Data Preparation 101: How To Use Query-By-Example To Get Your Research Dataset Ready For Primetime

Paul J. Lazarony, California State University, Northridge, USA
Donna A. Driscoll, California State University, Northridge, USA

ABSTRACT

Researchers are often distressed to discover that the data they wanted to use in their landmark study is not configured in a way that is usable by a Statistical Analysis Software Package (SASP). For example, the data needed may come from two or more sources and it may not be clear to the researcher how to get them combined into one analyzable dataset. Fortunately, there is hope. The query facility within database management software (DBMS), such as Microsoft Access, is particularly well-equipped to do the data reconfigurations necessary to get datasets ready for a SASP. Unfortunately, most researchers have either never been exposed to a DBMS or are unaware of the powerful data-transforming queries that they can perform. The purpose of this paper is to introduce researchers to some very helpful and relatively easy to learn techniques for solving common dataset misconfiguration problems. It is presented in three sets of hands-on examples: (1) how to import a dataset from an electronic spreadsheet into a database table, (2) how to use an easy-to-learn DBMS facility called Query-By-Example (QBE) to perform specific data reconfiguration tasks, and (3) how to import the reconfigured dataset from the DBMS into a SASP. The software used is MS Excel, MS Access, and SPSS (originally called Statistical Package for the Social Sciences). The examples are presented step-by-step so that the reader can follow along using files downloadable from the authors' website

Keywords: Database; Dataset; Query; Research; Spreadsheet; Statistical Analysis

INTRODUCTION

The authors of this paper are information systems faculty who have discovered that database techniques can be very helpful in dealing with common problems associated with research datasets. This paper is the second in a series that attempts to make these techniques more accessible to researchers in other fields.

Reconfiguration of datasets in a database can be approached in two basic ways: (a) Structured Query Language (SQL) and (b) Query-By-Example (QBE). SQL is a high-level computer language that involves writing computer code to do queries of a database; QBE allows the user to formulate queries using a graphical user interface (which allows the use of dropdown menus, toolbars, and wizards) and is thus much more user-friendly.

Our first paper in this series focused on the first of these approaches. Specifically, we used SQL to efficiently correct, recode, and reorganize a dataset downloaded from one table that was embedded in a web page. The current paper focuses on the second approach. Specifically, we will demonstrate how to use QBE to combine data stored in multiple tables in a variety of useful ways.

This paper is presented in three phases, each of which includes hands-on examples. In **Phase I**, we demonstrate the steps required to import a dataset stored in an **MS Excel** file into an **MS Access** database table. Then, in **Phase II**, we take the reader through a series of five examples that illustrate how to combine data that exist in multiple tables. Finally, in **Phase III**, we show how to import the newly combined dataset from an **MS Access** table into **SPSS** for statistical analysis.

As previously mentioned, this example is designed so that the reader may follow along step-by-step. The Web page containing the data can be found at **www.csun.edu/~pjl26399/dataprep101**. The following software will also be required to complete the example: **MS Excel**, **MS Access**, and **SPSS**. If **SPSS** is not available, **Phase III** could also be accomplished using another statistical analysis package such as **SAS** or **Minitab**. The dataset could also be returned to **MS Excel** for statistical analysis.

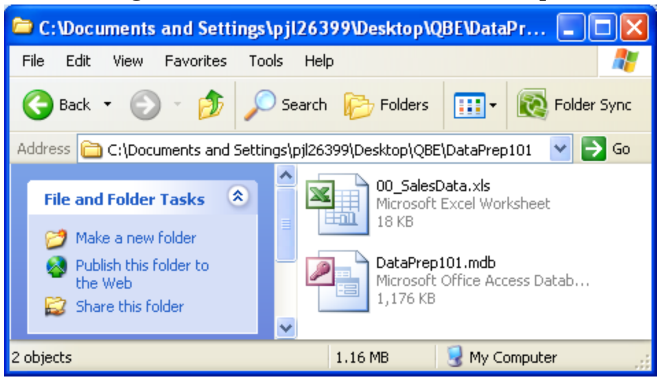
PHASE I: IMPORTING A DATASET FROM MS EXCEL INTO AN MS ACCESS TABLE

Although datasets could come to a researcher in a variety of forms, one of the most commonly used current formats is an **MS Excel** file. Depending on the particular dataset and the complexity of the required analysis, a researcher may sometimes choose to leave the data in **MS Excel** and simply do the analysis right there. At other times, however, it could be difficult to use the data in its original form. The following steps illustrate the ease of importing an **MS Excel** file into an **MS Access** table:

A. Download the **DataPrep101.mdb** and **00_SalesData.xls** files from the Web page shown above. Figure 1 shows the files after download. Be sure to save them in a new folder on your computer's hard drive, named **DataPrep101**.

B. Open the **00_SalesData.xls** in **MS Excel**. The file should look like Figure 2 below:

Figure 1: Files to Download for this Example

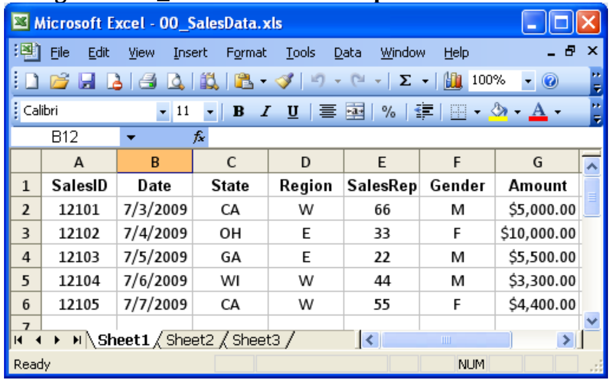

C:\Documents and Settings\pjl26399\Desktop\QBE\DataPr...
File Edit View Favorites Tools Help
Back Search Folders Folder Sync
Address C:\Documents and Settings\pjl26399\Desktop\QBE\DataPrep101 Go
File and Folder Tasks
Make a new folder
Publish this folder to the Web
Share this folder

00_SalesData.xls
Microsoft Excel Worksheet
18 KB

DataPrep101.mdb
Microsoft Office Access Datab...
1,176 KB

2 objects 1.16 MB My Computer

Figure 2: 00_SalesData.xls File Opened in MS Excel


Microsoft Excel - 00_SalesData.xls
File Edit View Insert Format Tools Data Window Help
100%
Calibri 11
B12
Sheet1 Sheet2 Sheet3
Ready NUM

	A	B	C	D	E	F	G
1	SalesID	Date	State	Region	SalesRep	Gender	Amount
2	12101	7/3/2009	CA	W	66	M	$5,000.00
3	12102	7/4/2009	OH	E	33	F	$10,000.00
4	12103	7/5/2009	GA	E	22	M	$5,500.00
5	12104	7/6/2009	WI	W	44	M	$3,300.00
6	12105	7/7/2009	CA	W	55	F	$4,400.00

C. The goal is to import the **00_SalesData.xls** spreadsheet file into a table in the **DataPrep101.mdb** database file. To complete this task, follow the steps below:

1. Open the **DataPrep101.mdb** file in MS Access. Your screen will look similar to Figure 3 below. Notice the Tables and Queries listed in on the left-hand side of the screen. We will be using these objects later in this paper (in **Phase II)**.
2. Notice the **Security Warning** in the middle of the screen. Click the **Options** button. The Microsoft Office Security Options dialog box will appear. See Figure 4.
3. The default is to leave the database file as read-only. Since changes to the database will be necessary, will need to enable all content. Click the radio button next to the **Enable this content option**. Then click **OK**. You will need to repeat these steps (2 & 3) each time you open the database file.
4. To import the **00_SalesData.xls** spreadsheet file, click the **External Data** tab, then click the Excel button on the toolbar. The Get External Data-Excel Spreadsheet dialog box will appear. See Figure 5.

Figure 3: DataPrep101.mdb File Opened in MS Access

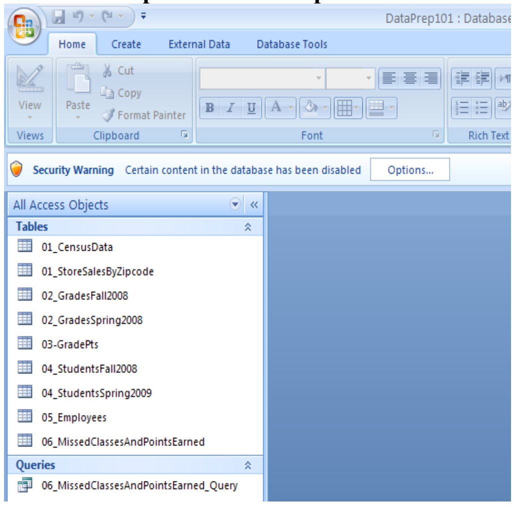
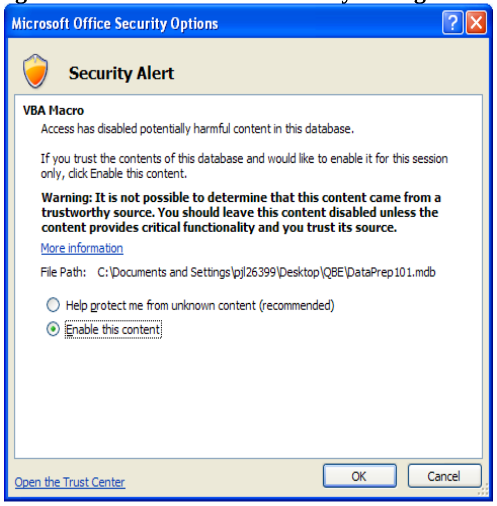


Figure 4: Microsoft Office Security Dialog Box



VBA Macro

Access has disabled potentially harmful content in this database.

If you trust the contents of this database and would like to enable it for this session only, click Enable this content.

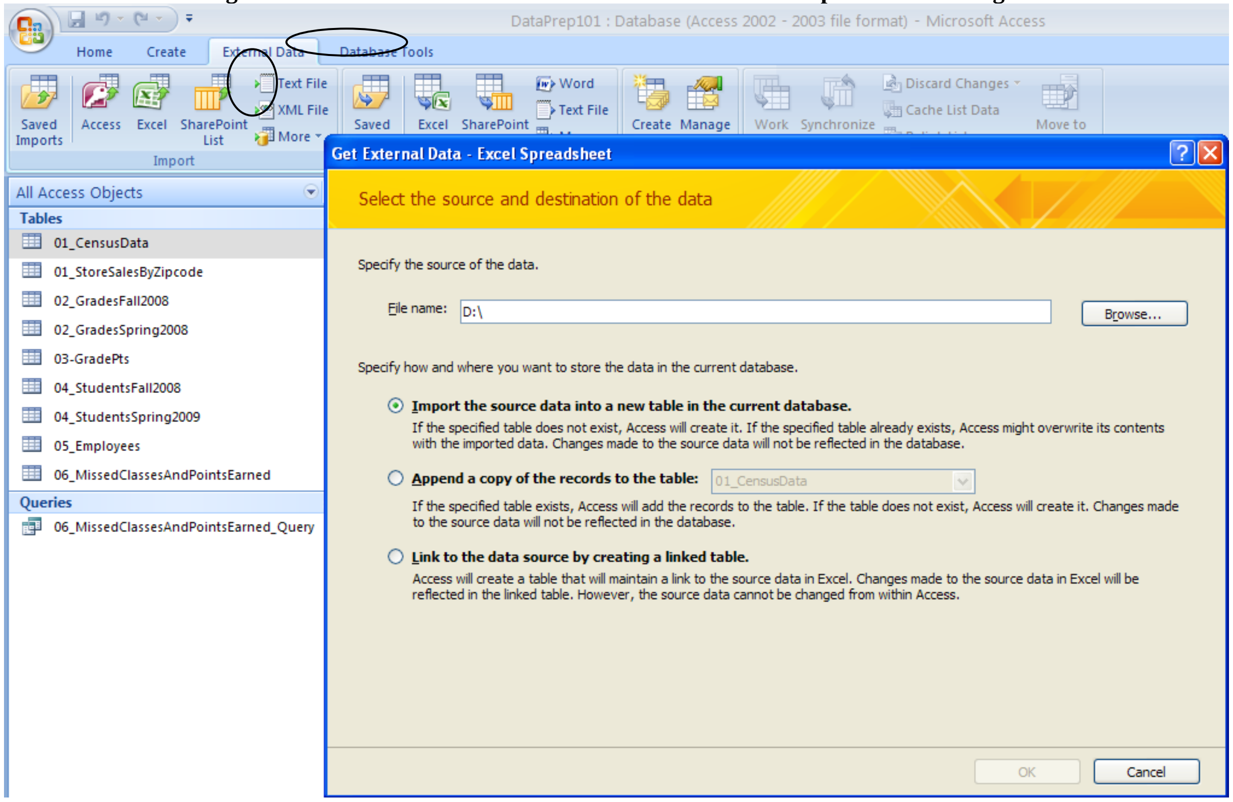
Warning: It is not possible to determine that this content came from a trustworthy source. You should leave this content disabled unless the content provides critical functionality and you trust its source.

More information

File Path: C:\Documents and Settings\pjl26399\Desktop\QBE\DataPrep101.mdb

- ○ Help protect me from unknown content (recommended)
- [✓ Enable this content]

Figure 5: Database Window with Get External Data-Excel Spreadsheet Dialog Box



Select the source and destination of the data

Specify the source of the data.

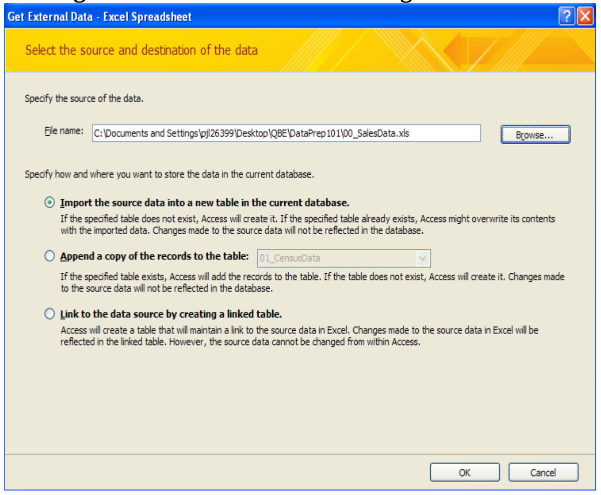
File name: D:\

Specify how and where you want to store the data in the current database.

