Data And Applications

Homework - 3

Abhinav Reddy Boddu

2021101034

Rohit Gowlapalli

2021101113

Gnana Prakash Punnavajhala

2021111027

Note: All the queries are single line (no nested queries) commands written in multiple lines just for readability purposes.

Query 1:

- Which Track Has Longest Length?

```sql
SELECT *
FROM Track
ORDER BY Milliseconds
DESC LIMIT 1;
```

(or)

```sql
SELECT *
FROM Track
WHERE Milliseconds = (
    SELECT MAX(Milliseconds)
    FROM Track
);
```

(or)

```sql
SELECT T1.*
FROM Track AS T1
LEFT JOIN Track AS T2
```

```
            ON T1.Milliseconds < T2.Milliseconds
    WHERE T2.Milliseconds IS NULL;
```

- Approach 1:

    - Select all columns from `Track`
    - Arrange them in descending order of `Milliseconds`
    - Limit the output to 1 row, i.e., the row having the highest value of `Milliseconds`

- Approach 2:

    - Select maximum `Milliseconds` from `Track`
    - Print all rows of the table `Track` where `Milliseconds` is the maximum

- Approach 3:

    - Left join the columns of the `Track` table to itself
    - The right tuple denotes the tuples which have the value of the `Milliseconds` column greater than the left tuple
    - The right tuple is `NULL` if there are no tuples on the left satisfying the condition
    - These left tuples have the maximum value of the `Milliseconds` column and thus, these tuples are selected to be displayed in the output

```
mysql> SELECT * FROM Track ORDER BY Milliseconds DESC LIMIT 1;
+---------+----------------+---------+--------------+----------+---------+----------+
| TrackID | Name           | AlbumID | Milliseconds | Bytes    | GenreID | ArtistID |
+---------+----------------+---------+--------------+----------+---------+----------+
|      15 | Under Pressure |       6 |       559000 | 55900000 |       4 |        2 |
+---------+----------------+---------+--------------+----------+---------+----------+
1 row in set (0.01 sec)

mysql> SELECT * FROM Track WHERE Milliseconds = (SELECT MAX(Milliseconds) FROM Track);
+---------+----------------+---------+--------------+----------+---------+----------+
| TrackID | Name           | AlbumID | Milliseconds | Bytes    | GenreID | ArtistID |
+---------+----------------+---------+--------------+----------+---------+----------+
|      15 | Under Pressure |       6 |       559000 | 55900000 |       4 |        2 |
+---------+----------------+---------+--------------+----------+---------+----------+
1 row in set (0.01 sec)

mysql> SELECT T1.* FROM Track AS T1 LEFT JOIN Track AS T2 ON T1.Milliseconds < T2.Milliseconds WHERE T2.Milliseconds IS NULL;
+---------+----------------+---------+--------------+----------+---------+----------+
| TrackID | Name           | AlbumID | Milliseconds | Bytes    | GenreID | ArtistID |
+---------+----------------+---------+--------------+----------+---------+----------+
|      15 | Under Pressure |       6 |       559000 | 55900000 |       4 |        2 |
+---------+----------------+---------+--------------+----------+---------+----------+
1 row in set (0.00 sec)
```

Query 2:

- How many audio tracks were listened to by the people of "Indian" nationality?

```
SELECT COUNT(DISTINCT TrackID) AS Count
FROM Listener AS A,
        ListeningTo AS B
WHERE A.Nationality='Indian'
        AND A.ListenerID=B.ListenerID;
```

- Approach:

    - Select only Nationality = "Indian" From Listener table
    - Select all the appropriate rows in the ListeningTo table with ListenerID as a foreign key
    - Count the number of distinct tracks

```
mysql> SELECT COUNT(DISTINCT TrackID) AS Count FROM Listener AS A, ListeningTo AS B WHERE Nationality='Indian' AND A.ListenerID=B.ListenerID;
+-------+
| Count |
+-------+
|    12 |
+-------+
1 row in set (0.01 sec)
```

Query 3:

- Which album takes up the maximum space?

```sql
SELECT A.*
FROM Album AS A,
        Track AS T
WHERE A.AlbumID=T.AlbumID
GROUP BY A.AlbumID
ORDER BY SUM(Bytes) DESC
LIMIT 1;
```

- Approach:

    - Select tables Album and Track with AlbumID as foreign key.
    - Group by AlbumID and calculate the sum of Bytes of a given album
    - Order the selected rows in the descending order of sum of bytes.
    - Limit the selected rows to be displayed to 1 to get the album of highest bytes, i.e., maximum space

```
mysql> SELECT A.* FROM Album AS A, Track AS T WHERE A.AlbumID = T.AlbumID GROUP BY A.AlbumID ORDER BY SUM(Bytes) DESC LIMIT 1;
+---------+----------------+
| AlbumID | Name           |
+---------+----------------+
|       6 | Under Pressure |
+---------+----------------+
1 row in set (0.01 sec)
```

