1. Software Installation Requirements

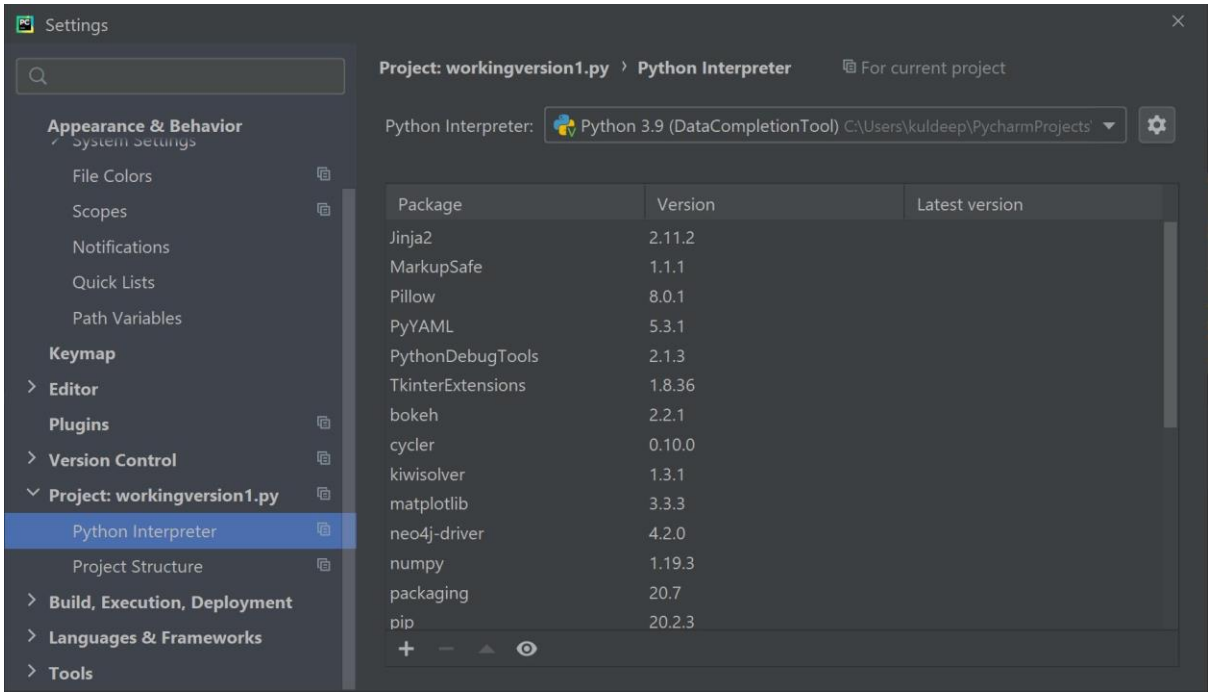
The main softwares required are shown below.

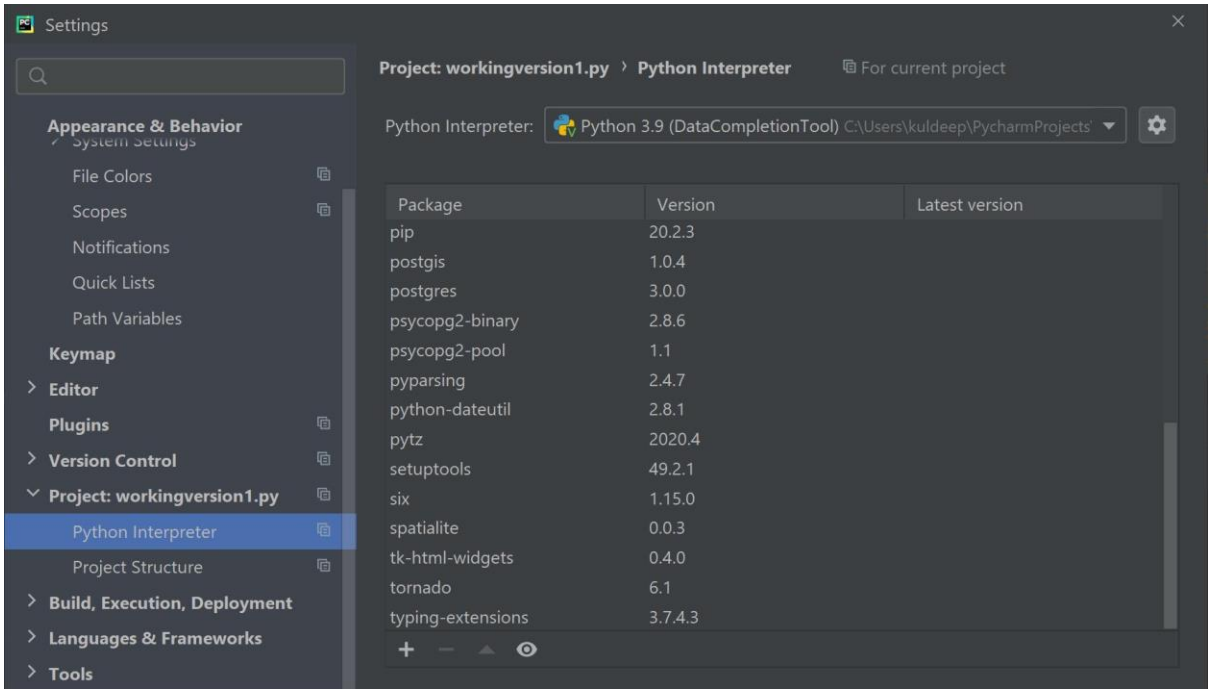
i. Java 11.0.9 with jre 8
ii. Python 3.9.1
iii. Postgis Bundle 3.0.3 for PostgreSQL 10
iv. Psycopg2 driver 2.8.6
v. Matplotlib 3.3.3

The below image provides more details on the installed softwares.