- [✓ **Import the source data into a new table in the current database.**]
  If the specified table does not exist, Access will create it. If the specified table already exists, Access might overwrite its contents with the imported data. Changes made to the source data will not be reflected in the database.
- ○ **Append a copy of the records to the table:** 01_CensusData
  If the specified table exists, Access will add the records to the table. If the table does not exist, Access will create it. Changes made to the source data will not be reflected in the database.
- ○ **Link to the data source by creating a linked table.**
  Access will create a table that will maintain a link to the source data in Excel. Changes made to the source data in Excel will be reflected in the linked table. However, the source data cannot be changed from within Access.

5. Click the **Browse** button. Locate the **00_SalesData.xls** spreadsheet file; click **Open**. The **Get External Database** dialog box should now look similar to Figure 6.
6. Leave the radio button checked next to **Import the source data into a new table in the current database**. Click **OK**. The **Import Spreadsheet Wizard** will appear (see Figure 7).

Figure 6: Get External Data Dialog Box-Screen 2



Get External Data - Excel Spreadsheet

Select the source and destination of the data

Specify the source of the data.

File name: C:\Documents and Settings\pjl26399\Desktop\QBE\DataPrep101\00_SalesData.xls Browse...

Specify how and where you want to store the data in the current database.

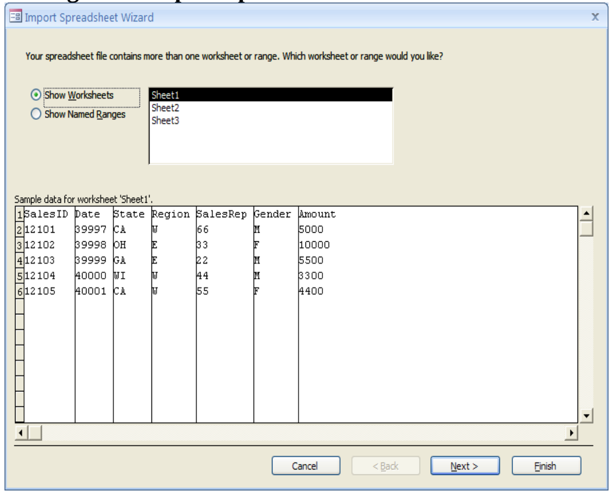
[✓ **Import the source data into a new table in the current database.**]
If the specified table does not exist, Access will create it. If the specified table already exists, Access might overwrite its contents with the imported data. Changes made to the source data will not be reflected in the database.

Append a copy of the records to the table: 01_CensusData
If the specified table exists, Access will add the records to the table. If the table does not exist, Access will create it. Changes made to the source data will not be reflected in the database.

Link to the data source by creating a linked table.
Access will create a table that will maintain a link to the source data in Excel. Changes made to the source data in Excel will be reflected in the linked table. However, the source data cannot be changed from within Access.

OK Cancel

Figure 7: Import Spreadsheet Wizard-Screen 1



Import Spreadsheet Wizard

Your spreadsheet file contains more than one worksheet or range. Which worksheet or range would you like?

[✓ Show Worksheets] Show Named Ranges

Sheet1
Sheet2
Sheet3

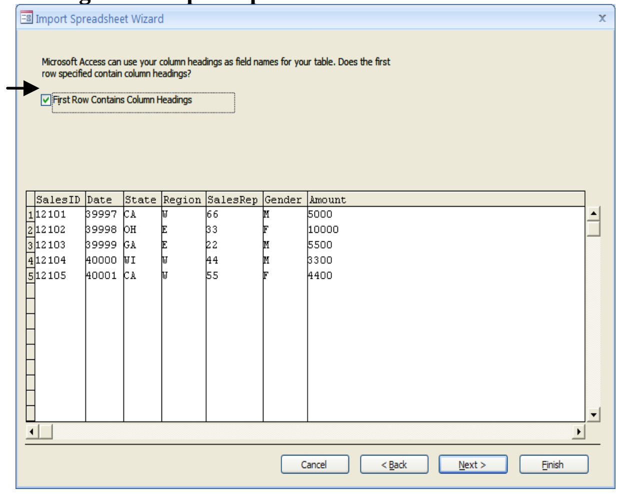
Sample data for worksheet 'Sheet1'.

1	SalesID	Date	State	Region	SalesRep	Gender	Amount
2	12101	39997	CA	W	66	M	5000
3	12102	39998	OH	E	33	F	10000
4	12103	39999	GA	E	22	M	5500
5	12104	40000	WI	W	44	M	3300
6	12105	40001	CA	W	55	F	4400

Cancel < Back Next > Finish

7. Notice that the all of the column headings and data from the **00_SalesData.xls** spreadsheet file are displayed in the wizard. Click **Next**. The second screen of the wizard appears (see Figure 8).
8. This second screen assumes that the first row of data from our spreadsheet file contains column headings. Since this assumption is correct, leave the checkmark next to **First Row Contains Column Headings**. Click **Next**. The third screen of the wizard appears (see Figure 9).

Figure 8: Import Spreadsheet Wizard-Screen 2



Import Spreadsheet Wizard

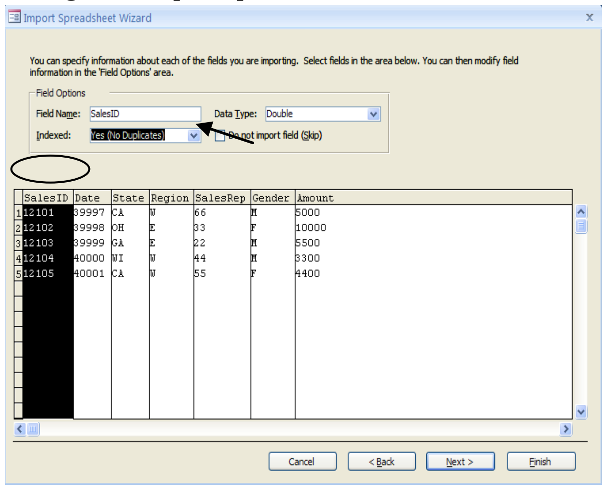
Microsoft Access can use your column headings as field names for your table. Does the first row specified contain column headings?

✓ First Row Contains Column Headings

	SalesID	Date	State	Region	SalesRep	Gender	Amount
1	12101	39997	CA	W	66	M	5000
2	12102	39998	OH	E	33	F	10000
3	12103	39999	GA	E	22	M	5500
4	12104	40000	WI	W	44	M	3300
5	12105	40001	CA	W	55	F	4400

Cancel < Back Next > Finish

Figure 9: Import Spreadsheet Wizard-Screen 3



Import Spreadsheet Wizard

You can specify information about each of the fields you are importing. Select fields in the area below. You can then modify field information in the 'Field Options' area.

Field Options

Field Name: SalesID Data Type: Double

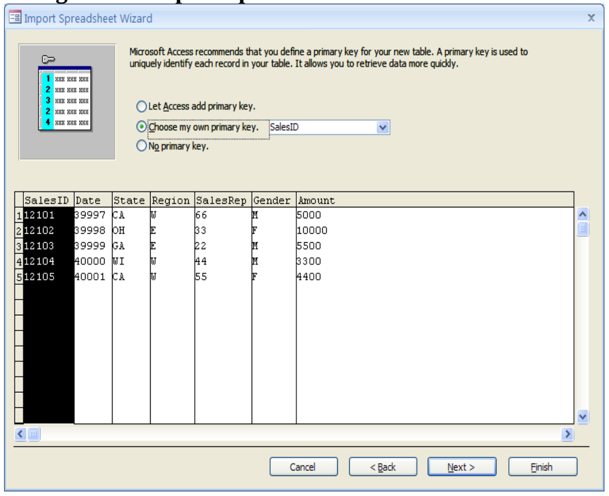
Indexed: Yes (No Duplicates) Do not import field (Skip)

	SalesID	Date	State	Region	SalesRep	Gender	Amount
1	12101	39997	CA	W	66	M	5000
2	12102	39998	OH	E	33	F	10000
3	12103	39999	GA	E	22	M	5500
4	12104	40000	WI	W	44	M	3300
5	12105	40001	CA	W	55	F	4400

Cancel < Back Next > Finish

9. This third screen of the wizard allows you to change the data type of each field (column), if needed, as well as indicate how the table will be indexed.. In our example, the only change we need is to have the SalesID column be set as a unique key field (so that it might later be used to link to a SalesID field in another table). To accomplish this, first click on the **SalesID** column heading. Second, click the down arrow next to **Indexed** and select **Yes (No Duplicates)**. Click **Next**. The fourth screen of the wizard appears (see Figure 10).
10. Click the radio button next to **Choose my own primary key**. This will select the **SalesID** as the primary key (or unique field) for this table. Click **Next**. The fifth and final screen of the wizard appears (see Figure 11).

Figure 10: Import Spreadsheet Wizard-Screen 4

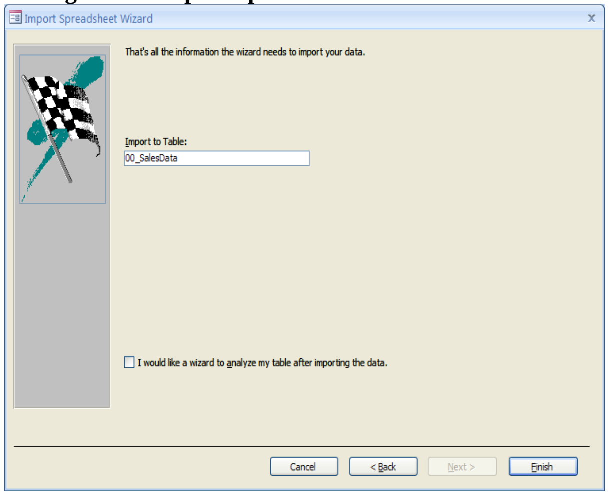


Microsoft Access recommends that you define a primary key for your new table. A primary key is used to uniquely identify each record in your table. It allows you to retrieve data more quickly.

Let Access add primary key. [✓ Choose my own primary key. SalesID] No primary key.

	SalesID	Date	State	Region	SalesRep	Gender	Amount
1	12101	39997	CA	W	66	M	5000
2	12102	39998	OH	E	33	F	10000
3	12103	39999	GA	E	22	M	5500
4	12104	40000	WI	W	44	M	3300
5	12105	40001	CA	W	55	F	4400

Figure 11: Import Spreadsheet Wizard-Screen 5



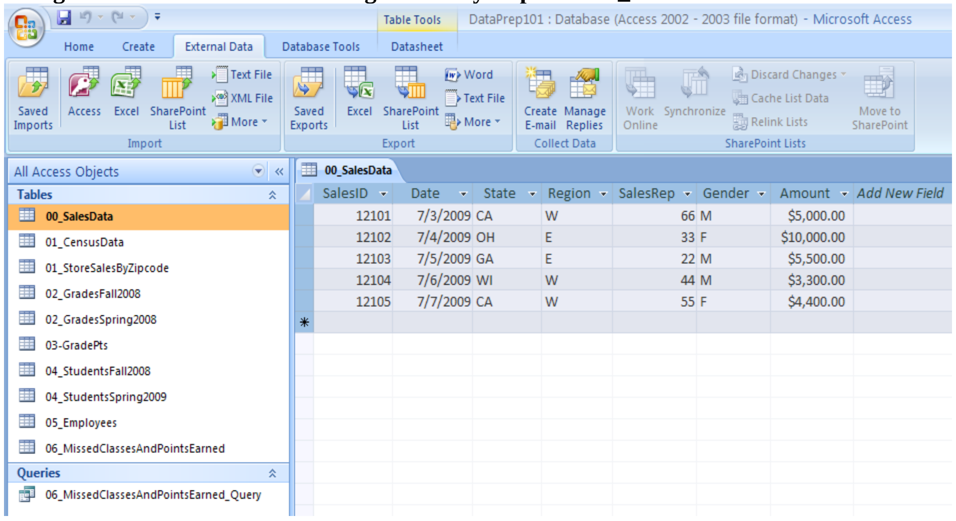
That's all the information the wizard needs to import your data.

Import to Table: 00_SalesData

☐ I would like a wizard to analyze my table after importing the data.

11. In the **Import to Table** text box, type in the new table name: **00_SalesData**. Click **Finish**. In the next dialog box, click **Close**. Congratulations, the **00_SalesData** table now appears in the list of tables of the left-side of the screen. Double click the new table name (**00_SalesData**) and the new table appears on the right (see Figure 12).

Figure 12: Database File Showing the Newly Imported 00_SalesData Table and Dataset



Tables:
- 00_SalesData
- 01_CensusData
- 01_StoreSalesByZipcode
- 02_GradesFall2008
- 02_GradesSpring2008
- 03-GradePts
- 04_StudentsFall2008
- 04_StudentsSpring2009
- 05_Employees
- 06_MissedClassesAndPointsEarned

Queries:
- 06_MissedClassesAndPointsEarned_Query

00_SalesData

SalesID	Date	State	Region	SalesRep	Gender	Amount	Add New Field
12101	7/3/2009	CA	W	66	M	$5,000.00	
12102	7/4/2009	OH	E	33	F	$10,000.00	
12103	7/5/2009	GA	E	22	M	$5,500.00	
12104	7/6/2009	WI	W	44	M	$3,300.00	
12105	7/7/2009	CA	W	55	F	$4,400.00	

12. To close the table, just press the **X** in the upper right corner of the table.
13. **Phase I** is now complete. The reader may now continue on to **Phase II** or close the **DataPrep101.mdb** file.

PHASE II: FIVE EXAMPLES OF HOW TO USE MS ACCESS' QUERY BY EXAMPLE TO RECONFIGURE DATASETS

Although the QBE facility in MS Access can be used to handle most types of data misconfigurations, some situations tend to occur fairly frequently and, unfortunately, without the use of such database techniques, can be either complete barriers to further progress on a project or a problem that ends up being dealt with by the brute force approach of fixing the problems by hand at a substantial cost in time, money, or both.

In **Phase II**, we focus on five relatively common data problems that can be solved using QBE in a tiny fraction of the time it would take to do it the old-fashioned (and labor-intensive) way:

A. combining the columns from two related tables (using a **JOIN** query),
B. adding the rows in one table into another table (using an **APPEND** query),
C. recoding a column in one table (using data stored in a second **CONVERSION** table),
D. dealing with tables having overlapping data (using **UNMATCHED** and **APPEND** queries), and
E. allowing the matching of values in one column that reference the values of another column in the same table (using a **SELF-JOIN** query).

These five situations are summarized in Table 1, along with a simple example of each that will then be described in detail. The reader is encouraged to work through these examples using the same database available from our website. (Please see the Introduction section of this paper for the file location.)

