focusing attention on pattern content as there was no illustration to help them (35% of photos) compared to the illustrated set of patterns (25% of photos). The analysis of the student models was determined by the percentage of correct patterns and correct links for each exercises and type of patterns and revealed that over the two exercises, students improved their ability to correctly identify the patterns but not the links. Questionnaire results indicated that students found patterns to be informative and useful especially as an aid to communication. Furthermore, students preferred using the illustrated patterns.

Critique

The observation methods that were used by this study added a new dimension to the previous studies where student’s behaviours in using the patterns were recorded for analysis. The concern is that the number of subjects was very small.

Pattern values

Students found the information presented in patterns clear, informative and easy to understand. Patterns were also seen to focus students’ discussion about UI modelling. UI patterns are an acceptable medium to present information to students. Illustrated patterns were preferred over narrative patterns.

3.7.3 Review Summary

Eleven published experiments were reviewed that focused on using HCI patterns in education. Only six experiments are included in this paper. The other five were excluded because the participants were not students, and the findings therefore not relevant.

The conclusions are based on two things.

- The way these experiments were conducted
- Whether the findings of the experiments were summarized in terms of pattern value.

From the review, a number of issues emerge:

1- Most of the experiments were case studies or pilot studies and not controlled experiments; no hypotheses or variables were specified in the experimental design. Some papers did not interpret or analyze their findings formally.
2- Design task documentation that could have conveyed more information and understanding of the use of pattern was often not made available.
3- The findings of the study in some experiments, was based on students saying the patterns helped them and there was no actual evaluation measuring the effect of the patterns.
4- The participating students were sometimes not a homogenous group; some had had considerably more knowledge and experience than others. Furthermore, few document the background of their participants and therefore do not address the issue of expertise when assessing the usability and efficacy of the tested patterns for novices.
5- In some experiments, the experiment formed part of the teaching of a formal academic course. Therefore, students might have been affected by their concern about marks, and not been entirely frank in their responses to questionnaires.

Moreover, the critique turns to the findings of these experiments. Many researchers have commented on the steep learning curve for design patterns. The pattern values that were clearly addressed are:

1. Pattern structure and content were useful: students found that the information presented in patterns was clear, informative and easy to understand. Patterns were also seen to focus the students on UI modeling. Students consider the pattern format useful in formulating their own design experiences.
2. Design patterns affect the efficiency of the design effort of novices as well as their learning efficiency in gaining design skills and enhancing the quality of the design. Furthermore, patterns can have a strong impact on the students, provided they address issues close to the level of their experience.
3. Patterns were considered to be an efficient and effective aid for evaluation, redesign and new design.
4. Finally, HCI design patterns can be used to teach basic HCI design principles and can lead to above-average retention of design principles and to a quick adoption of the pattern vocabulary, even amongst first year undergraduates.

One can argue that more evidence that is empirical could usefully contribute to this debate since most findings stem from studies with few participants and need to be confirmed with larger studies.

So far, individual patterns from different authors have been studied and the problems, as highlighted below, will make patterns difficult to be used by novice learner or students. The following specific issues have emerged:

- Problem specification is vague or not easily matched [114, 168]
- Some patterns are complex and present more than one problem and multiple related solutions under one name.
- Context not described efficaciously.

Generally, educators using patterns must rely on kind of knowledge specifications, and possibly some sort of task analysis to deliver the basis of the patterns. Without an understanding of the learnt field from the learner's and educators perspective, patterns have no valid use and therefore, can only be applied in a haphazard fashion.

3.8 Patterns and the Cognitive Load Theory

According to Mayer [34], significant learning happens when learners engage in correct cognitive processing during learning such as mentally organizing relevant information into a coherent structure, and integrating representation with each other and with prior knowledge retrieved. It is possible to argue that pattern is one way of efficient organization of information.

To understand how patterns can contribute to learning efficiency, researchers need first to study the cognitive implications of the use of design patterns to transfer knowledge. Therefore, a general understanding of the cognitive mechanisms involved in learning is needed. Cognitive Load (CL) Theory is the "cognitive effort made by a person to understand and perform his task (mental load and mental effort)" [98]. As was discussed in section 2.3, educational and cognitive psychologists generally differentiate among a number of different types of knowledge, including facts, concepts, procedures, strategies, and beliefs [38, 56]. The question here is: what kind of knowledge do patterns present?

Kolfschoten et al. [194] explored the cognitive effect of offering knowledge in the shape of design patterns and its implications for learning efficiency. They analysed the design pattern concept in light of CL theory. The following summarizes their findings:

1. Design patterns assisted novices in gaining faster understanding in modeling and design skills, while more experienced users felt disturbed and disrupted by the design patterns.

2. The use of design patterns goes beyond the efficiency of the design efforts; it constitutes learning efficiency of novices to gain design skills and it enhances the quality of their design.

The discussion outlined here in the general aspects of learning will be encountered during the course of this research project as it is paramount to bridge the philosophy of learning and the implication of new techniques (such as patterns) utilized in enhancing the effectiveness of teaching. The next chapters will emphasize these aspects.

3.9 Chapter Summary

This chapter examined the patterns endeavour in SE, HCI and pedagogical, looking in particular at the structure of patterns and methods used to organize patterns. In addition, it highlights the ways that patterns can be used, and the values they embody.

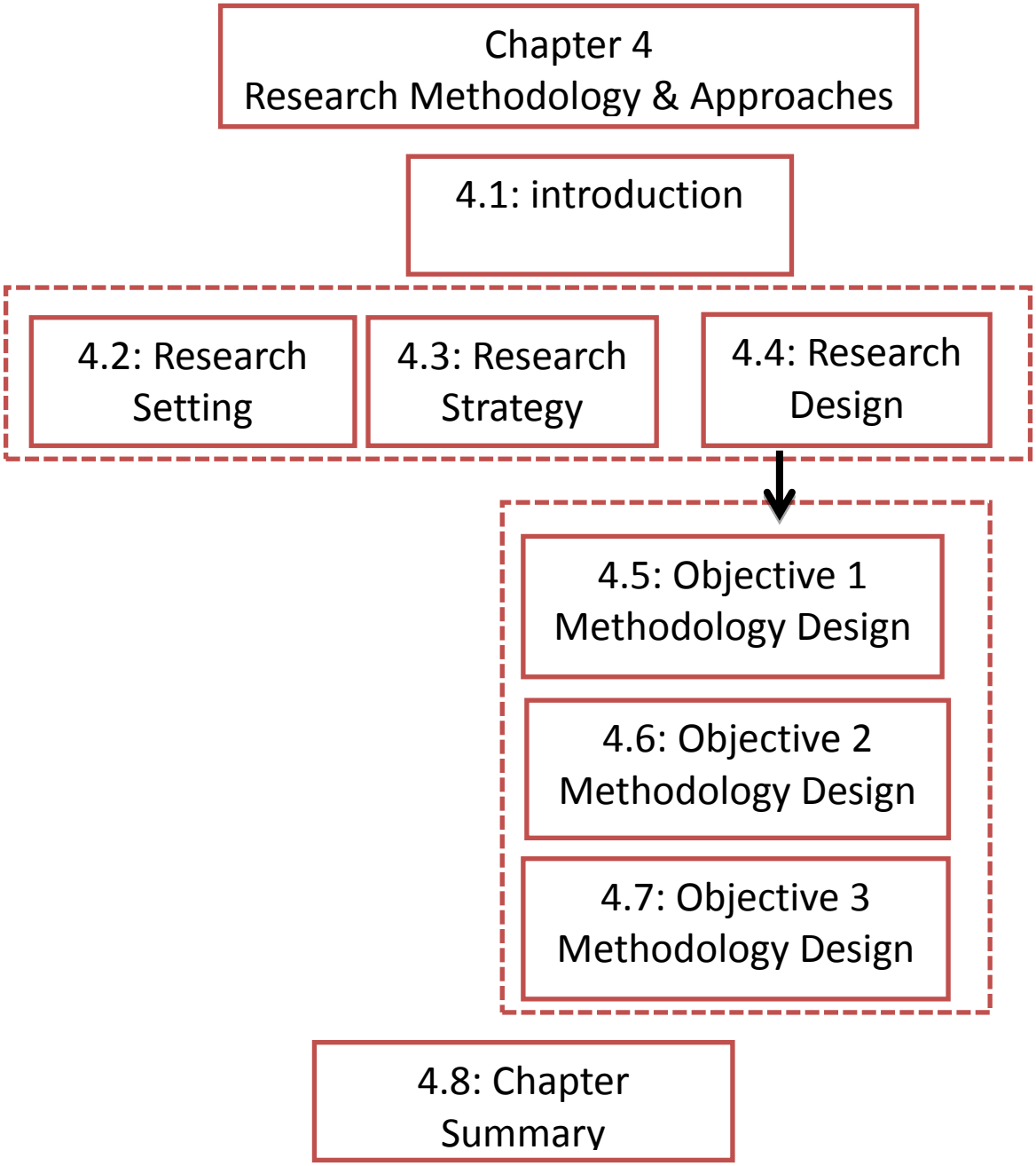
There is the lack of substantive evidence of patterns efficacy in education. Many agree that the studies that have been carried out have only examined simulated teaching activities rather than actual observation of "practical scenarios" which, in the end, may actually deliver different and unreliable outputs. Presenting this context, the researcher firmly believes that the research agenda for any patterns aiming to educate novices in a specific domain should be based on the following areas, namely:

- Investigating and improving the processes by which patterns are recognized, identified or discovered and recorded.
- Finding ways to organize, categorize, manage and maintain patterns and pattern language collections. The research needs to focus on finding the best technique to structure and organize the SQL patterns, so novices can easily understand and use them effectively. Hence, having a well-established technique that supports an intelligent management,

maintenance and retrieval of patterns will enhance and support their utilization in education.

- Evaluating the contribution that patterns and pattern language can make when used in education.

The next chapter, builds on the related research review, and provides insight into the structure of the research and the research framework. The researcher developed a combination of research methodologies (quantitative and qualitative).


Chapter 4
Research Methodology & Approaches
4.1: introduction
4.2: Research Setting
4.3: Research Strategy
4.4: Research Design
4.5: Objective 1 Methodology Design
4.6: Objective 2 Methodology Design
4.7: Objective 3 Methodology Design
4.8: Chapter Summary

Chapter 4: Research Methodology and Approach

This chapter provides insight into the structure of this research and its research framework. The researcher developed a combination of research methodologies (quantitative and qualitative) and used different analysis tools. In this way, the data were generated and analysed to inform the research questions of this study. The research framework is employed to prove thesis statement objectives:

1. **To identify SQL impediments that get in the way of learning performance.**
2. **To develop and design SQL patterns as informed by these research findings**.
3. **To assess the efficacy of the designed SQL patterns**.

4.1 Introduction

Computer Science education spans educational research, Computer Science (SC) research and other research areas such as Physiology. According to Almstrum *et al*. [196]:

> "Too much of the research in computing education ignores the hundreds of years of education, cognitive science, and learning sciences research that have gone before us". (pp. 191-192)

This research, as with all research designed in CS education, is in part based on investigation of principles in CSE [197], [198], [199], [200], [201], [202], [203], [204-206]. These studies focus on the different research areas in CS education

using different methodologies. In addition, cognitive science and linguistics research are considered as well.

In this chapter, section 4.2 describes the research setting by capturing the relevant parts of the research and showing how they relate or interact with each other. Section 4.3 explains the research strategy that offers the used plan to conduct the research and is followed by the research design in section 4.4. The first research objective is explored in section 4.5, which identifies SQL impediments that get in the way of learning performance. Section 4.6 explains the second objective of the research in which the different research methods were used to define structure, and manage SQL knowledge as SQL patterns. Section 4.7 presents the research methods that were used to evaluate the efficacy of SQL patterns as a teaching method, which is the third objective in the research. The chapter is summarized in section 4.8.

4.2 Research Setting

Research setting captures the relevant aspects of the research and how they interact with each other.

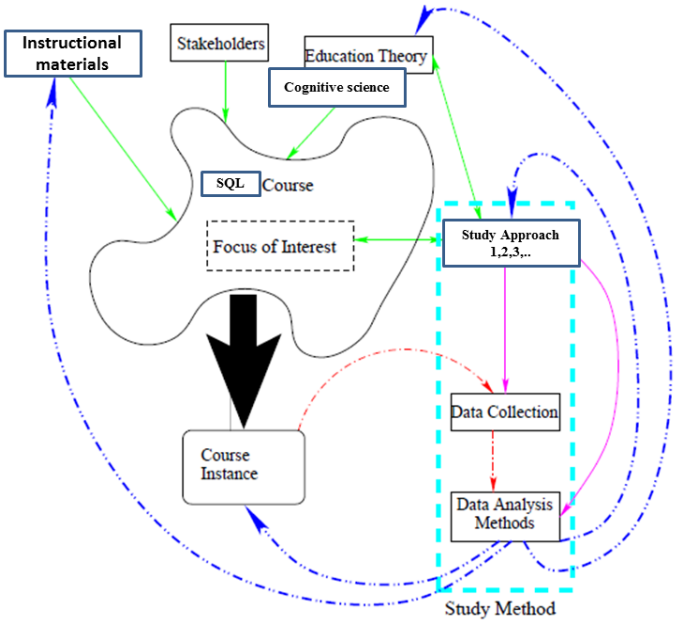


Figure 4.1: Applied Educational Research Adapted from Pears *et al.* [207]

Figure 4.1 represents the framework of the research which is adapted from Pears *et al*. [207]. During the investigation of any educational setting, focus of interest is an important aspect [207]. It explains what happens in the course context that the research aims to investigate.

The focus of interest in this research is the issues in teaching and learning SQL and how a new instructional material can be designed, developed and evaluated toward supporting the learning of SQL. Most studies and research in syntactic knowledge dimensions focus on novices, whose semantic knowledge is difficult to establish. Participants in this research are novices who are either currently studying SQL or have studied SQL earlier.

The next step is to frame the research general plan and to make sure that the research questions are addressed. The next section presents the research strategy and design.

4.3 Research Strategy

A "research strategy" offers a general plan for research. It ensures that research questions are answered using appropriate methodologies. In addition, it determines the type of findings that can result from the research.

To carry out this research, the researcher used different strategies at different stages of the research. The nature of the design of this research can be called a multi-strategy research [208],[209],[210],[211] where each method complements and builds on the strength of the other. The researcher applied a mixed method study [212] which attempts to bring data from qualitative and quantitative methods. A "research method" is the research instrument that is constructed to either guide or standardize data collection.

To achieve the objective of the research study, there are two different broad methodological approaches to select, which are: qualitative approach and quantitative approach. Throughout the design of this research, the researcher

considered the different aspects of a multi-strategy approach such as the sequence of the research methods data collection that were used either simultaneously or sequentially [213, 214] and the priority of the used method [214]. Grounded Theory is good for analysing data in exploratory studies; it was used to provide insight into the factors influencing learning. Grounded theory relies on the production of theoretical perspectives deriving from data. In this respect, the researcher focuses on the 'ground' – the data - and inductively generates more abstract concepts. Next section shows the research design.

4.4 Research Design

This research is based on the proposed framework for Computing Research Methods (CRM) [206] that is designed to facilitate teaching. It is grounded in four questions, which, collectively, describe the cycle of research. Each question anchors a quadrant in the process of computing research. The framework is illustrated in Figure 4.2.

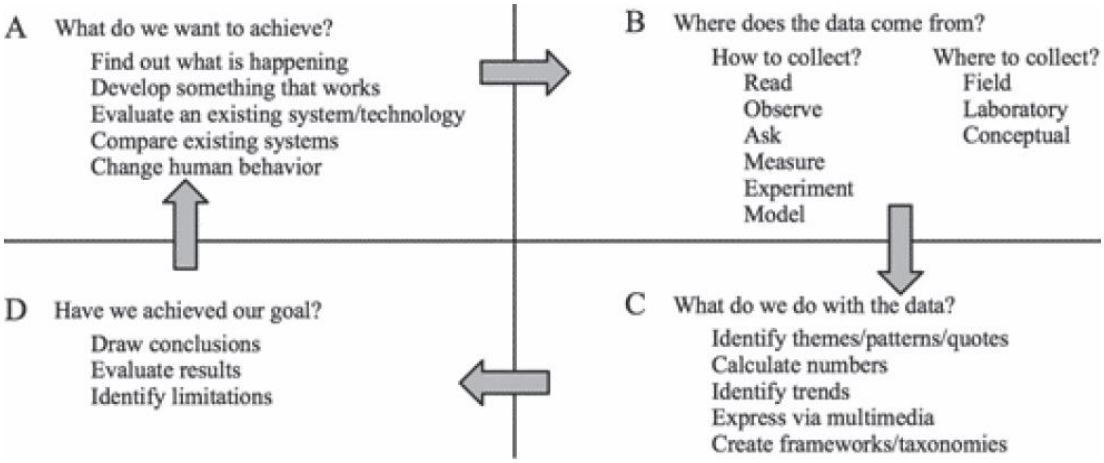


Figure 4.2: A Framework for CRM Designed to Facilitate Teaching [206]

This research is divided into three main research stages each answering one of three research questions. The design and the development is conducted into three objectives: (1) identify SQL impediments that get in the way of Learning Performance, (2) instructional design (SQL patterns design and development) process and (3) SQL patterns evaluation. The research is designed based on the

adaption of CRM framework. However, it was implemented in an iterative way that consists of three cycles as shown in Figure 4.3.

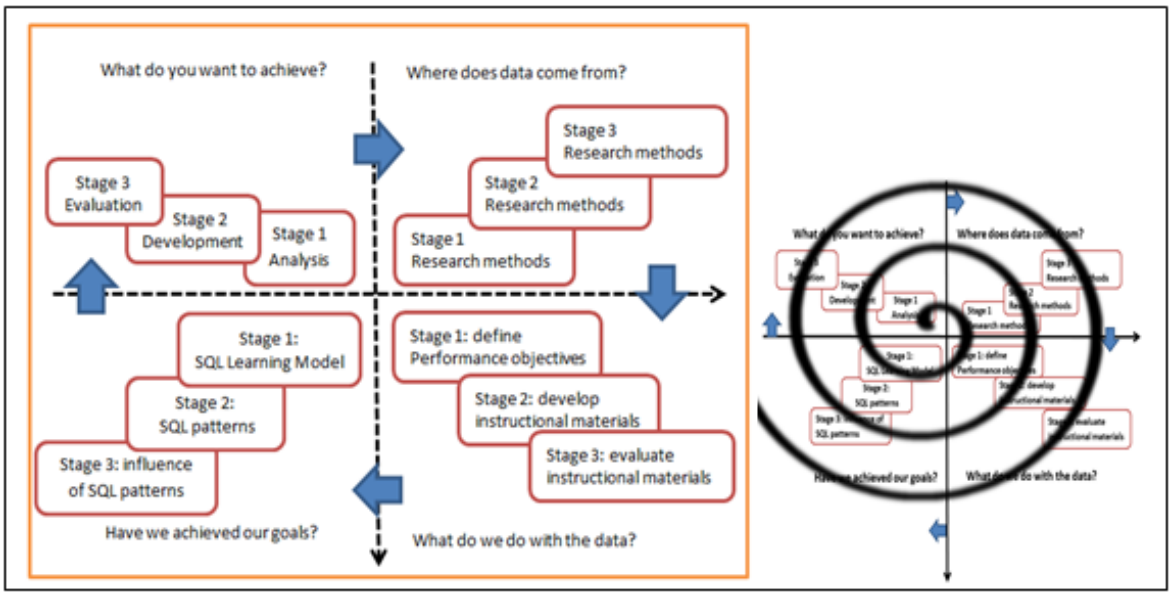


Figure 4.3: A Spiral CRM Framework for Research Methods Designed to Achieve the Intended Goals Adapted from [206]

Each cycle presents the research process in one stage through providing the answers for each of the four questions. The framework related questions for each stage are described throughout this chapter by answering the four dimensions in the above research framework:

- What do you want to achieve? This is done through the research questions in the next section.
- Where does data come from? This describes the instruments that are used to answer each research question.
- What do we do with the data? This explains the research procedures for each research objectives. The procedure is described in this chapter and more details are discussed in Chapters 5, 6 and 7.
- Have we achieved our goals? This highlights the results for each objective and the related discussion. Each research question results are reported in different chapters of this research; for example: RQ

1 results are reported in chapter 5, RQ 2 results are reported in chapter 6 and RQ 3 results are reported in chapter 7.

The results of each one affect the design and the implementation of the next. The next subsection presents an overview about each research objective.

4.4.1 Identify SQL impediments that get in the way of Learning Performance

At stage 1, the researcher aims to discuss and answer the research questions associated with learning SQL issues that are related to the learners and the SQL language.

This objective forms the first cycle of the research. It aims to explore the crosscutting factors that might influence entry-level undergraduate students' performance in learning SQL. This is done through employing different diagnostic tasks to explore novice's attitude and cognitive factors influencing their learning. An example is identifying the characteristic of students such as their previous knowledge and skills. Moreover, its purpose is to analyse the influence of the teaching and learning methods and approaches.

The main outcome of this objective will be a framework "model of SQL learning" which presents the performance objective. The purpose of it is to guide this study in designing an effective instructional material and for future work toward developing a matrix to assess learners' performance. The purpose of the second objective of this thesis is defined next.

4.4.2 Development of SQL patterns

This is built on the results reported in the previous section that proposed the framework in learning SQL. Its aim is to find the best element to design a new instructional material that considers the following aspects: the development of learners' and experts mental model throughout the learning process, learning

taxonomy which relates to SQL knowledge and skills, and the cross-cutting factors that influence the ease-of-learning of SQL.

In providing SQL instructional material, any research must address some critical aspects. Since the organization and representation of knowledge has the greatest impact on learning [63], then the knowledge should be structured within the instructional material. Moreover, learning happens in a predictable and mediated way, with subsequent knowledge and skills building on prior knowledge and understanding. Therefore, the sequence in which knowledge is presented is vital, so that it can indeed impact on the efficacy of learning. One of objective 2's goals is to find the optimal sequence for structuring and presenting SQL knowledge.

Moreover, it was found that integrating a cognitive model in the form of an expert user into novice instruction enhances learning [13].This is essentially the rationale for patterns of any field. At this stage of the research, it is aimed to apply pattern concepts in structuring and organizing SQL knowledge based on the conducted literature review and empirical tasks. This focuses on SQL patterns' identification, structure, and organization. This study was carried over the second year of the research to focus on patterns in general and SQL patterns in specific. It is divided into four processes:

- SQL patterns' identification
- SQL patterns' structure
- SQL patterns' organization
- Evaluation of the successful contribution of SQL patterns' collections and pattern languages in education

More emphasise was given to the evaluation of the influence of SQL patterns in objective 3 as is discussed next.

4.4.3 Assess the efficacy of the designed SQL patterns

Patterns are a widely accepted mechanism for supporting knowledge representation and transfer. This research's purpose is to set out an investigation on whether structuring SQL knowledge in patterns could meet the need for optimally-structured instructional material.

This is the stage where an evaluation of the effects of using SQL pattern approach in education is conducted on the use of SQL patterns in teaching SQL. The results of this are discussed in chapter 7.

The next section describes the research framework at Research objective 1: Identify SQL impediments that get in the way of Learning Performance.

4.5 Objective 1: Methodology Design

The purpose of this research is to identify the probable reason for a performance gap by examining the cognitive aspects, learning activities, and cross-cutting factors that affect learners' performance in SQL acquisition.

4.5.1 Research Questions

The research questions (see Table 4.1) investigate the factors that might influence the entry-level undergraduate students' performance in learning SQL. The first set of questions (Research Question1) is related to human factors. It is categorized under the characteristics of novice SQL learners. A number of factors in learning influence SQL novices, such as: their personal attitude, previous experience, problem solving skills, and acquisition abilities.

Many researchers studied the relationship between students' performance and personal attitude towards learning [215]. In mathematics, for example, [216-219] looked at the students' performance and their personal attitude. The

influence of the negative attitude among novices who are learning programming languages has been highlighted as well [220, 221].

The Analysis of SQL learning performance objectives		
Research Questions 1.1	What are the effects of the following novice SQL learner characteristics?	
	1	Personal attitude toward learning SQL
	2	Previous knowledge and experience
	3	Problem solving skills
Research Questions1.2	What are the effects of the following aspects of SQL language?	
	1	The declarative nature of SQL
	2	The syntax of SQL
	3	The content of SQL
Research Questions 1.3	What is the impact of the current teaching methods and approaches in the following aspects of learning SQL?	
	1	Novices' ability in reading and comprehension of SQL queries (query comprehension)
	2	Novices' ability to understand the given scenario (query formulation)
	3	Novices' ability to translate the given problem (query translation)
	4	Novices' ability to write non-trivial query (query writing), which is the application of their knowledge

Table 4.1: Research Questions Align with RQ 1

SQL language features, on the other hand, were considered as other factors, such as SQL nature, SQL syntax and SQL concepts. Research question 1.2, aims to investigate the factors related to learnt language features.

SQL is a non-procedural language; it merely states "what", not "how" [222]. Looking at the nature of SQL and comparing it to the way Computer Science students are taught and learn using "How"; one could argue that this might have some implication when learning SQL. Research question 1.3 examines the following cognitive factors:

- Students' abilities in reading and comprehension of SQL queries (query comprehension), and

- Students' skill to understand and analyse the given scenario (query formulation and translation),
- Students' talent to write non-trivial query (query writing)

The next section presents the related task definition and measurement.

4.5.2 Tasks Definitions and Measurement

A view about teaching and learning in SQL and the problems encountered within the teaching and learning approaches requires some understanding of the way in which students approach SQL. Thus, it is important to clearly understand both the aspects and the activities that are required in learning SQL.

Many researchers attempted to determine the factors that affect SQL learning and use, as was discussed in chapter 2 [4-6].The impact of query language features, on the other hand, was investigated in terms of learning and using the language [4, 5, 7, 81, 88, 223].The effect of the method of teaching a query language was also studied by [224].

In the related Computer Science Education research, exploring the factors that might predict success in introductory courses, such as programing language, is reported in many studies. The factors suggested in the literature include mathematical background and previous experience [225], logical reasoning ability and previous academic background [226], and learner attitude and academic motivation [227, 228]. To the best of the researcher's knowledge, there is no such research conducted in term of SQL learnability. The focus of the research methods used here is to measure the following:

- The influence of the characteristics of novice SQL learners. Novices understanding might be influenced by their: personal attitude, previous experience, problem solving skills, and acquisition abilities.

- The impact of the different features of SQL language, such as: SQL nature, SQL syntax and SQL concepts and knowledge.
- The effect of SQL teaching and learning methods and approaches and the related issues encountered. The cognitive factors that were evaluated are: students' ability to understand and analyse the given scenario (query formulation and translation), students' skills in reading and comprehension of SQL queries (query comprehension) and students' ability to write non-trivial query (query writing) which is the application of SQL syntax and semantic knowledge.

To measure the above, several instruments were employed. The next section describes the different research methods used at this stage.

4.5.3 Tasks Development

Different tasks were developed to answer research question 1.1. Figure 4.4 explains the sequence of the research methods used. The interviews and the cognitive tasks are used in the beginning, and they are followed by an online questionnaire. The questionnaire is used to explore the findings from the interview and the cognitive tasks to obtain more feedback from the educators. The student' questionnaire was conducted later to emphasize the highlighted issues by educators. The task analysis aimed to evaluate learners' skills in SQL comprehension.

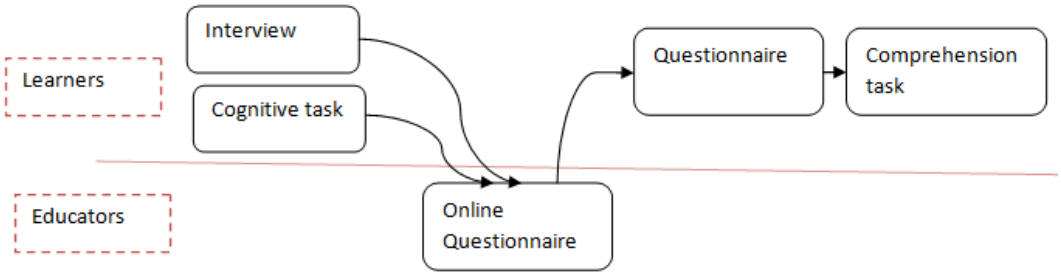


Figure 4.4: Sequence of the Research Methods (Research Question 1)

Different instruments were used to collect data related to the teaching and learning of SQL, as illustrated in Table 4.2.

	Method	Participants	Aims
1	Semi-structured Interview	7 Students	To understand the problems experienced in learning SQL
2	Cognitive task	7 students	To investigate students ability to explain in English how to solve query (Query translation) and write the related SQL (Query writing)
3	Questionnaire	75 students	To evaluate difficulties in learning SQL from learners' perspective (informed by 1)
4	Comprehension Task	64 students	Cognitive task focusing on student's ability in comprehending SQL.
5	Online questionnaire	14 teachers	To evaluate difficulties in teaching SQL from the educator perspective

Table 4.2: Research Methods used for Research Question 1

The next subsections describe in depth the used research method.

4.5.3.1 Semi-structured Interview

Qualitative accounts and opinions are essential in understanding the participants' human factors such as attitude, which is often necessary in social science and educational research [229, 230].

The semi-structured interview aims to give an overview about learners' attitudes toward learning and using SQL, and to identify the most difficult concepts they faced during their study. The semi-structured interview was further designed to provide in-depth information pertaining to participants' experiences and viewpoints of SQL nature, syntax, and content. The design was based on [231] guidelines. Interview protocols include a series of open-ended questions to foster discussion about the highlighted areas of the research and direct the interviewees towards the phenomenon. Therefore, data obtained can be broad enough to provide meaningful responses in relation to the research objectives without obliging a certain format, or way of responding upon the participant. Each interview consisted of questions concerned with the participants'

experiences of SQL learning and the different ways they view SQL nature, syntax and content. Each question serves as an 'opening' from which the interviewer develops a trail of further questions in order to achieve a mutual understanding of the target area. The semi-structured interviews that were conducted are based on two main areas, as shown in Table 4.3.

1	*Examine the relation between the personal feeling and the SQL knowledge and experience*
2	Rate the student's knowledge of different SQL basic concepts

Table 4.3: Main Components of Semi-Structured Interview

The first part examines the relation between the personal feelings and the SQL knowledge and experience using the following questions:

- How does writing SQL make you feel?
- How do you find learning SQL compared with learning other languages such as a programming language?
- How many courses have you taken that includes learning SQL?
- How skilled do you think you are in solving SQL problems?
- Do you have any work experience with SQL?
- What are the most difficult concepts in SQL that you find difficult to understand or apply?

The second part rates the student's knowledge of different SQL basic concepts where the researcher lists some SQL concepts and participants rate its difficulties using Likert-scale. The qualitative responses are supported by verbatim quotes from the interviews and text analysis. The Semi-structured interview question sheet can be found in Appendix A and the collected data are reported in section 5.2.1. The next subsection describes the cognitive task that is used to evaluate students' skills in problem solving. It is based on the SQL problem solving cognitive model that was discussed in section 2.3.2.

4.5.3.2 Comprehension Task: The Ability of Reading and Understanding SQL Statements and Comprehension Skills Task

The purpose of the task analysis is to investigate if students are able to read SQL statements and print the derived output from the given data. The task involved SQL query and the related Entity Relational Diagram (ERD), where the participants were asked to walk through the SQL command and explain what the SQL command is intended to perform. In addition, other information was gathered about the participants, such as their previous knowledge and experience in SQL by stating the number of courses they had in SQL. They were also asked to rate themselves. The responses were measured based on respondents' feedback on a set of 5-option Likert scales as 1 "Expert" to 5 "Not skilled". The task is shown in appendix E and data is presented in section 5.2.5.

4.5.3.3 Cognitive Task: Query Formulation, Translation and Writing

The aim of this task is to answer the following question: To what extent can students understand the given SQL problem and express how to solve it by applying their knowledge and skills?

The task investigates novices' ability to solve SQL problem by investigating their skills in query formulation and query translation that were discussed in section 2.3.2. Therefore, it is possible to assess their understanding of the given SQL problems. In addition, the task aims to evaluate their ability to write the related queries correctly.

The participants were seven students (two Computer Science third-year students, one Honour's Computer Science student, three Masters students and one PhD student) who were the same participants at the interview conducted prior to this task.

The participants were given the questions, as shown in Figure 4.5 below, and were asked to translate the problems provided in natural language and to solve

it by deciding what elements of the data model are relevant, and the necessary SQL concepts and operations to be applied. Then, they were asked to write the related SQL query. The task is shown in appendix B and data is presented in section 5.2.3.

Find the names and the hire dates for all employees who were hired before their managers, along with their manager's name and hire dates. Sort by employee name Note: all information is stored in table: Employee.

Figure 4.5: Question Used as Part of Task Analysis

The next section describes the method used to collect teachers' feedback about the SQL learnability in general. In addition, teachers were asked to reflect on the results collected from the previous research methods, which are the interview and the cognitive task.

4.5.3.4 Online Questionnaire - Teaching and Learning SQL (from teacher's point of view)

The online questionnaire method focused into a few areas: SQL features (such as SQL nature, SQL syntax and SQL content), and investigating the common difficulties in SQL concepts and the nature of the process in learning SQL from an educator's point of view. In addition, it reflected on students' responses to the tasks from the previous research methods, which examined the participants' ability in solving SQL problems.

The questionnaire was sent by email to several teachers who were either currently teaching SQL or had done prior research in teaching SQL. This instrument contained Likert-scale and open-ended items. The online-questionnaire can be found in Appendix C and data in section 5.2.3.

4.5.3.5 Students' Questionnaire: Investigating Key Issues in Learning SQL (from student' point of view)

The questionnaire's aim was to collect data from SQL learners who had done at least one course in SQL. It was designed based on different research on questionnaire design [232]. It focused in two main areas:

- Learners' view of different aspects of SQL (such as SQL nature, SQL syntax, and SQL content and the common difficulties in SQL concepts); and
- Learners' approaches, perception, misconception, and feeling in learning and applying SQL concepts.

Seventy-five students participated in this study. This research method was informed and consequent to the first two methods. The questionnaire is shown in appendix D and data is presented in section 5.2.4. The questionnaire was designed to investigate three questions, see Table 4.4 below.

	Question	Measurements
1	What are the most difficult concepts or most challenging in learning SQL?	Content analysis
2	Why are many students having problems in learning SQL?	Content analysis
3	How many students agree with the following statements? - Students solve SQL problem by trial and error - Students can easily read SQL statement - SQL syntax is easy to learn - Students can write only simple SQL statements - Students do not have problems in solving large complex queries - Students can join more than three tables and retrieve the required information - Students do not have problems with self-join table - Students do not have problems with using aggregated functions - Students do not have problems with group by clause - SQL is easy to use compared with other programming language	Likert-scale for 1 to 5

Table 4.4: Student's Questionnaire

Code reading or walkthrough, on the other hand, are important skills to novices in program learning [233]. Query comprehension cognitive task that involve query reading, query explanation and printing out the results is discussed in the next section.

4.5.4 Results Analysis

Various tools and methods were used to analyse the results. SPSS software was used to analyse data from both questionnaires. All these results are discussed in chapter 5. Grounded Theory was used to understand the factors influencing the success in learning and teaching SQL because of the complexity and range of issues amongst a group of participants who had similar problems in teaching and

learning of SQL, as well as the fact that this topic is an under-researched field of study.

The data analysis took place during the data collection period, and was thoroughly integrated with all aspects of it, including an analysis of every interview, questionnaires, and observation directly after they were given. In this way, each step of the data collection could feed into the analysis. It consisted of three strands that utilized mixed methods, and these were triangulated for the sake of rigour; balanced out the things students said during interviews and did in the cognitive task and either confirmed or contradicted with educators' viewpoints in the online questionnaire.

Results of the literature conducted in chapter 2 and the research method employed here are analysed towards determining the "SQL impediments that get in the way of learning performance" the outcome of research question 1. This analysis of the objective is aimed to define what knowledge and procedure need to be included in designing instructional materials. The objectives identified as a results of this should provide a map for designing the instruction and for developing the means to assess learner performance [35] which is Research question 2 aims. The next section presents the design of the development of SQL instructional materials, objective 2 of this research.

4.6 Objective 2: Methodology Design

This research purpose is to find the best way to structure, organize and evaluate SQL patterns in order to improve novices' performance in learning SQL by using them. It starts with research methods that aim to determine SQL knowledge. Then, other research methods are used to structure and organize the identified SQL knowledge. Patterns concepts and related research are employed to structure SQL knowledge and hence called SQL patterns.

4.6.1 Research Questions

The list of questions which are related to research question 2 (see Table 4.5) focused on the SQL knowledge identification, structuring, and organization. The questions are adapted from prior research in the area of instructional design [234],[63], [34], designing training and teaching materials for database query language [13], [235] and [236].

Research questions: SQL patterns design and development process	
1	How should SQL patterns be defined and what should they contain?
2	How should SQL patterns be identified?
3	How should SQL patterns be structured?
4	How can SQL patterns be organized?

Table 4.5: Research Questions Align with RQ 2– SQL Patterns Design Process

To answer the related questions, the areas of patterns identification methods (as discussed earlier in Chapter 3) are employed along with results documented from Research objective 1 outcome. The next section describes the process of the task development.

4.6.2 Tasks Definitions and Measurement

This research involves embarking on different research and methodology in order to design a new instructional material and to obtain more empirical evidence of its efficacy in learning SQL on the following areas, namely:

- Investigating and documenting the processes by which SQL knowledge are recognized, identified or discovered and recorded. This is done through employing cognitive psychology, literature and patterns design techniques.
- Evaluating different options to structure the individual SQL concepts.

- Finding ways to organize, categorize, manage, and maintain SQL knowledge. This is done by adapting some strategies aimed to facilitate judgment and simplify problem solving in complex queries [33]. For example checklist and component-level design
- Applying different evaluation methods in order to assess the usability of the proposed material.

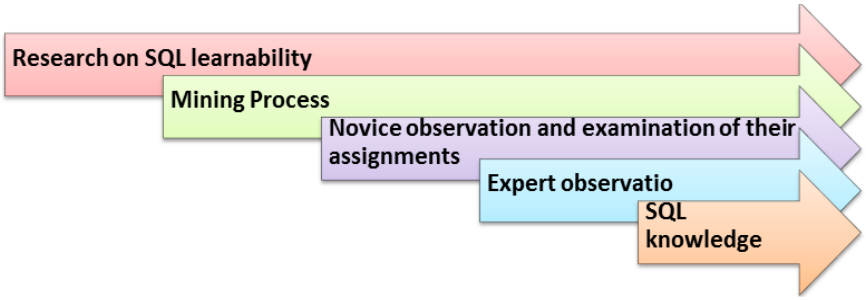


Figure 4.6: SQL Knowledge Identification Process

Different tasks (see Figure 4.6) that the researcher followed to collect and identify SQL knowledge:

- Defining SQL learning objectives (objective 1 outcome).
- Text mining process.
- Observing novices solving SQL query during in the lab to investigate the learner's cognitive steps during query solving [13].
- Collecting and examining examples and of SQL queries from students' submitted assignments.
- Conducting a cognitive task to investigate the experts' cognitive steps during query solving [13].

Then, other methods (see Figure 4.7) are used to structure and organize the knowledge as SQL patterns which aimed to answer the third and fourth questions:

- Conducting relevant literature on: patterns and patterns languages design in Architecture, HCI, SE and pedagogical patterns. In

addition literature on instructional design in education [113, 125, 129, 130, 172, 173, 237-240]. This was discussed in chapter 3.

- Sending a set of pattern to EuroPLOP for shepherding process and it was discussed in the related workshop as well.

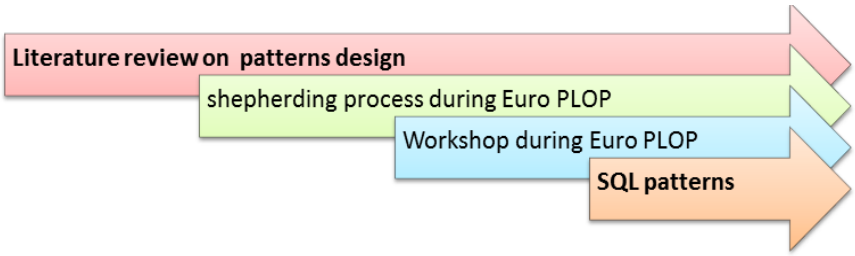


Figure 4.7: SQL Patterns Design Process

Literature review and critical reflection has been conducted on the use of patterns in Computer Science Education, for more details see section 3.7. Based on this review, the researcher modifies the design method of the initial set of the patterns and the evaluation method of the patterns. Then, this set was improved through the employment of different research methods (see Figure 4.7). More details are discussed in chapter 6. The next section presents the task definition and its related measurement.

4.6.3 Task Developments

The task was developed using a spiral model that consists of four main processes: identification, structuring, organization and evaluation distributed into three phases. This is illustrated in Figure 4.8 and Table 4.6. These questions were answered iteratively throughout different periods of this research.

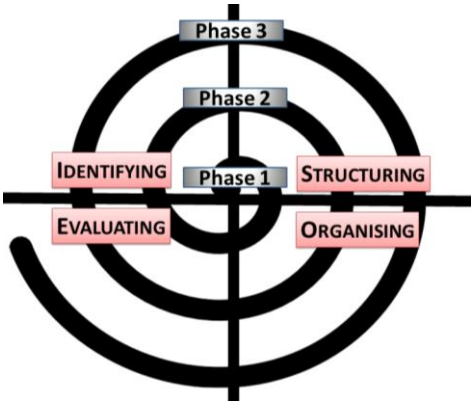


Figure 4.8: SQL Pattern's Design Phases

Different methods are used to design SQL instructional materials. The analysis of data gathered during qualitative and quantitative studies of SQL acquisition in achieving objective 1 guides objective 2 designs and development. Different instruments were used to collect data related to the design of instructional material for SQL learning (see Table 4.6).

Process	Method		Participants	Aims
Phase-1				
Identification	SQL Learning model Text mining			Collecting examples and knowledge in SQL text books
Structuring	Literature on problem based approach and instructional design			following the process students use in solving the query
Organization	Literature on Checklist			Matching the given problem to a set of patterns.
Evaluation	Case study		3 PhD students	Evaluate the use of SQL patterns in the process of solving a complex query.
	Interview		3 students	Reflect learners' point of view on the usability of the patterns.
	Questionnaire		5 academic	Reflect educators' point of view on the usability of the patterns.
Phase-2				
Identification	Novice observation		IM2 and DB3 students	To find out how students approach SQL.
	Content analysis		IM2 and DB3 students	Evaluate students assignment and analyse the errors
Structuring	PLOP	Interview	2 pattern writers'	shepherding process during Euro PLOP

		Focus group	5 pattern Writers	Workshop during Euro PLOP
Organization	Component level design			To find out the best to increase learners performance through …
Evaluation	Interview		10 students	Collect novice feedback
Phase-3				
Identification	Expert observation		2 expert students	To find out how expert students approach SQL.
Structuring	Previous phase 2&1			checklist, component level design
Organization	Same as phase 2			
Evaluation	Experiments, questionnaire		90 students	efficacy of SQL patterns on novice performance and satisfaction

Table 4.6: Research Methods Used for Research Question 2

The following subsection explains the design of the process of identifying and defining the patterns using text mining, observation of novices, and observation of experts.

4.6.3.1 Problem Solving Strategy Identification via Mining

SQL knowledge was identified through text data mining or knowledge discovery process. Mining concepts is the method used to discover knowledge from existing data available, solutions, or designs. According to Tan [241] text mining is:

> "The process of extracting interesting and non-trivial patterns or knowledge from text documents". (p. 65)

This method involves a review of database texts used to teach SQL. Thus, it is possible to identify common knowledge that relate to the core of SQL concepts.

To do that, the first decision was on the list of books that might be used. It was decided to use database textbooks which are available to the researcher. The text mining process is mainly based on natural language processing techniques, including text analysis, text categorization, information extraction, and summarization. The following steps were followed:

- Collect a set of database textbooks that are used to teach SQL concepts and are available in the university library.

- Identify the SQL misconceptions from both literature review and empirical research (chapter 2 and 5) and limit the text mining to those concepts.
- Analyze the text and search for SQL-relevant knowledge.
- Identify declarative knowledge from database texts and categorize the knowledge as follow:
    - SQL concepts definition and syntax "what"
    - SQL concepts application purpose "Why" and "When"
    - SQL concepts application method "How"
- Extract the information from the text and structure it into the following form
    - Highlight the "Problem" or "what" SQL concept.
    - Identify the related "Context" in which SQL concept Problem is likely to occur. In addition, determine the concern or the forces that make such a problem difficult to solve.
    - Find the "Solution" to the identified "Problem": how the concepts should be applied, relevant syntax, and rules.
    - Illustrate the solution with appropriate examples, which shows step-by-step how such a solution could be applied.
    - Highlight the impact of applying such a "Solution" to the "Problem" in the identified "Context."

The process of text mining provides an initial stage, delivering only a static understanding of how SQL pattern knowledge is presented in textbooks. The actual process by which SQL concepts are applied cannot be predicted without empirical evidence. Therefore, it is important to identify such knowledge through another approach such as observing and analysing students' work in applying SQL. To enhance the observation, research on problem solving strategies is conducted at phase 1. The next section discusses: observation of

real novices solving SQL queries during labs and Examining examples and samples of SQL queries from students' submitted assignments.

4.6.3.2 Problem Solving Strategy Identification through Observation

Researchers in the field of pattern identification agree that patterns ought to be identified with reference to design solutions through observation, rather than being constructed from theory. Therefore, cognitive aspects need to be taken into consideration. Instruction methods that apply what educators know about how students learn, remember, and use related skills can make the learnt subject meaningful and help students to perform better [62]. To achieve that, cognitive science suggested giving learners a problem and observing everything they do and say while attempting the solution. The cognitive task aims to formulate the process of SQL problem solving strategy. Thus, it consequent SQL knowledge identification through this kind of cognitive task or observation.

Time	Participants	Number
2009/10	Students registered in Information Management (IM2) course	17
2010/11	Students registered in Information Management (IM2) course	21
2010/11	Students registered Database (DB3) course	15

Table 4.7: Time Spent with Novice SQL Learners

Strategy identification by means of learner observation helps determine how learners apply such knowledge. Unstructured observations were conducted on a period of two semesters (see Table 4.7).

The process of SQL strategy observation and subsequent pattern refinement was important to understand how novices solved SQL problems; i.e. the steps followed to arrive at a solution to the given problem. These include:

- Remembering:
    - When they remembered the required knowledge, was it correct?

- Searching (Not Remembering):
    - How was the unremembered but required knowledge obtained? For example, did they refer to textbooks or teaching materials? or did they search the net to find similar problems and related solutions?
- Problem Solving:
    - Was the required knowledge identified correctly?
    - Was the knowledge correctly matched to the given problem context?
    - Did they search for visual examples on the Web?
    - Did they try different solutions? If so, why was a particular solution selected?
    - How did they react to their errors?

Different questions designed to direct the unstructured observation shown in Table 4.8 to find out participants strategy in solving the given tasks.

Question	**Aim**
How do students start solving the given task? Are there any initial questions about the context of the question?	Explaining how queries might be solved. To illustrate the steps learner followed in solving the given task.
What are the methods students use to get the required knowledge for solving the question?	Determining resources used to gather the required information.
General behaviour during problem solving	
What kind of questions students ask during problem solving? such as: SQL content "What" questions, application of SQL structure "How" questions or if there are any other high-level question about "when" and "why".	Studying learner decision in the applied solution.
What are the frequencies of the questions students ask? Are there any common misunderstandings or confusions in the task?	Do the available knowledge need to support by data models to enhance learners' understanding. Does available knowledge need to support by visual examples to enhance learners' understanding.

Table 4.8: The Questions Used to Direct the Unstructured Observation.

Content analysis has been used as a method for analysing messages and communication that participants have been asked to produce. The results of this method are reported in Section 6.3.2. The next sections describe SQL knowledge identification through expert observation.

4.6.3.3 Problem Solving Strategy identification Through Expert Observation

This section describes how experts use their knowledge to solve problems. Moreover, it discusses the related cognitive activity that they perform during problem solving through employing a "loud-talk" protocol. This made it possible to identify gaps in the novice knowledge since it supported comparison.

The experiment was run on personal computer to oversee each subject's approach, using a tool called SQL Pattern Based (SQLPB) that was developed by the researcher using Netbean platform. All the information about SQLPB is discussed in section 4.7. Additionally, Camtasia studio4 was used to record all participants' action in the screen and record all their explanation. All participants' trials and errors were recorded as well. Two participants were given a task (see Appendix F) to perform. They were MSc students at University of Glasgow. The observed experts had a long working experience of SQL. The task involved two questions as shown in Figure 4.9.

Q1: Give the titles of books that have more than one author. Q2: Display the names of borrowers who have never returned a book late

Figure 4.9: Expert Observation Task

All the related tables were available from the SQLPB tool. They were asked to write the SQL query that would help them to solve the given problem. The collected data were analysed using protocol analysis. The findings of observation are often difficult to interpret, because it is not clear why the participants' are behaving as they are. The collected data were analysed using content analysis. The data and result of this method is reported in section 6.3.3.

4.6.3.4 SQL Knowledge Structuring Design Methods

SQL patterns are knowledge and skills that exist in the expert's mind and are continuously applied in related scenarios. They need to be formulated in a structured way by either the experts themselves or by others in the same field. Once these knowledge and skills have been documented and approved by the experts (may be called pre-patterns at this stage) then they must be given to different users to try. If different users accept these pre-patterns, then they can be called patterns and can be published. SQL patterns are aimed to facilitate learner's knowledge and hence improve their performance. SQL patterns' identification and structure requires some specific knowledge in educational instructional design research. In addition, knowledge of how the patterns are structured in other fields would support this quest. Chapter 3 presented this literature review in patterns' structure in Architecture, SE and HCI.

The results reported in the development of section 4.6.4 of stage 1 guided this research to draw the outline of how SQL knowledge and skills might be delivered to learners.

The analysis of observation activities made it clear that instructional materials, such as their notes, did not guide students towards productive activities or to support effective problem solving. To help novices to achieve this level of expertise, the research proposes that the SQL patterns should be designed to:

- Highlight both the basic knowledge required to solve the problem and the advanced knowledge.
- Provide step-by-step SQL visual examples of the SQL being applied.
- Help in understanding the context of the problem. This depends on learners' previous schemata. Here, we tried to find out how such knowledge can be delivered.
- Support matching a problem to a solution in a simple format such as a checklist.

- The impact of applied concepts in such a problem context; for example, the reasons behind the chosen approach.

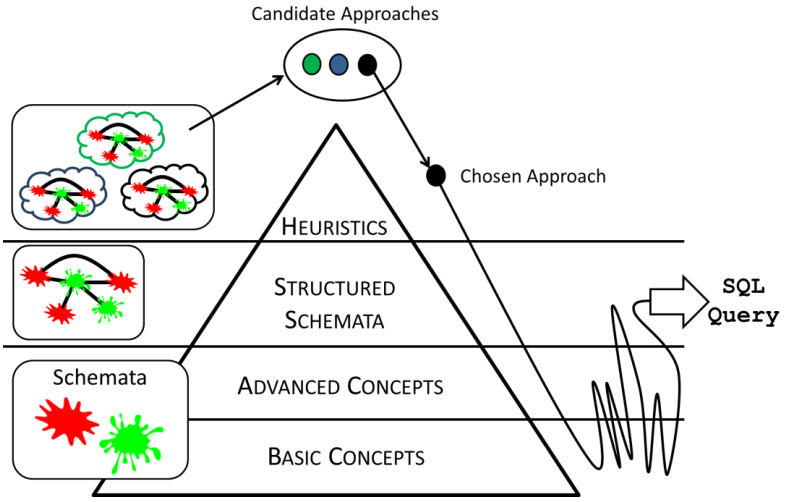


Figure 4.10: Instructional Materials Elements.

Figure 4.10 shows what kind of concepts or knowledge need to be available and how such elements interact with each other and with learner schemata. More details of the level of knowledge presented in this Figure 4.10 are given in section 6.3.2. The next section presents the methods of patterns structure and organization.

4.6.3.5 SQL Instructional Materials Organization Methods

The aim of this part of the study is to propose an approach for the management of designed materials viz SQL pattern collections. The goals are to support novices in two different tasks: a) the selection of the correct pattern from the collection; and b) the understanding of the relationship between patterns in the collection.

The pattern, within Alexander's [125] pattern language, are hierarchically connected to one another, in the way that higher level patterns are made up of lower level patterns, and these relationships are made explicit within the patterns. Many researchers highlighted the importance of organizing patterns and suggested one or more organizing principles. According to Salingaros [170]

> "A loose collection of patterns is not a system, because it lacks connections" (p.154)

Chapter 3 elaborated on different patterns' collection in Architecture, Software and HCI and their techniques in organizing and structuring patterns in patterns language. During SQL patterns design, the finding from the literature in patterns' organization was analysed and tested in terms of the applicability to SQL patterns. In stage 1, it was decided to use a checklist approach to relate SQL query to the related patterns which is a new approach in bridging the SQL problems and SQL patterns. Thus, novices could select the correct set of patterns.

Scriven [242] described checklist as a list of factors, properties, aspects, components, criteria, tasks, or dimensions, the presence, referent, or amount of which are to be considered separately, in order to perform a certain task. After the evaluation phase in stage 1, the researcher studied other possible techniques in linking the patterns. Scaffolding techniques were taken into consideration as well.

Here, solving a query problem might require the application of more than one pattern. The collection of SQL patterns was inspired by Alexander's [125] approach. Alexander's pattern language is hierarchically built. Each pattern is connected to one another: higher level patterns are made up of lower level patterns, and these relationships are explicit within the patterns.

It was believed that using an approach that students were more familiar with might lead to a better understanding. Therefore, Component-level design approach was employed to present the graphical representation of level of patterns to understand the relationship between the given problem and the checklist, the checklist and the patterns and the relation between patterns in the collection. Modelling component-level design were applied in software engineering to translate the design model into operational software [243]. More details are given in section 6.5.2.

4.6.3.6 Evaluation research Methods

SQL patterns were subjected to various evaluations throughout the identification, design, and usability stages. Instructional objectives were written up prior to the design of 5 patterns. According to Dick and Carey's [104] recommendations on instructional design, SQL patterns underwent a number of evaluations while in the developmental stages. These evaluations were used to "obtain data that [could] be used to revise [the] SQL patterns to make them more efficient and effective" [104]. These developmental evaluations consisted of an aesthetics and usability evaluation, subject matter expert (SME) evaluations and one-to-one evaluations. After completion of the SQL patterns' structuring and organization, they were also field tested to determine the effectiveness of the SQL patterns that were explored, as discussed later in chapter 7.

4.6.4 Results Analysis

Stage 2 data analysis, as subsequent to the previous step and as indicated in Figure 4.10 above, consists of classifying the collected data under instructional design phases of the following four processes: identification, structuring, organization, and evaluation of SQL knowledge. Chapter 6 and chapter 7 report all the data collection and results analysis of these four processes. Research question 2 findings agree on structuring SQL knowledge as SQL patterns and employing the later as instructional material to help novice master SQL skills. The next section describes the process of SQL patterns evaluation.

4.7 Objective 3: Methodology Design

At the core of SQL studies, discussed earlier in chapter 2, is the notion of measurement of ease-of-use. In this research, the focus is on the effect of SQL patterns in ease-of-learn of SQL by novices. The approach taken to such measurement is an extension of the field of human factors studies in query

language such as [5], [86] that discussed in section 2.5 and other specific research on the use of patterns in education which was discussed in section 3.7.

In this study, methods from the academic field of experimental psychology are applied to practical tasks of evaluation. The impact of SQL patterns on SQL knowledge acquisition is examined and the efficacy of SQL patterns is assessed in supporting SQL learning. To accomplish the measurement task, it was drawn upon techniques of experimental psychology, linguistic research [105, 244-246], general educational theory and studies and the use of patterns in education research as discussed in section 3.7.

4.7.1 Research Questions

Dearden and Finlay [113] suggest that a significant effort is now required to examine the use of patterns in education to demonstrate what benefits might be gained from a patterns' approach. As such, using patterns in education is part of the contributions that this thesis aims to achieve. The related research questions are presented in Table 4.9.

Research question 3: What is impact of SQL patterns in learners' performance?		
1	Do SQL patterns improve SQL acquisition?	
2	Do SQL patterns improve the following aspects of novices' performance?	
	A	Problem solving
	B	Quality of the solution (correctness and completion)
	C	Intermediate attempts
3	How have participants felt about the efficacy of the patterns?	

Table 4.9: Research Questions – The Effect of SQL Patterns in Learners' Performance

To answer the above research questions, it becomes clear that the diversity of this study requires the use of multiple strategies. During the research, data was collected from learners, educators, and relevant education theories. In addition, more emphasis is required not only on what learners say but how they actually learn.

There were three primary research questions. The first two are related to the proposed CS learning Taxonomy in section 2.2.1

1 The first research question examines the impact of SQL patterns on SQL knowledge acquisition.
2 The second research question assesses the efficacy of using SQL patterns and pattern language in learning and mastering SQL by evaluating the effect of SQL patterns on the learning process through three dimensions which are: Participant's problem solving skills, participants solution's quality (validity and completeness), and participant's nature of attempts.
3 The third question is regarding the participants' feeling about the efficacy of the patterns.

The next section describes the definition of the employed task and the related variables measurements.

4.7.2 Tasks Definitions and Measurement

Query language refers to the particular formal computer language with all its syntax and semantics, with which a user can express formally the required data and operations. Measures of SQL learning and use should be defined in order to test its effectiveness. Reisner [4] presented a list of standard experimental task that included problem solving, memorization, query writing, and query reading. Comprehension questions were added to Reisner list by Juhn and Naumann [247].

In this experiment, different tasks were employed such as memorization, query reading, query comprehension, problem solving and query writing. Within these tasks the following operations were included in the problem solving and query writing task: projection, selection, join, self-join, repeated relation, group, IN-subquery and exist-subquery. The queries covered a comprehensive range from the easy to the very difficult. There were five chosen queries covered by the

following semantic specifications: two or more entities (of different types or same type) connected by a relationship, attribute condition, two instances of the same type, aggregation of relationships, quantifiers for where, IN, exist and Subquery. Each query consists of different combination of operations and the previous query had no connection with the following one. The task can be found in Appendix J

The employed task is measured in terms of its variables. The next section defines the experiments' variable and their related measurements.

4.7.3 Experiment's Variable and their Measurement

In this study, the variables used can be divided into independents and dependents.

- Independent variables: An experiment between groups design was conducted to test the hypotheses. There were two independent variables: SQL patterns and other SQL materials.
- Dependents variables: Novice query performance is commonly measured in terms of participants' performance and participants' satisfaction [93].

Individual participants were randomly assigned to one of the two groups: experimental and control group. The experimental group is sometimes referred to as a pattern group. In this experiment, the participants' satisfaction was measured based on respondents' feedback on a set of 5-option Likert scales in the questionnaire. On the other hand, participants' performance was measured by performing two tasks: knowledge acquisition task (memorization, query reading, query comprehension tasks) and problem solving task (problem solving and query writing tasks). Knowledge acquisition was measured by the difference in participants' score in the pre-test and post-test. Problem solving task, on the other hand, was measured by four aspects:

1 Examining participants' problem solving skills (problem formulation and translation).
2 Examining participants' attempts in terms of error classification and analysis.
3 Examining participants' query written skills. This involves measures of the correctness of the query and measures of the percentage of question completion. Solution correctness which is a measure of the required knowledge (accurate match between the problem context and the related knowledge).
4 Time: measures the time spent to solve each question.

After the identification of the experiment's variables, then a description of the task procedures should take place. The next section presents a description of the task development procedures.

4.7.4 Task Development

The methods that were used in the second step of stage 3 are an experiment, and usability questionnaire. Table 4.10 illustrates the research methods.

	Method	Participants	Participants background
1	Experiment	90	Students studying under database specialization Higher College of Technology (HCT), Oman
2	Usability questionnaire	19	Students studying under database specialization HCT, Oman

Table 4.10: Research Methods Used for Research Question 3

To capture the aspects of ease-of-learning of SQL patterns in using SQL, the researcher developed a number of different tasks. To achieve that, eight instruments were employed (see Figure 4.11)

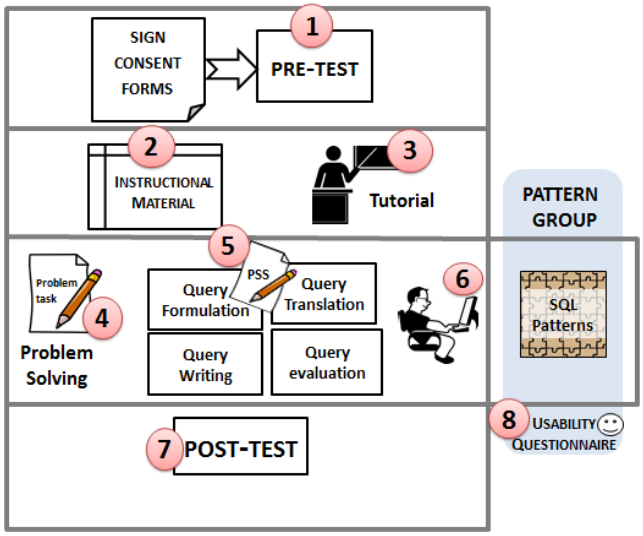


Figure 4.11: Research Instruments Used for Research Question 3

The following are the steps of the experiments' procedures:

- All participants were asked first to read and sign the informed and consent forms.
- Then, they took the pre-test .
- Participants in each group were given the same tutorial on particular SQL concepts.
- They were handed with the experiment material that consisted of either the patterns used for the experiment or other material such as lecture notes. Experiment group received five patterns: Natural join, Self-Join Pattern, Grouping Result Pattern, Filtering by Existence Pattern and Dynamic Filtering Pattern
- All participants received a task sheet; PSS sheet. Then, they were asked to use PSS forms to analyze and synthesize each question in the given task.
- They were asked to use the SQLPB tool to write the query.
- Then, all took the post test .
- The patterns group filled in a usability questionnaire in their own free time.

Brief descriptions about the different tests used in this experiment are shown in Table 4.11.

Instrument or test	Description of the test	Tasks involve
Pre-test or pre-learning test	Given before the experiments Evaluate participants' knowledge in SQL. Determine how easy it is to remember the meaning and the application of the examined SQL concepts	Query reading, query comprehension and memorization tasks
Post-test or relearning test	This is given after the experiments to evaluate participants' knowledge and understanding the application of SQL concepts. It determines how easy it is to remember the meaning and the application of the examined SQL concepts by novices who have participated in the experiments	Query reading, query comprehension and memorization tasks
Problem solving test	Shows how easy to understand the given problem context. Identify the facts by stating the required rows, columns and tables from the question. In addition, to examine participants in highlighting the required knowledge by stating which patterns or concepts should be used to solve the problem.	Problem solving task Patterns matching task (Query formulation and translation)
Query writing test	Present the participants performance in their ability to write a non-trivial SQL query to the given problem within a given time. This test also helps in identifying the common (syntax, semantic) errors novices attempt to make during the query writing, the misconception in SQL.	writing SQL query

Table 4.11: The Tests Used in The Experiment

The design of the different tests which were employed in the experiment was adapted by the supervisors and academic faculty from the regular problem in the normal courses at University of Glasgow and at Higher College of Technology, Oman.

The next section describes the design of the instruments that were employed in this experiment.

Pre-test task: The purpose of the pre-test is to investigate students' preconception of some SQL concepts and to make a comparison between the two groups. In addition, the pre-test is used to gather both demographic information and students' academic information on the number of SQL courses the subjects have completed their grades and their academic GPA. The pre-test questionnaire can be found in Appendix H.

The pre-test was designed based on learning theory to assess students' knowledge in these concepts: Joining tables, Sub-query and Aggregation. The process of validating the content of the test was conducted by two academic supervisors, two IT faculty members and one faculty from academic faculty at the Higher College of Technology, Oman. The moderation was done by reviewing the questions of the test independently to determine whether the questions measured the concepts that were being assessed. As a result, some questions were rephrased or removed and replaced by other questions that were more relevant to the specific topics.