Name	Installed On	Size	Version
Google Chrome	03-Dec-20		87.0.4280.88
Java(TM) SE Development Kit 11.0.9 (64-bit)	10-Dec-20	279 MB	11.0.9.0
Microsoft OneDrive	27-Dec-20	148 MB	20.201.1005.0009
Microsoft Update Health Tools	20-Nov-20	1.05 MB	2.70.0.0
Microsoft Visual C++ 2013 Redistributable (x64) - 12.0.40664	22-Dec-20	20.5 MB	12.0.40664.0
Microsoft Visual C++ 2015-2019 Redistributable (x64) - 14.28.29334	22-Dec-20	22.1 MB	14.28.29334.0
Microsoft Visual C++ 2015-2019 Redistributable (x86) - 14.28.29334	22-Dec-20	19.8 MB	14.28.29334.0
Microsoft Visual C++ Build Tools	20-Dec-20	3.12 GB	14.0.25420.1
Microsoft Visual Studio Installer	22-Dec-20		2.8.3074.1022
PostGIS Bundle 3.0.3 for PostgreSQL x64 10 (remove only)	13-Dec-20		
PostgreSQL 10	22-Dec-20	450 MB	10
PyCharm Community Edition 2020.3	10-Dec-20		203.5981.165
Python 3.7.8 (32-bit)	22-Dec-20	161 MB	3.7.8150.0
Python 3.9.1 (64-bit)	11-Dec-20	103 MB	3.9.1150.0
Python Launcher	10-Dec-20	1.79 MB	3.9.7280.0
Update for Windows 10 for x64-based Systems (KB4023057)	09-Aug-19	1.42 MB	2.61.0.0
Visual Studio Community 2019	21-Dec-20		16.8.30804.86
Windows SDK AddOn	22-Dec-20	152 KB	10.1.0.0
Windows Software Development Kit - Windows 10.0.18362.1	22-Dec-20	2.35 GB	10.1.18362.1

The next two images shows the required python bundles, that are shown in pycharm interpreter details.



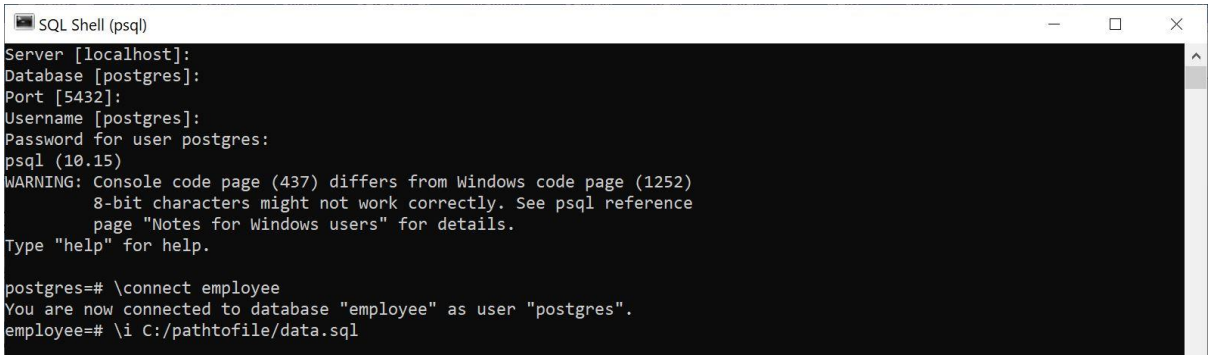


2. Preparing input

First step is to manually create the database. In this example, we have created an employee database.

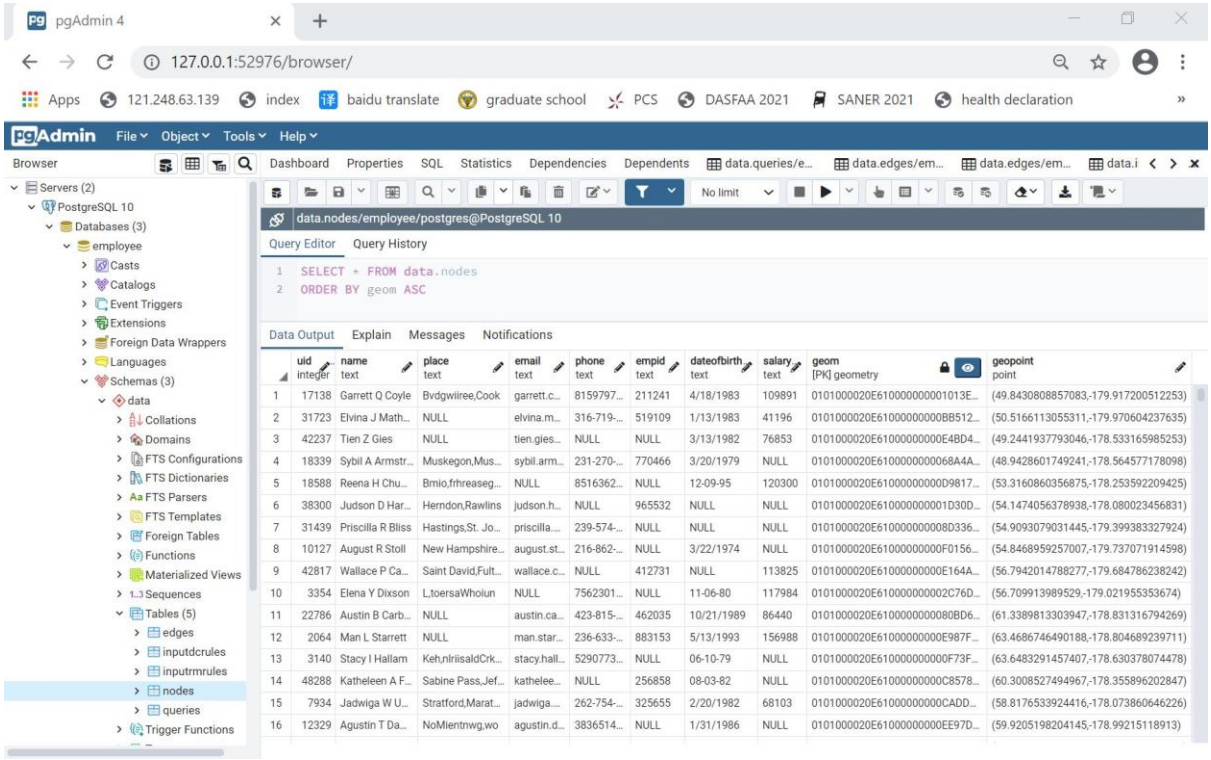
The data to be inserted is available in the form of .sql file. In particular is named as data.sql.