Query 4:

- Which nationality listens to music the least?

```sql
SELECT L.Nationality
FROM Listener AS L,
        ListeningTo AS LT
WHERE L.ListenerID=LT.ListenerID
GROUP BY L.Nationality
```

```
    ORDER BY SUM(LT.Milliseconds) ASC
    LIMIT 1;
```

- Approach:

  - Select `Listener` and `ListeningTo` using `ListenerID` as Foreign key
  - Group by `Nationality`
  - Order by the sum of `Time` for a given nationality
  - Limit the selected rows to be displayed to 1 to get the nationality having lowest listening time.

```
mysql> SELECT L.Nationality FROM Listener AS L, ListeningTo AS LT WHERE L.ListenerID=LT.ListenerID GROUP BY L.Nationality ORDER BY SUM(LT.Milliseconds) ASC LIMIT 1;
+-------------+
| Nationality |
+-------------+
| Indian      |
+-------------+
1 row in set (0.00 sec)
```

Query 5:

- Which genre is listened to by the most people among "Americans"?

```
SELECT G.*
FROM Genre AS G,
        Track AS T,
        ListeningTo AS LT,
        Listener AS L
WHERE L.Nationality='American'
        AND L.ListenerID=LT.ListenerID
        AND LT.TrackID=T.TrackID
        AND T.GenreID=G.GenreID
GROUP BY G.GenreID
ORDER BY COUNT(*) DESC
LIMIT 1;
```

- Approach:

  - Select all columns in `Genre` from `Genre`, `Track`, `ListeningTo` and `Listener` tables using `Nationality`, `ListenerID`, `TrackID`, `GenreID` as foreign keys.
  - Group them using `GenreID`
  - Order them by the number of times each genre is appearing in the table from highest to lowest, i.e., descending order
  - Limit the selected rows to be displayed to 1 to get the genre that appeared the most number of times

```
mysql> SELECT G.* FROM Genre AS G, Track AS T, ListeningTo AS LT, Listener AS L WHERE L.Nationality='American' AND L.ListenerID=LT.ListenerID AND LT.TrackID=T.TrackID A
ND T.GenreID=G.GenreID GROUP BY G.GenreID ORDER BY COUNT(*) DESC LIMIT 1;
+---------+------+
| GenreID | Name |
+---------+------+
|       4 | Rap  |
+---------+------+
1 row in set (0.00 sec)
```

Query 6:

- Which artist did not make any albums at all?

```sql
SELECT A.*
FROM Artist AS A
LEFT JOIN Track AS T
        ON A.ArtistID=T.ArtistID
GROUP BY A.ArtistID
        HAVING COUNT(T.ArtistID) = 0;
```

- Approach:

    - Select all columns in Artist from Artist and Track using ArtistID as foreign key
      (mandating every artist to appear using left join).
    - Group by ArtistID
    - Print all artists who have 0 rows in the Track table, which can be obtained by using the
      condition COUNT(T.ArtistID) = 0

```
mysql> SELECT A.Name FROM Artist AS A LEFT JOIN Track AS T ON A.ArtistID=T.ArtistID GROUP BY A.Name HAVING COUNT(T.ArtistID) = 0;
+----------------+
| Name           |
+----------------+
| Vennu Mallesh  |
+----------------+
1 row in set (0.00 sec)
```

Query 7:

- Which artists did not record any tracks of the "Pop" Genre type?

```sql
SELECT A.Name
FROM Artist AS A
LEFT JOIN Genre AS G LEFT JOIN Track AS T
        ON T.GenreID=G.GenreID AND G.Name='Pop' ON
A.ArtistID=T.ArtistID
GROUP BY A.ArtistID
        HAVING COUNT(T.ArtistID) = 0;
```

- Approach:

    - Left Join the table Genre to Track where GenreID in Genre and Track tables are same and
      the Genre is Pop after which left join the Artist table to this table where ArtistID in both
      the tables are same
    - Group by ArtistID and select the artists which did not appear in the first table

```
mysql> SELECT A.Name FROM Artist AS A LEFT JOIN Genre AS G LEFT JOIN Track AS T ON T.GenreID=G.GenreID AND G.Name='Pop' ON A.ArtistID=T.ArtistID GROUP BY A.Name HAVING COUNT(T
.ArtistID) = 0;
+---------------+
| Name          |
+---------------+
| Logic         |
| Vennu Mallesh |
+---------------+
2 rows in set (0.00 sec)
```