DBMS - Experiment 5

Name: Ayush Jain

Sapid: 60004200132

Div.: B1

Branch: Computer Engineering

AIM:

Perform Simple queries, string manipulation operations.

Theory:

The most commonly used SQL command is SELECT statement. SQL SELECT statement is used to query or retrieve data from a table in the database. A query may retrieve information from specified columns or from all of the columns in the table. To create a simple SQL SELECT Statement, you must specify the column(s) name and the table name. The whole query is called SQL SELECT Statement.

Syntax:

SELECT [DISTINCT | ALL] { * | [fieldExpression [AS newName]} FROM tableName [alias] [WHERE condition] [GROUP BY fieldName(s)] [HAVING condition] ORDER BY fieldName(s)

- **SELECT** is the SQL keyword that lets the database know that you want to retrive data.
- **[DISTINCT | ALL]** are optional keywords that can be used to fine tune the results returned from the SQL SELECT statement. If nothing is specified then ALL is assumed as the default.

- **{ * | [fieldExpression [AS newname]}** at least one part must be specified, "*" selected all the fields from the specified table name, fieldExpression performs some computations on the specified fields such as adding numbers or putting together two string fields into one.
- **FROM** tableName is mandatory and must contain at least one table, multiple tables must be separated using commas or joined using the JOIN keyword.
- **WHERE** condition is optional, it can be used to specify criteria in the result set returned from the query.
- **GROUP BY** is used to put together records that have the same field values.
- **HAVING** condition is used to specify criteria when working using the GROUP BY keyword.
- **ORDER BY** is used to specify the sort order of the result set.

Queries:

1. **Select all column from a table**

Select * from tableName;

2. **Select specific column of list from a table**

Select col1, col2 from tablename;

3. **Select …. where clause with various operators (<, >, <=, > =, IN, NOT IN, BETWEEN..AND, NOT BETWEEN… AND)**

Select col1 from tablename where col2>value1;
Select * from tablename where col2 in (value1, value2, value3)
Select * from tablename where col2 between value1 and value2
Select * from tablename where col2 NOT between value1 and value2

4. **Select …. where clause with multiple conditions**

Select * from tablename where col2=Value1 and/or col3=value2

5. **Select …. where clause with string matching**

- Starts with any character
- ends with any character
- have a specific substring
- character at specific position
- starts with a specific character and having specifically n no of characters
- starts and ends with different character

6. **Query your database by applying aggregate function**

MIN, MAX, COUNT, AVG, SUM (create alias also)

7. **Query your database by applying group by clause using one column and multiple column**

8. **Query your database by applying order by clause using one column and multiple column**

9. **Query your database by applying group by, order by and having clause simultaneously**

10. **Select clause with various date functions**

CURDATE(), CURRENT_TIME(), CURRENT_TIMESTAMP(), DATE(), DATEDIFF(), DAY(), DAYNAME(), EXTRACT(), MINUTE(), MONTH(), WEEK(), NOW(), YEAR()

11. **Perform set operation on multiple select queries (UNION)**

Performed Queries and screenshots:

1. Retrieve all attribute values from artist.

SELECT * FROM artist;

```
mysql> SELECT * FROM artist;
+-----------+------------------+--------+
| artist_id | artist_name      | rating |
+-----------+------------------+--------+
|         3 | Justin Bieber    |      8 |
|         4 | Ed Sheeran       |      9 |
|         5 | Doja Cat         |      7 |
|         7 | Juice WRLD       |      8 |
|         9 | Adele            |     10 |
|        10 | Lil Nas X        |      9 |
|        11 | Mythpat          |      9 |
|        12 | Triggered Insaan |      8 |
+-----------+------------------+--------+
```

