XNorthwind: Grammar-driven Synthesis of Large Datasets for DB Applications

Abejide Ade-Ibijola *Member, IAENG,* and George Obaido *Member, IAENG*

***Abstract*—Relational databases often come with sample databases. One known example is the Northwind database, often used as data repository for software testing and development purposes. The Northwind database includes hypothetical records of customers, companies, products, employee and so on. The number of records in the Northwind is however considered inadequate for large applications, where a developer or user may need a lot more, possibly, millions of records. In this paper, we have used a Context-free Grammar in describing the rules for the synthesis of exponentially many hypothetical datasets that are similar to the Northwind database. We referred to the resulting database as `XNorthwind` (Extended Northwind). The new grammar was implemented, resulting in thousands of unique data values across the eight different Northwind Data Tables. These datasets will find applications in training and development environments. A survey of 112 participants' perceptions showed that 94.6% agreed that the `XNorthwind` can be useful.**

***Index Terms*—Northwind, Sample database, Training dataset, Synthesis of things, Formal grammar applications.**

I. Introduction

With the advent of the Internet and other related technologies, various applications have emerged which has led to a high demand for data, stored in various database (DB) technologies [1]. In testing software applications before release, the higher the volume of data, the better the result derived from the system testing [2, 3, 4]. A number of applications have adopted sample DBs as practice environments for their testing and development tasks [5, 6, 7]. These sample DBs include the Sybase's `Pubs`, PostgreSQL's `Sakila` and Microsoft's `Northwind` [8]. In this work, we are interested in the Microsoft's Northwind DB, containing the records of a fictitious company known as the "Northwind Traders" [5]. The Northwind DB consist of hypothetical datasets that educates users with useful illustrations of a typical e-commerce scenarios and has been extensively used with many software applications and research projects [8, 9]. A 2018 study conducted on querying property graphs used the Northwind DB in a tool called `Gremlinator`, and authors reported good results [10]. Other tools that have used the Northwind DB are `OntoGrate` [11] and `SPARK` [5].

Despite the capabilities of the Northwind DB to support a diverse set of applications, the datasets is insufficient to meet the current demands of technologies that requires more data for their training needs [12, 13]. A comparative analysis study was conducted on the extraction and generation of Personal Data Reports (PDR) from two relational DBs (i.e. Northwind and TPC-H) [14]. Interestingly, the study showed that although, the Northwind possesses 3.7MB ($3.7 \cdot 10^3$ tuples) and TPC-H with 1GB (10^9 tuples) of datasets; the TPC-H datasets achieved a better accuracy because of the larger datasets. Roger [12] opined that the Northwind DB contains fewer records than one may find in most production DBs; hence, the current limited record size is not ideal to support a full fledged system. Taking this limitation into consideration, it has become imperative to create more datasets that can assist instructors and programmers in their training or development tasks.

In this work, we have used a Context-free Grammar (CFG) to describe the syntatic generation of tuples of hypothetical records, similar to the Northwind DB. This appears to be the first time such an approach is been extended to the automatic generation of large datasets to be used as a sample DB. The contributions are stated as follows.

We have:

1) designed a CFG for the synthesis of datasets for records of the Northwind DB,
2) implemented the CFG rules and shown that it produced 100,000 tuples (and could produced more) as opposed to 3,200 of the Northwind DB, and
3) evaluated this approach and shown that developers and application users agreed that large datasets can be useful.

The organisation of this paper is as follows. Section II presents the background to this work. Section III describes the grammar for the generation of `XNorthwind` datasets. Section IV presents the implementation details and result of the `XNorthwind` DB. Section V presents possible applications of the `XNorthwind` DB. Section VI presents the evaluation. In Section VII, we present the conclusion and provide future work.

II. Background and Related Work

In this section, we present the problem and a review of the relational DB model, focusing on the Northwind DB and application areas. We also presented definition of used terms.

A. The Problem

One major problem faced with most illustrative DBs such as the Northwind DB is its inability to provide enough datasets that meet the demands for testing critical applications before release [12, 13, 15]. In most cases, the datasets in the Northwind DB is insufficient to support large-scale applications. For enterprise applications, Rogers [12] identified that the Northwind DB does not qualify as a full fledged sample DB because the datasets is only ideal for small-sized wholesale or

Manuscript received September 11th, 2018; revised May 7th, 2019.

Abejide Ade-Ibijola is a Senior Lecturer in the Research Cluster on Formal Structures, Algorithms, and Industrial Applications, Department of Applied Information Systems, School of Consumer Intelligence and Information Systems, at the University of Johannesburg, Johannesburg, South Africa. website: https://www.abejide.org, e-mail: abejideai@uj.ac.za.

George Obaido is a PhD candidate at the School of Computer Science and Applied Mathematics, University of the Witwatersrand, Johannesburg, e-mail: rabeshi.george@gmail.com.

retail outfit mapped with procurement and order fulfillment processes. A similar study by Warren *et al.* [15] stated that the Northwind DB contains a small number of datasets, ideal to support a beginner learning DBs whilst inadequate to support large scale applications. Recently, an overview of problems faced while learning DBs indicated that large sample DB for training are not readily available, and most textbook examples are too oversimplified, and inadequate to cope with real-world scenario [16]. The author echoed that training students with a large sample DB would prepare them to cope with pressures at the workplace. In view of all these, synthesis of datasets in the Northwind DB is desirable. One benefit of this is that, it will provide software developers with enough datasets to use for deployment and testing applications. This work explores formal techniques using CFGs to solve this problem.

B. Relational Database

The relational DB model is one of the most simple structure for storing and organising data for easy retrieval [3, 17]. Since released in 1970s, it has found applications in large-scale commercial implementations of banking systems, airline reservation systems and in desktop computers for maintaining and storing of records [18]. The relational DB that we are interested in is the Microsoft Northwind DB.
Vicknair *et al* [19] described the Microsoft SQL Server as a relational DB that supports both desktop and web applications. Chung [20] highlighted the benefits of Microsoft SQL Server over other popular DBs such as tremendous ROI[1], Rapid Application Development; it is also good for data entry and reporting. The step by step installation guide for the Northwind DB is provided in [21]. The applications of Northwind DB is discussed in the next section.

C. Northwind Database and Applications

The Northwind DB contains eight tables and 3,200 tuples comprising of: Suppliers, Products, Orders, Shippers, Customers, OrderDetails, Categories and Employees [5]. As an illustrative DB, Northwind resemble a typical merchandise firm that undergoes sales transactions that occur between a company and its customers. This DB provides a model for table relationships, forms, queries, VBA[2], data access, and manipulation functionalities [22]. Borker [23] regarded the Northwind as an "intuitive" OLTP[3] system that stores and links tables by means of a primary key. Nelson [24] illustrates the Northwind DB using a schema showing entities and the relationship among them as seen in Figure 1. In the schema, orders are shipped by a Supplier with details stored in the Shippers table.
The Northwind DB have been extensively used in a number of applications such as:

Decision Support Systems Angermann *et al* [26] used the Northwind DB to demonstrate the efficiency of Taxo-semantics, a decision support system that was used to match an expression against other sources of knowledge. The study concluded that the Northwind DB improved the accuracy of the system. A recent study in 2018 by Runtuwene *et al.* [7] applied the Northwind DB for a comparative study for the Extract, Transformation and Loading (ETL) data integration processes. The study aimed to assist a BI[4] developer in processing data to produce useful information.

Semantic Web Applications A number of semantic web applications have used the datasets from the Northwind DB for their operations. Tools, such as `SPARK` [5] and `OntoGrate` [11] used the Northwind DB as a back-end for a keyword search and semantic web ontologies respectively.

Natural Language Systems Lumbantoruan *et al* [6] applied the Northwind DB in evaluating a *star schema*[5], that automatically generates and identifies noun words. A study conducted by Gelbukh [27] used the Northwind DB in the translation of queries expressed in natural languages; using prepositions and conjunctions into formal languages.

