SQL Queries

Malvica Lewis
Technological University of the Shannon
Masters in Data Analytics
Student ID: A00303932

21/12/2022

Contents

Private Function Entertainment

The data includes details of jockeys and the events where the jockeys performed. The data is saved in two different tables one being the parent table and the other being the child table.

The parent table is named as **Disc_Jockey** which includes the details of 30 jockeys. The table includes the following attributes:

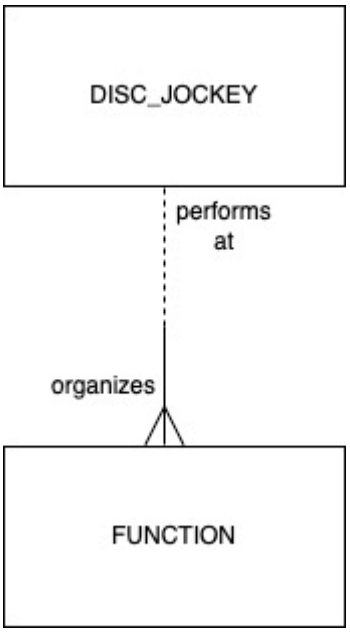
- **Disc_Jockey_Id** – The identification number of the jockey
- **Fname** – The first name of the jockey
- **Sname** – The surname of the jockey
- **Address** – The address of the jockey
- **Town** – The town where the jockey resides
- **County** – The county where the jockey resides
- **Date_Of_Birth** – The date of birth of jockey
- **Email** – The email address of jockey
- **Music_Type** – The genre of music performed by jockey
- **Availability** – The availability of the jockey – Midweeks, All Weeks and Weekends

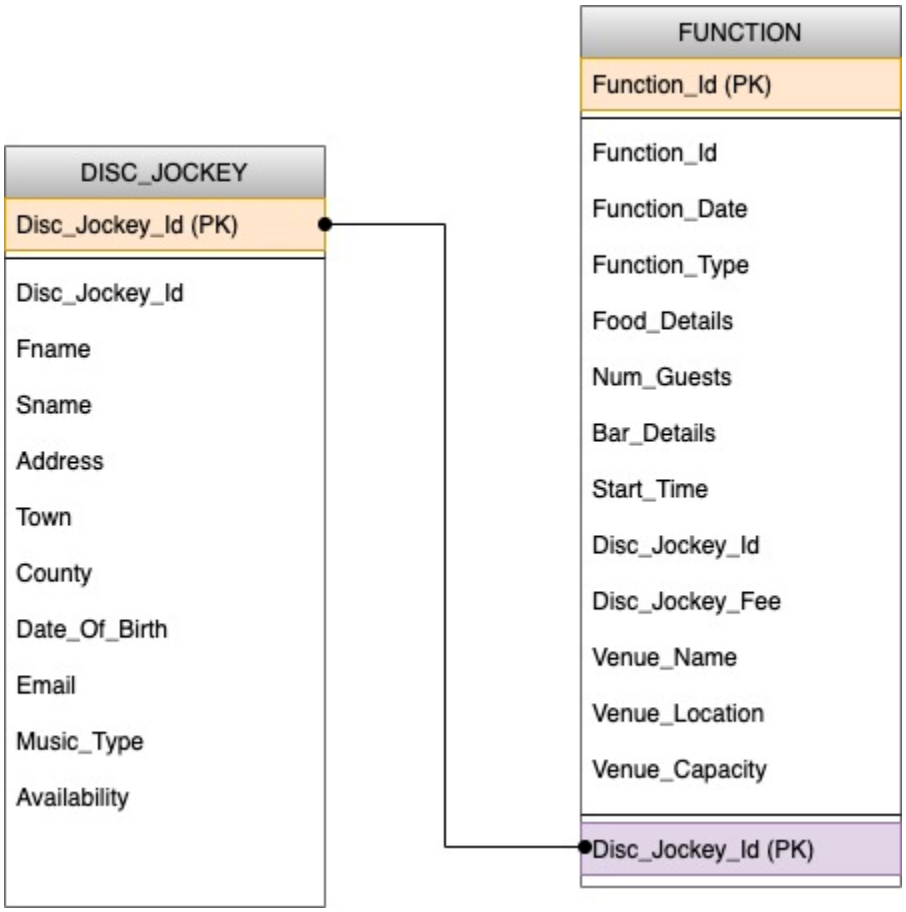
The child table is named as **Function** which includes the details of 70 events where the jockeys performed. The table includes the following attributes:

