AI

MDPI

Article

A Natural Language Interface to Relational Databases Using an Online Analytic Processing Hypercube

Fadi H. Hazboun, Majdi Owda and Amani Yousef Owda *

Department of Natural, Engineering and Technology Sciences, Arab American University, Ramallah P600, Palestine; f.hazboun@student.aaup.edu (F.H.H.); majdi.owda@aaup.edu (M.O.)
* Correspondence: Amani.Owda@aaup.edu

Abstract: Structured Query Language (SQL) is commonly used in Relational Database Management Systems (RDBMS) and is currently one of the most popular data definition and manipulation languages. Its core functionality is implemented, with only some minor variations, throughout all RDBMS products. It is an effective tool in the process of managing and querying data in relational databases. This paper describes a method to effectively automate the conversion of a data query from a Natural Language Query (NLQ) to Structured Query Language (SQL) with Online Analytical Processing (OLAP) cube data warehouse objects. To obtain or manipulate the data from relational databases, the user must be familiar with SQL and must also write an appropriate and valid SQL statement. However, users who are not familiar with SQL are unable to obtain relevant data through relational databases. To address this, we propose a Natural Language Processing (NLP) model to convert an NLQ into an SQL query. This allows novice users to obtain the required data without having to know any complicated SQL details. The model is also capable of handling complex queries using the OLAP cube technique, which allows data to be pre-calculated and stored in a multi-dimensional and ready-to-use format. A multi-dimensional cube (hypercube) is used to connect with the NLP interface, thereby eliminating long-running data queries and enabling self-service business intelligence. The study demonstrated how the use of hypercube technology helps to increase the system response speed and the ability to process very complex query sentences. The system achieved impressive performance in terms of NLP and the accuracy of generating different query sentences. Using OLAP hypercube technology, the study achieved distinguished results compared to previous studies in terms of the speed of the response of the model to NLQ analysis, the generation of complex SQL statements, and the dynamic display of the results. As a plan for future work, it is recommended to use infinite-dimension (n-D) cubes instead of 4-D cubes to enable ingesting as much data as possible in a single object and to facilitate the execution of query statements that may be too complex in query interfaces running in a data warehouse. The study demonstrated how the use of hypercube technology helps to increase system response speed and process very complex query sentences.

Keywords: OLAP hypercube; SQL; natural language processing; natural language query; data warehouse

Citation: Hazboun, F.H.; Owda, M.; Owda, A.Y. A Natural Language Interface to Relational Databases Using an Online Analytic Processing Hypercube. *AI* **2021**, *2*, 720–737. https://doi.org/10.3390/ai2040043

Academic Editor: Amir Mosavi

Received: 17 November 2021
Accepted: 10 December 2021
Published: 18 December 2021

1. Introduction

Natural Language Processing (NLP) is an important part of Artificial Intelligence (AI) used to create intelligent models that simulate human thinking. Due to its advanced capabilities, it can reduce the gap between machines and humans [1]. The primary goal of processing NLQs is a model that is able to translate English sentence structures [2] into processable machine code.

NLP has been used in developing systems to translate Natural Language (NL) sentences into SQL [3,4]. A query can be entered in natural language by the user. When the user enters the query in English, it is then translated into an SQL query [5]. There are many

difficulties in converting NLQs to SQL queries, such as complexity, which implies that a single term may have several meanings. In this case, a single word may be mapped to several meanings [5,6]. Another difficulty is the development of complex SQL queries that are executed on multiple database objects [7].

For users who are not fully familiar with complex database query languages such as SQL, accessing databases via questions in NL and obtaining the results of the query in an understandable format are not easy tasks [8]. These systems are intended for users who work with databases but lack SQL experience. Several studies have examined various solutions using NLP interfaces, and this remains an interesting research field. The focus of this research is NLP methods and their conversion into SQL statements, in addition to the challenges associated with big data, which may hinder the simulation that takes place between the natural language interface (NLI) and large databases. The volume of data produced globally each day is increasing tremendously. At the current pace, 2.5 quintillion bytes of data are being generated per day; however, this pace is accelerating as a result of the growth of the Internet of Things, the spread of social networking sites, and the interaction of people with digital services. The volume of data increased by 90% during the past two years, and this growth is of interest to researchers [9,10]. Organizations cannot function without data and, due to their quantity and rapid rate of generation, data have become the fuel that drives these organizations. These data are accumulated in huge datasets, collectively referred to as "big data", which must be analyzed to enhance decision making. However, organizations face challenges regarding big data. These include data quality, data storage, a lack of data science professionals, data validation, and the aggregation of data from various sources. One of the most pressing challenges for big data is the correct storage of these huge datasets to enable easy access and handling. Thus, studies have sought suitable solutions for storing large quantities of data in relational databases (RDBs) in a structured manner that facilitates the processes of managing and retrieving data [11].

The main challenge that NLI-RDBs developers continue to face is the automatic mapping or conversion of complex NLQs into SQL queries, particularly given the huge amount of data distributed in database tables and the challenges of dynamically displaying query results.

The systems reviewed in this paper proposed algorithms to handle English language queries made by a user to obtain an SQL query built using a number of techniques [3,12,13]. The NLIDB-OLAP novel architecture proposed in this paper built upon three main pillars. Firstly, several methodologies are used to process natural language for the interaction of the model with the user and to provide an effective analysis of the meanings and etymologies of the entered phrases, as follows:

1. Tokenize module;
2. Lemmatized and Stop-Word module;
3. Lexical module;
4. Semantic module;
5. POS_tagging module;
6. Syntactic module.

Secondly, the model interacts with the user and extracts data in real time and displays it dynamically. Thirdly, the key novelty of the proposed NLIDB-OLAP is the use of the OLAP hypercube to enable processing of a huge amount of data, which is a challenge in relational databases when using complex SQL query statements. The proposed solution in this study was to develop an interface system in which NLP techniques and data retrieval techniques are used to build and execute SQL statements, in addition to the usage of an OLAP hypercube [14,15]. The main contribution of this study is utilizing the idea of having an OLAP hypercube table which is based on unifying and coordinating data in a single multi-dimensional data warehouse object called the hypercube table. The hypercube table is queried, rather than information being distributed in more than one table, which requires complex and joint SQL statements. The model workflow is shown in Figure 1 [16].

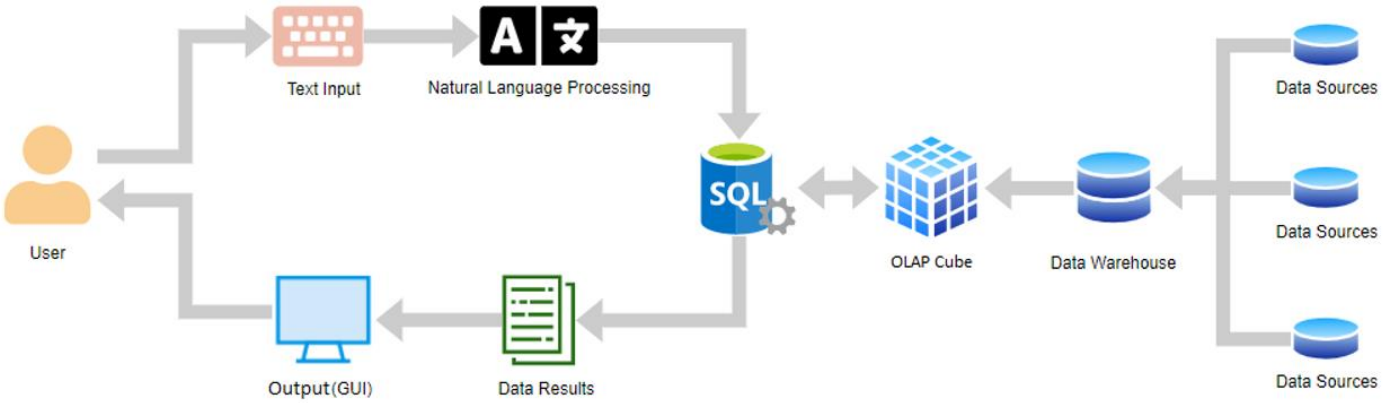

Figure 1. Model workflow.

