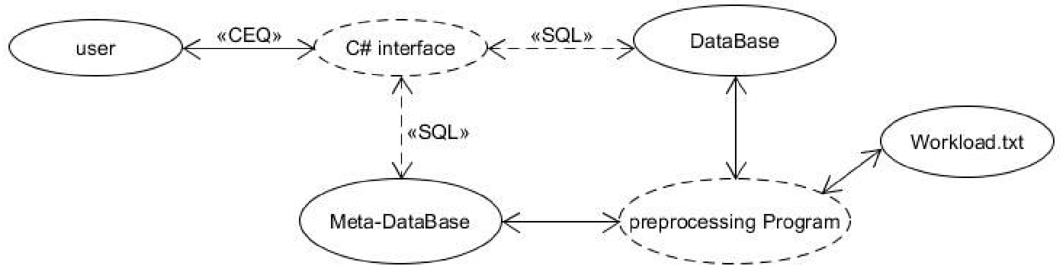
Lukas Donkers - 5523265
Roel van Warmerdam - 4300556

The goal of our application will be to create a metadatabase containing QF, IDF and Attribute Similarity data. After the metadatabase is complete, the user can enter CEQs (queries) into our application, along with a k-number, and our application will return the top-k relevant document results from the initial database.

When the CEQ returns answers, there are two possible problems: either there are too many results, or there are no results.
The many results problem can be solved by using QF, IDF and Attribute Similarity data to find the most relevant problems, and only select the best ones, therefore reducing the amount of results.
The zero results problem is solved the same way: because each document gets a ranking, there will always be documents to show.



Architecture sketch

- In order to create the metadatabase, we must do preprocessing using the initial database and workload file. This will calculate and store all QF, IDF and AS values as described below.
- In order to handle CEQs, the application must be able to retrieve the relevant metadata, calculate the score and ranking, and then find the top-k documents using that ranking.

Schema design metadatabase

- QF: For each column, and each possible value of that column (from the workload), we calculate the QF using (value) = (RQF(value)+1) / (RQFMAX(column)+1) (the +1 prevents a QF of 0 which might return bad results).We store these QF in the metaDatabase per column.
- IDF: For each column, and each possible value of that column (in autompg.sql),we count the amount of times that value exists in the database and calculate the IDF using IDF = log(N/F(w)) where N is the amount of tuples and F(w) the amount of times that value exists (in that column). We store these IDF in the metaDatabase per column
- Attribute similarity: For each column, and each possible value of that column (from the workload), we calculate the Attribute Similarity W(value) as the set of queries in which it

appears. We store these attribute similarities in the metadatabase per column. Then we can calculate the J(W(t) ,W(q)) = (| W(t) ∩ W(q) |) / (| W(t) ∪ W(q) |) which is the attribute similarity.

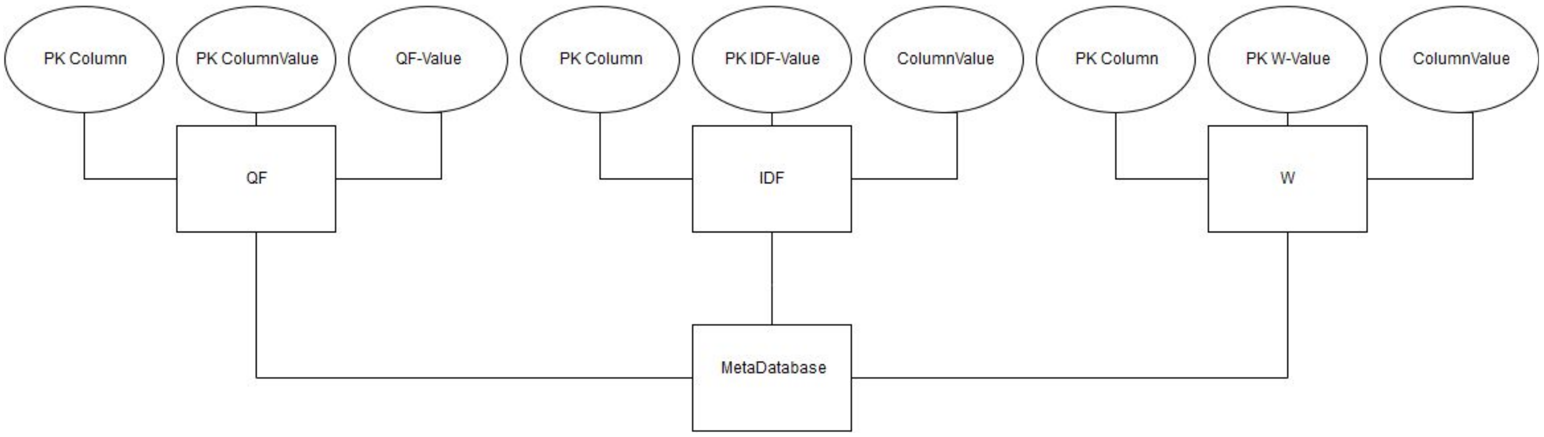
- sim(T,Q): The similarity between a query and a tuple is
  S(t, q) = QF(q) * IDF(q) * J(W(t),W(q)) if t = q, and 0 otherwise. We use the IDF here because without it the similarity is purely based on the workload.

In the end, when given a query, the ranking for any document can be calculated using the metadatabase values. For each term in the query, we find its QF, Attribute Similarity and IDF, and calculate S(t,q). We sum all these S(t,q) to form sim(T, Q), which is the final score used for ranking.

This means that, for each method (QF, IDF or Attribute similarity), we create one table. This table will have 2 primary keys: the column (e.g. MPG or cylinders) and the value (e.g. 18 or 15). The third value in each tuple is then the actual QF value.

QF

PK Column	**PK ColumnValue**	**QF-Value**
MPG	28	QF(MPG28) = (100+24+8)/RQFMAX$_{MPG}$ = 0..1
MPG	23	QF(MPG23)
...		
Cylinders	4	QF(Cylinders4)
Cylinders	6	QF(Cylinders6)


PK Column
PK ColumnValue
QF-Value
PK Column
PK IDF-Value
ColumnValue
PK Column
PK W-Value
ColumnValue
QF
IDF
W
MetaDatabase