- **Function_Id** - The identification number of the event
- **Function_Date** - The date when the event was organized
- **Function_Type** – The type of event - Birthday, Retirement, Wedding, Corporate Event
- **Food_Details** - The type of food served during the event - Finger Food, Sit Down Meal, Buffet, Tea and Sandwiches
- **Num_Guests** - The number of guests who attended the event
- **Bar_Details** – The details of the bar where the event was organized - No Alcohol, Free Bar, All Drink Charged to Guest)
- **Start_Time** – The start time of the event

- **Disc_Jockey_Id** – The identification number of the jockey
- **Disc_Jockey_Fee** – The fee charged by the jockey during an event
- **Venue_Name** - The name of the venue where the jockey performed
- **Venue_Location** – The location where the venue is situated
- **Venue_Capacity** – The capacity of the venue to occupy guests during an event

Entity Relationship Diagrams (ERD)





Create Table and Insert Commands

The following is the list of commands to create the Disc_Jockey and Function tables and inserts records to these tables.

```
Set Echo On
Set Termout On
Set Feedback On
Set Verify On
Set Heading On

Set Pagesize 2500
Set Linesize 180

Drop Table      Function;
Drop Table      Disc_Jockey;

Drop Table      Disc_Jockey;
Create Table    Disc_Jockey     (
        Disc_Jockey_Id          Number (3),
        Fname           Varchar2 (10)  Constraint Disc_Jockey_Fname_Nn Not Null,
        Sname           Varchar2 (11)  Constraint Disc_Jockey_Sname_Nn Not Null,
        Address         Varchar2 (29),
        Town            Varchar2 (14),
        County          Varchar2 (9),
        Date_Of_Birth Date,
        Email           Varchar2 (29),
        Music_Type      Varchar2 (12),
        Availability Varchar2 (8),
        Constraint      Disck_Jockey_Disc_Jockey_Id_Pk Primary Key (Disc_Jockey_Id),
        Constraint      Disc_Jockey_Email_Uq Unique (Email),
        Constraint      Disc_Jockey_Music_Type_Ck  Check (Music_Type In ('Country', 'Electro', 'Folk',
                        'Instrumental', 'Jazz', 'Rock', 'Opera', 'Pop', 'Techno')),
        Constraint      Disc_Jockey_Availability_Ck  Check (Availability In ('All Week', 'Midweeks',
                        'Weekends')));

Drop Table      Function;
```

```
Create Table    Function        (
        Function_Id             Number (3),
        Function_Date           Date,
        Function_Type           Varchar2 (15)  Constraint Function_Function_Type_Nn Not Null,
        Food_Details   Varchar2 (13),
        Num_Guests              Number (4),
        Bar_Details             Varchar2 (26),
        Start_Time              Number (2),
        Disc_Jockey_Id          Number(3),
        Disc_Jockey_Fee         Number(3),
        Venue_Name              Varchar2 (13),
        Venue_Location          Varchar2 (11),
        Venue_Capacity          Number (4),
        Constraint      Function_Function_Id_Pk Primary Key (Function_Id),
        Constraint      Function_Disc_Jockey_Id_Fk Foreign Key (Disc_Jockey_Id) References Disc_Jockey
                        (Disc_Jockey_Id),
        Constraint      Function_Function_Type_Ck  Check (Function_Type In ('Birthday', 'Corporate
                        Event', 'Retirement', 'Wedding')),
        Constraint      Function_Food_Details_Ck  Check (Food_Details In ('Buffet', 'Finger Food',
                        'Sandwiches', 'Sit Down Meal', 'Tea')),
        Constraint      Function_Bar_Details_Ck  Check (Bar_Details In ('All Drink Charged to Guest', 'Free
                        Bar', 'No Alcohol')),
        Constraint      Function_Num_Guests_Ck  Check (Num_Guests > 0),
        Constraint      Function_Disc_Jockey_Fee_Ck  Check (Disc_Jockey_Fee > 0));


/* Table Name:Disc_Jockey  */


 /*Disc_Jockey_Id,Fname,Sname,Address,Town,County,Date_Of_Birth,Email,Music_Type,Availability */

Insert Into  Disc_Jockey Values (10,'Maribelle','Bellee','71 Eagan Lane','Strabane','Tyrone','23-Jul-
1992','mbellee0@earthlink.net','Folk','All Week');

Insert Into  Disc_Jockey Values (20,'Aleece','Fulcher','56 Hintze Lane','Bushmills','Antrim','15-Jul-
1986','afulcher1@fotki.com','Instrumental','Midweeks');
```