2. Retrieve the cross product of artist and song.

SELECT * from artist, song;

```
mysql> select * from artist, song;
```

artist_id	artist_name	rating	song_id	song_title	length	album_id	genre	artist_id
3	Justin Bieber	8	21	Industry Baby	00:03:32	NULL	Pop rap	10
3	Justin Bieber	8	19	Perfect	00:04:23	4	Pop	4
3	Justin Bieber	8	18	Galway Girl	00:02:50	4	Pop	4
3	Justin Bieber	8	16	Smile	00:03:16	3	Hip hop	7
3	Justin Bieber	8	7	Hello	00:04:55	2	soul	9
3	Justin Bieber	8	5	Peaches	00:03:18	1	Pop	3
4	Ed Sheeran	9	21	Industry Baby	00:03:32	NULL	Pop rap	10
4	Ed Sheeran	9	19	Perfect	00:04:23	4	Pop	4
4	Ed Sheeran	9	18	Galway Girl	00:02:50	4	Pop	4
4	Ed Sheeran	9	16	Smile	00:03:16	3	Hip hop	7
4	Ed Sheeran	9	7	Hello	00:04:55	2	soul	9
4	Ed Sheeran	9	5	Peaches	00:03:18	1	Pop	3
5	Doja Cat	7	21	Industry Baby	00:03:32	NULL	Pop rap	10
5	Doja Cat	7	19	Perfect	00:04:23	4	Pop	4
5	Doja Cat	7	18	Galway Girl	00:02:50	4	Pop	4
5	Doja Cat	7	16	Smile	00:03:16	3	Hip hop	7
5	Doja Cat	7	7	Hello	00:04:55	2	soul	9
5	Doja Cat	7	5	Peaches	00:03:18	1	Pop	3
7	Juice WRLD	8	21	Industry Baby	00:03:32	NULL	Pop rap	10
7	Juice WRLD	8	19	Perfect	00:04:23	4	Pop	4
7	Juice WRLD	8	18	Galway Girl	00:02:50	4	Pop	4
7	Juice WRLD	8	16	Smile	00:03:16	3	Hip hop	7
7	Juice WRLD	8	7	Hello	00:04:55	2	soul	9
7	Juice WRLD	8	5	Peaches	00:03:18	1	Pop	3
9	Adele	10	21	Industry Baby	00:03:32	NULL	Pop rap	10
9	Adele	10	19	Perfect	00:04:23	4	Pop	4
9	Adele	10	18	Galway Girl	00:02:50	4	Pop	4
9	Adele	10	16	Smile	00:03:16	3	Hip hop	7
9	Adele	10	7	Hello	00:04:55	2	soul	9
9	Adele	10	5	Peaches	00:03:18	1	Pop	3
10	Lil Nas X	9	21	Industry Baby	00:03:32	NULL	Pop rap	10
10	Lil Nas X	9	19	Perfect	00:04:23	4	Pop	4
10	Lil Nas X	9	18	Galway Girl	00:02:50	4	Pop	4
10	Lil Nas X	9	16	Smile	00:03:16	3	Hip hop	7
10	Lil Nas X	9	7	Hello	00:04:55	2	soul	9
10	Lil Nas X	9	5	Peaches	00:03:18	1	Pop	3
11	Mythpat	9	21	Industry Baby	00:03:32	NULL	Pop rap	10
11	Mythpat	9	19	Perfect	00:04:23	4	Pop	4
11	Mythpat	9	18	Galway Girl	00:02:50	4	Pop	4
11	Mythpat	9	16	Smile	00:03:16	3	Hip hop	7
11	Mythpat	9	7	Hello	00:04:55	2	soul	9
11	Mythpat	9	5	Peaches	00:03:18	1	Pop	3
12	Triggered Insaan	8	21	Industry Baby	00:03:32	NULL	Pop rap	10
12	Triggered Insaan	8	19	Perfect	00:04:23	4	Pop	4
12	Triggered Insaan	8	18	Galway Girl	00:02:50	4	Pop	4
12	Triggered Insaan	8	16	Smile	00:03:16	3	Hip hop	7
12	Triggered Insaan	8	7	Hello	00:04:55	2	soul	9
12	Triggered Insaan	8	5	Peaches	00:03:18	1	Pop	3