It must be imported into the employee database using SQL Shell (psql) command: \i C:/pathtofile/data.sql (beware of the / direction) - as shown in the image below.



a) Nodes

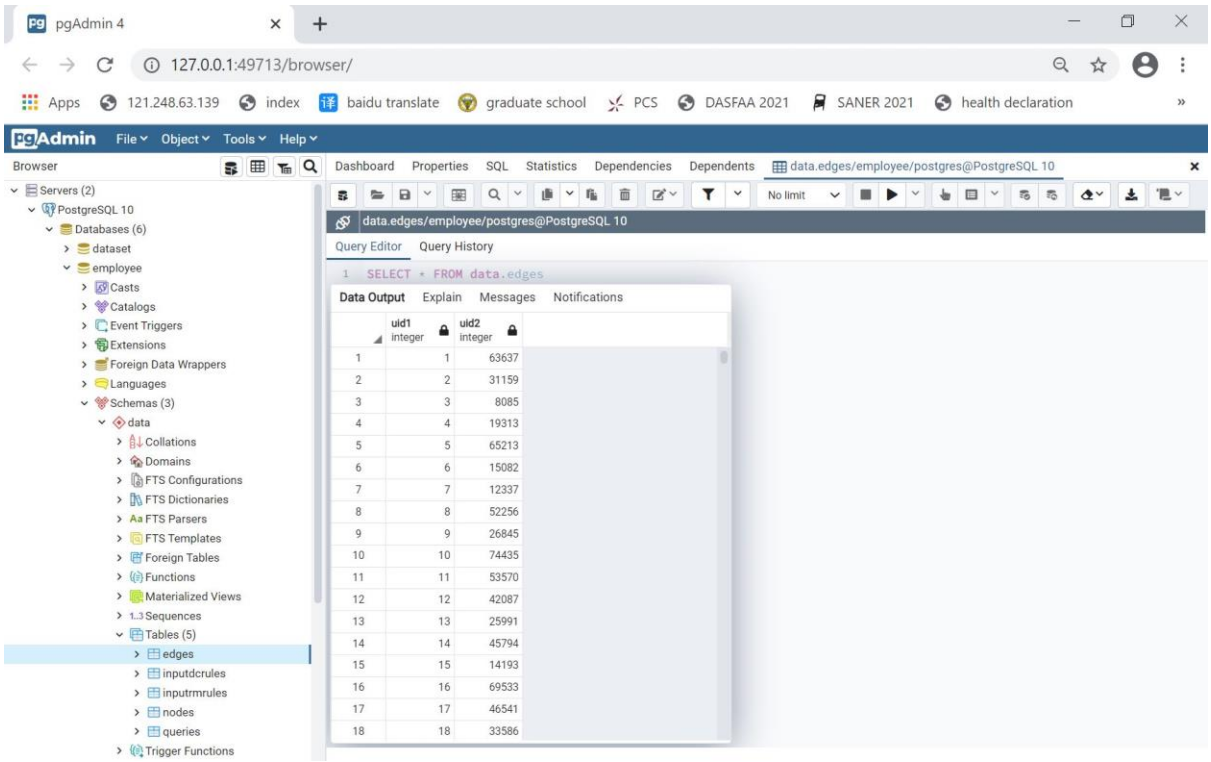
After loading .sql file into the database, the data is available for observation in the form of tables and functions. The nodes table provides details on the data loaded. In this particular example it provides information of employees. However, for the tool to work any attributes can be provided. The only requirement is the presence of a uid attribute unique for each row.



b) Edges

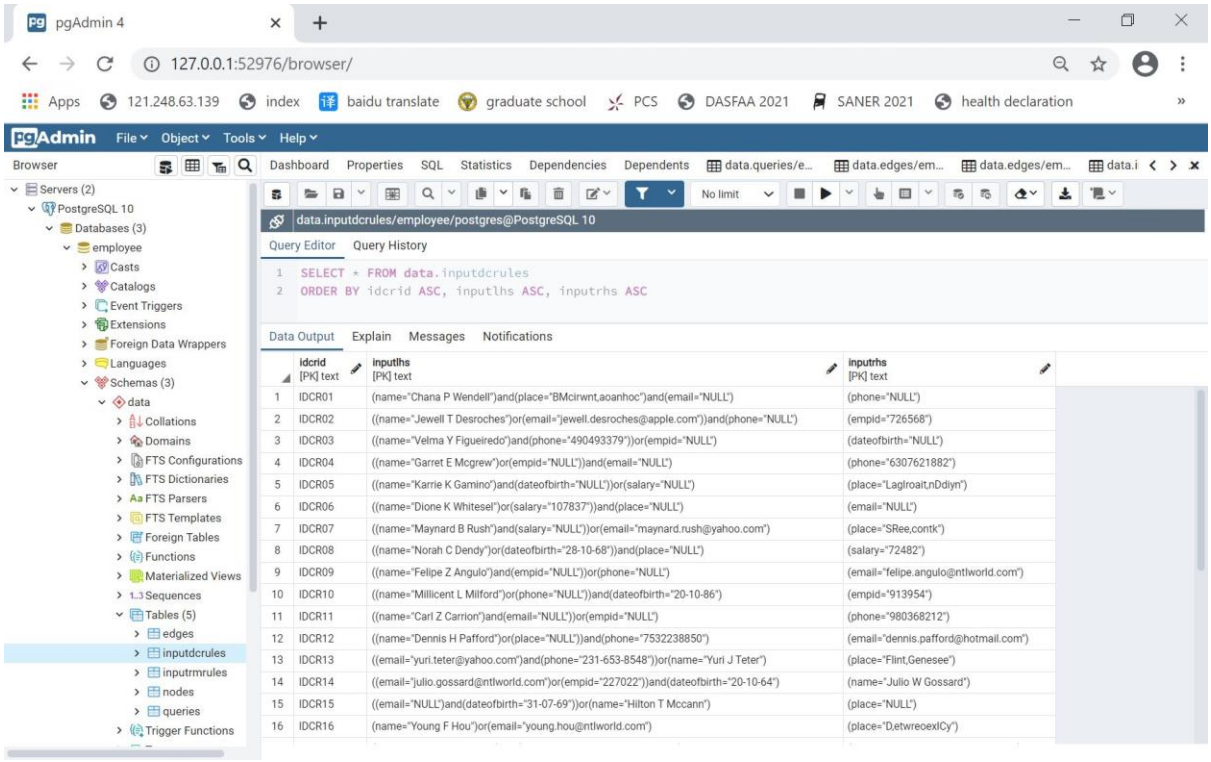
The edges table provides information on the relationships between rows in the nodes table. It mainly consists of 2 uid attributes (example: uid1, uid2) that identify the concerned rows of nodes table. More attributes can be added in this

