CS 562 Final Project

Olivia Eng, Jakob Gibson and Nick Cali

Problem Overview

- SQL queries are powerful tools
- basic SQL syntax makes expressing complicated or interconnected OLAP queries hard
- this is because one cannot "decouple" the formation of groups (the **group by** clause) and the computation of aggregates (such as **max**, **min**, **count**, **sum**, and **avg**)
  - i.e. one must, in standard SQL, compute these things in distinct queries and join them all together at the end

Solution

Luckily, there is a solution! Introduced in two papers:

1. *Querying Multiple Features of Groups in Relational Databases* by D. Chatziantoniou and K. Ross
2. *Evalutaion of Ad Hoc OLAP: In-Place Computation* by D. Chatziantoniou

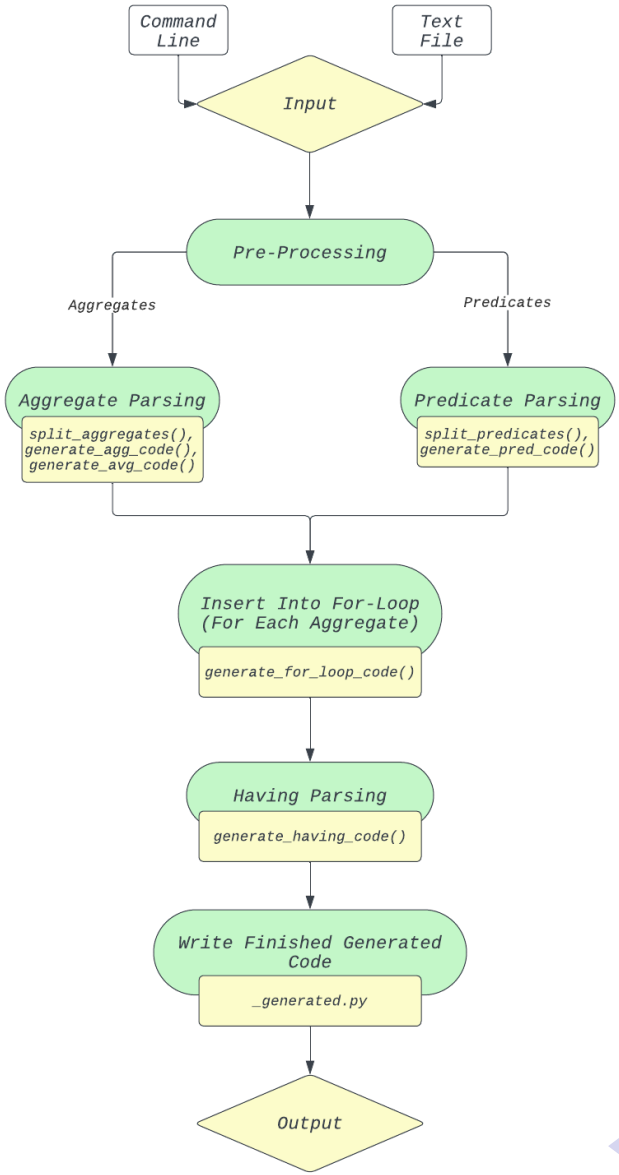
the Φ operator (an extension to relational algebra) is a way for us to express ourselves (and these complex OLAP queries) concisely.

- However, this being an addition to relational algebra, it does not have a direct implementation in SQL (that is, it is not found in any common implementation of SQL [MySQL, PostgreSQL, etc.])
- We thus wrote code that generates a file that, when ran, produces the same output as the phi operator detailed in the papers.

Solution: Implementation Specifics

1. Flow of query processing
2. Query structure (how the query is inputted)
3. Technologies used
4. Limitations of the project

Flow of Query Processing



Query Structure

Input can come from either:

1. Command Line
2. Text File

but both are formatted using the phi operator as a template:

1. **cust, min_1_quant, max_1_quant, min_2_quant, max_2_quant** $\rightarrow$ select clause
2. **2** $\rightarrow$ number of grouping variables
3. **cust** $\rightarrow$ grouping attribute(s)
4. **min_1_quant, max_1_quant, min_2_quant, max_2_quant** $\rightarrow$ aggregates to be computed
5. **2.state = 'NJ'** $\rightarrow$ predicates (if no number [e.g. **state = 'NJ'**], acts like the where clause, if no predicate is listed for a number that exists in the select clause, computes aggregate for the whole table [as shown above with 1])
6. **None** $\rightarrow$ having clause

Technologies Used

- Python 3.10 was the language of choice here
  - a good choice where inputs need to be parsed
  - supported by many robust features and libraries that make other necessary steps easier to implement (including the **psycopg2** library, the PostgreSQL API for Python, and the **tabulate** library, used to make the output of our program more human friendly to read [a lot of formatting is done automatically])
  - extremely common and well-known language

Technical Limitations of our Implementation

Despite our best efforts and intentions, there are some limitations to our final product:

- no error-checking nor sanitizing of inputs. This does include:
  - spaces
  - spelling
- in the **such that** clause (input 5), aggregates computed can be used in future predicates but not the other way around (i.e. grouping variable 2's predicate *can* be defined relative to the output of an aggregate of grouping variable 1, but not vice versa)
- in the **such that** clause, the predicates are assumed to be connected using *and*; as such *or* is not well-defined (e.g. one cannot have **1.state = 'NJ' or 1.state = 'NY'** as a predicate)