A Project Report

On

Reverse Engineering Database to ER Model

By

Meet Chirag Patel

2021A7PS2692H

&

Ishan Harsh

2021A7PS2854H

Under the supervision of

PROF. R GURURAJ

SUBMITTED IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS OF

CS F376: LABORATORY PROJECT

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI (RAJASTHAN)

HYDERABAD CAMPUS

(MAY 2024)

ACKNOWLEDGMENTS

I would like to express my heartfelt gratitude to everyone who helped make this project report on Reverse Engineering Database to ER Model a reality, under the supervision of Prof. R Gururaj at Birla Institute of Technology and Science (BITS) Pilani, Hyderabad Campus.

First and foremost, I want to express my heartfelt gratitude to Prof. R Gururaj for his unfailing advice, unshakable support, and essential mentoring throughout this endeavor. His depth of information, intelligent suggestions, and encouragement not only influenced the course of my report, but also expanded my understanding of DBMS systems.

I am grateful to my peers and colleagues for their collaborative attitude, constructive input, and steadfast support, all of which contributed greatly to the substance and quality of this project report.

Finally, I would like to offer my heartfelt gratitude to my family and friends for their continuous support, understanding, and encouragement during this effort. Their unfailing conviction in my talents, as well as their steadfast support, served as the foundation for my endurance and eventual success in accomplishing this endeavor. Their combined contributions have certainly helped to bring this project to a successful conclusion.

Birla Institute of Technology and Science-Pilani,

Hyderabad Campus

Certificate

This is to certify that the project report entitled "**Reverse Engineering Database to ER Model**" submitted by Mr. Meet Chirag Patel (ID No. 2021A7PS2692H) and Mr. Ishan Harsh (ID No. 2021A7PS2854H) in partial fulfillment of the requirements of the course CS F376, Design Project Course, embodies the work done by them under my supervision and guidance.

Date: 7 May 2024

(**Prof. R Gururaj)**

BITS- Pilani, Hyderabad Campus

ABSTRACT

Reverse Engineering of Relational Database Management Systems (RDBMS) to Entity-Relationship (ER) models is a crucial process in database management, aiming to comprehend and represent existing database structures in a more intuitive and efficient manner. The need for reverse engineering arises from the complexity of modern databases, where understanding the relationships between different entities is essential for effective management and optimization. By transforming RDBMS into ER models, we simplify the database structure, making it more accessible for analysis, design, and maintenance.

To equip ourselves with the latest knowledge and techniques in this field, we extensively reviewed research papers, such as [1],[2],[3],[4],[5] and more, gaining insights into various methodologies and advancements. This literature review provided us with a comprehensive understanding of the current state-of-the-art approaches and challenges in reverse engineering RDBMS to ER models.

Our implementation approach involved writing a python script using psycopg2 library to connect to PostgreSQL database and then extract information from the PostgreSQL database by listing all its user defined tables. We then systematically iterated through each table, utilizing the '\d' command or rather the ‘information_schema.’ command to gather metadata about their attributes and constraints like primary key, unique, not null, and foreign key. The gathered information was then stored in a dictionary data structure and individual files pertaining to each table for further processing and analysis. Our work contributes to the ongoing research and development efforts in database management and enhances our understanding of complex database systems.

TABLE OF CONTENT

INTRODUCTION

Reverse engineering a database into an Entity-Relationship (ER) model is a useful technique for understanding the database by extracting and modelling its conceptual core. As databases grow in size and complexity, they may get congested with duplicate tables, intricate links, and inefficient structures. Understanding the underlying ER model provides insight into the system's design, allowing stakeholders to identify areas for enhancement, normalization, and optimization. Furthermore, reverse engineering is a crucial approach for dealing with undocumented or outmoded databases since it enables stakeholders to provide thorough and up-to-date documentation for future use. Furthermore, in today's fast changing technical environment, businesses frequently do system migrations or integrations, which require a thorough understanding of existing data structures. Reverse engineering to an ER model allows for smoother data transitions and assures consistency in entity connections. Furthermore, when legacy systems change, the ER model obtained via reverse engineering provides as a platform for revamping the database structure to match modern technical and commercial needs while maintaining data integrity.

We thoroughly reviewed research publications on the reverse engineering of RDBMS to ER models. Our technique included a systematic review process in which we examined numerous methodologies, algorithms, and case studies described in [1], [2], [3], [4], [5], and other literatures. We developed a holistic strategy by integrating ideas from numerous sources, incorporating best practices and advances from current research. This synthesis not only provided us with a strong theoretical framework, but it also guided our execution strategy. Furthermore, by critically examining the strengths and limits of various methodologies, we were able to make educated judgments throughout the implementation process, improving our methodology's efficiency and accuracy. Thus, our implementation is not just a product of technical prowess but also a testament to our rigorous research-driven approach, which underpins its robustness and effectiveness.

RELATED WORK

[1] underlines the need of reverse engineering in preserving older systems with minimal documentation. The suggested technique seeks to improve software maintenance methods by extracting hidden data from database schemas. It transforms SELECT queries and SQL DDL database schema into XMI-formatted UML models, bridging the gap between application and data modelling. Incorporating object-oriented technology into relational databases simplifies system representation in a single language, making collaboration easier across development teams. Various DBRE approaches examined in [2] are effective at reverse engineering simple aspects like entities and relationships, but they struggle with more complex constructions like generalization, unions, and weak entities. The study proposes for the creation of a technique that can handle these intricacies with minimum human involvement, increasing productivity and lowering maintenance costs. Formal approaches are advised for developing unambiguous input-output models to improve the DBRE process.