Computer Science Education In an introductory course on IT Audit, Northwind was used as a tutorial DB for beginners [28]. The author stressed that although the Northwind DB was ideal for teaching, its datasets is inadequate for analysis in a vendor neutral environment. Similarly, Lavbič [29] proposed a system that applies *hints*, meant to assist students to solve SQL-related exercises. The system adopted the Northwind DB as the backend in solving problems in SQL related tasks.

Healthcare Systems Kaddoura *et al* [30] conducted a study that involved tracking and repairing damaged health care databases, the Northwind DB was used as the experimental db. The study showed that the Northwind DB performed better because of its data consistency. The result of this study were further replicated in similar studies [31, 32].

We have presented the application areas of the Northwind DB. It is important to note that the above-mentioned areas are some of application of this test DB. Other applications that have used the Northwind DB are discussed in [33, 34].

D. Definition of Terms

Noam Chomsky coined the term "Context-free Grammars" or CFGs while describing classes of formal grammars [35]. These grammars differs with their generative and recognitive capacity. Here, we define some terms used in this paper.

Definition 1. *(Context-free grammar [36]). A context-free grammar or CFG is a four-tuple, $G = (N, \Sigma, P, S)$ where:*

1) *N is a finite set of non-terminal symbols.*
2) *Σ represents a finite set of terminals symbols, disjoint from N.*
3) *P is a set of productions.*
4) *S is the start symbol.*

Each non-terminals can be replaced by a string of terminals to the right of the arrow represented as production rules. The rule of the form: $A \longrightarrow \alpha$, simply replaces A with

[1]Return On Investment
[2]Visual Basic for Applications
[3]Online Transaction Processing
[4]Business Intelligence
[5]A form of data warehouse modelling

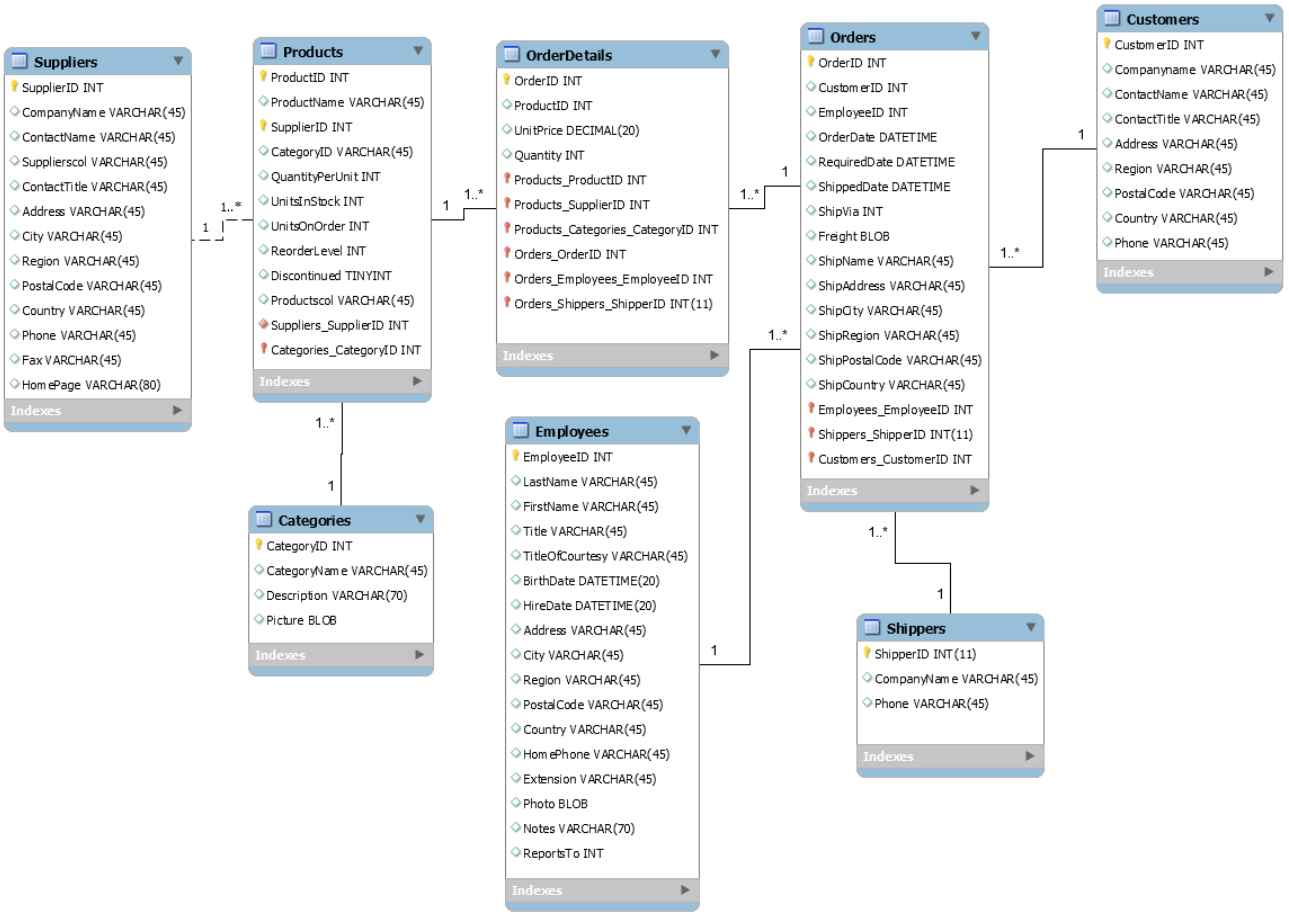


Fig. 1. The database schema of the Northwind DB (This image was redrawn from [25])

α, where A is the non-terminal or a left-hand side symbol and α are strings of right-hand side symbols or terminals.

E. Related Work

Formal grammars have been used in a wide range of applications. In this section, we present applications of CFGs to research similar to that discussed in this work.

Structural 3D Designs Formal Grammars have been extensively used in the design fields such as product design [37], architecture [38] and 3D modeling [39]. Christensen [40] extended the use of CFGs in a tool named `Structure Synth` for creating 3D images. The `Structure Synth` engine uses a recursive descent parser to create and transform rules stored in `4x4` matrices.

Profile Synthesis Ade-Ibijola [41] developed a tool based on a variation of CFGs, that automatically synthesises social media profiles using the Facebook user profile page as a test case. Lin [42] presented a tool aimed at assisting a digital forensic examiner to build behavioural profile from analysis of a network traffic. This tool applied CFGs to compare behavioural patterns and reduce the volume of evidence needed to analyse a network traffic.

Multimedia Applications A study was conducted by Pudaruth *et al.* [43] using CFGs to automatically generate song lyrics. The lyrics generator applied grammatical rules and statistical constraints derived from a song corpus to generate lyrics. `FINCHAN` [44] was developed using CFGs for the automatic comprehension and summarisation of financial instant messaging.

Natural Language Processing (NLP) A recent study by Velupillai *et al.* [45] showed that CFGs was used to identify pathological findings in radiology reports in clinical NLP data. The study showed that integrating CFGs to state-of-the-art NLP tools will advance clinical tools in the near future. Liang [46] built a parser using CFGs for natural language understanding in a question answering system. The study concluded that the CFGs were an essential component used to parse natural languages in this system.

Protein Synthesis One notable application of CFGs in RNA[6] structure prediction and detection of patterns in DNA[7] was presented in [47]. Experiments in this study concluded that the CFG approach was helpful in producing human-readable descriptors for the analysis of these protein sequences.

Program Synthesis Butler [48] proposed a system that uses the CFGs with a domain-specific extension to support variable binding and a type system to construct a program. A research study in 2018 by Ade-Ibijola [49] uses CFGs for the automatic generation of procedural programs in Python. The study concluded that the CFG approach used in this research can be applied to generate

[6]Ribonucleic acid
[7]Deoxyribonucleic acid

programs in many procedural programming languages.

Signal Processing Macko [50] used CFGs to syntactically analyse a VHDL (VHSIC Hardware Description Language) model used for digital signal processing before it is visualised and simulated. In this work, CFG was used to transform the VHDL into an intermediate form that conforms with processing of digital signals. A research study by Fanaswala and Krishnamurthy [51] extended the use of a variation of CFG and the reciprocal Markov model to model long-range signal dependencies. The authors stressed that the CFG possess the added advantage because of its expressive power and ability to deal with variable-range dependencies.