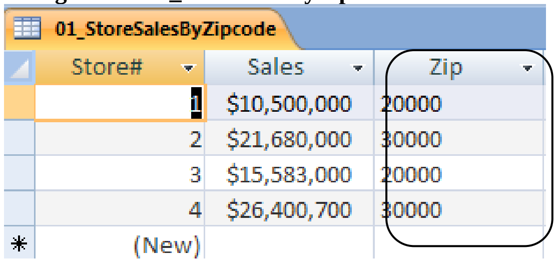
Table 1: Five Database Problems that can be Solved using QBE

	Situation Type	Example	
		Description	Illustration
A	**JOIN query** The columns of data might be in two separate datasets that have one common linking column.	One dataset of census data by zipcode and another of store sales data by store# which includes the zipcode for each store.	**Census Data:** Zipcode | Average Household Income **Store Sales by Zipcode:** Store # | Sales | Zipcode
B	**APPEND query** There might be two sets of rows having column sequences that are different from each other and that need to be combined into one larger dataset.	Two datasets of grade data in which the column names and column order are different and need to be aggregated into one comprehensive dataset.	**Grades Fall 2008:** SemesterID | SectionNo | StudentID | CourseNo | CourseTitle | LetterGrade **Grades Spring 2009:** SemesterID | StudentNum | TicketNo | CourseNo | ClassName | Grade
C	**CONVERSION query** One or more of the existing columns is not in analyzable form and needs to be recoded.	A grades dataset uses letters for grades, but the planned analysis requires numeric grade points. A second dataset containing letter grades and the corresponding grade points. The letter grades need to be converted into grade points.	**Grades Fall 2008:** SemesterID | SectionNo | StudentID | CourseNo | CourseTitle | LetterGrade **GradePts:** LetterGrade | GradePt
D	**Combining overlapping lists: UNMATCHED and APPEND queries.** There may be two sets of rows representing lists of people, places, or things existing at two different points in time that need to be combined into one comprehensive list.	Two lists of students for two different semesters need to be combined into one comprehensive list of unique students which does not include any duplicate records.	**Students in Major Spring 2008:** StudentID | LastName | FirstName **Students in Major Fall 2008:** StudentID | LastName | FirstName
E	**SELF-JOIN query** There might be two columns in the same dataset in which the values in one column reference the values in another.	The researcher wants to test whether an employee's stated opinion about the importance of home recycling is affected by his manager's stated opinion of home recycling. The available employee dataset includes columns of Employee ID, Manager's Employee ID, and Employee Home Recycling Importance, measured on a Likert-scale. A new Manager's Home Recycling Importance column needs to be generated.	**Employee:** EmployeeID | LastName | FirstName | ManagerID | Home Recycling Importance 10 Garcia Monica 1 20 Smith Mary 10 1 30 Doe John 10 1 40 Wong Steven 5 50 North Carla 40 5 60 Robinson Kazetta 40 5

A. *Combining the Columns from Two Related Tables (using a JOIN query).*

1. Open the **DataPrep101.mdb** file. Be sure to turn off the security features (see Figures 3 & 4). The following tables will be used for this example: (1) **01_StoreSalesByZipCode** and (2) **01_CensusData**. Figures 13 and 14 show the datasets for these two tables. Notice that the two tables have a common column, zip code. The goal will be to have a one dataset with the columns of Store#, Sales, Zip, and AvgHouseholdIncome.

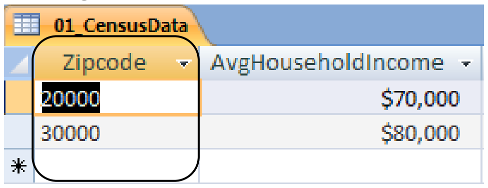
Figure 13: 01_StoreSalesByZipCode Table



01_StoreSalesByZipcode

Store#	Sales	Zip
1	$10,500,000	20000
2	$21,680,000	30000
3	$15,583,000	20000
4	$26,400,700	30000
(New)		

Figure 14: 01_CensusData Table



01_CensusData

Zipcode	AvgHouseholdIncome
20000	$70,000
30000	$80,000

2. A JOIN query will be used to achieve this goal. A JOIN is a specific form of SELECT query that combines the data from two or more tables based upon common fields. Here’s the process: Click the **Create** tab, then click on the **Query Design** button on the toolbar (see Figure 15). The **Show Table** dialog box appears (see Figure 16).

Figure 15: Creating a Query Using the Query Design Facility

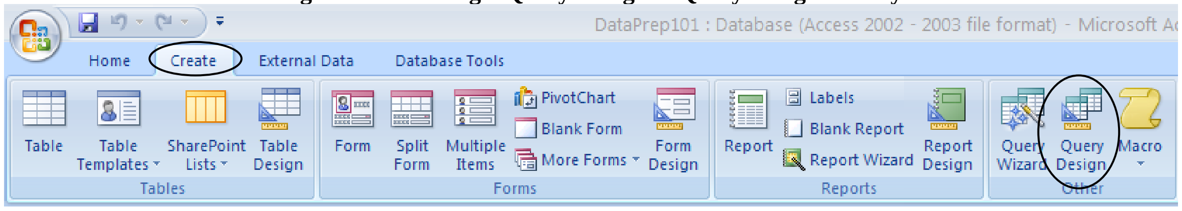
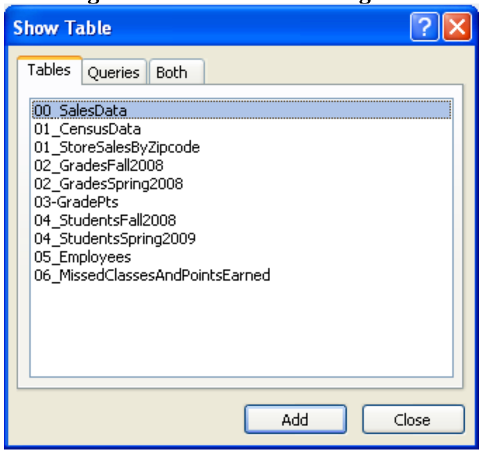


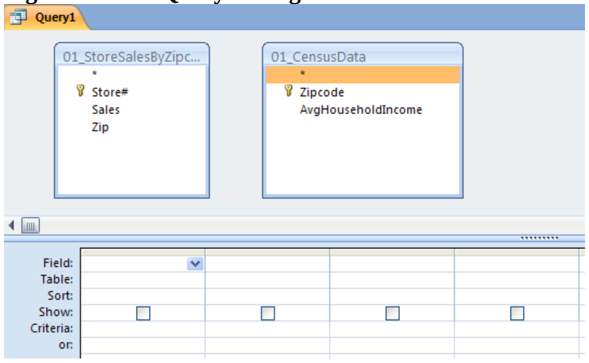
Figure 16: Show Table Dialog Box



00_SalesData
01_CensusData
01_StoreSalesByZipcode
02_GradesFall2008
02_GradesSpring2008
03-GradePts
04_StudentsFall2008
04_StudentsSpring2009
05_Employees
06_MissedClassesAndPointsEarned

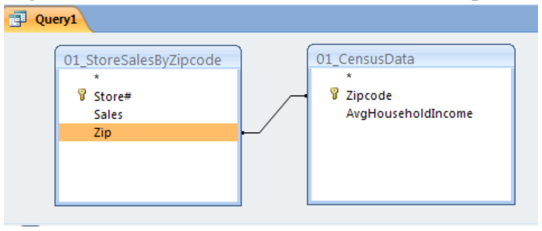
3. In the **Show Table** dialog box, select **01_StoreSalesByZipcode** table; click **Add**. Next, select the **01_CensusData** table; click **Add**. Click **Close** to close the dialog box. Your screen should now look like Figure 17.
4. The next step is to JOIN the two tables on the common field of zip code. This JOIN is imperative to the goal of creating one dataset. Click and drag the **Zip** field from the **01_StoreSalesByZipcode** table to the **Zipcode** field of the **01_CensusData** table. Your screen should now look like Figure 18.

Figure 17: Query Design View with Both Tables



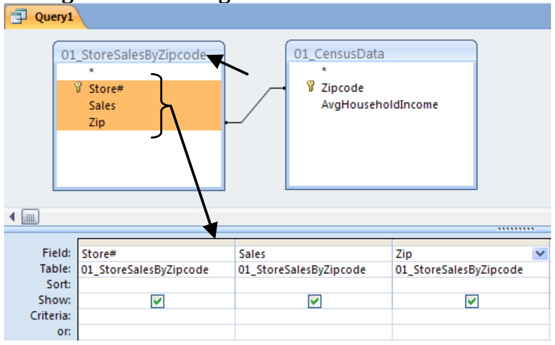
Field:				
Table:				
Sort:				
Show:	☐	☐	☐	☐
Criteria:				
or:				

Figure 18: Both Tables Joined on Zip Code



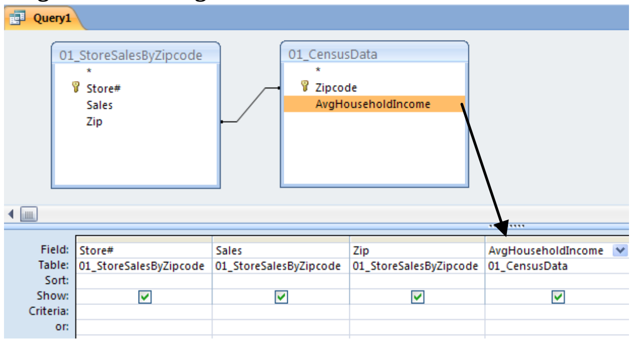
5. Now the fields from the tables need to be brought down to the field criteria grid. Double click the title bar of the **01_StoreSalesByZipcode** table box. Click and drag the highlighted fields from the table box down to the field criteria grid (see Figure 19).
6. From the **01_CensusData** table box, click and drag the **AvgHouseholdIncome** field to the field criteria grid. The **Zipcode** field does not need to be brought down, as it would only duplicate the zip codes in two columns. Your screen should now look like Figure 20.

Figure 19: Moving Fields to Field Criteria Grid



Field:	Store#	Sales	Zip
Table:	01_StoreSalesByZipcode	01_StoreSalesByZipcode	01_StoreSalesByZipcode
Sort:			
Show:	✓	✓	✓
Criteria:			
or:			

Figure 20: Moving Fields to Field Criteria Grid Part 2



Field:	Store#	Sales	Zip	AvgHouseholdIncome
Table:	01_StoreSalesByZipcode	01_StoreSalesByZipcode	01_StoreSalesByZipcode	01_CensusData
Sort:				
Show:	✓	✓	✓	✓
Criteria:				
or:				

7. The query is ready to be run. To accomplish this, click the **Run (!)** button on the toolbar (see Figure 21). Figure 22 shows the query results with data from both of the tables. ***Phase III*** will demonstrate how to import this new dataset into **SPSS**.

Figure 21: The Query Run Button

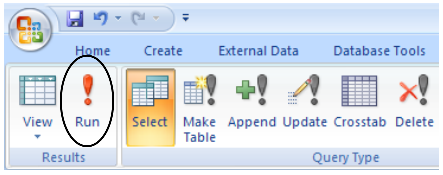
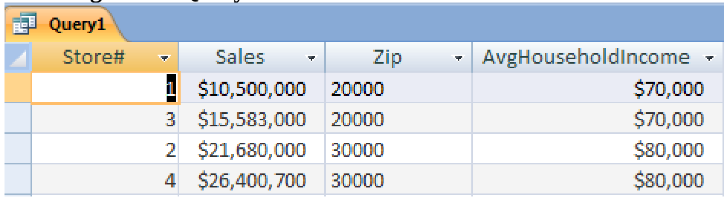


Figure 22: Query Results with Data from Both Tables

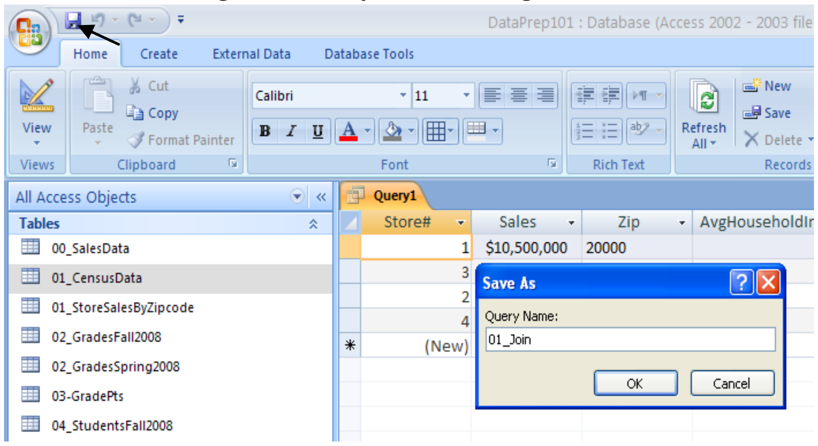


Query1

Store#	Sales	Zip	AvgHouseholdIncome
1	$10,500,000	20000	$70,000
3	$15,583,000	20000	$70,000
2	$21,680,000	30000	$80,000
4	$26,400,700	30000	$80,000

8. Click the **Save** button (see Figure 23). The **Save As** dialog box appears. Name this new query: **01_Join**. Click **OK**. The new query now shows up on the query list on the left-hand side of the screen (see Figure 24).

Figure 23: Query Save As Dialog Box



DataPrep101 : Database (Access 2002 - 2003 file

All Access Objects

Tables

- 00_SalesData
- 01_CensusData
- 01_StoreSalesByZipcode
- 02_GradesFall2008
- 02_GradesSpring2008
- 03-GradePts
- 04_StudentsFall2008

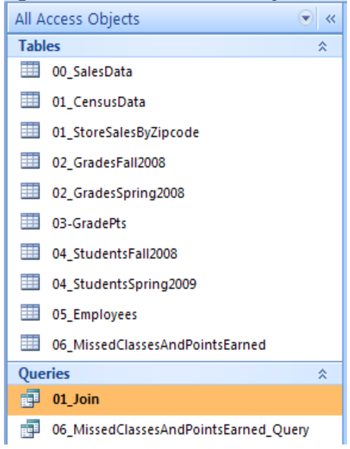
Query1

Store#	Sales	Zip	AvgHouseholdIr
1	$10,500,000	20000	
3			
2			
4			
(New)			

Save As

Query Name: 01_Join

Figure 24: 01_Join in Query List



All Access Objects

Tables

- 00_SalesData
- 01_CensusData
- 01_StoreSalesByZipcode
- 02_GradesFall2008
- 02_GradesSpring2008
- 03-GradePts
- 04_StudentsFall2008
- 04_StudentsSpring2009
- 05_Employees
- 06_MissedClassesAndPointsEarned

Queries

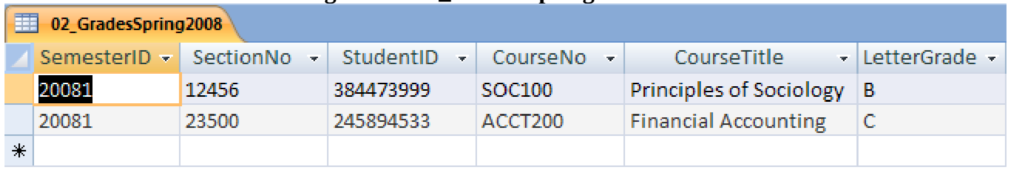
- 01_Join
- 06_MissedClassesAndPointsEarned_Query

9. Part A is now complete. The reader may now continue on to part B or close the **DataPrep101.mdb** file.

B. *Adding the Rows in One Table into Another Table (using an APPEND query).*

1. Open the **DataPrep101.mdb** file. Be sure to turn off the security features (see Figures 3 & 4). The following tables will be used for this example: (1) **02_GradesSpring2008** and (2) **02_GradesFall2008**. Figures 25 and 26 show the datasets for these two tables. Notice that the two tables have all fields matching, however they are not in the same order, nor do all of the column headings match. The goal will be to have a one new table (named: **02_GradesAll**) that includes all four records for 2008.

