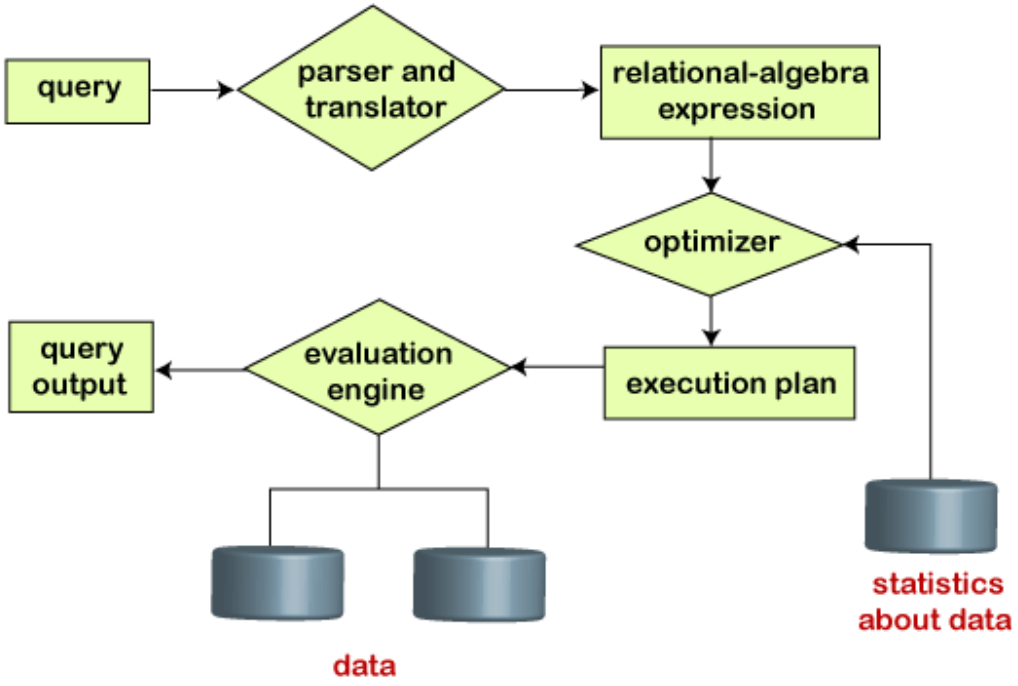
5. QUERY PROCESSING IN DBMS.

Query Processing is the activity performed in extracting data from the database. In query processing, it takes various steps for fetching the data from the database. The steps involved are:

1. Parsing and translation
2. Optimization
3. Evaluation



Steps in query processing

The query processing works in the following way:

Parsing and Translation

- The scanning, parsing, and validating module produces an internal representation of the query. The query optimizer module devises an execution plan which is the execution strategy to retrieve the result of the query from the database files.
- A query typically has many possible execution strategies differing in performance, and the process of choosing a reasonably efficient one is known as query optimization.
- The code generator generates the code to execute the plan. The runtime database processor runs the generated code to produce the query result.
- Relational algebra is well suited for the internal representation of a query.

The translation process in query processing is similar to the parser of a query. When a user executes any query, for generating the internal form of the query, the parser in the system checks the syntax of the query, verifies the name of the relation in the database, the tuple, and finally the required attribute value. The parser creates a tree of the query, known as 'parse-tree.' Further, translate it into the form of relational algebra. With this, it evenly replaces all the use of the views when used in the query.

It is done in the following steps:

Step-1:

Parser: During parse call, the database performs the following checks- Syntax check, Semantic check and Shared pool check, after converting the query into relational algebra.

Parser performs the following checks as (refer detailed diagram):

1. **Syntax check –** concludes SQL syntactic validity. Example:

   SELECT * FORM employee

   Here error of wrong spelling of FROM is given by this check.
2. **Semantic check –** determines whether the statement is meaningful or not. Example: query contains a table name which does not exist is checked by this check.
3. **Shared Pool check –** Every query possess a hash code during its execution. So, this check determines existence of written hash code in shared pool if code exists in shared pool then database will not take additional steps for optimization and execution.

Hard Parse and Soft Parse –

If there is a fresh query and its hash code does not exist in shared pool then that query has to pass through from the additional steps known as hard parsing otherwise if hash code exists then query does not passes through additional steps. It just passes directly to execution engine (refer detailed diagram). This is known as soft parsing.

Hard Parse includes following steps – Optimizer and Row source generation.

Step-2:

Optimizer: During optimization stage, database must perform a hard parse atleast for one unique DML statement and perform optimization during this parse. This database never optimizes DDL unless it includes a DML component such as subquery that require optimization.

It is a process in which multiple query execution plan for satisfying a query are examined and most efficient query plan is satisfied for execution. Database catalog stores the execution plans and then optimizer passes the lowest cost plan for execution.

Step-3:

Execution Engine: Finally runs the query and display the required result.