Insert Into Disc_Jockey Values (30,'Kingsley','Alders','105 Oxford Street','Cahir','Tipperary','11-Nov-1977','kalders2@google.fr','Rock','Midweeks');

Insert Into Disc_Jockey Values (40,'Vina','Carlett','09 Stang Terrace','Mohill','Leitrim','12-Oct-1986','vcarlett3@answers.com','Jazz','Weekends');

Insert Into Disc_Jockey Values (50,'Huntington','Hambridge','6 Luster Court','Monasterivn','Kildare','09-Feb-1997','hhambridge4@dailymail.co.uk','Opera','Midweeks');

Insert Into Disc_Jockey Values (60,'Mattheus','Brahm','3726 Lakewood Gardens Terrace','Tramore','Waterford','05-Jun-1993','mbrahm5@wisc.edu','Opera','All Week');

Insert Into Disc_Jockey Values (70,'Irwinn','Lunny','51 Spohn Junction','Trillick','Fermanagh','30-May-1980','iolunny6@hostgator.com','Country','Midweeks');

Insert Into Disc_Jockey Values (80,'Baxie','Houlson','6167 Doe Crossing Court','Ardee','Louth','08-Oct-1988','bhoulson7@businessinsider.com','Techno','Midweeks');

Insert Into Disc_Jockey Values (90,'Crawford','Aloway','1 Esker Road','Bantry','Cork','20-Aug-1977','caloway8@pinterest.com','Pop','All Week');

Insert Into Disc_Jockey Values (100,'Jojo','Poston','81608 Vahlen Road','Carrrickmore','Tyrone','10-Mar-1977','jposton9@delicious.com','Opera','Weekends');

Insert Into Disc_Jockey Values (110,'Brett','Christensen','51 Towne Plaza','Drumcliff','Sligo','13-Jan-1981','bchristensena@wix.com','Pop','Weekends');

Insert Into Disc_Jockey Values (120,'Shaun','Gormally','22 Golf View Terrace','Greystones','Wicklow','02-Jul-1982','sogormallyb@unicef.org','Jazz','Weekends');

Insert Into Disc_Jockey Values (130,'Charity','Castenda','49238 Merrick Circle','Kanturk','Cork','13-Nov-1984','ccastendac@reuters.com','Folk','Weekends');

Insert Into Disc_Jockey Values (140,'Mellie','Drynan','49910 Holy Cross Parkway','Maguiresbridge','Fermanagh','15-Jan-1992','mdrynand@noaa.gov','Jazz','All Week');

Insert Into Disc_Jockey Values (150,'Emmerich','Plackstone','5 Melrose Circle','New Ross','Wexford','08-Nov-1976','eplackstonee@usa.gov','Opera','Midweeks');

Insert Into Disc_Jockey Values (160,'Druci','Downgate','714 Northfield Junction','Strokestown','Roscommon','07-Dec-1988','ddowngatef@ihg.com','Techno','Weekends');