Together, all these areas have applied the use of CFGs for describing the languages used in these domains. Other applications areas of CFGs include: Fuzzy systems [52, 53], Safety systems [54, 55] and Software systems [56, 57]. In the next section, we describe the grammar formalism for synthesizing large datasets for `XNorthwind` DB.

III. Grammar Design for the XNorthwind Database

In the previous section, a wide range of applications areas of CFGs was presented. This section describes the use of CFGs for the automatic generation of large datasets in the XNorthwind DB.

$$\texttt{<comma>} \longrightarrow , \tag{1}$$
$$\texttt{<wspace>} \longrightarrow ws \tag{2}$$
$$\texttt{<period>} \longrightarrow . \tag{3}$$
$$\texttt{<dash>} \longrightarrow - \tag{4}$$
$$\texttt{<d>} \longrightarrow 0 \mid \ldots \mid 9 \tag{5}$$
$$\texttt{<b\_slash>} \longrightarrow \backslash \tag{6}$$
$$\texttt{<f\_slash>} \longrightarrow / \tag{7}$$
$$\texttt{<colon>} \longrightarrow : \tag{8}$$
$$\texttt{<brac\_o>} \longrightarrow (\tag{9}$$
$$\texttt{<brac\_c>} \longrightarrow) \tag{10}$$
$$\texttt{<sup\_id>} \longrightarrow 1 \mid \ldots \mid 5 \cdot 10^3 \tag{11}$$
$$\texttt{<cat\_id>} \longrightarrow 1 \mid \ldots \mid 5 \cdot 10^3 \tag{12}$$
$$\texttt{<emp\_id>} \longrightarrow 1 \mid \ldots \mid 2.1 \cdot 10^4 \tag{13}$$
$$\texttt{<ship\_id>} \longrightarrow 1 \mid \ldots \mid 10^3 \tag{14}$$
$$\texttt{<cus\_id>} \longrightarrow 1 \mid \ldots \mid 2.8 \cdot 10^4 \tag{15}$$
$$\texttt{<ord\_id>} \longrightarrow 1 \mid \ldots \mid 2 \cdot 10^4 \tag{16}$$
$$\texttt{<prod\_id>} \longrightarrow 1 \mid \ldots \mid 2 \cdot 10^4 \tag{17}$$

To generate the datasets, we describe the set of productions which are rules that make up the grammar. These rules replace the nonterminal symbols that appear on the left-hand side with terminal or nonterminals symbols on the right-hand side of the productions. In Production 1-10, we present the symbols that appear in some of the rules used for other productions. The initial productions show comma, white space ($wspace$), period ($period$), hyphen ($dash$), digits (d), backslash (b_slash), forward slash (f_slash), colon ($colon$), bracket open ($brac\_o$) and bracket close ($brac\_c$). Productions 11-17 is used to present the *id*s (primary) keys for each of the eight tables, and in some cases, they appear as foreign keys in some tables. For example, the `<ord_id>` appears as a primary key in the `Orders` table. Similarly, it is a foreign key in the `Orderdetails` table. Productions 11 and 12 allow for random supplier and category *ids* $\in$ [5000]. Production 13 allows for employee *ids* $\in$ [21000]. Production 14 allows for shippers *ids* $\in$ [1000]. Production 15 allows for customer *id* $\in$ [28,000]. Productions 16 and 17 allow for order and product *ids* $\in$ [20,000]. This amounts to 100,000 as opposed to 3,200 tuples contained in the `Northwind` DB.

In Productions 18-21, the orders and quantity of the tables are generated within the range presented. The quantity field is found in the Orderdetails and Products tables.

$$\texttt{<unit\_order>} \longrightarrow 0 \mid \ldots \mid 120 \tag{18}$$
$$\texttt{<units\_stk>} \longrightarrow 0 \mid \ldots \mid 100 \tag{19}$$
$$\texttt{<reorder\_l>} \longrightarrow 0 \mid \ldots \mid 30 \tag{20}$$
$$\texttt{<quantity>} \longrightarrow 1 \mid \ldots \mid 50 \tag{21}$$

Productions for names specified in the tables are presented within the range of 22 and 32. The `<fname>` symbol is specified as the first name of the field in the table where n_1 is the total number of first names that appear. The `<lname>` symbol specifies the last name of the field and n_2 is the number of last names in the field. The `<cat_name>` symbol represents the category names and n_3 is the total number of all category names contained in the field. In the `<comp_suffix>` symbol, this shows the company suffixes that may appear (e.g. Limited, Services, Agency, Consulting, Advisors, etc.) n_4 is the number of such suffixes. In Production 26, a company name is generated with first name and arbitrary company suffixes (e.g. Booysen Consulting). Production 27 shows how a contact is generated while n_5 shows the number of generated contacts. In Production 28, arbitrary ship names are generated and n_6 shows the total number of generated ship names. The ship via symbol, `<ship_via>`, is generated in Production 29. n_7 is the total number of such pattern. Production 30 describes the rules for product names and n_8 shows the total number of product names that appears. The `<report_to>` symbol is a concatenation of the first and last names as seen in Production 31. The `<cont_name>` holds if the first name and last name applies, indicated in Production 32.

$$\texttt{<fname>} \longrightarrow fn_1 \mid \ldots \mid fn_{n_1} \tag{22}$$
$$\texttt{<lname>} \longrightarrow ln_1 \mid \ldots \mid ln_{n_2} \tag{23}$$
$$\texttt{<cat\_name>} \longrightarrow cat_1 \mid \ldots \mid cat_{n_3} \tag{24}$$
$$\texttt{<comp\_suffix>} \longrightarrow cf_1 \mid \ldots \mid cf_{n_4} \tag{25}$$
$$\texttt{<comp\_name>} \longrightarrow \texttt{<fname><wspace><comp\_suffix>} \tag{26}$$
$$\texttt{<contact>} \longrightarrow con_1 \mid \ldots \mid con_{n_5} \tag{27}$$
$$\texttt{<ship\_name>} \longrightarrow shp_1 \mid \ldots \mid shp_{n_6} \tag{28}$$
$$\texttt{<ship\_via>} \longrightarrow shv_1 \mid \ldots \mid shv_{n_7} \tag{29}$$
$$\texttt{<prod\_name>} \longrightarrow pname_1 \mid \ldots \mid pname_{n_8} \tag{30}$$
$$\texttt{<report\_to>} \longrightarrow \texttt{<fname><comma><lname>} \tag{31}$$
$$\texttt{<cont\_name>} \longrightarrow \texttt{<fname><wspace><lname>} \tag{32}$$

Productions 33-36 is used for titles. The contact title symbol, <cont_title>, is generated in Production 33 (e.g. Purchasing Manager, Sales Manager, Owner, etc.). n_9 is the total number of contact titles that may appear. The <title_court> (title of courtesy) as seen in Production 34 holds if the Production 35 is satisfied. The different title types that we have in this list are: *Dr, Mrs, Mr* and *Miss* followed by a period. Production 36 shows the generation for employee titles that we may appear (e.g. Sales Representative, Vice President, Chairman etc.) and n_9 shows the total number of such titles.

$$\texttt{<cont\_title>} \longrightarrow cont_1 \mid \ldots \mid cont_{n_9} \quad (33)$$

$$\texttt{<title\_court>} \longrightarrow \texttt{<title\_types><period>} \quad (34)$$

$$\texttt{<title\_types>} \longrightarrow Dr \mid Mrs \mid Mr \mid Ms \quad (35)$$

$$\texttt{<emp\_title>} \longrightarrow empt_1 \mid \ldots \mid empt_{n_{10}} \quad (36)$$

Productions for supplier, category, description and notes is shown in 37-40. Hence, n_{11} to n_{14} show the total number of suppliers, categories, description and notes names.