The test consists of 12 questions, with a minimum score of 0 and a maximum of 12 which probed understanding of the interrelationship among the concepts involved in both problem solving tasks and query writing tasks.

Tutorial: The purpose of the tutorial is to explain to the subjects the assessed concepts and to make sure that both groups had the same level of knowledge before conducting the other tasks. In addition, SQL patterns were introduced to the experimental group. The tutorial description sheet can be found in Appendix K. The design tutorial was based on the informative approach by Bruer [62] stating the related conditional knowledge for each concept. Each of the examined concepts was introduced as a new knowledge, then its use was explained in terms of when and where to be used and why it is important to be used in such a context. An example was explained for each concept to make sure that the participants understood the intended concept.

To make sure that both groups get the same level of knowledge, one IT faculty attended the tutorial to provide additional support to the researcher. When one of the concepts was not explained well, the researcher gave advice to revisit that concept. At the end of the class, each group submitted the relevant materials.

SQL Patterns: The subjects in the patterns condition (Experimental group) used the list of the patterns as an instructional material to solve the given task. Each pattern represents a different concept.

The SQL patterns were designed based on a collaboration of many research on patterns writing and educational theory as was explained in chapter 3. Participants had a printed copy of the SQL patterns. In addition, they had the chance to look at them electronically via the tool as well. SQL patterns design and development is discussed in chapter 6.

SQL Lecture material : The subjects in the control group used normal lecture notes as a teaching method to solve the given task. The materials are used by the course (ITDB 3208, "SQL Concepts and Syntax") by teachers at Higher College of Technology, Oman. The relevant materials were refined and used to match the knowledge available in each pattern. Each participant had a hard and a soft copy of the materials.

Problem solving task: The purpose of this task was to measure the subjects' skills in solving a real scenario where they need to determine the correct SQL query. This task aims also to provide the research with the following:

- Problem solving strategy that each participant follows in each group.
- Correctness: the final submitted query solution was evaluated using the rubric in Appendix M, by two faculties and the researcher.
- Trials and errors: collection to quantify and qualify learners' progress in solving the given problem.

- Time-stamps: recording to examine the difference in time spent on the task between different groups of learners.

The task was designed to assess participant's ability to understand the context of the given problem scenario, to be able to translate the problem by finding the related facts within a given problem, identifying and analysing the possible cause, and listing out all the possible solutions. To achieve that, the cognitive models in solving SQL problems, as discussed in chapter 2, were considered.

Students' Problem Solving Strategy (PSS) Form: The PSS form aims to collect data that is used to assess individual skills during query problem solving, based on both analysis and synthesis of the given problems (Appendix M). The researcher identified three major dimensions that were consistently represented in many problem solving theories and included them in the rubric, which are:

- Analysis and Synthesis: Problem understanding assesses students' ability to understand the context of the given problem "query formulation". They should be able to highlight the related facts such as the required tables, relations, and columns to solve the problems, which relate to the query translation. Query formulation and translation cognitive tasks are related to three-stage cognitive model of database query in [13].

- Application: Knowledge Generation and application: by identifying which of the SQL concepts need to be applied in the given scenario? This might be related to query writing stage three-stage cognitive model of database query in [13].

- Problem solving evaluation: evaluating the students' skills in identifying and analysing the possible causes to the given problem and the impact of the employed solution. This is related to the high skills thinking of "Why" and "When".

This rubric or PSS form design was reviewed by an expert panel with extensive knowledge in Computer Science curriculum design.

Post-test task: The post-test task aimed to measure the change of participants' understanding of SQL concepts in response to the research teaching method. The post-test task can be found in Appendix I.

It was designed based on learning theory to assess students' knowledge in SQL concepts. Two academic supervisors, two IT faculties, and one academic at the Higher College of Technology, Oman conducted the process of content validity of the test. The moderation was done by reviewing the questions of the test independently to determine whether the questions measured the concepts that were being assessed. As a result, some questions were rephrased or removed and replaced by other questions that were more relevant to the specific topics.

The test consists of 12 questions, with a minimum score of 0 and a maximum of 12 which probed understanding of the interrelationships among the concepts involved in both problem solving task and query writing task.

Usability questionnaire: The usability questionnaire aimed to gather participants' opinion about the use of the patterns that has been used to help them in solving the task. The usability questionnaire can be found in Appendix N.

The questionnaire was designed based on ISO standard to evaluate the use of the patterns in terms of its usability. Various components of usability such as learn-ability, efficiency, memorability, errors, satisfaction, and utility component were used as highlighted by Nielsen [248]. Some of these components were used to test the usability of the used tool to solve the given task as well as the design of the patterns.

SQL Pattern Based (SQLPB) tool: The aim of the tool is to design an interface that would be used in conjunction with more conventional learning methodologies and tools so that participants would perform more effectively.

The tool also captured responses and tracked response times. Hence, all participants' trials and errors were recorded. In addition, it provided all needed instruction and training materials.

There is various software packages available that were developed specifically for learning and practicing SQL query formulation skills and these were discussed in section 2.3.2. The reviewed tools guide this research towards designing a new tool that overcomes the highlighted issues in section 2.3.3. The tool was designed on Netbeans platform.

There are five main windows that users can use. In the beginning, the participant can navigate the question that he/she is going to answer. Then, through another frame, a checklist can be used to select the appropriate knowledge or patterns to help in solving the selected question. Participants can then open another window which is a "pdf" file of the related document which is either the SQL patterns or SQL lecture notes. By this time, they have different options to those windows, specifically to: minimize, close, or change position such as making them as a side window. Once the learners are ready to solve the question, they can connect to the database, generate ERD diagram automatically from the related schema and open SQL command. They can open more than one SQL command at the same time. However, only one query can be executed at a time. They can view the results of more than one query. In other words, more than one output window can be viewed.

4.7.5 Experiment Setups (environments and materials)

The experiment was designed in pre-test, post-test, problem solving task and treatments control group and experimental group. Participants are placed randomly into both groups. Jarke *et al*. [249] discussed the criteria for testing the performance of more than one group. The language should be directed towards the same type of user, and to be used in a similar system environment, and using the same DBMS. SQL Evaluation was similar to the approach adopted in other research that were reviewed in section 2.4 such as [5, 81].

Individual participants completed the exercises at the IT department in HCT College. The overall procedure can be seen in Figure 4.12 where all participants read the information and signed the consent.

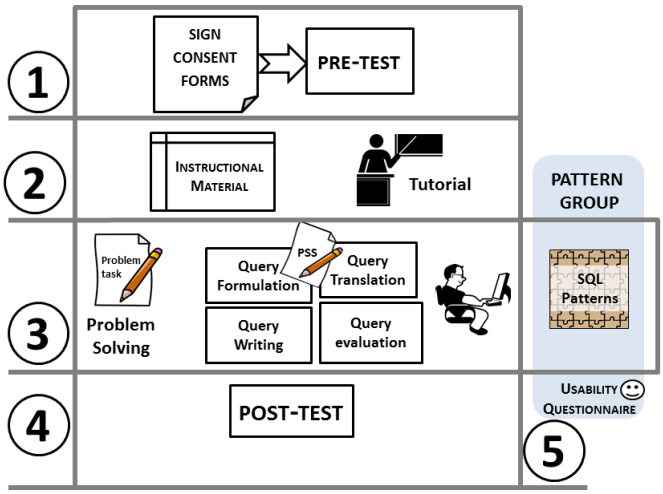


Figure 4.12: Experiment Steps

Both groups received the pre-test on whatever the instrument was used to assess the effect of the received teaching method before the treatments were given. After that, both groups attended a tutorial where the examined concepts are explained and discussed. The patterns' group was given a set of patterns while, for the other group, the SQL lecture notes and text books were made available for use. The experimental groups had fifteen minutes extra to explain to them about SQL patterns. Both groups received the same tasks. Then, each group participated in the task of problem solving. For the problem solving task, the participants first needed to decide what elements of the data model are relevant, and the necessary operations using PSS sheet. They needed to refer to the given materials (SQL patterns or lecture material) and use paper and pencil to formulate the required information. ER model was given to all participants on paper. Then, for the query writing, they had to write the related SQL query. The tools used recorded all the trials and errors attempted by each participant and the time taken for each question. In addition, the participants could have generated ER model from the given tools. Subsequently, both groups were requested to perform the post-test. The experimental group was asked to fill out

the usability questionnaire. The experiment was designed with a control of extraneous factors. Two IT faculties and the researcher then evaluated all the results. Both faculties were academic in IT department with more than 10 years in teaching database course experience.

The Experiment program Task description	Time/day
Information sheet and consent form < coffee, tea breakfast available to all>	Day1 (Control Group)CG, (Experimental Group)EG
Pre-test	9:00-9:20 (Control Group)CG, (Experimental Group)EG
Tutorial (CG)	Day2 9:00-10:10 CG
Coffee and snacks break 20 minutes	
main task –session 1	10:30 -11:30
10-15 minutes break	
Main task-session 2	11:40-12:30
Tutorial (EG)	Day3 9:00-10:10
Coffee and snacks break 20 minutes	
main task –session 1	10:30 11:30
10-15 minutes break-coffee and snacks	
Main task-session 2	11:40-12:30
Post-test	12:40-1:00 (Control Group)CG, (Experimental Group)EG

Figure 4.13: Experiment Program

The experiment procedure (see Figure 4.13) was approved by the Ethics Committee at University of Glasgow (see Appendix O).

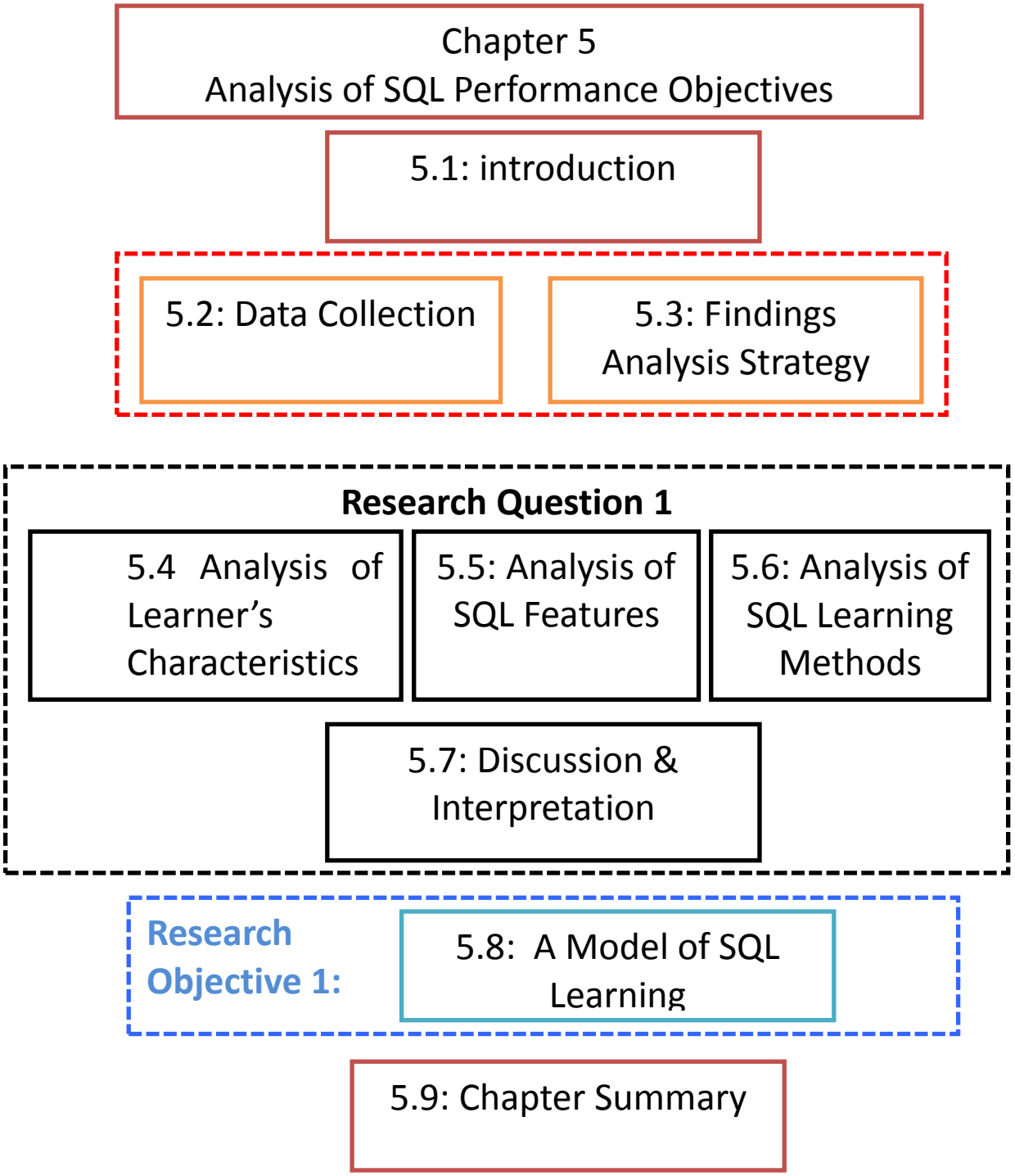
4.7.6 Results Analysis

Quantifying human performance in these complex cognitive tasks is a challenge. A central problem in this area is developing adequate techniques for measuring. Chapter 7 documents all the data collection, results analysis and discussion.

4.8 Chapter Summary

This chapter has introduced the research design methodology, which includes a description of the research structure and how this research has been conducted. Moreover, the chapter provided insight into the structure of the research and the research framework and the use of combined research approaches and different tools. Satisfactory data has been generated to answer the research questions. The research framework was employed to provide knowledge in the three areas: the first one is to find out the factors that affect novice performance in learning SQL; the second is the method that was used to form SQL knowledge which is presented as SQL patterns and employed as instructional materials; and finally the evaluation of the influence of SQL pattern in novice's performance.

Chapter 5, which follows, aims to answer the first part of the research questions; it explores the factors that might influence first year undergraduate student's performance in learning SQL. It covers the different diagnostic tasks that were used in this research to explore novice's attitude and cognitive factors. The first set of factors is categorized under the characteristics of novice SQL learners. SQL novices' are influenced by a number of factors in learning such as: their personal attitude, previous experience, problem solving skills, and acquisition abilities. SQL language features, on the other hand, were considered as another factor which includes: SQL nature, SQL syntax and SQL concepts and knowledge. The cognitive factors were evaluated as well, such as: students' ability to understand the given scenario (query formulation), students' skills in reading and comprehension of SQL queries (query comprehension) and students' ability to write non-trivial query (query writing) which is the application of their knowledge.


Chapter 5
Analysis of SQL Performance Objectives
5.1: introduction
5.2: Data Collection
5.3: Findings Analysis Strategy
Research Question 1
5.4 Analysis of Learner's Characteristics
5.5: Analysis of SQL Features
5.6: Analysis of SQL Learning Methods
5.7: Discussion & Interpretation
Research Objective 1:
5.8: A Model of SQL Learning
5.9: Chapter Summary

Chapter 5: Analysis of SQL Learning Performance Objectives

This chapter reports the results of research question one, identify SQL impediments that get in the way of learning performance, which aims to identify the learner characteristics and the factors that are associated with learners' knowledge and skill acquisition in learning SQL. This chapter also identifies the cognitive factors involved in solving SQL problems.

5.1 Introduction

The factors that might predict success in introductory courses, such as programing languages, was reported in many studies [225, 228, 250-252]. The factors suggested in the literature include: mathematical background and previous experience [217, 218, 253, 254], logical reasoning ability and previous academic background [226], learner attitude and academic motivation [227, 228].

Plenty of research has been conducted to evaluate SQL from human factors perspective in terms of how easy it is to learn, understand, and use (as was discussed in section 2.5). However, this research did not focus on the teaching and learning of SQL. Consequently, they did not focus on the cognitive activity that learners perform when solving SQL problem. Results are stated from a user's perspective rather than a learner's perspective. In addition, these studies do not reflect on their findings the use of educational theory and cognitive science, which are essential when the focus is on teaching and learning.

This chapter aims to analyse the characteristics of learner and context by identifying the factors that might influence entry-level undergraduate students' success in learning SQL. Then, the findings were formulated towards determining

SQL learner performance objectives [104] and structuring them as a framework labelled as “SQL Framework Model”. The SQL framework model is used as a map to facilitate the SQL instructional design objective.

The chapter is structured as follows: section 5.2 explains how data was collected from the different research methods that were applied and those were analysed as reported in section 5.3. Section 5.4 identifies different learner’s characteristics that influence learning SQL from both learners and educators perspectives. Section 5.5, on the other hand, examines the characteristics of the learning context by investigating different language features of SQL. Section 5.6 returns to the learners’ cognitive skills that were discussed in section 2.3. It explores the influence of current teaching methods and approaches on learner’s knowledge and skills in the light of these cognitive processes. Section 5.7 discusses this study’s findings about the learner and context characteristics. It highlights the factors in learning and teaching SQL and motivates the use of new instructional material, which better aligns with human cognition and learning styles. The results of the research methods in the previous sections are interpreted with a view to envisioning the performance objectives that should be achieved from any new instruction design as a framework called “SQL Framework Model” in section 5.8. Chapter 5 is summarized in 5.9.

5.2 Data Collection

To achieve the aim of the research, several methods have been applied to explore the influence of learners’ characteristics and the nature of SQL itself on the learning process and how to design a meaningful instruction. According to Kotze *et al. [255]:*

> “Think for a moment how tricky it is to construct a meaningful experience for others. You must first understand your audience, their needs, abilities, interests, and expectations, and how to connect with them.”

To answer Kotze's call, this study utilizes five different research methods, as shown in Table 5.1, to explore factors that might relate to novice SQL acquisition.

	Method	Participants	Aims
1	Semi-structured Interview	7 Students	to understand the problems experienced in learning SQL
2	Cognitive task	7 students	to investigate students' ability to explain in English how to solve query (Query translation) and write the related SQL (Query writing)
5	Online questionnaire	14 teachers	to evaluate difficulties in teaching SQL from the educator perspective
3	Questionnaire	75 students	to evaluate difficulties in learning SQL from learners' perspective (informed by 1)
4	Comprehension Task	64 students	Cognitive task focusing on student's ability in comprehend SQL.

Table 5.1: Research Methods Employed

The design of the above research methods was presented in chapter 4. The data collection of each method feeds in to the design of other methods, as shown in Figure 5.1.

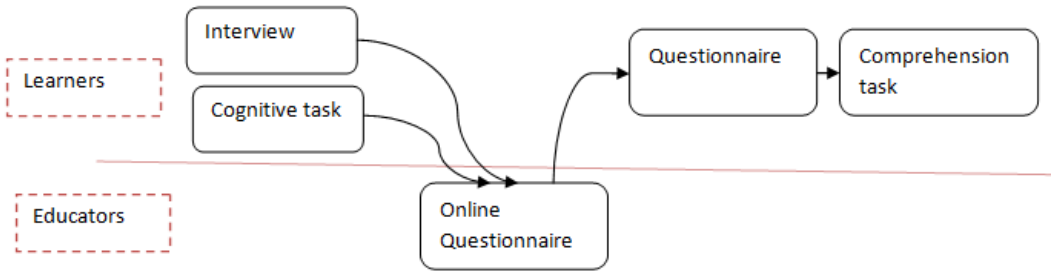


Figure 5.1: Research Methods Sequences

The next subsections describe each research method in details.

5.2.1 Semi-structured Interview

The Semi-structured interview was designed to deliver comprehensive information relating to participants' characteristics. Furthermore, it aims to analyse the learning process by exploring viewpoints of SQL nature, syntax and content. The data from semi-structured interviews was collected and categorized into three areas. The first part investigated the characteristics of novice SQL learners (e.g. their attitude towards learning, previous knowledge and experience). The second part analysed the data in terms of the SQL nature and learner's perspective of learning SQL. The third, rated the student's knowledge of different SQL basic concepts. The qualitative analysis is supported by verbatim quotes from the interviews and text analysis.

The participants were seven students (two third-year students, one BSc-Honour's student, three Masters Students and one PhD student). None of them had prior work experience using SQL. Participants were asked to evaluate their own skills in SQL problem solving using the following categories:

Expert	Advanced	Novice	Beginner	Not Knowledge

Figure 5.2: learners' Knowledge Rating

Six had taken two SQL courses and classified themselves as novices at SQL problem solving, while one considered herself to be within the advanced level of knowledge (she did two courses in SQL). Moreover, she was currently engaged in a project that required the use of SQL. Participants were asked about their attitudes towards learning and using SQL and to justify their personal feelings.

Five of the participants reported feeling slightly uncomfortable about using SQL and only two were comfortable with SQL. For the first part, there were many reasons participants had stated to explain why they felt slightly uncomfortable:

- Less experience or no experience apart from their classes: “I do not have all that much experience with it, the only time that I have contact with is only during class”.
- The nature of SQL: “SQL is quite different from programming language that I study. It requires a certain reasoning that I did not have"; “SQL is not like Java when you solve SQL problem you do not know which answer is the right one”; “Writing SQL takes me a while and I have to do trial and error”.
- SQL syntax: “I cannot see the relation between the statements and their context”.
- SQL concepts: “SQL concepts are not difficult to understand or apply as an individual concept but when you are given a complex situation where you have to apply many concepts then there is the problem”.

The two students who did feel comfortable with SQL attributed this either to their own attitude towards database concepts, in general, or to their accumulated experience with SQL. They rated themselves as advanced SQL writers. One of the students provided the following comment in this respect:

> “I like the whole concept of databases; I am very keen in learning about databases rather than programming”.

These results are discussed within this chapter. The second part of the interview showed how students rate the given concepts. This study is focusing on four concepts: table joins, nested query, grouping and relational algebra. Table 5.2 shows the result.

Courses concepts	**Very Easy**	**Easy**	**confusing**	**Hard**	**Very Hard**
	percentage				
Restricting data (limit the row that retrieve by the query)	14.29	71.43	14.29	0	0
Sorting data(sort the row that retrieve by the query)	14.29	71.43	0	14.29	0

Using group functions to report aggregating data (AVG, SUM,MAX,MIN,COUNT)	14.29	0	71.43	14.29	0
Grouping rows using GROUP BY	14.29	14.29	42.86	28.57	0
Displaying data from multiple data (self join, inner join, outer join)	14.29	14.29	28.57	57.14	0
Using sub query (single row, multiple row)	14.29	14.29	57.14	14.29	0

Table 5.2: Result of the Semi-Structured Interviews, SQL Misconception

Figure 5.3 shows that 71% of students agreed that using group function is confusing and 14% agreed that it is hard. 57% agreed that joining tables is a hard concept while nested query is confusing.

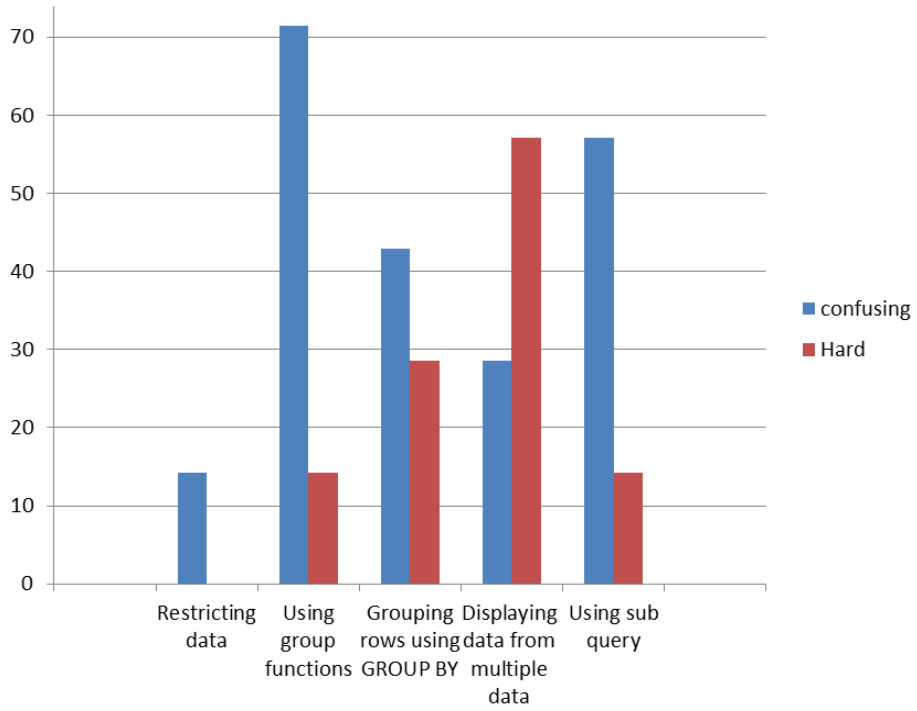


Figure 5.3: Students Rating of the Difficulties of SQL Concepts

The findings of this study are represented in the SQL misconception section later in this chapter.

5.2.2 Ability to Solve SQL Problems

Developing problem solving skills has been a focus of science educators for many years [47-51]. Learners can be characterized as good or poor problem solvers. To evaluate student's ability in solving SQL query, they were given an SQL task. The participants were the same as those who had been interviewed; task design was presented in section 4.5.2 and is shown in Figure 5.5. The task examined participants' ability:

- To explain in plain English how to solve SQL problems.
- To write non-trivial SQL Query correctly.

Students were given the following question:

> Find the names and the hire dates for all employees who were hired before their managers, along with their manager's name and hire dates.
>
> Sort by employee name
>
> Note: all information is stored in table: Employee.

Figure 5.4: Query Formulation Related Task

They were asked to translate the problem by deciding what elements of the data model are relevant, and the necessary SQL concepts and operations which need to be applied. Then, they were asked to write the related SQL query. Finally, students were asked to give some feedback about the question level of difficulties and provide insight towards their feeling in solving the question; for more details, refer back to section 4.5.2.

The task results show that 85% of the participants were able to translate the given problem into natural language. They were able to identify the required data. However, only 28% of them were able to state exactly which concepts of SQL should be used (Self-Join) and how it is to be achieved.

5.2.3 Educator's Perspective

To complete the picture and collect information from different perspectives, data were collected from educators. It investigates SQL nature, syntax, content and the common difficulties in teaching SQL concepts. In addition, it highlights the nature of the process in learning SQL. Moreover, it reflected on students' responses to the tasks from the previous research methods, which examined participants' ability to solve SQL problems.

The questionnaire was sent by email to several teachers who were either currently teaching SQL or had done research in teaching SQL. In total, fourteen academics and researchers participated. They had different levels of experience in teaching SQL, as shown in Figure 5.6.

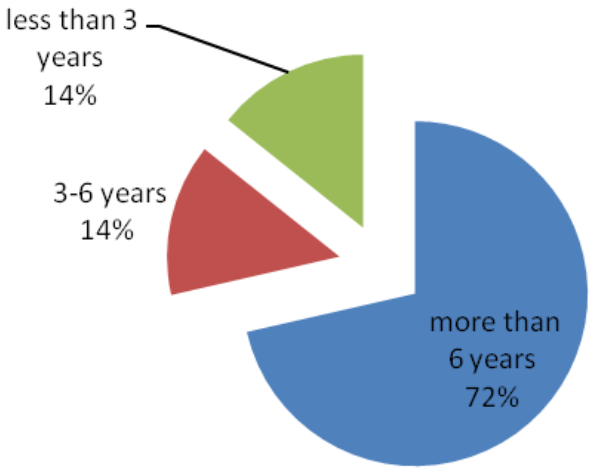


Figure 5.5: Teacher's Level of Experience with SQL

The following are the data collected from different parts of the questionnaire:

Question 1: *Do you believe a solid grounding in set theory helps students understand database concepts?* The answer, as shown in Figure 5.7, 11 (78%) participants agree while only 3 (21%) of them disagree and claim that they are teaching SQL without a prior knowledge of set theory.

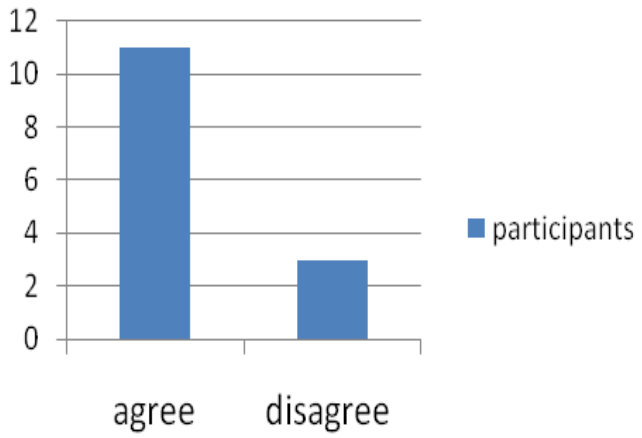


Figure 5.6: Do you believe a Solid Grounding in Set Theory Helps Students Understand Database Concepts?

Question 2: *Which concepts did you find most challenging to teach, or that students find difficult to understand? Please also say why do you think these particular concepts were problematical?*

There were many concepts listed as an answer to this question. Some are related to Database theory in general while others were SQL concepts. Here are those concepts:

1- Determinacy: because (young) students have no experience of its implication in the real world. For instance, when they are shown how data is used to produce a receipt, it may be the first time they see a receipt at all.

2- Normalization: It can only be taught effectively using stylized examples that hide the real meaning of the process.

3- Relational algebra: difficult to teach because it appears too theoretical to the students, and does not obviously have a practical application. Moreover, students do not have the mathematical background.

4- Nested Query: because it is difficult to conceptualize the process.

As it could be seen, some of the participants interpret the questions differently; for example: determinacy is not one of SQL concepts but some participants included it. It is possible to relate the determinacy with the nature of SQL learning and one of the issues that might be worth to consider in future studies. Normalization is not part of SQL concepts as well but it is related to Database Management courses and usually taught along with SQL.

While other participants had different opinions, for example:

> "In my opinion understanding this notion unravels all the other ones - but "determinants" is not the only way to think of it or describe it. Mathematical notions help (e.g. set theory) but vice versa, the mathematical notions can also follow learning database concepts".

Some gave reasons of why some SQL concepts are difficult such as

> "Students tend to find modelling…. and nested queries the most challenging concept of the ones I teach. A lot of this is thought to be because it requires very logical thinking and detailed interpretation of imprecise ordinary language and because it uses a formal modelling language/symbol set".

Question 3: *Sometimes when students are given an SQL query to write they can explain how to do it but cannot convert their thoughts into SQL. Why do you think this is?*

Participants gave different reasons about why students are having problems in both understanding and applying SQL:

1- Reading SQL statements: students cannot write SQL "because they cannot/they do not know how to read SQL statements."

2- SQL syntax: "Syntax details can be difficult; making concepts hard to embed in formal language", and "I think that it is because the ordering

of the syntax (SELECT…FROM…WHERE) is not a natural way of expressing a query - it is more usual to identify the constraints first, and then work out what tables are needed, and then work out how to join them together."

3- Solving the query by trial and error: "Is this still the case when students are allowed to compose the query by trial and error? If so then concepts are the problem (and they appear to get it right in English, only because English is subject to interpretation). If not, then syntax (translation into a formal language) is the problem."

4- SQL nature: "SQL is a very "tight" and minimalist language, and is not procedural."

Question 4: *We interviewed Master Level students who completed two courses in SQL during their master's studies. We asked them to solve the* following SQL problem:

Find the names and the hire dates for all employees who were hired before their managers, along with their manager's name and hire dates. Sort by employee name Note: all information stored in table: Employee.

Few of the students only were able to write the required SQL, although some were able to describe what needed to be done in order to solve the problem. Why do you think they couldn't write what is quite a simple query?

Different opinions were given regarding this question; some are related to the nature of the problem while others related to the content and the SQL concepts that were covered.

1- The illusion of complexity: "Possibly because it gives the illusion of complexity? So students will look for complex solutions? Alternatively because sub queries cause students to worry about the problem (instead of solving it)".

2- Students experience with solving SQL problem: “have the students been using SQL regularly and frequently between being taught and your interview? If SQL is not used even for a short time, the details of the language and the problem solving skills are forgotten”. This will agree on the founding that the student who rate himself as advanced and was working with SQL at that time was able to write the query.

3- SQL concepts (self-join):

- “The solution to the question requested students to query from the same table twice which is not logical and students cannot see. ‘Their managers' require that two copies of the same table be joined - they will not see this as the obvious thing to do at first.

- “Although it seems to be a simple query it is not: for beginners it is confusing to compare rows in the same table.”

- “self-join concept is one of the hardest concepts for students to conceive”

- And another participant responded along the same idea that

  “I would classify this query as fairly hard, as it requires a self-join, therefore an alias, and these are not used widely and confusing, because you need to clearly understand which version of which attribute you need to refer to at each point!”

- Another respondent wanted to assume this requires a self-join. “Easy to write SQL which is just a manipulation of the select statement. Self joins require an understanding of the underlying structure. Also of course depends on their previous experience with SQL”.

Question 5: *Classify the following concepts in how easy students find it to understand or to apply?* Results are shown in Table 5.3.

	Very Easy	Easy	Confusing	Difficult	Very Difficult
Restricting and sorting data	30.0% (3)	60.0% (6)	10.0% (1)	0.0% (0)	0.0% (0)
Using group functions to report aggregating data (AVG, SUM,MAX,MIN,COUNT)	0.0% (0)	54.5% (6)	27.3% (3)	18.2% (2)	0.0% (0)
Using single raw function to customize output.	0.0% (0)	40.0% (4)	20.0% (2)	20.0% (2)	0.0% (0)
Using group functions to report aggregating data(group functions, group by)	0.0% (0)	0.0% (0)	45.5% (5)	36.4% (4)	18.2% (2)
Displaying data from multiple data (self join, inner join, outer join)	00% (0)	18.2% (2)	27.3% (3)	45.5% (5)	9.1% (1)
Using sub query (single row, multiple row)	0.0% (0)	0.0% (0)	27.3% (3)	36.4% (4)	36.4% (4)

Table 5.3: Result of the Online Questionnaire, SQL Misconception

The next section presents the data collected from students' questionnaires.

5.2.4 Students' Questionnaire

The questionnaire's aim was to collect data from SQL learners who had done at least one course in SQL. It explores two main areas:

1. Learners' approaches, perceptions, and feelings about learning and applying SQL concepts. Responses were measured based on respondents' feedback on a set of 5-option Likert scales: 1 "strongly disagree" to 5 "strongly agree". For example :
    - I solve SQL problems by trial and error,
    - SQL syntax is easy to learn and understand,
    - I do not have any problem in writing large and complex queries.

2. Learners' views of different aspects of SQL (such as SQL nature, SQL syntax, and SQL content and the common difficulties in SQL concepts).

More details about the questionnaire design can be found in section 4.5.2. Seventy-five students from the University of Glasgow participated who were either studying at level 4 or Master's degree level. Figure 5.7 shows that

participants reported different levels of SQL knowledge and experience.

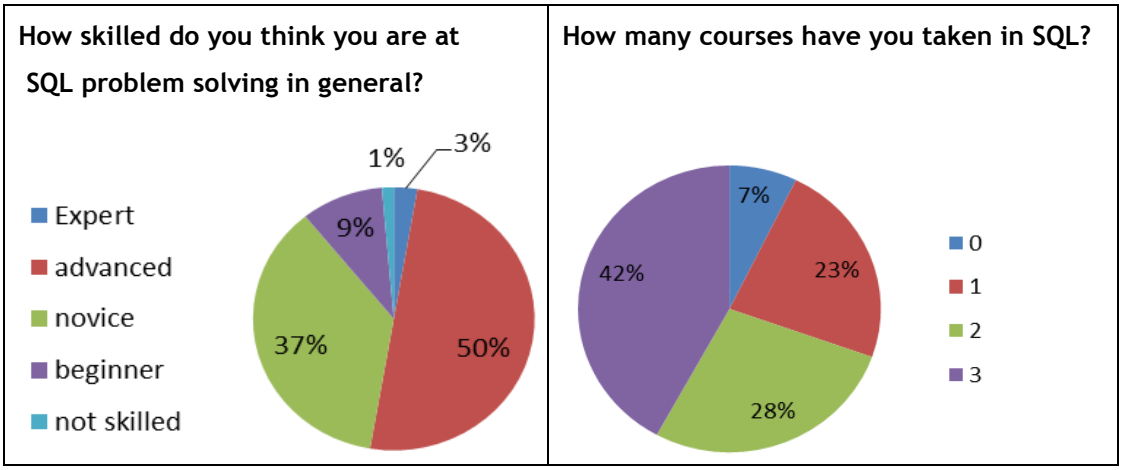


Figure 5.7: Participant's Level of Knowledge and Experience with SQL

Question 5 in the questionnaire answers is reported in table 5.4.

To what extent do you agree with the following statements		Strongly agree	agree	neutral	disagree	Strongly disagree
		Percentage				
1	I solve SQL problems by trial and error	10.7	44	25.3	10. 7	4
2	I can read and understand SQL statements easily	17.3	54.7	17.3	4	1.3
3	In general SQL syntax is easy to learn and understand	17.3	46. 7	21.3	6. 7	2. 7

4	**I can only write simple SQL statements**	6. 7	17.3	16	44	13.3
5	**I can solve a simple SQL problem**	45.3	33.3	9.33	2. 7	0
6	**I do not have problems in writing a large and complex queries**	8	17.3	36	28	5.33
7	**I know how to join more than three tables and retrieve specific columns**	25.3	38. 7	17.3	13.3	1.3
8	**I know how join a table to itself using SELF JOIN**	20	25.3	12	28	9.3
9	**It is easy for me to manipulate data using aggregate functions like SUM, AVG,COUNT,..**	33.3	37.3	18. 7	4	2. 7
10	**It is easy for me to query using aggregation by means of the Group by function**	22. 7	40	22. 7	8	4
11	**SQL is easy to use compare with other programming languages**	13. 3	21.3	34. 7	16	9.3

Table 5.4: Response to Question 5 in student's Questionnaire

This result is analysed and discussed at different points in this chapter. The next section show the data collected from the comprehension task.

5.2.5 SQL Comprehension

Comprehension is the ability to read and understand a query written by others. It is often necessary to read syntax composed by others for learning or other purposes [7]. Students' ability to comprehend SQL statements was considered in this study, as one of the factors that might affect learners' performance .The aim of this task was to examine participant's ability in reading SQL statements. The task design is discussed in section 4.5.2.5. The participants were sixty-four students in level 2 who were studying Database course. They were given SQL commands and asked to explain in plain English what the commands do and then they were asked to predict the result of the given query. The task can be found in Appendix E. Figure 5.10 below illustrates this task.

What is this SQL command trying to determine? Give your answer in plain English.

```
SELECT G.Name, P.Name, Date, Amount
FROM Picked pd, Gardener G, Plant P
WHERE P.PlantId = Pd.PlantFK
AND G.GardenerId =Pd.GardenerFK
AND  Pd.GardenerFK = 2
ORDER BY Date
```

GardenerID	Name	Age
0	Fadila	36
1	Salim	38
2	Tim	15
3	Erin	12

Gardener table

PlantID	Name	Sunlight	Weight	Water
0	Carrot	.26	.82	.08
1	Beet	.44	.80	.04
2	Corn	.44	.76	.26
3	Tomato	.42	.80	.16
4	Radish	.28	.84	.02

Plant table

PlantFK	GardenerFK	LocationFK	Date	Amount	Weight
0	2	0	08-18-2005	28	2.32
0	3	1	08-16-2005	12	1.02
2	1	3	08-22-2005	52	12.96
2	2	2	08-28-2005	18	4.58
3	3	3	08-22-2005	15	3.84
4	2	0	08-16-2005	23	0.5

Picked table

Figure 5.8: Analysis of Results Relating to SQL Query Comprehension

The results of the task, in Figure 5.8, are analysed and are presented in Figure 5.9 below. The maximum mark was 3 and the minimum was 0.

		Percentage
Valid	0/3	14.7
	1/3	29.3
	2/3	26.7
	3/3	28.0
	Total	98.7
Missing	System	1.3
Total		100.0

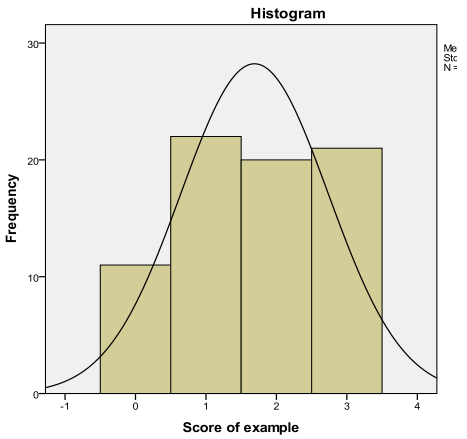


Figure 5.9: Task Analysis Results

In addition, other information was gathered about participants, such as their previous knowledge and experience in SQL by stating the number of courses they completed in SQL. They were also asked to rate themselves. Responses were measured based on respondents' feedback on a set of 5-option Likert scales: 1 "Expert" to 5 "Not skilled". The correlation between different aspects is illustrated in Table 5.5.

		How skilled do you think you are at SQL problem solving in general?	In how many courses have you taken in SQL?
Score of example	Pearson Correlation	.019	.185
	Sig. (2-tailed)	.873	.130
	N	73	68
How skilled do you think you are at SQL problem solving in general?	Pearson Correlation	1	-.495**
	Sig. (2-tailed)		.000
	N	74	68
In how many courses have you taken in SQL?	Pearson Correlation	-.495**	1
	Sig. (2-tailed)	.000	
	N	68	69

Table 5.5: Participant's Task Analysis Results **.Correlation Significant at the 0.01 Level (2-Tailed)

5.3 Analysis Strategy

Grounded Theory was used to analyse the collected data. It fits well with the chosen data collecting methods. Grounded theory is commonly used to provide insight into the factors influencing learning, use of literacies, mobility, and networking [256]. Grounded theory relies on the production of theoretical perspectives derived from data.

Grounded Theory was one of the main analytical methods used to understand the factors influencing the success in learning and teaching SQL. Due to the complexity and range of issues amongst a group of participants who had similar problems in teaching and learning of SQL, as well as the fact that this topic is an under-researched field of study.

The analysis took place during the data collection period, and was thoroughly integrated into all aspects of it, including an analysis of every interview, questionnaire, and observation directly after they were given. In this way, each step of the data collection could feed in to the analysis as shown in Figure 5.1. It consisted of three strands that utilized mixed methods, and these were triangulated for the sake of rigour and balancing out the things students said, during interviews, and did in the cognitive task which either confirmed or contradicted educators' viewpoints as gathered in the online questionnaire.

The data from the above study methods were collected and categorized in terms of three themes:

1. *The characteristics of novice SQL learners:* who might be influenced by a number of factors in learning such as personal attitude, previous experience, problem solving skills, and acquisition abilities.
2. The learning context: which involves different features of SQL language.
3. The impact of the current learning methods in learner skills in performing the different level of cognitive tasks (Figure 2.3) such as comprehension task and the problem solving (Figure 2.10):

- Students' skills in reading and comprehension of SQL queries (query comprehension);
- Students' ability to understand the given scenario (query formulation);
- Students' ability to translate the given scenario (query translation);
- Students' ability to write non-trivial query (query writing), which is the application of their knowledge.

Various tools and methods were employed to analyse the results. SPSS was used to analyse data from both questionnaires. A specifically tailored marking schema rubric was used to evaluate the results of the task analysis.

5.4 The Analysis of Learner's Characteristics

SQL novice learners lack the knowledge and the skills of experts in the learning progression. There are number of factors that influence the learners' performance in SQL. The literature highlighted a few of them as was discussed in section 2.5. The results analysis suggested the following factors: personal attitude, previous experience, problem solving skills, and acquisition abilities. The following subsection explores these factors, starting with a review of the literature, and then reporting on how the study's findings relate to this.

5.4.1 Learner's Personal Attitude towards Learning SQL

The literature does not explain why some students master SQL skills in particular better than others do. The role of personal attitude towards learning is undoubtedly one of the most controversial and fascinating areas of research.

Many researchers have proved that personal attitude towards learning plays a vital role in educational settings and influences learning processes which affects achievement [215, 257]. Some studies measured the relationship between students' achievements and personal attitude towards learning mathematics

[217, 218, 253, 254, 258, 259]. The influence of negative attitudes among novices learning programming languages has been highlighted as well [221, 260].

The interview questions sought to examine the relationship between personal feelings and achieved SQL knowledge and expertise. Participants who felt slightly uncomfortable claimed that they did not use SQL after completing their database course and even within the course had limited opportunities to practice. The comments of those who felt comfortable suggest a clear positive attitude towards SQL, which was notably missing from the responses of those who were not as comfortable with SQL. This finding supports the earlier finding by [261] in terms of all database concepts:

> "Our experience has demonstrated that beginning database students are often lackadaisical, in terms of motivation, to grasp the precise meaning and definitions of key terms used in the database field." (p. 4)

They added that the challenge for the Database Instructor is to enliven the "dry" introductory chapter, which emphasizes the concepts definition, by using some interesting exercises showing how concepts can be applied.

Determinacy might be another factor that affects learners' attitude. It was highlighted as an important issue during learning SQL, specifically by one of the educators in the online questionnaire:

> "Determinacy: because (young) students have no experience of its implication in the real world. For instance, when they are shown how data is used to produce a receipt, it may be the first time they see a receipt at all."

This research suggests that SQL learners' attitude issues can be tackled in many ways. Looking at the degree program design in the institute will raise some aspects of the concern, such as the numbers of courses that involve teaching SQL

concepts, skills, and the number of student projects that involve using database knowledge in general and SQL specifically. Teaching methods, on the other hand, play a vital role in student's attitudes towards SQL knowledge transfer. Providing a balance of theoretical and practical knowledge of SQL and presenting SQL concepts to the learners in an informative way might lead to a positive attitude towards learning SQL. Moreover, the content of the problems that students solve need to be sufficient to prompt interest.

Examining learners' emotions and attitudes during problem solving and query execution might give some indication of possible reasons behind the student's attitude in learning SQL. To do so, several aspects of how students solve SQL problems need to be explored. What errors do they produce? To what extent can they comprehend SQL and write non-trivial SQL commands? These are all factors that might have a direct effect on learners' attitude. There are some studies that suggest automatically detecting novice emotions during programming [262]. This could potentially be applied to writing SQL queries and detecting the emotions of learners at different stages of the problem-solving task. Applying such an emotional detection might help both the educators and the course designers in:

1- Identifying the level of difficulty: at different stages in problem solving; for example: what learners feel when: reading a problem, formulating, translating a problem, looking for the required knowledge, and writing the query.
2- Confirming the SQL misconceptions.
3- Evaluating the usability of the instructional materials used such as the used tools.
4- Differentiating emotions for various types of learners. For example, good learner's attitude towards a different stage of problem solving might be different than poor learners.

Applying such an emotional detection might also be seen as extremely invasive. In addition, its use should be embarked on only after scenario consideration of

all the issues. The next section examines the influence of learners' previous experience.

5.4.2 Learners' Previous Knowledge and Experience

Identifying learners' previous experience of the courses that they did before studying SQL can justify the nature of the influence. Some researchers suggest that differences in computer knowledge influences the success of query language performance [5]. SQL learners commonly have some prior knowledge of programming languages, mathematical concepts, and relational databases. The following subsections explore these aspects.

5.4.2.1 Previous Experience with Programming Languages

In the literature, SQL has been compared with other languages [5, 6, 33, 57, 81, 85, 86]. These researchers compared learnability of SQL and other languages such as SQUARE, TABLET and natural language. They found that participants with more programming background showed better performance than those with limited knowledge [5, 81].

During this study, many students spontaneously compared their experience in learning programming languages with learning SQL. This is evident in their responses, as shown in Table 5.6.

"SQL is quite different from programming language that I study. It requires a certain reasoning that I did not have".
"SQL is not like Java; when you solve SQL problem you do not know which answer is the right one".
"SQL is not like java; it is quite difficult to remember"
"Problem getting into the way of thinking in terms of tables as opposed to classes, etc.
"Imperative programming in which students are exposed is conceptually different from the filter of the Cartesian product "

Table 5.6: Participants' Response Towards SQL Experience

From what students said about their experience in both SQL courses and programming courses, it is worth exploring the effects in more details. Many students attempt to compare SQL with other languages, which sometimes mean that many do not like to study SQL related courses or even use SQL in a project. What is obvious in this study is that SQL learners need to be aware of the difference between SQL and other programming languages, which might improve their attitude towards learning. It is possible to argue that students need to be aware of the following main differences, as stated by Sengupta [263]:

- SQL is an established query language that has been used by many researchers of database query language.
- SQL has a formal foundation that leads to fewer semantic errors, which allowed the language to be optimized.
- SQL is a simple language that can be used even by non-programmer users and it is best for simpler tasks. This might be applicable to simple tasks only, as was discussed in section 2.5.1.

Moreover, the difference between SQL and traditional programming languages might easily cause a problem for those with limited experience in programming languages. For example, students learn that the condition of a loop statement needs to be applied continuously in programming languages such as Java, but only once during SQL query execution. One specific example of this was found during the solving of the cognitive task in section 5.2.2. When novices were asked to find the frequency of an item in the same column distributed within several rows, many students found it difficult to figure it out and some of them wanted to use loops when they were supposed to use either, self-join or nested query. Therefore, it is important to differentiate in teaching both languages and consider any previous knowledge students may have.

Chapter 7 will investigate learners' trial and error attempts and to uncover further evidence of the impact of their prior knowledge in programming on their SQL.

5.4.2.2 Previous experience with Set Theory courses

The other courses that might have an impact on SQL learning are mathematics courses. This was addressed in some of the responses provided by educators in the online questionnaire:

- “Mathematical notions help (e.g. set theory) but vice versa, the mathematical notions can also follow learning database concepts".

- “Not enough knowledge about mathematics and logic”

The question that was posed to educators who participated in the online questionnaire: “Do you believe a solid grounding in set theory helps students understand database concepts?” The answer, as shown in Figure 5.7, was quite surprising with current SQL course design where 78% of participants agree, only 21% of them disagree and claim that they are teaching SQL without requiring a prior knowledge of set theory. The question that might be asked to those who claim that they are teaching SQL without previous knowledge in set theory is: how do students without set theory knowledge perform in the course compared to those with previous experience? This could be a focus of a future study.

5.4.2.3 Previous experience with database knowledge

The knowledge of some courses, such as students' background in RDBMS, database structure and Relational algebra, might also have some impact on SQL learner’s ability to master SQL concepts. Relational algebra courses are difficult to teach because they appear too theoretical, and do have an obvious practical application.

This was addressed in some of the responses provided by educators in the online questionnaire:

- "Because of the algebra behind it, students struggle with relational algebra but not the SQL itself".

- "Due to the lack of understanding of RDBMS".

According to Robbert and Ricardo [264], SQL is an essential component of an introductory database course, but there is less support for Relational Algebra. McMaster *et al*. [265], on the other hand, describe how database instructors can teach Relational Algebra and Structured Query Language together through programming. They suggested that students had better understanding of both Relational Algebra and Structured Query Language during writing SQL query. The next section discusses the SQL learner problem solving skills.

5.4.3 Learner's Problem Solving Abilities

Problem solving has been a focus of science educators for many years [266-268]. There is much research, in education in general, and computing education research in particular, that recommends incorporating problem solving as a primary process of teaching and learning [47-51].

Mayer [34] examined whether students can be taught strategies that help them to become effective problem solvers.

In this section, an insight into the factors that affect the novice SQL learner's skill in solving queries is given. The following subsections explore the effects of these factors.

5.4.3.1 Learners' strategy during Problem solving

Learners' strategies in solving a problem might be reflected in their success in solving SQL queries. According to Ramalingam *et al*. [269], students attempt to code a solution before planning how the problem can be represented using different SQL constructs. In addition, novices lack the ability to divide the

problem into sub-problems and identify the related knowledge that should be used to solve these sub-problems [270].

Some educators in this study confirmed that students spend less time in understanding the given problem and planning the solution and more time in other related issues such as identifying the syntax and semantic errors and assessing the correctness of the generated results. This means that students lack the effective strategies of how to solve problems.

In the absence of effective problem-solving strategies, students deploy a hit and miss trial and error tactic [270]. This attributes poor problem solving skills to the SQL novice and argues for more recognition of the importance of teaching problem solving within SQL instruction. Further research into whether students can be taught strategies that help them become more effective problem solvers is required. Mayer [34] explored whether problem-solving strategies should be taught as general courses or within specific subject areas. He suggested that it is best to have students learn problem-solving within the task students are expected to perform. Mayer's suggestion was followed in this research so that the decision was made to design SQL instructional materials that facilitate SQL problem solving. More details are given in chapter 6. The next section emphasizes teaching of problem-solving skills and exploring this kind of knowledge that is required.

5.4.3.2 Learner's type of knowledge and problem solving skills

Novices often lack necessary problem solving skills [271-273].This is not because they cannot solve computing problems in particular, but because they cannot solve problems in general [274]. Looking at this issue in the research literature, different reasons were given for this lack of problem solving skills in education in general and computer science education in particular [269, 270]. Spohrer and Soloway [275] observed that students might know the individual statement syntax and semantics, but they fail to combine different features to solve

problems. Table 5.7 reports what educators who participated in this study mentioned:

Participant 1 “SQL concepts are not difficult to understand or apply as an individual concept, but when you are given a complex situation where you have to apply many concepts then there is the problem”.
Participant 2“In general queries require different aspect of SQL to perform the request function”.
Participant 3“Knowing language syntax does not mean students will be able to use it in problem solving”.

Table 5.7: Participants’ Response Towards Problem Solving

Participant 1 statements is in agreement with Spohrer and Soloway, 1986a [275] findings. Felix [276], on the other hand, highlighted the relationship between the cognitive structure of a language and problem solving cognitive tasks. This might explain what the second and third participant tried to say. Moreover, some of the students in this study agree that they do not have problems in understanding easy SQL problems and solving a simple query, but their major issue is when they are presented with a complex one. There is a significant correlation at .005 between perceived ability to solve simple SQL problems and perceived difficulty in complex Query writing. Table 5.8 summarizes these results.

		I can read and understand SQL statements easily	I can only write simple SQL statements	I can solve a simple SQL problem
I do not have problems in writing a large and complex queries	Pearson Correlation	.322**	-.468**	.290*
	Sig. (2-tailed)	.005	.000	.016
	N	75	75	68

Table 5.8: Correlation Between Solving Simple SQL Problems and Complex SQL Writing

The above finding might lead to a focus on the content of the problems that students usually solve. According to Willingham [277]:

> "Working on problems that are of the right level of difficulty is rewarding, but working on problems that are too easy or too difficult is unpleasant". (P.10)

Since it is not possible to make all SQL problems easy to solve, is it possible to make solving the problem easier or at least giving students the skills to have a reasonable chance of succeeding? This justifies the need for research on identifying the teachable aspects of the skill of problem-solving transfer.

The educators who participated in this study also said that learners need problem solving skills to be able to use syntactic and semantic knowledge of SQL:

> "Learners need to attain the skills of critical thinking and problem solving skills and then learn how to apply the language syntax and details".

What kind of skills do students need to be able to solve problems? Bayman and Mayer [80] define the following knowledge involved in studying programming languages which can be applied to learning SQL as well:

- Syntactic knowledge: This can be defined as knowledge in a language features, rules and grammar; for example, the syntax of the SQL such as group by, Exists, IN.
- Conceptual knowledge: is knowledge of SQL language construct and principles.
- Strategic knowledge: learners' ability to apply syntactic and conceptual knowledge to solve a novel problem. This is called problem solving skills in this research.

This section explores how each kind of knowledge is related to SQL teaching and learning of SQL writing skills.

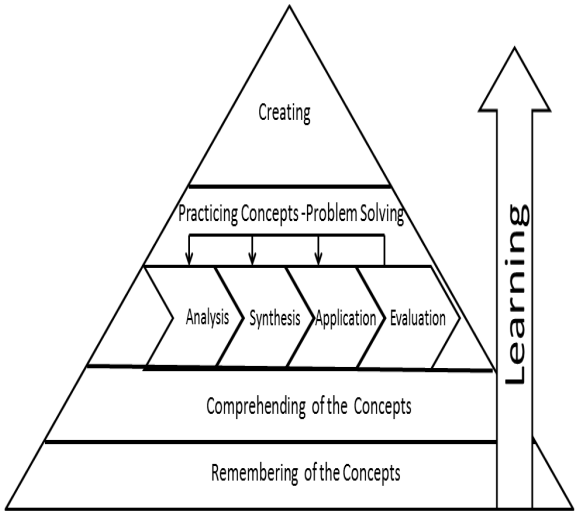


Figure 5.10 a:CS Learning Taxonomy

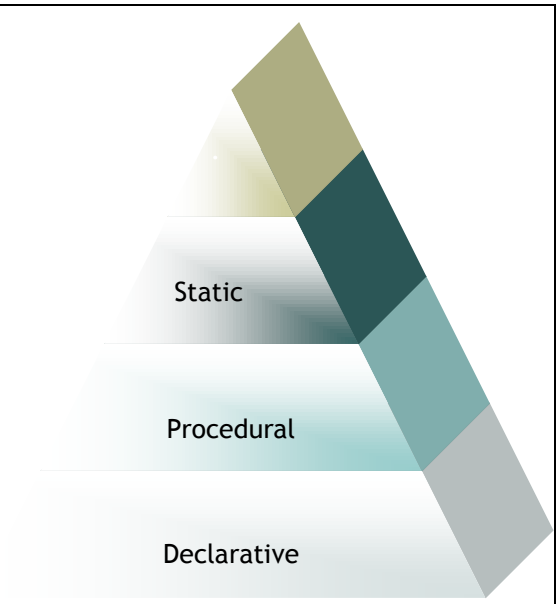


Figure 5.10 b: kind of knowledge in learning SQL

Figure 5.10: CS Learning Taxonomy (right) and its Related Kind of Knowledge (left)

Figure 5.10 presents the relation between different types of knowledge and CS learning taxonomy. *Declarative knowledge* is facts, concepts, or principles about something. Humans organize their declarative knowledge into a meaningful structure called schemata [278]. Applied to the domain of SQL, declarative knowledge refers to the SQL syntax and SQL principles such as the ability to explain a specific query command. *Procedural knowledge* is the active use of the declarative knowledge during problem solving. According to Anderson [78] model that applies to learning, declarative knowledge is converted into procedural knowledge after practicing and reflection upon examples, with more of these practice procedural knowledge become automatic and its use becomes less mentally. *Conditional knowledge*, on the other hand, is communicated "when the teacher explains to students why a strategy is important, when and where to use the strategy and how to evaluate its effectiveness" [79]. This relates to

"Create" in CS learning taxonomy. The next section investigates the learner acquisition of problem solving skills.

5.4.3.3 Learners' Acquisition of problem solving

Problem solving difficulties can also be related to an acquisition problem and not only a learning problem. Students manifest serious deficiencies when obliged to comprehend or write queries, which they encounter outside a helpful context, under exam pressure, or when it requires complex cognition.

Therefore, novice knowledge tends to be contextual rather than general. For example, from personal experience some students know what concepts mean in SQL and even how to apply them in a similar context that they have already seen, which indicates that they have the conceptual knowledge, but they have difficulties to apply those concepts in a novel scenario. Research suggests different approaches to overcome this issue. Some research developed problem solving models that show the relationship between learning the abstract knowledge [279, 280], which is decontextualized problem solving, and the transfer of it to other scenarios, which is used in different contexts. Mayer [34] suggests that the method of teaching problem solving should focus on modelling of the steps in the process of problem solving. He also insists that students practice relating their own problem-solving process to those recommended models.

Thus, it can be concluded that it is essential for students to solve problems in different contexts to build up the mental models they need to develop problem-solving skills. Greater emphasis should be given to the relationship between what is learned in the lecture and what is needed to solve the given problem in lab, and this has been a valuable contribution of the situated learning movement [70].

One of the factors that might affect students' problem solving skills is not being provided with sufficient opportunities to engage in more exercises throughout

the course they are undertaking, to which, in this respect, one of the students said:

> "The more problems I solve, the easier SQL becomes".

Some researchers suggest that teaching novices programming needs a lot of practice with basic material until they reach the level to automate these practices [281]:

> "To gain automation, it is probably important that the teaching process stresses continuous practice with basic materials to the point that they become overlearned". (p. 389)

This practice could help novices to develop a higher level of problem solving and avoid counter production tactical techniques like solving problems by trial and error. One of the educators participating in the online questionnaire said:

> "If SQL is not used even for a short time, the details of the language and the problem solving skills are forgotten".

This might be because of tactical problem-solving rather than development of strategic problem solving skills. It could also be related to the nature of SQL itself. Students need to actively engage in practical exercise in using SQL by using a well-designed instructional material that models steps in SQL problem solving. As a result, mastering the skills of understanding how to interpret the given problem and applying the correct solution is more likely to be achieved.

5.4.4 Summary of the Characteristic of Novice Learners

Based on learners' self-reports and experienced educators' comments, novice's performance in learning and using SQL is influenced by the following factors: personal attitude, previous experience, and problem solving skills in related

areas. Novice's personal attitude and determinacy toward learning and using SQL has some impact on their performance.

Moreover, novice's previous level of knowledge and experience in programming courses, mathematical background and the level of their acquisition in DBMS theory and concepts seem to contribute to poor performance in solving SQL problems and writing correct queries. It is possible to say that SQL learners might be characterized as follows:

- Students' negative attitude towards learning SQL may be due to lack of practice.
- Students are often lackadaisical, in terms of motivation, to grasp SQL knowledge.
- Students have no experience of its implication in the real world.
- Students lack knowledge of set theory.
- Students lack problem-solving skills.

Additionally, to the learner's characteristics, novices experience difficulties during query based problem solving. These are attributed to different reasons such us:

- Insufficient understanding of the concepts of different SQL constructs.
- Poor problem solving skills in general.
- Teaching instruction which emphasizes declarative knowledge which is 'what' and 'How' using a traditional teaching approach.
- Teaching instruction that does not guide students to develop problem-solving strategies, thus students lack both the ability to apply the required knowledge and the skills in solving diverse problem in many contexts, as was discussed in section 5.4.3.
- Not having the chance to be engaged in many exercises during the course had a negative impact on SQL learners in mastering SQL.

The next section discusses the influence of SQL features in learning SQL.

5.5 The Analysis of SQL Language Features

The influence of query language feature affects its 'ease-of-use' [4, 81]. SQL nature, syntax and content are different from other languages that novices learn in their courses. The question that this research is trying to answer is: to what extent do SQL nature, Syntax and content affect novice's learning?

5.5.1 SQL Nature

Some researchers argue that, to teach SQL, both learners and educators should understand the nature of the language and how it is different from or similar to learners' previous knowledge and skills. For example, the difference between programming languages and SQL, as highlighted by Sengupta and Dalkilic [263], was explored in section 5.4.2.1.

SQL is a non-procedural language; it "merely states what the result of the query is, not how to obtain it" [282] (P.84). Ramakrishnan [283] distinguished query language from programming language by identifying the purpose of each. Query languages should not be seen as programming languages; they serve different purposes. A query language is meant to be efficient and effective at data retrieval while programming languages perform computation.

In this study, participants related the difficulties of SQL to one aspect that is associated to SQL nature as a declarative language and not a procedural in terms of both its construct and purpose. Their comments are outlined in Table 5.9 below. Previous research confirms with what participants highlight. Welty and Stemple [81] compared SQL as a declarative language and TABLET as a procedural language and found that the procedural nature of the language affects the language 'ease-of-use'.

Educator	“Not enough practice or examples on concepts; courses emphasis is on how fast you learn things rather than how thoroughly they are learned”.
Educator	“SQL takes quite some time to master. In general, queries require different aspects of SQL to perform the request function”.
Educator	“SQL is a very "tight" and minimalist language, and is not procedural”.
Educator	“SQL is declarative and having a procedural mind set is easier”.
Student	“It is different to than other programming language. Logic behind it is different”.
Student	SQL problem is an inherent problem in a declarative natural language.
Student	“SQL is not the natural way of thinking”.
Student	“Because SQL uses command-line and there is no IDE program; also the compiler is real time then user tends to make a lot of mistake”, and “It’s too ambiguous too many ways to achieve the same thing, with no standard approach to problem solving”.

Table 5.9: Participant’s Response to SQL Nature

They conclude that "the concrete procedural model underlying the TABLET queries are missing in the less procedural SQL queries." Chapter 2 elaborated on the results of some studies, which compared SQL with other query languages.