Insert Into Disc_Jockey Values (170,'Audie','Tunmore','92 Walton Crossing','Ballymoe','Roscommon','27-Jan-1996','atunmoreg@live.com','Jazz','Weekends');

Insert Into Disc_Jockey Values (180,'Kaye','Asgodby','114 Straubel Trail','Downpatrick','Down','13-May-1992','kasgodbyh@tuttocitta.it','Country','Midweeks');

Insert Into Disc_Jockey Values (190,'Domeniga','Loisi','5007 Mcbride Place','Glenties','Donegal','08-Feb-1999','dloisii@usda.gov','Opera','Midweeks');

Insert Into Disc_Jockey Values (200,'Gaby','Bolliver','056 Tennyson Road','Lisnaskea','Fermanagh','21-Jul-1996','gbolliverj@theguardian.com','Techno','Weekends');

Insert Into Disc_Jockey Values (210,'Edithe','Eisenberg','576 Hazelcrest Crossing','Rosslare','Wexford','25-Dec-1990','eeisenbergk@imgur.com','Electro','Weekends');

Insert Into Disc_Jockey Values (220,'Serge','Josefsson','46 Sauthoff Plaza','Cappoquin','Waterford','17-Jan-1981','sjosefssonl@blogspot.com','Instrumental','Midweeks');

Insert Into Disc_Jockey Values (230,'Harald','Pearcey','860 Boyd Way','Carlow','Carlow','22-Jul-2001','hpearceym@dot.gov','Instrumental','Weekends');

Insert Into Disc_Jockey Values (240,'Shannon','Hegley','15 Oriole Crossing','Mountmellick','Laois','12-Oct-1983','shegleyn@businessinsider.com','Pop','All Week');

Insert Into Disc_Jockey Values (250,'Neddie','Packington','50 Saint Paul Pass','Tubbercurry','Sligo','22-Apr-1979','npackingtono@reddit.com','Electro','Midweeks');

Insert Into Disc_Jockey Values (260,'Feliza','Gonning','45 Harbort Circle','Glenties','Donegal','06-Dec-1997','fgonningp@narod.ru','Instrumental','Weekends');

```
Insert Into  Disc_Jockey Values (270,'Chane','Veel','7700 Lighthouse Bay Plaza','Bantry','Cork','24-Sep-1995','cveelq@latimes.com','Instrumental','Midweeks');

Insert Into  Disc_Jockey Values (280,'Bancroft','Bunworth','6 Logan Junction','Swords','Dublin','20-Jan-1999','bbunworthr@chronoengine.com','Instrumental','Midweeks');

Insert Into  Disc_Jockey Values (290,'Kim','Boeck','02396 Emmet Terrace','Athlone','Westmeath','10-Feb-1989','kboecks@techcrunch.com','Folk','Weekends');

Insert Into  Disc_Jockey Values (300,'Lynde','Gillie','6 Nelson Hill','Athlone','Westmeath','15-May-1987','lgilliet@google.co.jp','Rock','Weekends');

commit;


/* Table Name:Function  */


/*Function_Id,Function_Date,Function_Type,Food_Details,Num_Guests,Bar_Details,Start_Time,Disc_Jockey_Id,Disc_Jockey_Fee,Venue_Name,Venue_Location,Venue_Capacity */

Insert Into  Function Values (201,'21-Jul-2020','Corporate Event','Tea',1050,'Free Bar',5,230,325,'The Buzz','Ballybofey',2225);

Insert Into  Function Values (202,'12-Sep-2022','Wedding','Finger Food',1566,'All Drink Charged to Guest',10,20,340,'Tribes','Mountrath',1585);

Insert Into  Function Values (203,'25-Sep-2022','Corporate Event','Sandwiches',1565,'All Drink Charged to Guest',8,140,605,'Caribou','Drogheda',2000);

Insert Into  Function Values (204,'02-Sep-2022','Corporate Event','Finger Food',887,'Free Bar',9,160,220,'Stringfellows','Downpatrick',1600);

Insert Into  Function Values (205,'21-Feb-2020','Birthday','Tea',1687,'Free Bar',16,50,670,'Blazers','Claremorris',2000);
```