$$\texttt{<supp>} \longrightarrow sp_1 \mid \ldots \mid sp_{n_{11}} \quad (37)$$

$$\texttt{<categ>} \longrightarrow ctg_1 \mid \ldots \mid ctg_{n_{12}} \quad (38)$$

$$\texttt{<desc>} \longrightarrow desc_1 \mid \ldots \mid desc_{n_{13}} \quad (39)$$

$$\texttt{<notes>} \longrightarrow not_1 \mid \ldots \mid not_{n_{14}} \quad (40)$$

Productions 41-42 holds if either of the entries in the symbols are generated. Production 41 shows either an individual's gender is a male, a female or other as indicated in Production 41. The <discontinue> symbol shows either if a product should be continued or not as seen in Production 42.

$$\texttt{<gender>} \longrightarrow male \mid female \mid other \quad (41)$$

$$\texttt{<discontinue>} \longrightarrow yes \mid no \quad (42)$$

The <price> symbol satisfies the Productions 43-44. In this case, we opted for the South African currency symbol - Rands denoted as *R* (e.g R25.00). The <freight> shows a price if Productions 43 is satisfied.

$$\texttt{<price>} \longrightarrow R\texttt{<d>}^+\texttt{<period><d><d>} \quad (43)$$

$$\texttt{<freight>} \longrightarrow \texttt{<price>} \quad (44)$$

Productions 45-49 show country, ship country, region, city and ship city. The <ship_country> symbol is satisfied depending on the list of countries specified in Production 45. The countries that were generated in this production are: UK, USA, Germany, Australia, South Africa, Nigeria etc. n_{15} show the number of countries that appear in this list. Production 47 and 48 describe rules for the formulation of regions and cities. Here, n_{16} and n_{17} are the number of region and city names respectively. We enforced rules to ensure that this concatenation exists. For example, the city, Melbourne, matches with the Australian Victoria region. Production 49 holds if a city is generated, as seen in Production 48.

$$\texttt{<country>} \longrightarrow count_1 \mid \ldots \mid count_{n_{15}} \quad (45)$$

$$\texttt{<ship\_country>} \longrightarrow \texttt{<country>} \quad (46)$$

$$\texttt{<region>} \longrightarrow reg_1 \mid \ldots \mid reg_{n_{16}} \quad (47)$$

$$\texttt{<city>} \longrightarrow cty_1 \mid \ldots \mid cty_{n_{17}} \quad (48)$$

$$\texttt{<ship\_city>} \longrightarrow \texttt{<city>} \quad (49)$$

$$\texttt{<phone>} \longrightarrow \texttt{<s\_code><period><phone\_d>} \quad (50)$$

$$\texttt{<phone\_d>} \longrightarrow C \in \texttt{<d>}^+ : |C| = 7 \quad (51)$$

$$\texttt{<s\_code>} \longrightarrow \texttt{<brac\_o><d><brac\_c>} \quad (52)$$

$$\texttt{<fax>} \longrightarrow \texttt{<phone>} \quad (53)$$

$$\texttt{<extension>} \longrightarrow \texttt{<s\_code>} \quad (54)$$

$$\texttt{<p\_code>} \longrightarrow pc_1 \mid \ldots \mid pc_{n_{18}} \quad (55)$$

$$\texttt{<ship\_pcode>} \longrightarrow \texttt{<ship\_pcode><s\_code>} \mid \texttt{<ship\_pcode><s\_morecode>} \quad (56)$$

$$\texttt{<s\_morecode>} \longrightarrow \texttt{<brac\_o><d>}^+\texttt{\_<brac\_c>} \quad (57)$$

$$\texttt{<address>} \longrightarrow \texttt{<d><wspace><add\_list><comma><city>} \quad (58)$$

$$\texttt{<add\_list>} \longrightarrow addl_1 \mid \ldots \mid addl_{n_{19}} \quad (59)$$

$$\texttt{<ship\_add>} \longrightarrow \texttt{<address>} \quad (60)$$

Productions 50-57 specifies the symbol for phone, fax, extension and ship code. Productions 50-52 generates a phone number where C is a 7-digit pseudorandom number. The <s_code> symbol shows the prefixes that are used by service providers in South Africa (e.g. 061, 082, 084, 072, etc.). Production 53 shows the rules for a fax number. Every fax number is equivalent to a phone number. Production 54 holds if prefixes are satisfied in Production 52. Production 55 specify the rules for postal codes and n_{18} is the total number of postal code names. Productions 56 to 57 are recursively defined that allows more occurrences of values.

The <address> symbol satisfies the Productions 58-60. To generate an address, we specify a house number <d> followed by a street name <add_list>, and a city <city> (e.g. 54, Klein Street, Johannesburg). The <add_list> symbol holds a street name with n_{19} specifying the total number of street names. The <ship_add> holds if Production 58 is satisfied.

The <date> symbol satisfies Productions 61-70, and is composed of the terminal symbol: day of the week <d_wk>, days of the month <d_mnth>, month of the year <mnth_y> and year <yr>. The <birth_d> symbol as indicated in Production 66 holds, if an individual is between the ages of 18 to a retirement age of 65, according to the Gregorian calendar. Productions 67 to 70 holds if a date is satisfied.

<date> ⟶ <d_wk><dash><d_mnth>
<dash><mnth_y><dash>
<yr> (61)

<d_wk> ⟶ $Sunday$ |...| $Saturday$ (62)

<d_mnth> ⟶ <d><d> (63)

<mnth_y> ⟶ Jan |...| Dec (64)

<yr> ⟶ 1990 |...| 2018 (65)

<birth_d> ⟶ <date> ∋: <yr>
$\in [1954, 2000]$ (66)

<hire_d> ⟶ <date> (67)

<order_d> ⟶ <date> (68)

<req_d> ⟶ <date> (69)

<ship_d> ⟶ <date> (70)

<name> ⟶ $name_1$ |...| $name_{n_{20}}$ (71)

<homepage> ⟶ <protocol><colon>
<f_slash><f_slash>
<host_name><period>
<suffix><f_slash>
<folder><f_slash>
<filename> (72)

<protocol> ⟶ $http$ | $https$ | ftp (73)

<host_name> ⟶ <host><period>
<name> (74)

<host> ⟶ www | ftp | $mail$ (75)

<suffix> ⟶ com | uk | us | ng | za
| cd (76)

<folder> ⟶ <name> (77)

<file_suffix> ⟶ htm | $html$ | jpg | png |
txt (78)

<filename> ⟶ <name><period>
<file_suffix> (79)

The <homepage> symbol satisfies Productions 72-78. This rule basically specify use a protocol, followed by a colon and a double front slash with a host and domain name. This is followed by a period, a suffix, a single front slash and a folder, a single front slash and a file name. An example of the <homepage> symbol specify a complete web url address (e.g http://www.mydomain.com/folder/image.png). Production 79 describe a given name as described in Production 71 with a period and a file suffix.

Within Productions 1 to 78, we have defined the elements that are used to create the rules for the tables. Productions 80 to 87 specify the rules for the tables.
The complete formalism for the Shippers table in Production 80 with fields — *Shipper ID*, *Company Name* and *Phone* is derived from the Productions (14, 26, 50) and presented below:

<shippers_tb> ⟶ <ship_id><comp_name>
<phone> (80)

The productions for the Order Details table in Production 81 with fields — *OrderID*, *ProductID*, *UnitPrice*, *Quantity* are 16, 17, 43, 21 respectively.

<orderdetails_tb> ⟶ <ord_id><prod_id>
<price><quantity> (81)

The Categories table yields the fields — *CategoryID*, *CategoryName*, *Description* with Productions (12, 24, 39) respectively. The formalism as seen in Production 82 for this table produces:

<categories_tb> ⟶ <cat_id><cat_name>
<desc> (82)

The formalism for the Orders table; with fields such as *OrderID, CustomerID, EmployeeID, OrderDate, RequiredDate, ShippedDate, ShipVia, Freight, ShipName, ShipAddress, ShipCity, ShipRegion, ShipPostalCode*, and *ShipCountry* as seen in Production 83.

