Data Query Language

- *Single Row and Multi Row Functions*

Presented By:
- *Aniket Bhagwate*
- *Gurkirat Nagpal*
- *Ayush Shah*

Introduction

What is SQL ?

=> SQL is a standard language for accessing and manipulating databases.

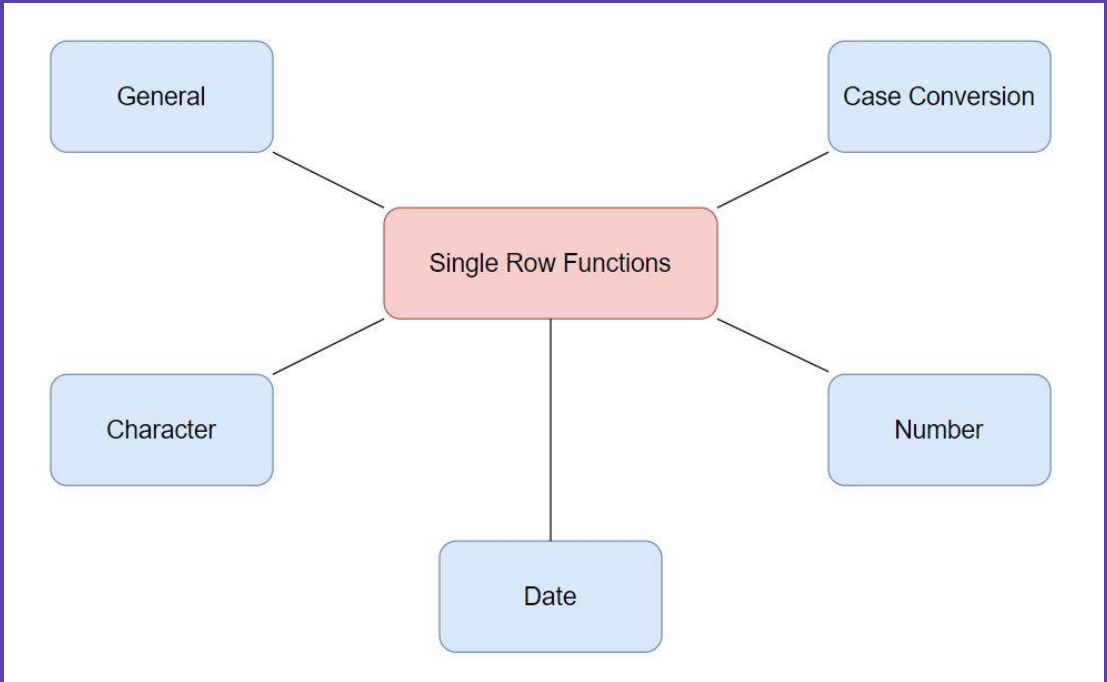
What is DQL ?

=> SQL commands are mainly categorized into four categories is a standard language for accessing and manipulating databases.

DDL –Data Definition Language
DQL – Data Query Language
DML – Data Manipulation Language
DCL – Data Control Language

Single Row Functions

=> Single Row functions - Single row functions are the one who work on single row and return one output per row.

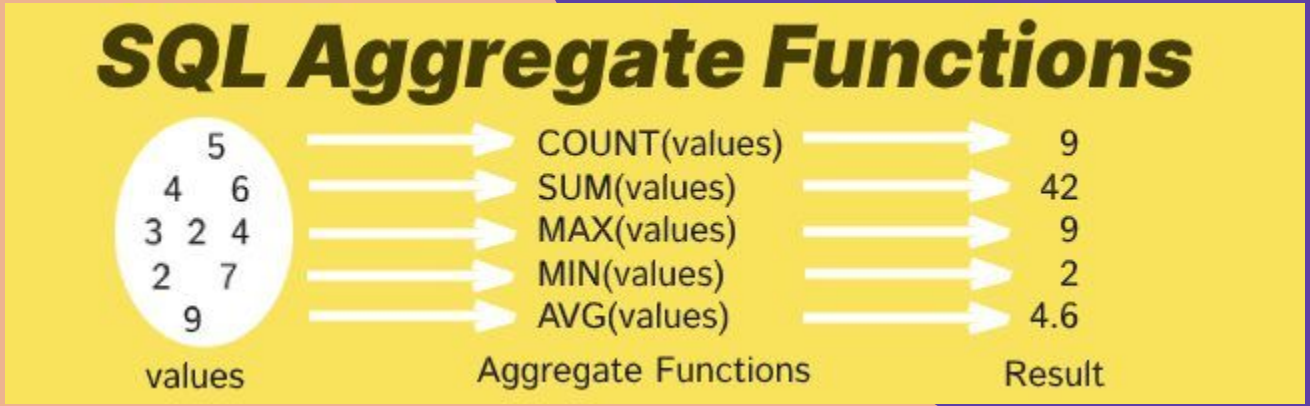


DEMO.....