table. However, for the tool to work atleast 2 uids must be present here as well as in nodes table**.**



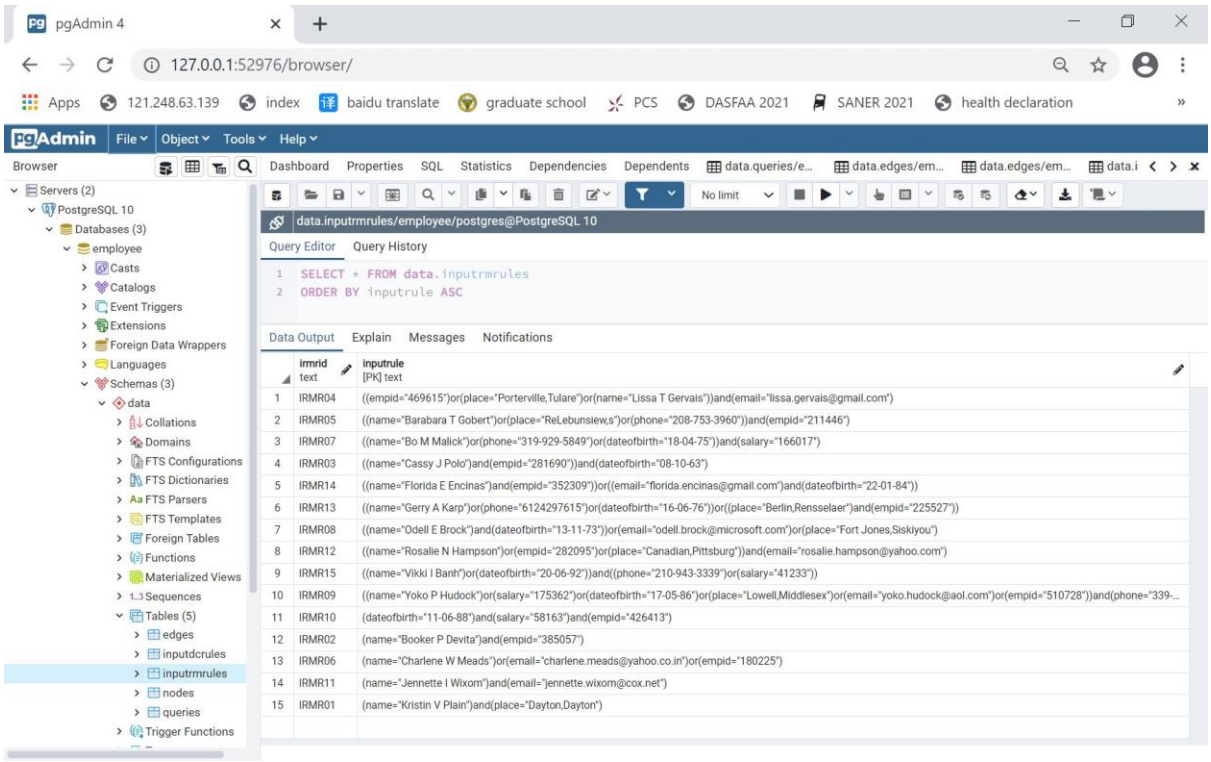
c) Data completion rules

The data completion rules table provides information on the left hand side and right hand side of the data completion rules as well as an identifier for it. The rule has a sequence of and’s/or’s with balanced paranthesis as well as attribute-value pairs.