The current research focused on providing solutions for developing the natural language interface and relational databases with OLAP hypercube technology. The aim was to enable the processing of large quantities of data and complex SQL statements that are difficult to use, especially by users who have no knowledge of executing SQL query statements. This technology will also help companies with big data provide automated chat services with customers and an integrated query system capable of addressing all database inquiries.

This paper is organized as follows: Section 2 introduces the main concepts of OLAP and the related literature. Section 3 discusses the related studies and other researchers' challenges in the same field. Section 4 outlines the proposed NLIDB-OLAP model workflow, and the algorithm structure is presented in Section 5. Section 6 presents the results and analysis. Section 7 provides the conclusion and discusses the scope of future research.

2. Online Analytical Processing (OLAP)

The term Online Analytical Processing (OLAP) first appeared in a 1993 whitepaper published by Arbor Software and was coined by Edgar F. Codd [17], the discoverer of relational databases. This product was introduced to the market as an "extended spreadsheet database" [17]. OLAP [18] is an organizational mechanism that enables users to query and extract data in an easy and fast manner to enable different forms of analysis. OLAP plays an important role in accounting and financial reports for large data that need a large amount of time to be processed [19]. The process of querying big data distributed in multiple tables results in complex [18], joint and overlapping SQL statements to perform a correct and accurate query of the data [15]. OLAP data are collected from multisource production environment databases, transferred to data warehouses and then cleaned and organized into data cubes to significantly improve query time. The idea of using the infinite OLAP cube n-D began with a study [19] which was confined to data mining technology; however, the author did not address the linkage of using a multidimensional OLAP hypercube table with NLP interfaces. The work in this paper explores the use of an OLAP hypercube table to help in developing an interface system that offers a better performance in translating NLQ to SQL.

3. Literature Review

NLP research began in the late 1940s when the idea of machine translation (MT) was investigated in 1946, with Weaver and Booth implementing the first natural language program on machine translation to crack codes during World War II [20]. After that, most of the systems that were created from this perspective were based on searching in the dictionary for the appropriate words for translation and rearranging words to suit the rules of word order of the synonymous language. This was carried out without looking at the lexical ambiguity inherent in the natural language, as it led to inaccurate and bad results [20]. This prompted researchers to find solutions more appropriate to languages,

which witnessed a significant development in the 1960s in the production of primary NLP systems [20–22]. By the 1970s, applications of NLP had developed dramatically, as rhetorical documents were used to create response-generator text meta descriptions such as McKeown's discourse planner, TEXT [23], and McDonald's response generator, MUMMBLE [24]. By the 1980s, the concept of natural language had expanded and there was a growing realization of the need to find solutions to the limitations of natural language programming and a general push towards applications that worked with the language in a broad real-world context. Natural language programming grew rapidly from that time until it underwent a major transformation in the early 1990s with the transition to relying on empirical methodologies versus the introspective generalizations that characterized the Chomsky era which had an impact on theoretical linguistics [25]. Many applications in our modern world require dealing with natural language interfaces with relational databases; research and development in this field continue to progress at a fast pace to provide the best user experience and perfect solutions to handling complex queries and large data volumes [26]. A study entitled "Text-to-SQL Generation for Question Answering on Electronic Medical Records" [27] aimed to provide services related to health care that are asked by patients in the form of queries through databases, so that these questions are translated into medical inquiries, and then, responses are made from medical records entered into the databases, where the questions are related to several tables, which requires complexity in query strings that may produce false results [27]. Their model was built on huge data that were divided into several tables due to the large medical data volume, and the correct retrieval of information was complicated due to complex medical terminology; the author processed and cleaned the data stored in databases for easy query processing and retrieval of data according to the questions asked and gave the correct answers. The study in [28], "Natural language Interface for Database: A Brief review" provided about an introduction to natural language interfaces and their processing of databases to provide an intelligent data system; this paper discussed the shortcomings in understanding natural language to achieve widespread language interfaces and their association with databases. The authors of [29] modeled SQL query logs as a query part diagram to improve the ability of language interfaces, access information within log information pipes, match words and terms used by the user, and enter them through the system according to their proposed NLI-DBS system to enhance the performance of language interfaces with the limitations of poor accuracy in converting NLQ to SQL, a bad effect of user sessions in an SQL query log, data matching confusion, and no ways were found to improve existing deep learning from start to finish.

The study in [30] aimed to take appropriate action using a computer by interpreting a sentence in natural language by processing and translating it accurately to extract data and summarize information from multiple data sources according to users' requests. SQL statements are passed to databases, and their results are displayed to the user through applications prepared for this purpose. The model handles simple queries and the search does not provide solutions to a complex query; not using a dictionary leads to inaccurate data extraction, and the fixed layout used limited the data result with fixed values. The experimental outcomes in [31] present new methods for analyzing a research dataset based on NLP and relational data analysis, and they propose to implement relational queries to investigate the possibility of using certain software tools that allow wide NLP references to deal with relational databases. The focus of the study in [31] was to limit the work to lexical analysis and the prediction of query sentences and successive words' weaknesses in text processing at the natural language level, which requires focusing on the development of additional query tools. In [24], the provision of accurate information to users from a railway database was studied, including the provision of seats or travel destinations. The paper aimed to develop a model that uses NLP by entering a question in natural language and it being analyzed by the model and converted into SQL query sentences and data extraction according to the required query, coding, and analysis. The mapping was used to provide the user with a comprehensive view of the available data with the

