Experiment No.5
Perform simple queries, string manipulation operations and aggregate functions.
Date of Performance:
Date of Submission:

Aim :- Write simple query to manipulate string operations and perform aggregate functions like (MIN, MAX, SUM, AVERAGE, COUNT).

Objective :- To apply aggregate functions and string manipulation functions to perform simple queries in the database system

Theory:

Simple Queries in SQL:

In SQL, a simple query is a request for data from a database table or tables. It allows users to retrieve specific information by specifying the columns they want to retrieve and any conditions for filtering rows based on certain criteria. Simple queries are the backbone of interacting with databases, enabling users to extract the data they need for analysis, reporting, or further processing.

String Manipulation Operations:

String manipulation operations in SQL involve modifying or transforming string values stored in database columns. These operations are crucial for tasks such as formatting data, combining strings, converting case, or extracting substrings. By using string functions and operators, users can manipulate text data to suit their requirements, whether it's for display purposes or for further analysis.

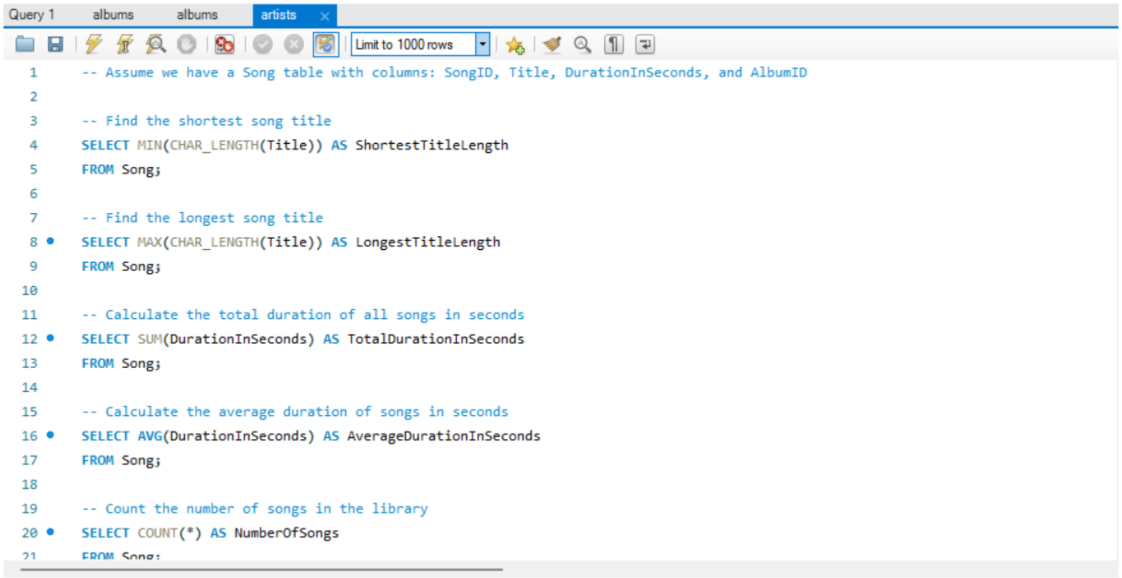
Aggregate Functions:

Aggregate functions in SQL are used to perform calculations on sets of values and return a single result. These functions allow users to summarize data across multiple rows, providing insights into the overall characteristics of the dataset. Common aggregate functions include calculating counts, sums, averages, minimums, and maximums of numerical values. They are essential tools for data analysis, enabling users to derive meaningful insights from large datasets.

Benefits of Understanding These Concepts:

- Data Retrieval: Simple queries allow users to fetch specific data from databases, facilitating data retrieval for various purposes.
- Data Transformation: String manipulation operations enable users to format and transform text data according to their needs, improving data consistency and readability.
- Data Analysis: Aggregate functions help users summarize and analyze large datasets, providing valuable insights into trends, patterns, and statistical measures.
- Data Reporting: By combining simple queries, string manipulation operations, and aggregate functions, users can generate reports and visualizations that communicate key findings effectively.

Implementation:



```sql
-- Assume we have a Song table with columns: SongID, Title, DurationInSeconds, and AlbumID

-- Find the shortest song title
SELECT MIN(CHAR_LENGTH(Title)) AS ShortestTitleLength
FROM Song;

-- Find the longest song title
SELECT MAX(CHAR_LENGTH(Title)) AS LongestTitleLength
FROM Song;

-- Calculate the total duration of all songs in seconds
SELECT SUM(DurationInSeconds) AS TotalDurationInSeconds
FROM Song;

-- Calculate the average duration of songs in seconds
SELECT AVG(DurationInSeconds) AS AverageDurationInSeconds
FROM Song;

-- Count the number of songs in the library
SELECT COUNT(*) AS NumberOfSongs
FROM Song;
```

Conclusion:

1. Write syntax and explanation for each of the five aggregate functions

Aggregate functions are SQL functions used to perform calculations on sets of values and return a single result. Here's a brief explanation of each aggregate function along with the syntax and an example of its usage:

MIN():

- **Syntax:** `MIN(expression)`
- **Explanation:** Returns the minimum value of a set.
- **Example:** Find the shortest song title.

MAX():

- **Syntax:** `MAX(expression)`
- **Explanation:** Returns the maximum value of a set.
- **Example:** Find the longest song title.

SUM():

- **Syntax:** `SUM(expression)`

- **Explanation:** Calculates the sum of all values in a set.
- **Example:** Calculate the total duration of all songs in seconds.

AVG():

- **Syntax:** `AVG(expression)`
- **Explanation:** Calculates the average of all values in a set.
- **Example:** Calculate the average duration of songs in seconds.

COUNT():

- **Syntax:** `COUNT(*)` or `COUNT(expression)`
- **Explanation:** Counts the number of rows in a set. `COUNT(*)` counts all rows, while `COUNT(expression)` counts rows where the expression is not null.
- **Example:** Count the number of songs in the library.