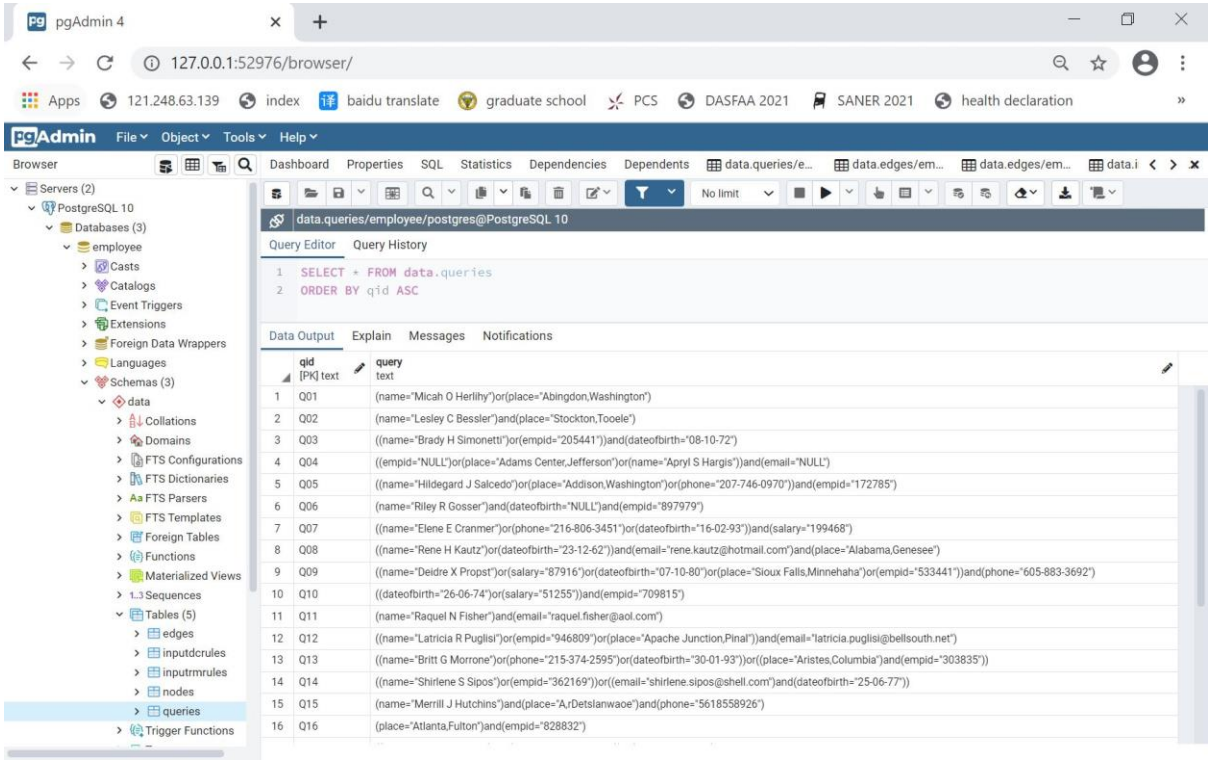
d) Record Matching Rules

The record matching rules table provides information on the identifier for the rule as well as the rule condition itself. The rule consists of balanced paranthesis as well as combination of and's/or's between attribute-value pairs of database.

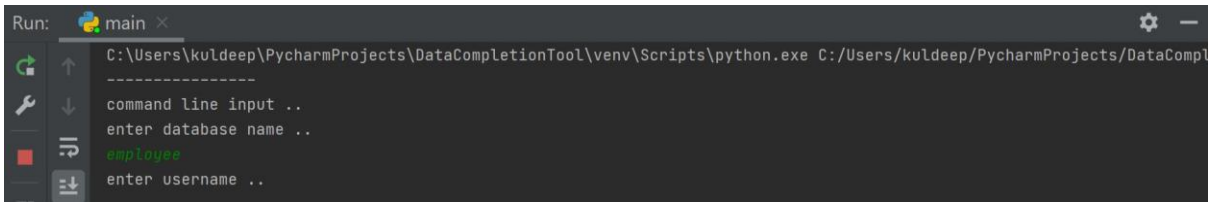


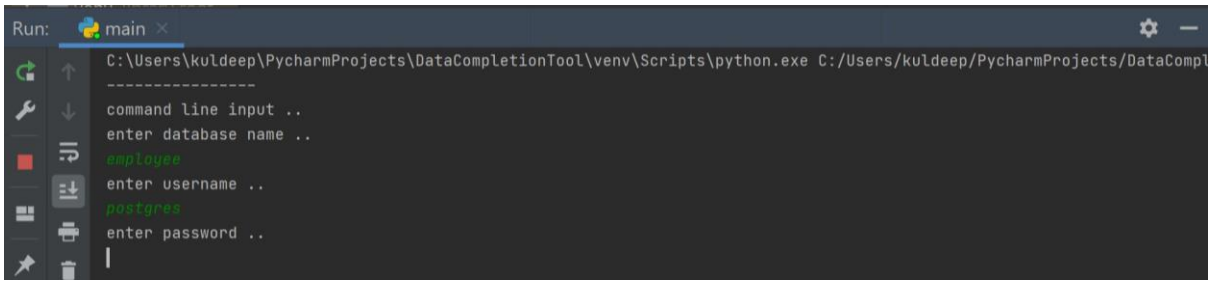
e) Queries

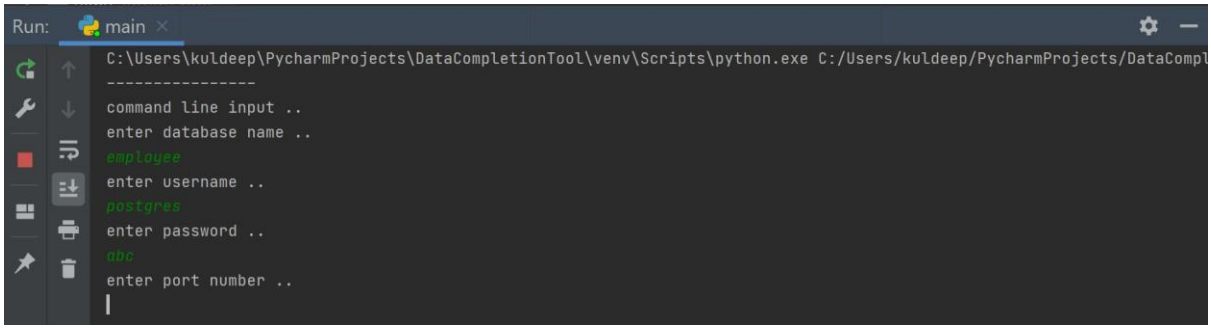
The queries table consists of where conditions of queries log collected in a table. It has 2 columns an identifier for query and the actual query condition itself.

