SQL ADVANCED

Cheatsheeeeet

Aggregate Functions

Aggregate functions perform calculations on a set of rows and return a single value, often used with **GROUP BY**

- **COUNT - Counts the number of rows in a column or table**
  SELECT **COUNT** (column_name) FROM table_name;
- **SUM - Calculates the total sum of numeric values in a column**
  SELECT **SUM** (column_name) FROM table_name;
- **AVG - Returns the average of numeric values in a column**
  SELECT **AVG** (column_name) FROM table_name;
- **MIN - Finds the smallest value in a column**
  SELECT **MIN** (column_name) FROM table_name;
- **MAX - Finds the largest value in a column**
  SELECT **MAX** (column_name) FROM table_name;

String Functions

String functions help manipulate and retrieve information from text data

- **CONCAT - Combines two or more strings into a single string**
  SELECT **CONCAT** (string1, string2);
- **SUBSTRING - Extracts a specific portion of a string based on starting position and length**
  SELECT **SUBSTRING** (column_name, start_position, length) FROM table_name;
- **LENGTH - Returns the total number of characters in a string, including spaces**
  SELECT **LENGTH** (column_name) FROM table_name;
- **TRIM - Removes leading and trailing spaces or specific characters from a string**
  SELECT **TRIM** (' ' FROM column_name) FROM table_name;

Date and Time Functions

Date and time functions allow operations on and extraction of date/time values

- **NOW - Returns the current date and time of the system or database.**
  SELECT **NOW** ();
- **DATEDIFF - Calculates the difference in days between two dates**
  SELECT **DATEDIFF** (date1, date2);
- **YEAR - Extracts the year part of a date value**
  SELECT **YEAR** (column_name) FROM table_name;
- **EXTRACT - Retrieves a specific component, such as year or month, from a date or time**
  SELECT **EXTRACT** (part FROM date_column) FROM table_name;

JSON Functions

JSON functions enable querying and manipulation of JSON data stored in db columns

- **JSON_OBJECT - Creates a JSON object using key-value pairs provided in the query**
  SELECT **JSON_OBJECT** ('key', value);
- **JSON_EXTRACT - Extracts specific values from a JSON object based on a given path**
  SELECT **JSON_EXTRACT** (column_name, '$.path') FROM table_name;

Join Operations

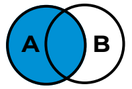
Join operations combine rows from two or more tables based on a related column, allowing complex queries across multiple datasets

- **INNER JOIN - Returns rows that have matching values in both tables**

```
SELECT table1.column_name, table2.column_name
FROM table1
INNER JOIN table2 ON table1.common_column = table2.common_column;
```

- **LEFT JOIN - Returns all rows from the left table, and the matching rows from the right table; unmatched rows from the right table are null**

```
SELECT table1.column_name, table2.column_name
FROM table1
INNER JOIN table2 ON table1.common_column = table2.common_column;
```

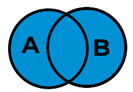


- **RIGHT JOIN - Returns all rows from the right table, and the matching rows from the left table; unmatched rows from the left table are null**

```
SELECT table1.column_name, table2.column_name
FROM table1
RIGHT JOIN table2 ON table1.common_column = table2.common_column;
```

- **FULL OUTER JOIN - Returns rows when there is a match in either table; unmatched rows are filled with nulls.**

```
SELECT table1.column_name, table2.column_name
FROM table1
FULL OUTER JOIN table2 ON table1.common_column = table2.common_column;
```



Window Functions

Window functions perform calculations across a defined range of rows that are related to the current row

- **ROW_NUMBER - Assigns a unique sequential number to each row within a partition of the result set**
  SELECT **ROW_NUMBER** () OVER (PARTITION BY column_name ORDER BY column_name) AS row_num FROM table_name;
- **RANK - Assigns a rank to each row within a partition, allowing for gaps in the ranking when there are ties**
  SELECT **RANK** () OVER (PARTITION BY column_name ORDER BY column_name) AS rank_num FROM table_name;
- **LEAD - Retrieves the value of the next row in the result set relative to the current row**
  SELECT **LEAD** (column_name) OVER (ORDER BY column_name) FROM table_name;
- **LAG - Retrieves the value of the previous row in the result set relative to the current row**
  SELECT **LAG** (column_name) OVER (ORDER BY column_name) FROM table_name;

Common Table Expressions (CTEs)

CTEs define temporary result sets that are more reusable within a query

- **BASIC CTE - Simplifies complex queries by defining a temporary result set with a name**

```
WITH cte_name AS (SELECT column_name FROM table_name WHERE condition)
SELECT * FROM cte_name;
```

- **RECURSIVE CTE- Allows hierarchical queries by iteratively processing data until a specified condition is met**

```
WITH RECURSIVE cte_name AS (
    SELECT column_name
    FROM table_name WHERE condition
    UNION ALLSELECT column_name FROM cte_name )
SELECT * FROM cte_name;
```

Mathematical Functions

Mathematical functions are used to perform arithmetic and advanced numeric operations on data

- **ROUND - Rounds a numeric value to a specified number of decimal places**
  SELECT **ROUND** (column_name, decimal_places) FROM table_name;
- **POWER - Calculates the result of raising a number to a specified power**
  SELECT **POWER** (column_name, exponent) FROM table_name;
- **SQRT - Returns the square root of a numeric value**
  SELECT **SQRT** (column_name) FROM table_name;
- **MOD - Finds the remainder when one number is divided by another**
  SELECT **MOD** (column_name, divisor) FROM table_name;

Logical Functions

Logical functions evaluate conditions and provide outputs based on specific criteria or handle special values like **NULL**

- **CASE - Implements conditional logic to return different values based on specified conditions**

```
SELECT CASE
            WHEN condition THEN result
            ELSE default_result
        END AS alias_name
FROM table_name;
```

- **NULLIF - Returns `NULL` if two expressions are equal; otherwise, it returns the first expression**
  SELECT **NULLIF** (column1, column2) FROM table_name;
- **COALESCE - Returns the first non-null value from a list of expressions or columns**
  SELECT **COALESCE** (column1, column2, default_value) FROM table_name;

Set Operations

Set operations combine results from two or more queries into a single dataset, with or without duplicate rows.

- **UNION - Combines the results of two queries into one, excluding duplicates**

```
SELECT column_name FROM table_name1
UNION
SELECT column_name FROM table_name2;
```

- **UNION ALL - Combines the results of two queries into one, including duplicates**

```
SELECT column_name FROM table_name1
UNION ALL
SELECT column_name FROM table_name2;
```

- **INTERSECT - Returns rows that are common in the results of both queries**

```
SELECT column_name FROM table_name1
INTERSECT
SELECT column_name FROM table_name2;
```

- **EXCEPT / MINUS - Returns rows that are present in the first query but not in the second**

```
SELECT column_name FROM table_name1
EXCEPT
SELECT column_name FROM table_name2;
```

SAI RAAM
YOUR FRIENDLY NEIGBORHOOD DATA GUY