5.5.2 SQL Syntax

The syntax of a language plays an important role in a learner's ability to use the language. SQL syntax is an important factor that needs to be focused on in terms of teaching and learning SQL. Knowing the syntax of a language does not mean that learners will be able to state a query explicitly [245]. The syntax is easy to state but hard to integrate into an inter-language [246]. The rigid demands of SQL syntax compared to the inexact and loose nature of the natural, or algorithmic language results in many students being unable to successfully write SQL [4, 57] as discussed in section 2.5.2. SQL uses a linear syntax that is written in normal left-to-right, top-to-bottom format [4].

In this study, the majority of students claim that they do not have problems with SQL syntax. Figure 5.11 below shows participants' feedback. Responses were measured based on respondents' feedback on a set of 5-option Likert scales: 1 "Strongly disagree" to 5 "strongly agree". The means was 2.3 and Std. Dev. was 1.046.

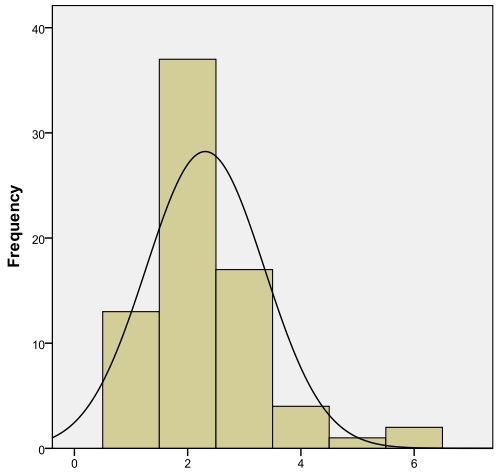


Figure 5.11: Participant's Response to SQL Syntax

Some participants argued that SQL is a simple language that uses simple English words and can be easily understood when the basic concepts are mastered. One of the students, who was currently working in a project using SQL and had positive feedback in learning SQL, said:

> "SQL is very simple... it's like English, once you understand what it is and how it works, it is for you to carry..."

Some learners found it easy to state the syntax of SQL concepts but very hard to integrate it into a real scenario. If so, then the problem is not SQL syntax knowledge but the synthesis skills as defined by Bloom et al. [37] that learners lack.

Knowing SQL syntax does not mean students will be able to apply it correctly in their course work or exams. They often correct errors by applying a 'patch' to the problem that allows a program to simply execute but this does not build a

deep understanding, nor do they know or can explain why it worked. This reflects how students perceive this issue in this study, which is solving SQL problems by trial and error. Although students claim that they do not have problems with SQL syntax, there was clear evidence that students are having problems when they try to apply SQL commands to solve the given query.

Some participants felt that there were issues with SQL syntax. This is shown in the relevant quotes provided by students and educators in Table 5.10 below.

Student	"I cannot see the relation between the statements and their context".
Student	"SQL syntax is quite difficult to remember…"
Educator	"Syntax details can be difficult; making concepts hard to embed in formal language", and "I think that it is because the ordering of the syntax (SELECT…FROM…WHERE) is not a natural way of expressing a query - it is more usual to identify the constraints first, and then work out what tables are needed, and then work out how to join them together."

Table 5.10: Participant's Response to SQL Syntax

If the SQL syntax order is not an intuitive way of expressing a query, then the way that SQL statement is explained to students should be in a natural way of thinking. Therefore, a further study in how to present SQL knowledge needs to be conducted aiming to present this theme: identify the constraints first, and then work out what tables are needed, and eventually work out how to join them together. This manipulation of presentation of SQL structure needs to be trailed and the results of such manipulation should be studied. The research might well deliver evidence about the effects of making the syntax sequence match the acquisition of the a natural sequence. The future research might suggest insight into enhancing of teaching SQL syntax to be close to natural way of thinking.

5.5.3 SQL Content

Fundamental SQL concepts are abstract in nature and have little or no real-world counterparts. Learners may not have sufficient preparation to grasp such concepts. Here, the investigation focuses on the issues of some difficult concepts in SQL; these difficulties are recognized as "SQL misconception" in [284]

As discussed in the literature presented in chapter 2 on SQL studies, some of the research reviewed indicates the type of misconceptions learner's face with SQL. Similar errors were identified later by Smelcer [285] study which shows that join clause omission was a frequent and troublesome error. Mitrovic [3] argues that grouping, join conditions, and the differences between aggregate and scalar functions are common sources of confusion. Borthick *et al*. [286] argues that SQL semantic errors include errors such as incorrect use of query operations or operands, missing parts of WHERE conditions, missing table-join conditions, and missing substring functions. An educator who participated in the online questionnaire highlighted Sub-query as one of the difficult concepts as was presented in section 5.2.3. More participants' quotes confirm that SQL semantics are hard to grasp are shown in Table 5.11 below.

"Not paying attention when learning the SQL basic. Understanding SQL concept",
"Some of SQL concepts are difficult to understand, working with complex queries is difficult"

Table 5.11: Participant's Response to SQL Content

In this study, both students and educators were asked to rate the difficulties of different SQL concepts using Likert scale (from very easy to very difficult). The results are shown in Figure 5.4 and 5.9. Accumulated results on the agreement on the difficulties of each concept are presented in Table 5.12.

SQL Concepts rating	students		Academic	
	confusing	Hard	confusing	Hard
Restricting data	14.29	0		0
Using group functions	71.43	14.29	27.3	18.2
Grouping rows using GROUP BY	42.86	28.57	45.5	36.4
Displaying data from multiple data	28.57	57.14	27.3	45.5
Using sub query	57.14	14.29	27.3	36.4

Table 5.12: Results on the Agreement on SQL Concepts Difficulties

The above findings confirm earlier studies by [4, 33, 57, 93, 285, 286] in which queries were classified as complex or simple. Simple queries include operations that use mapping, selection, projection, simple Boolean operations, and built-in functions. Complex queries include nested query, grouping, set operations, correlation variables, computed variables, and relational operators.

The research has confirmed the findings in the literature about the misconceptions in SQL, thus providing strong support for the validity of this study. Hence it can be argued that once students are able to understand these concepts, they will be able to learn and apply other advanced concepts. Therefore, it is crucial to focus the effort on clearing up the identified misconceptions.

5.5.4 Summary of SQL Features and their Relation to SQL Learning

According to Merrill [63]:

> "the careful analysis of subject matter content (knowledge) can facilitate both the external representation of knowledge for purposes of instruction (knowledge objects) and the internal representation and use of knowledge by learners (mental models)". (p.244)

To serve the aim of this research, an investigation on the nature of SQL concepts is essential. To summarize the above investigation, SQL, as a non-procedural language, describes the desired result without specifying how it is to be obtained. Step-by-step instruction achieving the result is not required by SQL compared to other procedural languages, such as Java. This might lead to difficulties when SQL is introduced to novices. Educators need to put more effort into explaining SQL's core differences. In addition, further study is required to investigate how SQL is taught, including the use of tools and teaching methods to ease SQL learnability.

5.6 Analysing the Influence of SQL Learning Methods and Approaches

Reisner [4] suggested that query language user performance could only be achieved by attending to the logistics in teaching and documentation of the language. Table 5.13 shows students' and educators' feedback about their experience in learning SQL and the issues in teaching SQL.

Educator	"Not enough practice or examples on concepts. Courses emphasis is on how fast you learn things rather than how thoroughly they are learned".
Educator	"If SQL is not used even for a short time, the details of the language and the problem solving skills are forgotten".
Student	"Important concepts are not explained in enough details, no margin of error".
Student	"We need more practice than theoretical view, not enough courses,…SQL taught badly".
Student	"Many ways to skin a cat, subtle difference between strategies". "Too many ways to achieve the same things".

Table 5.13: Participant's Response to SQL Learning Experience

In current SQL courses, from personal experience, students experience many difficulties in matching the knowledge learnt in lectures with the knowledge

required to solve SQL problems in the lab. Learners do not know how, when, or why to apply relevant knowledge. That is a result of not having experience in solving SQL problems and not building a mental model to support query solving. The collected data from participants yield some insight into learners' skills and knowledge in mastering SQL. This insight can be used to find out "what" the problems are that exist in the current teaching and learning approaches. In addition, some of the participants highlighted "how" these issues could be solved.

Learners cannot explain or provide understanding of "why" these problems occur. It is argued that it is crucial to understand why novices face such difficulties in order to provide solutions to address the identified problems.

In this research, understanding of "why" learners make mistakes during problem solving processes that involve different cognitive operators was discussed in section 2.3.2. This will provide insight into issues in the current approach of learning and teaching SQL.

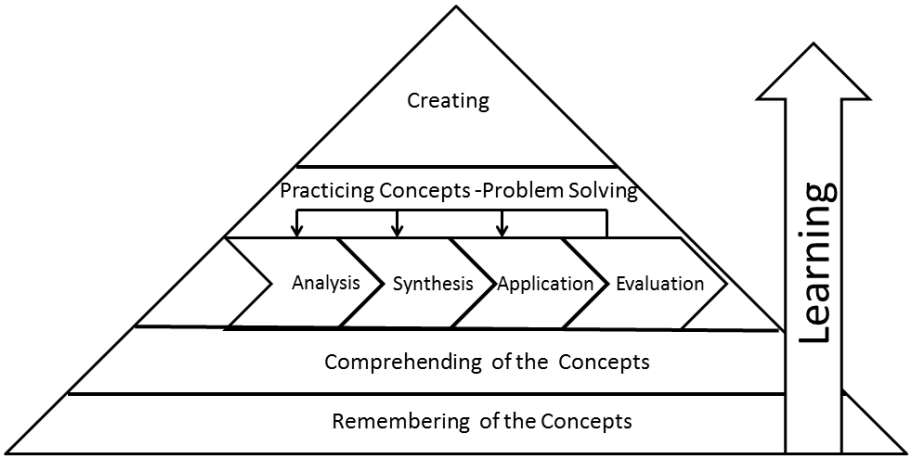


Figure 5.12: CS Learning Taxonomy

Does currently used teaching instruction help students to perform the analysis, synthesis, application and evaluation tasks? How easy is it for students to move from problem statement to output results? More exploration into students' activities during problem solving is urgently required.

The section examines students' ability to perform in two level of the proposed taxonomy in section 2.2.2 (Figure 5.12) comprehension and Practice.

1. Problem solving involves:
   - Analysis skills: Novices' ability in understanding the context when they are given the problem scenario (Query formulation).
   - Synthesis skills: Novices' ability to translate the given problem to database terms (Query translation).
   - Application skills: Novices' ability to write non-trivial SQL query. It is the application of their knowledge (Query writing).
2. Comprehension: Novices' skills in reading and comprehension of the given SQL queries when they need to explain in plain English the elements of the data and the related SQL concepts.

The following subsections explore these factors. However, before that, an overview of SQL curriculum design is provided.

5.6.1 SQL Curriculum Design

The sequence of the order in which concepts are introduced in SQL can have a major effect on SQL acquisition. There is a certain optimal sequence of learning of SQL concepts. SQL concepts build on knowledge and mastery of previous concepts. Various researchers [1, 261, 264, 287-293] studied the design of teaching database management system courses. Some research pointed out that the first chapter in Database textbooks is often 'dry'; this is because of the needed emphasis on defining the key terms in the field [261]. They added that the biggest challenge for the Database instructor is to enliven the first chapter through some interesting exercises.

To understand this issue in depth, studying SQL course design is the first stage that needs to be carried out. After a quick analysis of SQL course outline designs that are available to the researcher, it was clear that there are variations in the course design from one institute to another. Some courses are designed to

introduce *Data Definition Language* (DDL) before *Data Manipulation Language* (DML) while others emphasize on teaching DML only. For this research, it is not quite clear which approach is better and this is not considered extremely relevant to the goals of this research, and could be investigated in further future studies.

However, the researcher is not aware of any work that considered an optional sequence in the acquisition of SQL concepts or any empirical study that examined the effects of any specific course design in database. Most of the published studies deal with the course content, teaching methods and teaching tools. The *Water Fall Model* of teaching that applies learning taxonomies such as *Bloom Taxonomy* [53] might not be as effective since the sequence of teaching the knowledge is ignored. The researcher suggests that SQL course designers and educators might look at techniques and approaches that were used in computer science course design research, which considered the sequence of the course concepts and skills that learners were supposed to learn within the course.

In other words, they need to apply *Constructivist Learning Theory* or *Scaffolding techniques* [294]; for example, *Scaffolding* used as a building block approach to learning and is currently advocated by educators within the field of computer science [295]. Mead *et al*. [296] presented a formal structure, *the Anchor Graph*, which facilitates curricular planning and provides a context within which the anchor concept idea is based, which integrates and transforms earlier knowledge. The structure of an anchor graph is based on the idea that an anchor concept with a direct link to another anchor concept carries cognitive load for learning the new concept.

In addition, it is suggested to create different sets of SQL course designs that are based on *Spiral Model*. Empirical research needs to investigate the effects of different sequences in teaching. Another influence on the acquisition is learner's variation in the acquisition sequence. In addition, this acquisition might be related to the individual level of knowledge and problem solving skills.

The focus of this research is to identify the probable causes of poor learner performance. The next section examines learners' ability to solve SQL problem.

5.6.2 SQL Query Comprehension

Code reading or walkthrough are important skills to novices in learning program [233]. Query comprehension cognitive tasks might involve different tasks such as query reading, query explanation, and printing out the results [7]. Students' ability to comprehend SQL was investigated in this study as one of the factors that affect learners' performance.

One of the teachers participating in the online questionnaires mentioned that students could not write SQL "because they cannot/they do not know how to read SQL statements". There is a need to determine the extent to which students lack this skill. The task used in this research focused on participants' ability to comprehend SQL statements. The task involves Entity Relation Diagram (ERD) and SQL where the participants were asked to walk through the SQL commands and explain what output the SQL query command is intended to produce.

From this task, four core issues have emerged:

1. Some students were not able to either read or understand SQL statements, although they have completed some courses in SQL.
2. Some students have the ability to understand the SQL statements or a portion of it subjectively where they explained what the statements meant to do but they were unable to correctly predict the outcome of query statement.
3. There is a highly significant correlation between the number of courses that students studied and how they rate their skills in solving SQL problems at ($p < 0.01$).
4. There is no significant correlation between students' scores in the task and the number of courses and how students rate themselves in terms of

how skilled they are in solving SQL problems ($p > 0.05$). This means that students self-rating are not necessarily a reflection on their actual skills.

However, it can be argued that reading and understanding SQL statements are not a major or severe issue since only 15% of students were unable to give explanation to the given code in both experiments while 28% were able to give a correct explanation and print the right results. Nevertheless, the given SQL statements were perhaps simple or the covered concepts were easy for students to understand.

5.6.3 SQL Query Formulation and Translation

The first step that learners need to do during problem solving is query formulation. To what extent can students understand the given SQL problem and express how to solve it by applying their knowledge and skills?

SQL novices experience some difficulties in deciding about the data needed to solve the problem. This is attributed to learning context and generalization, which affect learners' performance. To emphasize on the content, when learners have a wide experience with a range of contexts, this can facilitate recognizing relevant information for generalization [297, 298]. Some educators who participated in this study provided the following responses, as shown in Table 5.14, which relate to the above discussion:

Educator	"Students are not practicing with real data and real examples".
Educator	"Students' ability to understand in a meaningful context exceeds students' ability to grasp decontextualized scenario and to give a solution".
Student	"To understand the SQL concepts, you need to work on a well understood set of data. Problem with lecture is that data are artificial and students might not understand the relationship".

Table 5.14: Participant's Response in Relation to SQL Query Formulation and Translation

To investigate this, a cognitive task was used, and the data presented in section 5.2.2. It shows that many students could say what they are supposed to do but not how. Many of the participants stated that they need to query from the same table and three of them said that self-join concepts need to be applied. However, only one was able to show how self-join could be used to solve the question.

One of the educators interprets the above finding:

> "Concepts are the problem (and they appear to get it right in English, only because English is subject to interpretation). If not, then syntax (translation into a formal language) is the problem."

The literature suggests explanation for this, according to [13]: failing to understand SQL concepts is behind the novice's ability in query translation. Chan [92] conducted an experiment that measures query performance in both query translation and query writing. He concluded that users could understand the relational model, but have difficulty in expressing the required operations in SQL.

Solving any problem is easy when it is supported with a concrete context [299, 300]. Accordingly, educators should avoid using data that is not easy to grasp, such as medical data or some statistical information that might be offensive to a specific culture or religion. Furthermore, the given scenario should set up the learners' expectations and it might be useful to relate problem to pre-existing knowledge so that learners' response to the problem might be more effective. Moreover, one might say that students appreciate learning more when what they are studying is of personal interest and relates to their daily activities. Hence, educators must comprehend that learners are using their real world knowledge of connecting when they are learning SQL and avoid any artificial data that is far from learners' context.

This confirms that SQL learners go through some challenges in deciding about the data needed to solve the problem and the process involved in solving the problem. Without a doubt, one could argue that contextualization is a vital factor that influences novice's ability in query formulation and translation and, when learners have a wide experience with a range of contexts, it can facilitate recognizing relevant information for generalization [297, 298]. SQL educators should emphasize the importance of using meaningful data (tables, columns and rows), and a scenario close to learners' environments when teaching them SQL. In addition, instructional material that facilitates exposing learners to a wide range of context need to be developed.

5.6.4 SQL Query Writing

Query writing is a stage where learners need to apply their knowledge of SQL syntax and form to the given scenario. It is crucial to find out errors novices make, and to classify them. Many students admit that they solve by trial and error. This study considered novice's skills in writing queries an important factor that affects their performance. Knowing what strategies novices deploy in query writing, and the errors they attempt might help in coming up with ways of addressing this issue.

Buck and Stucki [301] pointed out that the reasons for trial and error tactics can be attributed to course designs focusing on "writing code" which are application and synthesis skills and do not emphasize comprehension and analysis skills.

The result of the cognitive task revealed that only one student out of seven was able to write a correct query. The participants' failure in writing the related SQL query was discussed with educators who participated in the online questionnaire. Some educators related students' failure to the nature of the problem, while others related it to the content and the SQL concepts that were covered. This is evidenced by the quotes obtained during the online questionnaire by educators, as shown in the Table 5.15 below.

"The solution to the question requested students to query from the same table twice which is not logical and students cannot see. 'Their managers' requires that two copies of the same table be joined - they will not see this as the obvious thing to do at first".
"Although it seems to be a simple query, it is not; for beginners it is confusing to compare rows in the same table".
"Self-join concept is one of the hardest concepts for students to conceive".
"I would classify this query as fairly hard, as it requires a self-join, therefore an alias, and these are not used widely and confusing, because you need to clearly understand which version of which attribute you need to refer to at each point".
"Easy to write SQL which is just a manipulation of the select statement. Self joins require an understanding of the underlying structure. Also, of course, depends on their previous experience with SQL".
"Translation into a formal language is the problem".

Table 5.15: Participant's Response in Relation to SQL Query Writing

Table 5.16 below shows the relation between different characteristics of SQL learners, such as: the ability to comprehend SQL, writing simple and complex queries and solving simple SQL problems.

		I can read and understand SQL statements easily	I can only write simple SQL statements	I can solve a simple SQL problem
	N	75	75	68
I do not have problems in writing a large and complex queries	Pearson Correlation	.322**	-.468**	.290*
	Sig. (2-tailed)	.005	.000	.016

Table 5.16: Participant's Response in Relation to SQL Query Formulation and Translation

The results showed that there were a significant correlation between writing complex query and reading SQL (query comprehension). Query writing is an important factor that influences learners' query performance. Thus, SQL teaching materials need to focus on providing learners with the required knowledge in an effective way.

5.6.5 Summary

According to Reisner [4], SQL involves cognitive activities such as learning, understanding and remembering. The influence of the teaching methods and approaches in learning SQL that might predict success and failure in an introductory SQL course was discussed. Different diagnostic tasks were used to explore novice skills and knowledge. The cognitive factors that were evaluated consist of students' ability to understand and analyse the given scenario (query formulation and translation), students' ability to write non-trivial query (query writing) and students' skills in reading and comprehension of SQL queries (query comprehension).

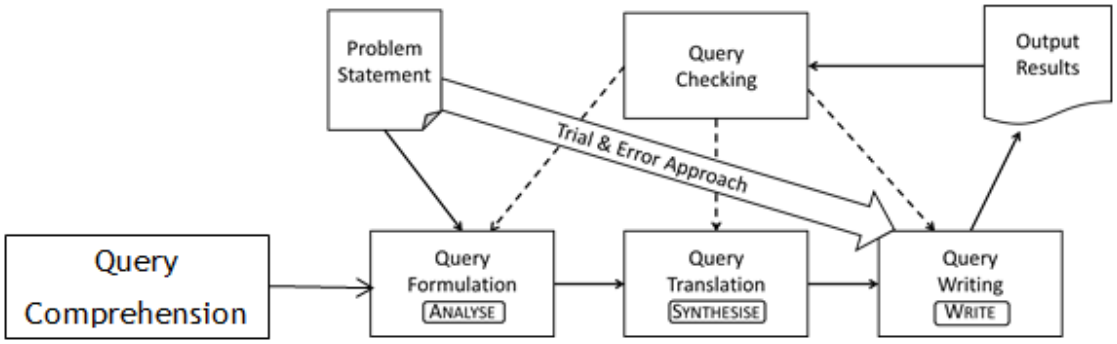


Figure 5.13: cognitive Model of Query Learning

Figure 5.13 presents a cognitive model of query learning which has been derived and confirmed by these studies. This model consists of cognitive tasks: Query comprehension, Query formulation, Query translation, Query writing and Query evaluation. The model combines the two models discussed in section 2.2.3: Ogden's [302] three-stage cognitive model of database query and Mannino [58] model of two steps.

The model suggests that learners should accumulate a wide experience with a range of contexts through query comprehension exercises, so their ability in query formulation and translation is improved.

5.7 Discussion and Interpretation

The aim of this discussion is to bring the highlighted factors together and propose more effective instructional material, which will better support the learning process by aligning with learner cognition and staged SQL acquisition processes. The discussion may start with the learning content such as SQL syntax and semantics. SQL features might be tackled as one of the reasons that affect novices in learning SQL as was discussed in section 5.5. It is possible to say that, SQL syntax and semantic have some influence on novice's performance. SQL is a declarative language that shows the desired result without specifying how it is to be achieved. Its main purpose is to retrieve data. The ordering of the syntax (SELECT…FROM…WHERE) is not a natural way of expressing a query; it is more usual to identify the constraints first, and then work out what tables are needed, and then work out how to join them together. The way that SQL statements are explained to students should be in a natural way. Syntax details' difficulties can have some impact on query writing by making concepts hard to embed in formal language.

Since SQL is a declarative language, more care needs to be taken when introduced to students. The study revealed some agreement on SQL misconception, such as different type of joins, nested query, and relational algebra, grouping function. The researcher believes that there is a relationship between the determinacy and the nature of SQL learning and this is one of the issues that might be worth considering for future studies.

SQL novices, on the other hand, are characterized by lacking the conceptual knowledge; i.e. a rich understanding of the language construct and the way in which they are used to solve problems [60]. They also lack strategic knowledge; i.e. the ability to apply syntactic and semantic knowledge to solve novel problems [61].

Isolating the teaching of the course core material and the problem solving strategy might have an effect on SQL skills acquisition. Thus, teaching approach and teaching material can address some of the factors that affect novices in learning SQL. The used instruction should aim to help learners know how and when to apply and foster development of different types of knowledge: *declarative knowledge* and *procedural knowledge* [78], *conditional knowledge [79]*, and the *syntactic, conceptual and strategic knowledge* [60, 80].

Research suggested that there is a need of integration between the teaching of problem-solving skills and the course material [49, 303] to provide an effective means of teaching transferable problem solving skills. According to Lockhead: "We should be teaching students how to think; instead we are primly teaching them what to think". [304](p.1)

Problem-solving teaching methods should focus on the modelling of the "process" steps rather than on the "product" and by giving students practice in comparing their strategies to those of models [34, 305]. As was discussed in section 2.6.2, this is related to analogical problem solving.

However, from the different research methods that were applied in this research, it is obvious that the majority of current teaching and learning approaches do not apply or encourage analogical problem solving. That is why many subjects deploy trial and error tactics. It is possible to argue that students suffer from an inability to recognize the context of the problem and do not develop the ability to abstract knowledge. Thus, students solve problems by trial and error by mapping random SQL concepts and knowledge to the solved problem without using any wisdom or strategy. In other words, this could be related to the contextualization of learning. The question is: to what extent should learning wholly tie to a specific context? According to *Anderson et al*. [70]:

> "If knowledge is wholly tied to the context of its acquisition, it will not transfer to other contexts. However, even without assuming

> extreme contextual dependence, one could still claim that there is relatively little transfer beyond nearly identical tasks to different physical contexts". (p. 3)

Nonetheless, one might argue that any instruction designer should have a balance between concrete and abstract instruction. In addition, they need to make the right decision about when narrower or broader contexts are required and when attention to narrower or broader skills is optimal for effective and efficient learning.

Information can be obtained via a click in any search engine. In contrast to experts, novices lack the domain knowledge [306] and the ability to recognize familiar context [307]. This suggests that students do not engage in analysis and synthesis: they move straight to application. On its own, application does not provide meaningful practice which leads to construction of schemata.

The question here is: how to build such skills and how to help students to recognize problem types or problem structures when they are solving new problems. Figure 5.14 presents SQL problem solving model based on this research. This model is an improvement to the suggested one in Figure 2.10. As was discussed in section 2.3.2, based on the cognitive tasks, the learner needs to perform in order to solve a query while at the same time building mental model and deep understanding question.

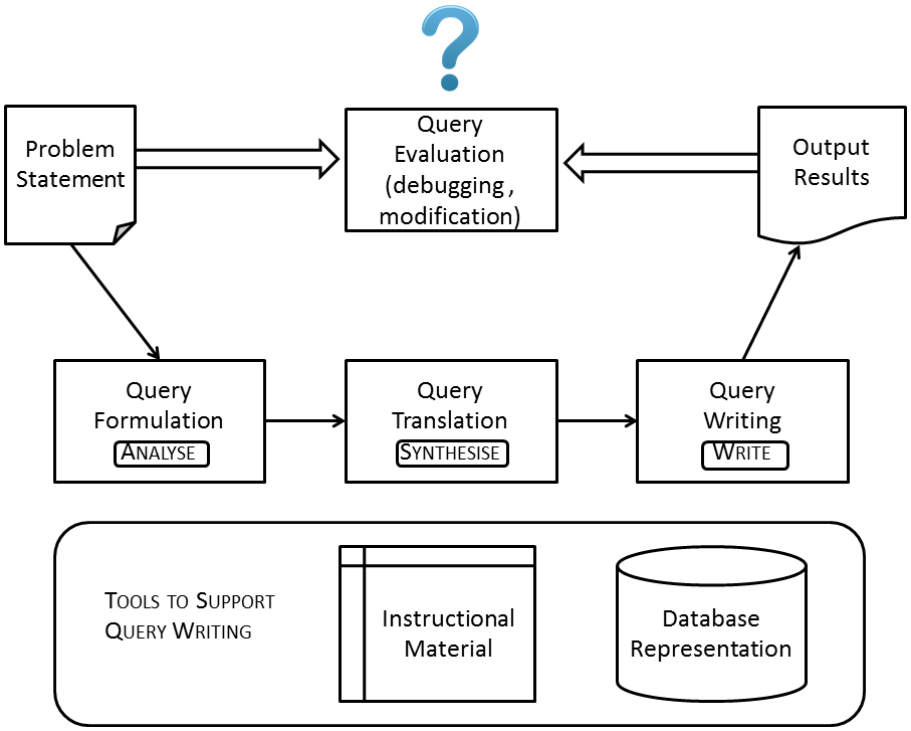


Figure 5.14: SQL Problem Solving – The Practice Stage of SQL Learning Taxonomy

To support this process, it is critical that educators work towards designing instruction that overcomes the identified issues in a way that makes them readily applicable and that student assimilate new knowledge in an efficient and effective way. *Learning efficiency* can be defined as the speed in which novices gain skills and knowledge that enable them to perform at an expert level. Thus, SQL needs to be taught using a well-designed instruction that:

- Takes account of individual differences in learners.
- Ensures that learners develop both a rich repertoire of syntax and semantics. It should be based on an informative approach that covers what, how, when and why.
- Ensures that learners focus predominantly on meaning; SQL learning process should provide a rich knowledge basis.
- Ensures that learners also focus on understanding; explore step-by-step mapping build a previous well established knowledge supported with text and diagrams.

- Based on human cognition that develops problem-solving strategy throughout the involvement of student's skills in problem solving.
- Emphasizes teaching problem solving strategy.
- Balances between abstract and concrete knowledge.
- Facilitates transfer and builds proficiency.

The significance of this study includes the large number of participants and the use of diverse and generalized stimuli. The study combined different research methods and approaches and collected both quantitative and qualitative data. Therefore, it provides opportunities to compare the different factors. The limitation of the study arises from the use of different participants in some of the study methods.

This discussion formulates the SQL learning performance objectives. The next section presents a model of SQL learning which maps the design of SQL instruction.

5.8 A Model of SQL Learning

The purpose of this research was to identify the probable reasons for a performance gap and to provide guidelines for query language teaching. This, in turn, informs educators about the major issues in teaching and learning SQL by highlighting the factors that affect ease-of-learning of SQL from both the literature and the application of different research methods, outlined in this chapter.

The reported results provided an analysis of both learner characteristics and learning context. As a consequence, SQL learning performance objectives can be structured. The objective should provide a map for designing the instruction and for developing the means to assess learners' performance [35]. The SQL framework model presents the objectives to be used as a map to facilitate the Instructional Design objective.

The framework proposed in this research is based on a new interpretation of mastering SQL as being influenced by cross-cutting human factors, nature of SQL and domain of SQL itself, learning theory (SQL learning taxonomy) and cognitive science (development of mental model throughout the learning process).

The framework aims to inform the design of SQL instructional material that will motivate students to respond positively to learning by internalizing it and making it part of their personal set of moral and ethical principles, so that they automatically behave according to its precepts, even under challenging circumstances. Figure 5.15 presents the proposed framework structure. The research employed in this chapter works toward confirming the proposed framework.

The framework consists of three main areas:

1- Cognition and mental models area: presents the development of mental model throughout the learning process.
2- SQL learning taxonomy area: illustrates the proposed learning taxonomy, which relates to SQL knowledge and skill acquisition. In addition, each part is related to the learning objectives.
3- SQL cross-cutting factors area that highlights the factors that influence the ease-of-learning of SQL.

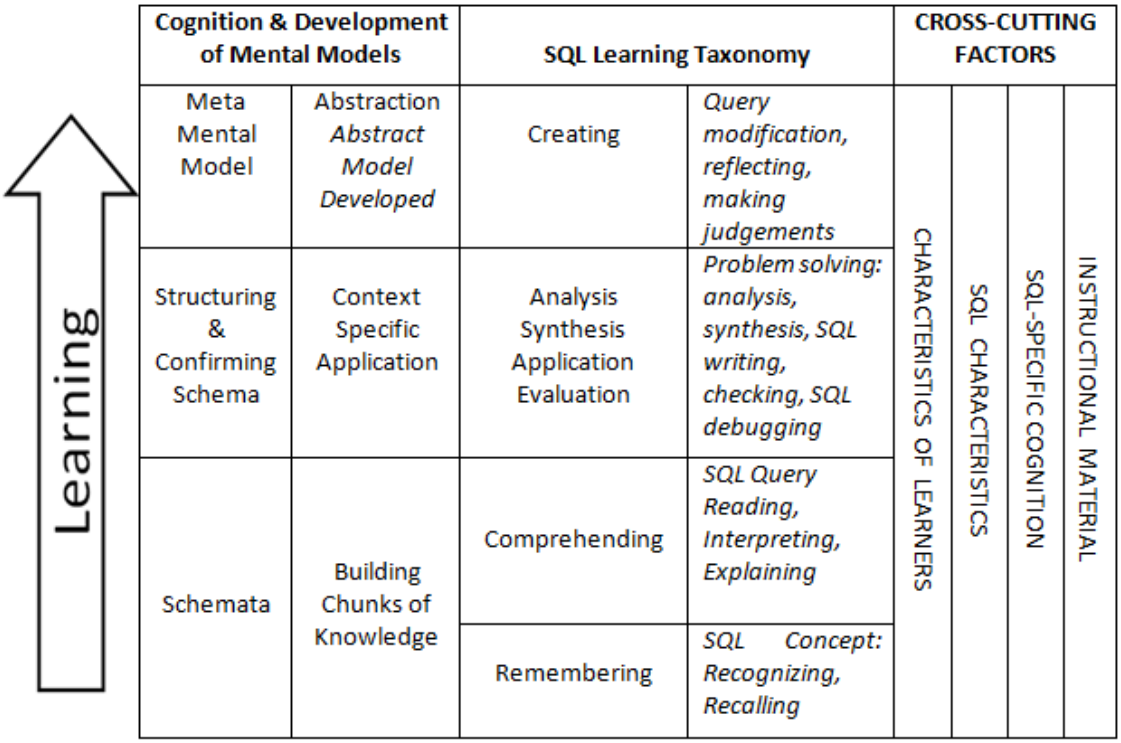


Figure 5.15: A Model of SQL Learning

To expand the discussion of the proposed framework in details, each area is explored individually in the following subsections.

5.8.1 Cognition and Mental Models

To learn SQL, the first dimension to look at is the supporting cognition principles. According to Robins at al., "learning" means the construction of *schemas* where a schema is "*a structured chunk of related knowledge*" [260]. Learning either constructs new schemas or modifies and combines existing schemas in order to produce new, more abstract schemas. A *mental model* is made up of a schema plus the cognitive processes for manipulating and modifying the knowledge stored in a schema [63]. In order to have a mastery of SQL, query writers draw on a mental model which is constructed from the requisite concepts (syntax and semantics), together with an understanding of how to apply the concepts within a particular context.

Knowledge of SQL syntax and semantics, on their own, is not sufficient to achieve mastery. Many students can parrot such knowledge in exams yet do not know how to apply it. What they appear to lack is an abstract construct, which can be applied to matching contexts. The mental model, with its schemata building blocks, is constructed when learners write SQL as part of a problem solving process. This process is depicted on the left of Figure 5.16.

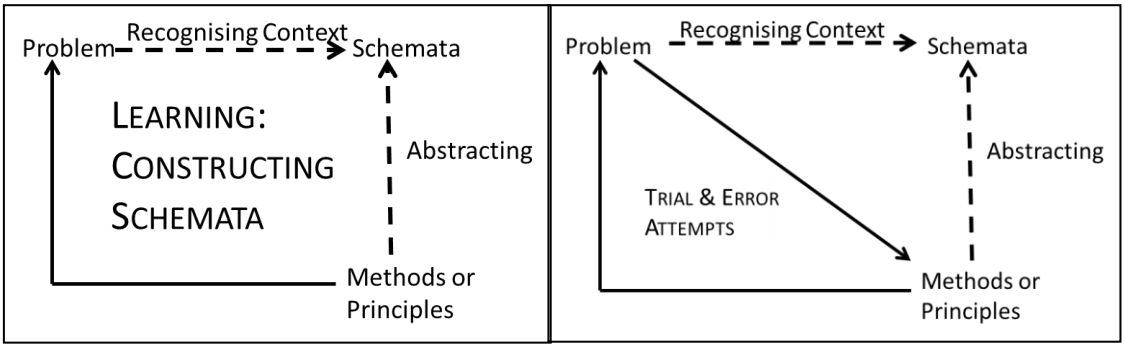


Figure 5.16: The Role of Schemata in Problem Solving (left) and a Trial and Error Approach (right)

This diagram depicts an ideal situation, where schemata are constructed during problem solving. What it does not convey is how the process can become derailed, and how best to ensure that problem-solving results in schemata formation.

In our combined years as database lecturers, we have frequently observed students engaging in the process depicted on the right of Figure 5.17. These learners inhabit the bottom left half of the triangle without entering the upper where deep learning can occur. The net effect is that schemata are not formed: for whatever reason, learners experience difficulties matching the knowledge learnt in lectures with the knowledge required to solve SQL problems. They do not know how, when, or why to apply relevant knowledge. They then solve the posed problems using a trial and error approach, trying various constructs successively without any perceivable strategy and without developing any deep understanding of the underlying principles.

It is essential to understand why this is happening, and then to design the appropriate teaching and instructional materials so that they align better with human cognition. This will help to consider how SQL *should* be taught so as to maximise the learner's opportunities to build the schemata that are required to achieve mastery.

What was highlighted at the beginning of this chapter is that if students get the chance to solve more problems, their skills will be more likely to improve. Moreover, having a wide experience with a range of context (schemata) can help learners identify relevant information (abstracting) for generalization. Hence, generalization issue is one of the issues that SQL novices suffer from and which can be overcome through an analogical problem solving approach. This leads to another diversion of the discussion that focuses on the transfer approach (see Figure 5.17). Cognitive scientist describes transfer as: "applying old knowledge in a setting sufficiently novel that it also requires learning new knowledge". [308] p283

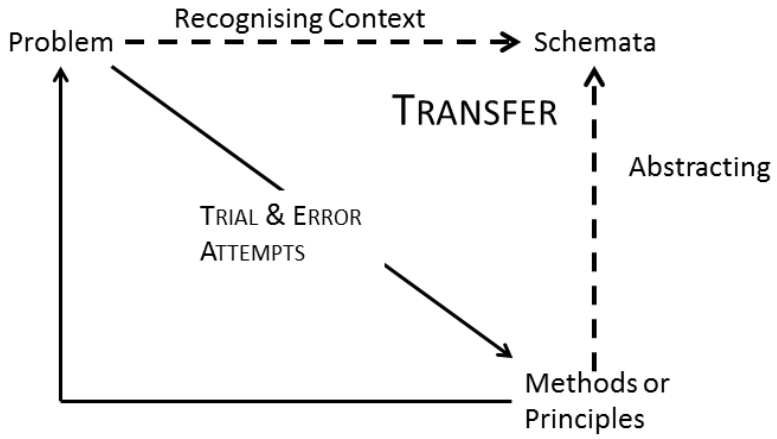


Figure 5.17: Transfer Approach

Transfer might be achieved by overcoming the deficiency in understanding why and when each concept should apply. This can be related to two factors: learners' ability to extract the required information from their own schemata (if there are such schemata) and the teaching instruction that lacks the design which focuses on the transfer. There is a lack of research on the effectiveness of SQL teaching materials and approaches. Hollingsworth [309] suggested teaching

undergraduate introductory classes on SQL query writing using *informed instruction* as in Bruer's school of thought [62] instead of traditional instruction.

5.8.2 SQL Learning Taxonomy

According to Cutts [41], learning taxonomies provide researchers with an essential shared vocabulary. Probably the most widely applied taxonomy was proposed by Bloom and Broder [305]. Anderson *et al*. proposed an updated version of Bloom's taxonomy to correspond with the ways learning objectives are typically described as cognitive activities [38]. They argue that those students' progress through Remembering, Understanding, Applying, Analyzing, Evaluating and, finally, Creating stages. Gorman [39] proposes a simpler model, arguing a progression from an understanding of *what*, followed by *how*, then *when* and finally *why*.

The applicability of a number of learning taxonomies to Computer Science has been considered [42] [43-45]. Lahtinen [45], in particular, investigated whether a subject-specific taxonomy would be of more use to computer science instructors than the existing generic ones. Lahtinen concluded that Bloom's cognitive activities were indeed applicable to Computing generally. Hence, there is some justification for applying them to SQL learning. Since Computing is essentially a skill-based subject, the three stages of Bloom, which constitute application of principles, are particularly important. These stages reflect the fact that problem solving is the essence of mastering computing skills. The fact that researchers recommend incorporating problem solving as a primary learning activity confirms this [47-51].

One could argue that solving problems and producing an effective and efficient solution are the core activities of the Computer Science practitioner. Computer Science, at its core, involves modelling the real world, representing domains of the most varied nature and complexity, representing knowledge in general and dealing with processes and solutions to problems in such domains. Therefore, any

proposed taxonomy should have, at its core, problem solving, to be engaged in after the basic knowledge is delivered and comprehended.

Therefore, the research proposed SQL learning taxonomy, which models how learners should assimilate specific SQL topics. It consists of four main areas, each of which map to the related cognitive processes:

- *Remembering SQL Concepts*: includes the following cognitive process: Recognizing and Recalling.
- *Comprehending SQL Concepts*: includes the process of SQL Reading, Interpreting and Explaining.
- *Practicing SQL* (problem solving): this level consists of three interleaving activities that represent SQL problem solving:
  - Problem formulation (Analysis),
  - Problem translation (Synthesis): Differentiating, Organizing, Attributing and translation of the given problem,
  - SQL Query writing (Application): Executing and Implementing query, and
  - SQL Query checking (Evaluation): Checking, Critiquing and evaluating the output results. For example, query debugging.
- *Creating*: includes reflecting, making judgments, and conceivably constructing mental models. For example, query modification tasks.

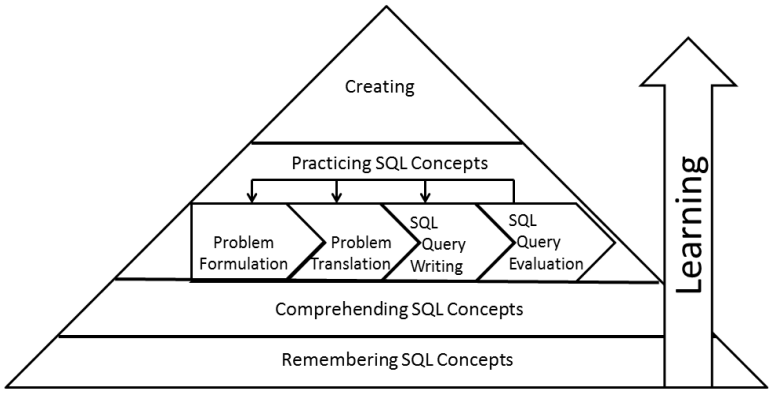


Figure 5.18: SQL Learning Taxonomy

This entire process will undoubtedly be affected by factors other than human cognition. The next step in understanding the SQL learning process is to look at the factors that affect SQL learners in general.

5.8.3 Cross-Cutting Factors

The SQL learning taxonomy, proposed in Figure 5.20, depicts a perfect learning model. It incorporates knowledge of human cognition, but does not necessarily accommodate individual learner differences. To identify the other factors that influence SQL learnability the research examined the SQL learner and learning process by applying different research methods (section 5.2). Grounded Theory was used to identify the factors in the responses, related to success in learning SQL (see section 5.3).

The emerging themes were:

- Learner's attributes: such as personal attitude, previous experience, lack of problem solving skills and general skill acquisition abilities.
- The features of the SQL language.
- SQL-specific cognitive tasks involved in the problem-solving process.
- The instructional materials provided during teaching activities.

5.9 Chapter Summary

SQL is a declarative computer language and is the de facto dominant database language. SQL knowledge and skills are essential to anyone working in IT. This chapter has explored a number of factors that might influence success in learning SQL. The study used various research methods and different participants to articulate these factors. Having looked at the evidence available to this study, the research confirms some general observations. Novice learners tend to incorporate real world experience and skills in their learning to understand or write SQL statements, without taking into consideration SQL nature. Developing a concrete mental model and understanding of the purpose

of SQL, how SQL is designed and a concrete understanding of how SQL statement is executed are important issues that need to be tackled in terms of learning SQL. Since SQL is a declarative language and does not require a procedural mind-set, more care needs to be taken when it is introduced to novice students.

At the end of this part of the study, it is possible to argue that learning SQL demands a high level of knowledge. Learners require the knowledge of "when" and "why" which is the informative teaching approach [62]. They lack the meta-cognitive skills where they have to identify, analyse, plan and give the correct solutions. To aid teaching for novices, it is necessary to identify the problems such as misconception in learning SQL, then identify the cause of the problems and finally suggest, implement and evaluate the proposed teaching method. This chapter identified the problem and suggested a new approach to consider the discussed issues.

In the next chapter, chapter 6, the researcher proposes a new ID as a teaching approach (learning SQL using SQL patterns). These patterns are designed based on the suggested framework that aligns the highest cognitive dimension. Chapter 7 reveals the effectiveness of SQL patterns on novice performance.

Chapter 6
SQL Pattern Design & Development

6.1: introduction

6.2: The Motivation for Using Patterns as an Instructional Material

Research Question 2

6.3 SQL Patterns Identification Process

6.4: SQL Patterns Structuring Process

6.5: SQL Patterns Organization Process

6.6: SQL Patterns Evaluation Process

Research Objective 1: SQL Patterns

6.7: Chapter Summary

Chapter 6: SQL Patterns Design and Development

This chapter presents the design and the development of SQL instructional material that employs the knowledge obtained through research on pattern research as discussed previously in chapter 3. Patterns traditionally structure knowledge in such a way that they can transfer best practice from experts to novices.

6.1 Introduction

In this chapter, there is a need to discover how best to structure the knowledge within the pattern description, and when it should be introduced during the learning process. It cannot be assumed that SQL patterns can be structured in exactly the same way as other more well-established patterns, so it is essential to carefully align them with what have been learnt about the SQL acquisition process in the previous chapters.

Chapter 5 reported the SQL performance objectives through development of the model for SQL learning (see Figure 6.1), which is the logical place to start when identifying SQL patterns and positioning them within the learning process. This model is grounded in Bloom's taxonomy [37] and is validated by studies of how novices learn to write SQL queries. Under the heading "Cognition and Development of Mental Models", it was demonstrated how mental models are constructed – starting with the development of individual schemata, moving on towards a meaningful structuring of schemata into hierarchies and constructed mental models.

The existence of these models suggests that the learner will be able to solve a variety of problems of similar nature; i.e. they have abstracted the core

principles and are able to apply them in many contexts – commonly referred to as heuristics. The SQL acquisition process is modelled in the model (see Figure 6.1), showing that learners need first to have a basic knowledge of SQL concepts, and an understanding of how to use them. They then have to practice applying these concepts to a variety of problems: analysing, synthesizing and evaluating. They ought to emerge from this stage with an appreciation of the core principles, with an ability to make judgments about strategies to be deployed. Learners who have progressed up to this upper level can be considered to have mastered SQL.

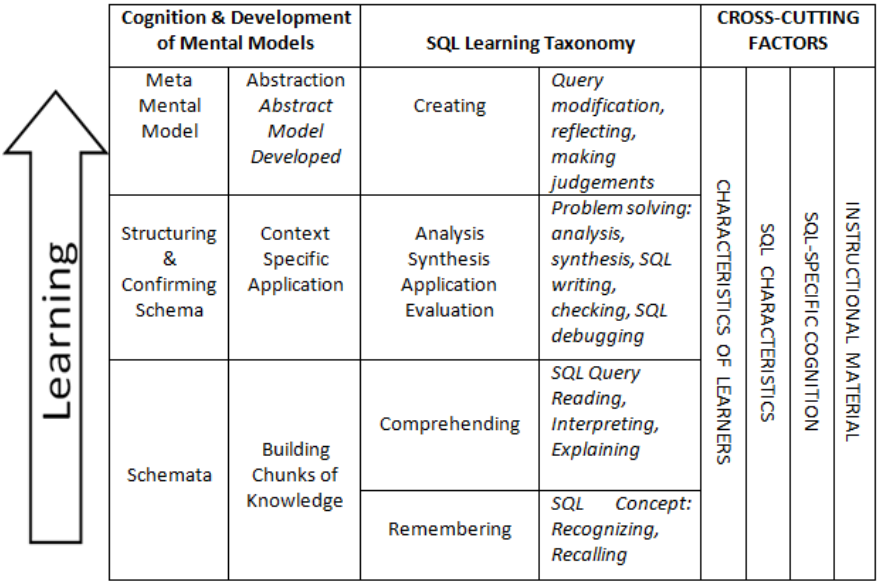


Figure 6.1: A Model of SQL Learning

Chapter 3 reports the application of patterns in different fields such as architecture, SE, HCI and pedagogical. Here, another kind of pattern is presented which is SQL pattern. SQL patterns are similar to other patterns in one important respect. In the same way that anyone can apply standard code design patterns in programming languages, he/she can also deploy patterns in writing SQL. Therefore, this research introduces a set of SQL patterns specifically tailored to help the novice learner to master SQL skills. This chapter is embarking on the development of different research and methodology in order to develop SOL patterns and to obtain more empirical evidence of their efficacy in learning SQL.

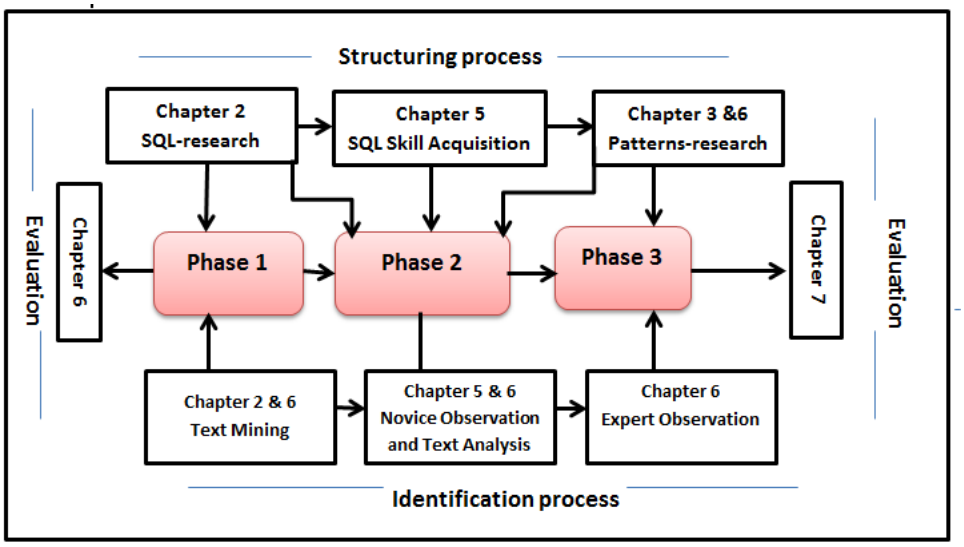


Figure 6.2: SQL Patterns Design Process

SQL patterns design process was developed in three Phases, which are covered in this chapter. It also covers the wisdom behind SQL patterns format and organization approaches. Section 6.2 describes the intent and the motivation behind SQL patterns. SQL patterns' identification process is explored in section 6.3. The processes that involve SQL pattern structuring and organization are explored in section 6.4 and section 6.5. The SQL pattern evaluation phase is discussed in section 6.6. Summary is given in section 6.7.

6.2 The Motivation for Using Patterns as an Instructional Material

Constructing database queries in SQL is a skill required of many academics, developers, and industry because it underlines all major applications. However, mastering this skill is a difficult process requiring considerable practice and effort on the part of the student. Several aspects of SQL cause difficulties. Many studies attribute this to the nature of SQL as a declarative language arguing that it is fundamentally different from the other programming languages that students have to learn [3, 10-12].

Patterns are a widely accepted mechanism for supporting knowledge transfer and therefore set out to investigate whether patterns could meet the need for

optimally-structured instructional material. Schlager and Ogden [13] found that incorporating a cognitive model in the form of expert user and product-independent knowledge into novice instruction enhances learning, and this is essentially the rationale for patterns of any kind. They showed that such a cognitive model framework could help to support learners, so the previously developed model of SQL learning in chapter 5 (see Figure 6.1) was ideal as a launching pad for the investigation into potential SQL pattern.

SQL patterns aim to enhance the learning experience for learners attempting to master SQL as well as to aid teachers in introducing SQL concepts to their students. In other words, providing SQL learners with SQL design patterns will help students to become familiar with common SQL problems and related solutions. In addition, during problem solving, SQL patterns provide the lacking knowledge in a convenient format. This does not rely on the learner's own assessment of their knowledge, which might well be completely wrong; the required knowledge is simply provided in a handy format for the learner to use.

SQL patterns approach, as a teaching method, is motivated by much research in the use of pattern concepts in Computer Science education such as the informative approach. The SQL pattern construction and implementation is motivated by the review of various studies.

- Bruner's Theory of Constructivism [234]: Instruction must be concerned with the experiences and contexts that make the student willing and able to learn.
- Merrill's theory [63]: which discuss knowledge structure; "greatest impact on learning results from the representation and organization of the knowledge to be learned. Knowledge structure refers to the interrelationships among knowledge components".
- Mayer research (Mayer, 2008): that was related to the way in which a body of knowledge can be structured so that it can be most readily grasped by learner. The most effective sequence in which to present material.

SQL patterns in this research are derived from the literature review in Chapter 2 and the analysis of the empirical research conducted in Chapter 5. Both considered different aspects of learning experiences and attempts to master SQL, and the gaps in the field of the instructional design. SQL patterns build on the existing knowledge of pattern design (Chapter 3) and Computer Science education research in general. The next section describes the process of SQL patterns identification.

6.3 SQL Patterns Identification Process

To report the process in development of SQL patterns, it makes sense to study other patterns identification methods and procedures. Patterns are not an optimistic collection of ideas or something ephemeral; they describe specific tried and tested techniques that are well recognized best practice in a particular field [125]. In the *Timeless way of building*, Alexander [125] described the existing patterns with respect to buildings:

> "We have been taught that there is no objective difference between good buildings and bad, good towns and bad. The fact is that the difference between a good building and a bad building, between a good town and a bad town, is an objective matter. It is the difference between health and sickness, wholeness and dividedness, self-maintenance and self-destruction. In a world which is healthy, whole, alive and self-maintaining, people themselves can be alive and self-creating. In a world which is unwhole and self-destroying, people cannot be alive: they will inevitably themselves be self-destroying, and miserable. But it is easy to understand why people believe so firmly that there is no single, solid basis for the difference between good buildings and bad. It happens because the single central quality which makes the difference cannot be named". (p. 25)

Patterns are not intended to state obvious solutions to trivial problems or to cover each possible solution's eventuality, but to capture important "big ideas" [161]. A pattern should explain how a problem should be solved and why the presented solution is appropriate and optimal in a particular context. Different approaches have been employed to identify patterns in different collections. Alexander [125] points out that patterns may be discovered in different ways, by identifying a problem and later finding a solution or by seeing a positive set of examples and therefore recognizing a solution. He describes how this is a process of discovery:

> "A pattern is a discovery in the sense that it is a discovery of a relationship between the context, forces, and relationships in space". (p. 259)

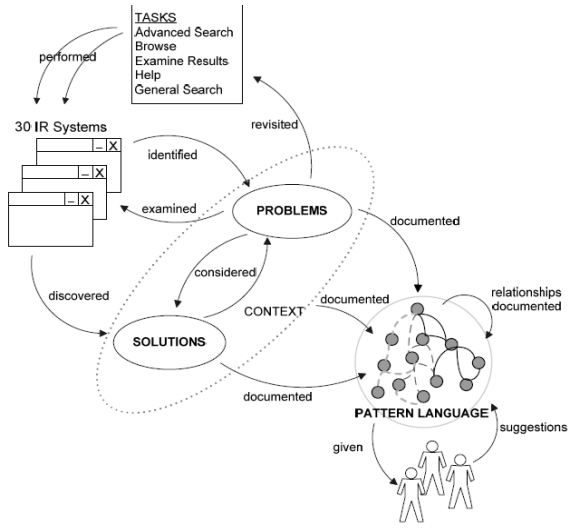


Figure 6.3: IRPLane Identification Process by Wania [310]

This approach, for example, is employed in the process of pattern language for information retrieval systems (aIRPLane) implemented by Wania [310], as shown in Figure 6.3.

This section elaborates on the processes involved in identifying SQL patterns. SQL patterns identification needs to focus on both the behavioural and the cognitive aspects of SQL acquisition. Understanding learner ability to perform

different cognitive tasks such as query formulation, translation and writing is essential to be able to design a new instructional design viz. the SQL patterns.

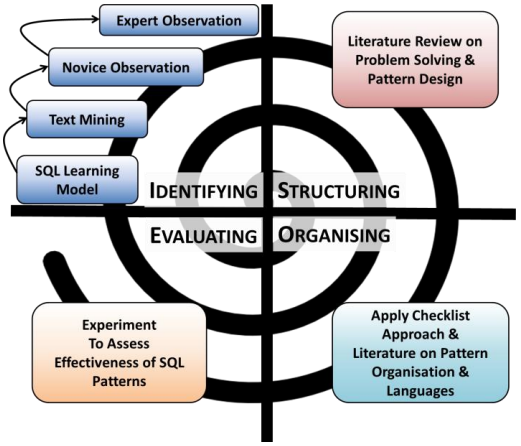


Figure 6.4: SQL Iterative Pattern Design Phases

The SQL patterns presented here emerged from an iterative research process (see Figure 6.4), which involved a review of educational research, uncovering relevant human factors related to SQL usability and psychology-related research. The process started with the aim of understanding the nature of SQL learnability: this was done by conducting a general overview of the literature about educational theory and the cognitive psychology research [37-39] and instructional design related research. The next step narrowed to cover CS educational research [46] and focused on problem solving skills. The SQL learning model emerged from the analysis of the educational literature, and was augmented by the analysis of data gathered during qualitative and quantitative studies of SQL acquisition in chapter 5. Having identified the possibility of deployed patterns to support SQL acquisition, the next subsection explains the process of identifying and defining the patterns using text mining, observation of novices, and observation of experts.

6.3.1 Problem Solving Strategy Identification via Mining

Patterns' mining was used to discover patterns from existing knowledge repositories, solutions, or designs. The mining metaphor has been used in workshops on patterns in Architectural Design and in HCI [311]). Patterns' mining

requires capturing practice that is both good and significant [312]. Patterns' mining was used during the SQL pattern identification process, as illustrated in Figure 6.5.

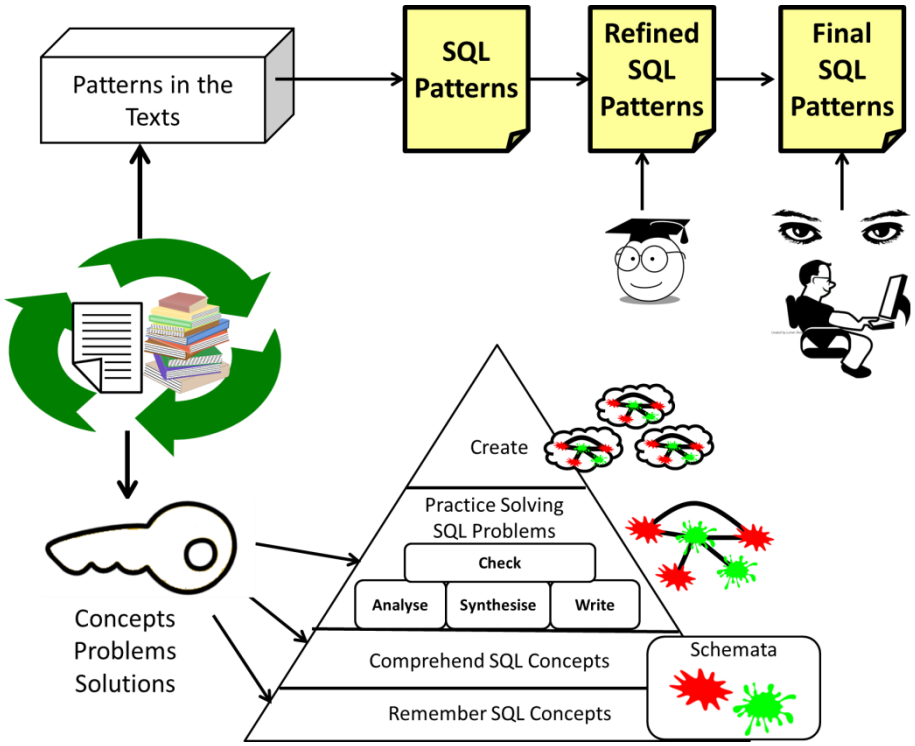


Figure 6.5: Mining Process

According to new instructional methods, there is a need first to apply what educators know about how students learn, remember, and use related skills. A text mining procedure was therefore deployed to extract this information from texts to identify common knowledge that relates to the core concepts and practices related to SQL query writing. The following steps were followed:

1- Identify SQL knowledge from database texts and categorize the knowledge into the following categories based on the SQL learning model (Figure 6.1):

   I. Identify the declarative or "Remembering" knowledge in terms of SQL concepts. Here, we mined data such as SQL facts or concepts. For example: joining, aggregation, and subquery.

II. Identify the procedural or “Comprehension” knowledge in terms of how SQL concepts are used in a certain context.
III. Identify the “Practice” skills by showing how the concepts should be applied in solving problems. For example, show a context scenario and explain how the relevant syntax and rules are applied. Also illustrate the scenario with appropriate examples which show, step-by-step, how such a concept should be applied.
IV. Identify the “Creating” activity. For example, finding evidence of generic principles being applied in particular contexts.

2- Identify the SQL misconceptions which could be corrected by the provision of patterns.

The above four categories were employed during the mining process using the knowledge management (KM) research that distinguishes between data, information and knowledge. The first step was deciding on the fact about individual SQL concepts that could be called data [313]. The second step was finding where these facts are applied. The third steps was looking on how such a fact is applied within the defined context by showing all related information on the steps of performing such a task. The latter is called the knowledge [313] of SQL concepts. Here we called it the “knowledge-of-context”. The last step is to find the abstract knowledge.

After the process of knowledge extraction and categorization, it was formulated as a set of patterns. The patterns’ mining process provided a starting point, delivering a static understanding of how SQL patterns’ knowledge was presented in textbooks and commonly used texts. How such SQL concepts are applied by query writers cannot be understood without empirical evidence, however. Hence, it is important to confirm the strategies deployed by observing and analysing novice SQL problem solving behaviour. The next section reports on the strategies, methods and approaches novice employ during SQL problem solving.

6.3.2 Problem Solving Strategy Identification through Observation

SQL patterns identification process was enhanced by focusing on both the behavioural and the cognitive aspects of applying SQL knowledge. As was discussed in chapter 5, understanding learners' ability to perform different SQL cognitive tasks such as query formulation, translation and writing is essential to be able to come up with a new instructional design that overcomes the identified issues. In addition, a model of SQL learning (Figure 6.1) highlighted the importance of cognition and the development of a mental model. To consider that, an observation task is required.

Researchers in the field of pattern identification agree that patterns ought to be identified with reference to design solutions through observation, rather than being constructed from theory. According to Alexander [125]: "In order to discover patterns which are alive we must always start with observation" (p.254). Furthermore, Fincher [314] points out that: "Patterns are not created or invented; they are identified via an invariant principle".

Strategy identification, by means of learner observation, helps determine what "best practice" SQL patterns should offer. To achieve that, cognitive science suggested giving learners a problem and observing everything they do and say while attempting the solution. Unstructured observations were conducted on a period of two semesters, as shown in Table 6.1 which summarizes the conducted observation.

Time	Participants	Participants
2009/10	Students registered in Information Management2 (IM2) course	17
2010/11	Students registered in (IM2) course	21
2010/11	Students registered in Database3 (DB3) course	15

Table 6.1: Time Spent with Novice SQL Learners

In this research, it is anticipated that particular problem solving strategies would emerge from the observation of learners during SQL writing. Each identified strategy should capture insights about the SQL problem solving process. Based on the observation process, the SQL patterns which were identified during the mining process were refined based on academics' experiences in teaching and using SQL and then modified based on the wisdom gained from the novice observation process.

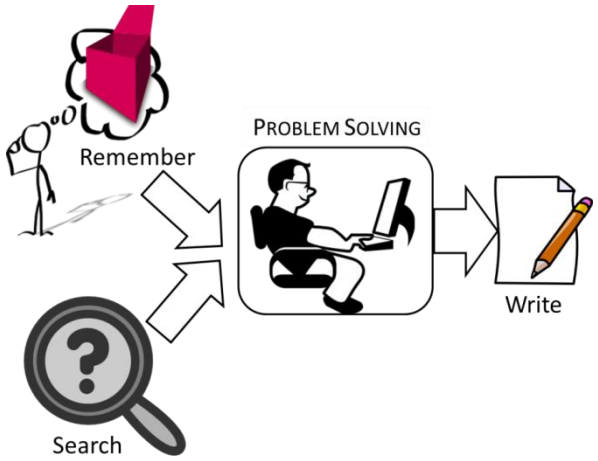


Figure 6.6: Patterns Identification Through Novice Observation

This section describes the process of SQL strategy observation and subsequent pattern refinement (see Figure 6.6). It was important to understand how novices solved SQL problems; i.e. the steps followed to arrive at a solution to the given problem. This includes:

- Remembering:
  - When they remembered the required knowledge, was it correct?
- Searching (Not Remembering):
  - How was the unremembered but required knowledge obtained? For example, did they refer to textbooks or teaching materials? or did they search the net to find similar problems and related solutions?
- Problem Solving:

- Was the required knowledge identified correctly?
- Was the knowledge correctly matched to the given problem context?
- Did they search for visual examples on the Web?
- Did they try different solutions? If so, why was a particular solution selected?
- How did they react to their errors?