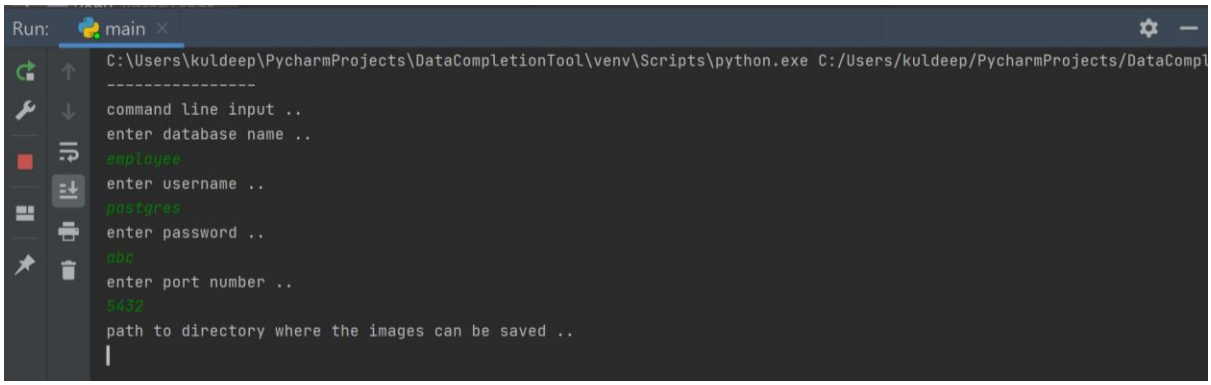


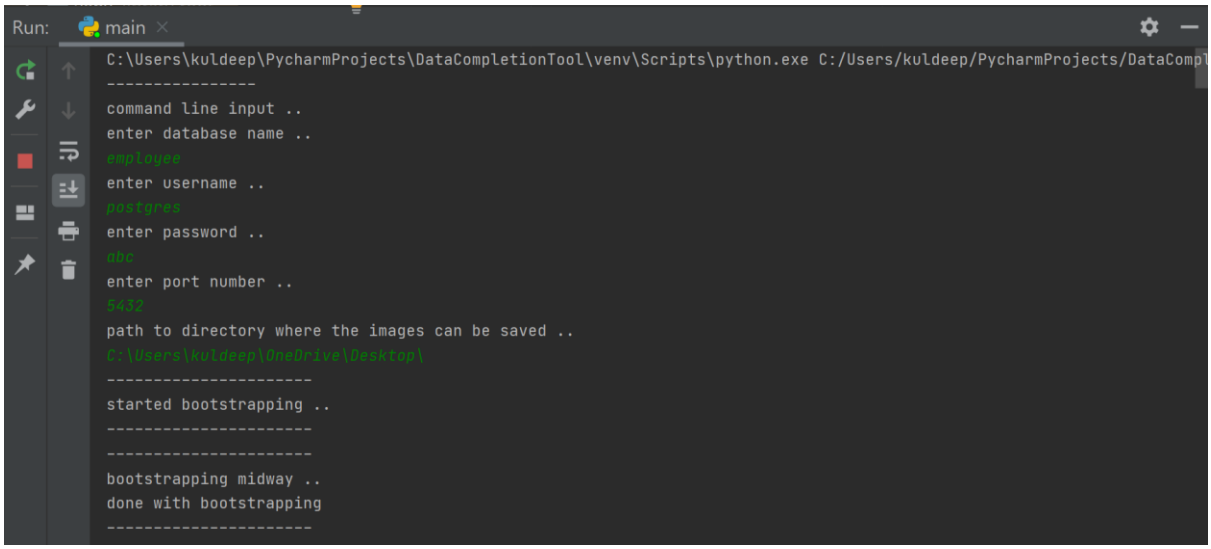
Once the database is prepared, the input to the tool consists of a series of entries in the form of database name, username, password, port number and path of where the images and gifs must be stored.











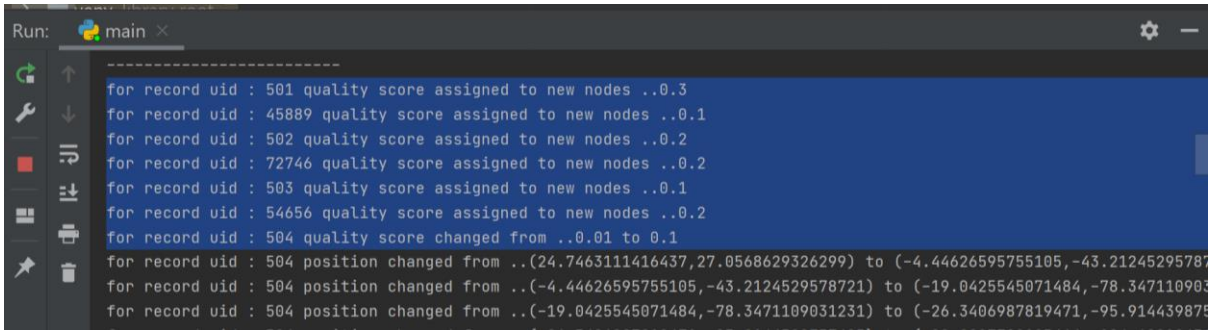
3. Output

The output consists of a sequence of steps of visualizations, altering data completion rules and efficiently executing them