<orders_tb> ⟶ <ord_id><cus_id>
<emp_id><order_d><req_d>
<ship_d><ship_via>
<freight><ship_name>
<ship_add><ship_city>
<region><ship_pcode>
<ship_country> (83)

The Customer table is formalised using its fields — *CustomerID, CompanyName, ContactName, ContactTitle, Address, Region, Postalcode, Country* as presented in Production 84.

<customer_tb> ⟶ <cus_id><comp_name>
<cont_name><cont_title>
<address><region>
<p_code><country>
<phone> (84)

The formalism for the Product table with fields — *ProductID, ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitsinStock, UnitsOnOrder, Reorderlevel* and *Discontinued* as displayed in Production 85.

<product_tb> ⟶ <prod_id><prod_name>
<sup_id><cat_id>
<quantity><units_stk>
<units_order><reorder_l>
<discontinue> (85)

The Employee table is formalised in Production 86 with fields — *EmployeeID, LastName, FirstName, Title, TitleofCourtesy, BirthDate, HireDate, Address, City, Region, Postalcode, Country, Homephone, Extension, Photo, Notes, ReportsTo*. The production for photo is beyond the scope of this paper.

```
<employee_tb> ⟶ <emp_id><lname><fname>
                <title><title_court>
                <birth_d><hire_d><address>
                <city><region><p_code>
                <country><phone><s_code>
                <notes><report_to>          (86)
```

The formalism for the Supplier table is derived from the fields — *SupplierID, CompanyName, ContactName, ContactTitle, City, Region, PostalCode, Country, Phone, Fax, Homepage*, as presented in Production 87.

```
<supplier_tb> ⟶ <sup_id><comp_name>
                <cont_name><cont_title>
                <city><region><p_code>
                <country><phone><fax>
                <homepage>                  (87)
```

IV. Implementation and Results

We have implemented the productions as described in Section 3 and presented a hypothetical DB called the `XNorthwind` (or Extended Northwind). `XNorthwind` was implemented using the .Net framework and the synthesized datasets were stored in Microsoft SQL Server. The synthesiser produced 100,000 iterations of datasets as opposed to 3,200 tuples of the Northwind DB. We have presented the results of two tables: Shippers and Customers table. Figure 2 shows the datasets in the Shippers table with 1,000 tuples as opposed to three tuples in the Northwind DB. This is described in Production 80 in Section III. Figure 3 shows the datasets in the Customers table of the first 10,000 tuples as opposed to 91 tuples in the Northwind DB. We have described the Customers table in Production 84 in Section III.

V. Applications of XNorthwind

In this section, we present possible applications of the `XNorthwind` DB that was presented in Section IV. Possible applications of the `XNorthwind` DB are:

1) new products and services can be tested using this database,
2) given its volume, it can be widely used in CRM[8] and ERP[9] applications, and
3) used in ITS[10] as a practice DB for teaching database concepts to students.

VI. Evaluation

We obtained the results of the evaluation through an online survey. This survey was carried out to obtain feedback from respondents on their perceptions of the generated datasets and its usefulness. The respondents were mostly educators and students' in the information systems and computer sciences disciplines from two South African universities namely: the University of Johannesburg and the University of the Witwatersrand. We received a total of 112 responses from the respondents. The respondents were initially asked to rate their knowledge with DBs on a rating scale (for example: one (1) indicating no experience at all and ten (10) for strongly experienced). We noticed that they all had knowledge with databases (See Figure 4(a)). Furthermore, we asked them if they have used the Northwind DB. 44.6% acknowledged that they have used the Northwind DB. We can agree that this number may represent students who may have only used the Microsoft DB for data storage without a clue of hypothetical datasets (See Figure 4(b)). Furthermore, we asked the respondents about the `XNorthwind`, and suggested if they think this DB can be useful to have. 94.6% agreed that the `XNorthwind` DB can be useful to have, 5.4% stayed indifferent and no respondent indicated that this DB is unusable (See Figure 4(c) – a combination of participants who 'strongly agreed' and 'agreed').

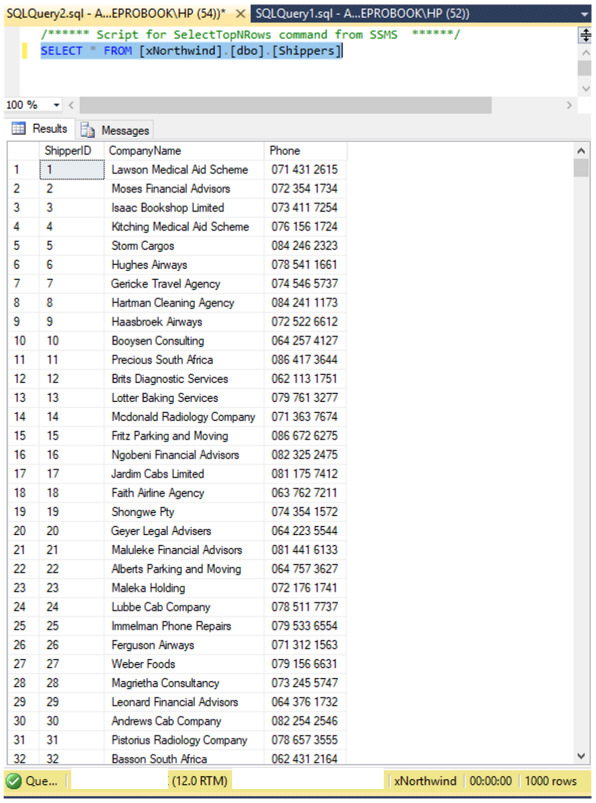
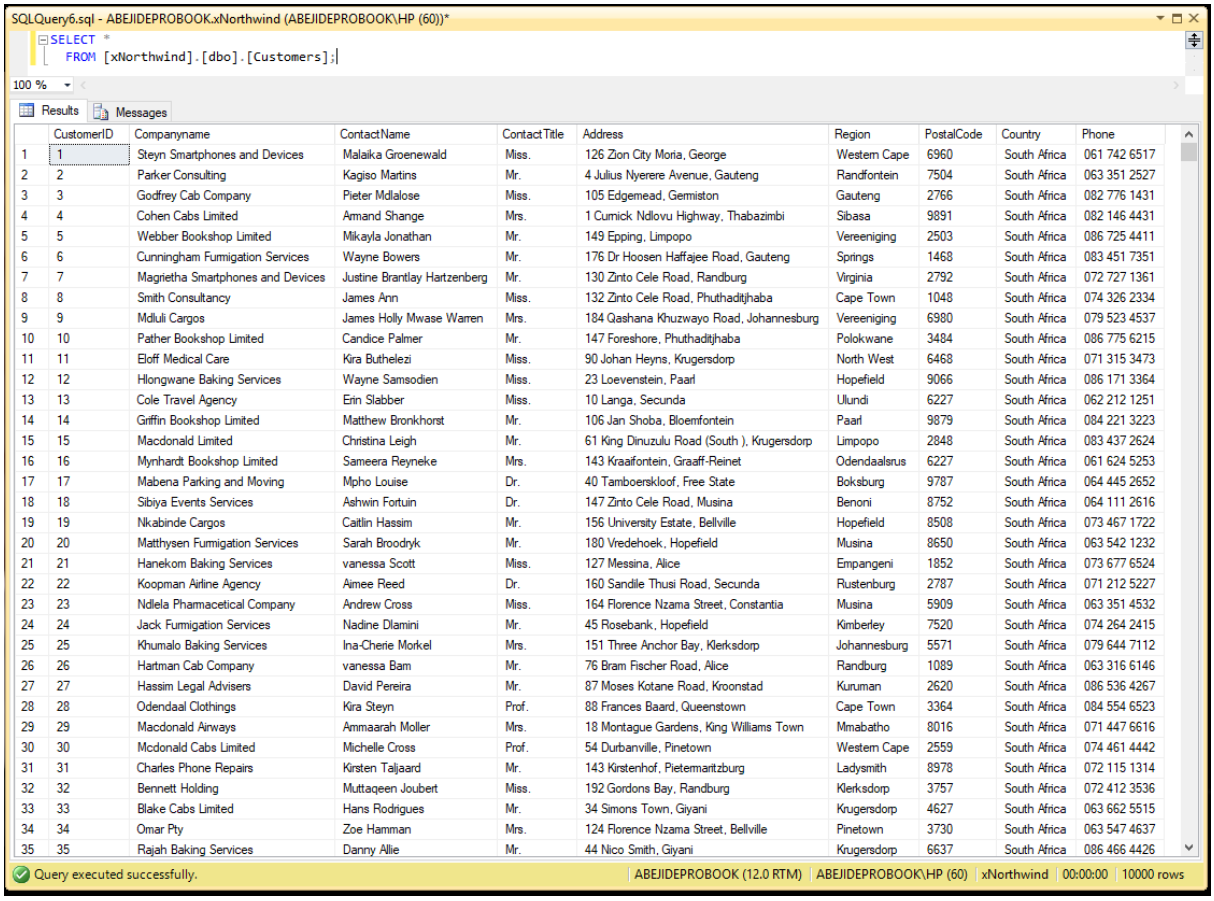