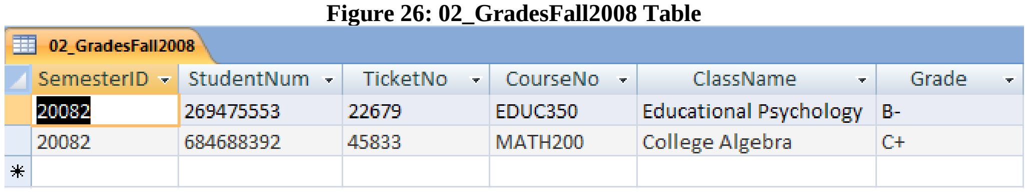
Figure 25: 02_GradesSpring2008 Table



02_GradesSpring2008

SemesterID	SectionNo	StudentID	CourseNo	CourseTitle	LetterGrade
20081	12456	384473999	SOC100	Principles of Sociology	B
20081	23500	245894533	ACCT200	Financial Accounting	C

Figure 26: 02_GradesFall2008 Table



02_GradesFall2008

SemesterID	StudentNum	TicketNo	CourseNo	ClassName	Grade
20082	269475553	22679	EDUC350	Educational Psychology	B-
20082	684688392	45833	MATH200	College Algebra	C+

2. Let's say that we want to be able to create the complete list of grades while leaving both of the original semester tables intact. We can accomplish this by first copying one of the two tables, say Spring, and then inserting duplicates of the fall records into the new table. Here's how it goes: The first step is to copy the **02_GradesSpring2008** table. On the left-hand side of the screen, under the Tables list, locate the **02_GradesSpring2008** table name (see Figure 27). Right click on it to make the shortcut menu appear. Choose **Copy** from the list.
3. Click the **Paste** button (see Figure 28), the **Paste Table As** dialog box appears. Name the new table: **02_GradesAll**, then click **OK**. The **02_GradesAll** table now shows up on the **Tables** list on the left-hand side of the screen. Remember, this table only includes the two records from the **02_GradesSpring2008** table. The next step is to add the two records from the **02_GradesFall2008**.

Figure 27: Copy 02_GradesSpring2008 Table

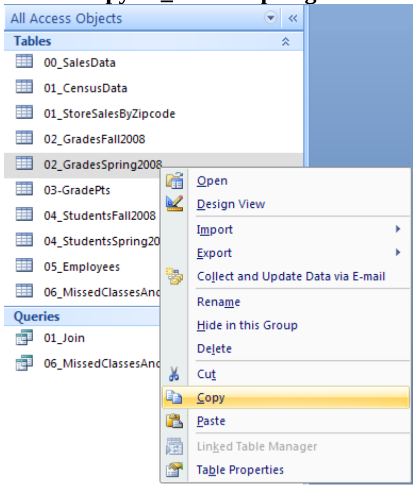


Figure 28: Paste Table As 02_GradesAll

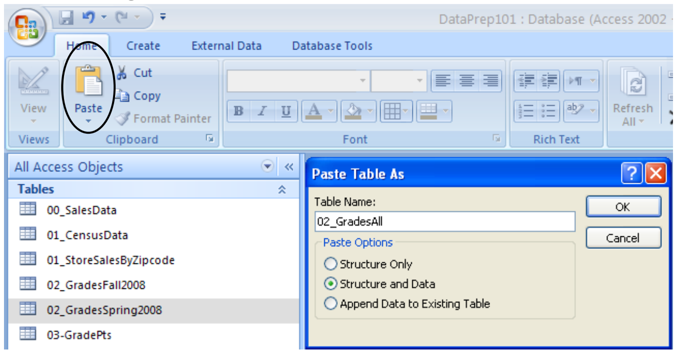


Table Name: 02_GradesAll
Paste Options: ( ) Structure Only [✓ Structure and Data] ( ) Append Data to Existing Table

4. An APPEND query will be used to achieve this goal. Click the **Create** tab, then click on the **Query Design** button on the toolbar (see Figure 15). The **Show Table** dialog box appears, click **Close**. Click the Append button on the toolbar (see Figure 29). The Append dialog box appears. Choose the **02_GradesAll** table as the file to append (or add records to), click **OK**.

Figure 29: Append to 02_GradesAll Table

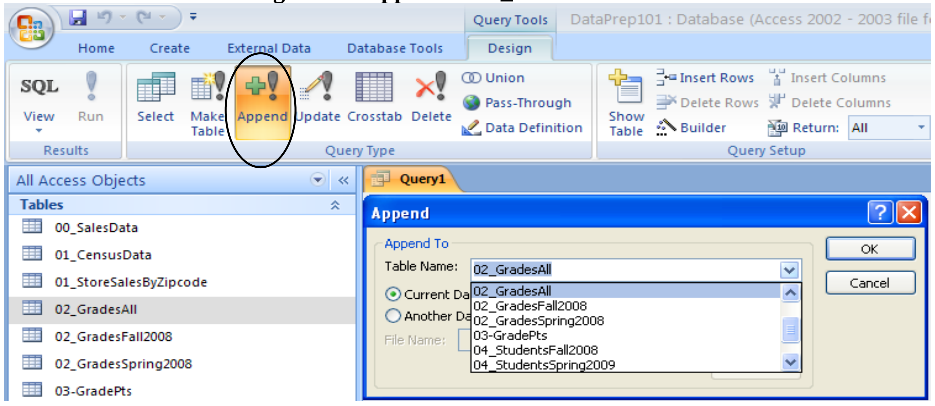


Table Name: 02_GradesAll

5. Click the **Show Table** button on the toolbar (see Figure 30); the **Show Table** dialog box appears. Highlight the **02_GradesFall2008** table from the list, click **Add**.
6. The next step is to add the fields from the **02_GradesFall2008** table box to the field criteria grid (see Figure 31). Double click the title bar of the **02_GradesFall2008** table box. Click and drag

the highlighted fields from the table box down to the field criteria grid. Notice that only the **SemesterID** and **CourseNo** fields are currently matching up with the fields in the **02_GradesAll** table because the fields were not named consistently for the two semesters. Now the remaining four fields in **02_GradesFall2008** need to be manually matched up with the corresponding field names in **02_GradesAll**.

7. On the **Append To** row (see Figure 32), click the cell under **StudentNum**, click the **Down Arrow** and choose **StudentID** from the list. Continue with the three remaining field pair (**TicketNo** =**SectionNo**, **ClassName** = **CourseTitle**, and **Grade** = **LetterGrade**).

Figure 30: Show Table Dialog Box—Select 02_GradesFall2008

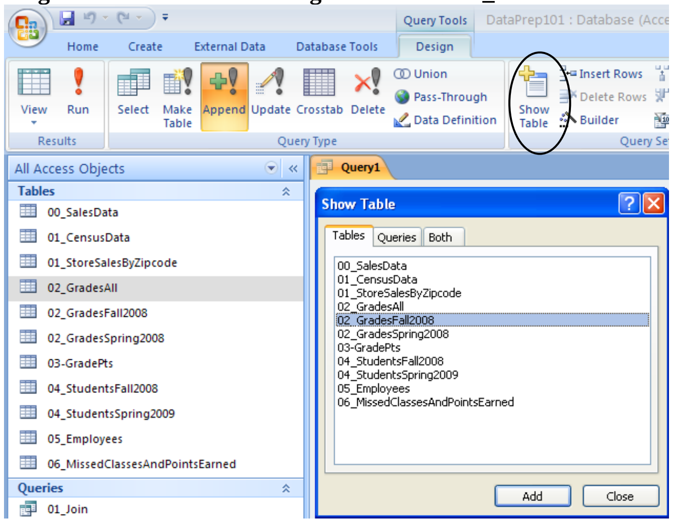
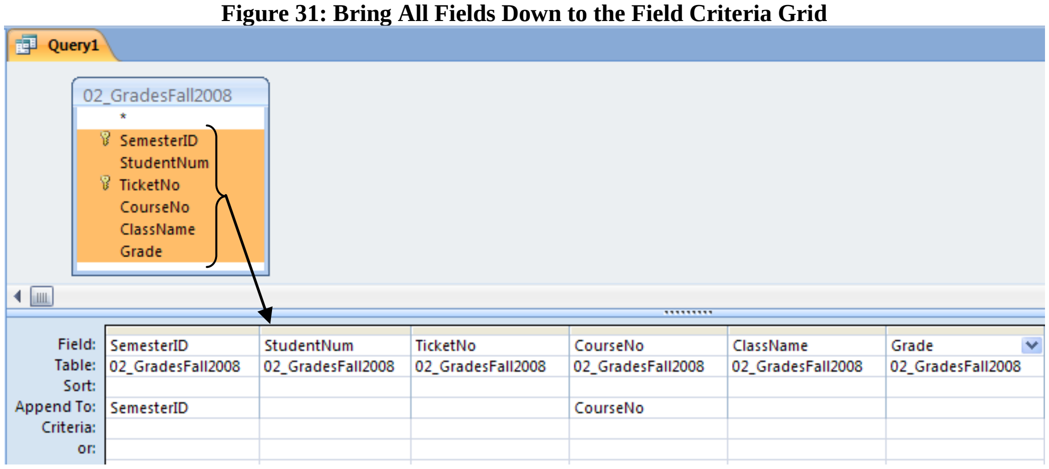
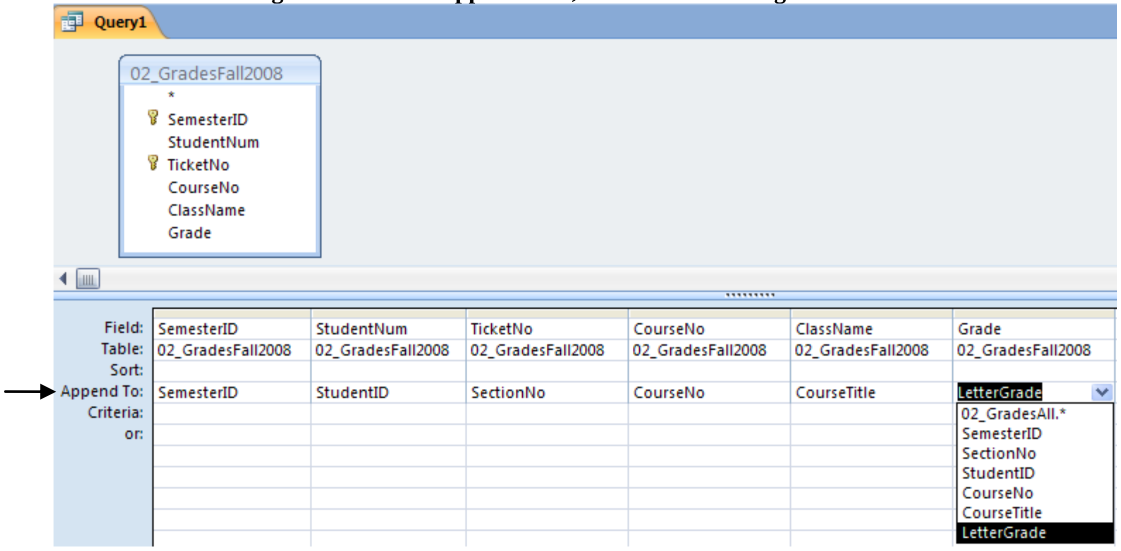


Figure 31: Bring All Fields Down to the Field Criteria Grid



Field:	SemesterID	StudentNum	TicketNo	CourseNo	ClassName	Grade
Table:	02_GradesFall2008	02_GradesFall2008	02_GradesFall2008	02_GradesFall2008	02_GradesFall2008	02_GradesFall2008
Sort:						
Append To:	SemesterID			CourseNo		
Criteria:						
or:						

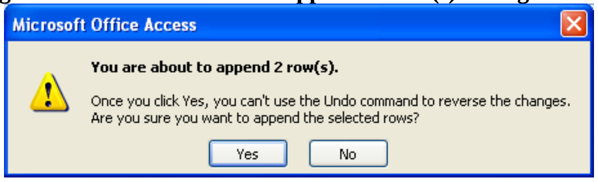
Figure 32: On the Append Row, Select the Matching Field Names



Field:	SemesterID	StudentNum	TicketNo	CourseNo	ClassName	Grade
Table:	02_GradesFall2008	02_GradesFall2008	02_GradesFall2008	02_GradesFall2008	02_GradesFall2008	02_GradesFall2008
Sort:						
Append To:	SemesterID	StudentID	SectionNo	CourseNo	CourseTitle	LetterGrade
Criteria:						
or:						

8. The final step is to run this APPEND query. Click **Run (!)**. A dialog box appears warning that "You are about to append 2 rows (see Figure 33). Since this is what we want to do, click **Yes**.
9. Congratulations, you have successfully combined the two data sets. Click the **Save** button and save this query as **02_CombiningDataSets** (see Figure 34). Click **OK**.

Figure 33: You Are About to Append 2 Row(s) Dialog Box

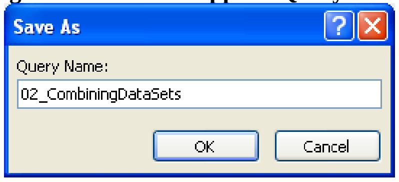


Microsoft Office Access

You are about to append 2 row(s).

Once you click Yes, you can't use the Undo command to reverse the changes.
Are you sure you want to append the selected rows?