limitation of querying only one statement without addressing the fetching of more data through multiple queries. Enhancing the Detection of Criminal Organizations in Mexico using machine learning (ML) and NLP was introduced in [32]. The use of NLP helped to extract encoded information from criminal groups on a daily basis in a systematic and reliable manner. The dictionary of actors used in this application was robust and included a comprehensive list of the most relevant criminals and their organizations. The natural language (NL) and NLP applications presented in this study provide strong empirical foundations for advancing objective academic and political analysis to better understand and monitor organized criminal activity. The authors in [33] discussed Aspect-Based Sentiment Analysis using NLP, which is characterized by three classifications. The authors explored both praise and criticism in post. The study in [34] proposed a standard to help big data OLAP designers choose the cube design best suited to their goals; the study identified the main requirements and trade-offs for designing a big data OLAP effectively. Cubes take advantage of data pre-clustering techniques. In [35], OLAP was used to tweak the path data that includes storing the sawn data at a specific time and point. The researchers intended to propose this technique due to the large volume of multi-dimensional data and analytical queries, which have three times the analysis capacity than ordinary data storage. The researchers in [36] presented a study on social robots, specifically in the tourism sector. Through NLP in Python, the researchers developed the detection of feelings in the text by converting speech to text, which allowed one to study the feelings of visitors to obtain results evaluating artworks in a museum. The authors in [37] presented a study on the design of a decision support system for an aviation engine control system; the results indicate that fact tables do not provide any information on how to group records when calculating the data aggregation, where they adopted the implementation of the hypercube as a separate database to support functional dependency, and multiple elements combined into one using a specially selected function union that provides the most efficient access to data in terms of speed. The researchers in [38] concluded that a joint query would also not produce the desired result, since joint queries must have concurrent schemas (groups of attributes after the SELECT operator); thus, it is necessary to specify all relationships with these attributes in all queries that are standardized. The OLAP hypercube solves this problem automatically without additional user efforts by using a joint query. In [39], a protocol designed to build and maintain a hypercube structure in a dynamic environment was designed, and it was found that implementing HyperD results in savings, especially in those networks where the ripple rate is small relative to the overall network activity. In addition, the network is more resilient to failures and bottlenecks. In [40], an examination of the relative costs of improving queries on databases by providing solutions about the structure of databases and their impact on network load was presented. The authors concluded that hypercube offers distinct cost advantages over stochastic topologies for query optimization. In [41], hypercube helped in business intelligence fields and created an instance of spatial data cubes that shared common dimensional levels. It demonstrated that data cube descriptive models can be used for the easy integration of heterogeneous data and SOLAP navigation in complex towers of data cubes. In [42], the study proved that the combination of both OLAP and data mining provides excellent solutions. OLAP mining databases enhance data mining and analysis capabilities directly into the database server. The paper also provided a brief introduction to these techniques and compared the data warehouse models the STAR schema and the Snowflake schema.

The development of modern technologies for communications and Internet networks has provided many solutions and facilities for humans to deal with various types of fields through the Internet or local networks [19]; this matter led to an increase in the volume of interactions with the technological environment and a huge increase in data [34,43]. These technologies helped people to work through digital interfaces. One of the most important interfaces are natural language interfaces that allow users to interact with the computer using a human language [29]. This study [44] discussed the theory of using an OLAP hypercube to deal with data, as the theory was successful in reducing the time to execute

the query. This study did not deal with building natural language interfaces, hence why our study aimed to build a query interface through natural language processing and query execution through an OLAP hypercube.

The research presented in this paper focuses on providing solutions for the development of the natural language interface along with relational databases with OLAP hypercube technology. This novelty integration will allow a very complex SQL statement to be processed on a huge amount of data with a very high speed of execution, which was difficult to deal with through natural language query interfaces in previous studies.

4. Methodology

4.1. Proposed Model

Keeping data inside databases allows us to retrieve the data at any time. Retrieval of data by a layman is very difficult using traditional methods because it requires a lot of effort and a strong knowledge of database architecture. The database management system can handle natural language queries through standard database languages [31].

The goal of our proposed work was to build an interface to provide a facility that enables the user to enter his/her query in English, which will be processed by several units by using Python Natural Language Tool Kit (NLTK) and other Python libraries to form an equivalent SQL query to be executed on a multi-dimensional OLAP cube and display the query results dynamically, so that only required data are displayed in the query statement. Figure 1 shows model workflow.

4.2. Multi-Dimensional OLAP 4D-Cube (Hypercube)

OLAP technology is implemented to facilitate model work and increase responsiveness around the results of SQL query statements [34]. Data are first collected from multiple data sources, stored in data warehouses, and then cleaned and organized into a data cube. Our four-dimensional OLAP cube was designed for university professors and their details such as name, gender, job title, and department they work in, as well as contact details. In addition, it contained information regarding their research interests, projects, publications, and Ph.D. students. The dimensions were populated with data that were organized hierarchically that was obtained from [45]. OLAP cubes are often pre-summarized across dimensions to significantly improve query time across relational databases.

A four-dimensional cube (hypercube) illustrated in Figure 2B was built, and data were filled in it; then, we could easily obtain all the relational information regarding the professor, publications, departments, and students by one SQL query on one table alone without having to perform a query from more than one table, as shown in Figure 2B, which saves a lot of time and avoids writing complex SQL statements [44].

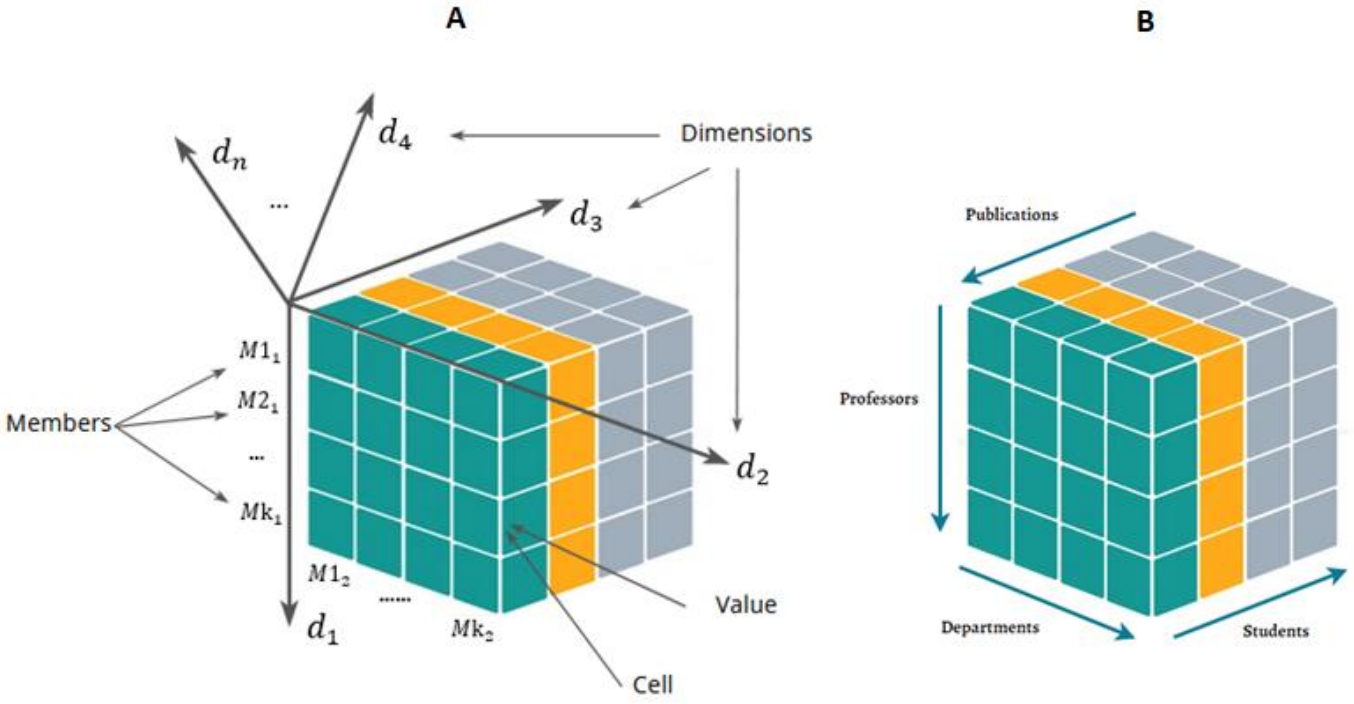

Figure 2. The hypercube sequence (**A**) and the 4-D hypercube (**B**).