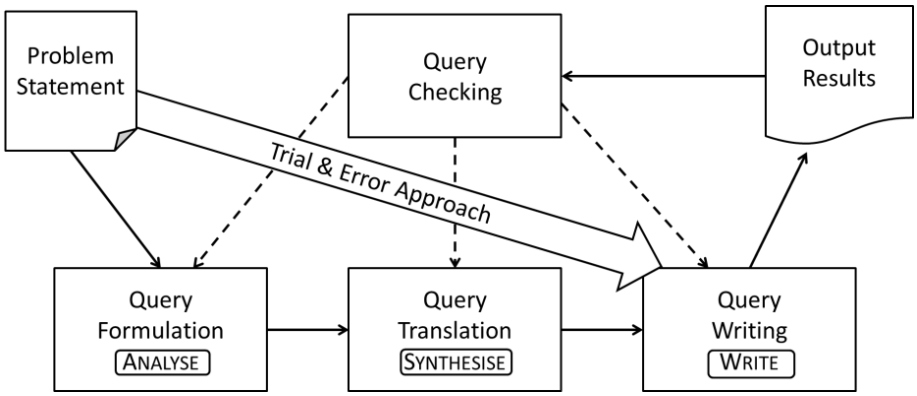


Figure 6.7: Novice Strategies in Problem Solving

The observation data, based on the observation of 53 students, revealed that many students lacked problem solving skills. Students often started to write SQL queries without taking the time to consider a number of different approaches. There was no attempt to choose an optimal approach from a number of candidate approaches. They behave tactically and do not take time to analyse the problem description and to consider what they should do before attempting to write the query. This tendency confirms previous research findings [315]. Students spent the bulk of their time solving syntax and semantic errors and assessing the correctness of the generated results (see Figure 6.7).

In addition, novices lacked the ability to sub-divide the problems into sub-problems and to identify the specific knowledge required to solve individual sub-problems [316]. The next step requires them to combine the identified sub-solutions to design a complete solution to the problem – synthesis. This, too, seems to be a skill that novices lack.

Less searching behaviour than expected was observed and when it did take place it was not always productive. Students searched for similar problems on the Web or spent some time looking at the lecture notes, trying to understand different concepts. This was often unproductive since they wasted time searching for concepts that were not relevant to the particular problem to be solved.

Observation of novices was invaluable in starting to understand how to design supporting instructional material. However, it also requires understanding what strategies were deployed by SQL experts since this is the behaviour we want to guide the novices towards. What emerged from this analysis was the fact that a particular intervention was required in supporting students during the problem solving phase, where they apply the basic SQL concepts and principles.

6.3.3 SQL Patterns Identification through Expert Observation

Patterns are a means of codifying experts' knowledge and expertise to facilitate knowledge transfer. Therefore, the content of patterns must be informed by experts' actual practices. This section presents a description of problem solving strategies deployed by two individual expert SQL query writers. The expert observation process is shown in Figure 6.8. The aim of this observation is to compare the problem solving activities of experts to contrast them to the patterns of behaviour observed by the previous analysis of novice problem solving.

The observed experts had a long working experience of SQL. The cognitive activities they performed during problem solving of two tasks were recorded by employing a "loud-talk" protocol. According to Dunbar [317], "loud-talk" protocols are good sources of information about tacit problem-solving processes. This facilitated comparison with novice behaviour.

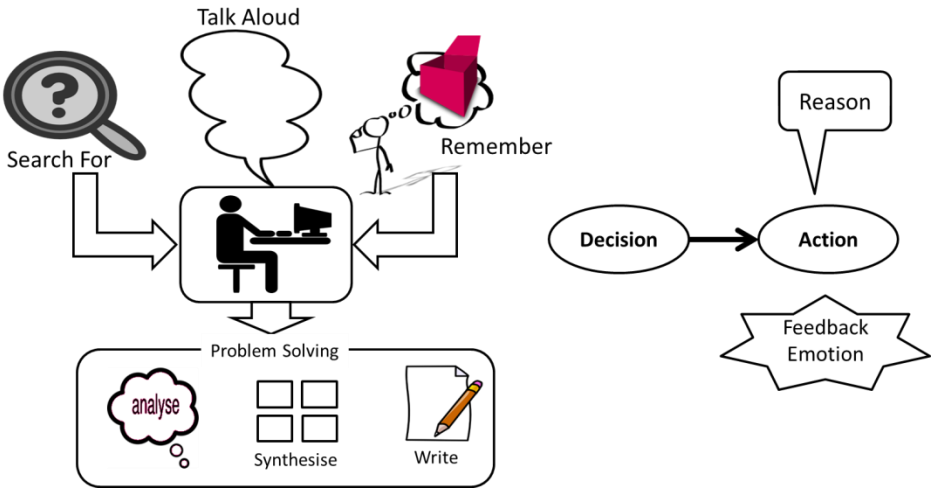


Figure 6.8: Expert Observation Process

The observation process includes all the cognitive activities, such as what they try to retrieve from their schemata (Remembering) and other information that they try to get from the web or other source of information (search-for).

The collected data was analysed based on [13, 59, 62, 234] models in how such knowledge can be classified and documented. All information was gathered and classified using a method adapted from [13]. They categorized the collected information into three categories: conceptual, procedural and rule or heuristic knowledge. The following subsections describe the methods and procedures used to collect data. Finally, the results of the analysed data are reported.

6.3.3.1 Methods and procedures

The goal of this study was to identify the ideas in time at which innovative SQL experts thinking occurs, capture this thinking on video tape, and then analyse the processes involved in the thinking and reasoning.

Q1: Give the titles of books that have more than one author.
Q2: Display the names of borrowers who have never returned a book late.

Figure 6.9: Expert Observation Task

The process begins by developing a task (see Figure 6.9) where participants need to follow some steps in order to solve the intended task. Participants were two Masters students at University of Glasgow who had good experience in using SQL. The aim of the task is to solve an SQL query. They were given two tasks to perform.

These participants used the SQLPB tool that was presented in chapter 4. Furthermore, Camtasia studio 4 was used to record all participants' actions and to record their explanations. There was no time limit for solving the given task.

The task design was discussed as was presented in section 4.6.3. All the related database tables were made available in the tool. Participants were asked to write the SQL query that would help them solve the given problem. The next section reports in more details how the data was collected and analysed.

6.3.3.2 Results analysis

According to Dunbar [317], think-aloud protocols are good sources of information about tacit problem-solving processes. In this study, think-aloud method was applied to investigate the expert thinking and reasons in solving SQL query.

The observation process recorded all cognitive activities (see Figure 6.10), such as schemata retrieval (Remembering) and about information searched for (Not Remembered).

In addition, the participants' feedback was coded as well. There are three codes used during the data collection (see Figure 6.10). Four codes were used to encode the collected data:

1- D: decision taken
2- A: action performed; e.g. analysis, synthesis, writing or evaluation.
3- R: reasons given (why A is done or D is made).
4- F: feedback or emotion experienced as a result of an action or decision

Analogy	Sources of analogy
A1→ Formulating the problem "number of loan …"	Schemata
D1→ Dividing the problem into sub problem or sub goals	Advanced knowledge
A2:→ Analysis : Joining two tables "joining the copy with titles"	Basic knowledge
R1→Reasons why *D1* " to get single table"	Basic knowledge
A3→ writing: Write SQL syntax (see Figure)	Basic knowledge
D2→ use subquery	Advanced knowledge
A4→Evaluation: Execute the *A3* without applying *D2* and checking the results	Basic knowledge
D5→to apply Self join for the table	Advanced knowledge
A12→ Writing: Modifying the query	Basic knowledge
A13→Evaluation: Modifying the query with no clear decision	Schemata
A14→Writing Applying D5	Schemata
A15: Analysis of the problem	Schemata
Applying aggregation	Advanced knowledge
A16: Writing: Iteration of changing the query	Advanced knowledge

Figure 6.10: Snapshot of the Cognitive Activities Performed by Experts

The collected information was categorized as conceptual (basic building blocks from Figure 6.1), schemata (knowledge of how concepts are used), or rule (abstract heuristic knowledge)

Experts, after reading the problem description, made an initial decision about the type of information that had to be applied. They then looked at the provided data model and verbally listed the possible approaches to solving the problem that they could deploy. After mulling it over, they settled on one particular approach and provided reasons for discarding the other options. Both experts used a divide-and-conquer approach and sub-divided the problem; they did not attempt to write the whole query at once. They wrote and tested the commands related to the sub-queries and then synthesized them to arrive at the final complete solution.

The most interesting part of this observation was the fact that the experts applied an implicit pattern matching approach to their assessment of the problem. They clearly tried to match a number of learned heuristics to the problem before settling on the best approach. One can only assume that they had internalized a number of abstract heuristics which they tried to match to the given problem before settling on a "best-fit" approach.

- Reading and understanding the problem - Search for more information from the Internet "Googling" - Problem Solving: • Analysis • Consulting ER model • Identify the available table holding the required data. • Rereading the problem. • Synthesis • Deciding which concepts to apply. • Searching for SQL syntax or relevant examples. • Writing • Start writing the first SQL query in the tool. - Checking: - Evaluate the result of the first attempt. - Manipulate the query with some justification (fixing the errors). This is done iteratively until they are satisfied. - Repeat sub-steps in number 3 until satisfied.	The participant whose process is depicted in Figure 8 broke the overall problem into a number of sub-problems. He first started by joining Book-Copy and Book-Title tables. At this stage a few actions (A1-A3) and a decision (D1) were performed and other decisions were pending. The participant was happy with his performance at this stage. He then applied another decision, ie. to use sub-queries The participant then exercised the decision to apply the self-join technique. However, he failed to apply it correctly. The participant then deployed aggregation and was satisfied that he had solved the problem.

Figure 6.11: Expert Cognitive Activities

Writing SQL involves cognitive activities such as learning, understanding and remembering [4]. To interpret Reisner's main classification, it is possible to say learning comes from searching for information and applying it. Understanding, on the other hand, is a reflection of what is "Said" and "Written". The above activities were classified using the following categories of what: "Said", "Wrote", "Remember" and "Search-for".

Research on reasoning has demonstrated that experimental participants make vast numbers of thinking and reasoning errors even in solving the most simple of

problems. During the task, many decisions (D) were made and reasons articulated. In Figure 6.10, "R" presents the reasons given by the participants. The above data (see Figure 6.11) guides the research into the type of knowledge that needs to be available to novices solving SQL problem. Actions and decisions were performed as a result of either participant's own interpretation of the available knowledge or by searching for specific knowledge. The next section discusses how the reported results contribute to SQL patterns development.

6.3.3.3 Discussion

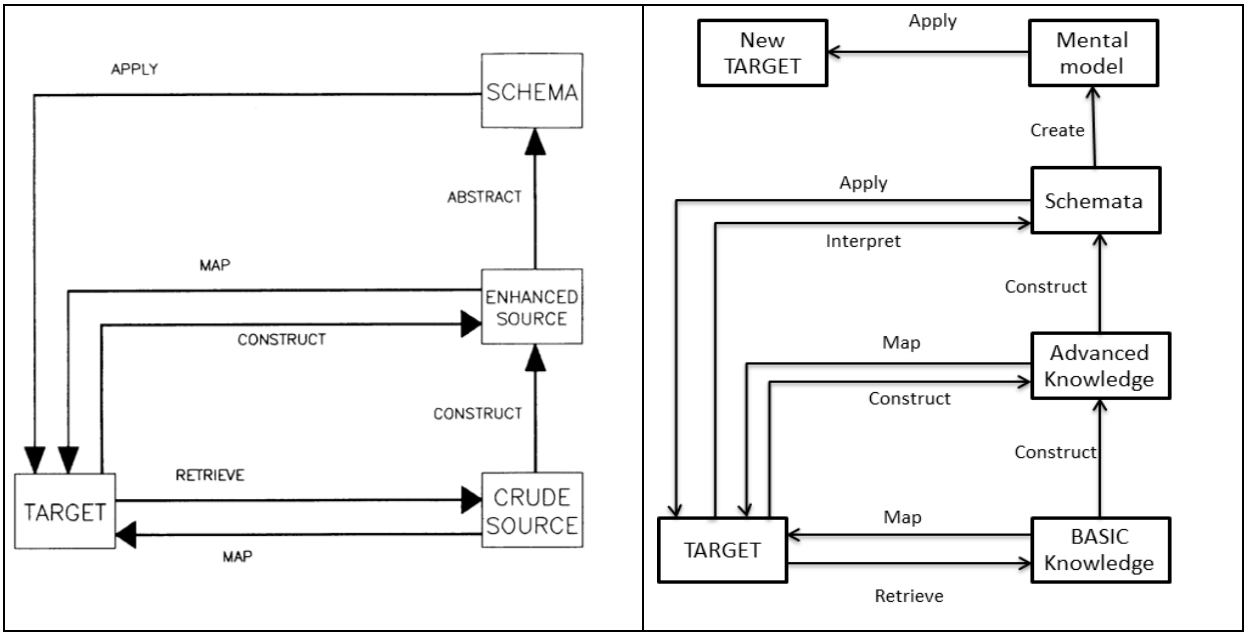


Figure 6.12: Expert Problem Solving [318] (left) and SQL Acquisition on Expert Model (right)

The reported findings inform the development of instructional materials that aim to expedite SQL learning. As was discussed earlier, the objective of any instructional materials is to support students through all stages (from problem statements formulation to query output evaluation) in learning and solving SQL query. This research goal was to investigate the analogy and reasoning strategies that leading SQL experts use while solving queries.

The left hand side of Figure 6.12 shows how experts solve a task using an analogical approach. The model is based on the ideas of Nersessian [318], which

depicts how scientists think and solve physical problems. The right half of figure 6.12 shows this model applied to SQL acquisition.

This model presents the different sources of knowledge and strategies experts deploy. Observation of experts' activities showed that they divided the problem into sub-problems. Then, for each sub problem, different relevant knowledge is applied to arrive at a sub-solution. When experts solved the first part, they applied basic knowledge. Then, as the problem requirements required more understanding, they applied advanced knowledge which was sometimes obtained by searching. They then applied problem solving strategies such as incremental development, division into sub-queries, consideration of a number of different ways of solving the problem, and choice of the optimal strategy.

Looking at how experts solve the task and comparing it with how similar tasks were solved by novices indicates the nature of the gap between expert and novice (see Figure 6.12). There was no evidence that novices struggled with knowledge of basic SQL syntax. They also understood how the SQL constructs ought to be used. However, novices clearly lacked the knowledge and skills required to solve novel problems. This is related to the "Practice" stage as seen in Figure 6.1. This, then, is where more effective instruction material needs to assist the process.

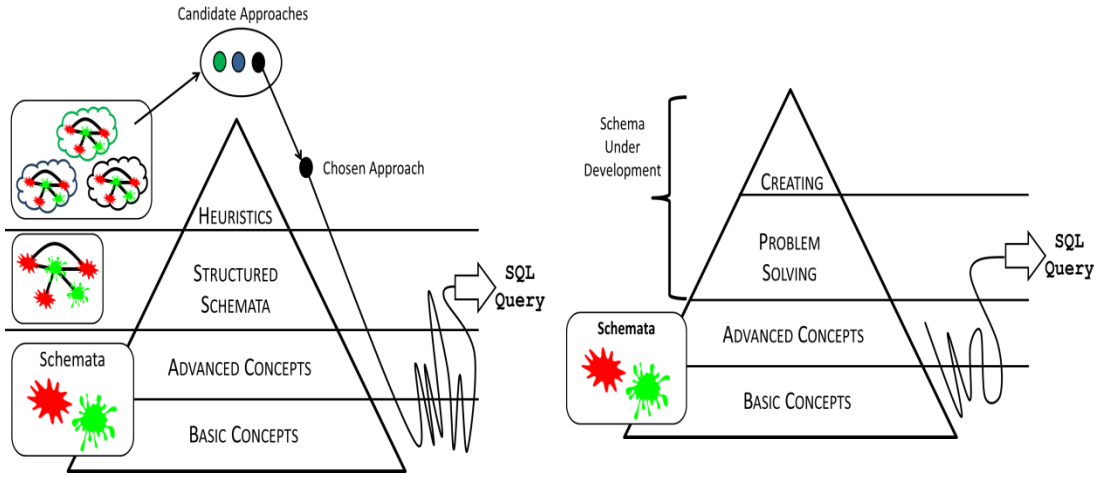


Figure 6.13: Typical Expert Actions (left) and Novice Actions (right)

This analysis and the comparison between novice and experts strategy (see Figure 6.13) allowed us to determine what type of knowledge and skills are required to solve the SQL problems. It was also possible to determine how the information should be presented to learners; i.e. the optimal sequence of information.

The results of experts observation study suggest that:

- Experts start solving the problem by formulating the problem and determining the context of it using the data model. This leads to the suggestion that learners need to be provided with data models to help them to understand the context of the problem.
- Expert knowledge is structured, connected and abstract. They have:
    - knowledge about “SQL Syntax and Semantics”
    - Knowledge about the meaning of SQL concepts “SQL Query comprehension”
    - Knowledge about how to apply SQL concepts in the given context

- Knowledge about the wisdom of SQL applicability in a certain context “problem-context-solution”. This is a high level of knowledge that novice lack as was discussed.
- Knowledge about the consequences of applying SQL concepts “impact-of-solution”. This is a skill of evaluating SQL concepts, which is a high level of knowledge that novices lack.

Observation of strategies made it clear that instructional materials, such as notes, did not guide students towards productive activities or to support effective problem solving. To help novices to achieve this level of expertise, it was proposed that the SQL patterns should include some specific knowledge (see Figure 6.14).

Knowledge Within Pattern	Learner knowledge
Provide students with data models to help them understand the context of the problem	Schema Formation
The impact of applying the pattern in such a problem context	Schema Formation
Support for matching a problem to a solution in a simple format such as a checklist	Schema Formation
A section which includes the basic knowledge required to solve the problem	Schemata
Step-by-step SQL visual examples of the pattern being applied	Schemata
Train students to deploy effective problem solving strategies as suggested by [34]	Encourage Engagement with Analysis and Synthesis Phases

Figure 6.14: Knowledge Within Pattern

6.3.4 Summary

The successful implementation of any instructional material (SQL patterns) will depend on the writer's understanding of different factors that influence SQL learnability. Generally, the researcher, to write an effective pattern, must rely

on cross-cutting factors such as: learner's characteristics, SQL language specifications, human cognition and the analysis of different methods and approaches that have been used in teaching SQL which were discussed in Chapter 5. Moreover, researchers should have some knowledge and experience on the sort of cognitive task analysis that is carried during query problem solving, which was explored in Chapter 2, and in the research methods that were employed in this section. Thus, it is easy to form the basis of the pattern's content, and ultimately serve as the link between the task requirement and the generic pattern.

SQL patterns are identified using two main strategies: mining and observation. The mining strategy process includes a collection of SQL knowledge from many texts that are related to the identified list of SQL misconception from the previous research in chapter 2 and chapter 5. Each identified SQL concept (problem) is related to knowledge in how to apply (solution) through appropriate (context). For each scenario that consists of "solution-to-problem-in-context", there is a list of concerns that shows why such concepts are more appropriate to apply in a certain context; i.e. why a "Problem" is difficult to solve. In addition to each "Solution", there is a list of the "Impact" results of applying such a solution in the identified context.

The results from the collaborated research (see Figure 6.2), that were conducted to identify SQL patterns, were interpreted towards deciding the main content of SQL patterns:

- Should emphasize declarative (Basic) knowledge, which is 'what', and 'How' using a traditional teaching approach. In addition, it should be based on an informative approach [319]) that covers when and why.
- Should be designed based on human cognition throughout the involvement of comprehension of SQL, Query formulation, Query translation and query writing (Schlager and Ogden, 1986).

- Should aim to guide students to the right strategy in problem solving, as was discussed in section 5.4.3.
- Understanding of the cognitive aspects of solving SQL problem, which was highlighted in section 5.6, section 6.2.1 and section 6.2.2.

The following is the list of the identified patterns:

- Dynamic Filtering Pattern
- Filtering by Existence Pattern
- Self-join Pattern
- Natural Join Pattern
- Grouping Result Pattern

According to Fincher [179]:

> "We believe that the converse between the identification and capture of individual stand-alone patterns without a corresponding structure is an activity which misses an essential meaning of pattern language."

Formulating the patterns correctly is a vital aspect in supporting patterns recognition by the SQL learner. The next section explores the methods and approaches in structuring or formatting SQL patterns.

6.4 SQL Patterns Structuring Process

According to Bayle *et al*. [160], it is relatively easy to observe phenomena in the world but much more difficult to use these observations to develop and extract good patterns. Patterns, to be useful, must present an abstraction of good practice at a meaningful level of granularity. Patterns should also present knowledge at graduating levels of abstraction [161]. Formulations that are too abstract will be impractical in real design use; those that are too specific will be difficult to reuse in new scenarios [113].

When the patterns are aimed to help novices learn something, then more caution is required. In this research, patterns are proposed as an instructional method that aims to present and organize SQL content to facilitate learning. According to Merrill [63], knowledge structure can be used to represent almost any processes that are defined in terms of properties. A condition for a process is some value on a property. A consequence for a process is a change in the value of a property. When the value of a property of an entity changes, the portrayal, either its appearance or its behaviour, also changes in a corresponding way. From Merrill's suggestion, to structure SQL knowledge, then each pattern might present:

- Process within properties,
- Condition of the process (value of properties),
- Consequenses of the process (change in the values of properties)

Fincher and Utting [312], on the other hand, compared abstraction in patterns to good teaching practice. They stated that, "Patterns should facilitate understanding of the principles embodied in specific examples, to identify what is important in the examples". From the above research, SQL patterns as discussed in section 5.7, should:

- Emphasize teaching of the context.
- Illustrate the abstraction process.
- Explore systematic mapping to build a previous well established knowledge supported with text and diagrams.
- Facilitate transfer.

In addition, the structure of the pattern must correspond to the user's conceptual and procedural understanding of the stages that a problem will require in terms of the application of the key concepts.

This research considers the possibility of delineating the optimal pattern format and to study the relation between different structures and their usability in

education. This leads to come up with a standard format that will maximize efficacy of the SQL pattern (in terms of how easy it is to learn, understand and remember or recall during the design or exams-building schemata and minimizing cognitive load).

Indeed, one can argue that SQL patterns need to be structured carefully in order to be effective and usable tools for both expert designers and novice learners. SQL patterns structure is developed through different stages of this research (see Figure 6.2). The following subsections describe related research in SQL patterns formulation.

6.4.1 SQL Patterns Format (Phase 1)

The initial format of SQL patterns in this research was based on the education theory that focuses purely on problem solving. In addition, it did not consider the actual form of pattern design. Students learn SQL by solving problems using Problem Based Learning (PBL) [320]. PBL was used as a framework to structure the patterns in phase 1 of SQL patterns design.

Anderson argues that discovery-based learning leads to greater retention of knowledge, which is obviously what PBL is striving towards [321]. However, in terms of learning theory, this discovery-based approach could lead to frustration and the student giving up. The aim of the pattern is to bridge the gap and to guide and ease the discovery process. This will prevent the student from wasting a great deal of time searching for answers in the wrong places. Whereas exploration of the available information is good if this activity is productive [3], unguided exploration could just as easily lead to students becoming discouraged and not learning anything meaningful.

The example shown in Table 6.2 demonstrates how students should solve a simple SQL problem.

Example 1: Write a query to display Employee Name, Department name and Salary for each employee that is earning salary between 500 and 1500.

Fact Identification:

1. Details of all employees must be displayed
2. Details are not all in the same table

Idea Generation:

1. Gather related information from multiple tables
2. Tables need to be joined by matching values in related columns. Need to select the required matching columns from the two tables; i.e. those which should match to ensure that the data in one table is related to the data in the other
3. Not all details need to be returned by the query

Knowledge deficiencies (Syntax):

1. Understand the correct terminology for this action; i.e. join
2. Determine the correct SQL syntax to gather information from multiple tables; i.e. name both tables in FROM, and use WHERE to specify which column values should be matched
3. Finding out how to filter details from joined tables i.e. specify column names in SELECT

New knowledge:

```
Select  e.Emp_name, d.Dept_name, e.Salary
From Employee e, Department d
Where e.Dept_Id = d.Dept_Id
And salary between 500 and 1500;
```

Table 6.2: Example Illustrating Problem Based Learning Steps

There are a few points that need to be noted regarding the above scenario:

- This scenario is a common query that is carried out daily in organizations.
- The facts identified will apply to all similar problems.
- The generated ideas should map into the facts.
- The knowledge deficiencies should provide the bridge between ideas and implementation. Without this, the learner might well get stuck after idea generation.

- Most similar queries use similar code (with a change in name of the tables and columns), which makes the pattern applicable in various contexts.

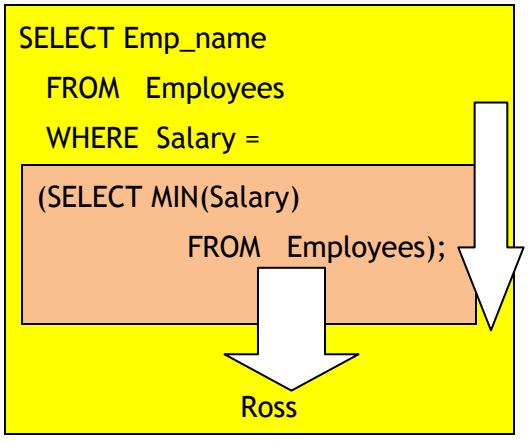
If this is the case, then, our approach is to collect all similar problems and their related solutions so that it can be used by others (perhaps a novice learner). This is ultimately what was referred to as a pre 'SQL pattern' at this phase. In other words, each generic problem type with its related facts, generated ideas and the required knowledge will be collected together to become a 'pattern'. The structure of the first set of patterns looked like the illustration in table 6.3.

Pattern section	Definition
Reference	This part will have a number, so each pattern will contain a unique number. This is used to link or refer different patterns
Name	Each pattern will have a name that is easy to remember and track
Keyword	A few words that summarize the content of the pattern. These keywords will be used for later search about any patterns when the collection is in electronic status.
Problem	SQL common problems will be presented here
Fact identification	A checklist of the problem related facts will be presented here
Idea generation	The solution will be based on checklist approach where a number of scenarios will be listed and the learner will select the most appropriate.
Knowledge required	Many code structures will be presented. Each will match one or more scenario that was selected on previous section (solution)
Examples	This will provide SQL code and table snapshot that refer to above code structure showing a step-by-step display of how the result of any query can be calculated. The reason of this is to use visual presentation and to animate the execution of the code so that students can develop better mental models of what is described.

Table 6.3: Patterns Structure in Phase 1

The first set of patterns was discussed by Al-Shuaily and Renaud [20]. An example of a pattern at phase 1 is illustrated in Table 6.4.

Reference	0002
Name	Using Subqueries
Keyword	Subquery
Problem	Gather information
Fact identification	[]Report information from one table – referred to as MAIN table []Filter the information based on data in another table – referred to as SECONDARY table []The returned MAIN table columns need to be filtered (OPTIONAL)
Idea generation	Describe the result needed from the secondary table Decide how to use that result to filter the main table's data Need to filter only the columns that are required by the query
Knowledge required	The query on the secondary table is called a subquery or inner query. It is usually enclosed in brackets in the outer query OUTER QUERY (INNER QUERY) The *inner query* returns a SET of values, and these values are used in the WHERE section of the *outer query* to filter rows in the main table. Eg. SELECT * FROM main WHERE somevalue IN (select values from secondary) the outer query checks for values IN the set returned by the inner query. The outer query can also check for the existence (or non-existence) of returned values. Eg. SELECT * FROM main WHERE EXISTS (select values from secondary where someconstraint) Or SELECT * FROM main WHERE NOT EXISTS (select values from secondary where someconstraint) If the inner query returns only one value, we could use: SELECT *

<table>
<tr><td></td><td>FROM main
WHERE values =
(select values from secondary)</td></tr>
<tr><td>Examples</td><td>
<table>
<tr><th>Emp_ Id</th><th>Emp_na me</th><th>Dept_ Id</th><th>Salar y</th></tr>
<tr><td>113</td><td>Ali</td><td>50</td><td>1500</td></tr>
<tr><td>205</td><td>Fay</td><td>50</td><td>1300</td></tr>
<tr><td>206</td><td>Ross</td><td>70</td><td>700</td></tr>
<tr><td>101</td><td>Ahmed</td><td>20</td><td>2000</td></tr>
<tr><td>100</td><td>King</td><td>20</td><td>5000</td></tr>
</table>
Get the name of the person who earns the lowest salary.
Facts: Main table is employee, secondary table is employee. We need only the name of the person who earns the lowest salary.
Ideas:
Inner query needs to return one value: the lowest salary.
Outer query needs to use this result to filter rows of employee table to return only the employee whose salary matches the lowest salary returned by the inner query.
 

</td></tr>
</table>

Table 6.4: An Example of a Pattern at Phase-1 Development

The early stage of pattern refinement, the patterns are refined after discussion with others with an experience in teaching SQL. Moreover, findings about the approach of learners were gathered and literature about patterns design conducted.

There was a debate about the general rules in writing a pattern, which took different format in the literature and its usability in teaching and learning. To the best of the researcher's knowledge, there were no such consideration in

developing patterns structure and its usability for novice users in other pattern areas such as HCI or SE patterns. Dearden and Finlay [113] call for further research on patterns structure.

From the observation reported in section 6.4.2, it was noticed that when students solve SQL problems, they experience many difficulties in matching the knowledge learnt with that knowledge required to the given SQL problems. That might also be related to students not knowing how to apply such knowledge, when to apply or why to apply it, as was discussed earlier. In addition, that might be a result of not having experience in solving SQL problems, as discussed in chapter 5.

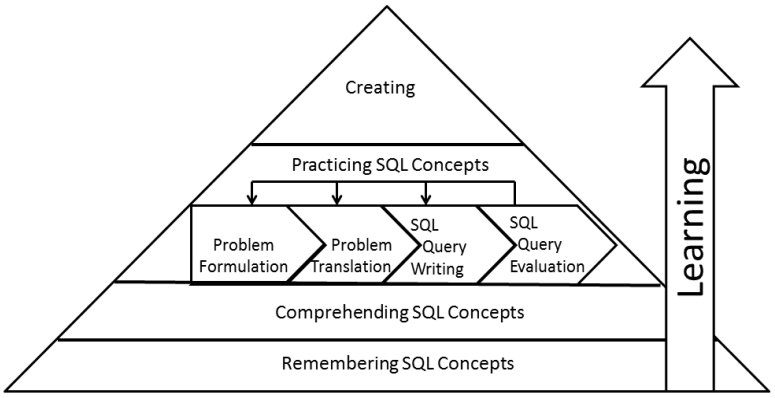


Figure 6.15: SQL Learning Taxonomy

Here, studying the learning taxonomy (see Figure 6.15) one could argue those students' attempts to perform in the upper level of the learning taxonomy which is problem solving "Practice" before having mastered the knowledge encapsulated in the lower levels. Such a shaky foundation, inevitably leads to poor results in learning SQL.

The phase 1 evaluation of the proposed SQL patterns agreed with the above finding. As a result, changes in the structure of the patterns were made. The next section explores patterns structure changes.

6.4.2 SQL Patterns Format (Phase 2)

In phase 2 of patterns design, different theories were applied to SQL patterns designed such as cognitive psychology in solving SQL problem (discussed in section 2.3.3), the discussion about SQL skill acquisition (elaborated in section 5.8) and patterns design (presented in section 3.4).

Furthermore, at this phase, SQL patterns designed were influenced with patterns' writers in terms of how to write patterns. Different documents were used to guide the patterns development [322] as shown in Figure 6.16.

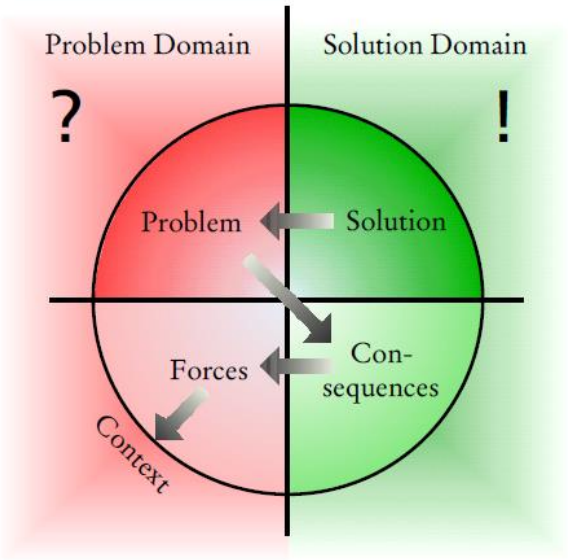


Figure 6.16: : Essential Pattern Section and Their Writing Order [322]

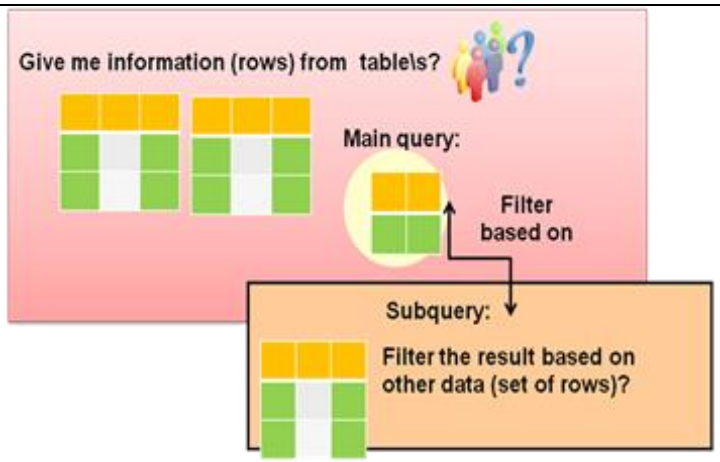
SQL patterns are structured differently through the *shepherding process* at Euro PLOP. The shepherding process is a peer communicative process, as is the writers' workshop. Shepherds are experienced patterns writers, who are assigned to assist the author (sheep) to improve his/her pattern. The shepherd helps the sheep into a more mature understanding of his or her pattern, usually using "The Language of Shepherding", according to which the shepherd must build a good relationship with the author, and maintain it throughout the shepherding process in order to prevent barriers and thus to reinforce effective communication. Shepherding process involves four stages development process:

1. Submission of an initial version to a Pattern Languages of Programs conference (EURO PloP)
2. Going through a shepherding process through which iteratively improves patterns.
3. The new version is evaluated for acceptance, and then the pattern is peer-reviewed at a Writers' Workshop.
4. The author incorporates the feedback received at the writers' workshop into the paper and produces the final version of the pattern before it goes into the final proceedings.

After SQL patterns were evaluated in the shepherding process, the final set of patterns included the following element: pattern name, context, problem statement, force, solution, consequences, and examples. An example of a pattern is illustrated in Table 6.5.

Then, the set of patterns were evaluated and enhanced through an addition of some feature and elements. The results of expert's observation, as reported in section 6.3.3, influenced the structure of SQL patterns slightly.

Name: Dynamic Filtering Criteria Pattern



Context
A user wants to construct SQL search query for a relational database management system where the information which user want to display are with changeable or unknown filtering criteria

Problem
How can you display a specific data (rows) from tables when conditions in WHERE clause are unknown or changeable?

Forces
Specifying the filtering criteria makes the search more rigid and required a lot of time when there is a change in the database.

Solution
In the WHERE clause use subquery to give you a list of data that are used to filter your data
Format

SELECT *column1, column2*
FROM *tables*
[WHERE] subquery

Consequences
Filtering criteria can be update automatically when the query is running, which means when the value of the table change filtering criteria will change dynamically. Therefore, the efficiency of the SQL code is better than having hard coding criteria.
Example
You are asked to Display the names of borrowers who have never returned a book late?
To solve the above problem you need to follow these steps: Specify the tables that you need to use to solve the above problem. The table Borrower and Loan.

- Specify the required columns that you need to get directly from both tables. Or\ and Specify any columns that you need to do some calculation to get it.
- Do you need to create any temporary data, specify how you will create.
- Do you need to create any temporary data, specify how you will create. For example, the information about the borrower who have never return a book late are not available directly from your tables. Therefore, you will need to create a subquery.

Table1: Borrower

BOR_ID	BOR_NAME	BOR_MAXBOOKS
1	Jack Jones	5
2	Betty Smith	5
5	Jenny Wren	8
7	Peter Piper	5
9	Jay Patel	5
11	Nancy Green	8
12	Billy Black	8
14	Keith Kettle	5
15	Polly Peck	5

In this example, there is no available data that shows the borrower with the criteria "never returned a book late" , to obtain such values a subquery is needed.

Result of the main Query is :

BOR_NAME
Betty Smith
Billy Black
Jenny Wren
Nancy Green
Peter Piper

```
SELECT      distinct bor_name
 FROM        Borrower b
WHERE       bor_id NOT IN
      ( SELECT      b.bor_id
        FROM      Loan l
       WHERE     b.bor_id = l.bor_id
       AND l.date_back > l.date_due)
```

Result of Subquery that filter data in the main query : =====➔ BRO_ID is 1, 9, 15, 14, 14 filtering criteria will change dynamically each time you run the query if the data change in your tables.

Table1: Borrower b

BOR_ID	BOR_NAME	BOR_MAXBOOKS
1	Jack Jones	5
2	Betty Smith	5
5	Jenny Wren	8
7	Peter Piper	5
9	Jay Patel	5
11	Nancy Green	8
12	Billy Black	8
14	Keith Kettle	5
15	Polly Peck	5

Table 2: loan l

BOR_ID	BC_ID	DATE_OUT	DATE_DUE	DATE_BACK
14	122	2003-02-16	2003-04-16	2003-04-02
14	121	2003-02-16	2003-04-16	2003-04-02
14	105	2003-02-16	2003-04-16	2003-04-02
14	107	2003-02-16	2003-04-16	2003-04-02
14	109	2003-02-16	2003-04-16	2003-04-02
14	120	2003-02-16	2003-04-16	2003-04-02
14	119	2003-02-16	2003-04-16	2003-04-02
1	106	2005-02-16	2004-02-01	2004-01-19
1	108	2006-02-16	2004-01-01	2004-01-19
9	101	2007-02-16	2004-03-02	2004-03-03
15	107	2011-05-16	2011-05-16	2011-05-17
14	108	2011-05-16	2011-05-18	2011-05-19
14	110	2011-05-16	2011-05-17	2011-05-20

Table 6.5: An Example of a Pattern at Phase 2 Development

The next section discusses the change of SQL patterns in phase 3.

6.4.3 SQL Patterns Format Phase 3

Patterns structure in phase 3 has changed to some extent. The finding of expert observation affects this phase. Merging the expert analogy (see Figure 6.17) and the model of SQL learning guides us to finding the missing information in the patterns' knowledge structure.

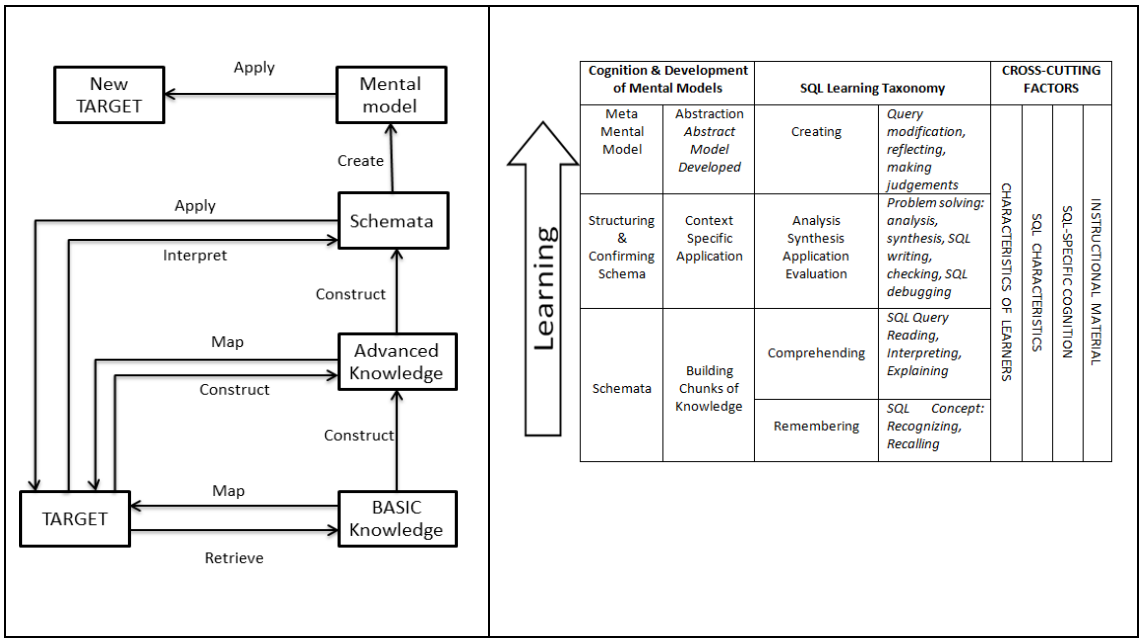


Figure 6.17: Expert Problem Solving [318] (left), A SQL Learning Model(right)

The question here is how to provide students with the advanced knowledge and to help them to construct schemata. The first decision was to change the Patterns' Examples section to be a worked-out example that shows step-by-step a scenario that was applied in each pattern. In addition, "Condition" is added to present the link between one pattern and other patterns. Condition expresses the condition where a pattern can be allocated to the pattern context. Patterns' diagram section was enhanced at this phase. Table 6.6 shows the patterns' format.

Another aspect of SQL pattern formulation is the language used to describe the content of the patterns. It might be classified as easy to understand making sure of selecting phrases that students are familiar with. This is in addition to trying

to avoid any jargon or ambiguous statements. Table 6.7 illustrates an example of the new version of SQL patterns.

Patterns' part	Description
Title	Patterns name
Diagram	Presents data model related to the patterns context
Condition	Presents a checklist of facts that patterns context might relate
Context	Presents the context of the patterns
Force	Presents Under which circumstances does the problem appear?
Problem	a short sentence that summarizes the problem and stated as a question
Solution	a solution statement that answers the question in the problem section
Worked example	Show step-by-step example how to apply the pattern
Consequences	Explain What happens if the solution is applied?

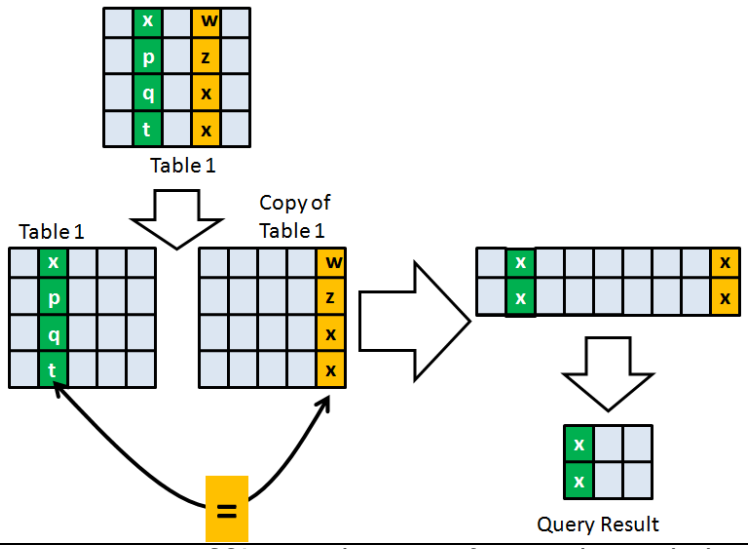
Table 6.6: Pattern Structure at Phase 3 Development

Self-Join Pattern:

SQL example query: Get the titles of the books that have more than one copy in the library (where all book details are stored in a single table)

IF
(all the data are in one table) **&**
(rows need to be filtered based on data in other rows in the same table)

THEN Look at "Self-join" pattern



Context	A user wants to construct SQL search query for a relational database management system where the information is spread over various rows within one table.
Problem	How can you compare values from different rows in the same column?
Forces	Searching for two different values to compare between them means that you will have two separate queries that need to be linked together. You can use nested SQL queries that make multiple references to the same table, but this might cause a performance problem.

Solution
Create two perspectives of the same table to be able to relate rows from that table with other rows from the same table, known as a self-join. This process efficiently connects a table with itself. This query joins a table to itself. It uses table name aliases so that each "instance" is easy to reference.
Format:
SELECT column1, column2
FROM *table* t1, *table* t2 ➔ t1, t2 are the table name aliases
[WHERE t1.column_name = t2.column_name]

Consequences
The disadvantage of this solution is that if you are joining huge table to itself requires a lot of memory resources in the DBMS.

Example:
list all information of employees and managers
Specify the information that you need to get which is the employee name and the name of their managers
Specify the tables that the information you need are in, in this case Employee table.
Specify the condition of your information.
Look closely to the condition more than one value in the same table
Then the best solution to do the comparison is to create a virtual copy of the table that has the two values which Employee table.

Give the original copy and the virtual copy two different alias and then join the two tables with itself or with a third table ➔this is called the self-join
The most common case where you'd use a self-join is when you have a table that references itself
To solve the above query you need write such SQL command:

EMPLOYEES (WORKER)

	EMPLOYEE_ID	LAST_NAME	MANAGER_ID
1	100	King	(null)
2	101	Kochhar	100
3	102	De Haan	100
4	103	Hunold	102
5	104	Ernst	103
6	107	Lorentz	103
7	124	Mourgos	100
8	141	Rajs	124
9	142	Davies	124
10	143	Matos	124

EMPLOYEE_ID	LAST_NAME
100	King
101	Kochhar
102	De Haan
103	Hunold
104	Ernst
107	Lorentz
124	Mourgos
141	Rajs
142	Davies
143	Matos

```
SELECT e1.last_name||' works for '||e2.last_name
   "Employees and Their Managers"
   FROM employees e1, employees e2
   WHERE e1.manager_id = e2.employee_id
```

Result

```
Employees and Their Managers
----------------------------
Rajs works for Mourgos
Raphaely works for King
Rogers works for Kaufling
Russell works for King
```

The join condition for this query uses the aliases e1 and e2 for the sample table employees:
e1.manager_id = e2.employee_id

Table 6.7: Patterns' Structure at Phase 3.

After an individual pattern was documented, different approach was evaluated to organize these patterns. The next section discusses SQL patterns' organization process.

6.5 SQL Patterns Organization Process

This section proposes an approach for the management of SQL patterns collections. The goals are to support students in two different tasks: (1) selecting the correct patterns and (2) understanding the relationship between patterns in the collection. To this end, it was suggested to use the Checklist approach for task (1) and a Graphical representation (inspired from component-level design) for task (1 and 2).

Different techniques were applied to collect and organize patterns to facilitate their use, as discussed in section 3.5. A pattern collection is any set of patterns that might include other subsets. Different approaches and methods were applied to organize the patterns. In this way, patterns' usability can be maximized. Section 3.5 presents different methods in organizing patterns in different collections.

SQL patterns collection or SQL pattern language should have the following characteristics like other patterns:

- Scalability: which exhibits the ability to develop and grow as more patterns are discovered and recorded within that field.
- Complexity: to enable the presentation of patterns in a simpler manner so that users with various levels of experience can grasp it.
- Flexibility: the patterns can be adapted to different design course's content, aim and objectives.
- Accessibility: patterns should be easy to find and use.
- Homogeneity: the format of different patterns within a collection will obey consistent structures so that users of that collection will find it more beneficial to jump from one pattern to another.

In addition, it should consider learning issues as patterns are intended to be used by learners. During SQL patterns' organization research considered, different techniques were used in organizing other patterns collections such as HCI, SE

and Architecture. The Pattern Language Markup Language (PLML) was introduced in 2003 [323].

Since SQL patterns are still in their early stage of investigation, the researcher did not consider the use of PCML an appropriate technique. However, the researcher decided to discuss the use of other helpful concepts such as checklist approach as a scaffolding technique [324].

6.5.1 Checklist Approach

A checklist was described by Scriven [242] as a list of factors, properties, aspects, components, criteria, tasks, and the presence of which are to be considered separately, in order to achieve a certain task.

There are different types of checklists, as defined by Scriven [242]. For example: sequential checklist, strongly sequential checklist, weakly sequential checklist, diagnostic checklist and the criteria of merit checklist. Diagnostic checklist is employed in this research because it aims to match the problem to the list of patterns.

Atul Gawande [325] followed this same approach and obtained some interesting results, which he recounts in his latest book, The Checklist Manifesto: How to Get Things Right. He argues strongly that checklists were an effective remedy to "ignorance, uncertainty and complexity" [325].

When students are given a complex task, they are up against three main difficulties: faulty memory, distraction and poor assessment of their competence. In addition, some query writers skip crucial steps even when they remember them. The checklist approach provides protection against such failure [325]. Some researcher recommend using checklists to support student learning and performance by suggesting that well-designed checklists identify steps students should take to complete complex tasks [326].

Here, the focus of the discussion is on checklists that support students in solving SQL queries. As was discussed earlier, students need to derive the related facts from the given SQL problem. Many students cannot list all the facts because they simply don't understand what is being asked. Providing students with similar SQL problems, written in a checklist format might well make it easier for them to match and select the related fact. Checklists were deployed in each pattern in the "Condition" section. It provides them with a tool, a way to advance over their current difficulties. The next section presents the graphical representation of SQL patterns.

6.5.2 Patterns' Graphical Representation

To help novices use patterns effectively, it was suggested to employ more than one theme to present the link between different patterns. Alexander developed a 250-pattern multi layered pattern language [125]. The pattern, within Alexander's pattern language, are hierarchically connected to one another, in the way that higher level patterns are made up of lower level patterns, and these relationships are made explicit within the patterns. Moreover, Alexander [125] explored the relationships between the patterns and the network in which the patterns exist by stating:

> "Each pattern sits at the centre of a network of connections which connect it to certain other patterns that help to complete it…and it is the network of these connections between patterns which creates the language…In this network, the links between the patterns are almost as much a part of the language as the patterns themselves" (pp.313-314).

Since each pattern could include more than one pattern, the collection of SQL patterns was inspired by Alexander [125] approach. However, Component-level design approach was employed to present the graphical representation of level of patterns. Modelling component-level design applied in software engineering to translate the design model into operational software [243].

Here, each level includes a list of patterns, conditions (checklist items) and connectors (see Figure 6.18). Level 0, however, does not include patterns. It presents a problem with a list of conditions and optional connectors. The conditions act as checklists which inform and guide learners in their choice of patterns.

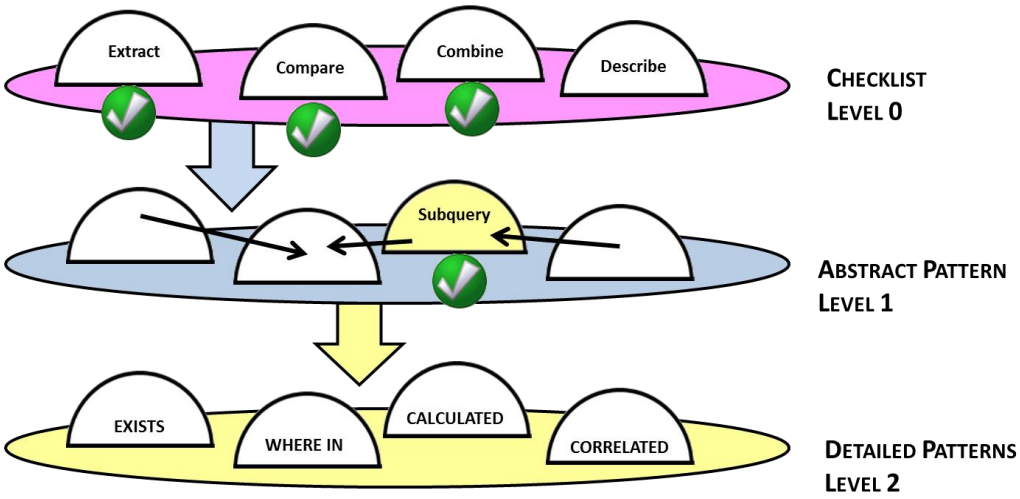


Figure 6.18: Graphical Representation

Figure 6.19 presents level 0. At this level, educators might start teaching the first stage in problem solving which is *query formulation*. The problem statement can be presented together with a list of conditions. Novices might be asked to draw a path between these condition list items. Moreover, as a starting point, educators could provide students with some initial paths to get the students started. The aim of this stage is to help novices to learn how the main goal can be divided into sub-goals. This is a strategy automatically applied by experts, and essential for correct SQL writing.

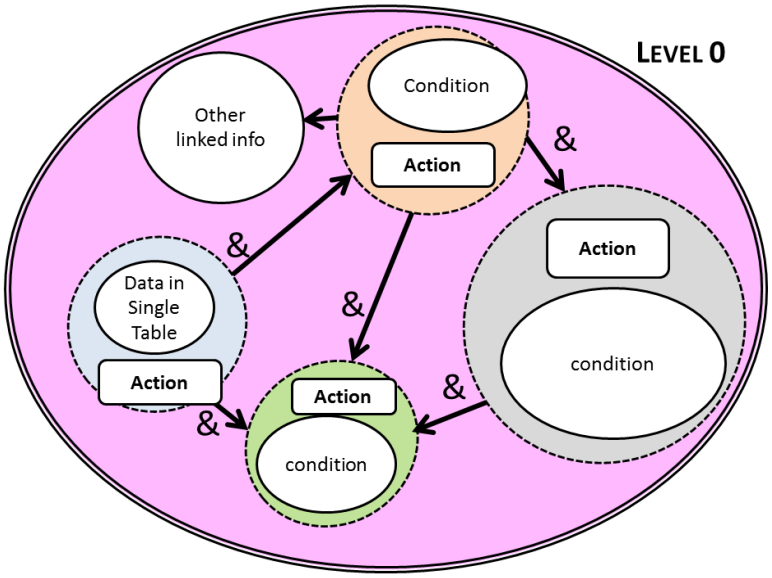


Figure 6.19: Level 0 SQL Patterns' Representation

Level 1 (see Figure 6.20) aims to facilitate students' skills in problem *analysis*. It guides novices to the main elements in the problem deliberation. Level 1 includes checklist conditions that are linked to one or more general patterns. The student is required to match the identified aspects of the problem with the stated conditions and then to be guided down the correct path to the appropriate pattern.

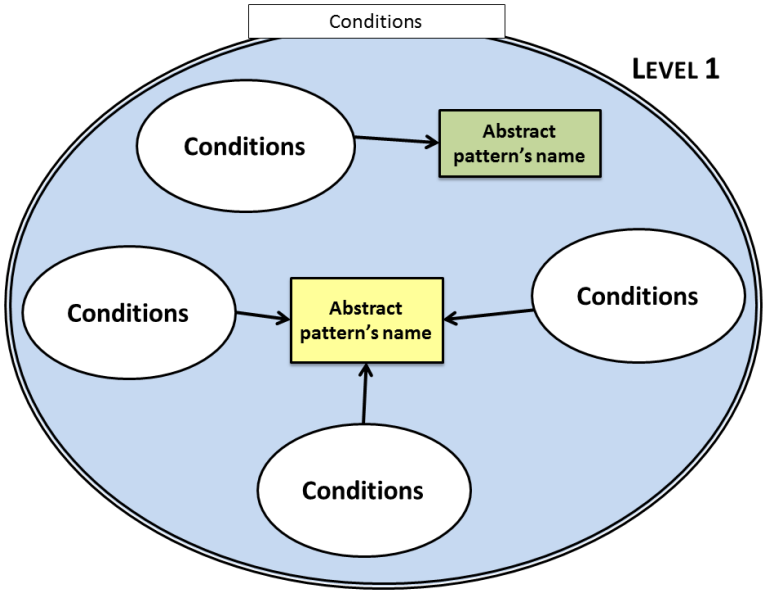


Figure 6.20: Level 1 SQL Patterns Representation

Level 2 (see Figure 6.21) presents further refinement into lower levels. For example "Aggregation" patterns could be refined into a further level, in this case level 2. It is vital to maintain information flow and continuity between the levels so that evidence of the hierarchical structure is obvious and visible.

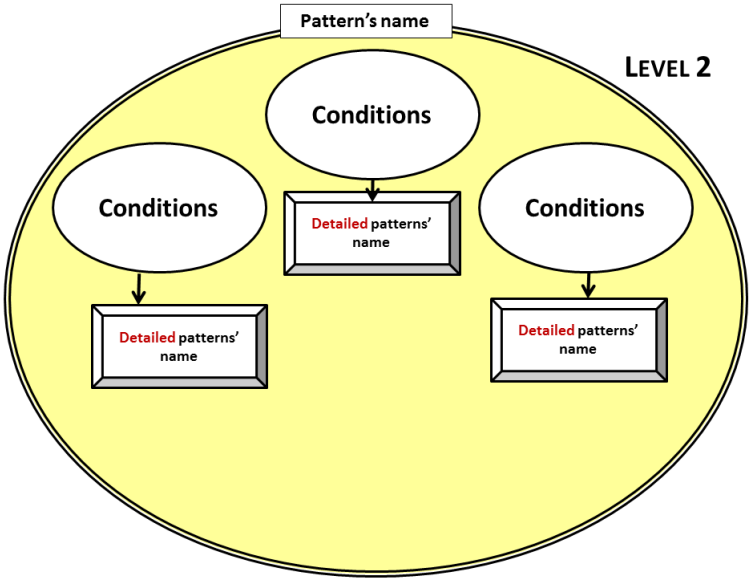


Figure 6.21: Level 2 of Patterns Presentation

The refinement of the component-level model continues until each pattern performs a simple function. That is, until the process presented by the model solves the problem in level 0 and the novice can write a complete SQL query.

Example 1

Display the names of borrowers who have never returned a book late. To solve this question, first you need to formulate the problem and understand its application domain. Level 0 (see Figure 6.22) aims to help students in understanding the problem. It provides them with a checklist as follow.

Step 1: Problem formulation and checklist generation

- ✓ Borrower return date and due date need to be compared: *Compare*
- ✓ Data is held in two tables: *Combine*
- ✓ Extract borrower name: *Extract*

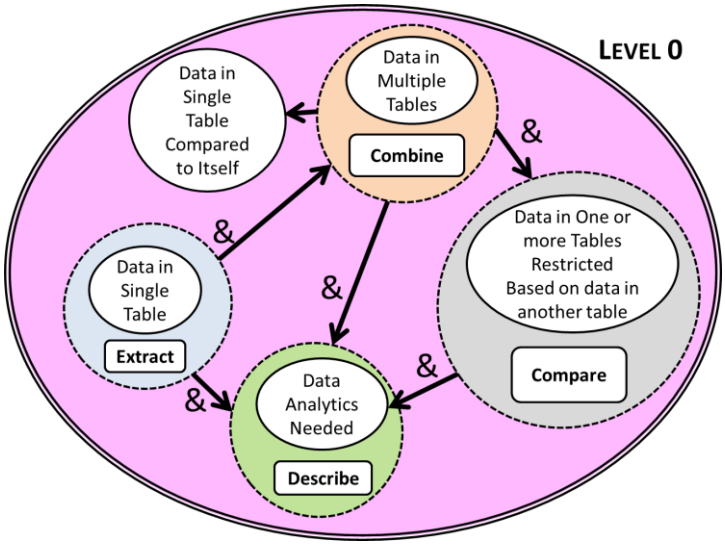


Figure 6.22: Level 0 SQL Pattern Representation

Step 2: Problem analysis and checklist connection

To solve the problem we need to match the defined checklist to the available patterns.

Level 1 (figure 6.23) aims to facilitate students' skills in problem *analysis*. Then, novices need to match the listed condition with the related patterns, as can be seen in table 6.7.

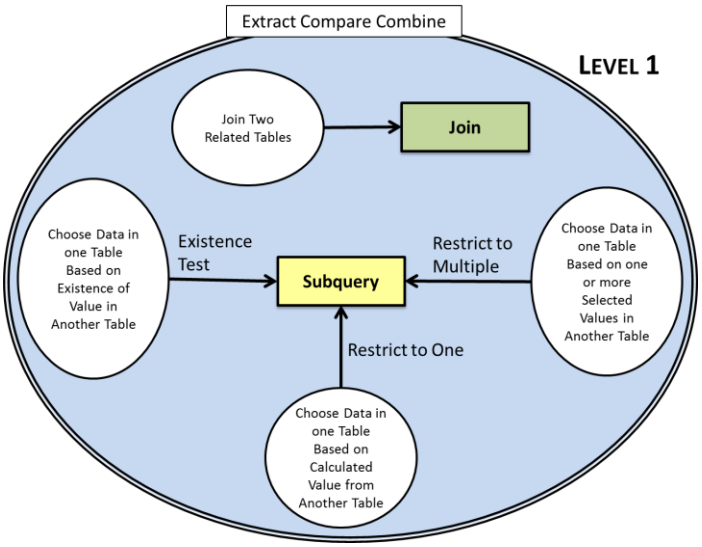


Figure 6.23: Level 1 of Pattern Presentation

Condition analysed	Pattern
Data in the Borrower table needs to be restricted based on existence of data in Loan table	Subquery
Only report late books: compare due date to hand in date. Provide author names	Compare Extract

Table 6.8: Checkilist and Related Patterns-Example 1

Step 3: problem application and Patterns' connection

Level 2 (see Figure 6.24) presents further refinement into lower levels. Each pattern in Level 1 could be connected to many patterns in Level 2. For example, Subquery pattern is connected to Where IN/ Not IN pattern (See Figure 6.25).

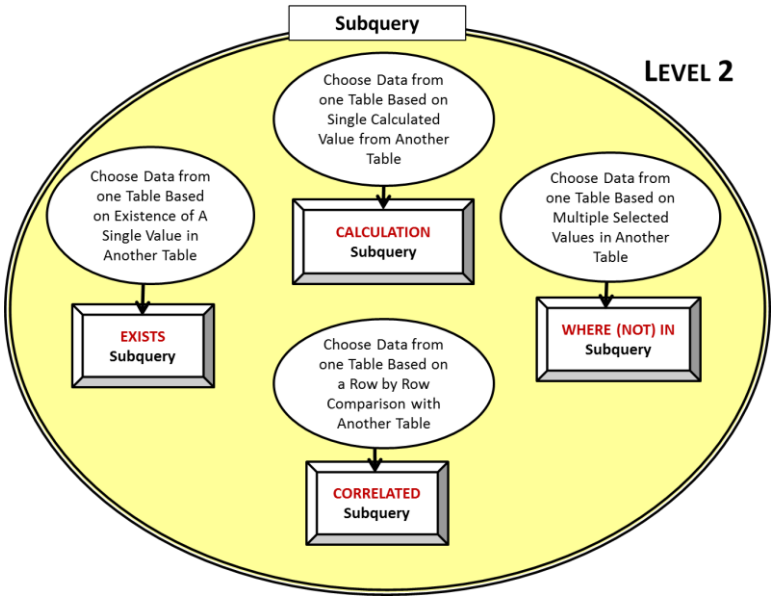


Figure 6.24: Level 2 of Pattern Presentation

The refinement of the component-level model continues until each pattern performs a simple function. That is, until the process presented by the model solves the problem in level 0 and the novice can write a complete SQL query (see Figure 6.25).