Insert Into Function Values (206,'07-Feb-2022','Wedding','Sandwiches',749,'No Alcohol',10,230,75,'Ginos','Ballinrobe',1600);

Insert Into Function Values (207,'25-Feb-2020','Birthday','Buffet',1396,'All Drink Charged to Guest',4,270,50,'Boogie','Ferbane',1400);

Insert Into Function Values (208,'20-Nov-2021','Corporate Event','Tea',1419,'No Alcohol',13,200,605,'Stringfellows','Downpatrick',1600);

Insert Into Function Values (209,'09-Jul-2020','Wedding','Sandwiches',330,'Free Bar',3,180,175,'Pogo','Raheny',2000);

Insert Into Function Values (210,'23-Aug-2022','Corporate Event','Tea',1325,'All Drink Charged to Guest',6,120,220,'The Island','Dungiven',1835);

Insert Into Function Values (211,'21-Oct-2022','Corporate Event','Buffet',543,'No Alcohol',6,30,300,'The Island','Dungiven',1835);

Insert Into Function Values (212,'16-Mar-2021','Corporate Event','Sit Down Meal',876,'No Alcohol',2,190,175,'Copperjacks','Longford',1500);

Insert Into Function Values (213,'13-Aug-2020','Retirement','Sandwiches',479,'No Alcohol',23,300,500,'Big Apple','Listowel',800);

Insert Into Function Values (214,'21-Feb-2020','Corporate Event','Sandwiches',666,'All Drink Charged to Guest',8,40,450,'Tribes','Mountrath',1585);

Insert Into Function Values (215,'24-Aug-2020','Corporate Event','Finger Food',1814,'All Drink Charged to Guest',7,130,605,'The Island','Dungiven',1835);

Insert Into Function Values (216,'12-Sep-2022','Corporate Event','Tea',1274,'No Alcohol',11,70,250,'Portal','Belleek',1650);

Insert Into Function Values (217,'02-Apr-2021','Corporate Event','Buffet',310,'No Alcohol',20,260,490,'Copperjacks','Longford',1500);

Insert Into Function Values (218,'11-Feb-2022','Wedding','Sit Down Meal',852,'No Alcohol',3,250,250,'Boogie','Ferbane',1400);

Insert Into Function Values (219,'24-Apr-2020','Retirement','Finger Food',1883,'No Alcohol',19,140,620,'Pogo','Raheny',2000);

Insert Into Function Values (220,'15-Jun-2021','Wedding','Tea',1201,'Free Bar',22,150,50,'Ginos','Ballinrobe',1600);

Insert Into Function Values (221,'09-May-2020','Corporate Event','Buffet',779,'Free Bar',10,130,620,'Taboo','Drumcliff',1625);

Insert Into Function Values (222,'24-Dec-2021','Corporate Event','Sandwiches',1177,'No Alcohol',24,100,50,'Swish','Carndonagh',2100);

Insert Into Function Values (223,'20-Apr-2021','Wedding','Sandwiches',1192,'No Alcohol',3,20,220,'Diceys','Greystones',1800);

Insert Into Function Values (224,'26-May-2021','Birthday','Buffet',1826,'Free Bar',1,30,250,'The Buzz','Ballybofey',2225);

Insert Into Function Values (225,'21-Jul-2022','Wedding','Finger Food',1531,'Free Bar',24,250,180,'Taboo','Drumcliff',1625);

Insert Into Function Values (226,'19-Feb-2021','Retirement','Buffet',415,'Free Bar',19,100,75,'The Buzz','Ballybofey',2225);

Insert Into Function Values (227,'20-Sep-2020','Retirement','Sandwiches',1037,'Free Bar',2,60,605,'Karma','Bangor',1900);

Insert Into Function Values (228,'09-Jun-2022','Retirement','Sit Down Meal',1987,'Free Bar',4,240,340,'Blazers','Claremorris',2000);

Insert Into Function Values (229,'03-Sep-2022','Retirement','Finger Food',1327,'All Drink Charged to Guest',12,150,315,'Stringfellows','Downpatrick',1600);