To implement the hypercube, the sequence theory was applied for the dimensions to describe the members, as shown in Figure 2A; the following values were used according to the sequence theory [37,39,41,44]:

D—the combination of the dimensions, but if we consider d_i to be the members of the dimension combinations, then we have $d_i \in D$, that is $D = \{d_1, d_2, d_3, d_4, \ldots, d_n\}$.

From this, n is a number of dimensions. $d_1, d_2, d_3, d_4, \ldots, d_n$ are members of the dimension combination that are placed on the axis of hypercube [37,44].

Each member of the dimensional combination consists of internal members placed according to the coordination axes of the hypercube. So, according to the internal members of the hypercube dimensions, we applied the following values: $M_d = \{M1_i, M2_i, M3_i, \ldots, Mk_i\}$—the internal combination of dimensions (members); $i = 1 \ldots n$ is the combination of values in the dimensions.

Accordingly, every dimension $d_1 = \{M1_1, M2_1, M3_1, \ldots, Mk_1\}$, $d_2 = \{M1_2, M2_2, M3_2, \ldots, Mk_2\}, \ldots \ldots, d_n = \{M1_n, M2_n, M3_n, \ldots, Mk_n\}$, consists of the combination of internal members. Here, k_i is the value of internal members of every dimension being taken. Taking M as the members of the cube dimensions, all the internal members of dimensions can be shown through the connection ($\cup$), $M = \{Md_1 \cup Md_2 \cup Md_3 \cup \ldots \cup Md_n\}$ [39].

The members' value of every dimension d_1 is the members' combination of dimension Md_1, the value of members d_2 is the members' combination of dimension Md_2, k represents the sequence number of the member for dimension, so $K_1 = Md_1$, and the values of members $K_n = Md_n$, as illustrated in Figure 2A.

In Figure 2B, each dimension represents an entity, d_1 represents departments, d_2 represents professors, d_3 represents students, and d_4 represents publications. In addition, each dimension member represents an attribute: $M1_1$ represents the first attribute (department id) in dimension d_1 (departments). The analysis can be performed by four types of OLAP analytic operations against a multidimensional object [46,47], namely:

1. Slicing;
2. Dicing;
3. Drill-Down/Up;
4. Roll-Up.

The hypercube illustrated in Table 1 was built by using the Oracle Analytic Workspace Manager (AWM) tool through the following steps:

- Create a cube and its dependent components including dimensions and members.
- Map the OLAP model to source data.
- Enable the materialized view rewrite to the cube.
- Load data into the dimensions and measures.
- Load and view OLAP cube data.

4.3. From NLP to SQL

The following steps illustrated in Figure 3 were taken to build the model and convert the natural language into SQL statements to be executed on databases:

Step 1: Tokenize Module

At this stage, the system implements tokenization on the entered query sentence by separating it into single words, each word representing a unique symbol "Token". Then, these words are stored in a separate list and passed to the Lemmatized and Stop-Word module [27]. The NLTK library was used in order to tokenize the input; further details will be explored with an example in the next walkthrough practical section.

Step 2: Lemmatized and Stop-Word Module

The system performs lemmatization on the output of the tokenized module. In addition, the stop-word module removes all the unwanted words from the list using the ignore list. Then, this is stored in a separate list and passed to the lexical module [3,48].

Step 3: Lexical Module

The lemmatized list is mapped with the dictionary. In this step, these words are replaced with words from the database dictionary and passed to syntactic analysis [49].

Step 4: Semantic Module

The system finds words that are considered as symbols and conditions, and then, the word is selected from the dictionary. (For example: If there is "less than or equal to" in the query, it is replaced with the symbol "<=") [50,51].

Step 5: POS Tagging Module

Parts of Speech tagging of tokens is carried out here. The tag signifies whether the word is a noun, adjective, or verb [52].

Step 6: Syntactic Module

At this point, the dictionary of attributes, keywords, and table names is preserved. Each encoded word is assigned an attribute in the dictionary [53].

Step 7: SQL Query Generation Module

In this module, the SQL query is generated using the output of the syntactic module. A walkthrough is highlighted with an example in the next section.

Table 1. OLAP hypercube table architecture.

D1		D2				D3		D4	
Department ID	**Department Name**	**PROF. ID**	**PROF. NAME**	**JTITLE**	**PHONE**	**STUDENTID**	**STUNAME**	**PUBID**	**PUBTITLE**
1	Mathematics and Computation	1	Majdi Owda	Senior Lecturer	4401612471520	2	Pei Lee	1	Template-Based Information Extraction System for Detection of Events on Twitter
								2	Information Extraction from Big Social Data
						7	John Staven	6	Conversation-Based Natural Language Interface to Relational Databases
								5	Crime Prevention on Social Networks Featuring Location Based Services
		5	David McLean	Senior Lecturer	4401612471536	17	Justen	9	An Adaptation Algorithm for an Intelligent Natural Language Tutoring System
								12	NLP Interfaces along with OLAP
		7	Lida Nejad	Principal Lecturer	970599255888	24	Liza N.	0	No Publication Yet
						25	Kojo B.	0	No Publication Yet
2	Digital Media Entertainment Technology	2	Keeley Crockett	Reader	4401612471497	8	Mohammed Kaleem	14	A Multi-Classifier Approach to Dialogue Act Classification Using Function Words.
								8	A Multi-Classifier Approach to Dialogue Act Classification Using Function Words.
								11	An Adaptation Algorithm for an Intelligent Natural Language Tutoring System
								7	Conversation-Based Natural Language Interface to Relational Databases
								0	No Publication Yet
						18	Rania	0	No Publication Yet
						19	Fuad	0	No Publication Yet
						26	Nawal K.	0	No Publication Yet
		3	James OShea	Principal Lecturer	4401612471546	21	Rani H.	0	No Publication Yet
						16	Samia	13	A Multi-Classifier Approach to Dialogue Act Classification Using Function Words.
						15	Huda	13	A Multi-Classifier Approach to Dialogue Act Classification Using Function Words.
						14	Carolina	13	A Multi-Classifier Approach to Dialogue Act Classification Using Function Words.
		6	Annabel Latham	Lecturer	4401612471495	19	Fuad	10	An Adaptation Algorithm for an Intelligent Natural Language Tutoring System
						22	Ruti K.	10	An Adaptation Algorithm for an Intelligent Natural Language Tutoring System
						20	Justen P.	10	An Adaptation Algorithm for an Intelligent Natural Language Tutoring System

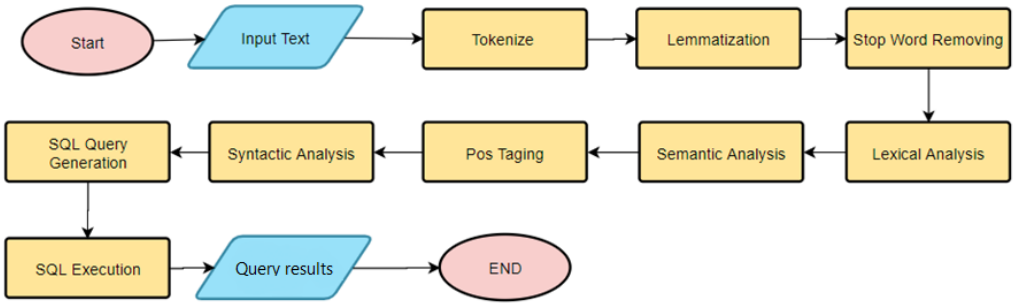

Figure 3. Model architecture.

