|  |  |
| --- | --- |
| **Ex No:** | **ADVANCED SQL COMMANDS** |
| **Date** | **23-1-24** |

**20CS2016L – Database Systems Lab URK22AI1023**

**Aim:**

To execute the given commands making use of aggregate functions, group by clause and order by clause.

# Description:

**Aggregate Functions:**

SQL aggregate functions operate on the multiset of values of a column of a relation, and return a value

The various aggregate functions are:

* **Sum**: - returns the sum of the values. Eg: select sum(sal) from emp;
* **Avg**:- returns the average of the values. Eg: select avg(sal) from emp;
* **Count**:- returns the number of elements in the collection. Eg: select count(\*) from emp;
* **Min**:- returns the minimum value in a collection. Eg: select min(sal) from emp;
* **Max**:- returns the maximum value in a collection. Eg: select max(sal) from emp;

The input to sum and average must be a collection of numbers, but the other operators can operate on collections of non-numeric data types, such as strings as well. The average function will return the average of the given tuple. The aggregation function count is used frequently to count the number of tuples in relation.

# Distinct Keyword

To eliminate the duplicates, the keyword *distinct* is used in the aggregation expression. SQL does not allow the use of keyword distinct with count (\*) to count the number of records in a relation. It is allowed to use distinct with max and min functions, even though the result does not change.

Eg: Select count (distinct job);

# GROUP BY Clause

To apply aggregate function to a group of sets of tuples. The attributes given in the group by clause are used to form groups. Tuples with some value on all attributes in the group by clause are placed in one group.

Eg: Select branch\_name,avg(bal) from account group by branch\_name;

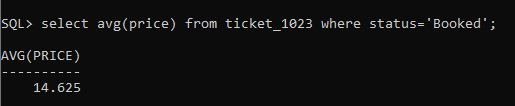
# ORDER BY Clause

This clause causes the tuples in the result of a query to appear in sorted order. We specify asc for ascending order and desc for descending order.

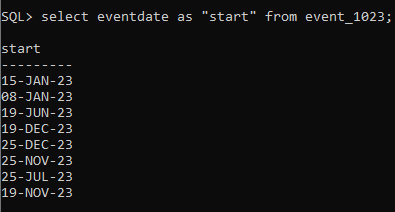
Eg: Select \* from loan order by amount desc;

# Advanced SQL Queries:

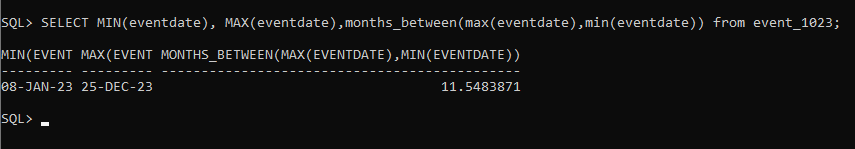
1. Find the average price of confirmed tickets



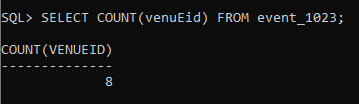
2. Display the starting date as “Start” of all the events from events table.



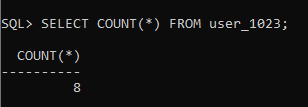
3. Find the Minimum date, maximum date of all the events, and the number of months in between the min and max dates



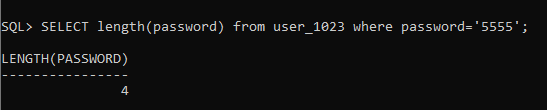
4. Find the total number of venues used to conduct the events



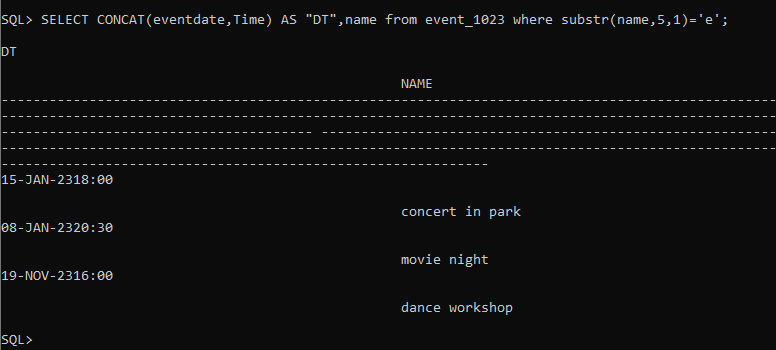
5. Find the number of users in User table



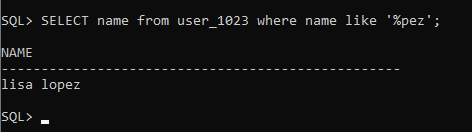
6. Find the length of password of User = p@ssw0rd from User table



