DBMS Project

Group Members: Devansh Naresh Bansal (B20CS094) Diksha Jena (B20CS013) Abhishek Rajora (B20CS002)

Problem : Automation of ER diagram of an RDBMS system.

Specifications / Requirements - Algorithm and Demonstration of a mechanism to automate designing an ER diagram of an RDBM, in context to ‘entities’ ‘relations’ and ‘attributes

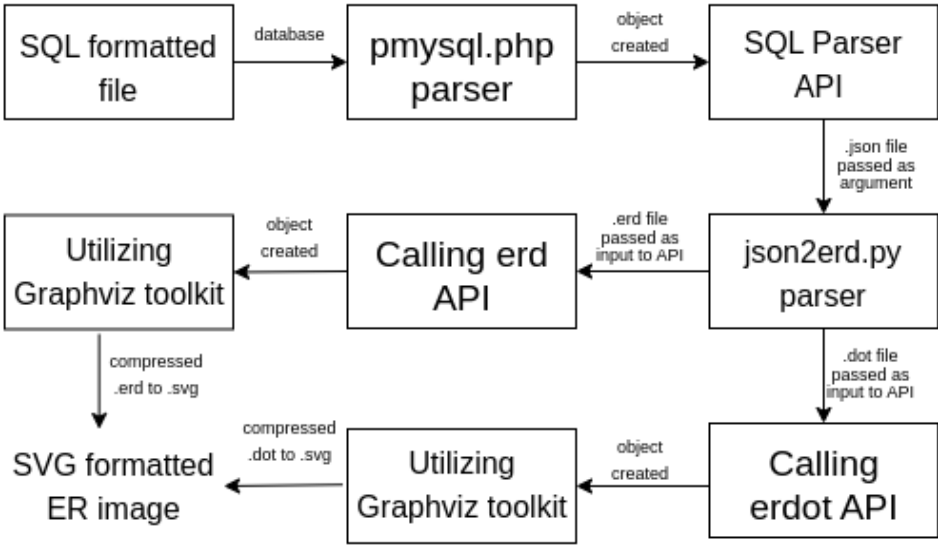
Based on complexity and mechanism involved there are two parts to the same problem.

Sub Problems :

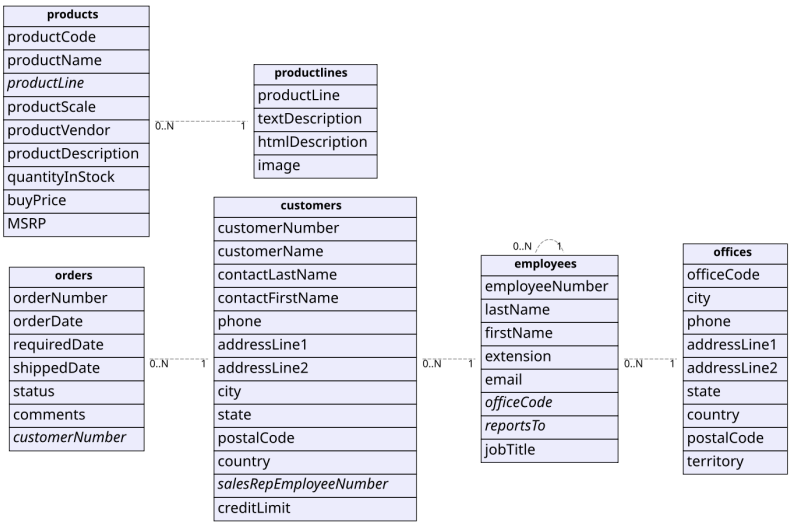
1. Table to Diagram : we have the code from which the database was created {basic information like entities, attributes, primary and foreign keys is already defined}
2. Theory/Raw Data’s diagram : to be used to create the database code. Only the scenario about which we need to create the database is known.