5. Walkthrough Practical Example

The following walkthrough takes the reader through a practical example to create an SQL query from a natural language query input: a natural language query in the English language. Figure 4 displays the techniques workflow to convert NLQ to SQL.

Step 1: Query entered by the user in natural language (text) is stored in a string.

Example. *fetch all the information of the professors who have department number is equal to one.*

Step 2: Divide the sentence into single words (tokens) by the spaces between the words in the sentence. These codes are saved in a separate list.

The token for the above query is as follows.

['fetch', 'all', 'the', 'information', 'of', 'the', 'professors', 'who', 'have', 'department number', 'is equal', 'to', 'one'].

Step 3: We use lemmatization to get the lemma of the tokens generated.

['fetch', 'all', 'the', 'information', 'of', 'the', 'professor', 'who', 'have', 'department number', 'is equal', 'to', 'one'].

Step 4: Only important tokens are selected for better processing after the tokens are compared with the ignore list. This is carried out through named entity extraction; a dictionary is implemented and consists of data within the database and metadata relating to building SQL statement architecture.

['fetch', 'all', 'professor', 'who', 'department number', 'equal', 'one'].

Step 5: Specific tokens are mapped with database hypercube words; depending on the important tokens from step 4, this mapping is carried out through the dictionary that contains all the words in the database hypercube and words within the SQL statement structure. Then, tokens are replaced by their equivalents.

['SELECT', '*', 'FROM 'prof'', 'WHERE', 'equal', '1', 'depid'].

Step 6: Specific values or conditions are determined by the dictionary of conditions that are applied by the algorithm if any conditions are defined and selected according to the correct SQL statements architecture. (eg. replacing 'equal' with "=").

['SELECT', '*', 'FROM prof ', 'WHERE', '=', '1', 'depid'].

Step 7: Parts of Speech tagging of tokens is carried out by using Python NLTK library tools. The tag is a part-of-speech tag, which signifies whether the word is a noun, adjective, verb, or other.

[('SELECT', 'NNP'), ('*', 'NNP'), ('FROM prof', 'NNP'), ('WHERE', 'NNP'), ('=', 'VBZ'), ('1', 'CD'), ('depid', 'NN')].

Step 8: Syntax analysis checks the text for meaningfulness compared to the rules of formal grammar by using NLTK.

['SELECT','*','FROM prof ','WHERE',' depid ','=','1',].

Step 9: SQL query generation.

SELECT * FROM prof WHERE depid = 1.

Step 10: The SQL query is executed on the database and user provided with output.

Model Screen Shots

The model query interface illustrated in Figure 4 shows the entry of the query text through the user and how the model processes the entered query text and transforms it into an SQL query statement.

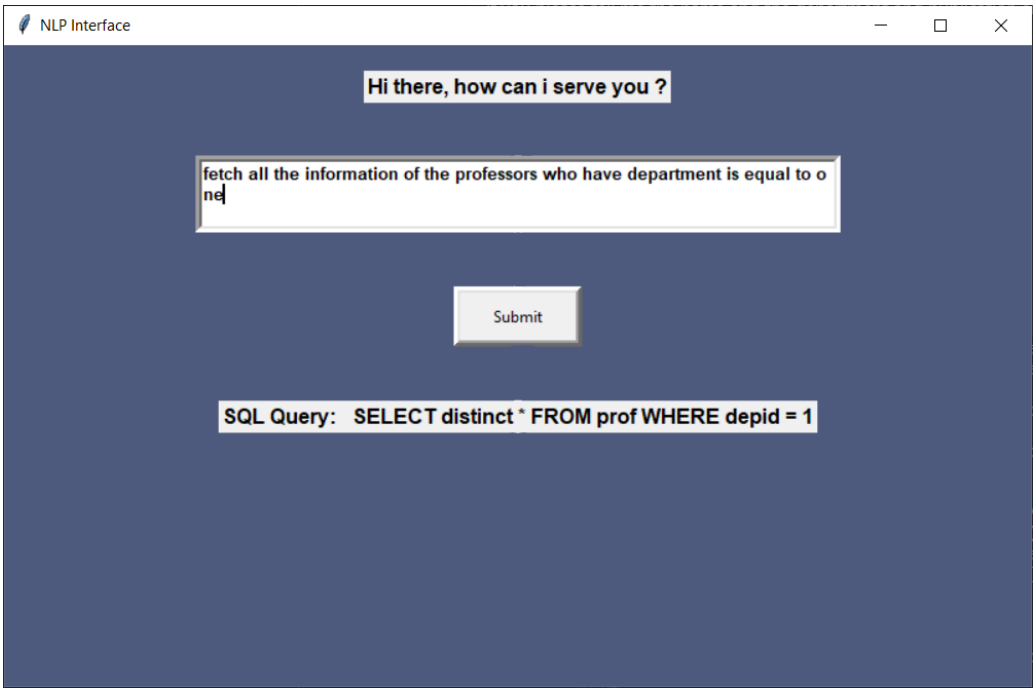

Figure 4. NLP interface query example (1).

Table 2 shows how the model displays the query results dynamically. The model showed all information related to the professors according to the query's request.

The results in Figure 5 show a user-specific query about the professor's name information, the professor's department, publications, and the affiliated students under the professor's supervision for the professor id number equal to one. In addition, it shows the model's processing of the compound query statement and its transformation into an SQL query.

Table 3 shows how the model displays the query results dynamically. In this table, the model only showed the information specified in the query statement (professor name, professor ID, professor publication, professor title, affiliated students).

Table 2. Example (1): query dynamic screen results.

NAME	DEPID	PUBTITLE	JTITLE	STUNAME
Majdi Owda	1	Template-Based Information Extraction System for Detection of Events on Twitter	Senior Lecturer	Pei Lee
Majdi Owda	1	Information Extraction From Big Social Data	Senior Lecturer	Pei Lee
Majdi Owda	1	Crime Prevention on Social Networks Featuring Location Based Services	Senior Lecturer	John Staven
Majdi Owda	1	Conversation-Based Natural Language Interface to Relational Databases	Senior Lecturer	John Staven

Figure 5. NLP Interface query dealing with another example (2).

Table 3. Example (2): dynamic screen results.