In [4], SQL2XMI is presented as a program that automates the translation of SQL schemas into UML-ER models in XMI 2.1 format. This allows for the modification of both application and data models using UML-based tools, which improves understanding and development of complex software systems. Unlike proprietary formats, SQL2XMI takes an open and flexible portability strategy that aligns with the OMG XMI 2.1 UML standard. [5] provides a method for generating a conceptual schema from a relational database, which is especially useful for outdated systems that lack detailed documentation. It detects properties describing table associations and possible keys by examining data manipulation instructions in application code. This systematic technique improves understanding and evolution of relational databases in a distributed computing environment, highlighting the relevance of data manipulation statements in queries and views when reverse engineering older systems with minimal DBMS information.

IMPLEMENTATION

The Python script we have implemented here is particularly tailored for PostgreSQL systems. Its significance lies in its ability to bridge the gap between raw database metadata and comprehensible documentation, thereby aiding in the understanding and management of database schemas.

The script initiates its functionality by importing the psycopg2 library, a popular PostgreSQL adapter for Python. This module enables seamless communication with the PostgreSQL database, granting access to its metadata and facilitating the extraction of essential information about database tables. We start by connecting the database which we must reverse engineer to make ER model by giving the database name, username, password, host, and port number inside the ‘connect()’ function. Then using ‘cursor()’ function we make a cursor which helps in executing SQL commands. Then using ‘execute()’ we execute the equivalent of ‘\dt’ command which is ‘information_schema.tables where table_schema=’public’’. We now have all user defined tables or in other words all the entity types required for ER model in the cursor which we store in a list called tables using ‘fetchall()’ function. We then iterate through all the tables to get all the information regarding their attributes and constraints. We get all the columns of each table by executing the command ‘information_schema.columns’ and store them. Similarly, we extract the constraints using ‘information_schema.table_constraints’. We then extract the foreign key and the table as well as entity it points to by executing the commands ‘information_schema.table_constraints where constraint_type = ‘FOREIGN KEY’’, ‘.key_column_usage’, and ‘constraint_column_usage’.

We have now the entity types, their attributes, their primary keys, and we can form relationships among them based on foreign keys. The relationships can be identified using the foreign key and Primary Key relationship between the entities. The relationship with weak entity can be identified by seeing the entities which have pair of Primary keys, and checking if one of the keys in the primary key pair is not a foreign key to any of the identified primary keys. That key will then be the weak key of the weak entity. Multivalued attributes can also be identified by an entity which has two attributes with one of the attributes being the foreign key to a Primary key in an entity. The second attribute will become an attribute of the entity to which the second key was pointing to. All the information gathered till now is stored in dictionary and each table has its own files made with all its information.

The script includes error handling using try-except blocks to catch any exceptions that may occur during database operations and print error messages. The script is intended to be run as a standalone program and can be executed from the command line or integrated into other Python scripts or applications. Overall, this script provides a convenient way to document the structure of tables in a PostgreSQL database, which is useful for us to make the ER model.

CONCLUSION

In conclusion, the process of reverse engineering database systems into Entity-Relationship (ER) models holds immense significance in modern database management practices. As outlined in the introduction, understanding the underlying ER model facilitates insights into system design, enabling stakeholders to identify areas for improvement, normalization, and optimization.

The related work section underscores the breadth and depth of research in the field of database reverse engineering, showcasing various methodologies and tools aimed at extracting and modelling database structures. Our implementation, as described in detail, leverages Python and psycopg2 to extract detailed information about tables in a PostgreSQL database, storing the data in a structured format for further analysis and documentation. The script's systematic approach, combined with error handling mechanisms, ensures reliability and robustness in extracting table information. The next step is to implement the logic to identify weak entity types and store this relation as well as implement the logic to get the cardinality of relationships (‘1:m’ or ‘m:n’) and then focus on implementation of getting composite attributes and also implementing the logic to identify those attributes that belong to a relationship through analysing the tables.

In essence, the culmination of our research-driven approach and practical implementation underscores the importance of database reverse engineering in modern data management practices. By bridging the gap between raw database metadata and comprehensible documentation, our script provides a valuable tool for database administrators, developers, and analysts to gain deeper insights into database structures and facilitate informed decision-making processes and we continue to build upon what we have built and expand it so as to extract as many details as we can from the database in order to form the best ER model we can.

REFERENCES

[1]. N. Iftekhar, M. R. Warsi, S. Zafar, S. Khan, and S. S. Biswas, "Reverse Engineering of Relational Database Schema to UML Model," *2019 International Conference on Electrical, Electronics and Computer Engineering (UPCON)*, Aligarh, India, 2019, pp. 1-6, doi: 10.1109/UPCON47278.2019.8980043.

[2]. Mian, N. A., Khan, S. A., & Zafar, N. A. (2013). Database Reverse Engineering Methods: What is Missing? *Research Journal of Recent Sciences ISSN*, *2277*, 2502.

[3]. Chiang, R. H., Barron, T. M., & Storey, V. C. (1994). Reverse engineering of relational databases: Extraction of an EER model from a relational database. Data & knowledge engineering, 12(2), 107-142.

[4]. M. H. Alalfi, J. R. Cordy and T. R. Dean, "SQL2XMI: Reverse Engineering of UML-ER Diagrams from Relational Database Schemas," 2008 15th Working Conference on Reverse Engineering, Antwerp, Belgium, 2008, pp. 187-191, doi: 10.1109/WCRE.2008.30.

[5]. Andersson, M. (1994). Extracting an entity relationship schema from a relational database through reverse engineering. In Lecture Notes in Computer Science (pp. 403–419). https://doi.org/10.1007/3-540-58786-1_93