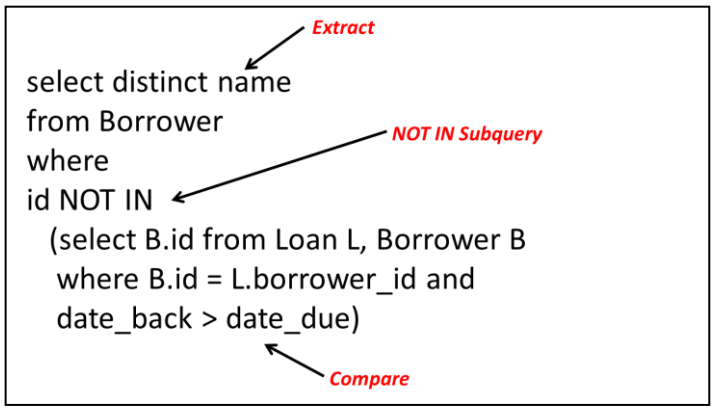


Figure 6.25: Deploying Patterns

Example 2

"For each author, find the total value of the books owned by the library. Give the names of authors with totals of £100 or more".

Step 1: Problem formulation and checklist generation

- ✓ Data is in multiple tables (Author & Book): *Combine*
- ✓ Calculate the total of the book values: *Describe*
- ✓ Restrict the results for total is 100 or more: *Compare*
- ✓ Print the author name: *Extract*

Step 2: Problem analysis and checklist connection

To solve the problem, we need to match the defined checklist to the available patterns (table 6.8).

Condition	Pattern
The value of each author's books needs to be retrieved. Author name in one table, value in another: data is in multiple tables	Join
Calculate the total value Restrict the total value to those total is 100 or more Get author name and total value	Aggregation Restriction Extract

Table 6.9: Checkilist and Related Patterns-Example 2

Step 3: Problem application and Pattern Connection

In this example, the pattern “Aggregation” is linked to other patterns such as “Aggregation Function”, and “Restrict Result”.

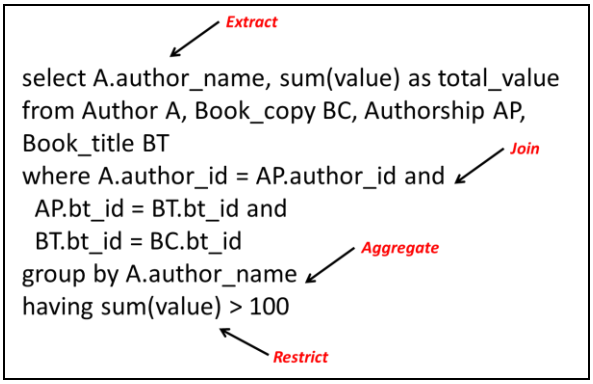


Figure 6.26: Deploying Patterns

The refinement of the component-level model continues until each pattern performs a simple function. That is, until the process presented by the model solves the problem in level 0 and the novice can write a complete SQL query (see Figure 6.26).

6.5.3 Discussion

During problem solving, novices use the pattern collection to guide and inform the correct SQL writing process. Figure 6.27 depicts the problem solving process using the checklist and graphical mechanisms.

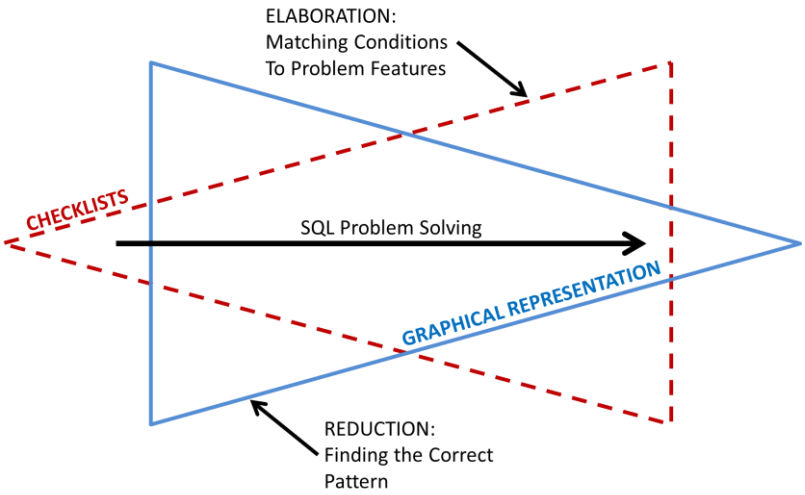


Figure 6.27: Problem Solving Using the Pattern Collection (adapted from Luseau's design Model [327])

Having the ability to match the problem features to the given list of conditions eases understanding of the problem context. Thus, novices are assisted in selecting the correct patterns. Further investigation is required to evaluate the effectiveness of the proposed approach in terms of novice query acquisition and problem solving performance.

6.5.4 Summary

The aim of this research was to find ways of organizing, categorizing, managing and maintaining patterns within pattern collections. The research focused on finding a suitable technique to structure and organize the SQL patterns, so that novices can easily understand and use them effectively. They should lead students to the correct pattern, and also help them to understand how patterns are linked to each other. It is believed that the combination of checklists and graphical notation provides us with a suitable patterns' organization mechanism.

The next section report the phases of SQL patterns' evaluation.

6.6 SQL Patterns' Evaluation Process

SQL patterns were evaluated in all the three phases of their design. The first evaluation was conducted during patterns' initial design (phase1) which aimed at refinement of the patterns (Figure 6.2). Phase 1 consisted of different evaluation methods. In this instance, case study evaluation and interviews with academics and students were used. In phase 2 evaluation, a set of patterns was carried out by collecting feedback from both patterns writers and novices. During phase 3, the impact of patterns on learners' performance was determined. Phase 3 is an empirical research process in which SQL patterns are used in a learning environment.

Quantitative methods, such as: observation, questionnaire and tasks analysis, were used. Phase 2 included PLOP shepherding process and interviews with students. The following subsections describe and report on the evaluation methods in phase 1.

6.6.1 Evaluation of Phase 1 of SQL Patterns Design

The main purpose of phase one evaluation was to assess the design of SQL patterns by guiding the researcher to capture the optimal foundation and structure of the patterns' structure and content. Thus, patterns' forms and content of SQL knowledge such as the text, diagrams and examples is formed and refined. The following are the qualitative methods that are used at this phase of SQL patterns evaluation to collect feedback from experts:

- First: Aesthetics and Usability Expert Evaluation: After the first set of patterns were identified and structured, they were discussed with academics to assess their content validity.

- Second: Subject Matter Expert Evaluation: The set of patterns was sent by email to five academics who taught database courses, and feedback was requested in terms of:

1- Content validity in terms of SQL content, syntax and whether the content was easy to understand, the use of terminology, examples, and diagrams.
2- Usability:
   - To what extent do you agree that these patterns can be used with the current teaching materials?
   - To what extent do you agree that these patterns will help learners in understanding SQL related knowledge?
   - To what extent do you agree that these patterns will help learners to solve the given problems by writing the correct SQL?

All the response were collected and analysed.

3- Case study: explores the use of SQL patterns in solving SQL problems outside the learning stage. In other words, this research tested individual people who were not attending any SQL courses at the time of experiment on how they experienced SQL patterns and how useful students, who were not enrolled in a formal class, found the use of patterns.

The research described here focuses on the application of SQL patterns in the process of solving complex queries. A design case offers a realistic framework for exploring, using and observing the usability of SQL patterns in practice. The context of research was solving SQL problem using SQL patterns. The task was to solve the given problem first without SQL patterns, and then using some relevant SQL patterns. The following SQL problem was given:

Write an SQL statement to find all employees who earn more than the average salary in their department. Display Last name, Salary, Department ID and the Average salary for the department. Sort by Average salary.

The participants were provided with the following patterns in the second stage of the task:

- “Group Function” pattern
- “Grouping Rows” pattern
- “Sub Queries” pattern
- “Querying from one table twice” pattern

The study results were analysed in light of a pre-task, post-task questionnaire and interview. A pre-task questionnaire was given to the participants to provide insight on their level of knowledge and experience in SQL.

Five PhD students participated. All considered themselves novice SQL developers. One student had worked with SQL before. Three students had 0-6 months and two had more than 3-5 years’ experience in working with SQL.

The participant’s solutions (with/without patterns) were analysed as follows:

- Skills in exploring the problem and identifying the related facts,
- Ability to match the given patterns to the given SQL problem, and
- Correctness of the SQL query.

The results revealed that most of the participants could not solve the problem without the given patterns as all claimed that it was hard to remember how to solve the query and all they could remember was the select statement. All of them agreed that SQL patterns helped them to recall their knowledge and provided them with the core SQL constructs they required to solve the given problem. However, most of the participants could not produce a 100% correct solution. Common participant errors include missing the linkage clause from self-join table query and they did not include the non-aggregated attributes in the GROUP BY clause although the given patterns included such information.

The main contribution lies in the fact that this case study investigates the usefulness of SQL patterns from the participant’s point of view and at the same time how correct the participant’s solution is. The current study is too small to be conclusive, but what emerges is that a more substantial study is required to

confirm the value of SQL patterns in helping the novice to solve more complex queries. The study is considered an indirect assessment of the conventional approach in teaching SQL patterns.

The results of this case study along with the patterns were discussed in TLAD 2010 [20]. All the collected feedback from TLAD workshop's participants was used to enhance the patterns design and development process.

6.6.2 Evaluation of Phase 2 of SQL Patterns Design

At phase 2, different kinds of evaluations were used. The importance of this kind of evaluation comes from some issues that have been explored by other research. Schlager and Ogden [13] argue that without an understanding of the task domain from the users prospective, any guidelines' documents are invalid. Phase 2 consists of different tasks of evaluation that considers experts and users in patterns' field. The following are the different evaluation procedures:

One-to-one evaluations with patterns' writers

In relation to Dick and Carey [104], the purpose of the one-to-one evaluation is to gather initial information about the clarity of the material, the impact that the instruction will have on the intended audience and the feasibility of the patterns, given the available time and context.

To do that, SQL patterns were submitted as a paper to European Conference on Pattern Language of Programming (Euro PLOP 2011). The paper initially went through a shepherding process, which is essentially a reviewing process. There were two types of shepherding: by email and face to face during the conference. At this task, SQL patterns structure had changed from their initial format as it was shown in Table 6.3 to a new structure as was illustrated in example in Table 6.5. Hence, new elements were added to each pattern such as Context, Forces and Consequences.

Focus group

During the Writers' Workshops, the set of patterns were discussed where a group of 5 patterns' writer sat together to evaluate the patterns in terms of clarity, validity and usability. All the feedback was collected and added to the patterns for improvement.

6.6.3Evaluation of Phase 3 of SQL Patterns Design

According to Flagg [328], the field test or implementation formative evaluation requires "testing the effectiveness of the [SQL patterns] under approximately normal use conditions". (p. 6)

All the details of the design of this evaluation are discussed in section 4.7. In addition, the results are reported in detail in chapter 7.

6.7 Chapter Summary

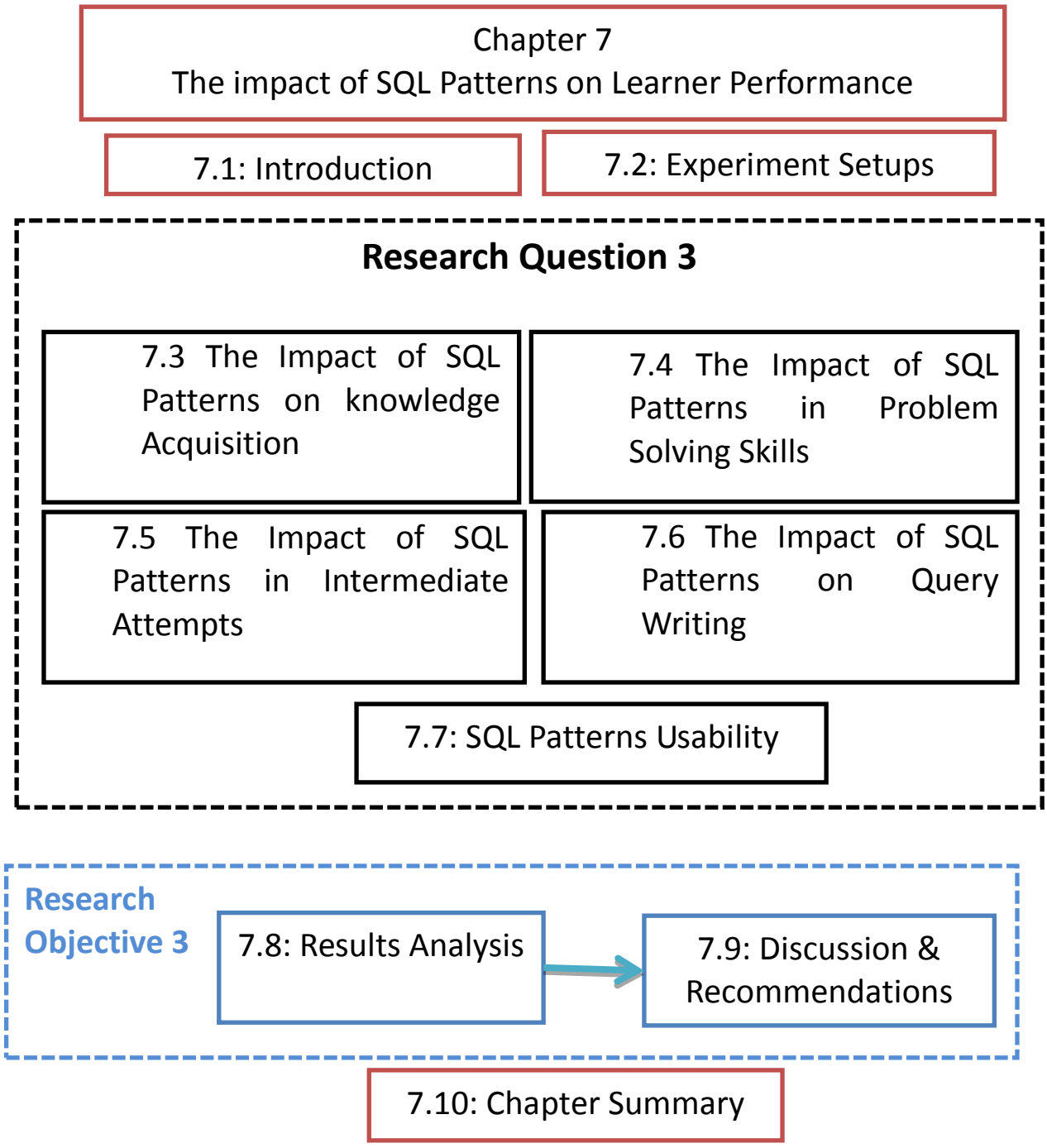
Patterns are not a collection of ideas and imaginary scenarios but, rather, they describe specific problems and solutions that exist in a particular field which facilitate and enable appropriate knowledge and experience transfer. The successful design process of any instructional material (SQL pattern) will depend on the designer understanding all the different factors that influence its learnability such as: learner characteristics, language specifications, human cognition and instructional material. For SQL patterns design, the writer must align with established wisdom about human cognition. Here, it has provided the guidance to inform SQL pattern content, which should ultimately serve as the link between the task requirement and the generic pattern.

The aim of this research was to find ways of identifying, structuring, organizing, categorizing, and managing patterns within pattern collections. The research focused on finding a suitable technique to structure and organize the SQL patterns, so that novices can easily understand and use them effectively. The

purpose was to lead students to the correct pattern, and also help them to understand how patterns are linked to each other. The combination of checklists and graphical notation could provide a suitable patterns' organization mechanism.

SQL Patterns provide the deficient knowledge in a convenient format and does not rely on the learner's own knowledge, which might well be completely wrong. The required knowledge is provided in a format that matches learners' cognition and the learning process for the learner to use. It is believed that learner's knowledge and mastery experience will be enhanced. This research continued by conducting further research in order to refine SQL patterns structure and collection. Furthermore, empirical evidence of SQL patterns efficacy in facilitating learning will be evaluated in chapter 7. Moreover, future research in an evaluation of the impact of SQL patterns on experts who used SQL for a period but are not SQL users anymore is suggested to determine how patterns helped them to retrieve their pre-existing knowledge and allow themselves to re-establish their previous mastery.

Chapter 7 discusses the evaluation of the use of SQL pattern collections and pattern languages in education during the learning and assessment stage.


Chapter 7
The impact of SQL Patterns on Learner Performance
7.1: Introduction
7.2: Experiment Setups
Research Question 3
7.3 The Impact of SQL Patterns on knowledge Acquisition
7.4 The Impact of SQL Patterns in Problem Solving Skills
7.5 The Impact of SQL Patterns in Intermediate Attempts
7.6 The Impact of SQL Patterns on Query Writing
7.7: SQL Patterns Usability
Research Objective 3
7.8: Results Analysis
7.9: Discussion & Recommendations
7.10: Chapter Summary

Chapter 7: The impact of SQL Patterns on Learner Performance

This chapter elaborates on the processes involved in evaluating the impact of SQL patterns compared with more traditional instructional materials, and the results associated with this evaluation process.

7.1 Introduction

SQL pattern evaluation focuses on both the behavioural and the cognitive aspects of SQL acquisition. Understanding learner ability to perform different cognitive tasks such as query analysis, synthesis and writing is essential to be able to assess the new instructional design, viz. the proposed SQL patterns.

SQL patterns were evaluated during all the three phases of their design process as explored in section 6.6. This chapter reports the results of SQL pattern evaluation in phase three. The purpose of the evaluation was to determine whether there is a link between the way individuals learn to solve problems and the way knowledge and problem solving skills encoded from SQL patterns compared to traditional material such as lecture notes, books and any search engine.

To evaluate the impact of SQL patterns in enhancing understanding and application of SQL, a model of SQL learning, which was proposed in chapter 5, was used as a base for this evaluation. There were three primary research questions, as illustrated in the following table.

Research Question 3: What is the impact of SQL patterns in learners' performance?			How was data derived?	How were the results analysed?
1	Do SQL patterns improve SQL knowledge acquisition?		Pre-test Post-test	Quantitative data analysis
2	Do SQL patterns improve the following aspects of novices' performance?			
	A	Problem solving	Problem solving test	Quantitative data analysis
	B	Intermediate attempts	Query writing test	Content analysis
	C	Query writing	Query writing test	Quantitative data analysis
3	How have participants felt about the efficacy of the patterns?		Questionnaire	Quantitative data analysis

Table 7.1 : Research Question 3

This Chapter follows the following structure:

Main topics			**Data presented in section**	**Data analysing reported in section**
1	Do SQL patterns improve SQL knowledge acquisition?		7.3	7.8.1
2	Do SQL patterns improve the following aspects of novices' performance?			
	A	Problem solving	7.4	7.8.2
	B	Intermediate attempts	7.5	7.8.3
	C	Query writing	7.6	7.8.4
3	How have participants felt about the efficacy of the patterns?		7.7	7.8.5
Discussion in 7.9				
Conclusion in 7.10				

Table 7.2: Chapter Structure

7.2 Experimental Setups

The design of the experiment followed the standard methodology and its outcomes were statistically evaluated. Its principal steps were as follows:

- Step 1: Experiment formulation and design. The details were provided in section 4.7. This involves the formulation of research hypotheses (see section 4.7.1) and Experiment setup as discussed in section 4.7.3.
- Step 2: Constructing the experiment described in section 4.7.5.
- Step 3: Data collection, Analysis and Interpretation. This is reported in this chapter.

Since the experimental design details were presented in section 4.7, here, an overview about the experiment main steps is illustrated is Figure 7.1.

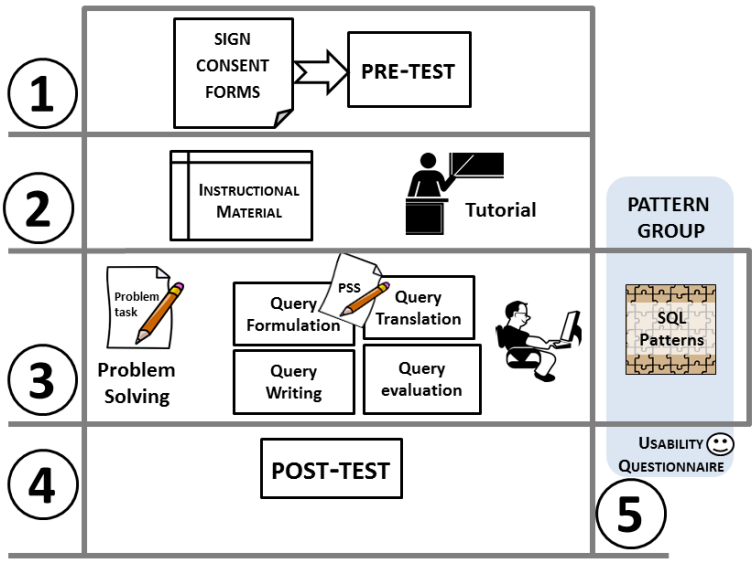


Figure 7.1: Experiment Procedures

The following are the steps of the experiments' procedures:

- All participants were asked first to read and sign the informed and consent forms (see Appendix L).

- Then, they took the pre-test (see Appendix H).
- Participants in each group were given the same tutorial on particular SQL concepts (see Appendix K).
- They were handed with the experiment material that consisted of either the patterns used for the experiment or other material such as lecture notes. Experiment group received five patterns: Natural join, Self-Join Pattern, Grouping Result Pattern, Filtering by Existence Pattern and Dynamic Filtering Pattern
- All participants received a task sheet (see Appendix J); PSS sheet (see Appendix M). The subjects were asked to use PSS forms to analyze and synthesize each question in the given task.
- They were asked to use the SQLPB tool to write the query.
- Then, all took the post test (see Appendix I).
- The patterns group filled in a usability questionnaire in their own free time (see Appendix N).

The experiment took place at the Higher College of Technology in Muscat, Sultanate of Oman. Current enrolments at HCT stand at 12,000 undergraduate students. The Information Technology Department offers four specializations: database, software engineering, network and multimedia. These are delivered through three levels: Diploma, Higher Diploma and Bachelor of Technology (B. Tech.). Ninety students participated in this study from the Database specialization as shown in Table 7.3. Students were randomly assigned to two groups: control and patterns.

Condition	N	Pre-test	evaluation task	Post-test
Control	48	48	30	30
Patterns	42	42	32	32

Table 7.3 : The numbers of students who participated in the tree tests in both groups

All of the students studied a course called SQL Syntax, where most of SQL concepts were covered. None of the students had any prior work experience in SQL. The participants' level of knowledge before and after the experiment is assessed using pre-test and post-test is discussed next. The next section reports the participants' knowledge as assessed using pre -tests.

7.3 The impact of SQL Patterns in Knowledge Acquisition

The first investigation starts with looking at SQL learning taxonomy that describes the learning stages at which a learner is learning a specific topic in SQL. To examine participants' ability to "Remember" the tested concepts and their ability to "Understand" and recognize their applicability, pre-tests were conducted. The results of both groups' pre-tests are reported next.

7.3.1 Students' Understanding of SQL Tested Concepts Prior to the Experiment

The pre-test results were used to investigate students' pre-conceptions of SQL concepts and to compare the two groups to ensure that they were similar in knowledge. Later differences could then be attributed to the experimental treatment.

Group Statistics					
	control	N	Mean	Std. Deviation	Std. Error Mean
Pre-test	C	48	2.9167	1.62210	.23413
	E	42	3.0952	1.72230	.26576

Table 7.4 : Participants Results in the Pre-test

The mean score (out of 12) of the pre-test for the control group was 2.9 (SD = 1.6) and for the experimental group was 3.1 (SD = 1.7). This difference was tested using an independent-sample *t*-test, and the results showed that there

was no significant difference between these two groups in terms of their previous knowledge of examined concepts in SQL (t (90) = -.506, p >0.05).

Independent Samples Test									
Pre-test	Levene's Test for Equality of Variances		t-test for Equality of Means						
	F	Sig.	T	Df	Sig. 2-tailed	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
								Lower	Upper
Equal variances assumed	1.710	.194	-.506	88	.614	-.17857	.35275	-.87959	.52245
Equal variances not assumed			-.504	84.787	.615	-.17857	.35418	-.88280	.52566

Table 7.5: Percentage of Students' Response in the Pre-test and Summary of Independent-Sample t-test (control group N=48, Experimental group N= 35 note: * not significant at 0.05)

Thus, it could be reasonably being assumed that students in the experimental group were comparable to students in the control group (see Tables 7.4 and 7.5). The results of both groups' post-tests are reported next.

7.3.2 The Impact of Patterns in Students' Understanding of SQL Concepts in Response to the Experiment

Table 7.6 and Table 7.7 present the students' post-test answers to the diagnostic questions. The results of the post-test indicated that mean score (out of 12) of the post-test for the control group was 5 (SD = 1.5) and for the experimental group was 6.47 (SD = 1.9).

Group Statistics					
	Control	N	Mean	Std. Deviation	Std. Error Mean
Post-test	C	29	5.00	1.535	.285
	E	30	6.47	1.907	.348

Table 7.6: Participants' Results in the Post-test

Both groups had better results in the post test as compared to the pre-test results. However, students in the patterns achieved considerably higher learning gains in examined concepts than students in the control group (t (59)= - 3.25, p <0.005) based on the post-test results.

Independent Samples Test									
Post-test	Levene's Test for Equality of Variances		t-test for Equality of Means						
	F	Sig.	T	Df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
								Lower	Upper
Equal variances assumed	2.725	.104	-3.247	57	.002	-1.467	.452	-2.371	-.562
Equal variances not assumed			-3.259	55.216	.002	-1.467	.450	-2.368	-.565

Table 7.7: : Percentage of Students' Response in the Post-test and Summary of Independent-Sample t-test (control group N=29, Experimental group N=30) note: * not significant at 0.05

A paired-sample test (see Table 7.8) was used to determine whether there was any improvement in students' acquisition from the pre-test to the post-test. Questions 3, 4, 6, 7, 8a, 8b, 9a, and 9b were used in the pre-test and the post-test. The paired-sample test results showed that participants in both groups had different results in some questions. For example, in question 6: in the control group showed no significant improvement in their understanding in response to the tradition material (p> 0.005). On the other hand, participants in the experimental group showed statistically significant improvements (p<0.005).

Paired Samples Test for control group		t	df	Sig. (2-tailed)	Paired Samples Test for experimental group		t	df	Sig. (2-tailed)
Pair 1	Question3 - Question3	-1.279	28	.212	Pair 1	Question3 - Question3	-.902	29	.375
Pair 2	Question4 - Question4	-.902	28	.375	Pair 2	Question4 - Question4	-1.989	29	.056
Pair 3	Question6 - Question6	-.701	28	.489	Pair 3	Question6 - Question6	-3.010	29	.005
Pair 4	Question7 - Question7	-2.703	28	.012	Pair 4	Question7 - Question7	-5.037	29	.000
Pair 5	Question8-a - Question8-a	-3.550	28	.001	Pair 5	Question8-a - Question8-a	-5.385	29	.000
Pair 6	Question8-b - Question8-b	-3.087	28	.005	Pair 6	Question8-b - Question8-b	-3.612	29	.001
Pair 7	Question9a - Question9a	-1.440	28	.161	Pair 7	Question9a - Question9a	-2.249	29	.032
Pair 8	Question9b - Question9b	.812	28	.424	Pair 8	Question9b - Question9b	-1.795	29	.083

Table 7.8: Paired-Sample Test

To examine the impact of the instructional materials on participants' ability to solve the problem, another task was conducted. The results are presented next. The following section reports the data that were collected regarding participants' problem solving skills.

7.4 The Impact of SQL Patterns in Problem Solving Skills

From the previous research and the results of section 5.4.3, it was concluded that students' understanding of problem solving strategy in learning SQL is an essential issue that educators need to focus on. An insight into the factors that affect the novice SQL learner's skill in solving queries was given in section 5.5.3.

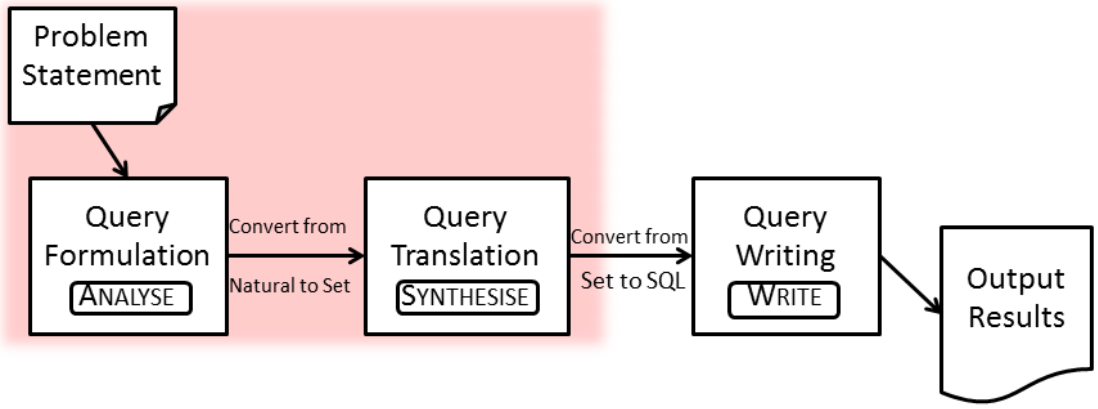


Figure 7.2: Novice Problem Solving Task

This section reports on an investigation which was conducted to find out if the use of patterns had any impact on participants' problem solving skills and, consequently, their performance. Since each pattern had a context, force, problem, solution and example, the focus is on how learners used the different parts of the pattern and if the main purpose of the pattern concepts has been helpful and efficacious.

The participants' problem solving performance can be measured based on the cognitive activities in the SQL problem solving cognitive model (see Figure 7.2), including:

1- Analysis/Query Formulation: ability to recognize the context of the given scenario by successfully matching the given scenario to the correct base

problem (pattern). Thus, a better performance in their problem analysis task. This is related to query formulation in the model.

2- Synthesis/Query Translation: ability to abstract the actual problem in the scenario and identifying the required data to solve the problem. In addition, to list all the difficulties in solving the problem and other possible solutions. This is related to query translation in the model.

7.4.1 Query formulation and translation.

During problem solving, learners perform different cognitive activities as discussed by [13, 57, 59] (see Figure 7.3). The initial task is what is called Query formulation [13, 59] discussed in section 2.2. This is the analysis stage which students decide what data they need to solve the problem. To do this, the proposed model suggested that students need first to identify the scenario context. Then, learners are assumed to be able to translate the problem in terms of relational sets; this is called query translation or synthesis stage[13, 59].

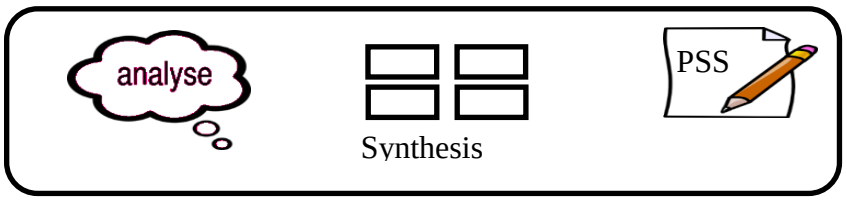


Figure 7.3: Analysis & Synthesise Stages

To guide students' to analyse and synthesise the given question during the task, the participants were given a template of "Problem Solving Strategy" (PSS) sheet (see Table 7.9) to help them to plan the tasks' solution.

Question1 Q1: list the name of all customers who have more than one order		Marks (0-4)
Concepts	Self-join and natural join	1
The tables that I need to work with.	CUSTOMER, PURCHASE_ORDER	0.5
The columns that I need to use.	customer.NAME	0.5
Any relation between the tables.	where c.CUSTOMER_ID= po1.CUSTOMER_ID and c.CUSTOMER_ID= po2.CUSTOMER_I	1
Any calculation or conditions that need to be done.	and po1.ORDER_NUM <>po2.ORDER_NUM	1

Table 7.9: Marking Schema for Sample Answer to Problem Formulation and Translation

They were requested to identify the relevant facts embedded in the given problem, such as context, data sources (table and column name). Subsequently, they were requested to specify how the data should be processed to identify the relation between the tables; mentioning any mathematical calculation, and anything else that was relevant. Table 7.10 gives an overview about each question used in the task.

Question number	Question type or idea
Question 1	join of two copies of the same table
Question 2	SUM function, Use GROUP BY to group the result, Filter the result with Having
Question 3	Data need to be filtered based on dynamic criteria, query can be generated temporary that have such dynamic information
Question 4	The main query is filtered based the existence of at least one matched result in 2(sub query-Exists operation)
Question 5	Using Count function and using GROUP BY to group the result

Table 7.10: Problem Solving Task

Table 7.11 shows the number of participants who submitted the PSS or other related sheet.

Query formulation type	Control	Experimental
PSS sheet	4	15
other sheet	12	8
Start the solution	14	9
Total (N)	**16**	**23**

Table 7.11: Number of Participants Submitted doc for Analysis and Synthesise

To evaluate the submitted data, the following sample answer and related rubric was used as shown in Table 7.9. The evaluation was conducted into two ways. For those who submitted the PSS sheet, their answers were compared directly with the sample answer such as in Table 7.9. For students who preferred to write in a blank paper, their answers were analysed in terms of the required data using the same elements in the PSS sheet. The average of each question is shown in Table 7.12.

Question	Q1	Q2	Q3	Q4
AVG-Pattern(N=23)	1.586957	1.840909	1.695652	0.565217
AVG-Control (N=16)	1.1875	1.84375	1.40625	0.4375

Table 7.12: The Average of each Question

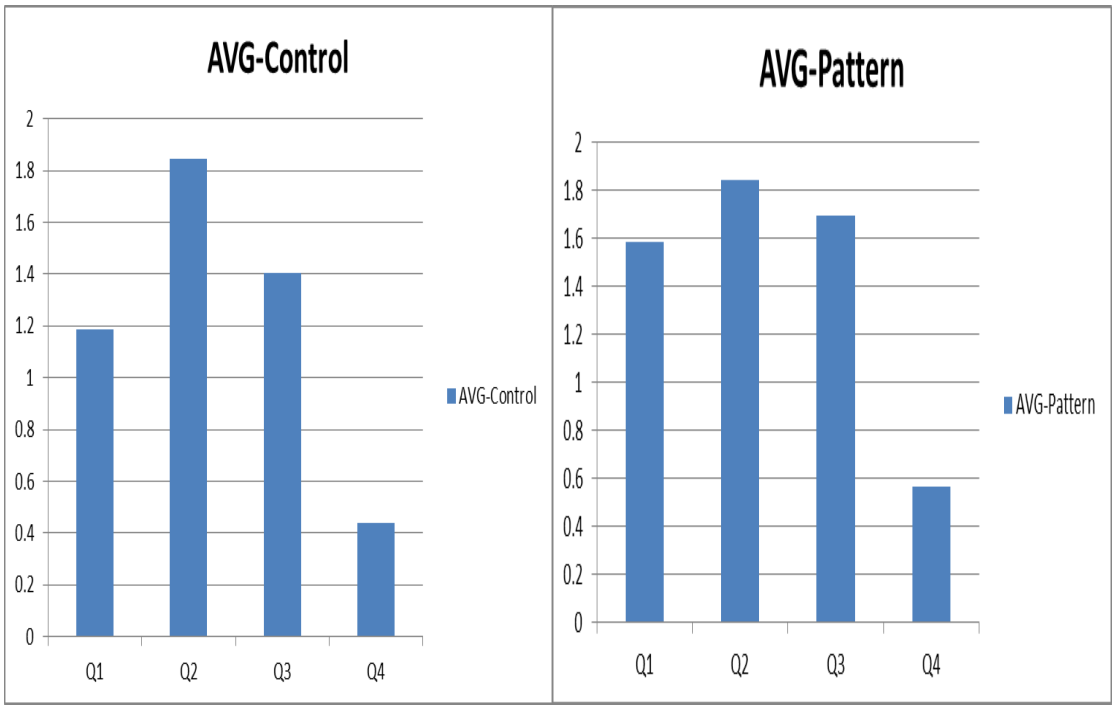


Figure 7.4: The Average of PSS-Control (left), The Average of PSS-experiment (right)

Figure 7.4 show the difference between the two groups. This finding are analyzed and discussed later.

7.5 The Impact of SQL Patterns in Intermediate Attempts

This section presents an investigation into the errors novices make when writing SQL queries as a result of solving the given task. The task used is the same as problem solving task discussed in the previous section (see Table 7.10)

This process analyses and categorizes students' attempts to gain an insight into the strategies novices deploy and errors they commit. Here, SQL errors attempted by both groups are explained by using Reason's and Rasmussen's models [329, 330] of human error focusing on the participants' cognitive behaviours'. No prior publications report on novice SQL problem solving. There has, however, been significant research into error in the area of teaching programming languages.

Researchers in Computer Science highlighted the importance of investigation into how students solve programming tasks. There is a need to provide an

> "explanation of programming skill that integrate ideas about knowledge representation with strategic model, enabling one to make prediction about how change in knowledge representation might give rise to particular strategies…"[331]

Therefore, the employed methods were adapted from different research areas, such as error analysis in linguistic research, error analysis in programming languages and other query languages. The next section presents the method used to graphically present students' attempts.

7.5.1 Decision tree

The aim of decision tree is to visualize the two groups' attempts. According to De Ville [332]

> "Decision trees are a form of multiple variable (or multiple effect) analyses. All forms of multiple variable analyses allow us to predict, explain, describe, or classify an outcome (or target)"

Although decision trees have not previously been used to analyse student attempts, it is possible to argue that decision trees might be a powerful tool in analysing student performance, since the tree is developed incrementally. It depicts a collection of one-cause, one-effect relationship [332] organized recursively. In addition, the decision tree could be used to assess the following:

1. The percentage of correctness (C%) of each attempt for each question. This gives an indication of cognitive performance.
2. The percentage of completion of each attempt for each question and the number of attempts (T1,..,Tn) for each question.
3. The time consumed between each attempt and the total from the initial attempt to the final query.

In this research, the decision tree was used to provide information as follow:

1. The percentage of correctness (C%) and of last attempt for each question.
2. The total time from the initial attempt to the final query.
3. Evaluate the behavior of students in terms of their development during the task, such as: switching between questions and number of attempts per question.
4. Identify, classify and categorize the errors that students make in both groups.

The Decision tree, as shown in Figure 7.5, examined the way in which SQL knowledge was used and applied. It attempted to identify relationships between the instructional material and student behaviour in a group of observations that form a data set. Thus, participant strategies in solving problems were recognized and visualized in the following scenario:

“Query solving strategy is a process where learners make attempts (Si) to solve the given scenario by giving an SQL query and then evaluating the execution of the query (Correct and complete). The feedback learners were getting from the first attempt was either proved (100% correct), if it met the desired goals, or rejected, if not. In the latter case learners either could try again or give up”. The following are two examples of the decision tree:

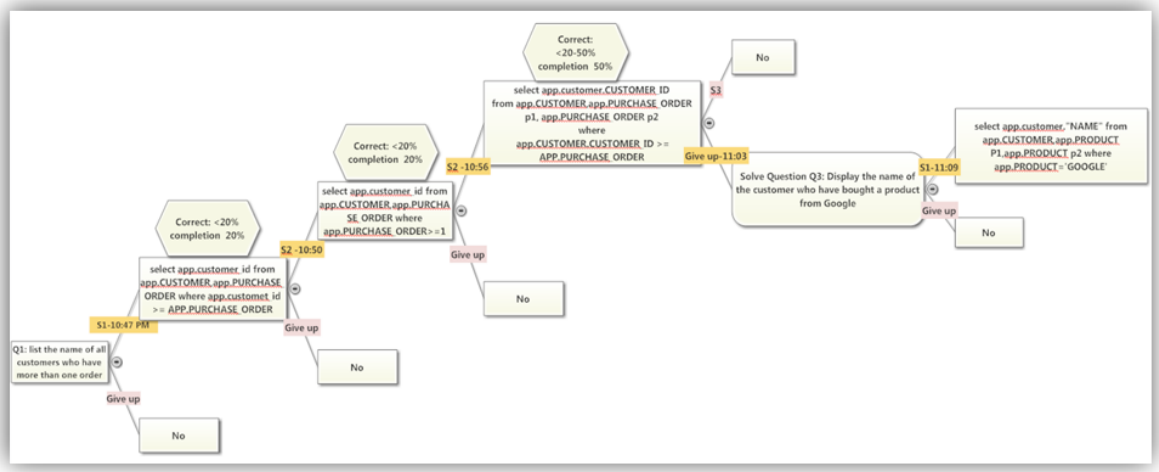


Figure 7.5: A sample of a decision Tree

The next section reports the data collected from the decision trees.

7.5.2 Overview of learners’ strategy

To give an overview of each student’s performance the measurements matrix, Table 7.13 was used. It consolidates the data presented in the decision trees. This analysis includes two types of data. The first provides an overview of the students’ strategies in solving the task. The second presents a detailed measure related to students’ performance for each question. This evaluation is discussed over the rest of the chapter.

No	Question					
1	Did the student solve all main four questions? How many questions did they answer?	QN	One	Two	Three	Four
		SN				
2	Which type of question did the student solve?	QN	Q1	Q2	Q3	Q4
		SN				
3	Did the student jump from one question to another without completion?	Yes			No	
4	How many attempts the student gave to this question					
Detailed analysis for each question(total four questions)						
	Student's percentage of completion of the final submitted	0%	20%	50%	75%	100%
	Student's percentage of correction of the final submitted	0%	<20%	20-50%	60-90%	100%
	How much time was spent on the question (in minutes)?	0 m	< 5 m	5-10 m	15-30m	30 or more

Table 7.13: Participant Performance Matrix

The next subsections reports the data collected from both groups using the following aspects:

- The number of questions solved per student.
- The type of question solved by students.
- Students behaviour (jumping from one question to another)
- Number of attempts per question.

7.5.2.1 Data from Control Group

To answer the first question: **Did the student solve all four main questions?** The number of questions solved by students in the control group is shown in Figure 7.6 with average 2.99.

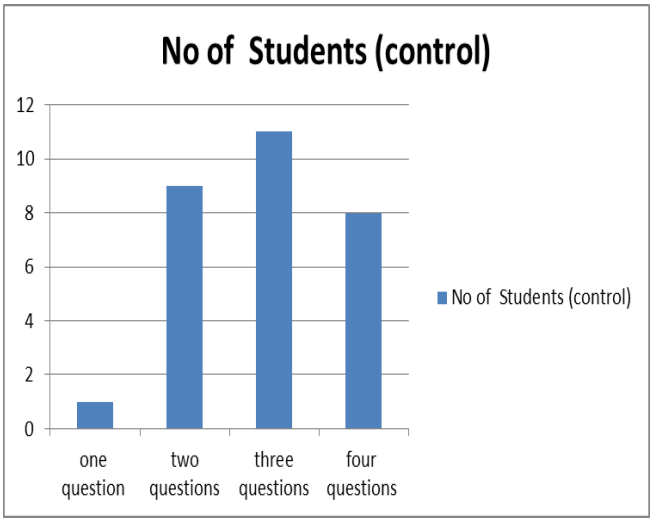


Figure 7.6: The Number of Questions Solved by Students

To answer the second question: **Which type of questions did the students solve?** The answer is shown in Figure 7.7

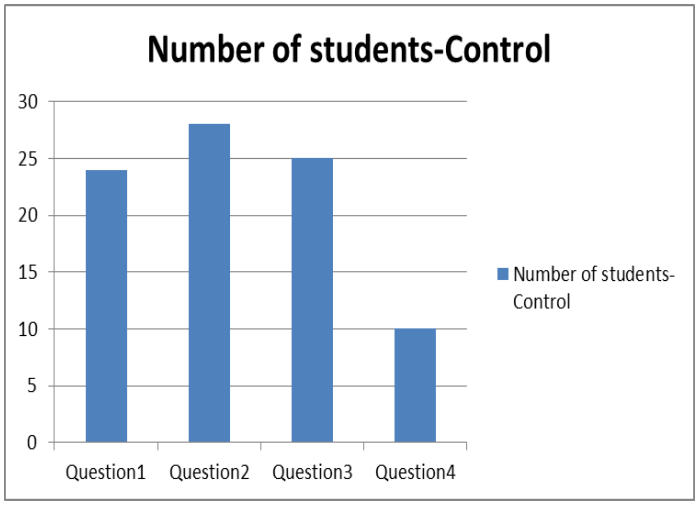


Figure 7.7: Number of Students among the Question Type

To find out student's strategies in solving the questions, the following question needs to be answered: question 3,**did the students jump from one question to another without completion?** The answer is shown in Figure 7.8.

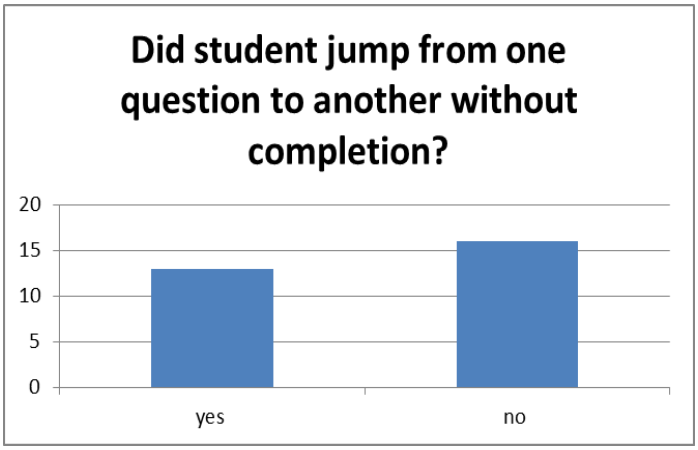


Figure 7.8: Students Strategy in Solving the Questions

Question 4: **How many times did the student attempt to solve the question?** Since the previous question shows that many students jumped from one question to another without completion, Table 7.14 show the strategy of counting the attempts.

STD1	Question 1	Question 2	Question 3	Question 4
Round 1	10	8	5	7
Round 2	5	5	6	0
Round 3	2	0	0	0
Total	17	13	11	7

Table 7.14: Example of the Number of Attempts Per Question

The total average of the number of attempts per question is shown in Table 7.15.

Control group	Q1 attempts	Q2 attempts	Q3 attempts	Q4 attempts
Max	18	31	19	13
AVG	8.153846	7.961538	8.615385	2.461538

Table 7.15: Number of Attempts per Question-Control Group

The query completeness, correctness and time required to finish the task is discussed in section 7.5. The next section reports the data that was collected from experimental group.

7.5.2.2 Data from Experiment Group

To answer the first question: **Did the student solve all four main questions?** The number of questions solved by students in the experiment group is show in Figure 7.9.

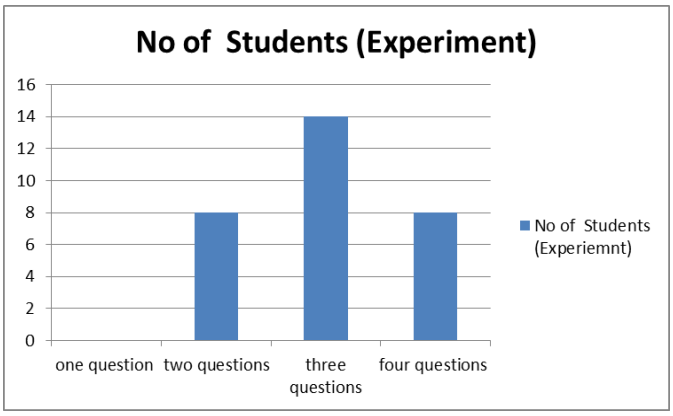


Figure 7.9: The Number of Questions Solved by Students

To answer the second question: **Which type of questions did the students solve?** The majority of students tried the first three questions but a few of them tried question 4 (see Figure 7.10).

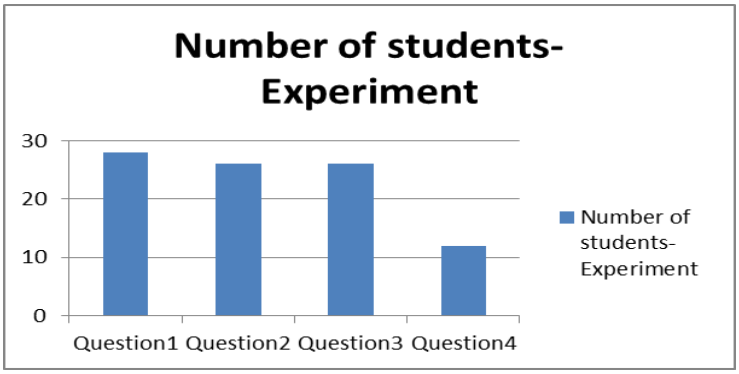


Figure 7.10: Number of Students among the Question Type

To find out students' strategies in solving the questions, the following question needs to be answered: **Did the students jump from one question to another without completion?**

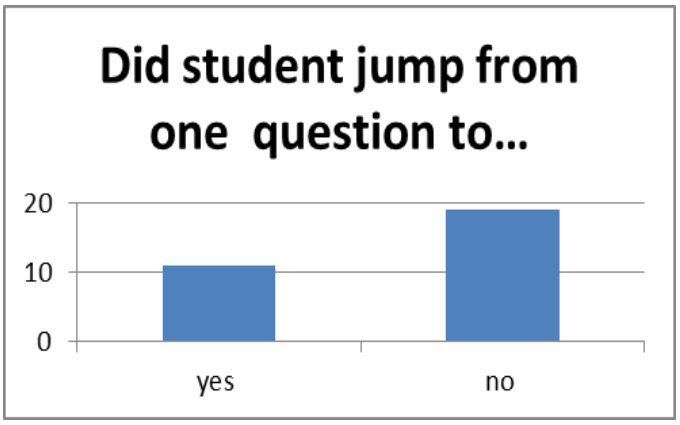


Figure 7.11: Students Strategy in Solving the Questions

For each question in the task the following question was answered: **How many attempts did the student give to this question?** Since the previous question (see Figure 7.11) shows that some students had jumped from one question to another without completion. The total average of the number of attempts per question is shown in Table 7.16.

Control group	Q1 attempts	Q2 attempts	Q3 attempts	Q4 attempts
Max	13	16	7	14
AVG	6.125	4.625	2.625	2.875

Table 7.16: Number of Attempts per Question-Patterns Group

According to Davies [331], research should go beyond characterizing the different strategies employed and focus on why such strategy emerged. The rest of the chapter investigates, in detail, some explanation of the differences between two groups. The next subsections aim to understand what students do during problem solving, what errors they make, how long they persist, and finally whether SQL patterns guide them towards producing accurate queries or not.

7.5.3 Participant Attempts: Collection, Classification and source Identification

The focus of this section is on the participants' attempts. Studying learners' errors has been identified in second language acquisition research [333]. Simultaneously, other fields which studied this include Computer Science education studies such as programming languages [275, 301, 334] and Database design, object-oriented design [335, 336].

As educators and researchers, it is important to improve the learning process, yet it is vital to act prudently. The first requirement is to align our intervention with basic human cognition principles. The second requirement is to consider how to maximize the learner's opportunities to build schemata within the constraints of an inevitable trial and error strategy. The first step in designing interventions, therefore, is to gain an understanding of the kinds of errors novices make during problem solving. To perform this study, four steps were carried out.

1. Collection and classification of all errors.
2. Identification of the source of errors.
3. Measurement of the frequency of the errors.
4. Analysis of errors in terms of SQL language, learner strategies and learning processes for both groups.

The next subsections explore the first two steps. The third and fourth steps are discussed in section 7.7.

7.5.3.1 Collection and Classification of the errors made by the two groups

All query attempts were collected via the SQLPB tool (see Figure 7.12) developed specifically to support this kind of analysis. The tool acts as a front-end client to the database server. It also provides students with an ER diagram of the database structure, and offers query submission and outcome windows.

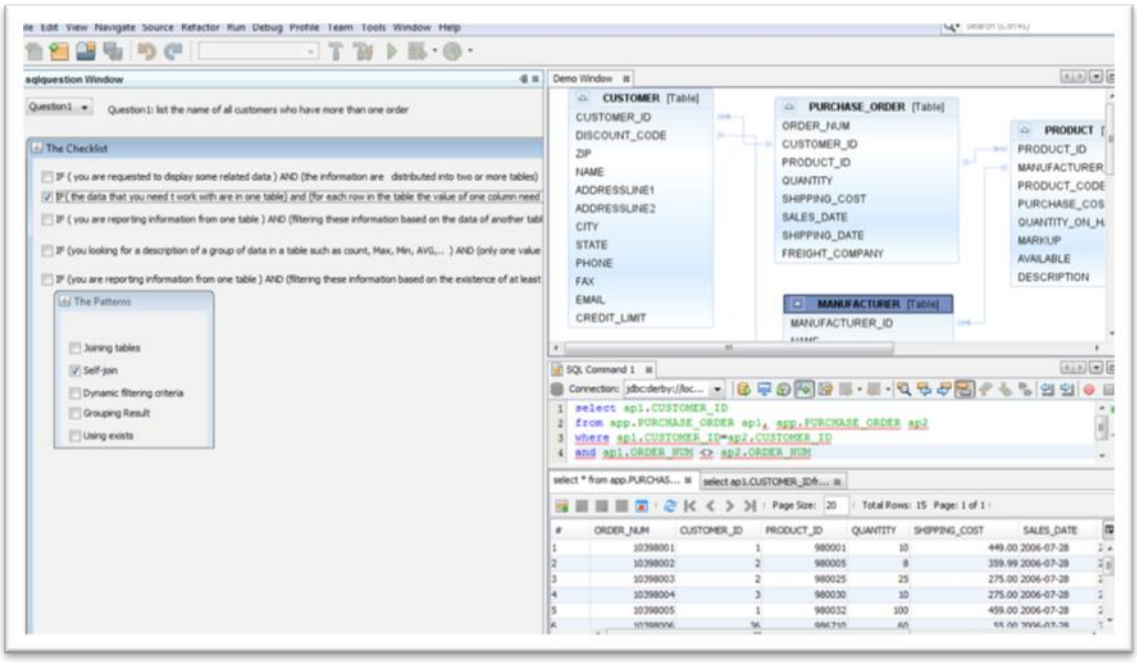


Figure 7.12: A Snapshot from the Tool

The executed queries are saved in the tool. Then, queries were collected and classified according to the related question (see Figure 7.13). Each query was examined and all the errors are highlighted.

```
select A.Customer_ID from PURCHASE_ORDER A,PURCHASE_ORDER B
WHERE A.ORDER_NUM = B.ORDER_NUM group by A.ORDER_NUM having B.ORD       11:54 AM
select A.Customer_ID from PURCHASE_ORDER A,PURCHASE_ORDER B
WHERE A.ORDER_NUM = B.ORDER_NUM group by B.ORDER_NUM having B.ORD       11:54 AM
select A.Customer_ID from PURCHASE_ORDER A,PURCHASE_ORDER B
WHERE A.ORDER_NUM = B.ORDER_NUM group by A.Customer_ID having A.O       11:53 AM
select A.Customer_ID from PURCHASE_ORDER A,PURCHASE_ORDER B
WHERE A.ORDER_NUM = B.ORDER_NUM group by A.Customer_ID having B.O       11:52 AM
select sum(purchase_cost)from product,PURCHASE_ORDER
where product.PRODUCT_ID = PURCHASE_ORDER.PRODUCT_ID group by PURCHASE_O        11:50 AM
select sum(purchase_cost)from product,PURCHASE_ORDER
where product.PRODUCT_ID = PURCHASE_ORDER.PRODUCT_ID group by product.cu        11:49 AM
select sum(purchase_cost)from product,purcahase_order
 where  product.PRODUCT_ID = purchase_order.PRODUCT_ID group by product.        11:48 AM
```

Figure 7.13: A Snapshot of the Collected Queries from the Tool

SQL errors were classified based on both the SQL learning taxonomy and on established wisdom related to human error [337]. The categorization set out in Figure 7.14 is a broad classification of the causes of human failures which can be related to the Skill, Rule, Knowledge based (SRK) approach [329, 330, 337, 338]. They make a distinction between errors and violations. The latter are not applicable here.

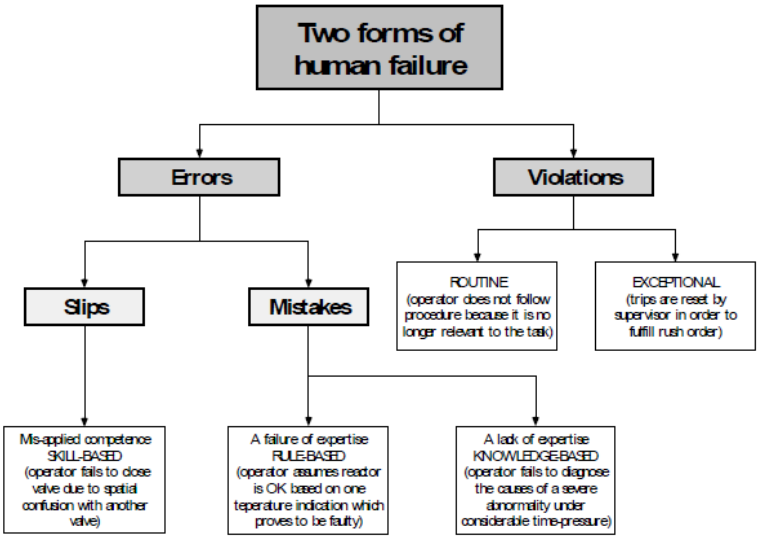


Figure 7.14: Classification of Human Errors adapted from Reason [337]

SQL errors are generally classified as either syntactic or semantic errors [339]. Syntactic errors occur because of an illegal string or character that has been written which is not a valid SQL query [340]. Sematic errors are defined as "any legal query that does not produce the intended results". More than 40 examples of semantic errors was listed in [339].

Brass and Goldberg [339] distinguish between two general types of SQL errors: syntactic and semantic. SQL errors were classified as minor data error, minor language error, error of substance and error of form [5, 33]. Both classifications are unsuitable to employ directly for this study purposes. One could argue that classification in terms of *syntactic* and *semantic* error was too coarse; a more detailed classification was mandatory to understand learners' misconception.

Reisner et. al,; Yen & Scamell [5] and [33]	Reisner [57]	This study	Rasmussen categories
minor language error	Infusion Errors	Knowledge	Knowledge-based
error of substance	Consistency Errors	Comprehension	Rule-based
	Overgeneralization Or Unconscious rule	Analysis	Rule-based
Minor data error	Data type Errors	Synthesise	Rule-based & Skill-based
error of form	Omission Errors	Application error	Skill-based
	Prior-knowledge Errors	External factors	Rule Interference

Table 7.17: The Error Classification Compare to other Research

A new model of error classification is proposed (see Figure 7.15). It combined the SQL learning Taxonomy with the human error classification depicted in Figure 7.14 in the proposed model with a consideration of other research such as [339] and [5, 33] as shown in Table 7.17.

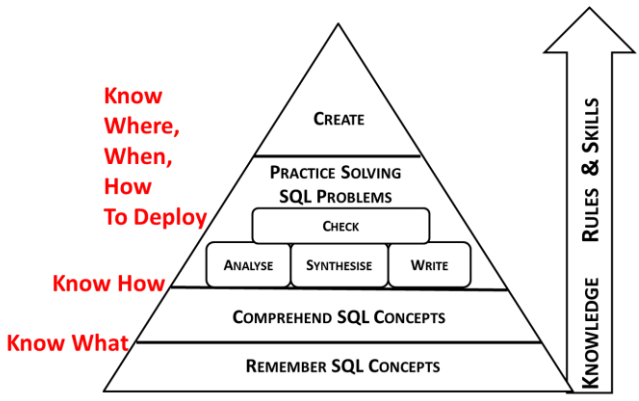


Figure 7.15: Error Classification Model

In this model, the errors were classified into three levels: “know-what” which related to knowledge-based error, “Know-How” that is similar to Rule-based and skill-based error as defined in [329, 330, 337, 338]. The boundary between these

three categories is not quite distinct and depends on the nature of error and the researcher perspective. In this study, the first three levels are considered. The top level is excluded, since only experts perform at this level. Evaluation "why" was also excluded since we did not have evidence to determine learners' assessment of their outcomes in terms of correctness. The process is shown in Figure 7.16.

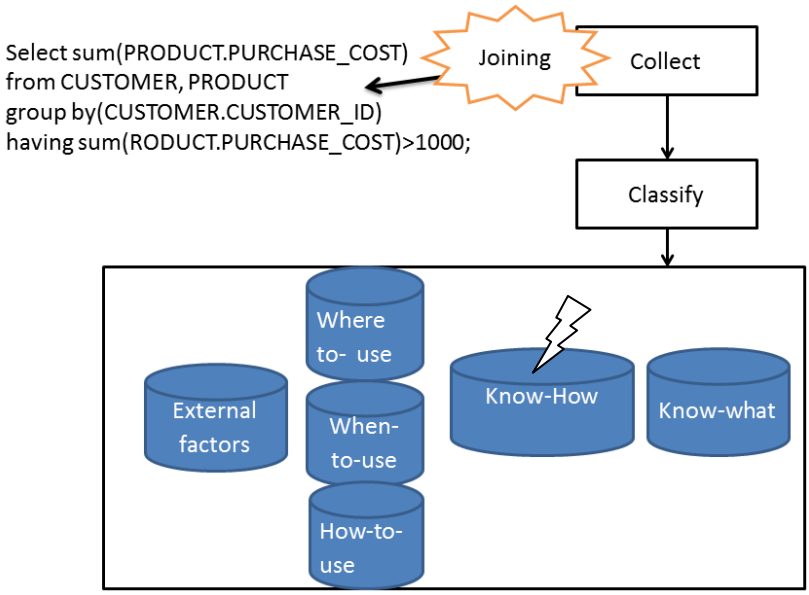


Figure 7.16: Error Classification Process

Error	Error category
select sum(PRODUCT.PURCHASE_COST) from CUSTOMER, PRODUCT group by(CUSTOMER.CUSTOMER_ID) having sum(RODUCT.PURCHASE_COST)>1000;	There is no real relation between the tables
select FREIGHT_COMPANY, count(C.customer_ID) from purchase_order P JOIN customer C where P.customer_ID=C.customer_ID group by P.customer_ID, C.customer_ID;	Know-How-to -use application errors of aggregation No join
select customer_ID,order_num from purchase_order where order_num>1;	Translation errors No join
select A.Customer_ID from PURCHASE_ORDER A,PURCHASE_ORDER B WHERE A.ORDER_NUM = B.ORDER_NUM group by B.ORDER_NUM	Translation errors

having B.ORDER_NUM>1; not applicable	
select name from customer where customer_id ???(select customer_id from purchase_order) = product_id(select product_id from product) And PRODUCT.MANUFACTURER_ID=MANUFACTURER.MANUFACTURER_ID in (select 'name' from MANUFACTURER where name='Googleselect 'name' from CUSTOMER	Application of subquery
select CUSTOMER_ID from PURCHASE_ORDER a1 ,PURCHASE_ORDER a2 where a1.key = a2.key and a1.PRODUCT_ID=a2.PRODUCT_ID;	Knowledge error
select app.CUSTOMER.NAME from customer where app.customer.CUSTOMER_ID in (select app.CUSTOMER.CUSTOMER_ID from app.PURCHASE_ORDER,app.PRODUCT,app.MANUFACTURER where app.MANUFACTURER.NAME='Google');	Missing the joining link in the subquery tables
select P.FREIGHT_COMPANY, count(C.customer_ID) from purchase_order.P JOIN customer.C where P.customer_ID=C.customer_ID group by P.customer_ID, C.customer_ID;	Application of join
select app.PRODUCT.DESCRIPTION from app.PRODUCT, app.PURCHASE_ORDER where app.PRODUCT.PRODUCT_ID exists (select app.PURCHASE_ORDER.PRODUCT_ID from app.PURCHASE_ORDER);	Miss use of Exists Application

Table 7.18: Example of Error Source Identification

Table 7.18 show examples of the collected errors. The analysis of these errors is discussed later. The next section reports the procedures of the identification of the source of errors.

7.5.3.2 Identification of the source of error

Following classification of the errors, the next step is error source identification which is essential in any method of error analysis. Chiang [341] points out that a successful identification of errors is a pre-cursor to any analysis of errors.

Many reasons could underlie human errors [337, 342]. It is crucial to understand what causes the errors before any attempt to reduce errors can be

implemented. Different causes of errors were highlighted and discussed by [343, 344] such as transfer or intra-lingual issues (Over generalization, Ignorance of the role instruction, incomplete application of the rule, and false conception hypothesized).

The reasons students make errors during query writing can more informatively be categorized, based on the learning taxonomy and human errors, into three distinct categories and the related source, as shown in Table 7.19.