Multi-Row Functions

Aggregate Functions

They operate on a set of rows and returns one result or one result per group example - min, max, sum, avg and count.

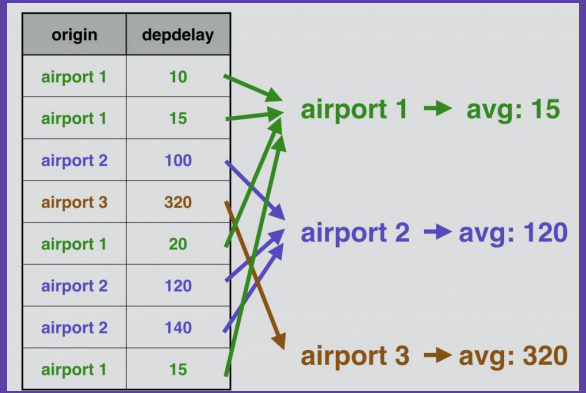


Credits: https://codingstatus.com/

Aggregation with Grouping

Group by clause

There are circumstances where we would like to apply the aggregate function not only to a single set of tuples, but also to a group of sets of tuples; we specify this wish in SQL using the group by clause. The attribute or attributes given in the group by clause are used to form groups.



Credit: https://data36.com/

Having Clause

At times, it is useful to state a condition that applies to groups rather than to tuples.This condition does not apply to a single tuple; rather, it applies to each group constructed by the group by clause. To express such a query, we use the having clause of SQL. SQL applies predicates in the having clause after groups have been formed, so aggregate functions may be used.

Credit: https://w3resorces.com/

Sub Queries Introduction

- A sub query is a SQL query nested inside a larger query.
- The subquery can be nested inside a SELECT, INSERT, UPDATE, or DELETE statement or inside another subquery.
- A subquery is usually added within the WHERE Clause of another SQL SELECT statement.
- You can use the comparison operators, such as >, <, or =. The comparison operator can also be a multiple-row operator, such as IN, ANY, or ALL.

Sub Queries: Guild lines

- A sub query must be enclosed in parentheses.
- A sub query must be placed on the right side of the comparison operator.
- Sub queries cannot manipulate their results internally, therefore ORDER BY clause cannot be added into a subquery. You can use an ORDER BY clause in the main SELECT statement (outer query) which will be the last clause.
- Use single-row operators with single-row subqueries.
- If a sub query (inner query) returns a null value to the outer query, the outer query will not return any rows when using certain comparison operators in a WHERE clause.

Type of Sub queries

- Single row sub query : Returns zero or one row.
- Multiple row sub query : Returns one or more rows.
- Multiple column sub queries : Returns one or more columns.
- Correlated sub queries : Reference one or more columns in the outer SQL statement. The sub query is known as a correlated subquery because the sub query is related to the outer SQL statement.
- Nested sub queries : Sub queries are placed within another subquery.

SUBQUERIES WITH INSERT

```
MariaDB [students]> select * from Orders;
+---------+---------+---------+------------+-----------+------------+
| Ord_Num | Ord_Amt | Adv_Amt | Ord_Date   | Cust_Code | Agent_Code |
+---------+---------+---------+------------+-----------+------------+
|  200114 |    3500 |    2000 | 2008-08-15 | C00002    | A008       |
|  200122 |    2500 |     400 | 2008-09-16 | C00003    | A004       |
|  200118 |    1500 |     100 | 2008-07-20 | C00023    | A006       |
|  200119 |    4000 |     700 | 2008-09-16 | C00007    | A010       |
|  200121 |    1500 |     600 | 2008-09-23 | C00008    | A004       |
|  200130 |    2500 |     400 | 2008-08-30 | C00025    | A011       |
+---------+---------+---------+------------+-----------+------------+
6 rows in set (0.001 sec)
```

```
MariaDB [students]> create table NewOrders(Ord_Num int, Ord_Amt int, Adv_Amt int,Ord_Date Date,Cust_Code char(10),Agent_Code char(4));
Query OK, 0 rows affected (0.345 sec)

MariaDB [students]> INSERT INTO NewOrders SELECT * FROM Orders WHERE Adv_Amt in(2000,5000);
Query OK, 1 row affected (0.203 sec)
Records: 1  Duplicates: 0  Warnings: 0

MariaDB [students]> select * from NewOrders;
+---------+---------+---------+------------+-----------+------------+
| Ord_Num | Ord_Amt | Adv_Amt | Ord_Date   | Cust_Code | Agent_Code |
+---------+---------+---------+------------+-----------+------------+
|  200114 |    3500 |    2000 | 2008-08-15 | C00002    | A008       |
+---------+---------+---------+------------+-----------+------------+
1 row in set (0.000 sec)
```