ID	NAME	DEPID	JTITLE	PHONE	DEPNAME	PUBID	PUBTITLE	STUID	STUNAME
2	Keeley Crockett	1	Reader	4401612471497	Mathematics and Computation	0	No Publication Yet	18	Rania
3	James OShea	1	Principal Lecturer	4401612471546	Mathematics and Computation	13	A Multi-Classifier Approach to Dialogue Act Classification Using Function Words.	14	Carolina
1	Majdi Owda	1	Senior Lecturer	4401612471520	Mathematics and Computation	6	Conversation-Based Natural Language Interface to Relational Databases	7	John Staven
5	David McLean	1	Senior Lecturer	4401612471536	Mathematics and Computation	12	NLP Interfaces along with OLAP	17	Justen
1	Majdi Owda	1	Senior Lecturer	4401612471520	Mathematics and Computation	1	Template-Based Information Extraction System for Detection of Events on Twitter	2	Pei Lee
1	Majdi Owda	1	Senior Lecturer	4401612471520	Mathematics and Computation	2	Information Extraction From Big Social Data	2	Pei Lee
2	Keeley Crockett	1	Reader	4401612471497	Mathematics and Computation	0	No Publication Yet	8	Mohammed Kaleem
7	Lida Nejad	1	Principal Lecturer	970599255888	Mathematics and Computation	0	No Publication Yet	24	Liza N.
7	Lida Nejad	1	Principal Lecturer	970599255888	Mathematics and Computation	15	Models of Dense Cores in Translucent Regions of Low Mass Star Formation	23	Rand J. K.
2	Keeley Crockett	1	Reader	4401612471497	Mathematics and Computation	0	No Publication Yet	26	Nawal K.
3	James OShea	1	Principal Lecturer	4401612471546	Mathematics and Computation	13	A Multi-Classifier Approach to Dialogue Act Classification Using Function Words.	15	Huda
6	Annabel Latham	1	Lecturer	4401612471495	Mathematics and Computation	10	An Adaptation Algorithm for an Intelligent Natural Language Tutoring System	22	Ruti K.

Table 3. *Cont.*

ID	NAME	DEPID	JTITLE	PHONE	DEPNAME	PUBID	PUBTITLE	STUID	STUNAME
1	Majdi Owda	1	Senior Lecturer	4401612471520	Mathematics and Computation	5	Crime Prevention on Social Networks Featuring Location Based Services	7	John Staven
5	David McLean	1	Senior Lecturer	4401612471536	Mathematics and Computation	9	An Adaptation Algorithm for an Intelligent Natural Language Tutoring System	17	Justen
7	Lida Nejad	1	Principal Lecturer	970599255888	Mathematics and Computation	0	No Publication Yet	25	Kojo B.
3	James OShea	1	Principal Lecturer	4401612471546	Mathematics and Computation	13	A Multi-Classifier Approach to Dialogue Act Classification Using Function Words.	16	Samia

6. Results and Analysis

The applications of NLP in SQL query generation are still a challenging area [54,55]. In this study, we focused on the high performance, accuracy, and responsiveness of SQL query generation. The proposed model in this paper was implemented using the NLP library in Python using the features provided such as syntactic, semantic analysis, NumPy arrays, tokenization, lemmatized stop-word removal, lexical, POS tagging, as well as the use of dictionaries and grammar for such analysis performed using the natural language to SQL query conversion tool connected on the Oracle 19c database by using CX_Oracle connector Along with a list of custom query words to form an SQL query, the execution was executed with a set of English sentences and generated SQL queries.

By adopting OLAP hypercube technology, the query can be executed on one table only, and thus the query process shortens the great effort in executing the query from several tables. The advantages of this model are the speed and the accuracy of the data response. In addition, the scheduled results are illustrated in Table 4.

In Table 4, the SQL_EXEC_TIME is the time in milliseconds that is taken to execute the SQL query on the databases, and TOTAL_EXEC_TIME is the time in milliseconds that the model takes to process NLQ on DBMS and retrieve the data and display the results.

The results in Table 4 were extracted by calculating the time in seconds that it took to perform the analysis of processing as follows:

1. Calculate the time taken to process the natural language, configure the query, execute, and display the results;
2. Calculate the time taken to execute SQL statements on databases.

The results showed that the model has a high-speed performance compared to the current systems and the studies carried out in this regard.

This model did not focus on the speed of execution and the representation of dynamic results alone, but rather the ability to process complex query sentences. OLAP hypercube technology provided the ability to perform a complex query on big data databases; this model helps organizations with large data, which allows its users to immediately query the data by providing the query in a highly accurate and fast manner with the fewest resources and equipment needed by the infrastructure.

Table 4. NLQ to SQL execution time.

USER QUERY ENTRIES	GENERATED SQL TEXT	NO. ROWS PROCESSED	MODULE	SQL EXEC TIME	TOTAL EXEC TIME
can you tell me prof names and title who publicationid more than 2	SELECT distinct name, jtitle FROM prof WHERE pubid > 2	12	python.exe	0.001208	0.085923
get all professors students and publication	SELECT distinct stuname, pubtitle FROM prof	69	python.exe	0.001831	0.093697
give me all professors students	SELECT distinct stuname FROM prof	16	python.exe	0.001287	0.084
get prof names and title who publicationid more than 2	SELECT distinct name, jtitle FROM prof WHERE pubid > 2	6	python.exe	0.000981	0.089831
could you please tell me the prof names and title	SELECT distinct name, jtitle FROM prof	12	python.exe	0.00142	0.101329
get prof names and title	SELECT distinct name, jtitle FROM prof	6	python.exe	0.000659	0.087733
tell me name about all professors	SELECT distinct name FROM prof	18	python.exe	0.00139	0.071274
tell me the name and publication for professors who id is equal to 1	SELECT distinct name, pubtitle FROM prof WHERE id = 1	8	python.exe	0.001116	0.091737
if you please tell me the name and the department and publication also the title for professors students who professor id is equal to one	SELECT distinct name, depid, pubtitle, jtitle, stuname FROM prof WHERE id = 1	8	python.exe	0.001092	0.089989
tell me the name and publication for professors students who professor id is equal to 1	SELECT distinct name, pubtitle, stuname FROM prof WHERE id = 1	8	python.exe	0.001024	0.096109
give me all professors students who professor id is more than 1 and less than nine	SELECT distinct stuname FROM prof WHERE id > 1 AND id < 9	56	python.exe	0.002042	0.096062
tell me the name of professors who id is equal to one	SELECT distinct name FROM prof WHERE id = 1	1	python.exe	0.00072	0.071885
Give all professors publication	SELECT distinct pubtitle FROM prof	18	python.exe	0.001126	0.092839
give me all professors informations	SELECT distinct * FROM prof	260	python.exe	0.011008	0.141568

7. Conclusions and Future Scope

Natural language interfaces are the most consistent and flexible interfaces for the user. The use of plain English and NLP will help users retrieve and manage data from databases. The user does not need to learn SQL or any other complex query language; user interfaces can be integrated with relational databases built through OLAP, which makes it easier for the user to deal with huge databases using this technology; in addition, the integrated query system provided a capability of covering all inquiries from Big Relational Databases.

The OLAP hypercube (4-D Cube) model presented in this study shows the ability to execute complex query clauses in one single command and solve the problem of displaying complex and multiple results by developing a dynamic display interface in which data display is limited to query results alone.

As a plan for future work, it is recommended to use infinite-dimensional n-D cubes instead of 4-D cubes to enable the ingestion of big data in a single object and to facilitate the execution of query statements that may be too complex in query interfaces running in a data warehouse. Abbreviations section summarizes all abbreviations used in the paper.

Author Contributions: F.H.H. designed and built the model, processed the data, and wrote the first draft of the manuscript; M.O. helped in the development of the model and in the processing of the data. A.Y.O. helped in the interpretation of the results and revised and edited the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research is funded by the Arab American University in Palestine.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data used for this article were collected by our research team and are available in [45].