<table>
<tr><th colspan="2">Error category</th><th>Source of error</th><th colspan="2">Description</th></tr>
<tr><td colspan="2">Know-What
NOT Remembering</td><td rowspan="2">Know ledge-based</td><td colspan="2">Lack of SQL knowledge
E.g.: Ignorance of the SQL syntax</td></tr>
<tr><td colspan="2" rowspan="2">Know-How
NOT Understanding</td><td colspan="2">Grasping the meaning of concepts.</td></tr>
<tr><td>Rule-based</td><td colspan="2">Understanding how to use the understood concept</td></tr>
<tr><td rowspan="3">Poor Problem Solving</td><td>Where-to-use</td><td rowspan="3">Rule-based & Skill-based</td><td>Problem formulation</td><td>Analysis : the ability to divide the given problem into sub-problems</td></tr>
<tr><td>When-to-use</td><td>Problem translation</td><td>Synthesise: the ability to relate above sub-problem to the correct SQL concepts</td></tr>
<tr><td>How-to-use</td><td>Application</td><td>Writing: making use of learned material in new or unfamiliar contexts)</td></tr>
<tr><td colspan="2">External factors</td><td colspan="2">Previous knowledge (learner schemata)</td><td>Other factors: Interference from other rules, heuristic and bias</td></tr>
</table>

Table 7.19: Source of Error

I. **Know-What:** such as ignorance of the SQL syntax similar to the syntax errors defined by [4, 83, 340]. This error can be attributed to a deficiency in learner knowledge. Reisner [57] defined different category of this error such as infusion errors.

II. **Know-How:** here students try to develop a rule based on their previous SQL understanding. Then they try to apply such a rule to solve the given question. The source of this type of error is incomplete acquisition of SQL concepts. This error is similar to overgeneralization

errors [57] ; a syntactically correct query that produces the wrong answer. This relates to learners' comprehension and use of the applicable SQL concepts. This is the semantic error mentioned by [340] or error of substance by both [5] and [33].

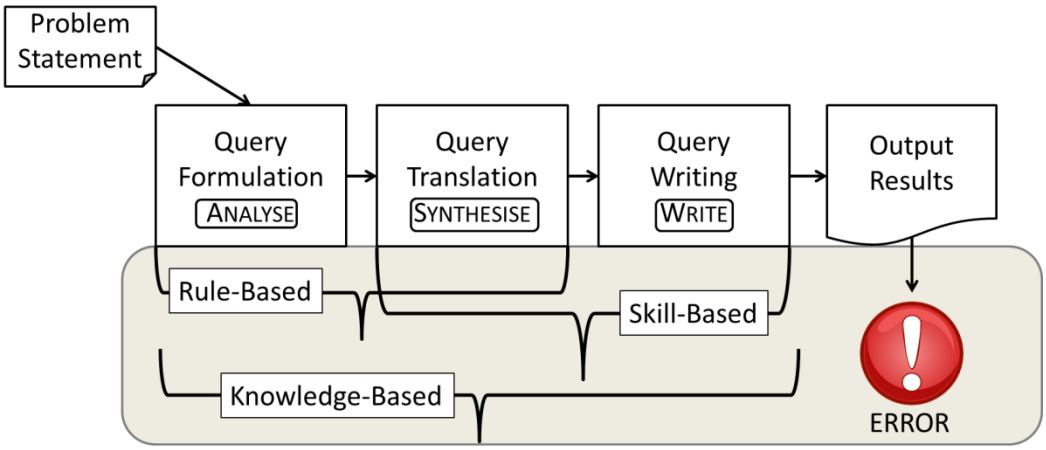


Figure 7.17: SQL Errors from SQL Nature and Cognition Perspective

III. **Poor Problem solving:** related to the learner's lack of problem-solving skills. A similar category of error was defined by Lahtinen [316]. We have categorized this as rule- or skill-based [337]. Rule-based errors are related to decisions. So, the activity carried out by learners in choosing a strategy is rule-based. Skill-based errors are related to actions – so, having chosen a rule, you execute it, and if you make a mistake there then this is a skill-based error The following are the three sources of error related to problem solving:

(1) **Analysis of the problem:** is the deficiency in understanding how to subdivide into sub-problems. Lahtinen [316] argue that students attempt to write a query before understanding the given required context. Examples of this error are: unused or wrong data (table or columns, rows) or the use of unnecessary data which slows the query down. The other form of this error is the decision of which rule to apply to solve the given problem. Failing to decide on the correct rule leads to this error.

(2) **Synthesis of the concept :** occurs when students fail to decide what elements of the data model are relevant, and the necessary

operations[59]. This is due to a learner lack of problem translation as a result of poor understanding of the given problem. So the decision about wrong data is ruled-based. However, the action in translation of the problem in terms of set (tables, columns and relation) is a skill-based error.

(1) **Applying concepts:** occurs when concepts are misused due to learners' lack of knowledge in knowing how to use the concepts correctly. [5] and [33] called this type of error "error of form". This error also can be related to Omission Errors [57] as a result of translation from natural language to SQL. This occurs when SQL concepts are used partially such as missing columns or expressions in SELECT list that is in the GROUP BY clause. Missing operations such as IN and Exists before Multiple-row subqueries, omitted HAVING clause, and omitted GROUP BY [4, 83, 340] using unnecessary concepts such as unnecessary DISTINCT and unnecessary join [339, 340] .Since the above errors are a list of actions then they are considered as skill-based errors.

7.6 The impact of SQL pattern in Participants' Performance at Query Writing

Each time the participants submit their query for execution, they need to evaluate the query output. Then participants either approve the output, if it meets the desired criteria, or reject it, if not. Participants' result evaluation was examined within the analysis of their attempts. The participants' query writing performance can be measured based on the cognitive activities in SQL problem solving cognitive model, including:

1- Writing: ability to use the correct and complete SQL concepts to solve the problem in the given scenario. This is related to query writing.
2- Evaluation: ability to check the correctness of the given query.

This section evaluates SQL queries that were submitted by students as their final solution to the given question. Each student was given four main questions to answer and her/his performance is measured in terms of the percentage of the final submitted answer.

7.6.1 Analysis of rubric and method

The quality of the last given attempt was measured by both the correctness and completion percentage. Completeness is related to the availability of the main elements (see Table 7.20).

Student percentage of completion of the final submitted	**0%**	**<20%**	**20-50%**	**60-90%**	**100%**
Query not given or completely wrong	✓				
The tables, columns, relations are missing or unnecessary data are used.		✓			
Concept is missing			✓		
Operation is missing				✓	
Completion					✓

Table 7.20: Completeness Rubric

The correctness was graded by using a numerical value. The scoring system is related to these of Yen & Scamell [33] and Kim [345] as shown in Table 7.21. Scoring system is used to evaluate the correctness of the query (see Table 7.20).

Kim [345]	Yen & Scamell [33]
Completely correct	No error
Spelling error	minor language error
	error of substance
	Minor data error
Operator error	error of form
Completely incorrect	Completely incorrect

Table 7.21: The Classification of the Correctness of the Final Attempt.

Measurement was determined separately by two graders. A totally correct answer gets a score of Max 100%, otherwise the score ranges from (0-100 %) as shown in Table 7.22. Example of error weighting is shown in Table 7.23.

Student percentage of correction of the final submitted	0%	<20%	20-50%	60-90%	100%
Completely incorrect	✓				
Tables and relation –data error		✓			
Any calculation, operation			✓		
Syntax error only				✓	
Completely correct					✓

Table 7.22: Query Correctness Rubric

	Common errors	Error weight
Syntax	Misspelling key word	-1
Joining	Missing the link between tables	-5
	Joining table that don't have actual relation in ERD	
	Joining tables that actual do not appear in FROM clause	
Aggregation	Group by columns: column in SELECT but not in Group by	-3
	Having statement omission or misused	-2
Sub query	comparison operators is missing	-3
	Mismatch between the row and column count and the comparison operator.	-2
	Mismatch between the data type of inner and outer queries	-3

Table 7.23: Examples of Error Weight

The next section reports the results of both groups.

7.6.2 Overview of both group performance

Table 7.24 shows the participants problem solving performance scores, the following were the two questions related to their performance at query writing:

Student percentage of completion of the final submitted	0%	<20%	20-50%	60-90%	100%
Student percentage of correction of the final submitted	0%	<20%	20-50%	60-90%	100%

Table 7.24: Participants Problem Solving Performance Scores

Data from Control Group

29 students submitted their answers for the given question. The result was assessed in term of its completion and correctness.

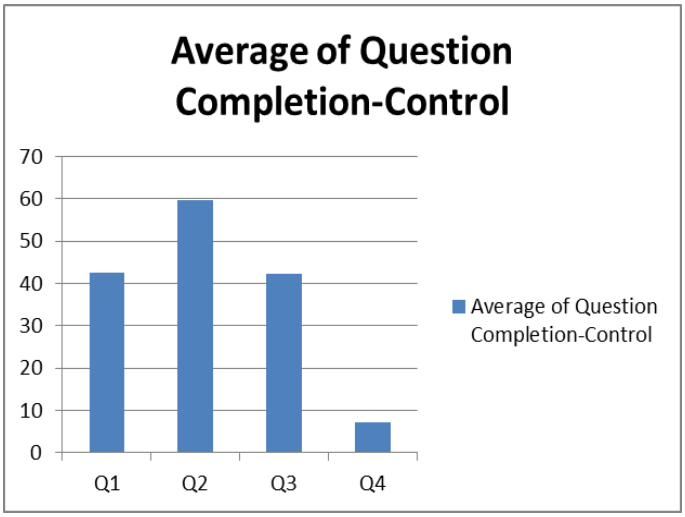


Figure 7.18: Average of Question Completion-Control

Figure 7.18 show the students average of each question in terms of how complete was the given answer.

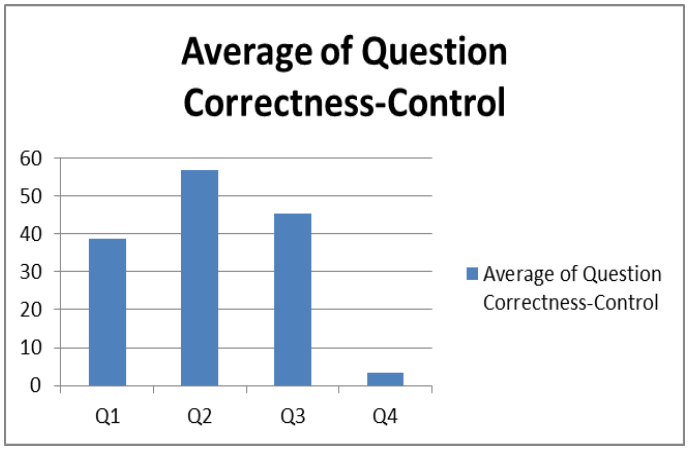


Figure 7.19: Average of Question Correctness-Control

Figure 7.19 show the students average of each question in terms of how correct was the given answer.

Data from Experiment Group

30 students submitted their answers for the given question. The result was assessed in term of its completion and correctness.

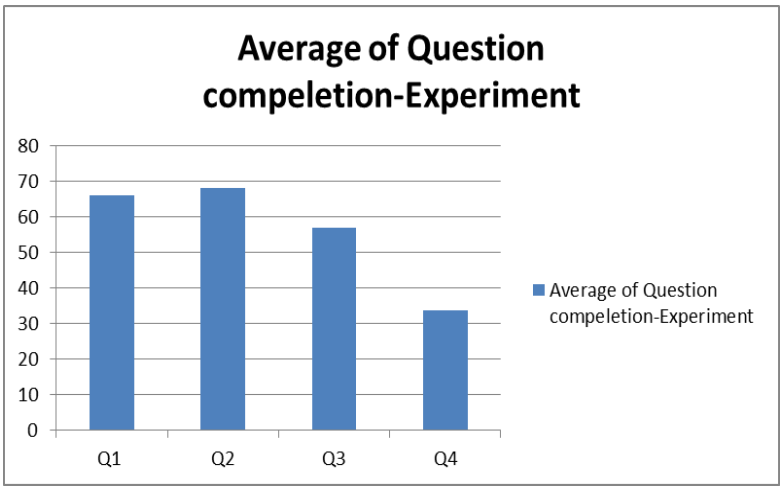


Figure 7.20: Average of Question Completion- Experiment

Figure 7.20 shows the students average of each question in terms of how complete was the given answer.

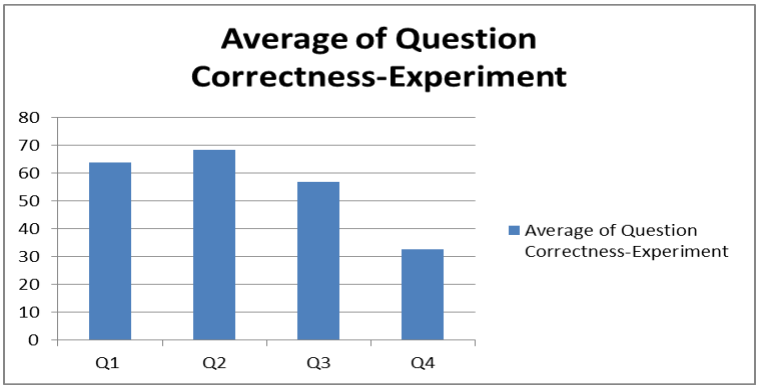


Figure 7.21: Average of Question Correctness- Experiment

Figure 7.21 shows the students average of each question in terms of how correct was the given answer. The result analysis for both groups is discussed in section 7.6. It is clear that pattern group (AVG 5.6) perform better than control group (AVG 3.69). However, both groups have similar issues to the difficulties of the given four questions. The next section reports the data collected from the third research question (see Table 7.1): **How have participants felt about the efficacy of the patterns?**

7.7 SQL Pattern Usability

Usability is measured using the three ISO metrics suggested by The International Organization for Standardization (ISO): effectiveness, efficiency and satisfaction. Nielsen [248] includes learnability with the efficiency (see Figure 7.22).

ISO 9241-11	Nielsen (1998)
Efficiency	Efficiency, Learnability
Effectiveness	Memorability Errors/Safety
Satisfaction	Satisfaction

Figure 7.22: Usability Measurements

Effectiveness and efficiency of SQL patterns were measured, in this study, using dependent variables such as Participants performance and satisfaction. Participant performance reflects the ability of the novices to solve the given problem effectively as was discussed in the previous sections. Satisfaction, on

the other hand, was measured based on respondents' feedback on a set of 5-option Likert scales in a questionnaire. The next section reports the results of participants' satisfaction.

Participants' Satisfaction

To evaluate user satisfaction, it is necessary to describe the mean scores for each of these questions.

Factors	Yes	Not sure	No
Enjoy learning from SQL patterns	100%	0	0
Recommended SQL patterns to other	100%	0	0
SQL pattern are easy to remember	94.7%	0	5.3%
Would like to use patterns in other related courses	84.2%	15.8%	0

Table 7.25: The Mean Score of Satisfaction

The results (see Table 7.25) show the participants enjoyed using SQL patterns, which point toward a positive attitude to learning SQL.

Moreover, the fact that novices were prepared to market the idea of the patterns, to recommend it to their friends and use them in other related courses was an indication of their overall satisfaction and willingness to accept and adopt the SQL patterns (see Table 7.26). Therefore, the satisfaction hypothesis of SQL patterns was supported.

Overall result			
Variables, scale:[Strong Disagree (1) to Strong Agree (5)]	Mean	Median	mode
SQL patterns helped me understanding SQL knowledge	4.05	4	4
SQL patterns structure was easy to understand	3.89	4	4
SQL patterns helped me understanding the given question better	4.11	4	4
SQL patterns helped me solving the question faster	3.79	4	4
Language of the patterns easy to understand	4.32	4	4

SQL patterns helped me in solving the post test	4.21	4	4
I was able to match the question with the related patterns using checklist	3.89	4	4
SQL patterns helped me feeling confident about the solution that I gave	3.95	4	4
Different parts of the patterns helped me to understand and solve the question in an efficient way	4.16	4	4
SQL patterns helped me to ease the way to perform the given task	4.32	4	4

Table 7.26: Mean, Median & Mode of Variables

Learnability and Time

The participants were asked about how long it took to understand the SQL patterns. Almost 50% of them felt it would be less than 5 hours (see Figure 7.23). This suggests that students would take some time to become familiar with their structure and content.

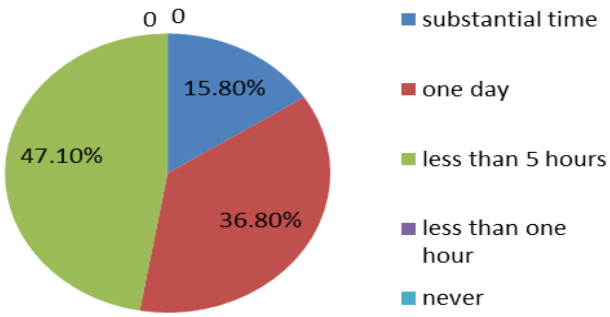


Figure 7.23: Time Required Understanding Patterns

The question that needs further exploration is how easy that is to remember and reuse. It is important to find, if the patterns helped the students to structure their knowledge. That is, whether knowledge transfers was more successful. Further research is required to examine the effects of patterns in long term. This can be done by using the pattern in more than one course semester or by testing knowledge retention after a period of time. The next section examines the usability of different part of the patterns.

The Effectiveness of Pattern Content in Problem Solving

The other part of the investigation is about the content of the patterns and if one part helps students more than others in solving the task.

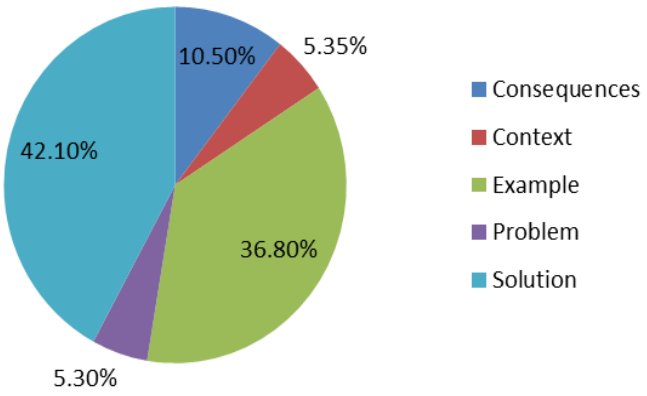


Figure 7.24: Which Part was More Helpful in problem solving?

It's clear that the majority of students depended heavily on the solution and example parts of the patterns (see Figure 7.24). While few students looked at the context of the problem, which confirm the previous finding about how novice solve SQL problem. The next part examines the content of SQL patterns and components believed useful by participants.

The Effectiveness of Pattern Content in supporting SQL Acquisition

This part explores the effectiveness of different parts in supporting SQL acquisition. The result shows that both "solution" and "Example" were the most useful, followed by the" context" (see Figure 7.25).

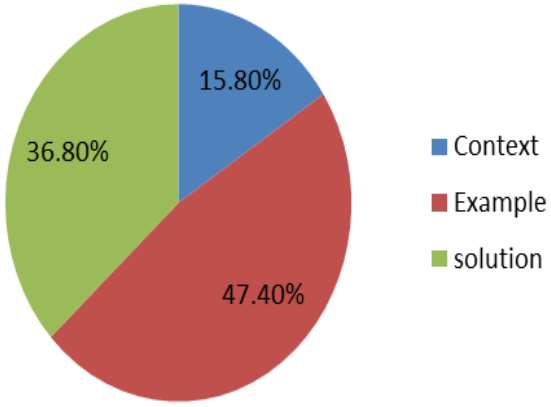


Figure 7.25: Which Part was More Helpful in SQL Acquisition?

7.8 Results Analysis

Here the experiment results for both participants' knowledge and skills is reported. All the participants' attempts were analysed to identify their strategies, assess their problem analysis and synthesis skills, track their errors and measure their solution quality (validity and completeness). The following subsections analyse these aspects in more detail.

7.8.1 The Impact of SQL pattern on knowledge Acquisition

The results of the pre and post-test can be analysed from different dimensions such as SQL knowledge misconception and the type of question participants could not solve. In general, both tests results show the identified SQL misconception were similar to conceptual difficulties that were identified in previous studies [3, 10]as shown in Table 7.27. It showed that the majority of students in both conditions had limited knowledge of the examined concepts of SQL including: Joining, Nested query, Grouping and Restricting Grouping.

SQL concepts	Pre-test	Post-test	Query writing test
Self-join	Question 2 &8	Question 2 & 8	Question 1
Aggregation	Question 3, 5	Question 3, 5, 10	Question 2
Subquery	Question 6, 7	Question 6, 7	Question 3
Exists	Question 4, 9	Question 4, 9	Question 4

Table 7.27: SQL concepts examined in the experiment

Generally, as data presented in section 7.3 the results indicated that both groups had a deeper understanding of the understanding concepts required to solve questions 7 and 8. However, a number of students in both groups did not make substantial progress following the treatment in question 3 (see Figure 7.26).

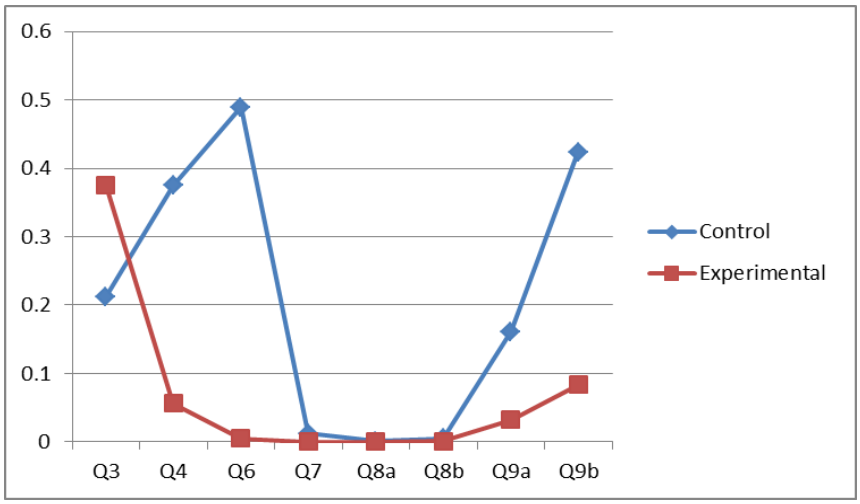


Figure 7.26: Paired Samples Test (Sig.2 tailed),

Participants in the experimental group demonstrated significant progress in their understanding ($p < 0.05$) in question 4 and question 9 compared to control group. Query writing test result as shown in Figure 7.27 that both groups performed better in question 2 compared to question 4. There is no significant difference in their performance for question 1 and question 3.

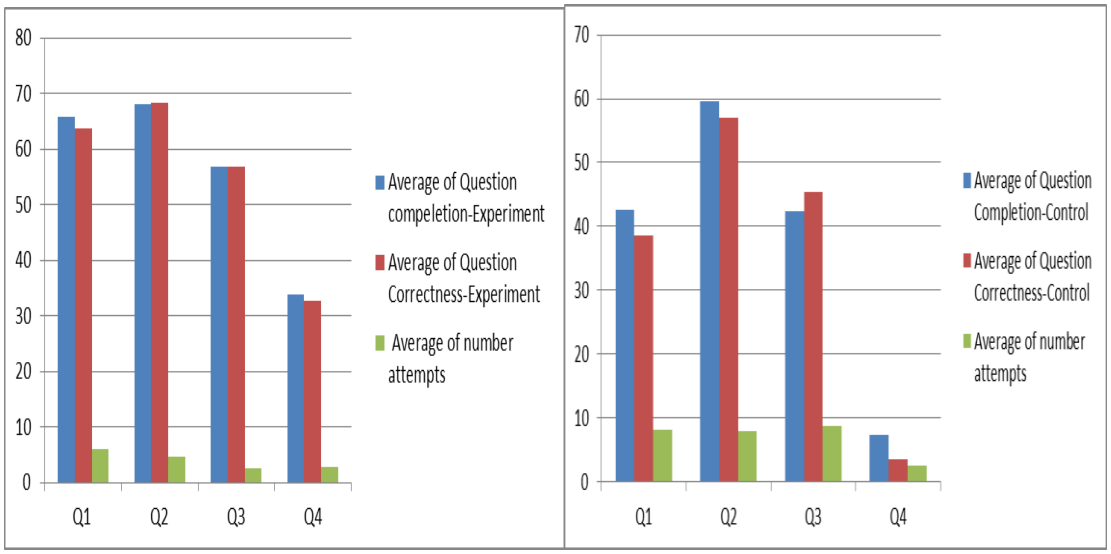


Figure 7.27: Both Groups Performance at Query Writing

These results could be interpreted as the level of concepts difficulties or other reasons. The next section discusses SQL misconceptions in greater depth.

7.8.1.1 The Misconception of the Examined Concepts

Of the twelve questions in the pre-test and post-test, questions three and five examined participants' knowledge of grouping and restricting grouping. The majority of students displayed limited knowledge of SQL aggregation in question 3. Confirming previous findings [3, 10] and the research findings summarized in section 5.5.3 where participants identified aggregation as a difficult concept. However, in the query writing test both groups performed significantly better in question 2 than in solving the rest of the questions.

Nested query and Predicate operators (exists) were not identified as misconceptions [3, 10]; but rather identified as one of the most difficult concepts in SQL competency [346]. Nested query concepts were evaluated in the pre- and post-tests in both questions 6 and 7. Both groups performed significantly good in question 7 with ($p < .05$). In the query writing test many students did not use the subquery correctly.

The majority of participants in the control group failed to answer question 4 correctly in the query writing test and did not show any signification improvement in question 9. It is possible to say that SQL patterns helped students to perform better in understanding and using Predicate operators (exists).

Joining tables was identified as a misconception [3, 10]. In the tests of this experiment, the researcher focused only on one type of join (self-join) in questions (2, 8a and 8b in pre- and post-tests) and question 1 in query writing test. In the pre-test question 8a was identified as the most difficult question by all participants. Both groups show a significant improvement in question 8. However, many students in the control groups failed to answer question 1 correctly.

These findings confirm the results from the literature, which serve to validate and confirm the most common SQL misconceptions. It has to be acknowledged that the type of the question might have an impact, while most students were aware of what a subquery is (question 10 in the pre-test), the majority couldn't answer question 7 that asked "when" to use subquery. This confirms students' ability to master "what" but not "when" (questions 8a and 8b). Few students were able to identify the need for a self-join but almost all failed to answer question 1 in the query writing test which required critical thinking about using the self-join. This supports the theory that students do not have problems with the concepts but rather lack the skills to apply the required knowledge [49].

7.8.2 The impact of SQL pattern in problem solving strategy

Problem solving is a fundamental skill that is needed by many learners in different fields of science [47-51]. The focus on problem solving has ranged from proposing that curricula be designed to encourage learning through problem solving [34, 272] to characterizing the problem solving processes and performance of learners [347, 348] focusing on the importance of exploring students' behaviours during problem solving. The participants' behaviour and

performance during problem analysis and synthesis are discussed here. Data were gathered from PSS or other submitted sheet as was discussed in section 7.4.

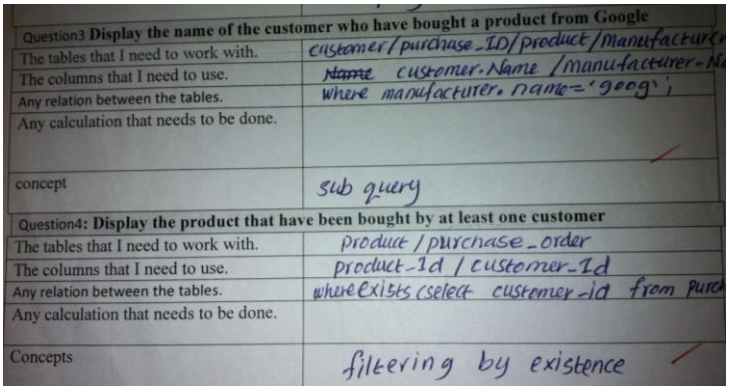
The value of the PSS sheet was evaluated by asking participants for their opinion. The results indicated that the majority of students who used it agreed that the PSS sheet helped them to organize their thoughts and encouraged them to start looking at the questions before thinking about the SQL. Some students preferred to start with a blank sheet of paper to brainstorm their ideas and write the query and claimed that the structured form of the PSS provided guidance but that an empty paper was better since it gave them more space to write, draw and erase. In the control group 14 participants and in the experimental group 9 (see Table 7.6) considered using the process of problem analysis a time-consuming process and claimed that it was better to commence writing the SQL commands directly. This confirms arguments made by [269] that findings that learners often attempt to code a solution before planning or identifying the relevant concepts.

The mean score for each question of the analysis and synthesis task for the control group was 1.2 and for the patterns group it was 1.4. At a more detailed level, the submitted sheet was used to assess participants' analysis process and understanding of the questions in terms of:

1- Analysis: To what extent were participants able to analyze the problem correctly? In addition to show if SQL patterns helped them to connect to their previous knowledge (schemata) better than the ordinary teaching materials.
2- To what extent SQL patterns helped them to produce correct decision or rules to solve the problem. Did participants in experiment group select the correct pattern for each question?

The analysis of the submitted sheets reveals that many of the participants in both groups failed to produce 100% correct rules at this stage. Comments such

as: “what I am asked to do”, “what is data I having, tables, column, and relations” reveal some confusion. As a result, many students in both groups used unnecessary data in their queries, such as unnecessary tables or columns or joining spurious tables or solved the question in a suboptimal fashion.



Question3 **Display the name of the customer who have bought a product from Google**	
The tables that I need to work with.	customer / purchase_ID / product / manufacturer
The columns that I need to use.	~~Name~~ customer.Name / manufacturer.N
Any relation between the tables.	where manufacturer.name='goog';
Any calculation that needs to be done.	
concept	sub query
Question4: Display the product that have been bought by at least one customer	
The tables that I need to work with.	product / purchase_order
The columns that I need to use.	product_id / customer_id
Any relation between the tables.	where exists (select customer_id from Purch
Any calculation that needs to be done.	
Concepts	filtering by existence

Figure 7.28: Example of Error in Query Formulation

Looking at Figure 7.28, students in both questions failed to specify the correct, tables, columns and relation between the tables. For example for question four

```
select  app.PRODUCT.DESCRIPTION
from   app.PRODUCT
where  exists(
select * from app.PURCHASE_ORDER, app.CUSTOMER
where app.PURCHASE_ORDER.PRODUCT_ID=app.PRODUCT.PRODUCT_ID
and app.PURCHASE_ORDER.CUSTOMER_ID= app.CUSTOMER.CUSTOMER_ID)
```

Figure 7.29: Sample Answer for Question 4

Many students in both group failed to specify the correct data and the related SQL concepts see Figure 7.29. Moreover, many students in experiment group were able to specify the correct pattern for question 2 and 3 as shown in Figure 7.30.

Question	Q1	Q2	Q3	Q4
Number of student who specify the correct pattern	11	16	16	6

Figure 7.30: Number of Student who Specify the Correct Pattern

These data are discussed and interpreted with other finding in section 7.9.

7.8.3 The Impact of SQL Patterns in Intermediate Attempts

A decision tree is appropriate to analyse students' errors because decision trees turn raw data (queries) into knowledge and hence awareness of issues. It enables researchers to deploy the knowledge in a simple, but powerful set of human readable rules [332] such as SQL misconceptions or the lack of problem solving skills. Contrastive Analysis was used to identify the possible sources of errors [349] while Error Analysis allowed us to carry out a statistical description of the identified errors as well.

The robustness of the findings was validated with different grading schemes (Chan and Wei, 1996). In addition other decisions were taken. For example, if the error is related to a wrong decision taken during problem analysis, which led to a totally wrong output, then this is classified as a "rule with extremely highly rated" (60% or more). If the rule is related to basic principles such as those related to the data (tables, and relation) then it is considered high (50%). However, errors such as those related to aggregation and grouping misconceptions are considered medium errors (40%-20%) and other syntax or minor errors weigh 20% or less. The reasons behind such classifications might be related to the importance of understanding such concepts. The next subsections report the analysis of students attempts from two sides: general learners' strategies in solving the different questions in the given task and their actual attempts (errors' analysis).

7.8.3.1 Result Analysis of Learner general strategy

From the analysed data, there was some difference in student strategies in solving tasks. The analysis revealed that the general behaviour of the two groups, in terms of problem solving, was different. The majority of participants in the control group move from one question to another without completion and then revisit un-completed questions later. In the experimental group, there

were fewer cases where the participants revisited previous questions. In addition, students in both groups, sometimes, try to apply different concepts apparently without any clear reasons. For example, they repeatedly execute the same query without changing it.

Table 7.28 presents an example of participants' strategies in one of the groups. (The example is selected randomly). The student is trying to answer question 1 that asks "**list the names of all customers who have placed more than one order".** As shown in Appendix J the related facts and knowledge are:

1- Two queries need to be done to search for customer order
2- Customers with two or more orders need to be displayed
3- Two copies of the table PURCHASE_ORDER need to be used
4- Each table will have different aliases name
5- The two copies of the table will be join
6- The join of two copies of the same table is called a self-join
7- The order-numbers need to be different.

Trial 1	select Customer_ID from PURCHASE_ORDER group by CUSTOMER_ID having ORDER_NUM >1	11:37 AM
Trial 2	select Customer_ID from PURCHASE_ORDER group by ORDER_NUM having Customer_ID >1	11:37 AM
Trial 3	select A.Customer_ID from PURCHASE_ORDER A,PURCHASE_ORDER B WHERE A.ORDER_NUM = B.ORDER_NUM and B.ORDER_NUM>1	11:42 AM
Trial 4	select sum(purchase_cost) from product,purcahase_order where product.PRODUCT_ID = purchase_order.PRODUCT_ID group by product.	11:48 AM
Trial 5	select sum(purchase_cost) from product,PURCHASE_ORDER where product.PRODUCT_ID = PURCHASE_ORDER.PRODUCT_ID group by product.cu	11:49 AM
Trial 6	select sum(purchase_cost) from product,PURCHASE_ORDER where product.PRODUCT_ID = PURCHASE_ORDER.PRODUCT_ID group by PURCHASE_O ORDER	11:50 AM

Trial 7	select A.Customer_ID from PURCHASE_ORDER A,PURCHASE_ORDER B WHERE A.ORDER_NUM = B.ORDER_NUM group by A.Customer_ID	11:52 AM
Trial 8	select A.Customer_ID from PURCHASE_ORDER A,PURCHASE_ORDER B WHERE A.ORDER_NUM = B.ORDER_NUM group by A.Customer_ID	11:53 AM
Trial 9	select A.Customer_ID from PURCHASE_ORDER A,PURCHASE_ORDER B WHERE A.ORDER_NUM = B.ORDER_NUM group by B.ORDER_NUM	11:54 AM

Table 7.28: Example of Control Group Problem Solving Strategy

Studying the above example, trial 1, one could say that the student has a sense of logic of the question as they try to group the Customer-ID and then check the value or ORDER-NUM. Looking at trial 2, the student changed his decision in terms of the grouped data and the filtered data. Grouping customers by order number is a bad decision and reveals the misconception in the relational database. Trial 3 show a good progress in the decision of using Self-join.

Trial 4, 5 and 6 the student tried to solve a different question. He spent eight minutes on this question and then went back to the previous question.

Trial 7 is the same as trial 3 with some change of replacing the condition in WHERE to Having. In trial 8 the student is executes the same query in trial 7, in trial 9 the same error in trial 2 is reintroduced!

It is conceivable that participants in both groups were incentivized to spend a considerable amount of effort searching for the required solution within the patterns changing their decisions randomly. However, there was a noticeable difference among participants in both groups where the decision correction tends to be better within the experimental group than the control group. One can argue that participants in the control group solved the problem without the incentive to explore options while the patterns group benefited from the structured knowledge within the patterns.

As this study is not aiming to provide detailed discussion on the different decisions or actions taken by participants, it is possible for future study to investigate the individual decisions and their applicability to the problem. In addition, the factors that influence the decisions taken by students could be explored.

7.8.3.2 The Result Analysis of Participants' Errors

Error analysis has been used to examine three aspects: how the concepts or a language are acquired, the learner's strategy and procedures in employing the target language [333]. It involves the statistical information of the error frequency and the constructive analysis that relates the error to different cognitive or behavioural reasons.

Statistical description of the identified errors

The methods used to count errors affect the statistics of errors and hence the results [349]. This section compares the frequency of each category. Examples of the distribution of errors for the examined SQL concepts are illustrated in Table 7.29.

	Common errors	**Count Control group**	**Count Experiment group**
Joining	Missing the link between tables	20	25
	Joining table that don't have actual relation in ERD	5	2
	Joining tables that actual do not appear in FROM clause	3	0
Aggregation	Group by columns: column in SELECT but not in Group by	7	12
	Having statement	3	1
Sub query	comparison operators is missing	9	5
	Mismatch between the row and column count and the comparison operator.	11	15
	Mismatch between the data type of inner and outer queries	5	0
Syntax errors	Any syntax error	35	39

Table 7.29: Error Frequency

This allowed identifying the most frequently occurring errors, which supports causative identification. For example, many researchers identified "Joining" as a misconception but no one gave a description of the kinds of errors students actually made when joining different tables. Here, it was possible to identify the source of the error (i.e. what are the difficulties in joining for example) and the level of difficulty, by observing error frequency. The error frequency of both groups (Table 7.29) shows that both attempted a similar number of errors. Thus, there is no impact of SQL patterns on errors frequency. There are many factors that influence the frequency of errors and it was not possible to make an accurate judgment since the numbers of students who solve each question are different.

Constructive Error analysis

Contrastive Analysis was used to identify the possible sources of errors [349] and to understand:

1. The innate nature of the learners' attitudes and skills in solving the question.
2. The effects of the nature of SQL language and SQL-specific cognitive tasks involved in the problem-solving process.
3. The effects of the learning concepts. Research was looking for evidence of SQL misconceptions (not just random guessing or a general one but rather clear, detailed misconceptions analysis.)

The following are two examples that were selected randomly to illustrate the constructive analysis process. The two examples show the answer given to question 1 in the task (see Table 7.30).

Q1: List the ID of all customers who have more than one order.	
Answer 1: SELECT CUSTOMER_ID FROM CUSTOMER, PURCHASE_ORDER WHERE PURCHASE_ORDER>=1	

Answer 2: SELECT CUSTOMER_ID FROM CUSTOMER C, PURCHASE_ORDER PO, PRODUCT P WHERE C.CUSTOMER_ID= PO. CUSTOMER_ID AND PO.PRODUCT-ID= P. PRODUCTID	

Table 7.30: Example of the Most Common Errors

The analysis is divided into three main points as was mentioned above:

First: To understand the innate nature of the learners' attitudes and skills: They reveal a lack of understanding of the context of the question and of how to extract the data from the provided data mode. For example in question answer 1: more than 58% of the participants wrote this query solution as a first attempt to solve the question. It is clear that participants experienced difficulties understanding the context. Hence, they failed in identifying the data and knowledge needed to solve the question (query translation).

Another common error is shown in answer 2 which uses unnecessary data in the query. In fact, many students unnecessarily included the "Customer" table in their solution. It was noticeable that many of students in both groups gave Answer 1 in their first attempt to solve that question. That could be related to their surface experience in solving similar questions. This confirms that students do not spend enough time understanding the given problem. Even so, some were able to give a reasonable answer to the question.

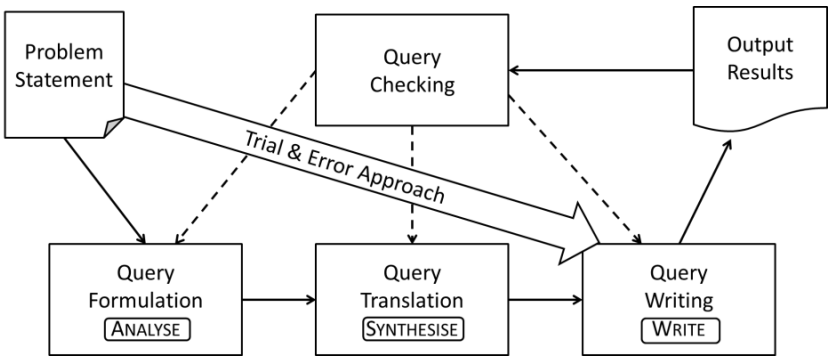


Figure 7.31: Problem Solving Process

This seems to confirm that participants lack essential problem solving skills. The time they spent understanding the given problem was clearly insufficient to analyse and plan. More time was spent in identifying the syntax and semantic errors and assessing the correctness of the generated results. In the absence of effective problem-solving strategies students deploy a hit and miss trial and error tactic [270].

Second: to understand effects of the nature of SQL language and SQL-specific cognitive tasks involved in the problem-solving process. In Answer 1 students missed the linked between the two tables. This shows a lack of understanding of the underlying set theory. To answer the question, students did not think in term of what are the sets of data that I need to work with. They tried to examine the content of PURCHASE_ORDER and not the actual frequency. This could be analysed from the nature of SQL. The declarative nature of SQL is considered one of the main causes of such difficulties learners' experience through many times they tried to solve different questions. It requires learners to think in sets rather than step-wise [350] without providing a process for achieving results compared to procedural languages.

Third: To understand effects of the learning concepts. The results presented in section 7.8.1 showed difficulties in using self-join.

The next section reports on the analysis of the completeness and the correctness of the final attempt submitted by participants in both groups.

7.8.4 The impact of SQL patterns in Query writing

The purpose of the analysis that is related to the data collected from query writing task is as follows:

- Correctness and completion of the final submitted query
- the time used to complete the task, data gathered from the tool
- error analysis

Query correctness and completeness

Comparing the number of attempts, correctness and completion of both groups shows the impact of SQL patterns.

	Total average attempts	Average of question completion	Average of question correction
Control group	6.79	37.92	36.11
Experiment group	4.06	56.15	55.41

Table 7.31: Overall Participants' Performance

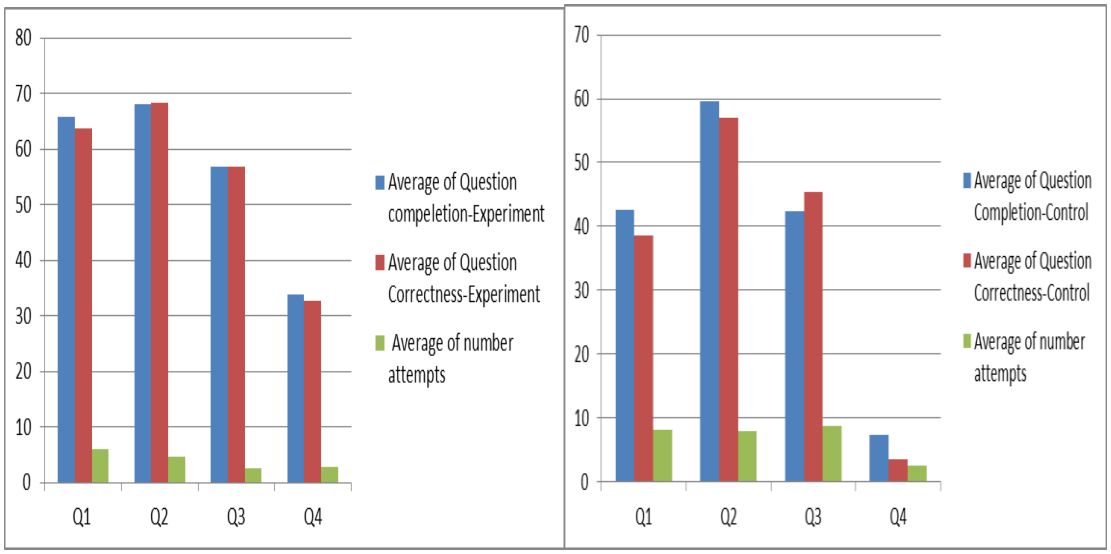


Figure 7.32: Both Groups Performance at Query Writing

The collected data (see Table 7.31 and Figure 7.32) shows that with SQL patterns there are fewer attempts and better results in term of the query correctness and completion.

Generally, both groups have deficiencies in the evolution task. The lack of query evaluation skill might be attributed to many reasons, such as:

- Course design does not explicitly teach students to evaluate their results, and they are not taught strategies to help them to evaluate their results.
- Students under pressure do not spend time checking their answers. However, Goldberg [340] found that the number of errors students made in written examinations were evident as in homework despite of the extra time and the resources they had during the homework task.
- It could be argued that at the time of evaluation, students do not want to spend any more effort and just submit their solution. This could be related to the nature of cognitive effort; i.e. when students are actively involved in difficult cognitive reasoning, it seems they lose effort or interest in the problem [351]. If true it's a matter of insufficient motivation and not trying hard enough.

This finding will need to consider how the effort was distributed among the tasks in the experiment, and how to keep students motivated and self-controlled. So it could be possible to maintain the flow of the task.

Time Distribution

The participant's time distribution for each question was different in terms of the type of the question (see Figure 7.33 and Figure 7.34). Generally, participants in both conditions spent less time on planning their solutions and more time correcting errors.

AVG.C	23.69231	24.38462	26.76923	4.923077
AVG.P	16.30769	21.46154	24.38462	19.76923

Figure 7.33 : Time Allocation per Question

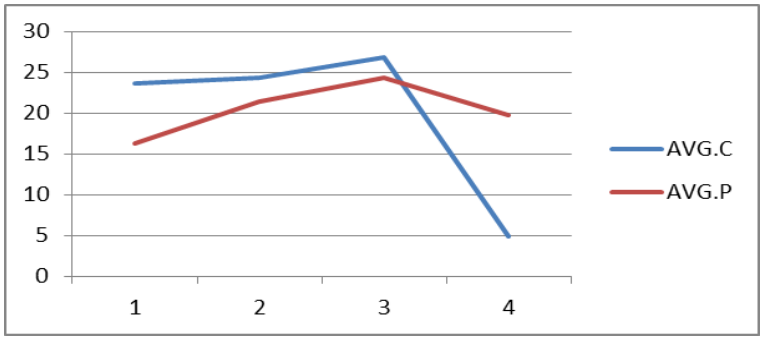


Figure 7.34: Time Distribution Pattern

Control groups spent more time on the first, second and third questions compared to the patterns group. They spent less time on question four. Few students tried to solve the question. This indicates that the patterns did not help them to solve that particular question more quickly. Clearly this is a matter for further investigation.

7.9 Discussion and Recommendation

The time during this research was spent on mainly the following activities:

- studying the issues in learning and teaching SQL from different perspectives,
- observing learners,
- discussing the highlighted issues with researcher and educators from a diversity of options,
- designing and developing the instructional materials, and
- testing the designed materials

All these are interrelated with each other. A discussion of one of them without a consideration of the others will not be complete. Therefore, all of these are brought together.

In the conducted experiment, different tasks were employed such as memorization, query reading, query comprehension, problem solving and query

writing. Within these tasks, the following operations were included in the problem solving and query writing task: projection, selection, join, self-join, repeated relation, group, IN-subquery and exist-subquery.

The experiment group received five patterns: Natural join, Self-Join Pattern, Grouping Result Pattern, Filtering by Existence Pattern and Dynamic Filtering Pattern to help them learning the examined SQL concepts. The discussion of the impact of SQL patterns focuses on five main points: SQL acquisition, participants' problem solving skills, query correctness and completion, error analysis and finally the usability of the patterns.

7.9.1 SQL Acquisition

Chapter 6 discussed the ways that patterns help in learning the intended concepts. Figure 7.35 shows the development of learners' schemata through the progress of different learning tasks. Here the first two levels are discussed.

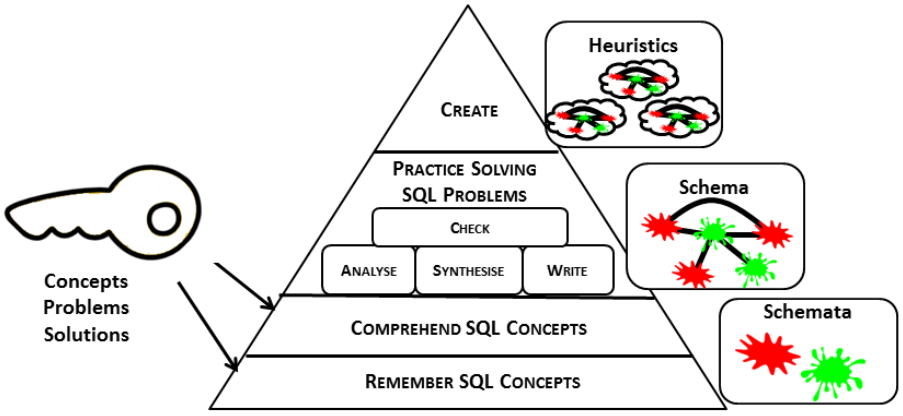


Figure 7.35: The Effect of Patterns in SQL Acquisition

Many of students in both groups were able to answer "what" question but many failed in showing their understanding of the related concepts. It is possible to say that although not all students in the experiment group were able to solve all the questions in the post test, the resulting analysis revealed that patterns had a positive impact on SQL acquisition. This could be related to the misconception of

the tested concepts. However, the results indicate that SQL patterns helped student to understand concepts better. For example, participants in the experiment group showed a significant improvement in their understanding in question 4 and question 9a compared to control group. These two questions used to examine participants' comprehension knowledge (Exist operation).

In fact, previous research as [3, 10, 284, 346] report that no empirical evidence existed to confirm the difficulties of these concepts. But, this study shows that most participants had difficulties in understanding how to apply the Exists operator, how to distinguish between Exists and IN operators and why to use Exists operator. Most of the participants did not know why and when to apply subquery. Joining of tables was one of the basic concepts that students should understand, but many students struggled to apply correctly. Therefore, it is possible to identify these concepts as threshold concepts in SQL learning and to argue that SQL pattern had a positive impact. The section discussed the first two cognitive levels in the learning taxonomy. The next section discusses the third cognitive task (problem solving) as shown in Figure 7.31.

7.9.2 Problem solving skills

This study helped us to think about how SQL should be taught. It is clear that students need to learn and master the knowledge before proceeding to problem solving. They also clearly need support during the problem solving process: formulation, translation, and application.

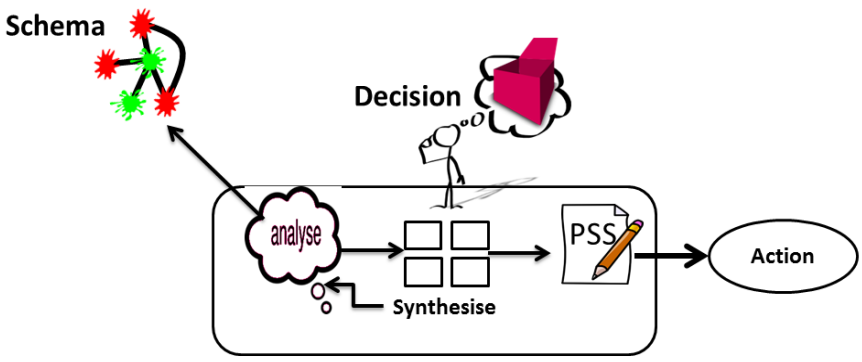


Figure 7.36: Participant Analysis Skill

In some cases, students were able to understand the given problem but failed in carrying out the execution thereof. The second research goal was to evaluate the effects of using SQL patterns in participant's cognitive tasks to solve the problems. The results show that many participants failed to analyse the problems in the task correctly. This could be interpreted to their level of schemata, as shown in Figure 7.36.

As was discussed in section 6.3.3, observation shows that the experts applied an implicit pattern matching approach to their assessment of the problem. They clearly tried to match a number of learned heuristics to the problem before settling on the best approach. However, students in this study appeared not to be able to apply heuristics. One can only assume that experts had internalized a number of abstract heuristics which they tried to match to the given problem before settling on a "best-fit" approach. Although, this research tried to embed such knowledge in the given patterns, some participants were not able to apply them all. The inevitable consequence is that novice schemata are not formed. This novice tendency is confirmed in the literature. Edwards points out that this trial and error approach does not lead to deep learning [352].

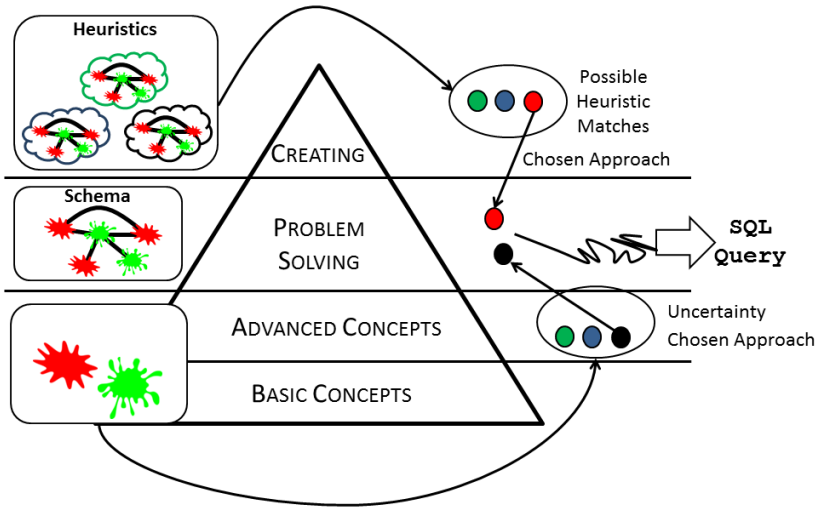


Figure 7.37 : The Role of Schemata in Problem Solving in Expert and Novice

Figure 7.37 shows the difference between experts and novices schemata. It could be possible to argue that since novices schemata are not completely

formed, they tend to judge under uncertainty [351]. To make the best use of patterns is to use them over a long period of time. In addition, the research is suggesting teaching problem solving in an opposite way to examine the learner schemata.

Recommendation: teaching problem solving from evaluation to formulation

Throughout this research, much discussion about teaching problem solving in SQL courses took place. Many issues were raised related to novices' skills in solving the given problem. Here, a new approach is proposed that is based on the idea of teaching it oppositely, Inspired by Ardens' idea of W*hatever you think, think the opposite* [353]. How that is done is highlighted in Figure 7.38.

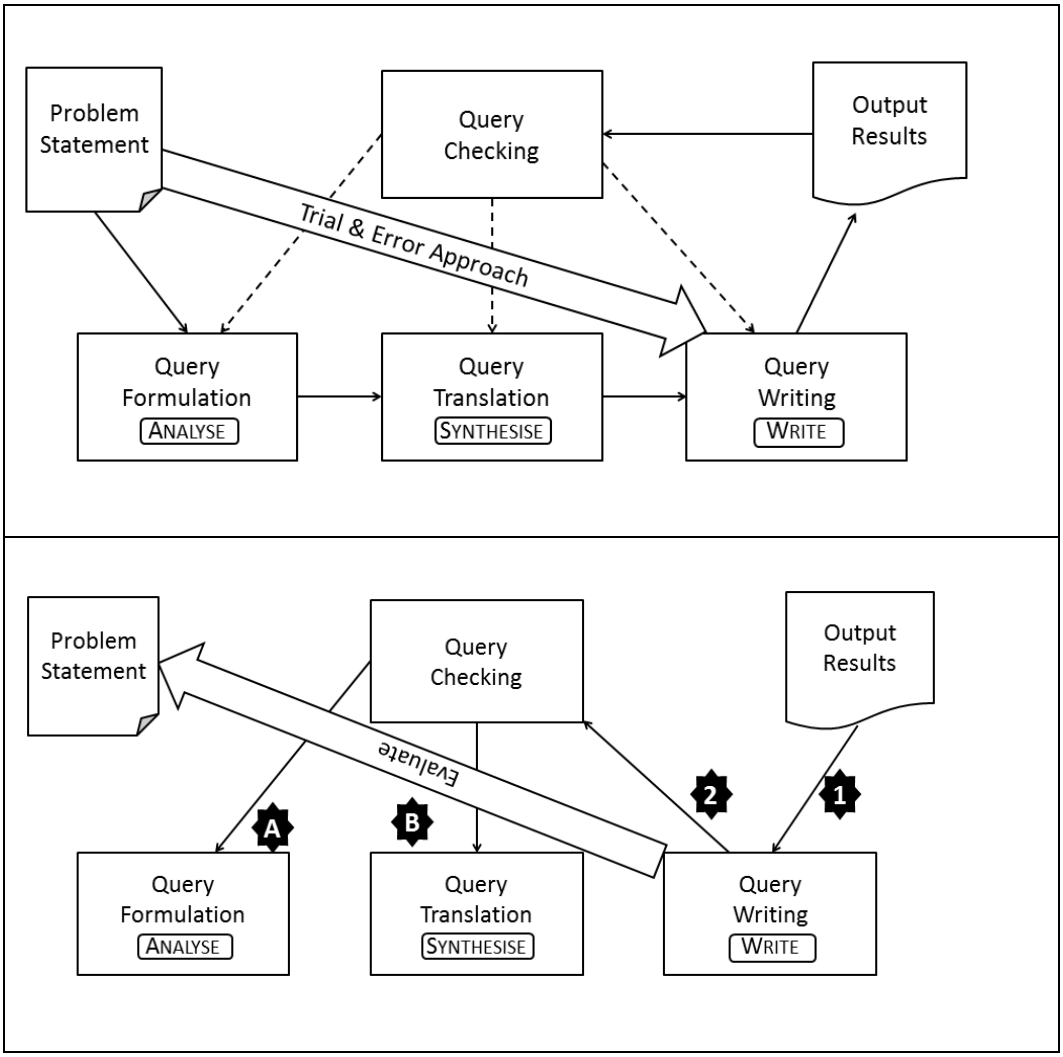


Figure 7.38: problem solving model(top), teaching problem solving oppsitly (bottom)

Here, two possible ways are demonstrated in terms of teaching problem solving oppositely:

- Show results first: give students a certain set of data and the related SQL patterns. Then ask them to write the related query. This will reduce the mental effort of thinking and guessing about which data to use and to some extent which concepts of SQL to apply.
- Evaluation first: instead of asking students to solve a problem starting by analyzing, synthesizing, writing and evaluating, ask them to evaluate a given query, providing them with the related SQL patterns. This task can vary in its complexity. For example:
    - Evaluate its syntax and semantic
    - Evaluate the correctness of used data
    - Evaluate the query in terms of the problem context
- Formulate and analyze: give them the query and ask them to highlight the table, columns, and SQL concepts that were applied to solve the question.
- Finally, ask students to modify the given query to solve a different problem.

What is the benefit of doing this? The reasons of recommending this opposite strategy in teaching problem solving is the “law of least effort” [351] which shows that people avoid speeding up their mental work during frequent switching of tasks (trial and error). Here, one can argue that asking people to judge if the certain things are correct is mentally easier than developing such things.

The next objective of this is to analyse errors and to determine whether SQL patterns are useful as an effective instructional design. A similar approach was used by Chiang [341] in second language research. The next discussion is around SQL, learners and SQL patterns as instruction materials.

7.9.3 Error analysis

Studies on trial and error in second language acquisition [341, 343] were used to support the employed approach to analyse SQL errors that make the cross-cutting factors within the proposed model of SQL learning as a base for the error analysis. This research suggested an explanation of the high frequency error sources. The aim of this is to lead to a general understanding of what makes SQL so hard to learn. Participants' errors could be discussed from three perspectives: learner strategy and the nature of the learning process, nature of SQL itself and the type of instruction.

The Learning Process and Learners' Strategy

Decision Trees were employed to examine the way in which SQL knowledge was applied. From the detailed analysis of the submitted attempts, along with the time spent, generally the majority of participants did not spend adequate time understanding the given question and identifying the context of the given problem during the problem analysis. More time was spent on identifying syntactical and semantic errors and assessing the correctness of the generated results with an average of minutes per question as recorded by SQLPB tool. As a result, many students did not succeed in translating the problem correctly, particularly in their initial attempts.

All the decision trees were analysed in terms of learner strategy or behaviour during the process of solving the given four tasks. The results revealed a common strategy framework as shown in Figure 7.39.

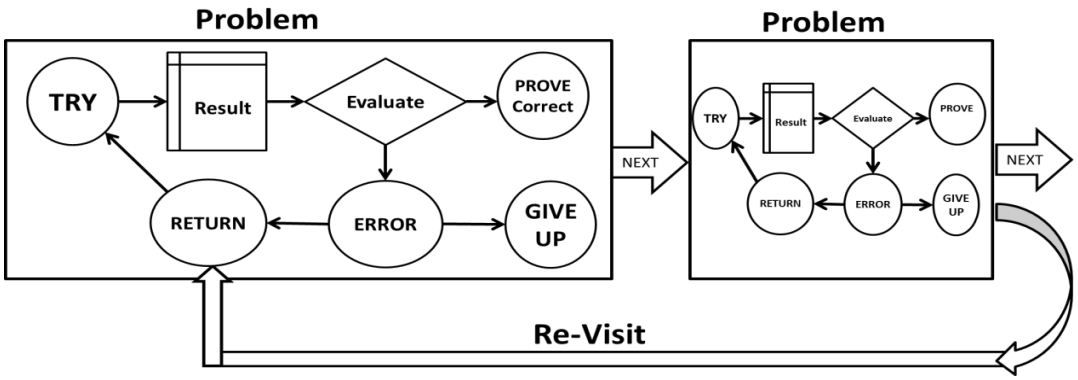


Figure 7.39: Trial and Error Strategy

Students often moved from one question to another without completion, and then revisited unsolved questions after attempting another problem. It was not yet understood what led students to abandon a problem. Attempts records will not deliver these insights. This calls for a future research on learner's behaviour during problem solving, specifically into their judgment under limited knowledge.

Within the discussion of participant's strategy in solving the query, a few things need to be linked. While most of the participants in the experimental group spent a considerable amount of time and effort searching for the required solution within the patterns, many from the control group preferred to solve the problem without a specific goal.

Cross-checking (or query evaluation), on the other hand, is one of the missing elements in participants' problem solving strategies. From the analysis of the submitted attempts, it was clear that the majority of participants did not evaluate their results correctly to determine whether the output matched the problem statements. That was obvious from looking at errors in students' attempts, compared to their problem translation. Some of the participants asked the tutor if the query results were correct, but no one asked if they wrote the query correctly. From the performance of query writing of both groups, it's possible to conclude that SQL patterns helped some participants to evaluate their attempts.

To evaluate this conjecture, it was vital to study in details the attempts generated during problem solving to demonstrate the role of heuristics in judgment and the effect of the employed instructional materials.

The Nature of Error

Students may incorrectly perceive a query problem as being easy [7]. Consequently, they neglect thinking about the semantics of the problem [350]. Conversely, students lack problem solving expertise. Thus, it was found that many of errors were rule-based and/or skill-based.

Skill-based errors are those that students make in carrying out a particular course of action identification of error based on Ogden's categorization: problem analysis or formulation, problem synthesis or translation and query writing [59]. Examples are shown in Table 7.32.

Students were asked to find customer with more than one order		
Analysis Rule-based	**SELECT** customer_id **FROM** customer, purchase_order WHERE purchase_order >=1	This reports the actual value rather than order frequency. The role developed to solve the query was incorrect.
Synthesis Rule- or skill-based	**SELECT** customer_id **FROM** purchase_order a1, purchase_order b1, product Where a1.customer_id=product_id	Here, the decision or role developed is correct which is using self-join. However, the set of data (table, columns) is incorrect. For example, no need for table product and there is data type error in WHERE clause.
Application Skill-based	**SELECT** customer_id **FROM** purchase_order po1, po2 **WHERE** po1_id =po2._id and po1.order-num <> po2.order-num	Self-join concept used incorrectly. Learners realize that they need two copies of the same table but don't know to translate this concept into correct SQL command.

Table 7.32: Example of Errors

Analysis error is due to the lack in understanding the given problem and thus inability to develop the correct rule to solve it. Synthesis errors arise due to students' inability to translate the problem in terms of relational sets [350]. Application errors, on the other hand, occur as a result of students' inability to understand how SQL commands operate and how they should be used in conjunction with each other. One could argue that the ordering of SQL statements (SELECT…FROM…WHERE) is not a natural way of expressing a query, as it specifies the constraints first, then enumerates the tables to be used, and then specifies how they ought to be linked. Many observed errors were related to a clear lack of understanding of different segments of SQL operatives.