3. Retrieve all song's id, title and length of Ed Sheeran which has artist_id=4.

SELECT song_id, song_title, length
FROM song
WHERE artist_id=4;

```
mysql> SELECT song_id, song_title, length
    -> FROM song
    -> WHERE artist_id=4;
+---------+-------------+----------+
| song_id | song_title  | length   |
+---------+-------------+----------+
|      18 | Galway Girl | 00:02:50 |
|      19 | Perfect     | 00:04:23 |
+---------+-------------+----------+
```

4. Retrieve all song's id, title and length of Ed Sheeran which has artist_id=4 and length of 2mins 50s.

SELECT song_id, song_title, length
FROM song
WHERE artist_id=4 and length="0:2:50";

```
mysql> SELECT song_id, song_title, length
    -> FROM song
    -> WHERE artist_id=4 and length="0:2:50";
+---------+-------------+----------+
| song_id | song_title  | length   |
+---------+-------------+----------+
|      18 | Galway Girl | 00:02:50 |
+---------+-------------+----------+
```

5. Retrieve all song's id, title and length of Ed Sheeran which has artist_id=4 and length of 2mins 50s.

SELECT song_id, song_title, length
FROM song as s, artist as a
WHERE s.artist_id=a.artist_id and length="0:2:50";

```
mysql> SELECT song_id, song_title, length
    -> FROM song as s, artist as a
    -> WHERE s.artist_id=a.artist_id and length="0:2:50";
+---------+-------------+----------+
| song_id | song_title  | length   |
+---------+-------------+----------+
|      18 | Galway Girl | 00:02:50 |
+---------+-------------+----------+
```

6. Retrieve distinct podcast type from podcast.

SELECT distinct podcast_type FROM podcast;

```
mysql> SELECT distinct podcast_type FROM podcast;
+--------------+
| podcast_type |
+--------------+
| Intro        |
| Story        |
| Knowledge    |
| Fun          |
+--------------+
4 rows in set (0.00 sec)
```

7. Retrieve all podcast type from podcast.

SELECT all podcast_type FROM podcast;

```
mysql> SELECT all podcast_type FROM podcast;
+--------------+
| podcast_type |
+--------------+
| Intro        |
| Story        |
| Story        |
| Knowledge    |
| Intro        |
| Fun          |
| Fun          |
| Fun          |
| Fun          |
| Fun          |
+--------------+
```

8. Retrieve all users whose email account is of gmail.

SELECT username
FROM user
WHERE email like "%@gmail%";

9. Retrieve all users whose phone numbers start with 9.

SELECT username
FROM user
WHERE mobileNo like '9_____';

10. Retrieve all the song_titles as song_name for all songs which belong to album id. 4.

SELECT song_title as song_name
FROM song
WHERE album_id=4;

```
mysql> SELECT song_title as song_name
    -> FROM song
    -> WHERE album_id=4;
+-------------+
| song_name   |
+-------------+
| Galway Girl |
| Perfect     |
+-------------+
```

11. Retrieve all artists whose rating is between 6 and 9.

SELECT * FROM artist
WHERE rating between 6 and 9;

```
mysql> SELECT * FROM artist
    -> WHERE rating between 6 and 9;
+-----------+------------------+--------+
| artist_id | artist_name      | rating |
+-----------+------------------+--------+
|         4 | Ed Sheeran       |      9 |
|         5 | Doja Cat         |      7 |
|         7 | Juice WRLD       |      8 |
|        10 | Lil Nas X        |      6 |
|        11 | Mythpat          |      9 |
|        12 | Triggered Insaan |      8 |
+-----------+------------------+--------+
```

12. Retrieve artist's name, rating and their corresponding song's name, length, genre ordered alphabetically in ascending order by song's name and descending order of artist's rating.