Fig. 2. `XNorthwind`: Output showing synthesised shippers' table

In addition, we asked the respondents if they think that the `XNorthwind` has wider application than the original `Northwind`. About 95.5% strongly believed that `XNorthwind` has wider application over the Northwind owing to the extra datasets. 4.5% stayed indifferent and no respondent agreed that `XNorthwind` is disadvantageous to have (See Figure 4(d)). Lastly, we asked the respondents to suggest an application of large datasets (`XNorthwind`). 41.1% suggested that large datasets can be used extensively. 58% had no idea and 0.9% stayed indifferent. We believe that majority of our respondents are students and may not have used sample DB (See Figure 4(e)). Given these feedback, we conclude that the generation of large datasets can be useful.

[8]Customer Relationship Management
[9]Enterprise Resource Planning
[10]Intelligent Tutoring Systems



SQLQuery6.sql - ABEJIDEPROBOOK.xNorthwind (ABEJIDEPROBOOK\HP (60))*

```
SELECT *
  FROM [xNorthwind].[dbo].[Customers];
```

100 %

Results | Messages

	CustomerID	Companyname	ContactName	ContactTitle	Address	Region	PostalCode	Country	Phone
1	1	Steyn Smartphones and Devices	Malaika Groenewald	Miss.	126 Zion City Moria, George	Western Cape	6960	South Africa	061 742 6517
2	2	Parker Consulting	Kagiso Martins	Mr.	4 Julius Nyerere Avenue, Gauteng	Randfontein	7504	South Africa	063 351 2527
3	3	Godfrey Cab Company	Pieter Mdlalose	Miss.	105 Edgemead, Germiston	Gauteng	2766	South Africa	082 776 1431
4	4	Cohen Cabs Limited	Amand Shange	Mrs.	1 Cumick Ndlovu Highway, Thabazimbi	Sibasa	9891	South Africa	082 146 4431
5	5	Webber Bookshop Limited	Mikayla Jonathan	Mr.	149 Epping, Limpopo	Vereeniging	2503	South Africa	086 725 4411
6	6	Cunningham Furmigation Services	Wayne Bowers	Mr.	176 Dr Hoosen Haffajee Road, Gauteng	Springs	1468	South Africa	083 451 7351
7	7	Magrietha Smartphones and Devices	Justine Brantlay Hartzenberg	Mr.	130 Zinto Cele Road, Randburg	Virginia	2792	South Africa	072 727 1361
8	8	Smith Consultancy	James Ann	Miss.	132 Zinto Cele Road, Phuthaditjhaba	Cape Town	1048	South Africa	074 326 2334
9	9	Mdluli Cargos	James Holly Mwase Warren	Mrs.	184 Qashana Khuzwayo Road, Johannesburg	Vereeniging	6980	South Africa	079 523 4537
10	10	Pather Bookshop Limited	Candice Palmer	Mr.	147 Foreshore, Phuthaditjhaba	Polokwane	3484	South Africa	086 775 6215
11	11	Eloff Medical Care	Kira Buthelezi	Miss.	90 Johan Heyns, Krugersdorp	North West	6468	South Africa	071 315 3473
12	12	Hlongwane Baking Services	Wayne Samsodien	Miss.	23 Loevenstein, Paarl	Hopefield	9066	South Africa	086 171 3364
13	13	Cole Travel Agency	Erin Slabber	Miss.	10 Langa, Secunda	Ulundi	6227	South Africa	062 212 1251
14	14	Griffin Bookshop Limited	Matthew Bronkhorst	Mr.	106 Jan Shoba, Bloemfontein	Paarl	9879	South Africa	084 221 3223
15	15	Macdonald Limited	Christina Leigh	Mr.	61 King Dinuzulu Road (South), Krugersdorp	Limpopo	2848	South Africa	083 437 2624
16	16	Mynhardt Bookshop Limited	Sameera Reyneke	Mrs.	143 Kraaifontein, Graaff-Reinet	Odendaalsrus	6227	South Africa	061 624 5253
17	17	Mabena Parking and Moving	Mpho Louise	Dr.	40 Tamboerskloof, Free State	Boksburg	9787	South Africa	064 445 2652
18	18	Sibiya Events Services	Ashwin Fortuin	Dr.	147 Zinto Cele Road, Musina	Benoni	8752	South Africa	064 111 2616
19	19	Nkabinde Cargos	Caitlin Hassim	Mr.	156 University Estate, Bellville	Hopefield	8508	South Africa	073 467 1722
20	20	Matthysen Furmigation Services	Sarah Broodryk	Mr.	180 Vredehoek, Hopefield	Musina	8650	South Africa	063 542 1232
21	21	Hanekom Baking Services	vanessa Scott	Miss.	127 Messina, Alice	Empangeni	1852	South Africa	073 677 6524
22	22	Koopman Airline Agency	Aimee Reed	Dr.	160 Sandile Thusi Road, Secunda	Rustenburg	2787	South Africa	071 212 5227
23	23	Ndlela Pharmacetical Company	Andrew Cross	Miss.	164 Florence Nzama Street, Constantia	Musina	5909	South Africa	063 351 4532
24	24	Jack Furmigation Services	Nadine Dlamini	Mr.	45 Rosebank, Hopefield	Kimberley	7520	South Africa	074 264 2415
25	25	Khumalo Baking Services	Ina-Cherie Morkel	Mrs.	151 Three Anchor Bay, Klerksdorp	Johannesburg	5571	South Africa	079 644 7112
26	26	Hartman Cab Company	vanessa Bam	Mr.	76 Bram Fischer Road, Alice	Randburg	1089	South Africa	063 316 6146
27	27	Hassim Legal Advisers	David Pereira	Mr.	87 Moses Kotane Road, Kroonstad	Kuruman	2620	South Africa	086 536 4267
28	28	Odendaal Clothings	Kira Steyn	Prof.	88 Frances Baard, Queenstown	Cape Town	3364	South Africa	084 554 6523
29	29	Macdonald Airways	Ammaarah Moller	Mrs.	18 Montague Gardens, King Williams Town	Mmabatho	8016	South Africa	071 447 6616
30	30	Mcdonald Cabs Limited	Michelle Cross	Prof.	54 Durbanville, Pinetown	Western Cape	2559	South Africa	074 461 4442
31	31	Charles Phone Repairs	Kirsten Taljaard	Mr.	143 Kirstenhof, Pietermaritzburg	Ladysmith	8978	South Africa	072 115 1314
32	32	Bennett Holding	Muttaqeen Joubert	Miss.	192 Gordons Bay, Randburg	Klerksdorp	3757	South Africa	072 412 3536
33	33	Blake Cabs Limited	Hans Rodrigues	Mr.	34 Simons Town, Giyani	Krugersdorp	4627	South Africa	063 662 5515
34	34	Omar Pty	Zoe Hamman	Mrs.	124 Florence Nzama Street, Bellville	Pinetown	3730	South Africa	063 547 4637
35	35	Rajah Baking Services	Danny Allie	Mr.	44 Nico Smith, Giyani	Krugersdorp	6637	South Africa	086 466 4426

Query executed successfully. | ABEJIDEPROBOOK (12.0 RTM) | ABEJIDEPROBOOK\HP (60) | xNorthwind | 00:00:00 | 10000 rows

Fig. 3. `XNorthwind`: Output showing synthesised customers' table

VII. Conclusion and Future Work

This paper has described a new approach for the generations of datasets using CFGs. The CFG rules were implemented, and large hypothetical data records were injected into an SQL Server database called the `XNorthwind`. The synthesized datasets were stored in Microsoft SQL Server. We have shown that this approach can be used to synthesise large datasets. Evaluation results obtained through a survey showed that majority of the participants agreed that large datasets can be useful.