SUBQUERIES WITH UPDATE

```
MariaDB [students]> select * from NewOrders;
+---------+---------+---------+------------+-----------+------------+
| Ord_Num | Ord_Amt | Adv_Amt | Ord_Date   | Cust_Code | Agent_Code |
+---------+---------+---------+------------+-----------+------------+
|  200114 |    3500 |    2000 | 2008-08-15 | C00002    | A008       |
+---------+---------+---------+------------+-----------+------------+
1 row in set (0.000 sec)
```

```
MariaDB [students]> UPDATE NewOrders SET Ord_Date ="2010-01-16" WHERE Ord_Amt-Adv_Amt <(SELECT MAX(Ord_Amt) FROM Orders);
Query OK, 1 row affected (0.136 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [students]> select * from NewOrders;
+---------+---------+---------+------------+-----------+------------+
| Ord_Num | Ord_Amt | Adv_Amt | Ord_Date   | Cust_Code | Agent_Code |
+---------+---------+---------+------------+-----------+------------+
|  200114 |    3500 |    2000 | 2010-01-16 | C00002    | A008       |
+---------+---------+---------+------------+-----------+------------+
1 row in set (0.001 sec)
```

SUBQUERIES WITH DELETE OPERATION

```
MariaDB [students]> select * from NewOrders;
+---------+---------+---------+------------+-----------+------------+
| Ord_Num | Ord_Amt | Adv_Amt | Ord_Date   | Cust_Code | Agent_Code |
+---------+---------+---------+------------+-----------+------------+
|  200114 |    3500 |    2000 | 2010-01-16 | C00002    | A008       |
+---------+---------+---------+------------+-----------+------------+
1 row in set (0.001 sec)

MariaDB [students]> DELETE FROM NewOrders WHERE Ord_Amt-Adv_Amt<(SELECT MAX(Adv_Amt) FROM Orders);
Query OK, 1 row affected (0.033 sec)

MariaDB [students]> select * from NewOrders;
Empty set (0.001 sec)
```

Ex) A High School -> Creating the Tables

```
MariaDB [(none)]> create database students;
Query OK, 1 row affected (0.298 sec)

MariaDB [(none)]> use students;
Database changed
MariaDB [students]> create table Students(ID int,Name char(20),Class_ID int,GPA float);
Query OK, 0 rows affected (1.621 sec)
```

```
MariaDB [students]> create table Teachers(ID int,Name char(20),Subject char(10),Class_ID int,Salary float);
Query OK, 0 rows affected (0.195 sec)

MariaDB [students]> create table Classes(ID int,Grade int,Teacher_ID int,Num_of_students int);
Query OK, 0 rows affected (0.275 sec)

MariaDB [students]> show tables;
+--------------------+
| Tables_in_students |
+--------------------+
| classes            |
| students           |
| teachers           |
+--------------------+
3 rows in set (0.001 sec)
```

Boiler Code – Inserting the Data

```
MariaDB [students]> insert into Students values(1,"Jack Black",3,3.45);
Query OK, 1 row affected (0.722 sec)

MariaDB [students]> insert into Students values(2,"Daniel White",1,3.15);
Query OK, 1 row affected (0.138 sec)

MariaDB [students]> insert into Students values(3,"Kathrine Star",1,3.85);
Query OK, 1 row affected (0.287 sec)

MariaDB [students]> insert into Students values(4,"Helen Bright",2,3.10);
Query OK, 1 row affected (0.028 sec)

MariaDB [students]> insert into Students values(5,"Steve May",2,2.40);
Query OK, 1 row affected (0.063 sec)

MariaDB [students]> insert into Teachers values(1,"Elizabeth Grey","History",3,2500);
Query OK, 1 row affected (0.402 sec)

MariaDB [students]> insert into Teachers values(2,"Robert Sun","Literature",null,2000);
Query OK, 1 row affected (0.143 sec)

MariaDB [students]> insert into Teachers values(3,"John Churchill","English",1,2350);
Query OK, 1 row affected (0.206 sec)

MariaDB [students]> insert into Teachers values(4,"Sara Parker","Math",2,3000);
Query OK, 1 row affected (0.037 sec)

MariaDB [students]> insert into Classes values(1,10,3,21);
Query OK, 1 row affected (0.131 sec)

MariaDB [students]> insert into Classes values(2,11,4,25);
Query OK, 1 row affected (0.036 sec)

MariaDB [students]> insert into Classes values(3,12,1,28);
Query OK, 1 row affected (0.036 sec)
```

