**Summary –Day4**

**Name:Tejaswini Gokanakonda**

## **Roll no**:**DE142**

## **Date:08-11-2024**

**MySQL REGEXP**

**Pattern Matching**: MySQL supports pattern matching using regular expressions with the REGEXP operator (synonym: RLIKE).

**Metacharacters**:

* \* - Matches zero or more instances of the preceding string.
* + - Matches one or more instances of the preceding string.
* . - Matches any single character.
* ? - Matches zero or one instance of the preceding string.
* ^ - Matches the beginning of a string.
* $ - Matches the end of a string.
* [abc] - Matches any character listed within the brackets.
* [^abc] - Matches any character not listed within the brackets.
* [A-Z] and [a-z] - Matches uppercase and lowercase letters, respectively.
* [0-9] - Matches any digit.
* [:<:] and [:>:] - Matches beginning and end of words, respectively.
* [:class:] - Matches character classes, like [:alpha:] for letters or [:space:] for whitespace.

**Examples**:

* **Beginning of String**: '^sa' matches names starting with "sa".
* **End of String**: 'on$' matches names ending in "on".
* **Pattern Alternation**: 'be|ae' matches either "be" or "ae".
* **Character Classes**: '[:alpha:]' for alphabetic characters, [a-z] for lowercase.
* **Word Boundaries**: '^n' matches names starting with "n".

**OVER and PARTITION BY clauses in SQL**

* **OVER & PARTITION BY Clauses**:
  + The PARTITION BY clause is part of the OVER clause, used in SQL window functions like AVG(), MAX(), and RANK().
  + Window functions operate on a set of rows (window frames) that can vary based on each record in the query.
* **Syntax Example**:

**Example query:**SELECT car\_make, car\_model, car\_price,

AVG(car\_price) OVER() AS "overall average price",

AVG(car\_price) OVER (PARTITION BY car\_type) AS "car type average price"

FROM car\_list\_prices;

* + This calculates the overall average car price and the average price by car type.
* **Comparison to GROUP BY**:
  + GROUP BY groups records and applies aggregate functions, collapsing individual records.
  + In contrast, window functions with PARTITION BY maintain access to each record’s fields.
* **Window Frames and Bounds**:
  + Default bounds for window frames include options like UNBOUNDED PRECEDING, CURRENT ROW, and UNBOUNDED FOLLOWING.
  + The ORDER BY within the OVER clause organizes records in a window for functions like LEAD() and LAG().

This covers the main uses and syntax of the OVER and PARTITION BY clauses along with examples.

**GROUP BY, FILTER, and data aggregation**

**GROUP BY Clause:**

* Groups data and applies aggregate functions like AVG or SUM for summary data.
* Ensures all non-aggregated columns in SELECT are also in the GROUP BY clause for accurate results.
* Example: Calculate average deal value by sales agent.

**Filtering with JOINs and GROUP BY**:

* Joins allow grouping by attributes across tables.
* Example: Find average deal values by manager using sales and team data.

**FILTER Modifier**:

* Applies conditions within aggregate functions to refine results.
* Example: Count deals over a certain value with FILTER(WHERE condition).

**WHERE vs. FILTER**:

* WHERE limits data before aggregation; FILTER can refine multiple aggregates in the same query.

**Practical Examples**:

* Aggregate sales data, calculate averages, use constraints (UNIQUE, NOT NULL, CHECK) on table columns.
* Demonstrations of joins (UNION, INTERSECT, EXCEPT) and managing table relations with PRIMARY KEY and FOREIGN KEY.