In future, we will extend this tool to automatically generate picture fields as seen in the Category table.

Acknowledgment

This work is based on research supported by the National Research Foundation (NRF) of South Africa (Grant Number: 119041). Any opinion, findings and conclusions or recommendations expressed in this material are those of the authors and therefore the NRF does not accept liability in regard thereto.

References

[1] M. Keith, M. Schincariol, and M. Nardone, "An in-depth guide to Java persistence APIs," in *Pro JPA 2 in Java EE 8*. Springer, 2018, pp. 1–24.

[2] G. Bell, T. Hey, and A. Szalay, "Beyond the data deluge," *Science*, vol. 323, no. 5919, pp. 1297–1298, 2009.

[3] E. Meijer and G. Bierman, "A co-relational model of data for large shared data banks," *Queue*, vol. 9, no. 3, p. 30, 2011.

[4] N. May, W. Lehner, S. Hameed, N. Maheshwari, C. Müller, S. Chowdhuri, and A. K. Goel, "SAP HANA-from relational OLAP database to big data infrastructure." in *EDBT*, 2015, pp. 581–592.

[5] Y. Luo, W. Wang, and X. Lin, "Spark: A keyword search engine on relational databases," in *24th International Conference on Data Engineering*. IEEE, 2008, pp. 1552–1555.

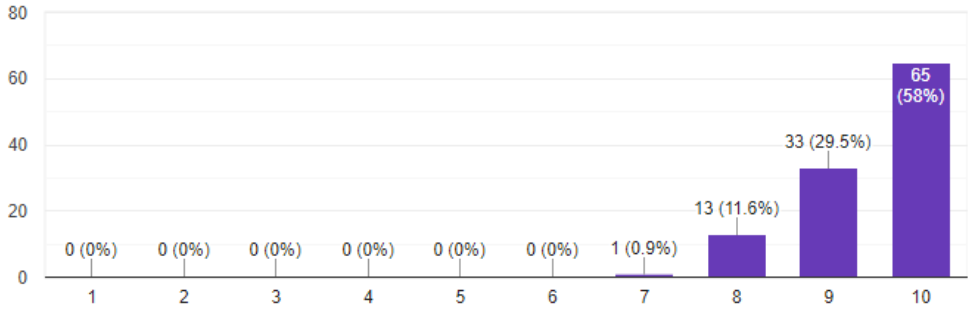
[6] R. Lumbantoruan, E. M. Sibarani, M. V. Sitorus, A. Mindari, and S. P. Sinaga, "An approach for automatically generating star schema from natural language," *Telkomnika*, vol. 12, no. 2, p. 501, 2014.

[7] J. Runtuwene, I. Tangkawarow, C. Manoppo, and R. Salaki, "A comparative analysis of extract, transformation and loading (ETL) process," in *IOP Conference Series: Materials Science and Engineering*, vol. 306, no. 1. IOP Publishing, 2018, pp. 1–8.

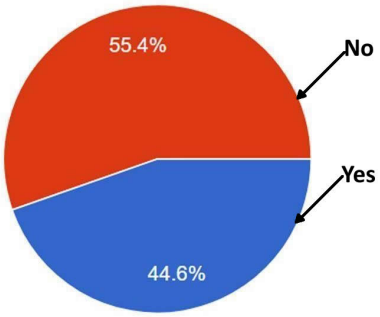
[8] R. A. Pazos R, J. J. González B, M. A. Aguirre L, J. A. Martínez F, and H. J. Fraire H, "Natural language interfaces to databases: an analysis of the state of the art," *Recent Advances on Hybrid Intelligent Systems*, pp. 463–480, 2013.

[9] J. Raissi, "IPSec offload performance," in *Proceedings of the IEEE Southeast Conference*. IEEE, 2004, pp. 222–228.

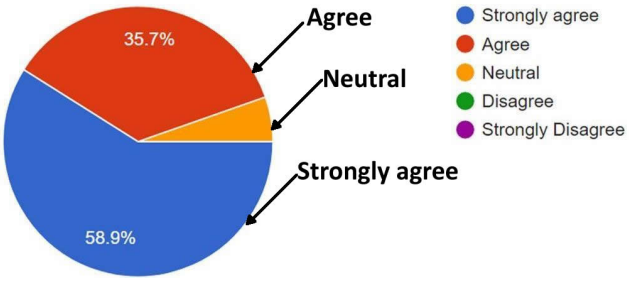
[10] H. Thakkar, D. Punjani, Y. Keswani, J. Lehmann, and S. Auer, "A stitch in time saves nine–SPARQL querying of property graphs using Gremlin traversals," *arXiv preprint arXiv:1801.02911*, pp. 1–24, 2018.



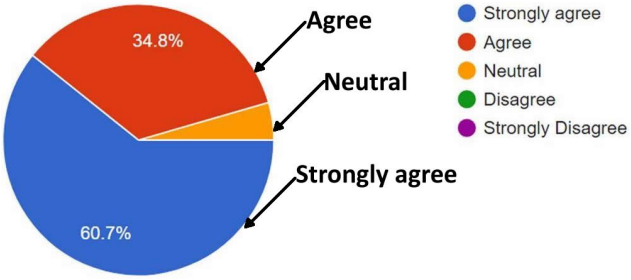
(a) Rate your experience with databases? (1 for no experience and 10 for strongly experienced)



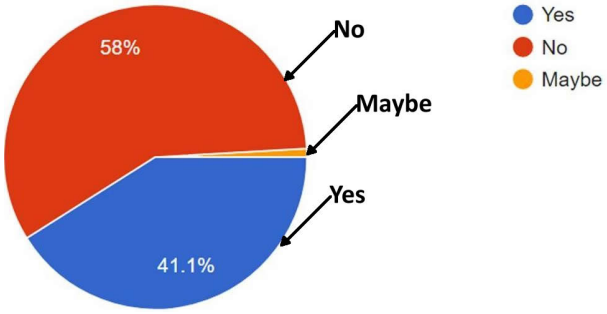
(b) Usage of the Northwind DB



(c) Usefulness of the datasets of XNorthwind



(d) Wider application of XNorthwind



(e) Think of any application of large dataset

Fig. 4. The result of the evaluation

[11] D. Dou, H. Qin, and P. Lependu, "OntoGrate: Towards automatic integration for relational databases and the semantic web through an ontology-based framework," *International Journal of Semantic Computing*, vol. 4, no. 01, pp. 123–151, 2010.

[12] R. Jennings, *Microsoft Access 2010 in depth*. Pearson Education, 2010.

[13] K. Sankar, *Fast Data Processing with Spark 2*. Packt Publishing Ltd, 2016.

[14] G. J. Fakas, B. Cawley, and Z. Cai, "Automated generation of personal data reports from relational databases," *Journal of Information & Knowledge Management*, vol. 10, no. 02, pp. 193–208, 2011.

[15] N. P. Warren, M. T. Neto, S. Misner, I. Sanders, and S. A. Helmers, *Business intelligence in Microsoft Sharepoint 2013*. Pearson Education, 2013.

[16] K.-B. Yue, "Using a semi-realistic database to support a database course," *Journal of Information Systems Education*, vol. 24, no. 4, p. 327, 2013.

[17] J. Paredaens, P. De Bra, M. Gyssens, and D. Van Gucht, *The structure of the relational database model*. Springer Science & Business Media, 2012, vol. 17.

[18] M. Levene and G. Loizou, *A guided tour of relational Databases and beyond*. Springer Science & Business Media, 2012.