SQL nature and SQL specific-cognition

SQL does not provide a process for achieving results compared to procedural languages. The declarative nature of SQL is considered one of the main causes of the difficulties learners' experience. It requires learners to think in sets rather than step-wise [350]. Moreover, the rigid demands of SQL syntax compared to the inexact and loose nature of natural or algorithmic language results in many students being unable to write correct SQL [4, 57]. SQL uses a linear syntax that is written in normal left-to-right, top-to-bottom format [4].

The aim was to determine whether SQL features and limitations have any effect on SQL errors. It was found that SQL features did indeed cause difficulties. For example, Figure 7.40 illustrates one query with two errors. The first error is "Know-How" or rule-based: the participant attempted to gather data from two unrelated tables (table *product_code* and *purchase_order*). This shows a lack of understanding of the underlying set theory.

```
SELECT customer_id, sum(purchase_cost)
FROM product_code join by purchase_order
WHERE purchase_cost>=1000;
```

Figure 7.40: Two Error: Rule-Based & Syntax

The second error is syntax: the use of the word "join by" which might have arisen as a result of the natural language translation to SQL. This errors source is categorized under external factors.

```
SELECT sum(PURCHASE_COST)
FROM CUSTOMER, PRODUCT
GROUP BY (CUSTOMER_ID);
```

Figure 7.41: Multiple errors

Syntax errors can be considered minor since they can easily be detected on a simple search of authoritative sources. Other errors occur because of faulty SQL-specific cognition. Another example, Figure 7.41 shows a query with two errors. The first being that the column mentioned in the "SELECT" phrase is ambiguous. The second, more serious error, is that the query is missing the phrase to link the two tables. This kind of omission error [57] might be related to two categories of error.

1. Rule-based errors that occur due to lack of understanding of set theory.
2. Application or skill-based errors that occur during the transition from synthesis to write SQL query; i.e. transition from inexact and loose nature of natural or algorithmic language to the rigid demands of SQL syntax.

Here, the error was classified as rule-based. It is clear that some learners do not understand the JOINING concept; dozens of this error type were observed at the first writing attempt. Other errors such as skill-based errors that result from a lack of problem solving skills were highlighted as well.

Recommendation to "Teach Syntax Naturally"

Inspired by Ardens' idea of *Whatever you think, think the opposite* [353] here a new way of teaching SQL might need to be tried which is opposite of the current way of teaching. Under the above two examples, shown in Figure 7.40 and 7.41, the following is suggested: teach SQL syntax naturally.

As it was said before, SQL does not provide a step-by-step achieving result compared to procedural languages. It needs learners to think in sets rather than step-wise [350]. Moreover, the rigid demands of SQL syntax compared to the inexact and loose nature of natural or algorithmic language domino effect in many students being unable to write correct SQL [4, 57].

What about teaching SQL syntax in an opposite direction? Could it look more natural? For example, by saying "since these two objects have a relation, get me the following elements from them". Figure 7.42 shows an example of SQL syntax that present naturally and that can help students to avoid one of the most occurring mistakes which is JOINING.

WHERE "these table have such relation"
FROM "List their name"
SELECT "specify the column required"

Figure 7.42: SQL syntax

This question needs some investigation and empirical study. But, theoretically, it is possible to argue that:

- Highlighting the joining condition or the linking between the tables prevent students from forgetting it.
- Mentally helping them to understand that unless such relation exists, you are not supposed to retrieve columns from two or more different tables.

Instructional materials

The following are the differences between the two groups in terms of the type of errors found in their attempts. It is possible to argue that these errors occurred as a result of the type of instructional material used during the query writing task.

- Know-what errors: different syntax errors students made in both groups. It is not possible to say that patterns helped them to avoid some syntax error particularly in their first attempt.
- Know-how errors: the prevalence of wrong rule application applied by control group produce was more than in patterns groups. Especially those rules that were related to question one and four in the given task. Here, it is possible to say that the applied checklist helped participants in the pattern group to produce the correct rules.
- Analysis errors: many students fail in analyzing the problems in the given task in their first attempt. However, some of the participants were better in evaluating their initial attempts and adjusting their approach towards achieving the problem goal although there were no significant differences between both groups. Here, it is possible to say that those who make use of the "context" and checklist were better in analyzing the problem.
- Synthesis errors: these errors depend heavily on the previous two errors. For example, when the produced rules are incorrect or problems analyzed wrongly, then, as a consequence, the translation of the problem will be incorrect. Here, no such difference manifests in either groups.
- Application error: it was not possible to judge that one group attempt more application error than others. Both made mistakes that were classified as application errors.

Future research would be to find a way for errors to deliver more informative and helpful feedback. SQL error messages are particularly obscure and unhelpful and it is often difficult to detect the effects of semantic errors from query outcomes. From the above discussion, it is possible to find and identify the impact of SQL patterns. The following discussion is related to what students said about SQL patterns.

7.9.4 SQL Patterns Usability

To start a discussion about SQL patterns usability, it is important to examine to what participants said. During the task, both groups referred to the diagram and examples more than the text. Some said: “I don’t like to read text while I am solving a problem; I prefer short examples and figures”. Examples of students’ comments about the patterns are shown in Figure 7.43. Comments are provided verbatim (English is not corrected).

“It help us to solve complex problem and understand SQL” “SQL patterns improve us by introducing good useful things about good thing about SQL” “helped us to solve problem easily” “diagrams”, “I like join between the tables” “get the plan how to answer the SQL query: what, how, why” “use every join and operation like sub query and function” “I liked the diagram helped me to understand” “SQL patterns provide with sample queries that I try to write query similar of them”, “give clear picture about how to join between more than one table”

Figure 7.43: Participant Comments

It is clear that the impact of pattern content and structure on students’ knowledge and skills was positive. Students like to learn using patterns. They found that some pattern parts were more useful than others.

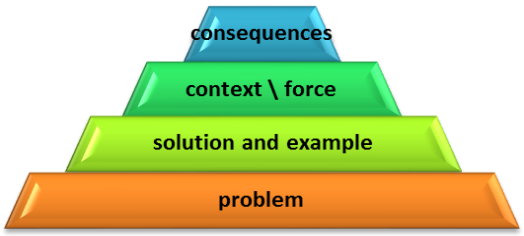


Figure 7.44: Pattern Content in the Learning Taxonomy

In fact many students did not value the knowledge embedded in the force or consequences of the patterns but they focused on the “How” section during problem solving. To understand this, it was suggested to place the content of the pattern in learning taxonomy such as Gorman [39] (see Figure 7.44). Here, it could be argued that context, force and consequence parts of the patterns are

related more to judgment or “why” in the learning taxonomy which novices lack. In addition, such information is not intrinsically pleasurable and that novice avoids such knowledge when possible. This might be because they do not see the value of such knowledge during problem solving and providing them of “How to achieve that task” is the minimal knowledge that they look for. This supports the early finding about their behaviour when starting solving a problem: writing the code is a first steps and a large amount of time is wasted in solving errors.

One could argue that although the knowledge required to guide students was available, students still felt they could not apply it effectively. It might be due to limitations of the study itself. However, it is believed that if a student gets more time to practice using SQL patterns, she/he can solve the given problem more effectively.

The question that needs more investigation is: why did students not see the value of the “Force” and “Consequences” components? This might be related to the positioning of these two parts in the learning taxonomy. One can argue that Force and Consequences are high level of knowledge. Thus, novices might not see the value of it yet or may not be able or ready to use it as was discussed in the previous section. They may perceive its value over time. This confirms the need for more longitudinal study to follow on from this one.

SQL patterns helped novice in learning about SQL knowledge and guide them towards solving problems more effectively. This motivates us to refine these patterns and create other subcategories of patterns. The chapter is summarized next.

7.10 Summary

This study delivered insights into how SQL learning happens. It is clear that students need to learn and master the basic knowledge before proceeding to problem solving. They also need support when they practice writing SQL with well-designed instructional materials. An effective intervention should support

students during all the learning stages highlighted in the proposed SQL learning model and, most importantly, the problem-solving phases. SQL patterns were examined in terms of their efficacy in both SQL knowledge transfer and building problem solving skills. Six main elements were applied to determine its efficacy:

1 Providing the required knowledge.
2 Providing them with a strategy to lead them through the essential phases of problem solving.
3 Guiding learners to analyze and synthesize the given problem.
4 Supporting them during the search of how SQL concepts are applied through the availability of both syntax and semantic knowledge.
5 Encouraging learners to evaluate the output of the given problem.
6 Motivating learners to learn and use the knowledge in form of patterns.

The analysis of the data revealed that patterns did have a positive impact on both SQL acquisition and problem solving development.

Error analysis, on the other hand, was employed to understand the reasons behind the errors, which learners commit during problem solving. This chapter reported on an investigation into the errors novices in both groups made when they solved SQL problems. Understanding errors made it possible to analyse and categorise different types of SQL error. It also suggested explanation for the errors by using different research and approaches in error classification such as Reason and Rasmussen's models of human error, thus facilitating actions and strategies to prevent recurrence of these errors. This should improve the learning strategy, teaching methods and approaches. The study of errors may lead to a better understanding of problem solving strategies that novices deploy. As a result, the difficult facets of SQL learning can be highlighted as areas for focus. In addition, this will contribute to the refinement of teaching SQL methods and the need for SQL specific tools.

In this research, understanding for why learners (patterns groups) make mistakes during query writing provided valuable insights into the refinement of the patterns as well. The next chapter presents the conclusion of this dissertation.

Chapter 8: Conclusion

This chapter presents a summary of the contributions, detailing how the thesis statement has been proved, and suggesting possible future work.

8.1 Research Contribution

This research enhances the understanding of the problems besetting the learning of SQL as well as demonstrating the effects of SQL patterns on knowledge acquisition. The following are the main research contributions:

- ***A model of SQL learning***

The research contributes to theoretical knowledge by proposing an empirically validated SQL-specific learning model. This model depicts the SQL learning process and the cross-cutting aspects that impact learning.

- ***Set of efficacious SQL patterns***

A set of SQL patterns were designed and developed, informed by the literature on learning and by the SQL learning model. The research contributions include:

- *SQL pattern design strategy*: The research employed pattern concepts and other related research to structure SQL knowledge in the form of patterns.
- *SQL pattern organization and presentation model*: the collection of SQL pattern are organised visually based on the concept of checklists and component-level design, adapted from the field of software development.
- *SQL pattern evaluation*: the effectiveness of the SQL patterns was empirically verified. This yielded new insights into typical novice

problem-solving strategies and a taxonomy of the types of errors SQL novices make.

These contributions resulted in publications in peer-reviewed conference proceedings [18-22].

8.2 Achievement of Thesis Statement's Objectives

SQL learners encounter well-documented difficulties that impair the SQL acquisition process. The purpose of this research was to determine whether SQL patterns could play a role in improving SQL acquisition by novices. Hence the thesis statement was:

It is possible to create SQL patterns which improve SQL learning by novices.

The thesis statement was broken down into three objectives as follows, each of which was addressed in an interrelated manner in this thesis.

1- **To identify impediments that imped SQL novice learning performance.** This required an investigation into the issues related to learning SQL. These issues were addressed in chapter 2. They were extended and corroborated by surveys with teachers and students. The results were reported in chapter 5 and interpreted in the context of learning and cognitive theories and reviews of problem solving. It was determined that the following concepts: grouping, join conditions, and the differences between aggregate and scalar functions are common sources of confusion. In addition, SQL, as a non-procedural language, describes the desired result without specifying *how* it is to be obtained. Step-by-step instruction achieving the result is not required by SQL compared to other procedural languages, such as Java. This leads to difficulties when SQL is introduced to novices.

2- **To develop SQL patterns to support novices**. It aimed to identify the design of a new instructional material, building on the results of objective 1, "A model of SQL learning". The model was ideal as a launching pad for the investigation into potential SQL pattern. Pattern concepts and related

research covered in the literature in chapter 3 were employed to structure and organize the SQL patterns. The SQL pattern design and development process was addressed in chapter 6. The following patterns resulted:

- Dynamic Filtering Pattern
- Filtering by Existence Pattern
- Self-join Pattern
- Natural Join Pattern
- Grouping Result Pattern

3- **To assess the efficacy of the SQL patterns**. An experiment was carried out with novices to determine whether SQL patterns eased to the learning process. The impact of SQL patterns on SQL knowledge acquisition was examined and the efficacy of SQL patterns assessed in terms of how well it supports SQL problem solving. Objective 3 was addressed in chapter 7 which combined the models established in previous chapters and evaluated SQL patterns established by objective 2. This was done in the form of experiment which included different tasks. All the participants' attempts were analyzed to identify their acquisition of the examined concepts learning strategies, to assess their problem analysis and synthesis skills, to track their errors and to measure their solution quality (validity and completeness). The results were analyzed as follow:

- The Impact of SQL pattern on knowledge Acquisition
- The impact of SQL pattern on problem solving strategy
- The Impact of SQL Patterns in Intermediate Attempts
- The impact of SQL patterns on Query writing

The discussion of the results revealed that most participants had difficulties in understanding how to apply the Exists operator, how to distinguish between Exists and IN operators and why to use Exists operator. Most of the participants did not know why and when to apply subqueries. Joining of tables was one of the basic concepts that students should understand, but many students struggled to join correctly. Therefore, it was possible to recognize these concepts as

threshold concepts in SQL learning and to argue that SQL patterns had a positive impact on learning. In addition, students were able to understand the given problem in the problem solving task but failed in carrying out the execution thereof. The second research goal was to evaluate the effects of using SQL patterns to solve the problems. The results show that many participants failed to analyse the problems correctly. This could be related to their level of schemata, expert clearly tried to match a number of learned heuristics to the problem before settling on the best approach. However, novices in this study appeared not to be able to apply heuristics. Moreover, error analysis has been used to examine three aspects: how the concepts are acquired, the learner's strategy and procedures in employing it. It involves the statistical information of the error frequency and a constructive analysis that relates the error to different cognitive or behavioural reasons. The error frequency of both groups reveals that both made a similar number of errors. Thus, SQL patterns do not impact on errors frequency. There are many factors that influence the frequency of errors but it was not possible to isolate these since the numbers of students who solve each question are different. The error constructive analysis process shows that many students in both groups reveal a lack of understanding of the context of the question and of how to extract the data from the provided data mode.

Since the three objectives have been met it can be concluded that the thesis statement is confirmed.

8.3 Future Work

There is much potential for future work in the following areas:

- ***Teaching Practice***

The importance of the learning process for an SQL-like query language, as a consequence, could be further addressed through future research, such as: how much instruction is required in order to achieve a desirable level of competence?

What type of instruction or construction will be most effective in helping novices to master SQL?

Another direction for future study would be to compare employing SQL patterns as an instructional method with a constructive approach such as an apprenticeship. A well-designed experiment could examine both SQL acquisition and learner performance in solving queries with either SQL patterns or as part of an apprenticeship.

Such studies will assist in improving and refining the proposed SQL learning model and specifically the SQL learning taxonomy and the SQL problem solving model.

- ***Pattern Design and Development***

The management, organization, and maintenance of pattern languages require more investigation. Hence, a software tool that supports the management, maintenance and retrieval of patterns will enhance and support their utilization in education. Furthermore, many agree that the studies that have been carried out have only examined simulated teaching activities rather than actual observation of "practical scenarios". In the end, this may render different outputs. It is firmly believed that the research agenda for pattern usability and efficacy in education should focus on the following areas, namely:

- **Identification process:** investigating and improving the processes by which patterns are recognized, identified or discovered and recorded. For future research these processes should need to incorporate observation of experts working with SQL problem solving: to observe them analysing, synthesising, decision-making and evaluating. This must be conducted in actual working environment with actual business scenarios.
- **Presentation process:** This research proposed a mechanism for structuring the patterns in a visual format. Further research is required to validate this visual format and to determine its efficacy.

- **Evaluation process:** Further evaluation is necessary of the contribution that SQL patterns and pattern language can make when used in both education and industry. Embedding the patterns in actual courses will deliver valuable insights. For example:

    - **Level 1:** examine the students' ability to use the patterns in comprehension tasks.

    - **Level 2:** examine the students' ability to use patterns in formulation tasks. In this case, the task should not use the "solution" and "example" part. The aim here is to examine the context, problem, force and consequences. This test will help in examining the efficiency of these parts in each pattern and thus reformulating them if required. The purpose here is to examine the processes novices engage in during problem solving.

    - **Level 3:** examine the students' ability to use patterns in debugging or evaluating existing solutions or queries.

Finally, research is required to find a way for errors to deliver more informative and helpful feedback. SQL error messages are particularly obscure and unhelpful and it is often difficult to detect the sources of semantic errors from query outcomes.

SQL patterns have the potential to support SQL acquisition; this research can be considered as a first step in this process. I hope others will take up the challenge of furthering this work, as I ward to do myself.

List of References

1. Martínez-González, M.M. and G. Duffing, *Teaching databases in compliance with the European dimension of higher education: Best practices for better competences.* Education and Information Technologies, 2007. **12**(4): p. 211-228.

2. Connolly, T. and C. Begg, *A Constructivist-Based Approach to Teaching Database Analysis and Design.* Journal of Information Systems Education, 2006. **17**(1): p. 43-53.

3. Mitrovic, A., *Learning SQL with a computerized tutor.* SIGCSE Bull., 1998. **30**(1): p. 307-311.

4. Reisner, P., *Human Factors Studies of Database Query Languages: A Survey and Assessment.* ACM Comput. Surv., 1981. **13**(1): p. 13-31.

5. Reisner, P., R.F. Boyce, and D.D. Chamberlin, *Human factors evaluation of two data base query languages: square and sequel*, in *Proceedings of the May 19-22, 1975, national computer conference and exposition*1975, ACM: Anaheim, California. p. 447-452.

6. Thomas, J.C. and J.D. Gould, *A psychological study of query by example*, in *Proceedings of the May 19-22, 1975, national computer conference and exposition*1975, ACM: Anaheim, California. p. 439-445.

7. Shneiderman, B., *Improving the human factors aspect of database interactions.* ACM Trans. Database Syst., 1978. **3**(4): p. 417-439.

8. Shneiderman, B., *Software psychology: Human factors in computer and information systems (Winthrop computer systems series).* 1980.

9. Suh, K.S. and A.M. Jenkins, *A Comparison of Linear Keyword and Restricted Natural Language Data Base Interfaces for Novice Users.* Information Systems Research, 1992. **3**(3): p. 252-272.

10. Kearns, R., S. Shead, and A. Fekete, *A teaching system for SQL*, in *Proceedings of the 2nd Australasian conference on Computer science education*1996, ACM: The Univ. of Melbourne, Australia. p. 224-231.

11. Raadt, M.d., S. Dekeyser, and T.Y. Lee, *Do students SQLify? improving learning outcomes with peer review and enhanced computer assisted assessment of querying skills*, in *Proceedings of the 6th Baltic Sea conference on Computing education research: Koli Calling 2006*2006, ACM: Uppsala, Sweden. p. 101-108.

12. Sadiq, S., et al., *SQLator: an online SQL learning workbench*. SIGCSE Bull., 2004. **36**(3): p. 223-227.

13. Schlager, M.S. and W.C. Ogden, *A cognitive model of database querying: a tool for novice instruction*. SIGCHI Bull., 1986. **17**(4): p. 107-113.

14. Davis, H., *Individual Houses in Groups: The Pattern Language in a Teaching Studio*. JAE, 1983: p. 14-19.

15. Cline, M.P., *The pros and cons of adopting and applying design patterns in the real world*. Commun. ACM, 1996. **39**(10): p. 47-49.

16. Astrachan, O., et al., *Design patterns: an essential component of CS curricula*. ACM SIGCSE Bulletin, 1998. **30**(1): p. 153-160.

17. Faroult, S. and P. Robson, *The Art of SQL*2006: O'Reilly Media, Inc.

18. Al-Shuaily, H. and K. Renaud, *A Model of SQL Learning* in *(submitted for review)*.

19. Al-Shuaily, H. and K. Renaud, *SQL Pattern Design and Development*, in *(submitted for review)*.

20. Al-Shuaily, H. and K. Renaud. *SQL PATTERNS A NEW APPROACH FOR TEACHING SQL*. in *8th International Workshop on the Teaching, Learning and Assessment of Databases*. 2010.

21. Al-Shuaily., H. *Analyzing the Influence of SQL Teaching and Learning Methods and Approaches*. in *10th International Workshop on the Teaching, Learning and Assessment of Databases*. 2012. UK, London Higher education Academy

22. Al-Shuaily, H. and K. Renaud, *SQL Pattern Organization and Presentation*

(*submitted for review*).

23. Silberschatz, A., H. Korth, and S. Sudarshan, *Database System Concepts*2011: New York: McGraw-Hill.

24. Elmasri, R. and S.B. Navathe, *Fundamentals of Database Systems*2004: Addison Wesley.

25. Connolly, T.M. and C. Begg, *Database Systems: A Practical Approach to Design, Implementation, and Management*2001: Addison-Wesley Longman Publishing Co., Inc. 1236.

26. Date, C.J., *An Introduction to Database Systems (8th Edition)*2004: Addison Wesley.

27. Navathe, S. and R. Elmasri, *Fundamentals of Database Systems (Sixth Edition)*2010: Addison Wesley.

28. Calero, C., M. Piattini, and F. Ruiz, *Towards a database body of knowledge: a study from Spain*. SIGMOD Rec., 2003. **32**(2): p. 48-53.

29. Robbert, M.A. and C.M. Ricardo, *Trends in the evolution of the database curriculum*. SIGCSE Bull., 2003. **35**(3): p. 139-143.

30. Eaglestone, B. and M.B. Nunes, *Pragmatics and practicalities of teaching and*

learning in the quicksand of database syllabuses. Journal of Innovations in Teaching and

Learning for Information and Computer Sciences, 2004. **3**(1).

31. Shackelford, R., et al., *Computing Curricula 2005: The Overview Report*, in *Proceedings of the 37th SIGCSE technical symposium on Computer science education*2006, ACM: Houston, Texas, USA. p. 456-457.

32. Jarke, M. and Y. Vassiliou, *A framework for choosing a database query language*. ACM Comput. Surv., 1985. **17**(3): p. 313-340.

33. Yen, M.Y.-M. and R.W. Scamell, *A human factors experimental comparison of SQL and QBE*. Software Engineering, IEEE Transactions on, 1993. **19**(4): p. 390-409.

34. Mayer, R.E., *Learning and Instruction*2008, Upper Saddle River,: NJ: Pearson Merrill Prentice Hall.

35. Morrison, G.R., et al., *Designing effective instruction*2010: Wiley.

36. Osguthorpe, R., *Conducting literature searches for instructional development projects*, in *Journal of Instructional Development*1985, Springer New York. p. 20-24.

37. Bloom, B.S., et al., *Taxonomy of educational objectives: The classification of educational goals. Handbook 1: Cognitive domain*1965, New York:: David McKay.

38. Anderson, L.W.E., et al., *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives (Complete edition)*2001, New York: Longman.

39. Gorman, M.E., *Types of Knowledge and Their Roles in Technology Transfer*, in *The Journal of Technology Transfer*2002, Springer Netherlands. p. 219-231.

40. Anderson, J., *The architecture of cognition.* Cambridge, Mass.: HalYard University Pres, 1983.

41. Cutts, Q., et al., *The abstraction transition taxonomy: developing desired learning outcomes through the lens of situated cognition*

10.1145/2361276.2361290, in *Proceedings of the ninth annual international conference on International computing education research*2012, ACM: Auckland, New Zealand. p. 63-70.

42. Fuller, U., et al., *Developing a computer science-specific learning taxonomy.* SIGCSE Bull., 2007. **39**(4): p. 152-170.

43. Johnson, C.G. and U. Fuller. *Is Bloom's taxonomy appropriate for computer science?* in *Proceedings of the 6th Baltic Sea conference on Computing education research: Koli Calling 2006*. 2006. ACM.

44. Starr, C.W., B. Manaris, and R.A.H. Stalvey. *Bloom's taxonomy revisited: specifying assessable learning objectives in computer science*. in *ACM SIGCSE Bulletin*. 2008. ACM.

45. Lahtinen, E. *A categorization of novice programmers: A cluster analysis study*. in *Proceedings of the 19th annual Workshop of the Psychology of Programming Interest Group*. 2007.

46. Bower, M., *A taxonomy of task types in computing*. SIGCSE Bull., 2008. **40**(3): p. 281-285.

47. Linn, M.C. and M.J. Clancy, *The case for case studies of programming problems*. Communications of the ACM, 1992. **35**(3): p. 121-132.

48. Bonar, J. and E. Soloway, *Preprogramming Knowledge: A major source of misconceptions in novice programmers*. Human-Computer Interaction, 1985. **1**: p. 133-161.

49. Ismail, M.N., et al., *Instructional strategy in the teaching of computer programming: A need assessment analyses*. TOJET, 2010. **9**(2): p. 125-131.

50. Gray, P.H., *A problem-solving perspective on knowledge management practices*. Decision Support Systems, 2001. **31**(1): p. 87-102.

51. Aamodt, A., *A knowledge-intensive, integrated approach to problem solving and sustained learning*. Knowledge Engineering and Image Processing Group. University of Trondheim, 1991: p. 27-85.

52. Niemierko, B., *Pomiar sprawdzajacy w dydaktyce: Teoria i zastosowania*1990: Pan. Wyd. Naukowe.

53. Karen Renaud, H.A.S., Richard Cooper. *Facilitating Efficacious Transfer of Database Knowledge and Skills*. in *7th HEA Workshop on Teaching, Learning and Assessment of Databases* 2009. Birmingham

54. Fuller, U., et al., *Developing a computer science-specific learning taxonomy*. ACM SIGCSE Bulletin, 2007. **39**(4): p. 152-170.

55. Pollock, E., P. Chandler, and J. Sweller, *Assimilating complex information*. Learning and Instruction, 2002. **12**(1): p. 61-86.

56. Kilpatrick, J., J. Swafford, and B. Findell, *Adding it up: Helping children learn mathematics*2001: National Academies Press.

57. Reisner, P., *Use of Psychological Experimentation as an Aid to Development of a Query Language*. Software Engineering, IEEE Transactions on, 1977. **SE-3**(3): p. 218-229.

58. Mannino, M.V., *Database Application Development and Design,*2001: McGraw-Hill Company,Inc.

59. Ogden, W.C., *Implications of a cognitive model of database query: comparison of a natural language, formal language and direct manipulation interface*. ACM SIGCHI Bulletin, 1986. **18**(2): p. 51-54.

60. McGill, T.J. and S.E. Volet, *A conceptual framework for analysing students' knowledge of programming*. Journal of Research on Computing in Education, 1997. **29**(3): p. 276-297.

61. Shneiderman, B. and R. Mayer, *Syntactic/semantic interactions in programmer behavior: A model and experimental results*. International Journal of Parallel Programming, 1979. **8**(3): p. 219-238.

62. Bruer, J.T., *Schools for thought: A science of learning in the classroom*1994: MIT press.

63. Merrill, M.D. *Knowledge objects and mental models*. in *Advanced Learning Technologies, 2000. IWALT 2000. Proceedings. International Workshop on*. 2000.

64. Bruner, J., *Toward a theory of instruction.*1966, New York: Newton.

65. Conklin, M. and L. Heinrichs, *In search of the right database text*. J. Comput. Small Coll., 2005. **21**(2): p. 305-312.

66. Shneiderman, B., *Improving the human factors aspect of database interactions*. ACM Transactions on Database Systems (TODS), 1978. **3**(4): p. 417-439.

67. Churcher, C., *Beginning SQL Queries: From Novice to Professional (Beginning from Novice to Professional)*2008: Apress. 240.

68. Donahoo, M.J. and G.D. Speegle, *SQL: Practical Guide for Developers*2005: Morgan Kaufmann.

69. McNamara, D.S., et al., *Are good texts always better? Interactions of text coherence, background knowledge, and levels of understanding in learning from text*. Cognition and Instruction, 1996. **14**(1): p. 1-43.

70. Anderson, J.R., L.M. Reder, and H.A. Simon, *Situated learning and education*. Educational researcher, 1996. **25**(4): p. 5-11.

71. Brown, J.S., A. Collins, and P. Duguid, *Situated cognition and the culture of learning*. Educational researcher, 1989. **18**(1): p. 32-42.

72. Brusilovsky, P., et al., *Learning SQL Programming with Interactive Tools: From Integration to Personalization*. Trans. Comput. Educ., 2010. **9**(4): p. 1-15.

73. Dietrich, S.W., E. Eckert, and K. Piscator, *WinRDBI: a Windows-based relational database educational tool*. SIGCSE Bull., 1997. **29**(1): p. 126-130.

74. Prior, J.C. and R. Lister, *The backwash effect on SQL skills grading*. SIGCSE Bull., 2004. **36**(3): p. 32-36.

75. Prior, J.C., *Online assessment of SQL query formulation skills*, in *Proceedings of the fifth Australasian conference on Computing education - Volume 20*2003, Australian Computer Society, Inc.: Adelaide, Australia. p. 247-256.

76. Kenny, C. and C. Pahl, *Automated tutoring for a database skills training environment*. SIGCSE Bull., 2005. **37**(1): p. 58-62.

77. Raadt, M.d., S. Dekeyser, and T.Y. Lee, *Do students SQLify? improving learning outcomes with peer review and enhanced computer assisted assessment of querying skills*, in *Proceedings of the 6th Baltic Sea conference on Computing education research: Koli Calling 2006*2006, ACM: Uppsala, Sweden. p. 101-108.

78. Anderson, J., *Skill acquisition: Compilation of weak-method problem solutions*. Psychological Review, 1987. **92**: p. 192-210.

79. Winograd, P. and V. Hare, *Direct instruction of reading comprehension strategies: The nature of teacher explanation*. In C. E. Weinstein, E. T. Goetz, & P. A. Alexander (Eds), Learning and study strategies: Issues in assessment, instruction and evaluation (pp121-139).1988, San Diego: Academic Press.

80. Bayman, P. and R.E. Mayer, *Using conceptual models to teach BASIC computer programming*. Journal of Educational Psychology, 1988. **80**(3): p. 291-298.

81. Welty, C. and D.W. Stemple, *Human factors comparison of a procedural and a nonprocedural query language*. ACM Trans. Database Syst., 1981. **6**(4): p. 626-649.

82. Vassiliou, Y. and M. Jarke. *Query languages—a taxonomy*. in *Proc. of the NYU symposium on user interfaces on Human factors and interactive computer systems*. 1984. Ablex Publishing Corp.

83. Welty, C., *A comparison of a procedural and a nonprocedural query language: syntactic metrics and human factors.*1979: University of Massachusetts Amherst. 379.

84. Welty, C., *Human Factors Studies of Database Query Languages: SQL as a Metric*, in *Journal of Database Management (JDM)*1990, IGI Global. p. 2-11.

85. Boyle, J.M., K.F. Bury, and R.J. Evey, *Two Studies Evaluating Learning and Use of QBE and SQL*. Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 1983. **27**(7): p. 663-667.

86. Hvorecky, J., M. Drlik, and M. Munk. *Enhancing database querying skills by choosing a more appropriate interface*. in *Education Engineering (EDUCON), 2010*. 2010.

87. Bates, M. and R.J. Bobrow, *Information retrieval using a transportable natural language interface*. SIGIR Forum, 1983. **17**(4): p. 81-86.

88. Suh, K.S. and W.C. Perkins. *The effects of a system echo in a restricted natural language database interface for novice users*. in *System Sciences, 1994. Proceedings of the Twenty-Seventh Hawaii International Conference*. 1994.

89. Small, D.W. and L.J. Weldon, *An experimental comparison of natural and structured query languages*. Human Factors: The Journal of the Human Factors and Ergonomics Society, 1983. **25**(3): p. 253-263.

90. Vassiliou, Y., et al., *Natural language for database queries: a laboratory study*. MIS Q., 1983. **7**(4): p. 47-61.

91. Turner, J., et al., *Using restricted natural language for data retrieval: a plan for field evaluation*. Information Systems Working Papers Series, Vol, 1982.

92. Chan, H.C., *A Two-stage Evaluation of User Query Performance for the Relational Model and SQL.*, in *PACIS*2007, AISeL. p. 118.

93. Chan, H.C., K.K. Wei, and K.L. Siau, *User-database interface: the effect of abstraction levels on query performance**. MIS Q., 1993. **17**(4): p. 441-464.

94. Lochovsky, F.H. and D.C. Tsichritzis, *User performance considerations in DBMS selection*, in *Proceedings of the 1977 ACM SIGMOD international conference on Management of data*1977, ACM: Toronto, Ontario, Canada. p. 128-134.

95. Leitheiser, R.L. and S.T. March, *The Influence of Database Structure Representation on Database System Learning and Use*. Journal of Management Information Systems, 1996. **12**(4): p. 187-213.

96. Lu, H., H.C. Chan, and K.K. Wei, *A survey on usage of SQL*. SIGMOD Rec., 1993. **22**(4): p. 60-65.

97. Welty, C., *Correcting user errors in SQL*. International Journal of Man-Machine Studies, 1985. **22**(4): p. 463-477.

98. Sweller, J., J.J.G. Van Merrienboer, and F.G.W.C. Paas, *Cognitive architecture and instructional design*. Educational psychology review, 1998. **10**(3): p. 251-296.

99. Gagne, R.M., et al., *Principles of instructional design*. Performance Improvement, 2005. **44**(2): p. 44-46.

100. Gagne, R.M. and L.J. Briggs, *Principles of instructional design*1974: Holt, Rinehart & Winston.

101. Briggs, L.J., *Instructional design: Principles and applications*1991: Educational Technology.

102. Smith, P.L. and T.J. Ragan, *Instructional design*1999: Wiley New York, NY.

103. Gustafson, K.L. and R.M. Branch, *What is instructional design.* Trends and issues in instructional design and technology, 2002: p. 16-25.

104. Dick, W. and L. Carey, *The systematic design of instruction (4th ed.)*1996, New York: NY: Harper Collins.

105. Ellis, R., *Instructed second language acquisition: A literature review*2005: Research Division, Ministry of Education.

106. Rackliffe, V.B., *ADVANCED STRUCTURED QUERY LANGUAGE INSTRUCTION FOR ENGINEERS OF THE OFFICE OF INFORMATION TECHNOLOGY AT BRIGHAM YOUNG UNIVERSITY*, in *Department of Instructional Psychology and Technology*2005, Brigham Young University.

107. Quilici, J.L. and R.E. Mayer, *Role of examples in how students learn to categorize statistics word problems.* Journal of Educational Psychology, 1996. **88**(1): p. 144.

108. Bloom, B.S. and L.J. Broder, *Problem-solving processes of college students.* Supplementary educational monographs, 1950.

109. Gick, M.L. and K.J. Holyoak, *Analogical problem solving.* Cognitive psychology, 1980. **12**(3): p. 306-355.

110. Aamodt, A. and E. Plaza, *Case-based reasoning: Foundational issues, methodological variations, and system approaches.* AI communications, 1994. **7**(1): p. 39-59.

111. Keane, M., *On retrieving analogues when solving problems.* The Quarterly Journal of Experimental Psychology, 1987. **39**(1): p. 29-41.

112. Alexander, C., *Notes on the synthesis of form*1964, Cambridge: MA: Harvard University Press.

113. Dearden, A. and J. Finlay, *Pattern languages in HCI: A critical review.* Human-Computer Interaction, 2006. **21**(1): p. 49-102.

114. Hennipman, E.J., E.J. Oppelaar, and G. van der Veer, *Pattern languages as tool for discount usability engineering.* Interactive Systems. Design, Specification, and Verification, 2008: p. 108-120.

115. Wania, C.E., *Examining the Impact of an Information Retrieval Pattern Language on the Design of Information Retrieval Interfaces*, 2008, Drexel University.

116. Alexander, C., et al., *A pattern language*1977, Oxford, UK: Oxford University Press.

117. Gamma, E., et al., *Design Patterns: Abstraction and Reuse of Object-Oriented Design*, in *Object-Oriented Programming SE - Lecture Notes in Computer Science*, O. Nierstrasz, Editor 1993, Springer Berlin / Heidelberg. p. 406-431-431.

118. Riehle, D. and H. Züllighoven, *Understanding and using patterns in software development*. TAPOS, 1996. **2**(1): p. 3-13.

119. Coplien, J.O. and A.W.O. Alexander, *Software patterns*. 1996.

120. Gamma, E., et al., *Design patterns: Elements of reusable object-oriented design*, 1995, Addison-Wesley Reading, MA;.

121. Schach, S.R., *Object-oriented and classical software engineering*. Vol. 6. 2005: McGraw-Hill higher education.

122. Ljubojevic, D. and D. Laurillard, *A theoretical approach to distillation of pedagogical patterns from practice to enable transfer and reuse of good teaching* in *European LAMS and learning design conference : sharing great ideas*2010: Oxford.

123. Brown, W.H., R.C. Malveau, and T.J. Mowbray, *AntiPatterns: refactoring software, architectures, and projects in crisis*. 1998.

124. Crawford, W. and J. Kaplan, *J2EE design patterns*2003: O'Reilly Media, Incorporated.

125. Alexander, C., *The timeless way of building*1979, Oxford, UK: Oxford University Press.

126. Alexander, C., et al., *The production of houses*1985, Oxford, UK: Oxford University Press.

127. Alexander, C., et al., *A new theory of urban design*1987, Oxford, UK: Oxford University Press.

128. Alexander, C., et al., *The Oregon experiment*1975, Oxford, UK: Oxford University Press.

129. Alexander, C., *The linz café/das kafe linz*1982, Oxford, UK: Oxford University Press.

130. Borchers, J., *A pattern approach to interaction design*2001, Chichester, UK: Wiley.

131. Garlan, D. and N. Delisle, *Formal specifications as reusable frameworks*. VDM'90 VDM and Z—Formal Methods in Software Development, 1990: p. 150-163.

132. Wirfs-Brock, A., et al. *Designing reusable designs(panel session): experiences designing object-oriented frameworks*. in *Proceedings of OOPSLA / ECOOP 90, Addendum: systems, languages, and applications*. 1990. New York: ACM Press.

133. Buschmann, F., K. Henney, and D.C. Schmidt, *Pattern Oriented Software Architecture: On Patterns and Pattern Languages*. Vol. 6. 2007: Wiley.

134. Coad, P., *Object-oriented patterns*. Commun. ACM, 1992. **35**(9): p. 152-159.

135. Riehle, D. and H. Züllighoven, *A pattern language for tool construction and integration based on the tools and materials metaphor*. Pattern languages of program design, 1995. **1**: p. 9-42.

136. Norman, D.A. and S.W. Draper, *User centered system design; new perspectives on human-computer interaction*1986: L. Erlbaum Associates Inc.

137. Pemberton, L. *The promise of pattern languages for interaction design*. in *Human Factors Symposium*. 2000.

138. Tidwell, J. *Common ground: a pattern language for human-computer interface design*. 1999 [cited 2012 3/12]; Available from: http://www.mit.edu/~jtidwell/common_ground.html.

139. Tidwell, J. *UI patterns and techniques*. Zdroj: www. mit. edu/~ jtidwell 2002 [cited 2012 3/12]; Available from: http://time-tripper.com/uipatterns/Introduction.

140. Tidwell, J., *Designing interfaces*2010: O'Reilly Media, Incorporated.

141. Borchers, J.O., *A pattern approach to interaction design*. Cognition, Communication and Interaction, 2008: p. 114-131.

142. Yacoub, S.M. and H.H. Ammar. *A pattern language of statecharts*. in *Proc. Fifth Annual Conf. on the Pattern Languages of Program (PLoP'98)*. 1998.

143. Van Welie, M. and G.C. Van der Veer. *Pattern languages in interaction design: Structure and organization*. in *Proceedings of interact*. 2003.

144. Borchers, J. *Teaching HCI Design Patterns: Experience From Two University Courses*. 2002. Position paper for "Patterns in Practice" workshop at CHI.

145. Sharp, H., et al., *Pedagogical patterns\—successes in teaching object technology: a workshop from OOPSLA '96*. SIGPLAN Not., 1996. **31**(12): p. 18-21.

146. Bergin, J., et al., *The Pedagogical Pattern Project*. Retrieved June, 2000. **12**: p. 2003.

147. Sharp, H., M.L. Manns, and J. Eckstein, *Evolving pedagogical patterns: The work of the pedagogical patterns project*. Computer Science Education, 2003. **13**(4): p. 315-330.

148. Haberman, B., *Pedagogical patterns: A means for communication within the CS teaching community of practice*. Computer Science Education, 2006. **16**(2): p. 87-103.

149. Muller, O., B. Haberman, and H. Averbuch. *(An almost) pedagogical pattern for pattern-based problem-solving instruction*. in *ACM SIGCSE Bulletin*. 2004. ACM.

150. Muller, O. and B. Haberman, *Supporting abstraction processes in problem solving through pattern-oriented instruction*. Computer Science Education, 2008. **18**(3): p. 187-212.

151. Diggelen, W. and M. Overdijk, *Grounded design: Design patterns as the link between theory and practice*. Computers in Human Behavior, 2009. **25**(5): p. 1056-1066.

152. Bergin, J. *Fourteen pedagogical patterns*. in *fifth European conference on pattern languages of programs, Irsee, Germany. Retrieved June*. 2000.

153. Barfield, L., et al., *Interaction design at the Utrecht School of the Arts.* ACM SIGCHI Bulletin, 1994. **26**(3): p. 49-86.

154. Seffah, A., *Learning the ropes: human-centered design skills and patterns for software engineers' education.* interactions, 2003. **10**(5): p. 36-45.

155. Meatball. *Anti Pattern,*. 2007 [cited 2009 April,]; Available from: http://c2.com/cgi/wiki?AntiPattern.

156. Koenig, A., *C++: Patterns and antipatterns.* Journal of Object Oriented Programming, 1995. **8**: p. 46-46.

157. Laplante, P.A. and C.J. Neill, *Antipatterns: Identification, Refactoring, and Management*2005: Auerbach Publications.

158. VanBiljon, J., et al., *The use of anti-patterns in human computer interaction: wise or III-advised?*, in *Proceedings of the 2004 annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries*2004, South African Institute for Computer Scientists and Information Technologists: Stellenbosch, Western Cape, South Africa. p. 176-185.

159. Kotz;, P., K. Renaud, and J.v. Biljon, *Don't do this - Pitfalls in using anti-patterns in teaching human-computer interaction principles.* Comput. Educ., 2008. **50**(3): p. 979-1008.

160. Bayle, E., et al., *Putting it all together: towards a pattern language for interaction design: A CHI 97 workshop.* SIGCHI Bull., 1998. **30**(1): p. 17-23.

161. Winn, T. and P. Calder, *Is This a Pattern?* IEEE Softw., 2002. **19**(1): p. 59-66.

162. Beck, K. and R.E. Johnson, *Patterns Generate Architectures*, in *Proceedings of the 8th European Conference on Object-Oriented Programming*1994, Springer-Verlag. p. 139-149.

163. Beck, K. and W. Cunningham, *Using Pattern Languages for Object-Oriented Programs*, in *OOPSLA-87*1987.

164. Borchers, J.O., *A Pattern Approach to Interaction Design Cognition, Communication and Interaction*, S. Gill, Editor 2008, Springer London. p. 114-131.

165. Blackwell, A.F. and S. Fincher, *PUX: patterns of user experience*. interactions, 2010. **17**(2): p. 27-31.

166. Welie, M.v. *A Pattern Library for Interaction Design*. 2008 [cited 2010; Available from: http://www.welie.com/index.php.

167. Fincher, S. *The Pattern Gallery*. 2000 2012 [cited 2013 February]; Available from: http://www.cs.kent.ac.uk/people/staff/saf/patterns/gallery.html.

168. Specker, M. and I. Wentzlaff, *Exploring usability needs by human-computer interaction patterns*. Task Models and Diagrams for User Interface Design, 2007: p. 254-260.

169. Granlund, A., D. Lafrenière, and D.A. Carr. *A pattern-supported approach to the user interface design process*. in *Proceedings of HCI International*. 2001.

170. Salingaros, N.A., *The structure of pattern languages*. Architectural Research Quarterly, 2000. **4**(2): p. 149-161.

171. Todd, E.G., E.A. Kemp, and C.H.E. Phillips, *UMM: a maturity model for UI-pattern languages*, in *Proceedings of the 12th Annual Conference of the New Zealand Chapter of the ACM Special Interest Group on Computer-Human Interaction*2011, ACM: Hamilton, New Zealand. p. 33-40.

172. Borchers, J.O., *A pattern approach to interaction design*, in *Proceedings of the 3rd conference on Designing interactive systems: processes, practices, methods, and techniques*2000, ACM: New York City, New York, United States. p. 369-378.

173. Dearden, A., et al., *Evaluating pattern languages in participatory design*, in *CHI '02 extended abstracts on Human factors in computing systems*2002, ACM: Minneapolis, Minnesota, USA. p. 664-665.

174. Dearden, A., et al. *Evaluating pattern languages in participatory design*. in *CHI'02 extended abstracts on Human factors in computing systems*. 2002 a. ACM.

175. Dearden, A.M., et al., *Using pattern languages in participatory design*, in *Proceedings of the Participatory Design Conference (PDC 2002)*, T. Binder, J. Gregory, and I. Wagner, Editors. 2002 b, CPSR: Malmö, Sweden.

176. Wania, C.E. and M.E. Atwood. *Pattern languages in the wild: exploring pattern languages in the laboratory and in the real world*. in *Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology*. 2009. ACM.

177. Wesson, J. and L. Cowley. *Designing with patterns: Possibilities and pitfalls*. in *Proceedings of the 2nd Workshop on Software and Usability Cross-Pollination: The role of Usability Patterns, INTERACT 2003*. 2003.

178. Seffah, A. *The evolution of design patterns in HCI: from pattern languages to pattern-oriented design*. in *Proceedings of the 1st International Workshop on Pattern-Driven Engineering of Interactive Computing Systems*. 2010. ACM.

179. Fincher, S. and P. Windsor. *Why patterns are not enough: some suggestions concerning an organising principle for patterns of UI design*. in *CHI'2000 Workshop on Pattern Languages for Interaction Design: Building Momentum*. 2000.

180. Gaffar, A., et al. *MOUDIL: A comprehensive framework for disseminating and sharing HCI patterns*. in *CHI'03 Workshop on Perspectives on HCI patterns: Concepts and Tools*. 2003.

181. Henninger, S. and P. Ashokkumar, *An Ontology-Based Infrastructure for Usability Design Patterns*. Proc. Semantic Web Enabled Software Engineering (SWESE), Galway, Ireland, 2005: p. 41-55.

182. Koukouletsos, K., et al., *Teaching Usability Principles with Patterns and Guidelines*, P. KotzÃ©, et al., Editors. 2009, Springer US. p. 159-174.

183. Beck, K., et al., *Industrial experience with design patterns*, in *Proceedings of the 18th international conference on Software engineering*1996, IEEE Computer Society: Berlin, Germany. p. 103-114.

184. Griffiths, R., L. Pemberton, and J. Borchers. *Usability Pattern Language: Creating a community*. in *Proceedings of the INTERACT'99 7th International Conference on Human-Computer Interaction*. 1999.

185. Erickson, T., *Towards a pattern language for interaction design*, 2000, 2000) Workplace Studies: Recovering Work Practice and Informing Systems Design. Cambridge. Cambridge University Press.

186. Jalil, M., S.A. Noah, and S. Idris, *Evaluating the effectiveness of a pattern application support tool for novices*, in *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education*2010, ACM: Bilkent, Ankara, Turkey. p. 239-243.

187. Jalil, M.A. and S.A.M. Noah. *The Difficulties of Using Design Patterns among Novices: An Exploratory Study*. in *International Conference on Computational Science and its Applications, 2007*. 2007.

188. Weir, S.B., *An investigation of significant technical factors affecting the learning and using of design patterns*, 2006, UTAH STATE UNIVERSITY. p. 67.

189. Van Welie, M., G.C. Van Der Veer, and A. Eliëns. *Patterns as tools for user interface design*. in *International Workshop on Tools for Working with Guidelines*. 2000.

190. Chung, E.S., et al., *Development and evaluation of emerging design patterns for ubiquitous computing*, in *Proceedings of the 5th conference on Designing interactive systems: processes, practices, methods, and techniques*2004, ACM: Cambridge, MA, USA. p. 233-242.

191. Saponas, T.S., et al. *The impact of pre-patterns on the design of digital home applications*. in *Proceedings of the 6th conference on Designing Interactive systems*. 2006. ACM.

192. Cowley, N. and J. Wesson, *An Experiment to Measure the Usefulness of Patterns in the Interaction Design Process Human-Computer Interaction - INTERACT 2005*, M. Costabile and F. PaternÃ², Editors. 2005, Springer Berlin / Heidelberg. p. 1142-1145.

193. Kotzé, P., et al. *Patterns, anti-patterns and guidelines-Effective aids to teaching HCI principles*. in *Proc. of The First Joint BCS/IFIP WG13. 1/ICS/EU CONVIVIO HCI Educators' Workshop (2006)*. 2006.

194. Kolfschoten, G., et al., *Cognitive learning efficiency through the use of design patterns in teaching*. Computers & Education, 2010. **54**(3): p. 652-660.

195. Bernhaupt, R., M. Winkler, and F. Pontico. *User interface patterns: A field study evaluation*. in *IADIS international conference-interfaces and human computer interaction 2009*. 2009.

196. Almstrum, V.L., et al., *Challenges to computer science education research*. SIGCSE Bull., 2005. **37**(1): p. 191-192.

197. Comer, D.E., et al., *Computing as a discipline*. Commun. ACM, 1989. **32**(1): p. 9-23.

198. Stasko, J., et al., *Models and areas for CS education research*. SIGCSE Bull., 2001. **33**(1): p. 388-389.

199. Berglund, A., et al., *Learning educational research methods through collaborative research: the PhICER initiative*, in *Proceedings of the tenth conference on Australasian computing education - Volume 78*2008, Australian Computer Society, Inc.: Wollongong, NSW, Australia. p. 35-42.

200. Kinnunen, P.i., V. Meisalo, and L. Malmi, *Have we missed something?: identifying missing types of research in computing education*, in *Proceedings of the Sixth international workshop on Computing education research*2010, ACM: Aarhus, Denmark. p. 13-22.

201. Joy, M., et al., *Categorising computer science education research*, in *Education and Information Technologies*2009, Springer Netherlands. p. 105-126.

202. Pears, A., et al., *Constructing a core literature for computing education research*. ACM SIGCSE Bulletin, 2005. **37**(4): p. 152-161.

203. Pears, A., et al., *A survey of literature on the teaching of introductory programming*. SIGCSE Bull., 2007. **39**(4): p. 204-223.

204. Almstrum, V.L., et al., *Challenges to computer science education research*. SIGCSE Bull., 2005. **37**(1): p. 191-192.

205. Perrenet, J., *Differences in beliefs and attitudes about computer science among students and faculty of the bachelor program*. SIGCSE Bull., 2009. **41**(3): p. 129-133.

206. Holz, H.J., et al., *Research methods in computing: what are they, and how should we teach them?* SIGCSE Bull., 2006. **38**(4): p. 96-114.

207. Pears, A., M. Daniels, and A. Berglund, *Describing computer science education research: an academic process view*. SIMULATION SERIES, 2002. **34**(1): p. 99-104.

208. Morse, J.M., *Principles of mixed methods and multimethod research design*. Handbook of mixed methods in social and behavioral research, 2003: p. 189-208.

209. Creswell, J.W., *Qualitative inquiry and research design: Choosing among five approaches*2012: SAGE Publications, Incorporated.

210. Johnson, R.B. and A.J. Onwuegbuzie, *Mixed methods research: A research paradigm whose time has come*. Educational researcher, 2004. **33**(7): p. 14-26.

211. Creswell, J.W. and V.L.P. Clark, *Designing and conducting mixed methods research*2007: Wiley Online Library.

212. Leech, N. and A. Onwuegbuzie, *A typology of mixed methods research designs*, in *Quality & Quantity*2009, Springer Netherlands. p. 265-275.

213. Morse, J.M., *Critical Issues in Qualitative Research Methods*1994, London: Sage Publications,.

214. Morse, J., *Approaches to Qualitative-Quantitative Methodological Triangulation*. Nursing Research, 1991. **40**(2): p. 120-123.

215. Ames, C. and J. Archer, *Achievement goals in the classroom: Students' learning strategies and motivation processes*. Journal of Educational Psychology, 1988. **80**(3): p. 260.

216. Fishbein, M. and I. Ajzen, *Belief, attitude, intention and behavior: An introduction to theory and research*1975.

217. Davies, N. and J. Savell. *Maths is like a bag of tomatoes": Student attitudes upon entry to an Early Years teaching degree*. in *Teacher Education Forum of Aotearoa New Zealand Conference, Christchurch*. 2000.

218. Grootenboer, P., *Beliefs, attitudes and feelings students learn about mathematics*. Far East Journal of Mathematical Education, 2010. **5**(1): p. 31-52.

219. Grootenboer, P. *Affective factors in learning to teach*. 2001. New Zealand Association of Research in Education conference, Christchurch, New Zealand.

220. Chinn, D., et al. *Study habits of CS1 students: what do they do outside the classroom?* in *Proceedings of the Twelfth Australasian Conference on Computing Education-Volume 103*. 2010. Australian Computer Society, Inc.

221. Rodrigo, M.M.T., et al. *Affective and behavioral predictors of novice programmer achievement*. in *ACM SIGCSE Bulletin*. 2009. ACM.

222. Date, C., *An Introduction to Database Systems, 1977*, Addison-Wesley Publishing Company, Inc.

223. Vassiliou, Y., et al., *Natural Language for Database Queries: A Laboratory Study*. MIS Quarterly, 1983. **7**(4): p. 47-61.

224. Schlager, M.S. and W.C. Ogden, *A cognitive model of database querying: a tool for novice instruction*

10.1145/22339.22357. SIGCHI Bull., 1986. **17**(4): p. 107-113.

225. Wilson, B.C. and S. Shrock. *Contributing to success in an introductory computer science course: a study of twelve factors*. in *ACM SIGCSE Bulletin*. 2001. ACM.

226. Boyle, R., J. Carter, and M. Clark, *What makes them succeed? Entry, progression and graduation in Computer Science*. Journal of Further and Higher Education, 2002. **26**(1): p. 3-18.

227. Roddan, M., *The determinants of student failure and attrition in first year computing science*. Computing Science, Glasgow University, project Summer, 2002.

228. Rountree, N., et al. *Interacting factors that predict success and failure in a CS1 course*. in *ACM SIGCSE Bulletin*. 2004. ACM.

229. Patton, M.Q., *Qualitative Evaluation and Research Methods*1990, Newbury Park: Sage Publications.

230. Miles, M. and A. Huberman, *Qualitative Data Analysis; an Expanded Source Book*1994, Thousand Oaks, CA: Sage Publications.

231. McNamara, C. *General guidelines for conducting interviews.* 2009 [cited 2010 January]; Available from: http://managementhelp.org/evaluatn/intrview.htm.

232. Chesson, R., *â€¦ Design a Questionnaire â€" A Ten-stage Strategy.* Physiotherapy, 1993. **79**(10): p. 711-713.

233. Soloway, E., *Learning to program = learning to construct mechanisms and explanations.* Commun. ACM, 1986. **29**(9): p. 850-858.

234. Bruner, J.S., *Toward a theory of instruction*. Vol. 59. 1966: Belknap Press.

235. Rackliffe, V.B., *Advanced Structured Query Language Instruction for Engineers of the Office of Information Technology at Brigham Young University (2005)*. . 2005.

236. Blakey, J.P., *Database training for novice end users : a design research approach : a thesis presented in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Information Systems at Massey University, Albany, New Zealand*, 2008, Massey University L2 - http://hdl.handle.net/10179/880.

237. Borchers, J.O., *CHI meets PLoP: an interaction patterns workshop.* SIGCHI Bull., 2000. **32**(1): p. 9-12.

238. Borchers, J.O. and J.C. Thomas, *Patterns: what's in it for HCI?*, in *CHI '01 extended abstracts on Human factors in computing systems*2001, ACM: Seattle, Washington. p. 225-226.

239. Koukouletsos, K., et al., *Teaching Usability Principles with Patterns and Guidelines Creativity and HCI: From Experience to Design in Education*, P. KotzÃ©, et al., Editors. 2009, Springer Boston. p. 159-174.

240. Grill, T. and M. Blauhut, *Design Patterns Applied in a User Interface Design (UID) Process for Safety Critical Environments (SCEs), HCI and Usability for Education and Work*, A. Holzinger, Editor 2008, Springer Berlin / Heidelberg. p. 459-474.

241. Tan, A.H. *Text mining: The state of the art and the challenges*. in *Proceedings of the PAKDD 1999 Workshop on Knowledge Disocovery from Advanced Databases*. 1999.

242. Scriven, M., *This memo covers some of the basic features of checklists and their application in evaluation, but it does not claim to exhaust their logic or methodology*. 2000.

243. Roger, S., *Software Engineering a Practitioner's Approach*. McGrow-Hill International Edition, 2005.

244. Felix, S. *Competing cognitive structures in second language acquisition*. in *European-North American Workshop on Cross-Linguistic Second Language Acquisition Research, Lake Arrowhead CA, September. Cited and discussed in Birgit Harley (1986) Age in Second Language Acquisition. Clevedon: Multilingual Matters*. 1981.

245. Lightbown, P.M., *Great expectations: Second-language acquisition research and classroom teaching*. Applied Linguistics, 1985. **6**(2): p. 173-189.

246. Krashen, S.D. and T.D. Terrell, *The natural approach: Language acquisition in the classroom*. 1983.

247. Juhn, S. and J. Naumann. *The effectiveness of data representation characteristics on user validation*. in *Proceedings of the Sixth international Conference on information Systems*. 1985. Indianapolis, Indiana.

248. Nielsen, J. and J.A.T. Hackos, *Usability engineering*. Vol. 125184069. 1993: Academic press San Diego.

249. Jarke, M., et al., *A field evaluation of natural language for data retrieval*. Software Engineering, IEEE Transactions on, 1985(1): p. 97-114.

250. Fincher, S., et al. *Predictors of success in a first programming course*. in *Proceedings of the 8th Australasian Conference on Computing Education-Volume 52*. 2006. Australian Computer Society, Inc.

251. Leeper, R. and J. Silver. *Predicting success in a first programming course*. in *ACM SIGCSE Bulletin*. 1982. ACM.

252. Bergin, S. and R. Reilly. *Programming: factors that influence success*. in *ACM SIGCSE Bulletin*. 2005. ACM.

253. Aiken Jr, L.R., *Update on attitudes and other affective variables in learning mathematics*. Review of Educational Research, 1976: p. 293-311.

254. Ma, X. and N. Kishor, *Assessing the relationship between attitude toward mathematics and achievement in mathematics: A meta-analysis*. Journal for research in mathematics education, 1997: p. 26-47.

255. Kotzé, P. and P. Purgathofer, *Designing Design Exercises—From Theory to Creativity and Real-world Use*. Creativity and HCI: From Experience to Design in Education, 2009: p. 42-59.

256. Strauss, A. and J. Corbin, *Basics of qualitative research: Procedures and techniques for developing grounded theory*, 1998, Thousand Oaks, CA: Sage.

257. Pekrun, R., *Emotions as drivers of learning and cognitive development*. New Perspectives on Affect and Learning Technologies, 2011: p. 23-39.

258. Caraway, S.D., *Factors Influencing Competency in Mathematics Among Entering Elementary Education Majors*. 1985.

259. Fennema, E. and J. Sherman, *Sex-related differences in mathematics achievement, spatial visualization and affective factors*. American educational research journal, 1977. **14**(1): p. 51-71.

260. Robins, A., J. Rountree, and N. Rountree, *Learning and teaching programming: A review and discussion*. Computer Science Education, 2003. **13**(2): p. 137-172.

261. Mohtashami, M. and J.M. Scher. *Application of Bloom's Cognitive Domain Taxonomy to Database Design*. in *Proceedings of ISECON (information systems educators conference)*. 2000.

262. Khan, I.A., R.M. Hierons, and W.P. Brinkman. *Mood independent programming*. in *Proceedings of the 14th European conference on Cognitive ergonomics: invent*. 2007. Citeseer.

263. Sengupta, A. and M. Dalkilic, *DSQL-an SQL for structured documents*. Lecture notes in computer science, 2002: p. 757-760.

264. Robbert, M.A. and C.M. Ricardo, *Trends in the evolution of the database curriculum.* ACM SIGCSE Bulletin, 2003. **35**(3): p. 139-143.

265. McMaster, K., S. Sambasivam, and S. Hadfield. *Relational Algebra and SQL: Better Together*. in *Proceedings of the Information Systems Educators Conference ISSN*. 2012.

266. Reif, F., *Teaching problem solving.* The Physics Teacher, 1981.

267. Heller, P., R. Keith, and S. Anderson, *Teaching Problem Solving Through Cooperative Grouping (Part 1): Group Versus Individual Problem Solving.* MAA NOTES, 1997: p. 159-172.

268. Baiocco, S.A. and J.N. DeWaters, *Successful College Teaching: Problem-Solving Strategies of Distinguished Professors*1998: ERIC.

269. Ramalingam, V., D. LaBelle, and S. Wiedenbeck. *Self-efficacy and mental models in learning to program*. in *ACM SIGCSE Bulletin*. 2004. ACM.

270. Lahtinen, E., K. Ala-Mutka, and H.-M. J;rvinen, *A study of the difficulties of novice programmers.* SIGCSE Bull., 2005. **37**(3): p. 14-18.

271. Jonassen, D.H., *Instructional design models for well-structured and III-structured problem-solving learning outcomes.* Educational Technology Research and Development, 1997. **45**(1): p. 65-94.

272. Schoenfeld, A.H., *Teaching problem-solving skills.* The American Mathematical Monthly, 1980. **87**(10): p. 794-805.

273. Henderson, P.B. *Anatomy of an introductory computer science course*. in *ACM SIGCSE Bulletin*. 1986. ACM.

274. Beaubouef, T., R. Lucas, and J. Howatt, *The UNLOCK system: enhancing problem solving skills in CS-1 students.* ACM SIGCSE Bulletin, 2001. **33**(2): p. 43-46.

275. Spohrer, J.C. and E. Soloway, *Novice mistakes: are the folk wisdoms correct?* Commun. ACM, 1986. **29**(7): p. 624-632.

276. Felix, S.W., *On the (in) applicability of Piagetian thought to language learning.* Studies in Second Language Acquisition, 1981. **3**(02): p. 179-192.

277. Willingham, D.T., *Why don't students like school: A cognitive scientist answers questions about how the mind works and what it means for the classroom*2009: Jossey-Bass.

278. Rumelhart, D.E., *Schemata: the building blocks of cognition. In: R.J. Spiro etal. (eds) Theoretical Issues in Reading Comprehension*1980, Hillsdale: NJ: Lawrence Erlbaum.

279. Newell, A. and H.A. Simon, *Human problem solving*. Vol. 14. 1972: Prentice-Hall Englewood Cliffs, NJ.

280. Bransford, J.D. and B.S. Stein, *The IDEAL Problem Solver. A guide for improving thinking, learning, and creativity*. A Series of Books in Psychology, New York: Freeman, 1984, 1984. **1**.

281. Wiedenbeck, S., *Novice/expert differences in programming skills*. International Journal of Man-Machine Studies, 1985. **23**(4): p. 383-390.

282. DATE, C.J., *An introduction to databasesystems (2nd ed.)*1977, Reading,Mass: Addison-Wesley.

283. Ramakrishnan, R., *Database Management Systems*, 1999, McGraw-Hill.

284. Bampton, M., *Addressing misconceptions, threshold concepts, and troublesome knowledge in GIScience education*. Teaching Geographic Information Science and Technology in Higher Education, 2011: p. 117-132.

285. Smelcer, J.B., *User errors in database query composition*. International Journal of Human-Computer Studies, 1995.

286. Borthick, A., et al., *The effects of normalization on end-user query errors: An experimental evaluation*. International Journal of Accounting Information Systems, 2001. **2**(4): p. 195-221.

287. Nelson, D., et al. *An evaluation of a diverse database teaching curriculum and the impact of research*. in *LTSN-ICS Teaching, Learning and Assessment in Databases Workshop*. 2003. Citeseer.

288. Njovu, C., *Teaching, learning and assessment on Msc Database Units at University of Greenwich*. Teaching, Learning and Assessment in Databases (TLAD), 2003: p. 89-93.