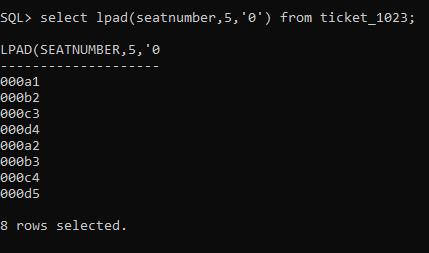
7. Concatenate the data and time of Event table as “Date-Time” and display Date-Time and the name of the event if the 5th character of event name is ‘e’



8. Find the user names whose name ends with “pez” from user table



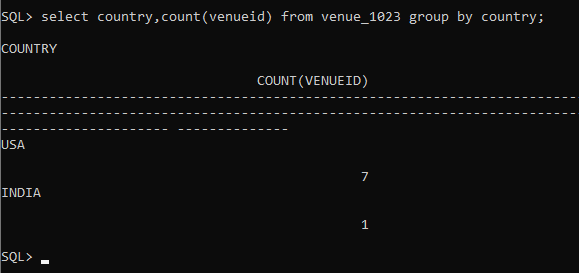
9. Left pad the seat number of Ticket table with “000”



10. Display the event details conducted at the same Venue\_ID



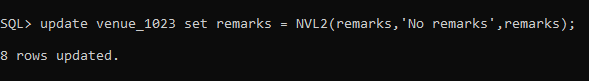
11. Find out the number of venues in each country from Venue table



12. Add a column named “Remarks” in Venue table. Fill the remarks column with “No Remarks” values using NVL2 command. Print VenueID, Country, and Remarks column.

A black background with white text

Description automatically generated

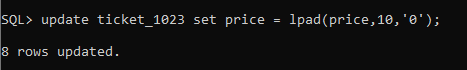


13. Use round and trunc functions to round off and truncate the value 25.235789 to 2 decimal positions using dual table.

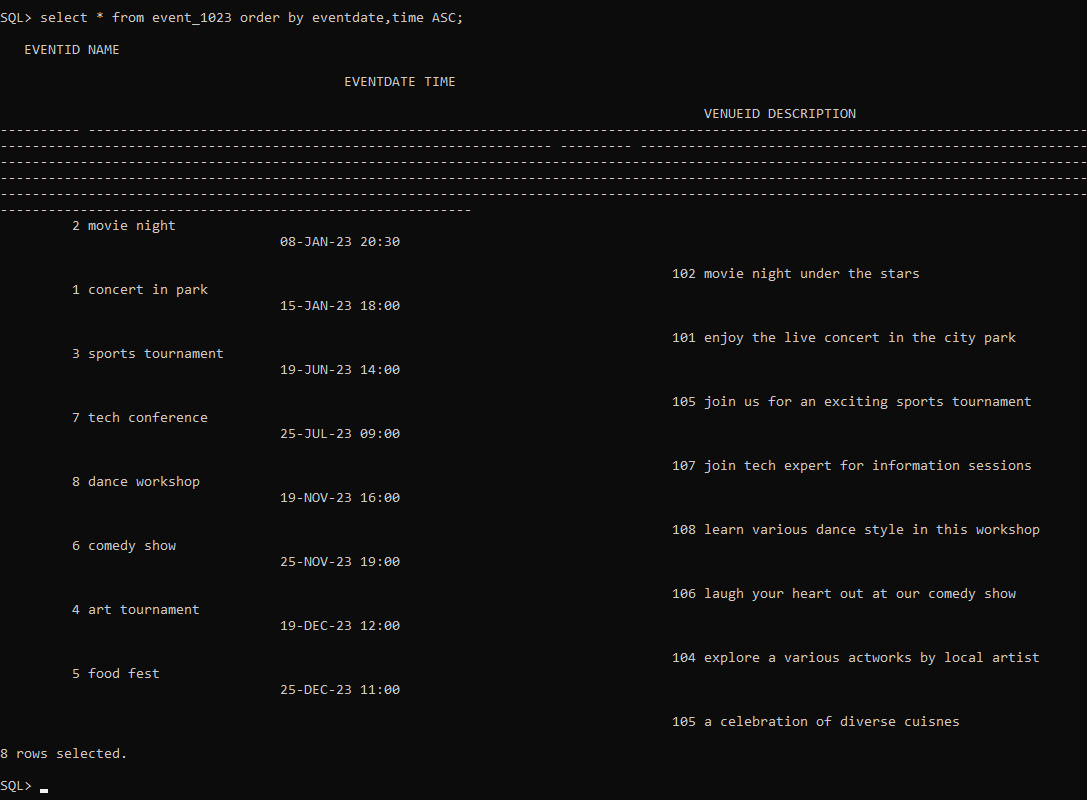
A black background with white text

Description automatically generated

14. Prefix price column with a value of 0 in ticket table to make the length of price =10 digits.



15. Retrieve all events ordered by date and time in ascending order.



**Result:**

The given commands making use of aggregate functions, group by clause and order by clause are executed successfully.