a) Visualization

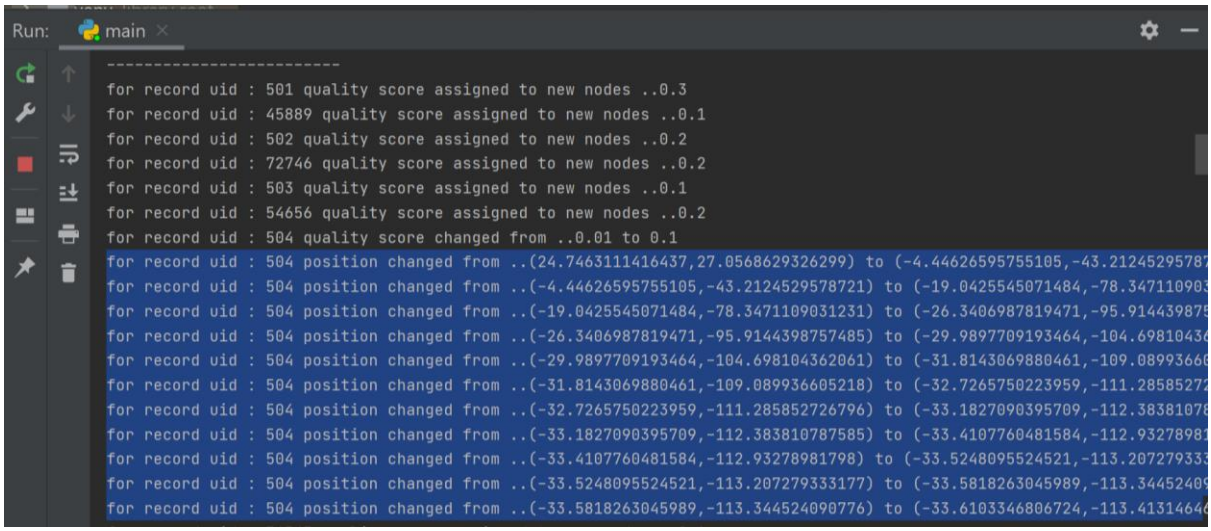
In the visualization part, spatial positions are assigned to data records corresponding to display positions. Their color intensity is continuously modified based on movement arising due to application of record matching rules.

i. Quality Score



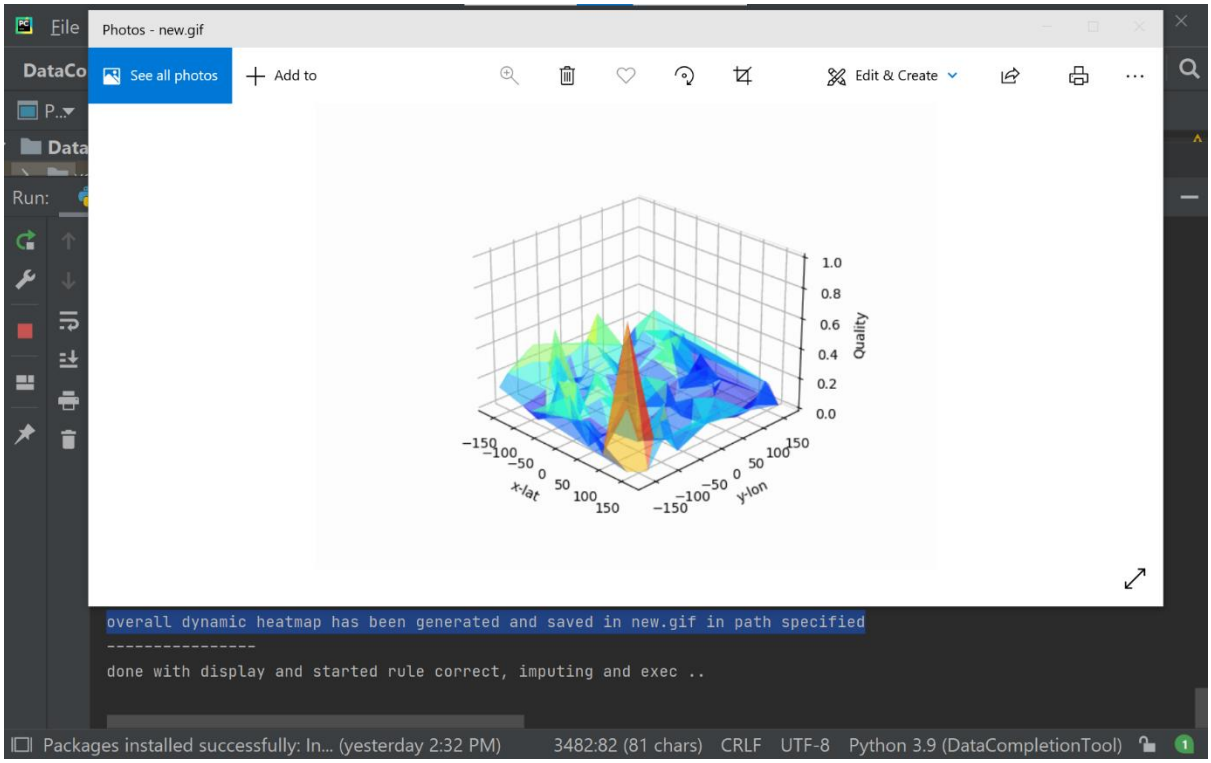
ii. Change in position of nodes

The spatial positions of nodes are also continuously changing due to the application of record matching rules and leveraging their properties



iii. Generated heatmap

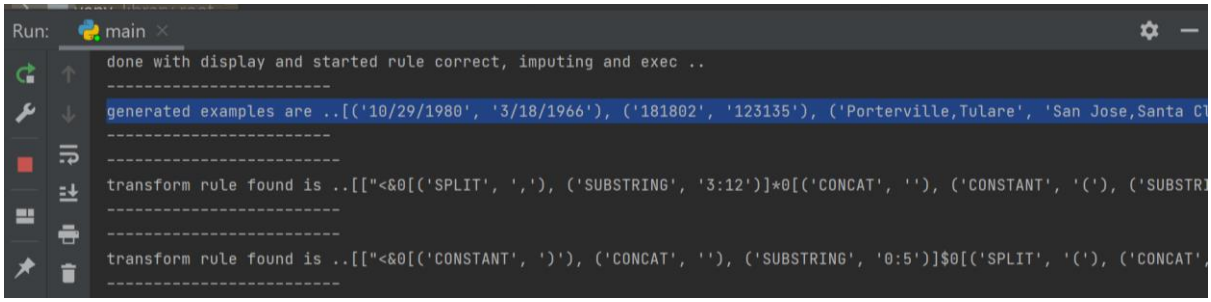
Apart from the textual generated output, the tool also produces output in the form of dynamic heatmap that reflects continuously changing quality scores and display spatial locations of records based on application of record matching rules with their properties.