[19] C. Vicknair, M. Macias, Z. Zhao, X. Nan, Y. Chen, and D. Wilkins, "A comparison of a graph database and a relational database: a data provenance perspective," in *Proceedings of the 48th Annual Southeast regional conference*. ACM, 2010, p. 42.

[20] L. Chung, "Database evolution: Microsoft Access within an organization's database strategy," *Retrieved October*, vol. 29, p. 2012, 2012.

[21] S. F. Gilani, V. V. Agarwal, J. Reid, R. Raghuram, J. Huddleston, and J. H. Pedersen, *Beginning C# 2008 databases: From Novice to Professional*. Apress, 2008.

[22] J. N. Dyer and C. Rogers, "Adapting the Access Northwind database to support a database course," *Journal of Information Systems Education*, vol. 26, no. 2, p. 85, 2015.

[23] S. Borker, "Business intelligence data warehousing," Ph.D. dissertation, Citeseer, 2006.

[24] G. S. Nelson, "Planning for and designing a data warehouse: A hands on workshop," in *Hands on Workshop presented at the SAS Global Forum Conference.*

Orlando, Florida, 2007, pp. 1–16.

[25] Microsoft, "Downloading sample databases," https://docs.microsoft.com/en-us/dotnet/framework/data/adonet/sql/linq/downloading-sample-databases, 2017, accessed: 2018-06-05.

[26] H. Angermann, Z. Pervez, and N. Ramzan, "Taxo-semantics: Assessing similarity between multi-word expressions for extending e-catalogs," *Decision Support Systems*, vol. 98, pp. 10–25, 2017.

[27] A. Gelbukh, G. Sidorov, H. Fraire *et al.*, "Prepositions and conjunctions in a natural language interfaces to databases," in *International Symposium on Parallel and Distributed Processing and Applications*. Springer, 2007, pp. 173–182.

[28] I. H. Elifoglu and A. F. Fitzsimons, "Case study in an auditing in an ODBC environment: Using Northwind data for IT Auditing," *ASBBS Proceedings*, vol. 20, no. 1, p. 136, 2013.

[29] D. Lavbič, T. Matek, and A. Zrnec, "Recommender system for learning SQL using hints," *Interactive Learning Environments*, vol. 25, no. 8, pp. 1048–1064, 2017.

[30] S. Kaddoura, R. A. Haraty, A. Zekri, and M. Masud, "Tracking and repairing damaged healthcare databases using the matrix," *International Journal of Distributed Sensor Networks*, vol. 11, no. 11, pp. 1–8, 2015.

[31] R. A. Haraty, M. Zbib, and M. Masud, "Data damage assessment and recovery algorithm from malicious attacks in healthcare data sharing systems," *Peer-to-Peer Networking and Applications*, vol. 9, no. 5, pp. 812–823, 2016.

[32] R. A. Haraty, S. Kaddoura, and A. S. Zekri, "Recovery of business intelligence systems: Towards guaranteed continuity of patient centric healthcare systems through a matrix-based recovery approach," *Telematics and Informatics*, vol. 35, no. 4, pp. 801–814, 2018.

[33] M. Nagao and H. Seki, "An FCA approach to mining quantitative association rules from multi-relational data," *International Journal of Computational Intelligence Studies*, vol. 6, no. 4, pp. 366–383, 2017.

[34] C. M. Pompiliu, S. A.-M. Ramona *et al.*, "Business intelligence integrated solutions," *Ovidius University Annals, Economic Sciences Series*, vol. 17, no. 2, pp. 185–189, 2017.

[35] J. Segovia-Aguas, S. Jiménez, and A. Jonsson, "Generating Context-free Grammars using classical planning," pp. 1–7, 2017.

[36] A. V. Aho, R. Sethi, and J. D. Ullman, "Compilers, principles, techniques," *Addison Wesley*, vol. 7, no. 8, p. 9, 1986.

[37] M. J. Pugliese and J. Cagan, "Capturing a rebel: modeling the Harley-Davidson brand through a motorcycle shape grammar," *Research in Engineering Design*, vol. 13, no. 3, pp. 139–156, 2002.

[38] I. Demir, D. G. Aliaga, and B. Benes, "Proceduralization for editing 3D architectural models," in *Fourth International Conference on 3D Vision*. IEEE, 2016, pp. 194–202.

[39] Y. Dehbi, F. Hadiji, G. Gröger, K. Kersting, and L. Plümer, "Statistical relational learning of grammar rules for 3D building reconstruction," *Transactions in GIS*, vol. 21, no. 1, pp. 134–150, 2017.

[40] M. H. Christensen, "Structural Synthesis using a Context-free design Grammar approach," in *International Conference of Generative Art*, 2009, pp. 104–109.

[41] A. Ade-Ibijola, "Synthesis of social media profiles using a probabilistic Context-free Grammar," in *Pattern Recognition Association of South Africa and Robotics and Mechatronics, 2017*. IEEE, 2017, pp. 104–109.

[42] A. C. Lin, "Network Analysis with Stochastic Grammars," Air Force Institute of Technology Wright-Patterson AFB OH Graduate School of Engineering and Management, Tech. Rep., 2015.

[43] S. Pudaruth, S. Amourdon, and J. Anseline, "Automated generation of song lyrics using CFGs," in *Seventh International Conference on Contemporary Computing*. IEEE, 2014, pp. 613–616.

[44] A. Ade-Ibijola, "FINCHAN: A grammar-based tool for automatic comprehension of financial instant messages," in *Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists*. ACM, 2016, p. 1.

[45] S. Velupillai, D. Mowery, B. R. South, M. Kvist, and H. Dalianis, "Recent advances in clinical natural language processing in support of semantic analysis," *Yearbook of medical informatics*, vol. 10, no. 1, p. 183, 2015.

[46] P. Liang, "Learning executable semantic parsers for natural language understanding," *Communications of the ACM*, vol. 59, no. 9, pp. 68–76, 2016.

[47] W. Dyrka and J.-C. Nebel, "A stochastic Context-free Grammar based framework for analysis of protein sequences," *BMC bioinformatics*, vol. 10, no. 1, p. 323, 2009.

[48] E. Butler, K. Siu, and A. Zook, "Program synthesis as a generative method," in *Proceedings of the 12th International Conference on the Foundations of Digital Games*. ACM, 2017, p. 6.

[49] A. Ade-Ibijola, "Syntactic generation of practice novice programs in Python," in *Annual Conference of the Southern African Computer Lecturers' Association*. Springer, 2018, pp. 158–172.

[50] D. Macko and K. Jelemenská, "HDL model verification based on visualization and simulation," in *Proceedings of the World Congress on Engineering*, vol. 2, 2012, pp. 1–6.

[51] M. Fanaswala and V. Krishnamurthy, "Detection of anomalous trajectory patterns in target tracking via stochastic Context-free Grammars and reciprocal process models," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 1, pp. 76–90, 2013.

[52] H. Liao, Z. Xu, and X.-J. Zeng, "Hesitant fuzzy linguistic VIKOR method and its application in qualitative multiple criteria decision making," *IEEE Transactions on Fuzzy Systems*, vol. 23, no. 5, pp. 1343–1355, 2015.

[53] H. Wang, Z. Xu, and X.-J. Zeng, "Hesitant fuzzy linguistic term sets for linguistic decision making: Current developments, issues and challenges," *Information Fusion*, vol. 43, pp. 1–12, 2018.

[54] A. Adem, A. Çolak, and M. Dağdeviren, "An integrated model using SWOT analysis and hesitant fuzzy linguistic term set for evaluation occupational safety risks in

life cycle of wind turbine," *Safety science*, vol. 106, pp. 184–190, 2018.

[55] E.-R. Olderog, "Space for traffic manoeuvres: An overview," in *Symposium on Real-Time and Hybrid Systems*. Springer, 2018, pp. 211–230.

[56] K. A. Buragga and N. A. Zafar, "Formal parsing analysis of Context-free Grammar using left most derivations," in *International Conference on Software Engineering Advances*, 2011.

[57] A. Sellink and C. Verhoef, "Scaffolding for software renovation," in *Proceedings of the Fourth European of Software Maintenance and Reengineering*. IEEE, 2000, pp. 161–172.