Figure 34: Save the Append Query

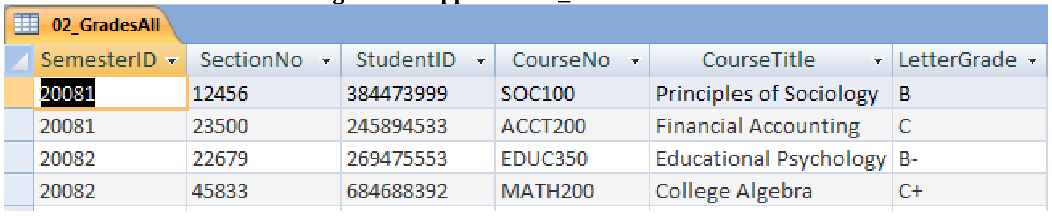


Save As

Query Name: 02_CombiningDataSets

10. The **02_GradesAll** table now includes all four grade records from Spring and Fall 2008 semesters (see Figure 35).

Figure 35: Appended 02_GradesAll Table



02_GradesAll

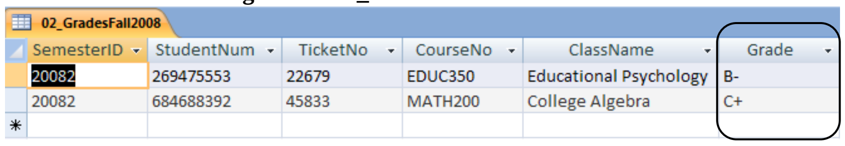
SemesterID	SectionNo	StudentID	CourseNo	CourseTitle	LetterGrade
20081	12456	384473999	SOC100	Principles of Sociology	B
20081	23500	245894533	ACCT200	Financial Accounting	C
20082	22679	269475553	EDUC350	Educational Psychology	B-
20082	45833	684688392	MATH200	College Algebra	C+

11. Part B is now complete. The reader may now continue on to part C or close the **DataPrep101.mdb** file.

C. Recoding a Column in One Table (using data stored in a second CONVERSION table).

1. Open the **DataPrep101.mdb** file. Be sure to turn off the security features (see Figures 3 & 4). The following tables will be used for this example: (1) **02_GradesFall2008** and (2) **03_GradePts**. Figures 36 and 37 show the datasets for these two tables. Notice that the **02_GradesFall2008** table has a **Grade** column and the **03_GradePt** table includes both a **LetterGrade** and a **GradePt** field. Our goal here will be to create one new dataset that contains all of the fields from the **02_GradesFall2008** table and also includes the **GradePt** column.

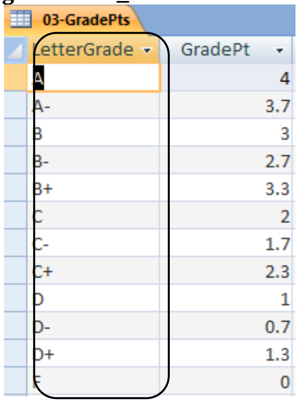
Figure 36: 02_GradesFall 2008 Table



02_GradesFall2008

SemesterID	StudentNum	TicketNo	CourseNo	ClassName	Grade
20082	269475553	22679	EDUC350	Educational Psychology	B-
20082	684688392	45833	MATH200	College Algebra	C+
*					

Figure 37: 03_GradePts Table

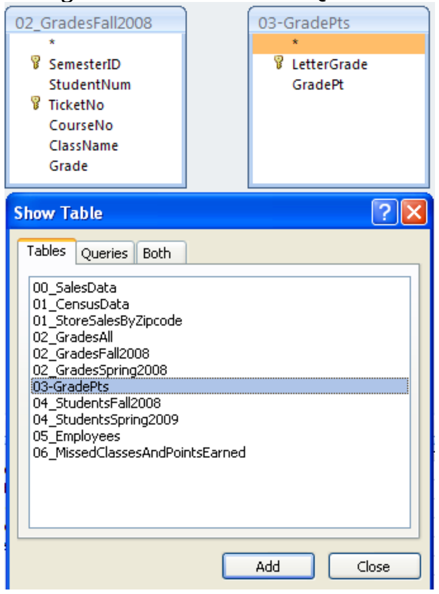


03-GradePts

LetterGrade	GradePt
A	4
A-	3.7
B	3
B-	2.7
B+	3.3
C	2
C-	1.7
C+	2.3
D	1
D-	0.7
D+	1.3
F	0

2. Click the **Create** tab, then click on the **Query Design** button on the toolbar (see Figure 15). The **Show Table** dialog box appears.
3. Choose the **02_GradesFall2008** table from the list; click **Add** (see Figure 38). Next, choose the **03_GradePts** table from the list; Click **Add**. Click **Close**.
4. The next step is to JOIN the two tables on the common field of Grade/LetterGrade (see Figure 39). To do this, click and drag the **Grade** field from the **02_GradesFall2008** table to the **LetterGrade** field of the **03_GradePts** table.

Figure 38: Add Tables to QBE



02_GradesFall2008
*
SemesterID
StudentNum
TicketNo
CourseNo
ClassName
Grade

03-GradePts
*
LetterGrade
GradePt

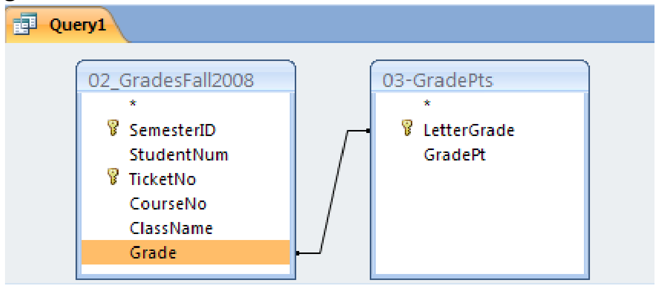
Show Table

Tables | Queries | Both

00_SalesData
01_CensusData
01_StoreSalesByZipcode
02_GradesAll
02_GradesFall2008
02_GradesSpring2008
03-GradePts
04_StudentsFall2008
04_StudentsSpring2009
05_Employees
06_MissedClassesAndPointsEarned

Add | Close

Figure 39: JOIN the Two Tables on Grade/LetterGrade Fields



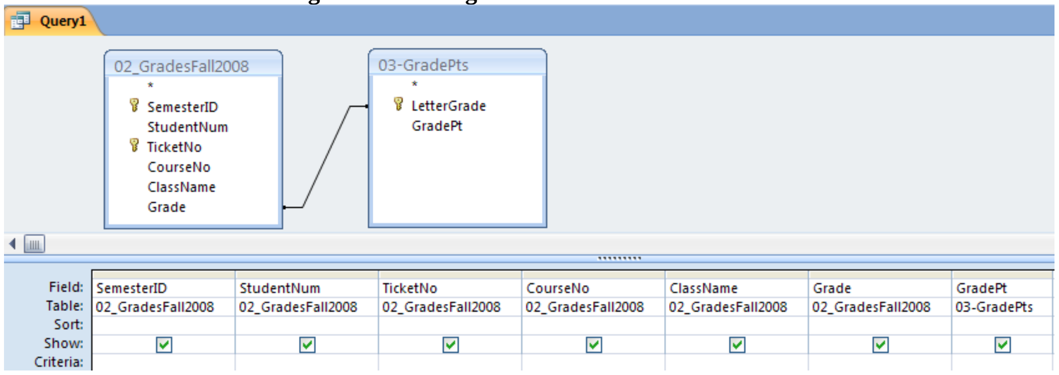
Query1

02_GradesFall2008
*
SemesterID
StudentNum
TicketNo
CourseNo
ClassName
Grade

03-GradePts
*
LetterGrade
GradePt

5. Now the fields from the tables need to be brought down to the field criteria grid. Double click the title bar of the **02_GradesFall2008** table box. Click and drag the highlighted fields from the table box down to the field criteria grid.
6. From the **03_GradePts** table box, click and drag the **GradePt** field to the field criteria grid. The **LetterGrade** field does not need to be brought down, as it would only duplicate the **Grade** field from the other table. Your screen should now look like Figure 40.

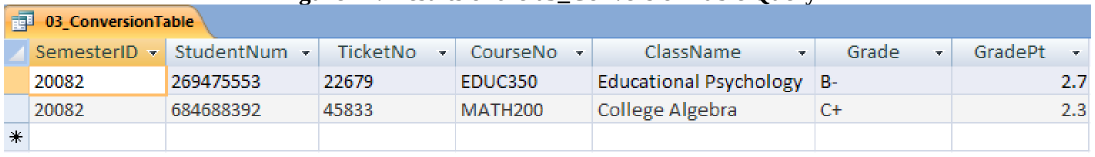
Figure 40: Moving Fields to Field Criteria Grid



Field:	SemesterID	StudentNum	TicketNo	CourseNo	ClassName	Grade	GradePt
Table:	02_GradesFall2008	02_GradesFall2008	02_GradesFall2008	02_GradesFall2008	02_GradesFall2008	02_GradesFall2008	03-GradePts
Sort:							
Show:	✓	✓	✓	✓	✓	✓	✓
Criteria:							

7. The final step is to run this query. Click **Run (!)**. Congratulations, you have successfully used a CONVERSION table to combine the two datasets. The results of the query are in Figure 41. Click the **Save** button and save this query as **03_ConversionTable.** Click **OK**.

Figure 41: Results of the 03_ConversionTable Query



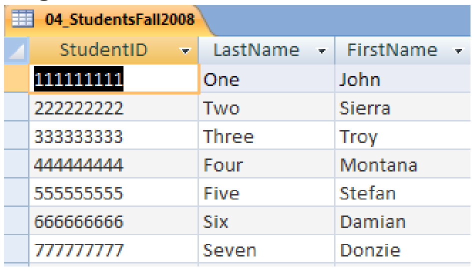
SemesterID	StudentNum	TicketNo	CourseNo	ClassName	Grade	GradePt
20082	269475553	22679	EDUC350	Educational Psychology	B-	2.7
20082	684688392	45833	MATH200	College Algebra	C+	2.3
*						

8. Part C is now complete. The reader may now continue on to part D or close the **DataPrep101.mdb** file.

D. *Dealing with Tables Having Overlapping Data (using UNMATCHED and APPEND queries).*

1. Open the **DataPrep101.mdb** file. Be sure to turn off the security features (see Figures 3 & 4). The following tables will be used for this example: (1) **04_StudentsFall2008** and (2) **04_StudentsSpring2009**. Figures 42 and 43 show the datasets for these two tables. Notice that the two tables have exactly the same fields, however some of the rows are overlapping (i.e., showing in both tables). The ultimate goal is to have one complete list of unique students in a table named **04_StudentsAll**.

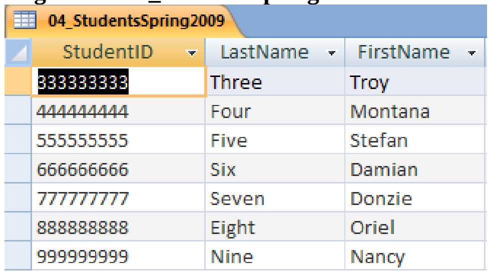
Figure 42: 04_StudentsFall2008 Dataset



04_StudentsFall2008

StudentID	LastName	FirstName
111111111	One	John
222222222	Two	Sierra
333333333	Three	Troy
444444444	Four	Montana
555555555	Five	Stefan
666666666	Six	Damian
777777777	Seven	Donzie

Figure 43: 04_StudentsSpring2009 Dataset



04_StudentsSpring2009

StudentID	LastName	FirstName
333333333	Three	Troy
444444444	Four	Montana
555555555	Five	Stefan
666666666	Six	Damian
777777777	Seven	Donzie
888888888	Eight	Oriel
999999999	Nine	Nancy

2. The first step is to copy the **04_StudentsFall2008** table. On the left-hand side of the screen, under the **Tables** list, locate the **04_StudentsFall2008** table name (see Figure 44). Right click on it to make the shortcut menu appear. Choose **Copy** from the list.
3. Click the **Paste** button (see Figure 45); the **Paste Table As** dialog box appears. Name the new table: **04_StudentsAll**, then click **OK**. The **04_StudentsAll** table now shows up on the **Tables** list on the left-hand side of the screen. Remember, so far this new table only includes the records from the **04_StudentsFall2008** table. The next step is to find the unmatched records from the **04_StudentsSpring2009** table (LastNames Eight and Nine).

Figure 44: Copy 04_StudentsFall2008 Table

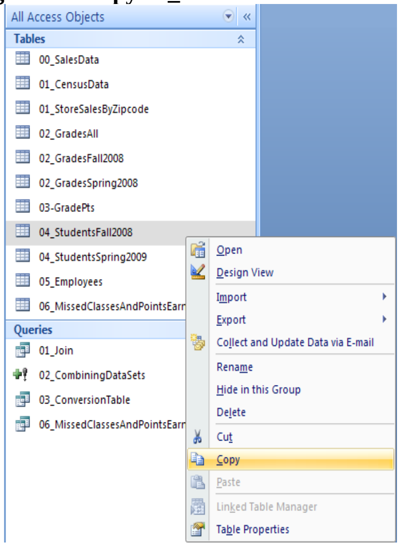
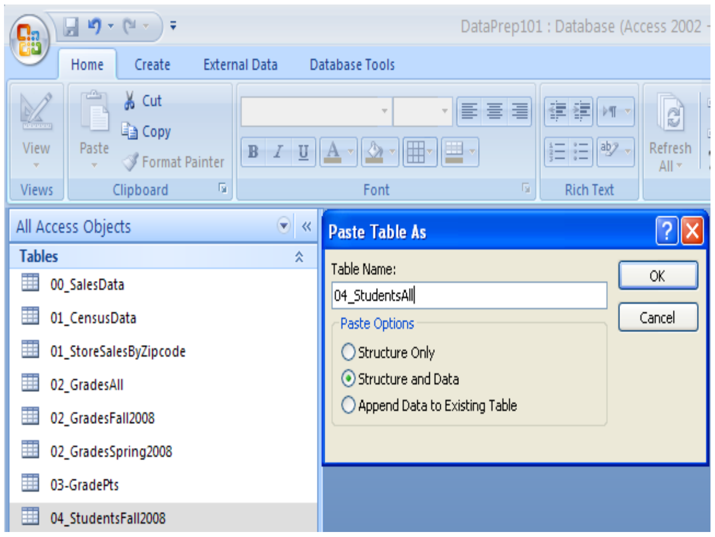


Figure 45: Paste Table As 04_StudentsAll



Paste Table As

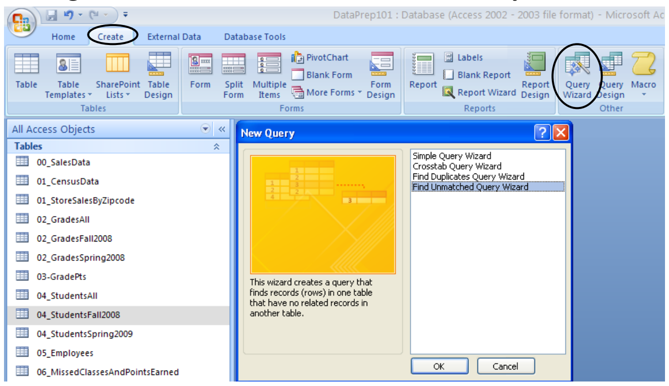
Table Name: 04_StudentsAll

Paste Options

Structure Only [✓ Structure and Data] Append Data to Existing Table

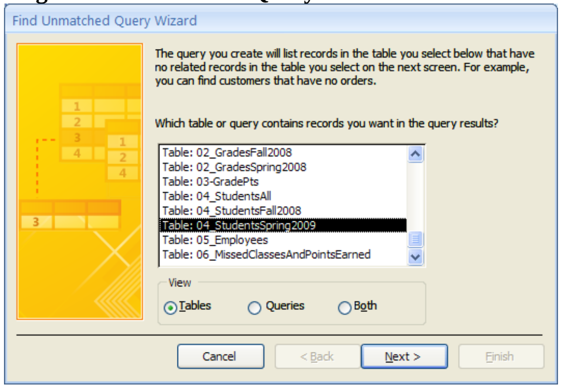
4. Click the **Create** button then click the **Query Wizard** button, the **New Query** dialog box appears (see Figure 46). Select **Find Unmatched Query Wizard** from the list; click **OK**. The next step is to choose the table with the new data (in this case **04_StudentsSpring2009**).
5. In the first dialog box of the **Find Unmatched Query Wizard**, select the **04_StudentsSpring2009** table from the list**,** click **Next** (see Figure 47). The next step is to choose the table that contains the related records (in this case, **04_StudentsAll**).