Boiler Code – Tables

```
MariaDB [students]> select * from Students;
+------+---------------+----------+------+
| ID   | Name          | Class_ID | GPA  |
+------+---------------+----------+------+
|    1 | Jack Black    |        3 | 3.45 |
|    2 | Daniel White  |        1 | 3.15 |
|    3 | Kathrine Star |        1 | 3.85 |
|    4 | Helen Bright  |        2 |  3.1 |
|    5 | Steve May     |        2 |  2.4 |
+------+---------------+----------+------+
5 rows in set (0.098 sec)

MariaDB [students]> select * from Teachers;
+------+---------------+------------+----------+--------+
| ID   | Name          | Subject    | Class_ID | Salary |
+------+---------------+------------+----------+--------+
|    1 | Elizabeth Grey | History    |        3 |   2500 |
|    2 | Robert Sun     | Literature |     NULL |   2000 |
|    3 | John Churchill | English    |        1 |   2350 |
|    4 | Sara Parker    | Math       |        2 |   3000 |
+------+---------------+------------+----------+--------+
4 rows in set (0.000 sec)

MariaDB [students]> select * from Classes;
+------+-------+------------+-----------------+
| ID   | Grade | Teacher_ID | Num_of_students |
+------+-------+------------+-----------------+
|    1 |    10 |          3 |              21 |
|    2 |    11 |          4 |              25 |
|    3 |    12 |          1 |              28 |
+------+-------+------------+-----------------+
3 rows in set (0.000 sec)
```

Select Subqueries – Basic Example

```
MariaDB [students]> select AVG(GPA) FROM Students;
+--------------------+
| AVG(GPA)           |
+--------------------+
| 3.1900000095367433 |
+--------------------+
1 row in set (0.097 sec)

MariaDB [students]> select * from Students WHERE GPA>3.19;
+------+---------------+----------+------+
| ID   | Name          | Class_ID | GPA  |
+------+---------------+----------+------+
|    1 | Jack Black    |        3 | 3.45 |
|    3 | Kathrine Star |        1 | 3.85 |
+------+---------------+----------+------+
2 rows in set (0.034 sec)

MariaDB [students]> select * from Students WHERE GPA>(select AVG(GPA) FROM Students);
+------+---------------+----------+------+
| ID   | Name          | Class_ID | GPA  |
+------+---------------+----------+------+
|    1 | Jack Black    |        3 | 3.45 |
|    3 | Kathrine Star |        1 | 3.85 |
+------+---------------+----------+------+
2 rows in set (0.005 sec)
```

Examples of Queries on the Table

```
MariaDB [students]> Select AVG(Num_of_students) FROM Classes WHERE Teacher_ID IN(select ID FROM Teachers WHERE Subject = "English" OR Subject="History");
+----------------------+
| AVG(Num_of_students) |
+----------------------+
|              24.5000 |
+----------------------+
1 row in set (0.215 sec)

MariaDB [students]> select * from Students WHERE Class_ID = (select ID from Classes where Num_of_Students = (select MAX(Num_of_students) from Classes));
+------+------------+----------+------+
| ID   | Name       | Class_ID | GPA  |
+------+------------+----------+------+
|    1 | Jack Black |        3 | 3.45 |
+------+------------+----------+------+
1 row in set (0.123 sec)
```

```
MariaDB [students]> SELECT Subject, MAX(salary_by_subject.avg_salary) AS max_salary FROM(SELECT Subject, AVG(Salary) AS avg_salary FROM Teachers GROUP BY Subject) salar
y_by_subject;
+---------+------------+
| Subject | max_salary |
+---------+------------+
| English |       3000 |
+---------+------------+
1 row in set (0.148 sec)
```