Thus, we can understand the working of a query processing in the below-described diagram:

Suppose a user executes a query. As we have learned that there are various methods of extracting the data from the database. In SQL, a user wants to fetch the records of the employees whose salary is greater than or equal to 10000. For doing this, the following query is undertaken:

SELECT EMP_NAME FROM EMPLOYEE WHERE SALARY>10000;

Thus, to make the system understand the user query, it needs to be translated in the form of relational algebra. We can bring this query in the relational algebra form as:

- $\sigma_{salary>10000}\ (\pi_{Emp\_Name}(Employee))$
- $\pi_{Emp\_Name}(\sigma_{salary>10000}\ (Employee))$

After translating the given query, we can execute each relational algebra operation by using different algorithms. So, in this way, a query processing begins its working.

Evaluation

For this, with addition to the relational algebra translation, it is required to annotate the translated relational algebra expression with the instructions used for specifying and evaluating each operation. Thus, after translating the user query, the system executes a query evaluation plan.

Query Evaluation Plan

- In order to fully evaluate a query, the system needs to construct a query evaluation plan.
- A query evaluation plan defines a sequence of primitive operations used for evaluating a query. The query evaluation plan is also referred to as **the query execution plan**.
- A **query execution engine** is responsible for generating the output of the given query. It takes the query execution plan, executes it, and finally makes the output for the user query.

Optimization

- The cost of the query evaluation can vary for different types of queries. Although the system is responsible for constructing the evaluation plan, the user does need not to write their query efficiently.
- Usually, a database system generates an efficient query evaluation plan, which minimizes its cost. This type of task performed by the database system and is known as Query Optimization.
- For optimizing a query, the query optimizer should have an estimated cost analysis of each operation. It is because the overall operation cost depends on the memory allocations to several operations, execution costs, and so on.

Finally, after selecting an evaluation plan, the system evaluates the query and produces the output of the query.

Example:

SELECT LNAME, FNAME FROM EMPLOYEE WHERE SALARY > (SELECT MAX (SALARY) FROM EMPLOYEE WHERE DNO=5);

The inner block

(SELECT MAX (SALARY) FROM EMPLOYEE WHERE DNO=5)

- Translated in: ∏ MAX SALARY (σDNO=5(EMPLOYEE))

The Outer block

SELECT LNAME, FNAME FROM EMPLOYEE WHERE SALARY > C

- Translated in: ∏ LNAZME, FNAME (σSALARY>C (EMPLOYEE))

(C represents the result returned from the inner block.)

- The query optimizer would then choose an execution plan for each block.
- The inner block needs to be evaluated only once. (Uncorrelated nested query).
- It is much harder to optimize the more complex correlated nested queries.

External Sorting

It refers to sorting algorithms that are suitable for large files of records on disk that do not fit entirely in main memory, such as most database files..

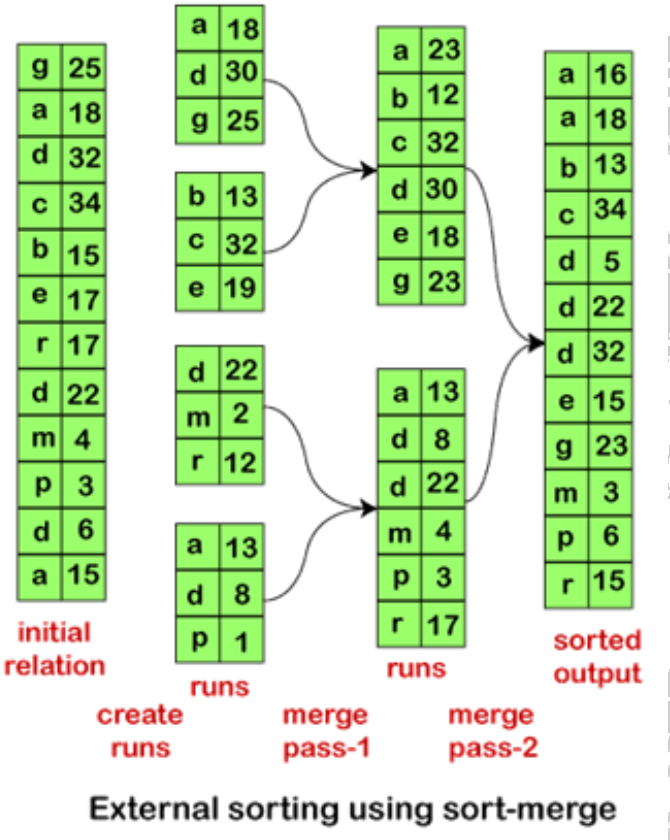
ORDER BY.

Sort-merge algorithms for JOIN and other operations (UNION, INTERSECTION). Duplicate elimination algorithms for the PROJECT operation (DISTINCT).

Typical external sorting algorithm uses a sort-merge strategy:

Sort phase: Create sort small sub-files (sorted sub-files are called runs).

Merge phase: Then merges the sorted runs. N-way merge uses N memory buffers to buffer input runs, and 1 block to buffer output. Select the 1st record (in the sort order) among input buffers, write it to the output buffer and delete it from the input buffer. If output buffer full, write it to disk. If input buffer empty, read next block from the corresponding run. E.g. 2-way Sort-Merge



External sorting using sort-merge