Figure 46: Launch the Find Unmatched Query Wizard



This wizard creates a query that finds records (rows) in one table that have no related records in another table.

Figure 47: Unmatched Query Wizard-Screen 1


Find Unmatched Query Wizard
Table: 02_GradesFall2008
Table: 02_GradesSpring2008
Table: 03-GradePts
Table: 04_StudentsAll
Table: 04_StudentsFall2008
Table: 04_StudentsSpring2009
Table: 05_Employees
Table: 06_MissedClassesAndPointsEarned
View
Tables
Queries
Both
Cancel
< Back
Next >
Finish

The query you create will list records in the table you select below that have no related records in the table you select on the next screen. For example, you can find customers that have no orders.

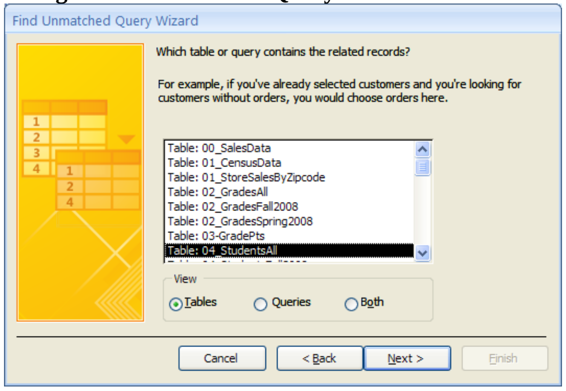
Which table or query contains records you want in the query results?

Table: 04_StudentsSpring2009

View: [✓ Tables] Queries Both

6. In the second dialog box, select the **04_StudentsAll** table from the list; click **Next** (see Figure 48). The next step is to identify which fields are matching in both tables.
7. In the third dialog box, make sure that the **StudentID** field is selected on both sides (indicating that the records in the two tables should be compared on the basis of this key field); click **Next** (see Figure 49). The next step is to identify which fields should show in the query results.

Figure 48: Unmatched Query Wizard-Screen 2



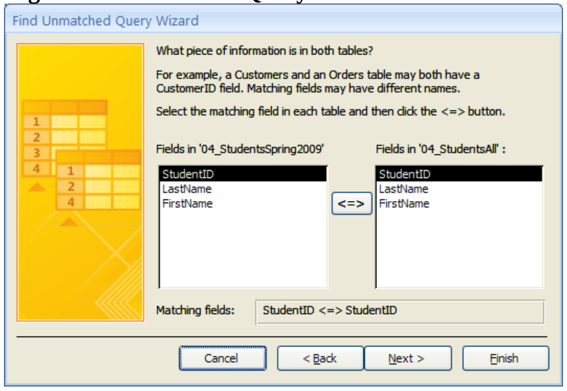
Which table or query contains the related records?

For example, if you've already selected customers and you're looking for customers without orders, you would choose orders here.

Table: 04_StudentsAll

View: [✓ Tables] Queries Both

Figure 49: Unmatched Query Wizard-Screen 3



What piece of information is in both tables?

For example, a Customers and an Orders table may both have a CustomerID field. Matching fields may have different names.

Select the matching field in each table and then click the <=> button.

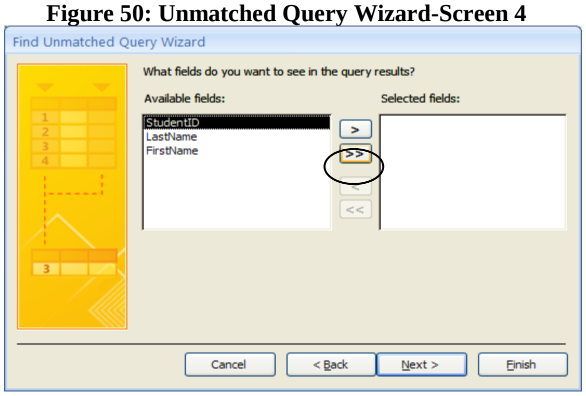
Fields in '04_StudentsSpring2009': StudentID

Fields in '04_StudentsAll' : StudentID

Matching fields: StudentID <=> StudentID

8. In the fourth dialog box, move all three fields (**StudentID**, **LastName**, and **FirstName**), from the left to the right, by clicking the >> button; click **Next** (see Figure 50).
9. This query must be named. In the next dialog box, type in **04_Spring2009NewStudents**; click **Finish** (see Figure 51). The query results are now showing on the screen (see Figure 52). Notice that only the unmatched records (i.e., the new students for Spring 2009) are in the list.

Figure 50: Unmatched Query Wizard-Screen 4

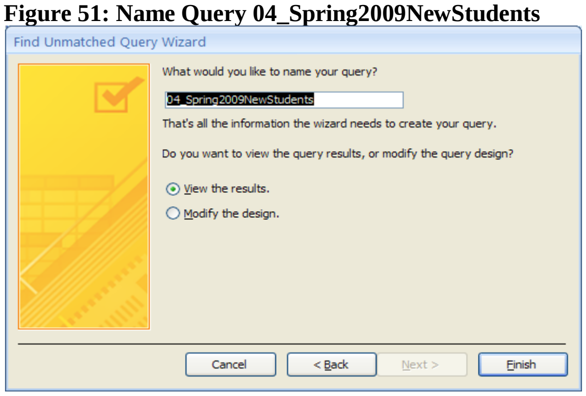


What fields do you want to see in the query results?

Available fields:

Selected fields:

Figure 51: Name Query 04_Spring2009NewStudents



What would you like to name your query?

04_Spring2009NewStudents

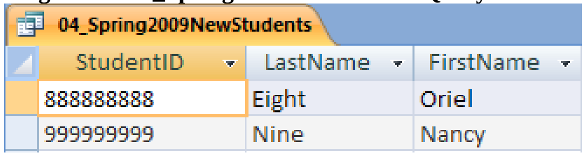
That's all the information the wizard needs to create your query.

Do you want to view the query results, or modify the query design?

[✓ View the results.]

Modify the design.

Figure 52: 04_Spring2009NewStudents Query Results

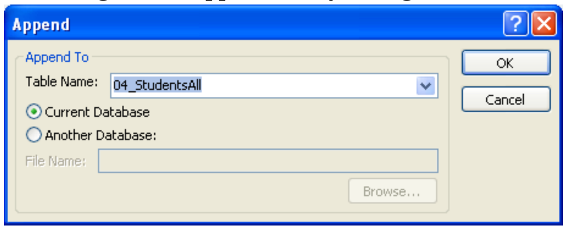


04_Spring2009NewStudents

StudentID	LastName	FirstName
888888888	Eight	Oriel
999999999	Nine	Nancy

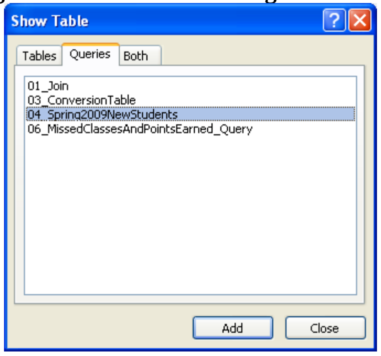
12. Now that the unmatched records have been identified, the next step is to APPEND (or add) these two records to the **04_StudentsAll** table. Click the **Create** tab, then click on the **Query Design** button on the toolbar (see Figure 15). The **Show Table** dialog box appears, click **Close**. Click the **Append** button on the toolbar (see Figure 29). The **Append Query** dialog box appears. Choose the **04_StudentsAll** table as the file to append (or add records to); click **OK** (see Figure 53).
13. Click the **Show Table** button on the toolbar. The **Show Table** dialog box appears. Click on the **Queries** tab and select the **04_Spring2009NewStudents** query from the list (see Figure 54). Click **Add**. Click **Close**. Your screen should now look similar to Figure 55.

Figure 53: Append Query Dialog Box



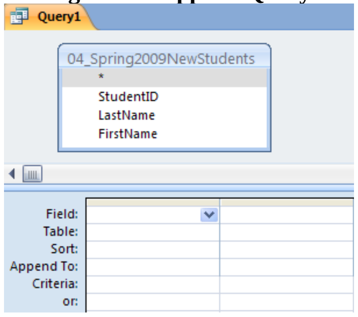
Append To
Table Name: 04_StudentsAll
(●) Current Database
( ) Another Database:
File Name:

Figure 54: Show Table Dialog Box



01_Join
03_ConversionTable
04_Spring2009NewStudents
06_MissedClassesAndPointsEarned_Query

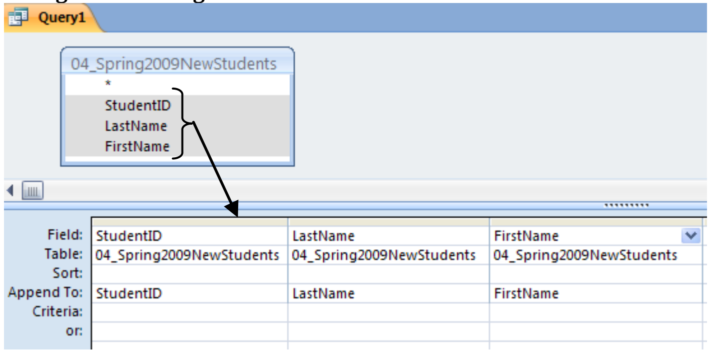
Figure 55: Append Query



14. The next step is to add the fields from the **04_Spring2009NewStudents** query box at the top of the screen to the field criteria grid at the bottom (see Figure 56). Double click the title bar of the **04_Spring2009NewStudents** query box. Click and drag the highlighted fields from the query box down to the field criteria grid.
15. The next step would normally be to select the matching fields on the **Append To** row. Since the three field names are identical in the **04_Spring2009NewStudents** query and the **04_StudentsAll** table, they already match.
16. The final step is to run this APPEND query. Click **Run (!)**. A dialog box appears warning that "You are about to append 2 rows." Click **Yes**.
17. Congratulations, you have successfully combined the two data sets. Click the **Save** button and save this query as **04_2008And2009Students**. Click **OK**.

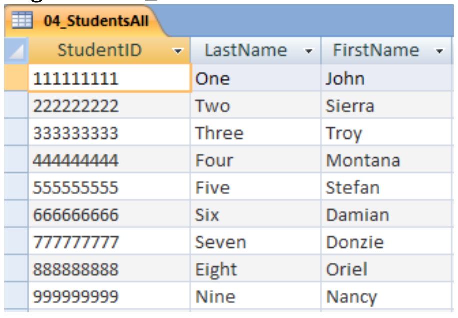
18. If you view the **04_StudentsAll** table, it will now contain the list of nine unique students from both semesters (See Figure 57).

Figure 56: Bring All Fields Down to the Field Criteria Grid



Field:	StudentID	LastName	FirstName
Table:	04_Spring2009NewStudents	04_Spring2009NewStudents	04_Spring2009NewStudents
Sort:			
Append To:	StudentID	LastName	FirstName
Criteria:			
or:			

Figure 57: 04_StudentsAll Dataset



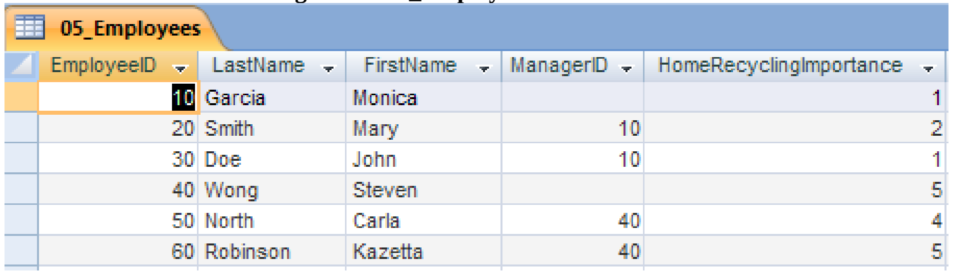
StudentID	LastName	FirstName
111111111	One	John
222222222	Two	Sierra
333333333	Three	Troy
444444444	Four	Montana
555555555	Five	Stefan
666666666	Six	Damian
777777777	Seven	Donzie
888888888	Eight	Oriel
999999999	Nine	Nancy

19. Part D is now complete. The reader may now continue on to part E or close the **DataPrep101.mdb** file.

E. *Allowing the Matching of Values in One Column that Reference the Values of Another Column in the Same Table (using a SELF-JOIN query).*

1. Open the **DataPrep101.mdb** file. Be sure to turn off the security features (see Figures 3 & 4). The following table will be used for this example: **05_Employees** (see Figure 58). Notice that the **ManagerID** is actually the **EmployeeID** of the manager. The statistical goal is to see if the employee's opinion of **HomeRecyclingImportance** is influenced by (i.e., correlated with) the opinion of their manager. To this requires a dataset that contains the columns: **EmployeeID**, **EmpOpinion**, **ManagerID**, and **MgrOpinion**. Of course, the reader will correctly note that the **05_Employees** table only contains the **HomeReyclingImportance** opinion of the Employee, not the Manager. But, since managers are, themselves employees, their opinions on recycling are also stored in the table. For example, Mary Smith's (Employee #20) Rating is 2 and her manager's (Monica Garcia, Manager #10) rating is 1. The process for extracting data from a table in this way involves making a virtual copy of the original **05_Employees** table and then linking the copy to the original and is called a SELF JOIN.

