REPORT

To do before executing the parser_loader.py file:

-> please make sure that source.txt is in the same directory

-> please change the following parameters according to your local system:

1.DB_HOST ----> change it to your local hostname or IP address

2.DB_NAME----> change it to your database name

3.DB_USER-----> change it to your system's username

4.DB_PASS------> fill it with your user's password

Some Data Cleansing:

The parser code in the file removes all the dots beside letters of abbreviated words in author names by adding spaces to disambiguate with the authors whose names are given without dots. It removes the repetitions in the list of co-authors of each paper. It ensures that no report cites itself by any chance. It fills the missing data with empty strings, and for authors, it shows "not found".

High-level Design:

It uses the codes mentioned in the problem sheet to find the attributes of the source files. We classified them into different strings and finally inserted them into their required table after sorting the data acquired from the source file.

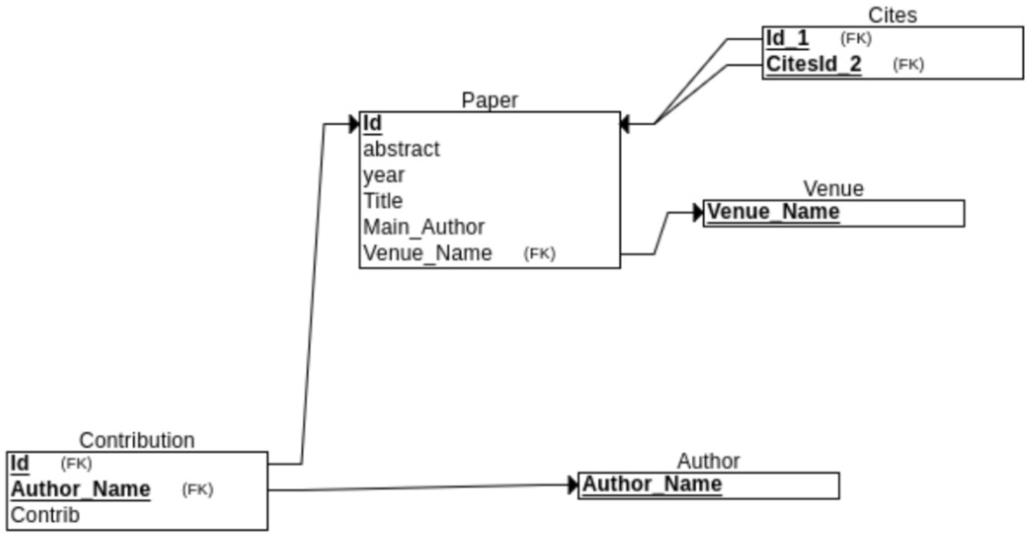
The same file parser_loader.py will do the parsing and loading of the source. It passes through the source file two times, first to fill tables with primary keys and without foreign keys, and second to fill tables with primary keys and foreign keys.

The SQL code to create tables is present in and executed by the “parser_loader.py file” itself.

Use this command to restore the database from the ‘pg_dump_file’ in terminal: “pg_restore -d ***dbname filename***”.

It can be restored even manually in ‘pgAdmin4’.

This is the schema we followed to create the relational database:


Cites
Id_1 (FK)
CitesId_2 (FK)
Paper
Id
abstract
year
Title
Main_Author
Venue_Name (FK)
Venue
Venue_Name
Contribution
Id (FK)
Author_Name (FK)
Contrib
Author
Author_Name