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

SQL	Structured Query Language
RDBMS	Relational Database Management Systems
NLQ	Natural Language Query
OLAP	Online Analytical Processing
NLP	Natural Language Processing
AI	Artificial Intelligence
NLQs	Natural Language Queries
NL	Natural language
NLI	Natural Language Interface
RDBs	Relational Database
MT	Machine translation
ML	Machine learning
NLTK	Natural Language Tool Kit
AWM	Oracle Analytic Workspace Manager

References

1. Joshi, S.; Arindom, R.; Dikshit, T.; Anish, B.; Deep, A.G.; Pallav, P. Conceptual paper on factors affecting the attitude of senior citizens towards purchase of smartphones. *Indian J. Sci. Technol.* **2015**, *8*, 83–89. [CrossRef]
2. Giordani, A.; Moschitti, A. Generating SQL queries using natural language syntactic dependencies and metadata. In *International Conference on Application of Natural Language to Information Systems*; Springer: Berlin/Heidelberg, Germany, 2012; Volume 7337, pp. 164–170. [CrossRef]
3. Approach, P. Conversion of Natural Language Statement into SQL Query using. In Proceedings of the 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 29–31 March 2018; pp. 488–491.

4. Sanyal, H.; Shukla, S.; Agrawal, R. Natural Language Processing Technique for Generation of SQL Queries Dynamically. In Proceedings of the 2021 6th International Conference for Convergence in Technology (I2CT), Maharashtra, India, 2–4 April 2021; pp. 1–6. [CrossRef]
5. Djahantighi, F.S.; Norouzifard, M.; Davarpanah, S.H.; Shenassa, M.H. Using natural language processing in order to create SQL queries. In Proceedings of the 2008 International Conference on Computer and Communication Engineering, Kuala Lumpur, Malaysia, 13–15 May 2008; pp. 600–604. [CrossRef]
6. Bhadgale, A.M.; Gavas, S.R.; Goyal, P.R. Natural Language To Sql Conversion System. *Int. J. Comput. Sci. Eng. Inf. Technol. Res.* **2013**, *3*, 161–166.
7. Kaur, S.; Bali, R.S. SQL generation and execution from natural language processing. *Int. J. Comput. Bus. Res.* **2012**. Available online: http://researchmanuscripts.com/isociety2012/54.pdf (accessed on 19 November 2021).
8. Popescu, A.-M.; Etzioni, O.; Kautz, H. Towards a theory of natural language interfaces to databases. In Proceedings of the 8th International Conference on Intelligent User Interfaces, Miami, FL, USA, 12–15 January 2003; p. 327. [CrossRef]
9. Painuly, S.; Sharma, S.; Matta, P. Big Data Driven E-Commerce Application Management System. In Proceedings of the 2021 6th International Conference on Communication and Electronics Systems (ICCES), Coimbatre, India, 8–10 July 2021. [CrossRef]
10. Cappa, F.; Oriani, R.; Peruffo, E.; McCarthy, I. Big Data for Creating and Capturing Value in the Digitalized Environment: Unpacking the Effects of Volume, Variety, and Veracity on Firm Performance. *J. Prod. Innov. Manag.* **2021**, *38*, 49–67. [CrossRef]
11. Abourezq, M.; Idrissi, A. Database-as-a-Service for Big Data: An Overview. *Int. J. Adv. Comput. Sci. Appl.* **2016**, *7*, 157–177. [CrossRef]
12. Uma, M.; Sneha, V.; Sneha, G.; Bhuvana, J.; Bharathi, B. Formation of SQL from natural language query using NLP. In Proceedings of the 2019 International Conference on Computational Intelligence in Data Science (ICCIDS), Chennai, India, 21–23 February 2019; pp. 1–5. [CrossRef]
13. Singh, G.; Solanki, A. An algorithm to transform natural language into SQL queries for relational databases. *Selforganizology* **2016**, *3*, 100–116.
14. Al Taleb, T.M.J.; Hasan, S.; Mahd, Y.Y. On-line analytical processing (OLAP) operation for outpatient healthcare. *Iraqi J. Sci.* **2021**, *2021*, 225–231. [CrossRef]
15. Naeem, M.A.; Bajwa, I.S. Generating OLAP queries from natural language specification. In Proceedings of the International Conference on Advances in Computing, Communications and Informatics, Chennai, India, 3–5 August 2012; pp. 768–773. [CrossRef]
16. Mincheva, Z.; Vasilev, N.; Antonov, A.; Nikolov, V. *NLP Using Database Context*; Eurorisk Systems Ltd.: Varna, Bulgaria, 2020.
17. Date, C.J.; Edgar, F. Codd: August 23rd, 1923–April 18th, 2003, A tribute and personal memoir. *SIGMOD Rec.* **2003**, *32*, 4–13. [CrossRef]
18. Colliat, G. OLAP, Relational, and Multidimensional Database Systems Characteristics of On-Line Analytical Processing. *Scenario* **1996**, *25*, 64–69.
19. Zaiane, O.R.; Xin, M.; Han, J. Discovering web access patterns and trends by applying OLAP and data mining technology on web logs. In Proceedings of the IEEE International Forum on Research and Technology Advances in Digital Libraries-ADL'98, Santa Barbara, CA, USA, 22–24 April 1998; pp. 19–29. [CrossRef]
20. Joseph, S.R.; Hloman, H.; Letsholo, K.; Sedimo, K. Natural Language Processing: A Review. *Int. J. Res. Eng. Appl. Sci.* **2016**, *6*, 1–8.
21. Owda, M.; Bandar, Z.; Crockett, K. Information extraction for SQL query generation in the Conversation-Based Interfaces to Relational Databases (C-BIRD). In *KES International Symposium on Agent and Multi-Agent Systems: Technologies and Applications*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6682, pp. 44–53. [CrossRef]
22. Owda, M.; Owda, A.Y.; Gasir, F. A Comprehensive Methodology for Evaluating Conversation-Based Interfaces to Relational Databases (C-BIRDs). *Adv. Intell. Syst. Comput.* **2021**, *1251*, 196–208. [CrossRef]
23. McKeown, K.R. Discourse strategies for generating natural-language text. *Artif. Intell.* **1985**, *27*, 1–41. [CrossRef]
24. Rubinoff, R. Adapting MUMBLE: Experience with Natural Language Generation TEXT's Message Vocabulary. In Proceedings of the The Fifth National Conference on Artificial Intelligence (AAAI-86), Philadelphia, PA, USA, 11–15 August 1986; pp. 200–211.
25. Grosz, B.J. Natural language processing. *Artif. Intell.* **1982**, *19*, 131–136. [CrossRef]
26. Owda, M.; Bandar, Z.; Crockett, K. Conversation-Based Natural Language Interface to Relational Databases. In Proceedings of the 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology, Silicon Valley, CA, USA, 5–12 November 2007; pp. 363–367. [CrossRef]
27. Wang, P.; Shi, T.; Reddy, C.K. Text-to-SQL Generation for Question Answering on Electronic Medical Records. In Proceedings of the Web Conference 2020, Taipei, Taiwan, 20–24 April 2020; pp. 350–361. [CrossRef]
28. Nihalani, N.; Silakari, S.; Motwani, M. Natural language Interface for Database—A Brief review. *Int. J. Comput. Sci. Issues* **2011**, *8*, 600–608.
29. Baik, C.; Jagadish, H.V.; Li, Y. Bridging the semantic gap with SQL query logs in natural language interfaces to databases. In Proceedings of the 2019 IEEE 35th International Conference on Data Engineering (ICDE), Macao, China, 8–11 April 2019; pp. 374–385. [CrossRef]
30. Nagare, P.; Indhe, S.; Sabale, D.; Thorat, G.P.D.Y.; Chaturvedi, P.G.K. Automatic SQL Query Formation from Natural Language Query. *Int. Res. J. Eng. Technol.* **2017**, *4*, 1589–1591. Available online: https://www.irjet.net/archives/V4/i5/IRJET-V4I5310.pdf (accessed on 19 November 2021).