Figure 58: 05_Employees Table Dataset



EmployeeID	LastName	FirstName	ManagerID	HomeRecyclingImportance
10	Garcia	Monica		1
20	Smith	Mary	10	2
30	Doe	John	10	1
40	Wong	Steven		5
50	North	Carla	40	4
60	Robinson	Kazetta	40	5

2. To start, click the **Create** tab, then click on the **Query Design** button on the toolbar (see Figure 15). The **Show Table** dialog box appears. Select the **05_Employees** table from the list. Click **Add**, then click **Add** again. You should now have two **05_Employees** table boxes showing (see Figure 59).

3. The next step is to JOIN the two copies of the table on the common field. This JOIN is unique, in that **ManagerID** and **EmployeeID** will be the common fields, therefore allowing us to create a SELF JOIN. Click and drag the **ManagerID** field from the **05_Employees** table box to the **EmployeeID** field of the **05_Employees_1** table box. Your screen should now look like Figure 60.

Figure 59: Two 05_Employee Table Boxes

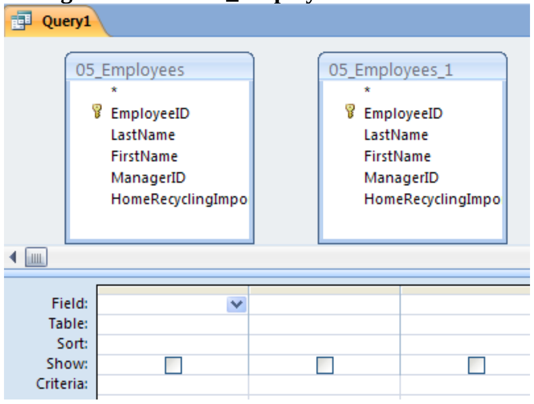
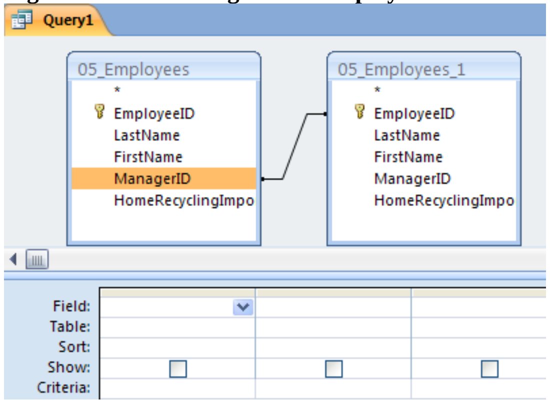


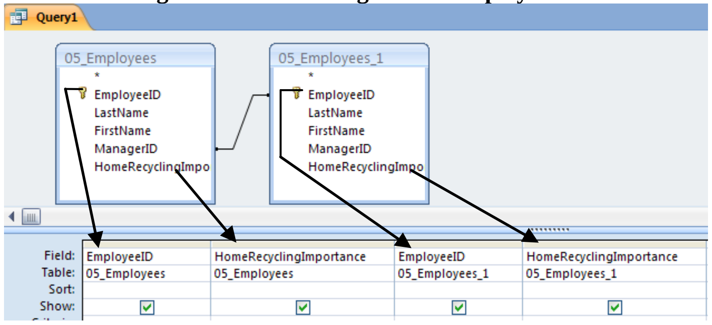
Figure 60: Join ManagerID to EmployeeID



4. The next step is to bring down the required fields to the field criteria grid (see Figure 61). From the **05_Employees** table box double-click the **EmployeeID**, then double-click the **HomeRecyclingImportance**. Next, from the **05_Employees_1** table box double-click the **EmployeeID**, then double-click the **HomeRecyclingImportance**.

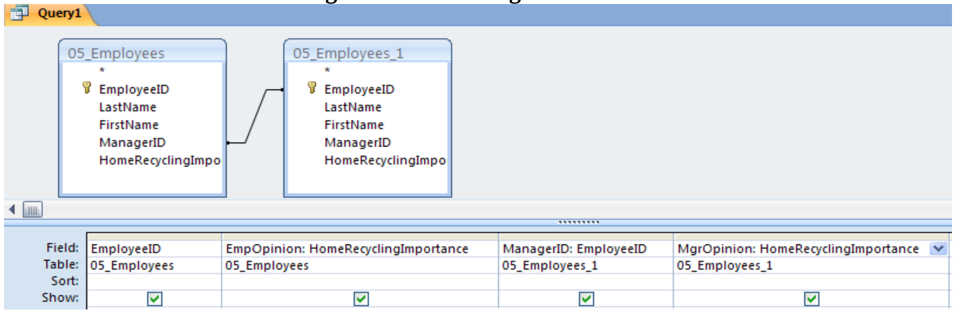
5. The next step is to rename the two **HomeRecyclingImportance** fields to reflect the identity of the opinion and to rename the second **EmployeeID** field to **ManagerID** (see Figure 62). Type in **EmpOpinion:** in front of the first **HomeRecyclingImportance** field. Type in **ManagerID:** in front of the second **EmployeeID**. Finally, type in **MgrOpinion:** in front of the second **HomeRecyclingImportance** field.

Figure 61: Join ManagerID to EmployeeID



Field:	EmployeeID	HomeRecyclingImportance	EmployeeID	HomeRecyclingImportance
Table:	05_Employees	05_Employees	05_Employees_1	05_Employees_1
Sort:				
Show:	✓	✓	✓	✓

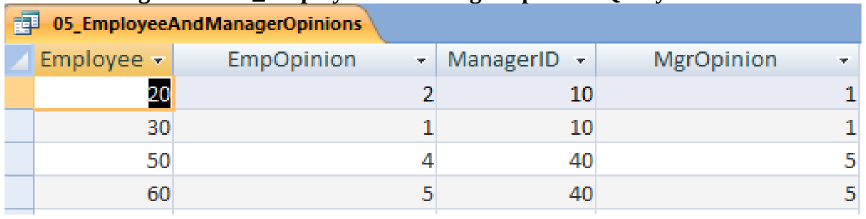
Figure 62: Renaming of the Fields



Field:	EmployeeID	EmpOpinion: HomeRecyclingImportance	ManagerID: EmployeeID	MgrOpinion: HomeRecyclingImportance
Table:	05_Employees	05_Employees	05_Employees_1	05_Employees_1
Sort:				
Show:	✓	✓	✓	✓

6. The final step is to run this query. Click **Run (!)**. Congratulations, you have successfully used a SELF JOIN by using two copies of the same table. The results of the query are in Figure 63. It appears that the opinion of the manager does impact the opinion of the employee. Click the **Save** button and save this query as **05_EmployeeAndManagerOpinions.** Click **OK**.

Figure 63: 05_EmployeeAndManagerOpinions Query Results



05_EmployeeAndManagerOpinions

Employee	EmpOpinion	ManagerID	MgrOpinion
20	2	10	1
30	1	10	1
50	4	40	5
60	5	40	5

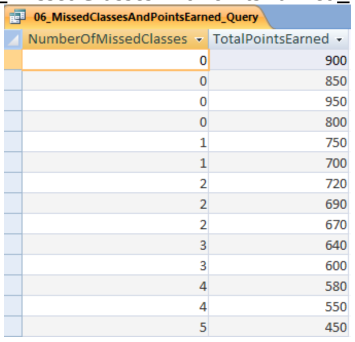
7. **Phase II** is now complete. The reader may now continue on to **Phase III** or close the **DataPrep101.mdb** file.

PHASE III: IMPORTING A RECONFIGURED DATASET FROM MS ACCESS INTO SPSS.

Now that the five common ways of reconfiguring datasets using **QBE** have been covered, this third and final phase of the paper explains how to bring a reconfigured dataset from **MS Access** into **SPSS** for statistical analysis. Specifically, the reader will be importing the **06_MissedClassesAndPointsEarned_Query** dataset from **MS Access** into **SPSS** and run a regression on this data. Although **SPSS** will be used in this example, many other statistical analysis packages (including **MS Excel**, **SAS**, or **Minitab**) could also be used.

*A. The following steps will be used in importing the **06_MissedClassesAndPointsEarned_Query** from **MS Access** into **SPSS.** Figure 64 shows the query results in **MS Access**.*

Figure 64: 06_MissedClassesAndPointsEarned_Query Results



06_MissedClassesAndPointsEarned_Query

NumberOfMissedClasses	TotalPointsEarned
0	900
0	850
0	950
0	800
1	750
1	700
2	720
2	690
2	670
3	640
3	600
4	580
4	550
5	450

1. Launch **SPSS** so that a new data editor window appears (see Figure 65).
2. Before the query may be imported, a "**Microsoft Data Source**" must first be established. This is done by the use of **SPSS's Database Wizard**. To launch the wizard, from the menu select **File**/**Open Database**/**New Query** (see Figure 66).

Figure 65: New SPSS Data Editor Window

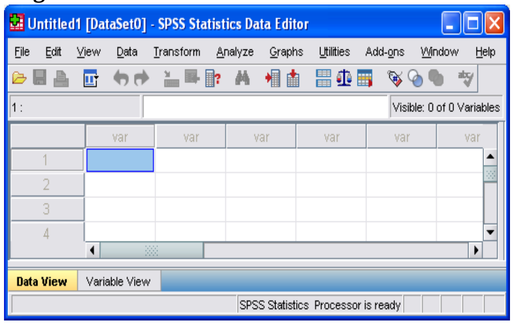
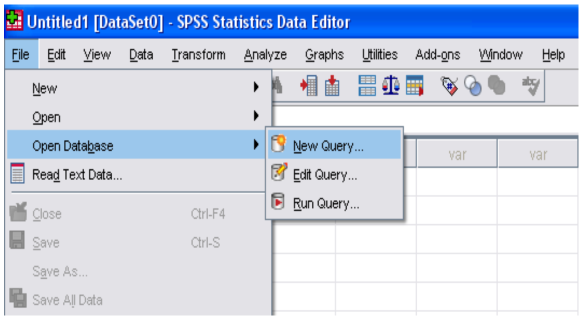
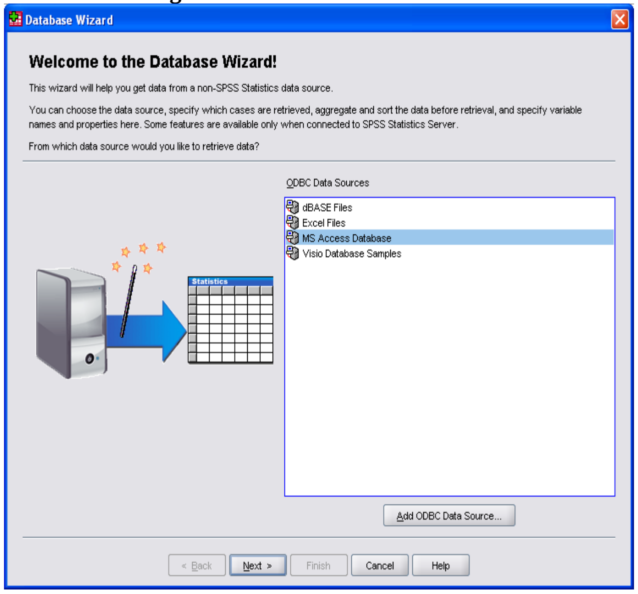

Untitled1 [DataSet0] - SPSS Statistics Data Editor
File Edit View Data Transform Analyze Graphs Utilities Add-ons Window Help
Visible: 0 of 0 Variables
Data View Variable View
SPSS Statistics Processor is ready

Figure 66: Accessing the Database Wizard


Untitled1 [DataSet0] - SPSS Statistics Data Editor
File Edit View Data Transform Analyze Graphs Utilities Add-ons Window Help
New
Open
Open Database
Read Text Data...
Close Ctrl-F4
Save Ctrl-S
Save As...
Save All Data
New Query...
Edit Query...
Run Query...

3. The **Database Wizard** appears. Select **MS Access Database** from the list; click **Next** (see Figure 67).
4. The **ODBC Driver Login** window appears. The next step is to locate the **DataPrep101.mdb** file. Click the **Browse** button; locate the file; click **OK** (see Figure 68).

Figure 67: Database Wizard



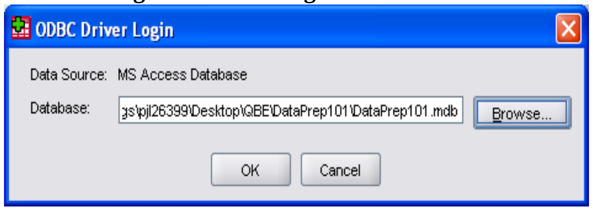
Welcome to the Database Wizard!

This wizard will help you get data from a non-SPSS Statistics data source.

You can choose the data source, specify which cases are retrieved, aggregate and sort the data before retrieval, and specify variable names and properties here. Some features are available only when connected to SPSS Statistics Server.

From which data source would you like to retrieve data?

Figure 68: Selecting the Database

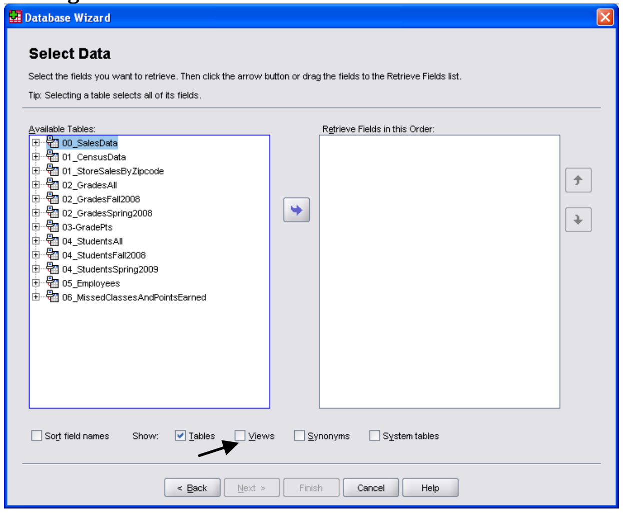


Data Source: MS Access Database

Database: gs\pjl26399\Desktop\QBE\DataPrep101\DataPrep101.mdb

5. The **Database Wizard-Select Data** window appears (see Figure 69). Notice that all of the tables are showing on the left-hand side of the window, however, none of the queries are visible. Click the **Checkbox** next to **Views** and the queries are also included in the list. (see Figure 70).
6. Select the **06_MissedClassesAndPointsEarned_Query** from the list, click the to move the two fields from the query to the right-hand side of the screen. Your screen should now look like Figure 71.