b) Transformation and execution

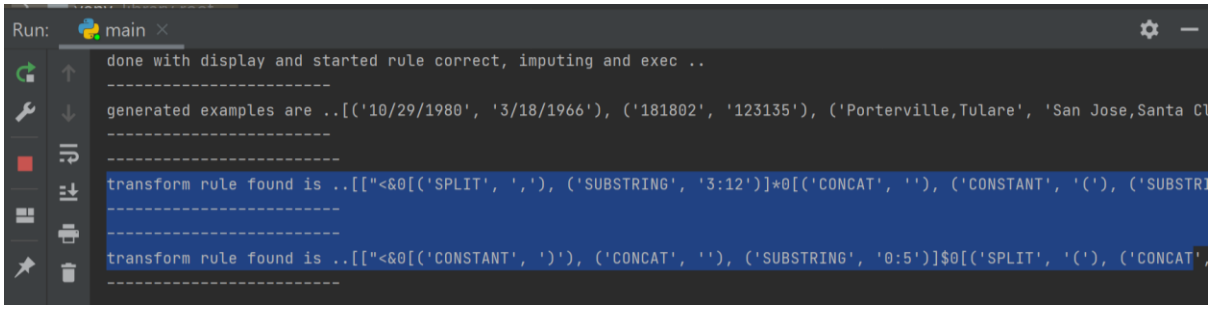
i. Generated examples

Next, the tool generates examples for data transformations from the database using combination of data completion rules and record matching rules along with their properties



ii. Generated transform rule

Next, the tool produces the actual data transformation rules from the examples generated In previous step

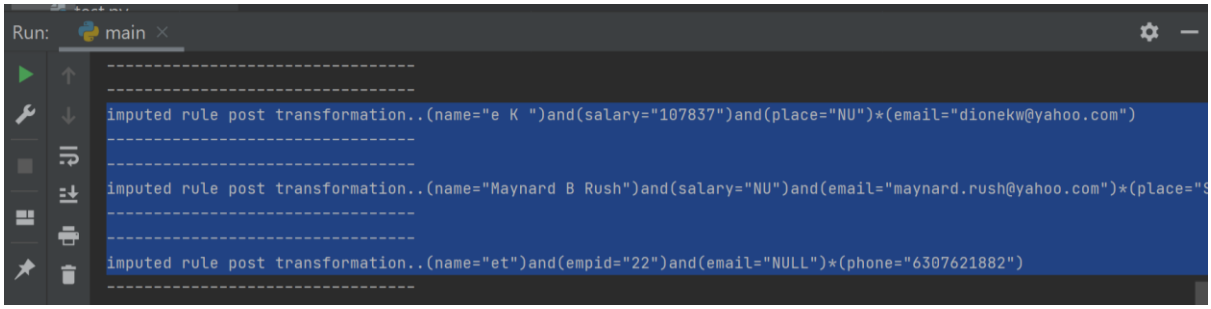


iii. Altered rules

Next, the tool produces a set of altered rules via the application of earlier data transformation rules generated

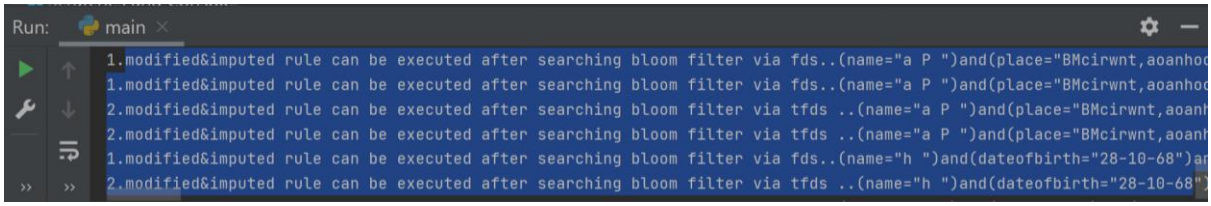
iv. Imputed rules

Next, the tool produces a set of imputed data completion rules with NULL values in the data completion rule filled in from record matching rules



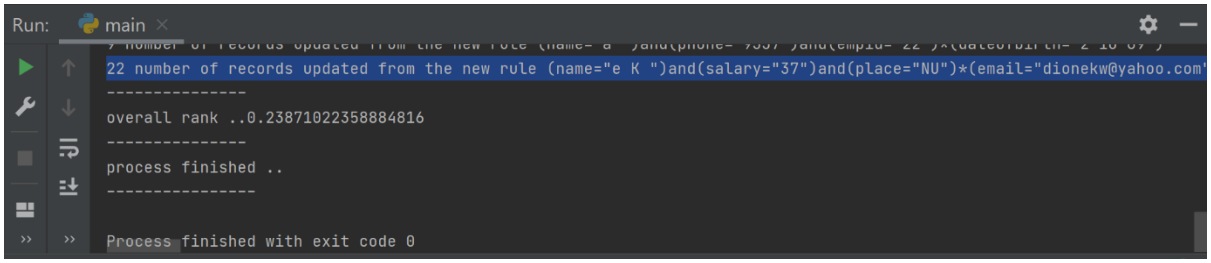
v. Modified rules executable found via summary

Next, we identify a set of data completion rules that can be executed using a summary of incomplete database built using combination of data completion rules and record matching rules



vi. Records updated using modified rule

Next, we apply the modified data completion rules on the incomplete database and compute the number of records modified as a result



vii. Overall rank

Next we compute the overall rank for the data completion process leveraging the queries table