31. Stoica, A.; Pu, K.Q.; Davoudi, H. NLP Relational Queries and Its Application. In Proceedings of the 2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI), Las Vegas, NV, USA, 11–13 August 2020; pp. 395–398. [CrossRef]
32. Osorio, J.; Beltran, A. Enhancing the Detection of Criminal Organizations in Mexico using ML and NLP. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020. [CrossRef]
33. Nayyer, R.; Mishra, D. Sentiment Analysis using NLP: Survey Paper. *Int. J. Res. Sci. Commun.* **2021**, *1*, 1–4.
34. Tardío, R.; Maté, A.; Trujillo, J. A new big data benchmark for olap cube design using data pre-aggregation techniques. *Appl. Sci.* **2020**, *10*, 8674. [CrossRef]
35. Wisnubhadra, I.; Baharin, S.K.; Emran, N.A.; Setyohadi, D.B. Qb4mobolap: A vocabulary extension for mobility olap on the semantic web. *Algorithms* **2021**, *14*, 265. [CrossRef]
36. WGraterol; Diaz-Amado, J.; Cardinale, Y.; Dongo, I.; Lopes-Silva, E.; Santos-Libarino, C. Emotion detection for social robots based on nlp transformers and an emotion ontology. *Sensors* **2021**, *21*, 1322. [CrossRef]
37. Tovkach, S.S. Hypercube Architecture of Information for Aviation Engine Control System. In Proceedings of the 2020 IEEE 6th International Conference on Methods and Systems of Navigation and Motion Control (MSNMC), Kyiv, Ukraine, 20–23 October 2020; pp. 69–72. [CrossRef]
38. Zykin, S.V. Formation of hypercube representation of relational database. *Program. Comput. Softw.* **2006**, *32*, 348–354. [CrossRef]
39. Toce, A.; Mowshowitz, A.; Stone, P.; Bent, G.; Park, H. HyperD: A hypercube topology for dynamic distributed federated databases. In Proceedings of the 5th Annual Conference International Technology Alliance, Wrexham, North Wales, UK, 8–11 September 2011.
40. Javanmard, M.M.; Ahmad, Z.; Kong, M.; Pouchet, L.N.; Chowdhury, R.; Harrison, R. Deriving parametric multi-way recursive divide-and-conquer dynamic programming algorithms using polyhedral compilers. In Proceedings of the 18th ACM/IEEE International Symposium on Code Generation and Optimization, San Diego, CA, USA, 22–26 February 2020; pp. 317–329. [CrossRef]
41. Mowshowitz, A.; Kawaguchi, A.; Toce, A.; Nagel, A.; Bent, G.; Stone, P.; Dantressangle, P. Query Optimization in a Distributed Hypercube Database. In Proceedings of the Fourth Annual Conference of ITA, Guangzhou, China, 26–28 May 2017.
42. Kasprzyk, J.P.; Devillet, G. A data cube metamodel for geographic analysis involving heterogeneous dimensions. *ISPRS Int. J. Geo-Inf.* **2021**, *10*, 87. [CrossRef]
43. Al-Aiad, A.; El-Shqeirat, T. Text mining in radiology reports (Methodologies and algorithms), and how it affects on workflow and supports decision making in clinical practice (Systematic review). In Proceedings of the 2020 11th International Conference on Information and Communication Systems (ICICS), Irbid, Jordan, 7–9 April 2020; pp. 283–287. [CrossRef]
44. Uskenbayeva, R.K.; Kozhamzharova, D.K.; Kurmangaliyeva, B.K.; Bektemyssova, G.B.; Mukazhanov, N.K. Multidimensional indexing structure development for the optimal formation of aggregated indicators in OLAP hypercube. In Proceedings of the 2014 14th International Conference on Control, Automation and Systems (ICCAS 2014), Seoul, Korea, 22–25 October 2014; pp. 1466–1470.
45. Owda, M.; Crockett, K.; Alghamdi, A. Natural Language Interface to Relational Database (NLI-RDB) Through Object Relational Mapping (ORM). In *Advances in Computational Intelligence Systems*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 449–464. [CrossRef]
46. Banerjee, S.; Bhaskar, S.; Sarkar, A.; Debnath, N.C. A formal OLAP algebra for NoSQL based data warehouses. *Ann. Emerg. Technol. Comput.* **2021**, *5*, 154–161. [CrossRef]
47. Kiruthika, S.; Umamaheswari, E. Obtaining relevant Data cubes in OLAP for Efficient Online Decision Support Systems. *Ann. Rom. Soc. Cell Biol.* **2021**, *25*, 5862–5865.
48. Felber, T. Machine Learning Models for COVID-19 Fake News Detection Shared Task. *Nature* **2021**, *388*, 539–547.
49. Pazos, R.R.A.; González, B.J.J.; Aguirre, L.M.A.; Martínez, F.J.A.; Fraire, H.H.J. Natural language interfaces to databases: An analysis of the state of the art. *Stud. Comput. Intell.* **2013**, *451*, 463–480. [CrossRef]
50. Giordani, A.; Moschitti, A. Semantic mapping between natural language questions and SQL queries via syntactic pairing. In Proceedings of the International Conference on Applications of Natural Language to Information Systems, Saarbrücken, Germany, 24–26 June 2009; Volume 5723, pp. 207–221. [CrossRef]
51. Nguyen, A.T.; Dao, M.H.; Nguyen, D.Q. A Pilot Study of Text-to-SQL Semantic Parsing for Vietnamese. *arXiv* **2020**, arXiv:2010.01891. [CrossRef]
52. Qi, P.; Zhang, Y.; Zhang, Y.; Bolton, J.; Manning, C.D. Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. *arXiv* **2020**, arXiv:2003.07082. [CrossRef]
53. Mihajlovi, S.; Kupusinac, A.; Ivetić, D.; Berković, I. The Use of Python in the field of Artifical Intelligence. In Proceedings of the International Conference on Information Technology and Development of Education (ITRO 2020), Zrenjanin, Serbia, October 2020; pp. 1–5.
54. Ott, N. Aspects of the automatic generation of SQL statements in a natural language query interface. *Inf. Syst.* **1992**, *17*, 147–159. [CrossRef]
55. Androutsopoulos, I.; Ritchie, G.; Thanisch, P. *Masque/sql An E cient and Portable Natural Language Query Interface for Relational Databases*; Database Technical Paper; Department of AI, University of Edinburgh: Edinburgh, UK, 1994; pp. 1–7.