Figure 69: Database Wizard-Select Data-Screen 1



Database Wizard

Select Data

Select the fields you want to retrieve. Then click the arrow button or drag the fields to the Retrieve Fields list.

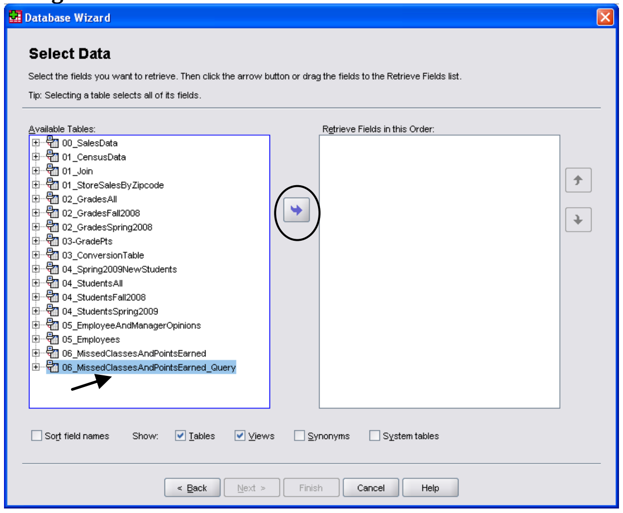
Tip: Selecting a table selects all of its fields.

Available Tables:
- 00_SalesData
- 01_CensusData
- 01_StoreSalesByZipcode
- 02_GradesAll
- 02_GradesFall2008
- 02_GradesSpring2008
- 03-GradePts
- 04_StudentsAll
- 04_StudentsFall2008
- 04_StudentsSpring2009
- 05_Employees
- 06_MissedClassesAndPointsEarned

Retrieve Fields in this Order:

Sort field names Show: [✓ Tables] Views Synonyms System tables

Figure 70: Database Wizard-Select Data-Screen 2



Database Wizard

Select Data

Select the fields you want to retrieve. Then click the arrow button or drag the fields to the Retrieve Fields list.

Tip: Selecting a table selects all of its fields.

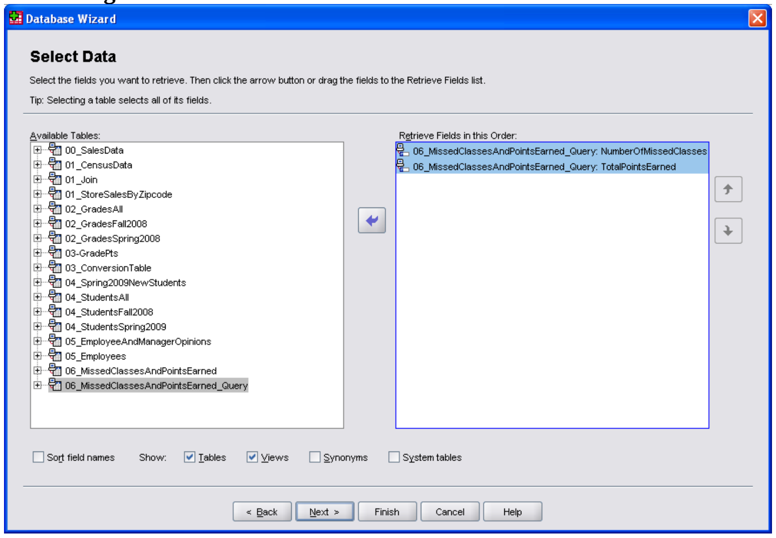
Available Tables:
- 00_SalesData
- 01_CensusData
- 01_Join
- 01_StoreSalesByZipcode
- 02_GradesAll
- 02_GradesFall2008
- 02_GradesSpring2008
- 03-GradePts
- 03_ConversionTable
- 04_Spring2009NewStudents
- 04_StudentsAll
- 04_StudentsFall2008
- 04_StudentsSpring2009
- 05_EmployeeAndManagerOpinions
- 05_Employees
- 06_MissedClassesAndPointsEarned
- 06_MissedClassesAndPointsEarned_Query

Retrieve Fields in this Order:

Sort field names Show: [✓ Tables] [✓ Views] Synonyms System tables

7. Notice that both the **NumberOfMissingClasses** and the **TotalPointsEarned** fields are now listed. Click **Finish**. The data is now showing in the **SPSS Data Editor** (see Figure 72). Congratulations, you have successfully imported the dataset into **SPSS**. Follow these same steps to bring in any table or query from an **MS Access** database.

Figure 71: Database Wizard-Select Data-Screen 3



Database Wizard

Select Data

Select the fields you want to retrieve. Then click the arrow button or drag the fields to the Retrieve Fields list.

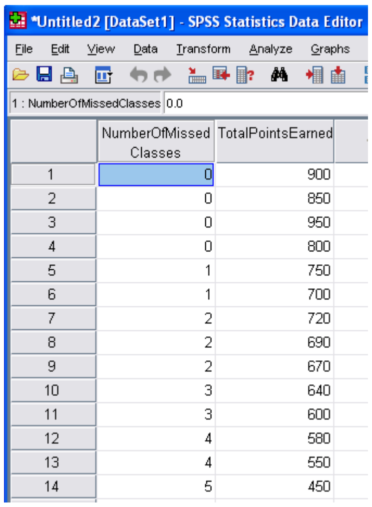
Tip: Selecting a table selects all of its fields.

Available Tables:
- 00_SalesData
- 01_CensusData
- 01_Join
- 01_StoreSalesByZipcode
- 02_GradesAll
- 02_GradesFall2008
- 02_GradesSpring2008
- 03-GradePts
- 03_ConversionTable
- 04_Spring2009NewStudents
- 04_StudentsAll
- 04_StudentsFall2008
- 04_StudentsSpring2009
- 05_EmployeeAndManagerOpinions
- 05_Employees
- 06_MissedClassesAndPointsEarned
- 06_MissedClassesAndPointsEarned_Query

Retrieve Fields in this Order:
- 06_MissedClassesAndPointsEarned_Query: NumberOfMissedClasses
- 06_MissedClassesAndPointsEarned_Query: TotalPointsEarned

Sort field names Show: [✓ Tables] [✓ Views] Synonyms System tables

Figure 72: Data in SPSS Data Editor



*Untitled2 [DataSet1] - SPSS Statistics Data Editor

1 : NumberOfMissedClasses 0.0

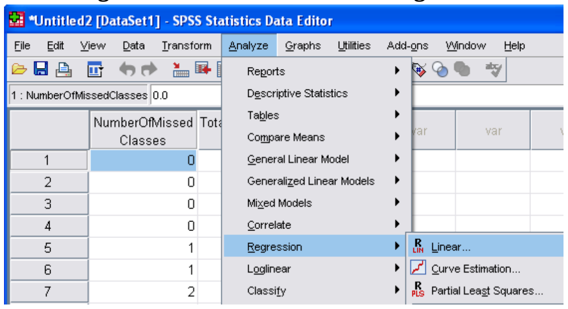
	NumberOfMissed Classes	TotalPointsEarned
1	0	900
2	0	850
3	0	950
4	0	800
5	1	750
6	1	700
7	2	720
8	2	690
9	2	670
10	3	640
11	3	600
12	4	580
13	4	550
14	5	450

B. *The next step is to run the regression in SPSS. In this dataset the **TotalPointsEarned** is the dependent variable and the **NumberOfMissingClasses** is the independent variable. Follow these steps to run the regression.*

1. From the menu, select **Analyze/Regression/Linear** (see Figure 73). The Linear Regression Dialog Box Appears (see Figure 74).

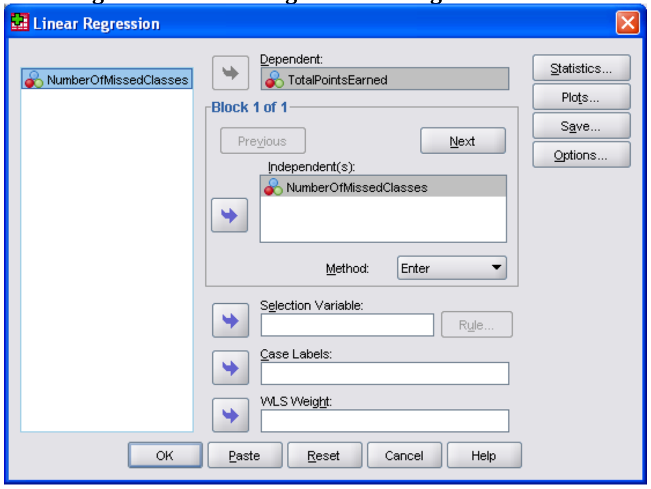
2. On the right-hand side of the screen, select **TotalPointsEarned** and click the next to the **Dependent:** variable. Next, select the **NumberOfMissingClasses** and click the next to the **Independents(s):** variable. Your screen should look similar to Figure 74.

Figure 73: Launch the Linear Regression



	NumberOfMissed Classes
1	0
2	0
3	0
4	0
5	1
6	1
7	2

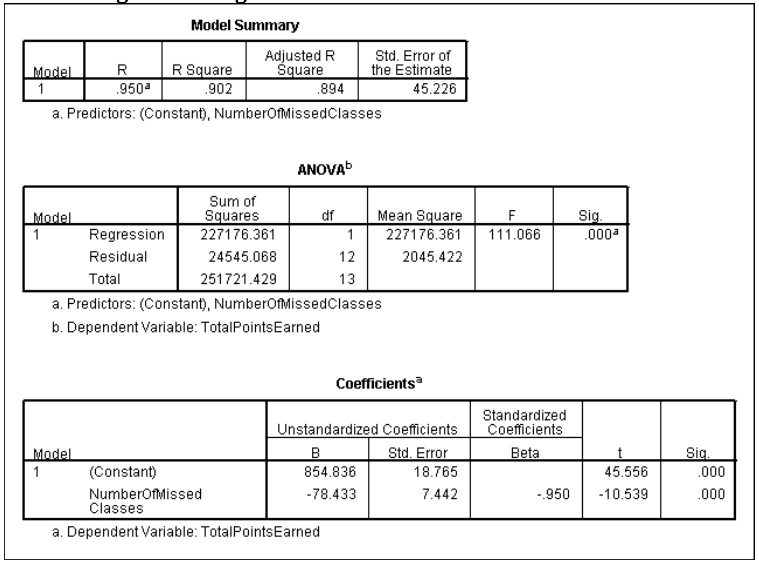
Figure 74: Linear Regression Dialog Box



Dependent: TotalPointsEarned
Independent(s): NumberOfMissedClasses
Method: Enter
Selection Variable:
Case Labels:
WLS Weight:

3. The regression is now ready to be run. Click **OK**. The results of the regression are now displayed in the **SPSS Statistics Viewer** (see Figure 75). By looking at the right-hand column of the **Coefficients** table, it is clear that independent variable **NumberOfMissingClasses** is shown to have a significant relationship (i.e, .000) with the dependent variable, **TotalPointsEarned**. This result could be interpreted as meaning that the more classes a student misses, the lower their total class points earned will be. Obviously this is a contrived dataset, however the two authors are also professors and we would like to believe that this is true!
4. Now that the regression is complete, it is time to store the results. Save the SPSS Statistical Viewer file as **06_MissedClassesAndPointsEarned.spo**; save the data file as **06_MissedClassesAndPointsEarned.sav**. Close **SPSS**.

Figure 75: Regression Results in SPSS Statistics Viewer



Model Summary

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	.950[a]	.902	.894	45.226

a. Predictors: (Constant), NumberOfMissedClasses

ANOVA[b]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	227176.361	1	227176.361	111.066	.000[a]
	Residual	24545.068	12	2045.422		
	Total	251721.429	13			

a. Predictors: (Constant), NumberOfMissedClasses

b. Dependent Variable: TotalPointsEarned

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients		
		B	Std. Error	Beta	t	Sig.
1	(Constant)	854.836	18.765		45.556	.000
	NumberOfMissed Classes	-78.433	7.442	-.950	-10.539	.000

a. Dependent Variable: TotalPointsEarned

CONCLUSION

The purpose of this paper was to help researchers deal with ill-behaved data through the use of relational database software, with particular focus on QBE (Query-By-Example). We believe we have accomplished this goal through a demonstration consisting of the following steps:

- How to import a misconfigured dataset from **MS Excel** into an **MS Access** table.
- Five examples of how to use **MS Access'** QBE to reconfigure datasets.
- How to import a reconfigured dataset from **MS Access** into **SPSS**.

The broader purpose of this paper is to demystify database software and encourage the reader to learn more about the usefulness of database tools and techniques in their own field of expertise. To this end, we have included in our reference list a number of database theory and MS Access texts that could be used for further explanations of database topics. We hope that they will prove helpful.

AUTHOR INFORMATION

Dr. Paul J. Lazarony is an Associate Professor of Information Systems at California State University, Northridge. His current research interests are assessment, database, enterprise-wide systems, and IT project management. He received his PhD in Computer Technology and Business Education from the Ohio State University.

Dr. Donna A. Driscoll is a Professor of Information Systems at California State University, Northridge. Her current research interests are database, information assurance, and web-based systems development. She received her PhD in Business Administration (Accounting and Research Statistics) from the University of Southern California.

REFERENCES

1. Groh, M. et al. (2007). *Access 2007 Bible*, Wiley Publishing, Indianapolis, IN.
2. Hoffer, A. and Prescott, M. (2009). *Modern Database Management*, Ninth Edition, Pearson Prentice Hall, Upper Saddle River, NJ.
3. Kroenke, D. and Auer, D. (2010). *Database Concepts,* Fourth Edition, Prentice Hall, Boston, MA.
4. Kroenke, D. and Auer, D. (2010). *Database Processing: Fundamentals, Design, and Implementation*, Eleventh Edition, Prentice Hall, Boston, MA.
5. Post, G. (2005). *Database Management Systems*, Third Edition, McGraw-Hill, Boston, MA.
6. Rob, P. and Coronel, C. (2007). Database Systems: Design, Implementation and Management, Seventh Edition, Thomson Course Technology, Boston, MA.
7. Shelly, G. and Cashman, T. (2007). *Microsoft Office Access 2007: Comprehensive Concepts and Techniques*, Thomson Course Technology, Boston, MA.