Methodology

1. **Table to Diagram**
    a. **Approach :** From the code file, we have commands to create/alter the database and fill in/update the entries, along with some other commands like select, view etc. Starting from here, the DDL and DML commands are segregated. DDL commands are used to figure out entities and attributes. The properties of the attributes are also defined in the same. They are used to extract relations and their cardinality is determined by the DML commands. For the diagram part once all the information is known, we use API’s to automate the process. The sample size of the input needs to be handleable else generation would be difficult.
    b. **Algorithm :**
        i. Read the sql file and parse it (.php script written), find out all the tokens present among the required commands (DDL).
        ii. Use sql parser API to extract entities and their features from the tokens determined previously. A json file is generated.
        iii. The json file is passed as input into the python script (ehne’s erdot) to produce a .dot with all the inputs (entities, relations etc) to be fed in graphviz.
        iv. If the test file is of .er format instead of .json, we use andrew gallant’s haskell file to generate the dot file for ER diagram.
        v. Graphvis takes in the .dot file to produce the final diagram in various file graphic formats.



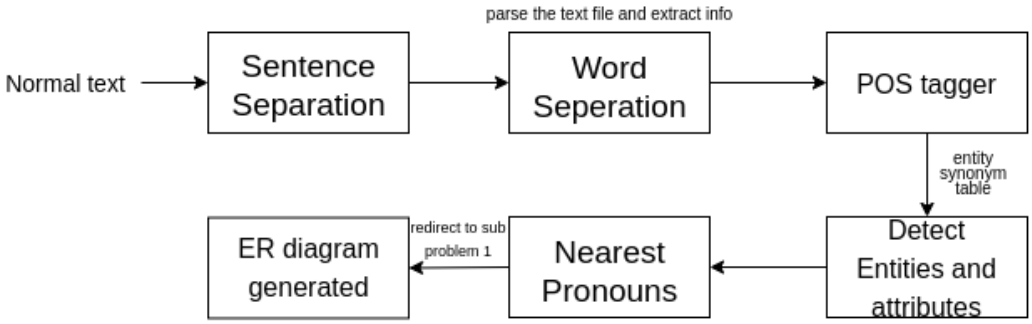
c. **Implementation :** pmysql.php and json2erd.py convert a .sql file into one of the two formats suitable for input to the above two programs, .json and .er.
Output



2. Raw Data's Diagram

a. **Approach :** Natural Language Processing (NLP) implementation involved
   i. First figure out the main words or requirements from the file, use nlp to find categories of words into grammars like nouns, verbs, adjectives, quantifiers, etc.
   ii. Then segregate the tokens for identifying the key-words which will help in creation of the database, like key-words for entities, for attributes etc.

iii. Word separator and NLTK POS tagger are used to get the table names, and attribute names, our task is to identify the relationship between entities.
iv. NLP is again used to figure out the possible relationships based on the database context. Another method to find basic relations might be to find attributes of different entities having the same name.
v. Following the language command syntax, code is generated from the variables included. Named entities from annotated domain specific databases generate the database.
vi. Support Vector Machine is used to chunk multiword entity names.
vii. The final dictionary state is maintained with all the information to generate the ER Diagram after all the sentences are processed.
viii. Use Graph-viz to render the final graph into a SVG file.



b. **Reference**
https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8614009

Lessons learnt

1. Amalgamation of different domains of worlds like ml, dbms, npl etc to create an effective and whole solution.
2. The sample of input sql formatted file parsed should be constrained to less than 10 attributes and less tables to create a uniform distribution of relations in the compressed svg file.
3. The sentences parsed into sub problem 2 should not create any complexity or ambiguity as the NLP tokenizer would not be able to relate the context defined for the database.
4. Continuous Work throughout the semester reduces workload.

Requirements of APIs

- erd API
- SQL parser
- erdot
- graphviz

Compiling and generating ER diagram from the sample database

- `First install the erd API into local device using the official erd API documentation`
- `cd into the directory where sample database and script is loaded`
- `php pmysql.php sample.sql > sample.json`
- `./json2erd.py -t busu < sample.json > sample.erd`
- `erd < sample.erd -f svg > sample.svg`

Contributions

- **Abhishek Rajora:** Worked on sub problem 1 and Implemented pmysql.php and json2erd.py parser utilizing three different APIs.
- **Diksha Jena:** Analyzed the problems and their solutions, utilized the graphviz toolkit to produce svg images.
- **Devansh Bansal:** Worked on the approach of extending sub problem 1 to Normal text using NLP.