SELECT artist_name, rating, song_title, length, genre
FROM artist, song
WHERE artist.artist_id=song.artist_id
ORDER BY rating desc, song_title asc;

```
mysql> SELECT artist_name, rating, song_title, length, genre
    -> FROM artist, song
    -> WHERE artist.artist_id=song.artist_id
    -> ORDER BY rating desc, song_title asc;
+---------------+--------+---------------+----------+---------+
| artist_name   | rating | song_title    | length   | genre   |
+---------------+--------+---------------+----------+---------+
| Adele         |     10 | Hello         | 00:04:55 | soul    |
| Ed Sheeran    |      9 | Galway Girl   | 00:02:50 | Pop     |
| Ed Sheeran    |      9 | Perfect       | 00:04:23 | Pop     |
| Juice WRLD    |      8 | Smile         | 00:03:16 | Hip hop |
| Lil Nas X     |      6 | Industry Baby | 00:03:32 | Pop rap |
| Justin Bieber |      5 | Peaches       | 00:03:18 | Pop     |
+---------------+--------+---------------+----------+---------+
```

13. Retrieve names of songs which don't have an album.

SELECT song_title
FROM song
WHERE album_id is NULL;

```
mysql> SELECT song_title
    -> FROM song
    -> WHERE album_id is NULL;
+---------------+
| song_title    |
+---------------+
| Love nwantiti |
| Industry Baby |
+---------------+
```

14. Retrieve maximum length of a song.

SELECT song_title, length
FROM song
WHERE length = (SELECT MAX(length) FROM song);

```
mysql> SELECT song_title, length
    -> FROM song
    -> WHERE length = (SELECT MAX(length) FROM song);
+------------+----------+
| song_title | length   |
+------------+----------+
| Hello      | 00:04:55 |
+------------+----------+
```

15. Retrieve minimum length of a song.

SELECT song_title, length
FROM song
WHERE length = (SELECT MIN(length) FROM song);

```
mysql> SELECT song_title, length
    -> FROM song
    -> WHERE length = (SELECT MIN(length) FROM song);
+-------------+----------+
| song_title  | length   |
+-------------+----------+
| Galway Girl | 00:02:50 |
+-------------+----------+
```

16. Retrieve average length of a song.

SELECT AVG(rating)
FROM artist;

```
mysql> SELECT AVG(rating)
    -> FROM artist;
+--------------+
| AVG(rating) |
+--------------+
|       7.7500 |
+--------------+
```

17. Retrieve total number of songs.

SELECT count(*) as numberOfSongs
FROM song;

```
mysql> SELECT count(*) as numberOfSongs
    -> FROM song;
+---------------+
| numberOfSongs |
+---------------+
|             7 |
+---------------+
```

18. Retrieve album title, song's title, length, genre, and the corresponding album's average rating as per the artists which is greater than 8.

```
SELECT album_title, song_title, length, genre, AVG(rating)
FROM album, song, artist
WHERE song.album_id=album.album_id and song.artist_id=artist.artist_id
GROUP BY album_title
HAVING AVG(rating)>8;
```

```
mysql> SELECT album_title, song_title, length, genre, AVG(rating)
    -> FROM album, song, artist
    -> WHERE song.album_id=album.album_id and song.artist_id=artist.artist_id
    -> GROUP BY album_title
    -> HAVING AVG(rating)>8;
+-------------+-------------+----------+-------+-------------+
| album_title | song_title  | length   | genre | AVG(rating) |
+-------------+-------------+----------+-------+-------------+
| 25          | Hello       | 00:04:55 | soul  |     10.0000 |
| Divide      | Galway Girl | 00:02:50 | Pop   |      9.0000 |
+-------------+-------------+----------+-------+-------------+
```

19. Retrieve total rating of all the artists.

```
SELECT sum(rating) as totalRating
FROM artist;
```

```
mysql> SELECT sum(rating) as totalRating
    -> FROM artist;
+-------------+
| totalRating |
+-------------+
|          62 |
+-------------+
```

Conclusion:

Select queries with various clauses like group by, order by and aggregate functions is implemented in mysql.