289. Robbert, M.A., et al. *The database course: What must be taught*. in *SIGCSE Bulletin Proceedings of 31st SIGCSE Technical Symposium on Computer Science Education, March*. 2000.

290. Ullman, J.D. *Improving the efficiency of database-system teaching*. in *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*. 2003. ACM.

291. Calero, C., M. Piattini, and F. Ruiz, *Towards a database body of knowledge: A study from Spain*. SIGMOD RECORD, 2003. **32**(2): p. 48-53.

292. Eaglestone, B. and M.B. Nunes, *Pragmatics and practicalities of teaching and learning in the quicksand of database syllabuses*. Journal of Innovations in Teaching and Learning for Information and Computer Sciences, 2004. **3**(1).

293. Domínguez, C. and A. Jaime, *Database design learning: A project-based approach organized through a course management system*. Computers & Education, 2010. **55**(3): p. 1312-1320.

294. Hogan, K.E. and M.E. Pressley, *Scaffolding student learning: Instructional approaches and issues*1997: Brookline Books.

295. Chang, K.E., Y.T. Sung, and S. Chen, *Learning through computer-based concept mapping with scaffolding aid*. Journal of Computer Assisted Learning, 2001. **17**(1): p. 21-33.

296. Mead, J., et al., *A cognitive approach to identifying measurable milestones for programming skill acquisition*. pedagogy, 2006. **21**: p. 38.

297. Son, J.Y. and R.L. Goldstone, *Contextualization in perspective*. Cognition and Instruction, 2009. **27**(1): p. 51-89.

298. DiSessa, A., J. Wagner, and J. Mestre, *What coordination has to say about transfer*. Transfer of learning from a modern multi-disciplinary perspective, 2005: p. 121-154.

299. Baranes, R., M. Perry, and J.W. Stigler, *Activation of real-world knowledge in the solution of word problems*. Cognition and Instruction, 1989. **6**(4): p. 287-318.

300. Nisbett, R.E. and L. Ross, *Human inference: Strategies and shortcomings of social judgment*1980: Prentice-Hall Englewood Cliffs, NJ.

301. Buck, D. and D.J. Stucki, *Design early considered harmful: graduated exposure to complexity and structure based on levels of cognitive development*. SIGCSE Bull., 2000. **32**(1): p. 75-79.

302. Ogden, W.C., *IMPLICATIONS OF A COGNITIVE MODEL OF DATABASE QUERY: COMPARISON OF A NATURAL LANGUAGE, FORMAL LANGUAGE AND DIRECT MANIPULATION INTERFACE*. SIGCHI Bull., 1986. **18**(2): p. 51-54.

303. Dalton, D.W. and D.A. Goodrum, *The effects of computer programming on problem-solving skills and attitudes.* Journal of Educational Computing Research, 1991. **7**(4): p. 483-506.

304. Lockhead, J., *An introduction to cognitive process instruction*. Cognitive Process Instruction1979, Philadephia: Franklin Institute Press.

305. Bloom, B.S. and L.J. Broder, *Problem-solving Processes of College Students: An Explanatory Investigation*1950: University of Chicago Press.

306. Reiter-Palmon, R. and J.J. Illies, *Leadership and creativity: Understanding leadership from a creative problem-solving perspective*. The Leadership Quarterly, 2004. **15**(1): p. 55-77.

307. Chase, W.G. and H.A. Simon, *Perception in chess.* Cognitive psychology, 1973. **4**(1): p. 55-81.

308. Larkin, J.H., *What kind of knowledge transfers? . In L. B. Resnick (Ed.), Knowing, learning, and instruction: Essays in honor of Robert Glaser (pp. 283-305)* 1989, Hillsdale, : NJ: Lawrence Erlbaum Associates.

309. Hollingsworth, J.E., *Teaching query writing: an informed instruction approach*. SIGCSE Bull., 2008. **40**(3): p. 351-351.

310. Wania, C.E., *Examining the Impact of an Information Retrieval Pattern Language on the Design of Information Retrieval Interfaces*, 2008, Drexel University.

311. van Welie, M., K. Mullet, and P. McInerney. *Patterns in practice: a workshop for UI designers*. in *CHI'02 extended abstracts on Human factors in computing systems*. 2002. ACM.

312. Fincher, S. and I. Utting. *Pedagogical patterns: their place in the genre*. in *ACM SIGCSE Bulletin*. 2002. ACM.

313. Tian, J., Y. Nakamori, and A.P. Wierzbicki, *Knowledge management and knowledge creation in academia: a study based on surveys in a Japanese research university*. Journal of Knowledge Management, 2009. **13**(2): p. 76-92.

314. Fincher, S. *Patterns for HCI and Cognitive Dimensions: two halves of the same story*. in *Kuljis, J., Baldwin, L., Scoble, R., Proceedings of the Fourteenth Annual Workshop of the Psychology of Programming Interest Group*. 2002.

315. Ramalingam, V., D. LaBelle, and S. Wiedenbeck, *Self-efficacy and mental models in learning to program*. SIGCSE Bull., 2004. **36**(3): p. 171-175.

316. Lahtinen, E., K. Ala-Mutka, and H.M. Järvinen. *A study of the difficulties of novice programmers*. in *ACM SIGCSE Bulletin*. 2005. ACM.

317. Dunbar, K., *How scientists think: On-line creativity and conceptual change in science*. Creative thought: An investigation of conceptual structures and processes, 1997: p. 461-493.

318. Nersessian, N.J., *How do scientists think? Capturing the dynamics of conceptual change in science*. Cognitive models of science, 1992. **15**: p. 3-44.

319. Bruner, J.S., *Schools for Thought*.1993: The MIT Press.

320. Hmelo-Silver, C.E., *Problem-Based Learning: What and How Do Students Learn?* Cindy E. Hmelo-Silver, 2004. **16**(2).

321. Anderson, J.R., et al., *Cognitive tutors (1995): Lessons learned*. . Journal of the Learning Sciences,, 1995. **4**(2): p. 167-207.

322. Wellhausen, T. and A. Fießer, *How to write a pattern? A rough guide for first-time pattern authors*, 2011.

323. Fincher, S. and J. Finlay. *Perspectives on HCI patterns: concepts and tools (introducing PLML*. in *Interfaces*. 2003. Citeseer.

324. Pea, R.D., *The Social and Technological Dimensions of Scaffolding and Related Theoretical Concepts for Learning, Education, and Human Activity*. The Journal of the Learning Sciences, 2004. **13**(3): p. 423-451 CR - Copyright © 2004 Taylor & Francis, Ltd.

325. Gawande, A., *The checklist manifesto: How to get things right*2010: Profile Books.

326. Rowlands, K.D., *Check It Out! Using checklists to support student learning.* English Journal, 2007: p. 61-66.

327. Laseau, P., *Graphic Thinking for Architects & Designers*2000: Wiley.

328. Flagg, B.N., *Formative evaluation for educational technologies*1990: L. Erlbaum Associates.

329. Rasmussen, J., *Human errors. A taxonomy for describing human malfunction in industrial installations.* Journal of occupational accidents, 1982. **4**(2): p. 311-333.

330. Rasmussen, J., *The definition of human error and a taxonomy for technical system design*1987: Toronto: John Wiley & Sons.

331. Davies, S.P., *Models and theories of programming strategy.* Int. J. Man-Mach. Stud., 1993. **39**(2): p. 237-267.

332. De Ville, B., *Decision trees for business intelligence and data mining: using SAS enterprise miner*2006: Sas Inst.

333. Corder, S.P., *The significance of learners' errors.* International review of applied linguistics, 1967. **5**(4): p. 161-170.

334. Spohrer, J.G. and E. Soloway, *Analyzing the high frequency bugs in novice programs*, in *Papers presented at the first workshop on empirical studies of programmers on Empirical studies of programmers*1986, Ablex Publishing Corp.: Washington, D.C., United States. p. 230-251.

335. Antony, S.R. and D. Batra, *CODASYS: a consulting tool for novice database designers.* ACM Sigmis Database, 2002. **33**(3): p. 54-68.

336. Fessakis, G., A. Dimitracopoulou, and V. Komis, *Improving database design teaching in secondary education: action research implementation for documentation of didactic requirements and strategies.* Computers in Human Behavior, 2005. **21**(2): p. 159-194.

337. Reason, J., *Human error*1990: Cambridge university press.

338. Rasmussen, J., *What can be learned from human error reports*. Changes in working life, 1980: p. 97-113.

339. Brass, S. and C. Goldberg, *Semantic errors in SQL queries: A quite complete list*. Journal of Systems and Software,Quality Software, 2006. **79**(5): p. 630-644.

340. Goldberg, C. *Do you know SQL? About Semantic Errors in Database Queries*. in *7th Workshop on Teaching, Learning and Assessment in Databases, Birmingham, UK, HEA*. 2009. Citeseer.

341. Chiang, T.H., *Error analysis: A study of errors made in written English by Chinese learners*. Taipei: The Crane Publishing Co., Ltd, 1981.

342. Reason, J., *Human error: models and management*. Bmj, 2000. **320**(7237): p. 768-770.

343. Bootchuy, T., *An analysis of errors in academic English writing by a group of first-year Thai graduates majoring in English*, 2008, MA thesis, Kasetsart University, Thailand.

344. Richards, J.C., *Error Analysis: Perspectives on second language acquisition*, 1974, London: Longman.

345. Kim, S.W., *High-level data language design: An investigation of human factors in database query*, 1979, Minnesota: Univ. Minnesota.

346. Renaud, K. and J. van Biljon, *Teaching SQL - Which Pedagogical Horse for This Course?*

Key Technologies for Data Management, H. Williams and L. MacKinnon, Editors. 2004, Springer Berlin / Heidelberg. p. 244-256.

347. Carlson, M.P. and I. Bloom, *The cyclic nature of problem solving: An emergent multidimensional problem-solving framework*. Educational Studies in Mathematics, 2005. **58**(1): p. 45-75.

348. Schoenfeld, A.H., *Problem solving in the United States, 1970-2008: research and theory, practice and politics*. ZDM, 2007. **39**(5): p. 537-551.

349. Huang, J., *Error analysis in English teaching: A review of studies*. Journal of Chung-San Girls' Senior High School, 2002. **2**: p. 19-34.

350. Dekeyser, S., M.d. Raadt, and T.Y. Lee, *Computer assisted assessment of SQL query skills*, in *Proceedings of the eighteenth conference on Australasian database - Volume 63*2007, Australian Computer Society, Inc.: Ballarat, Victoria, Australia. p. 53-62.

351. Kahneman, D., *Thinking, fast and slow*2011: Farrar, Straus and Giroux.

352. Edwards, S.H., *Using software testing to move students from trial-and-error to reflection-in-action.* ACM SIGCSE Bulletin, 2004. **36**(1): p. 26-30.

353. Arden, P., *Whatever You Think Think the Opposite*2006: Portfolio.

Appendix A: The Semi-structured Interview

Learning SQL - Interview Questions:

Name: Institute:

1. What degree are you currently pursuing?
2. How does writing SQL make you feel? Why?

Comfortable ( ) Slightly Comfortable ( ) Neutral ( ) Uncomfortable ( ) Slightly Uncomfortable ( )

3. How many SQL courses have you taken? (in and out of University)
4. How skilled do you think you are at SQL problem solving in general?

   Expert Advanced Novice Beginner Not Knowledge

5. How many months/years job experience do you have in working with SQL?

6. What are the most difficult concepts you found difficult to understand or apply? Why

7. Classify the following concepts as difficult or easy?

Courses concepts	Very Easy	Easy	confusing	Hard	Very Hard	I can't remember this topic
Using SELECT statement to retrieve data						
Restricting data (limit the row that retrieve by the query)						
Sorting data(sort the row that retrieve by the query)						
Using group functions to report aggregating data (AVG, SUM,MAX,MIN,COUNT)						
Grouping rows using GROUP BY or HAVING						
Courses concepts	Very Easy	Easy	confusing	Difficult	Very Difficult	I can't remember this topic
Displaying data from multiple data (self join, inner join, outer join)						
Using sub query (single row, multiple row)						
Using the set operator (union, intersect, minus)						
Using DDL statement to create and manage tables						

Appendix B: Problem Analysis and Synthesis Task

In plain English, explain how to solve this SQL problem by describing what you will do?

Find the names and the hire dates for all employees who were hired before their managers, along with their manager's name and hire dates. Sort by employee name

Note: all information stored in table : Employee.

Name	Null	Type
EMPLOYEE_ID	NOT NULL	NUMBER(6)
FIRST_NAME		VARCHAR2(20)
LAST_NAME	NOT NULL	VARCHAR2(25)
EMAIL	NOT NULL	VARCHAR2(25)
PHONE_NUMBER		VARCHAR2(20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
SALARY		NUMBER(8,2)
COMMISSION_PCT		NUMBER(2,2)
MANAGER_ID		NUMBER(6)
DEPARTMENT_ID		NUMBER(4)

8. Write SQL statement for the previous question.

12: The SQL problem that you solved was:

Very easy ( ) easy ( ) Neutral ( ) difficult ( ) very difficult ( )

Appendix C: Online questionnaire- Academic

Learning SQL – Teacher Questionnaire									
1	Name (Optional)								
2	How many years have you been teaching SQL courses?		0-3		3-6		6-12		
3	Do you believe a solid grounding in set theory helps students understand database concepts?		Yes				No		
4	Which concepts did you find most challenging to teach, or that students find difficult to understand? Why?								
5	Classify the following concepts in how easy students find it to understand or to apply Tick pleas		Very Easy	Easy	Confusing	Difficult	Very Difficult	did not teach this topic	
	A	Using SELECT statement to retrieve data							
	B	Restricting and sorting data							
	C	Using single raw function to customize output.							
	D	Using group functions to report aggregating data (group functions, group by)							
	E	Displaying data from multiple data (self join, inner join, outer join)							
	F	Using sub query (single row, multiple row)							
	G	Using the set operator (union, intersect, minus)							
	H	Using DDL statement to create and manage tables							
6	Why do you think many students find joining tables a difficult concept?								
7	When setting SQL course work what range of concepts are you trying to cover? Please tick								

A	Using SELECT statement to retrieve data	
B	Restricting and sorting data	
C	Using single raw function to customize output.	
D	Using group functions to report aggregating data (group functions, group by)	
E	Displaying data from multiple data (self join, inner join, outer join)	
F	Using sub query (single row, multiple row)	
G	Using the set operator (union,	

			intersect, minus)		
		H	Using DDL statements to create and manage tables		
	Other? Please specify:				
8	Do you think the concepts undergraduate students are taught, and the queries they are asked to solve, are sufficient for them to master the SQL skills they will need in the workplace? Comments:			Yes	No
9	Consider this statement: "The more SQL problems students are given to solve, the better their SQL skill will be."			Agree	Disagree
10	We interviewed a few masters' students who completed two courses in SQL during their master's studies. We asked them to solve the following SQL problem: *Find the names and the hire dates for all employees who were hired before their managers, along with their manager's name and hire date. Sort by employee name* None of the students could write the required SQL, although some were able to describe what needed to be done in order to solve the problem. Why do you think they couldn't write what is quite a simple query?				
	Why do you think so many students lose their SQL skills after completing their degrees?				
11	Sometimes when students are given an SQL query to write they can explain how to do it but cannot convert their thoughts into SQL. Why do you think this is?				
12	What type of assessment is more effective in learning SQL?		individual project\assignment		group project\ assignment
	Why?				

Appendix D: Learning SQL – Questionnaire

1	Name (Optional)	Degree you are currently pursuing:					
2	In how many course have you taken in SQL?	0	1	2	3		
3	How skilled do you think you are at SQL problem solving in general??	Expert	Advanced	Novice	Beginner	Not skilled	
4	Which SQL concepts did you find most challenging to apply or difficult to understand? Why?						

5		To what extent do you agree with the following statements	Strongly agree	agree	neutral	disagree	Strongly disagree
	1	I solve SQL problems by trial and error					
	2	I can read and understand SQL statements easily					
	3	In general SQL syntax is easy to learn and understand					
	4	I can only write simple SQL statements					
	5	I can solve a simple SQL problem					
	6	I do not have problems in writing a large and complex queries					
	7	I know how to join more than three tables and retrieve specific columns					
	8	I know how join a table to itself using *SELF JOIN*					
	9	It is easy for me to manipulate data using aggregate functions like *SUM, AVG,COUNT,..*					
	10	It is easy for me to query using aggregation by means of the *Group by* function					
	11	SQL is easy to use compare with other programming languages					

6	Why do you think many students have problems in learning or using SQL?

Appendix E: Comprehension Task

What is this SQL command trying to determine? Give your answer in plain English. Now say what you think the result of the query will be, when applied to the given Database.

```
SELECT Gardener.Name, Plant.Name, Date, Amount
FROM Picked, Gardener, Plant
WHERE Plant.PlantId = Picked.PlantFK
AND Gardener.GardenerId =Picked.GardenerFK
AND  Picked.GardenerFK = 2
ORDER BY Date
```

GardenerID	Name	Age
0	Fadila	36
1	Salim	38
2	Tim	15
3	Erin	12

Gardener

PlantID	Name	Sunlight	Weight	Water
0	Carrot	.26	.82	.08
1	Beet	.44	.80	.04
2	Corn	.44	.76	.26
3	Tomato	.42	.80	.16
4	Radish	.28	.84	.02

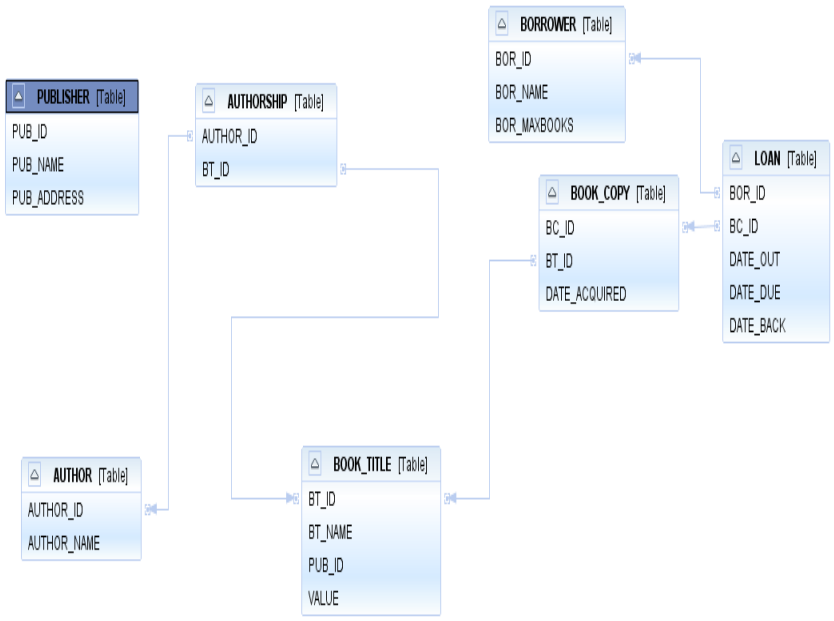
Plant

PlantFK	GardenerFK	LocationFK	Date	Amount	Weight
0	2	0	08-18-2005	28	2.32
0	3	1	08-16-2005	12	1.02
2	1	3	08-22-2005	52	12.96
2	2	2	08-28-2005	18	4.58
3	3	3	08-22-2005	15	3.84
4	2	0	08-16-2005	23	0.5

Picked

Appendix F: Expert Observation Task

Al Amal's Library keeps information on books held, borrowers who borrow these books and the loans of these books, which the borrowers make. In addition information is held about the authors and publishers of these books.Note that the **Book_copy** table holds information on the physical books stored in the library whereas the **Book_title** table holds information on a particular publication of a book (For example, there are two copies of 'Winnie the Pooh', with bc_id of 101 and 102.). A book may have a number of authors and this is indicated in the **Authorship** table.The attribute **bor_maxbooks** indicates the maximum number of books that a borrower can borrow at a time. Also, a book, which is still out on loan, will have a blank **date_back** field in the **loan** table.



Questions:

Q1: Give the titles of books that have more than one author.

Q2: Display the names of borrowers who have never returned a book late

Appendix G: SQL Patterns Evaluation Task one

Write SQL statement to find all employees who earn more than average salary in their department. Display Last name, Salary, Department Id and the Average salary for the department. Sort by, Average salary.

Solution: without SQL patterns.

Solution: with SQL patterns.

Patterns Extra information

To calculate the average salary	004 “group function” pattern
To sort by average salary	005 “grouping Rows” pattern
To find the salary that is more than average salary	002” using Sub queries” pattern.
To find all employees who earn more than average salary in their department	001 “querying from one table twice” pattern

DESCRIBE employees

```
Name                           Null     Type
------------------------------ -------- -------------
EMPLOYEE_ID                    NOT NULL NUMBER(6)
FIRST_NAME                              VARCHAR2(20)
LAST_NAME                      NOT NULL VARCHAR2(25)
EMAIL                          NOT NULL VARCHAR2(25)
PHONE_NUMBER                            VARCHAR2(20)
HIRE_DATE                      NOT NULL DATE
JOB_ID                         NOT NULL VARCHAR2(10)
SALARY                                  NUMBER(8,2)
COMMISSION_PCT                          NUMBER(2,2)
MANAGER_ID                              NUMBER(6)
DEPARTMENT_ID                           NUMBER(4)
```

SELECT * FROM employees;

	EMPLOYEE_ID	FIRST_N...	LAST_N...	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMI...	MANAGER_ID	DEPARTMENT_ID
1	100	Steven	King	SKING	515.123.4567	17-JUN-87	AD_PRES	24000	(null)	(null)	90
2	101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-89	AD_VP	17000	(null)	100	90
3	102	Lex	De Haan	LDEHAAN	515.123.4569	13-JAN-93	AD_VP	17000	(null)	100	90
4	103	Alexander	Hunold	AHUNOLD	590.423.4567	03-JAN-90	IT_PROG	9000	(null)	102	60
5	104	Bruce	Ernst	BERNST	590.423.4568	21-MAY-91	IT_PROG	6000	(null)	103	60
6	107	Diana	Lorentz	DLORENTZ	590.423.5567	07-FEB-99	IT_PROG	4200	(null)	103	60
7	124	Kevin	Mourgos	KMOURGOS	650.123.5234	16-NOV-99	ST_MAN	5800	(null)	100	50
8	141	Trenna	Rajs	TRAJS	650.121.8009	17-OCT-95	ST_CLERK	3500	(null)	124	50
9	142	Curtis	Davies	CDAVIES	650.121.2994	29-JAN-97	ST_CLERK	3100	(null)	124	50
10	143	Randall	Matos	RMATOS	650.121.2874	15-MAR-98	ST_CLERK	2600	(null)	124	50
11	144	Peter	Vargas	PVARGAS	650.121.2004	09-JUL-98	ST_CLERK	2500	(null)	124	50
12	149	Eleni	Zlotkey	EZLOTKEY	011.44.1344.429018	29-JAN-00	SA_MAN	10500	0.2	100	80
13	174	Ellen	Abel	EABEL	011.44.1644.429267	11-MAY-96	SA_REP	11000	0.3	149	80
14	176	Jonathon	Taylor	JTAYLOR	011.44.1644.429265	24-MAR-98	SA_REP	8600	0.2	149	80
15	178	Kimberely	Grant	KGRANT	011.44.1644.429263	24-MAY-99	SA_REP	7000	0.15	149	(null)
16	200	Jennifer	Whalen	JWHALEN	515.123.4444	17-SEP-87	AD_ASST	4400	(null)	101	10
17	201	Michael	Hartstein	MHARTSTE	515.123.5555	17-FEB-96	MK_MAN	13000	(null)	100	20
18	202	Pat	Fay	PFAY	603.123.6666	17-AUG-97	MK_REP	6000	(null)	201	20
19	205	Shelley	Higgins	SHIGGINS	515.123.8080	07-JUN-94	AC_MGR	12000	(null)	101	110
20	206	William	Gietz	WGIETZ	515.123.8181	07-JUN-94	AC_ACC...	8300	(null)	205	110

Appendix H: SQL Patterns Evaluation Task – Pre-test

Name (optional) _____ Date:

Gender: Male Female level : diploma higher diploma BeTch

How much you have got in SQL and Synatx course? A B C D other ……

How much your GPA? ……. **<u>Please answer the following questions:</u>**

1	Group functions return one result per row	True	False
2	Self-join is joining two different tables by matching their keys	True	False
3	Which of the following SQL statements is correct?	***Tick the correct answer***	
	SELECT CustomerName, COUNT(CustomerName) FROM Orders		
	SELECT CustomerName, COUNT(CustomerName) FROM Orders ORDER BY CustomerName		
	SELECT CustomerName, COUNT(CustomerName) FROM Orders GROUP BY CustomerName		
	All of them		
4	The EXISTS operator ensures that the search in the inner query does not continue when at least one match?	True	False
5	By using a WHERE clause, you can exclude rows after dividing them into groups? ***Tick the correct answer***		
	Ture	False	
6	Creating subqueries to query values based on ……criteria		

<table>
<tr><td rowspan="2"></td><td colspan="4">(fill in the answer)</td></tr>
<tr><td>(a) dynamic</td><td>(b) unknown</td><td>c. Changeable</td><td>d.All of them</td></tr>
<tr><td>7</td><td colspan="4">If your filtering criteria in WHERE CONDITION is dynamic it change each time you run the query then you assume to use _____ (fill in the answer)</td></tr>
<tr><td>8a</td><td colspan="4">When you are asked to compare two values in the same table within the same column then you will need to _____ (fill in the answer)</td></tr>
<tr><td rowspan="2">8b</td><td colspan="4">What do you call the join in this example?

SELECT a.sales_person_id, a.name, a.manager_id, b.sales_person_id, b.name
FROM sales_person a, sales_person b
WHERE a.manager_id = b.sales_person_id;</td></tr>
<tr><td colspan="4">Answer:</td></tr>
<tr><td rowspan="2">9</td><td colspan="4">Look at these two queries and answer the following questions:</td></tr>
<tr><td colspan="2">Query 1: SELECT columns
FROM tables
WHERE EXISTS (subquery);</td><td colspan="2">Query 2: SELECT columns
FROM tables
WHERE column1 IN (subquery);</td></tr>
<tr><td>9a</td><td colspan="4">Query 1 uses Exists operation and query 2 uses IN operation. Bothe operations are used to _____ the result of the main query. (fill in the answer)</td></tr>
<tr><td>9b</td><td colspan="4">When the subquery return at least one row the _____ operator return true (fill in the answer)</td></tr>
<tr><td>10</td><td colspan="4">Define subquery? where you can use subquery.</td></tr>
</table>

Appendix I: SQL Patterns Evaluation Task – Post-test

Name (optional) _____ Date:

Gender: Male Female **Please answer the following questions:**

1	A query within a query where the inner query is evaluated for each row in the outer query is called	***Tick the correct answer***	
	Join		
	Subquery		
	View		
	All of the above		
2	Joining a Table to itself is called Inner Join Outer Join Equi-Join Self Join	***Tick the correct answer***	
3	Which of the following SQL statements is not correct?	***Tick the correct answer***	
	SELECT CustomerName, COUNT(CustomerName) FROM Orders		
	SELECT CustomerName, COUNT(CustomerName) FROM Orders ORDER BY CustomerName		
	SELECT CustomerName, COUNT(CustomerName) FROM Orders GROUP BY CustomerName		
	All of them E. None of them		
4	The EXISTS operator ensures that the search in the inner query does not continue when at least one match?	True	False
5	Which of the following order of execution Oracle uses a SQL query containing the		

	clause having, where, group by and group function is used?			
	a. where, group by, group function, having b. group by, having, where, group function c. having, group by, where, group function d. group function, having, group by, where			
6	Creating subqueries to query values based on ……criteria ***(fill in the answer)***			
	(a) dynamic	(b) unknown	c. Changeable	d. All of them
7	If your filtering criteria in WHERE CONDITION is dynamic it change each time you run the query then you assume to use _____ ***(fill in the answer)***			
8a	When you are asked to compare two values in the same table within the same column then you will need to _____ ***(fill in the answer)***			
8b	What do you call the join in this example? SELECT a.sales_person_id, a.name, a.manager_id, b.sales_person_id, b.name FROM sales_person a, sales_person b WHERE a.manager_id = b.sales_person_id;			
	Answer:			
9	Look at these two queries and answer the following questions:			
	Query 1: SELECT columns FROM tables WHERE EXISTS (subquery);		Query 2: SELECT columns FROM tables WHERE column1 IN (subquery);	
9a	Query 1 uses Exists operation and query 2 uses IN operation. Bothe operations are used to filterthe result of the main query. explain the difference between them.			

9b	When the subquery return at least one row the Exists operator return ----- ***(fill in the answer)***
10	Consider the emp table having columns empno, ename Which of eth following SQL query fetches empno that occur more than twice in the emp table select count(*) from emp group by empno having count(*) >2; select empno, count(*) from emp having count(*) >2; select empno, count(*) from emp where count(*) >2; select empno, count(*) from emp group by empno having count(*) >2;

Analysis of pre test

Question	Related concept	Max Mark /100	Std Mark
1	Grouping	2	
2	Self join	2	
3	Grouping	2	
4	Existence	2	
5	Grouping	2	
6	Subquery	2	
7	Subquery	2	
8	Self join	2	
9	Self join	2	
10	Exists filtering and subquery	2	
11	Grouping	2	

Appendix J: SQL Patterns Evaluation Task – Questions

Q1: list the name of all customers who have more than one order

Acceptable facts:

1- Customer name need to be displayed with condition “more than one order”
2- Customer order need to be check
3- PURCHASE_ORDER table will be used

Acceptable knowledge:

8- Two query need to be done to search for customer order
9- Customer with two or more order need to be displayed
10- Two copy of the table PURCHASE_ORDER need to be used
11- Each table will have different allies
12- The two table will be join
13- The joint of two copy of the same table called self join
14- The order numbers need to be check are different

Q2: For each customer find the total purchase cost that he\she order limit you answer to those with total 1000 or more

Acceptable facts:

1- Total purchase need to be calculated
2- PURCHASE_COST column need to be used to calculated the cost.

3- Customer name or customer ID column will be used with a limit condition for those with total purchase 1000 or more
4- The following table are needed PRODUCT, PURCHASE_ORDER, CUSTOMER(optionally)

Acceptable knowledge:

1- Using SUM function to calculate the total of PURCHASE_COST column
2- Use group by to group the result either the customer name or customer id
3- Join the required tables using their primary keys PRODUCT, PURCHASE_ORDER, CUSTOMER(optionally)
4- Filter the grouped result using HAVE to limit the result with those total purchase 1000 or more

Q3: Display the name of the customer who have bought a product from Google

Acceptable facts:

1- Customer name need to be displayed with condition “bought a product from Google” from table Customer
2- Google is a manufacture so MANUFACTURER table will be used
3- There is no direct link between table Customer and table MANUFACTURER
4- There is no table in the database have such information “customer who bought a product from Google ”

Acceptable knowledge:

1- The names of the customers are in Customer table and it is our main query.
2- These name need to be filtering based on a dynamic criteria which is those only with Google manufacture. There is no table in the database have such information “customer who bought a product from Google” a query can be generated temporary that have such dynamic information .

3- The main query in 1 is filtered based on all the matched result in 2(sub query)

Q4: Display the product that have been bought by at least one customer

Acceptable facts:

1- PRODUCT table will be used and Description column will be selected it is our main query
2- List of all products that have been bought by customer need to be generated temporary that have such dynamic information
3- PURCHASE_ORDER and CUSTOMER tables need to be used

Acceptable knowledge:

1- The Description of the products is in PRODUCT table is selected and it is our main query.
2- List of all products that have been bought by customer need to be generated temporary that have such dynamic information. There is no table in the database have such information a query can be generated temporary that have such dynamic information .
3- The main query in 1 is filtered based the existence of at least one matched result in 2(sub query)

Q5: list the name of the freight company and the number of customers who used it.

Acceptable facts:

1- Total number of customers need to be calculated

2- Customer name column need to be used to calculated the number of customer.
3- FREIGHT_COMPANY will be used to group the result
4- The following table are needed CUSTOMER,PURCHASE_ORDER

Acceptable knowledge:

1- Using Count function to calculate the number of customers
2- Use group by to group the result either the customer name or customer id
3- Join the required tables using their primary keys PURCHASE_ORDER, CUSTOMER.
4- No filtering is required.

Appendix K: SQL Patterns Evaluation Task – Tutorial

Main Task	time
Introduction • The aim of the tutorial • The task that will be cover • Evaluation of the tutorial	5 mints
Problem solving task • How to use the grid • Fact identification and the required knowledge	10 mints
Checklist • How to use check list • The purpose of the checklist	5 mints
Patterns description *for patterns' group only* • What are patterns? • How the patterns look like? • How to use them?	15 mints
This process per question each Q 10 mints X 5 = 50 mints	50 mints

Then one question at a time will be present to students, all will be given the sheet of problem solving(SPSS) sheet. Question will be discuss in term of fact identification and knowledge requirement. Then how to use the checklist to identify the correct patterns.	2 mints
Each student will be ask to think about the display question's fact and knowledge and which patterns will be using and then facts and knowledge will be discussed	2 mints
student will be asked to look at the related material\patterns and solve the query	7 mints
solution will be discussed	4 mints
Group discussion **For example: When all questions are discussed students will be divided into two groups and asked them to study the given material\patterns by solving one hard question or mini questions the aim of this that all get time to study the material\patterns and understand them.**	30 mints
Total	2 hrs

Extra document: check list for problem solving

Problem solving strategy task
Structured strategy for problem solving Questions
(To be used by students)

Step 1: Identify, define and understand the problem

You need to collect data - the facts (as opposed to working with opinions or pet theories about the problem).

- ✓ The tables that I need to work with.
- ✓ The columns that I need to use.
- ✓ Any relation between the tables.
- ✓ Any calculation that needs to be done.

Step 2: Identify and analyse possible causes

- ✓ The data that I am looking for is not available in one table.
- ✓ The values that I need are not available in one column or one row.
- ✓ The values that I need require some modification such as calculating, adding to other data or comparing against other data.
- ✓ I need some data temporary for a special purpose only.
- ✓ My data needs to be checked according to its existence in other data.
- ✓ The data that I need to work with is in one table and for each row in the table the value of one column needs to be compared to all values in other or similar columns.

Step 3: Generate solutions

This has to happen after you have identified causes, of course. Also, search for more than one (potential) solution. This will give options from which to make decisions.

- ✓ Tables need to be joined.
- ✓ Columns need to be displayed.
- ✓ Some functions need to be applied to some columns.
- ✓ Some data needs to be grouped and one result required to be generated per group.

- ✓ *Special Purpose Data from different tables* need to be generated temporary to serve other main data such as filtering these data by checking the existence of at least one value or all values.
- ✓ The data that I need to work with is in one table and for each row in the table the value of one column need to be compared to all values in other or similar columns. Therefore two copy of the same tables need to be generated.

Step 4: Select one or combine more solutions to test out

Step 5: Plan of Action

A big or frequently re-occurring problem may need ***a plan of action***, but often you can go directly from step 3 to step 6.

Step 6: Corrective action

Take corrective action by implementing the selected solution. This will create change, of course.

Step 7: check the results

Do this by collecting more data - have you **solved** the problem?

If not, loop back to step 1.

Step 8: Improve your work

Continue to improve by asking: how can you make the solution better?

- ✓ I am confident with my answer.
  A: 0% b: 25% c: 50% d: 75% e: 100%
- ✓ I believe this is the best answer that I can come with.
  A: 0% b: 25% c: 50% d: 75% e: 100%
- ✓ I am sure that there are other alternative options to solve the above question that experts can come with.
  A: 0% b: 25% c: 50% d: 75% e: 100%

Appendix L: SQL Patterns Evaluation Task – consent Form

This experiment is part of a PhD research. The aim of this experiment is to investigate the effect of SQL PATTERNS on learner's knowledge and query writing performance. The experiment will take about six hours distributed over several days to complete. At the start of the experiment, you will need to complete a pre test task, then a short tutorial about SQL patterns will be presented, and two kinds of the tasks you will need to perform. The first one you need to show how to solve a set of SQL query, the second task you will be using an interface where you will need to write SQL queries to solve the given task. At the end of the experiment, you will be asked to complete a questionnaire.

All results will be held in strict confidence, ensuring the privacy of all participants. No personal participant information will be stored with the data all information will be anoymised. Online data will be stored in a password protected computer account; paper data will be kept in a single-occupant locked office.

A feedback email message will be sent to all participants, after the data has been analyzed.

Your participation in this experiment will have no effect on your marks for the subject at this, or any other university.

Please note that it is the SQL patterns, not you, that are being evaluated. You may withdraw from the experiment at anytime without prejudice, and any data already recorded will be discarded
If you have any further questions regarding this experiment, please contact:
Huda Al-shuaily
huda@dcs.gla.ac.uk

I have read this information sheet, and agree to voluntarily take part in this experiment:

Name:_____ Email:_____
Signature:----------

Appendix M: SQL Patterns Evaluation Task – SPSS Form

Question1 **Q1: list the name of all customers who have more than one order**	
The tables that I need to work with.	
The columns that I need to use.	
Any relation between the tables.	
Any calculation or conditions that needs to be done.	
Concepts	
Question2 **: For each customer find the total purchase cost that he\she order limit you answer to those with total 1000 or more**	
The tables that I need to work with.	
The columns that I need to use.	
Any relation between the tables.	
Any calculation that needs to be done.	
Concepts	
Question3 **list the name of the freight company and the number of customers who used it.**	
The tables that I need to work with.	
The columns that I need to use.	
Any relation between the tables.	
Any calculation that needs to	

be done.	
Question4: **Display the name of the customer who have bought a product from Google**	
The tables that I need to work with.	
The columns that I need to use.	
Any relation between the tables.	
Any calculation that needs to be done.	
Concepts	
Question5 **Display the product that have been bought by at least one customer**	
The tables that I need to work with.	
The columns that I need to use.	
Any relation between the tables.	
Any calculation that needs to be done.	
Concepts	

SPSS form with sample solution

Question1 **Q1: list the name of all customers who have more than one order**	
The tables that I need to work with.	*CUSTOMER, PURCHASE_ORDER*
The columns that I need to use.	*customer."NAME*
Any relation between the tables.	*where c.CUSTOMER_ID= po1.CUSTOMER_ID and c.CUSTOMER_ID= po2.CUSTOMER_ID*
Any calculation or conditions that needs to be done.	*and po1.ORDER_NUM <>po2.ORDER_NUM*

Concepts	*Self-join and natural join*
Question2 **: For each customer find the total purchase cost that he\she order limit you answer to those with total 1000 or more**	
The tables that I need to work with.	*PRODUCT, PURCHASE_ORDER*
The columns that I need to use.	*CUSTOMER_ID, sum(PURCHASE_COST) as total*
Any relation between the tables.	*where p.PRODUCT_ID= PO.PRODUCT_ID*
Any calculation that needs to be done.	*group by PO.CUSTOMER_ID having sum(p.PURCHASE_COST)>1000*
Concepts	*Aggregation , join*
Question3 **list the name of the freight company and the number of customers who used it.**	
The tables that I need to work with.	*CUSTOMER,PURCHASE_ORDER*
The columns that I need to use.	*count(CUSTOMER."NAME"), PURCHASE_ORDER.FREIGHT_COMPANY*
Any relation between the tables.	*CUSTOMER.CUSTOMER_ID=PURCHASE_ORDER.CUSTOMER_ID*
Any calculation that needs to be done.	*group by PURCHASE_ORDER.FREIGHT_COMPANY*
Question4**: Display the name of the customer who have bought a product from Google**	
The tables that I need to work with.	*CUSTOMER, PRODUCT*
The columns that I need to use.	*CUSTOMER.NAME PURCHASE_ORDER.CUSTOMER_ID*
Any relation between the tables.	*app.PURCHASE_ORDER.PRODUCT_ID=app.P*

	RODUCT.PRODUCT_ID app.PURCHASE_ORDER,app.MANUFACTURER
Any calculation that needs to be done.	*.CUSTOMER.CUSTOMER_ID IN select app.PURCHASE_ORDER.CUSTOMER_ID from app.PRODUCT, app.PURCHASE_ORDER,app.MANUFACTURER where app.PURCHASE_ORDER.PRODUCT_ID=app.PRODUCT.PRODUCT_ID and app.PRODUCT.MANUFACTURER_ID=app.MANUFACTURER.MANUFACTURER_ID 'and app.MANUFACTURER."NAME"='Google*
Concepts	*Subquery, join*
Question5 **Display the product that have been bought by at least one customer**	
The tables that I need to work with.	*PRODUCT, app.PURCHASE_ORDER, app.CUSTOMER*
The columns that I need to use.	*.PRODUCT.DESCRIPTION*
Any relation between the tables.	*.PURCHASE_ORDER.PRODUCT_ID=app.PRODUCT.PRODUCT_ID and app.PURCHASE_ORDER.CUSTOMER_ID= app.CUSTOMER.CUSTOMER_ID*
Any calculation that needs to be done.	*where exists*
Concepts	*Subquery, filtering by exists*

Appendix N:SQL Patterns Evaluation-Usability Questionnaire

Thank you very much for participating in our research project. Your feedback will be crucial for further improvements of this research and we would be most grateful if you could take time to fill this questionnaire. The questionnaire is anonymous, and you will not be identified as an informant without your consent. You may at any time withdraw your participation, including withdrawal of any information you have provided. By completing this questionnaire, however, it will be understood that you have consented to participate in the project and that you consent to publication of the results of the project with the understanding that anonymity will be preserved. **Please answer the following questions:**

Patterns Evaluation if you have used the pattern only						
1	Did you enjoy learning from SQL Patterns				Yes	No
2	Would you recommend SQL patterns to other students?				Yes	No
3	How much time did you need to understand the patterns content?				***Tick the correct answer***	
	a. substantial time (most of the session)					
	b. one day					
	c. less than 5 hours					
	d. less than one hour					
	e. never					
4	Would you like to use SQL Patterns in the other related courses				***Tick the correct answer***	
	a. Yes b. No c. I am not sure					
5	How much SQL patterns helped you to understand the related concepts? ***Tick the correct answer***					
	a. 1(nothing)	b. 2	c. 3	d. 4	e. 5(very much)	
6	Did you find the structure of the pattern's easy to understand?					
	Very Difficult	difficult	Understandable	Easy	very easy	
7	Did SQLPB help you to understand the given question better					
	Strongly Disagree	Disagree	Neither	agree	Strongly agree	
8	Did SQL patterns help you to solve the given question faster?					

	Strongly Disagree	Disagree	Neither	agree	Strongly agree
9	What did you like in particular about SQL patterns?				
10	Which of these patterns was the most helpful to you? Arrange from the most helpful to least: o Join-Self o Grouping Result o Existence filtering o Dynamic Filtering Critera o Restricting Grouped Result o Natural join				
11	Which part of the pattern you find most helpful in solving the given question: A: context b: example c: force d: solution e: problem f: consequences				
12	Which part of the patterns helped you to understand the related concept a: context b: problem c: force d: solution e: example f: consequences				
13	The language of the pattern was easy to understand				
	Strongly Disagree	Disagree	Neither	agree	Strongly agree
14	SQL patterns helped me to solve the question in the post test?				
	Strongly Disagree	Disagree	Neither	agree	Strongly agree
15	SQL patterns are easy to remember			Yes	No
16	I was able to match the given question with correct patterns using the checklist				
	Strongly Disagree	Disagree	Neither	agree	Strongly agree
17	SQL Patterns helped me to feel confident about the solution that I gave to each question				
	Strongly Disagree	Disagree	Neither	agree	Strongly agree
18	Different parts of the pattern (problem, solution, force, example,..) helped me to understand and solve the question in a an efficient way?				
	Strongly Disagree	Disagree	Neither	agree	Strongly agree
19	The patterns helped me to ease the way to perform the given task				
	Strongly Disagree	Disagree	Neither	agree	Strongly agree

Appendix O: SQL Patterns Evaluation –Ethical clearance

ETHICS-CSE00865: Online Submission Decision

You forwarded this message on 3/24/2012 9:41 AM.

Marie-HÃ©lÃ¨ne Grosbras [m.grosbras@psy.gla.ac.uk]

Sent: Tuesday, March 20, 2012 6:18 PM

To: Huda Al-Shuaily

ETHICS APPROVAL FORM
(CSE)

NAME OF PROPOSER: Huda Al-Shuaily
EMAIL ADDRESS: huda@dcs.gla.ac.uk
Department/Group/Centre: IT
PROJECT TITLE: SQL PATTERNS
PROJECT REFERENCE NUMBER: CSE00865

I can confirm that this project has been considered by the CSE ethics committee and approved.

Marie-HÃ©lÃ¨ne Grosbras (Ethics Officer for CSE)

Appendix p: SQL Patterns

Dynamic Filtering Pattern:

SQL example query: Display the names of borrowers who have never returned a book late

IF

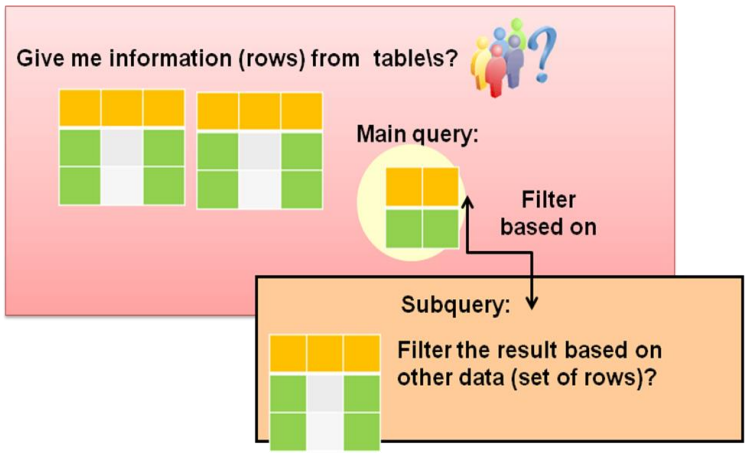
(you are reporting information from one table)

AND

(filtering these information based on the data of another table)

THEN

Look at "Dynamic filtering criteria " patterns



Context	A user wants to construct SQL search query for a relational database management system where the information which user want to display are with changeable or unknown filtering criteria
Problem	**How can you display a specific data (rows) from tables when conditions in WHERE clause are unknown or changeable?**
Forces	Specifying the filtering criteria makes the search more rigid and required a lot of time when there is a change in the database.

Solution

In the WHERE clause use subquery to give you a list of data that are used to filter your data

Format:

SELECT *column1*, *column2*
FROM *tables*
[WHERE] subquery

Consequences

Filtering criteria can be update automatically when the query is running, which means when the value of the table change filtering criteria will change dynamically. Therefore, the efficiency of the SQL code is better than having hard coding criteria.

Example: Display the names of borrowers who have never returned a book late.

Table1: Borrower

BOR_ID	BOR_NAME	BOR_MAXBOOKS
1	Jack Jones	5
2	Betty Smith	5
5	Jenny Wren	8
7	Peter Piper	5
9	Jay Patel	5
11	Nancy Green	8
12	Billy Black	8
14	Keith Kettle	5
15	Polly Peck	5

```
SELECT    distinct bor_name
 FROM     Borrower
WHERE     bor_id NOT IN
   ( SELECT    b.bor_id
     FROM      Loan l
     WHERE     b.bor_id = l.bor_id
     AND       l.date_back > l.date_due)
```

In this example, there is no available data that shows the borrower with the criteria **"never returned a book late"** , to obtain such values a subquery is needed.

Result of the main Query is :

BOR_NAME
Betty Smith
Billy Black
Jenny Wren
Nancy Green
Peter Piper

Result of Subquery that filter data in the main query : =====➔ BRO_ID is 1, 9, 15, 14, 14 filtering criteria will change dynamically each time you run the query if the data change in your tables.

Table1: Borrower b

BOR_ID	BOR_NAME	BOR_MAXBOOKS
1	Jack Jones	5
2	Betty Smith	5
5	Jenny Wren	8
7	Peter Piper	5
9	Jay Patel	5
11	Nancy Green	8
12	Billy Black	8
14	Keith Kettle	5
15	Polly Peck	5

Table1: loan l

BOR_ID	BC_ID	DATE_OUT	DATE_DUE	DATE_BACK
14	122	2003-02-16	2003-04-16	2003-04-02
14	121	2003-02-16	2003-04-16	2003-04-02
14	105	2003-02-16	2003-04-16	2003-04-02
14	107	2003-02-16	2003-04-16	2003-04-02
14	109	2003-02-16	2003-04-16	2003-04-02
14	120	2003-02-16	2003-04-16	2003-04-02
14	119	2003-02-16	2003-04-16	2003-04-02
1	106	2005-02-16	2004-02-01	2004-01-19
1	108	2006-02-16	2004-01-01	2004-01-19
9	101	2007-02-16	2004-03-02	2004-03-03
15	107	2011-05-16	2011-05-16	2011-05-17
14	108	2011-05-16	2011-05-18	2011-05-19
14	110	2011-05-16	2011-05-17	2011-05-20

Filtering by Existence Pattern:

SQL example query: Display name of borrower who have at least return one book late to the library

IF

(you are reporting information

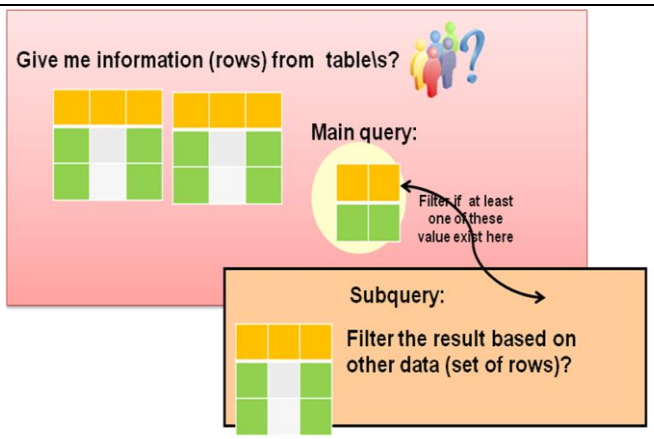
from one table

) **AND**

(filtering these information based on the existence of at least one value in the data of another table)

THEN

Look at "Filtering by existence " patterns



Context	A user wants to construct SQL search query that associates the data in one source with the matching or missing data in another source
Problem	**How can you produce efficiently a set of data by testing the existence of at least one matching record in your data to one of other data source?**
Forces	Filtering the data by its repetitive existence in other data sources causes performance problems and is not efficient. Searching a huge record is not efficient in term of speed and memory of the machine.

Solution: Test the existence of certain data within a SUBQUERY to implement a filtering logic using EXISTS or not EXISTS operation in SQL.

The following are the steps you need to apply:

- Report information from one or more tables – referred to as MAIN query or OUTER QUERY
- Filter the information(rows) of MAIN query based on data– referred to as subquery or inner query. It is usually enclosed in brackets in the outer query. For example: OUTER QUERY (INNER QUERY)
- The *inner query* returns a SET of values, and these values are used in the WHERE EXISTS section of the *outer query* to filter rows in the main table.
- when the subquery returns at least one row, the EXISTS operator returns TRUE. If the value does not exist, it returns FALSE.
  SELECT * \ values FROM main table WHERE EXISTS (subquery)

Format:

SELECT *column1, column2*

FROM *table*1, *table*2
[WHERE ***EXISTS\ not EXISTS (subquery)***

Consequences

EXISTS are intended to improve query performance because When the first record is found in the sub query, the conditional statement is set to TRUE and aborted. EXISTS are preferred over IN operator. This is because IN operator will check all matching records in the list. Therefore, EXISTS The also enhance the performance by reducing the number of records that SQL Server needs to process

Example : list the name of borrower who have at least return one book late to the library7

Table1: Borrower

BOR_ID	BOR_NAME	BOR_MAXBOOKS
1	Jack Jones	5
2	Betty Smith	5
5	Jenny Wren	8
7	Peter Piper	5
9	Jay Patel	5
11	Nancy Green	8
12	Billy Black	8
14	Keith Kettle	5
15	Polly Peck	5

In this example, there is no available data that shows the borrower with the criteria **"at least returned a book late once"** , to obtain such values a subquery is needed.

```
SELECT    distinct bor_name
 FROM     Borrower
WHERE     exists
 ( SELECT     *
   FROM     Loan l, Borrower b
   WHERE    b.bor_id = l.bor_id
  AND       l.date_back > l.date_due
```

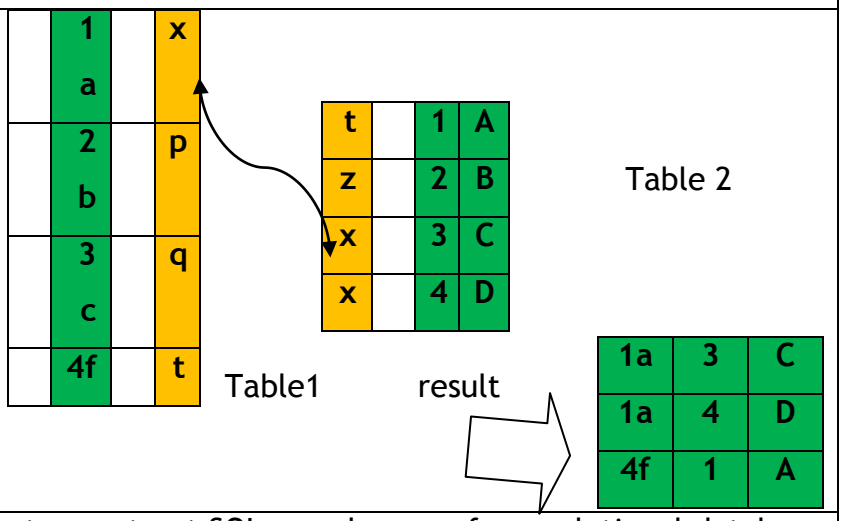
Result of Subquery that filter data in the main query : =====➔ filtering criteria her is changeable, unknown or dynamically that is why you need subquery each time you run the query.

Natural-Join Pattern:

SQL example query: list all information of employees and their department

IF
(all the data are in more than one table) **&**
(rows need to be filtered based on data in other rows in the tables)

THEN Look at "natural-join" pattern



Context	A user wants to construct SQL search query for a relational database management system where the information is spread over various rows within more than one table.
Problem	**How can you gather information that is distributed within more than one table?**
Forces	Searching two different values to compare between them means that you will have two separate queries that need to be linked together. You can use nested SQL queries that make multiple references to the table, but this might cause a performance problem.

Solution

Join tables to be able to relate rows from one table with other rows from the other table, known as a natural-join.

Format:

```
SELECT column1, column2
 FROM table1, table2
 [WHERE table1.column_name = table2.column_name]
```

Consequences

The disadvantage of this solution is that if you are joining huge tables requires a lot of memory resources in the DBMS.

Example:

list all information of employees and their department

- Specify the information that you need to get which is the employee name ,ID, …
- Specify the tables that the information you need are in, in this case Employee table and Departments.
- Specify the condition of your information. joining two tables

Select Employee_ID, Depatment_ID, Department_Name
From Employees E, Departments D
Where E. Depatment_ID = D. Depatment_ID

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
102	De Haan	90
202	Fay	20
205	Higgins	110
206	Gietz	110

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
10	Administration	1700
20	Marketing	1800
50	Shipping	1500
60	IT	1400
80	Sales	2500
90	Executive	1700
110	Accounting	1700
190	Contracting	1700

EMPLOYEE_ID	DEPARTMENT_ID	DEPARTMENT_NAME
200	10	Administration
201	20	Marketing
202	20	Marketing
124	50	Shipping
144	50	Shipping

Self-Join Pattern:

SQL example query: Get the titles of the books that have more than one copy in the library (where all book details are stored in a single table)

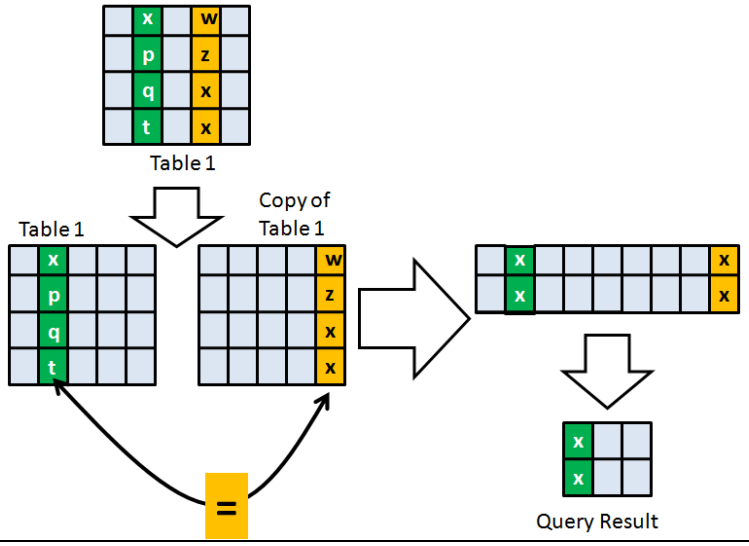
IF

(all the data are in one table)

&

(rows need to be filtered based on data in other rows in the same table)

THEN Look at "Self-join" pattern



Context

A user wants to construct SQL search query for a relational database management system where the information is spread over various rows within one table.

Problem

How can you compare values from different rows in the same column?

Forces

Searching for two different values to compare between them means that you will have two separate queries that need to be linked together. You can use nested SQL queries that make multiple references to the same table, but this might cause a performance problem.

Solution

Create two perspectives of the same table to be able to relate rows from that table with other rows from the same table, known as a self-join. This process efficiently connects a table with itself. This query joins a table to itself. It uses table name aliases so that each "instance" is easy to reference.

Format:

SELECT *column1, column2*
FROM *table* t1, *table* t2 ➔ t1, t2 are the table name aliases
[WHERE t1.column_name = t2.column_name]

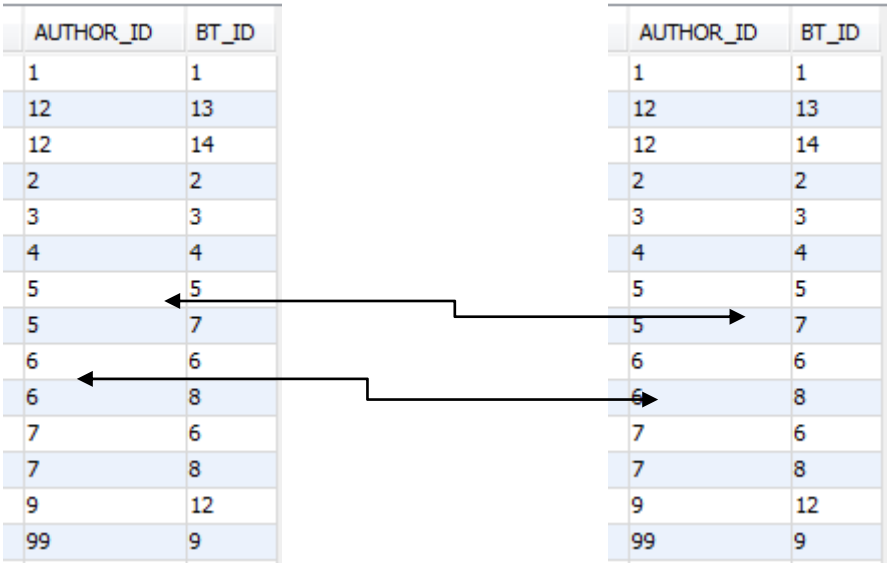
Consequences

The disadvantage of this solution is that if you are joining huge table to itself requires a lot of memory resources in the DBMS.

Example1: "Give the ID of books that have more than one author". You need to search or query in the authorship table twice, first to find the book title id and the author id and then to find if the same book has another author**.**

- Specify the tables that you need to use. And which table that you need to join to itself. In the above example you need to apply self join concept on Authorship table .
- Specify the required columns that you need to get directly from both tables.
- Specify any relation between the tables if you are getting information from more than one table for example WHERE ap1.bt_id = ap2.bt_id
- You need to write the following SQL statement:

SELECT distinct Bt_ID
FROM authorship ap1, authorship ap2
WHERE ap1.bt_id = ap2.bt_id
AND ap2.author_id <> ap1.author_id;



AUTHOR_ID	BT_ID
1	1
12	13
12	14
2	2
3	3
4	4
5	5
5	7
6	6
6	8
7	6
7	8
9	12
99	9

AUTHOR_ID	BT_ID
1	1
12	13
12	14
2	2
3	3
4	4
5	5
5	7
6	6
6	8
7	6
7	8
9	12
99	9

Result

Employees and Their Managers

Rajs works for Mourgos
Raphaely works for King
Rogers works for Kaufling
Russell works for King

Grouping Result Pattern:

SQL example query: For each author, find the total value of the books owned by the library that he/she wrote

IF

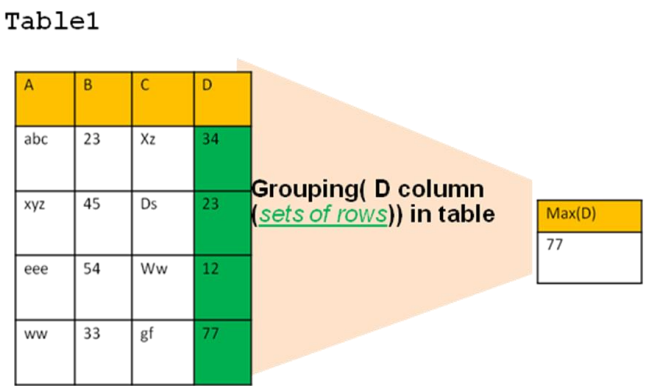
(you looking for a description of a group of data in a table such as count, Max, Min, AVG,..)

AND

(only one value per group is required)

THEN

Look at "Grouping Result " patterns



Context

A user wants to generate a characteristic description of a set of data through grouping identical data into one subset OR wants to evaluate\summarize all the data within each set of column in a table to provide a special purpose data.

Problem

How can you produce a group of data within each set of column in a table to provide a special purpose data?

Forces

Characteristic description need to apply to the data in the table such as caluclating SUM, MAX, MIN, AVG and Count. And only one value per group is required.

Solution

You can divide rows in a table into smaller groups by using the GROUP BY clause. And to evaluate or summarize a set of data you need to use one or more of group functions such as SUM, MAX, MIN, AVG

Format:

SELECT ***column, group_function(column)***
FROM ***table***
[WHERE ***condition*****]**
[GROUP BY ***group_by_expression*****]**

Consequences

Applying this patterns has the following affects:

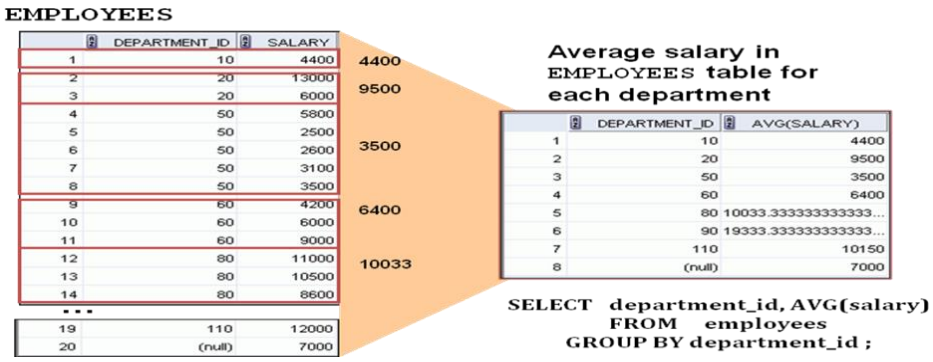
- It is not permissible to include column names in a SELECT clause that are not

referenced in the GROUP BY clause. The only column names that can be displayed, along with aggregate functions, must be listed in the GROUP BY clause and this will affect the query performance because will lead to unnecessary grouping.

- All the records in SELECT statement are either aggregated or covered in the group by statement.

Examples:

Example1: **list the department ID with the average salary per department**



- Specify the tables that you need to use which is Employee
- Specify the required columns that you need to get directly from the table: department_id and salary.
- Specify any columns that need to do some calculation or evaluation to get it. Average of salary
- Specify any relation between the tables.
- When grouping, keep in mind that all columns that appear in your SELECT column list, that are not aggregated (used along with one of the SQL aggregate functions), have to appear in the GROUP BY clause too in this case department_id. The result is grouped by the borrower name.