Insert Into Function Values (230,'09-Jun-2022','Wedding','Sandwiches',283,'All Drink Charged to Guest',3,120,220,'Blazers','Claremorris',2000);

Insert Into Function Values (231,'13-Mar-2020','Birthday','Buffet',1095,'No Alcohol',23,150,180,'Tribes','Mountrath',1585);

Insert Into Function Values (232,'02-Mar-2020','Birthday','Finger Food',740,'Free Bar',2,120,430,'Big Apple','Listowel',800);

Insert Into Function Values (233,'30-Oct-2022','Corporate Event','Sit Down Meal',636,'No Alcohol',3,240,300,'Stringfellows','Downpatrick',1600);

Insert Into Function Values (234,'21-Aug-2022','Corporate Event','Sandwiches',931,'No Alcohol',15,100,300,'Karma','Bangor',1900);

Insert Into Function Values (235,'12-Apr-2021','Birthday','Tea',338,'All Drink Charged to Guest',23,260,315,'Copperjacks','Longford',1500);

Insert Into Function Values (236,'01-Jan-2021','Retirement','Tea',1839,'Free Bar',6,60,620,'Blue Note','Athboy',1900);

Insert Into Function Values (237,'08-Apr-2021','Corporate Event','Sit Down Meal',1538,'Free Bar',1,230,75,'Jacks','Moyne',1850);

Insert Into Function Values (238,'15-Nov-2020','Birthday','Buffet',1761,'Free Bar',12,190,430,'Karma','Bangor',1900);

Insert Into Function Values (239,'10-Nov-2020','Retirement','Sit Down Meal',1209,'All Drink Charged to Guest',7,100,100,'Jacks','Moyne',1850);

Insert Into Function Values (240,'06-Mar-2021','Corporate Event','Sandwiches',1770,'No Alcohol',24,130,75,'Diceys','Greystones',1800);

Insert Into Function Values (241,'03-Dec-2021','Wedding','Sit Down Meal',1546,'No Alcohol',21,300,175,'Diceys','Greystones',1800);

Insert Into Function Values (242,'08-Apr-2022','Corporate Event','Buffet',1287,'No Alcohol',11,10,180,'The Venue','Portadown',1325);

Insert Into Function Values (243,'31-Oct-2020','Retirement','Sandwiches',1797,'All Drink Charged to Guest',18,150,250,'Jacks','Moyne',1850);

Insert Into Function Values (244,'04-Sep-2021','Wedding','Sit Down Meal',722,'No Alcohol',9,130,50,'Stringfellows','Downpatrick',1600);

Insert Into Function Values (245,'23-Jan-2021','Corporate Event','Tea',373,'All Drink Charged to Guest',23,280,220,'The Venue','Portadown',1325);

Insert Into Function Values (246,'28-Oct-2022','Retirement','Sit Down Meal',1117,'Free Bar',11,70,220,'Tribes','Mountrath',1585);

Insert Into Function Values (247,'27-Jul-2021','Corporate Event','Tea',341,'No Alcohol',1,210,500,'Tribes','Mountrath',1585);

Insert Into Function Values (248,'01-May-2020','Corporate Event','Finger Food',1635,'All Drink Charged to Guest',22,300,300,'Portal','Belleek',1650);

Insert Into Function Values (249,'07-Aug-2021','Wedding','Sandwiches',818,'All Drink Charged to Guest',13,250,250,'The Buzz','Ballybofey',2225);

Insert Into Function Values (250,'02-Feb-2021','Retirement','Tea',1589,'All Drink Charged to Guest',23,120,490,'The Island','Dungiven',1835);

Insert Into Function Values (251,'12-Nov-2022','Corporate Event','Sandwiches',328,'No Alcohol',14,150,120,'The Venue','Portadown',1325);

Insert Into Function Values (252,'22-Aug-2022','Wedding','Sandwiches',1586,'Free Bar',7,70,325,'Ginos','Ballinrobe',1600);

Insert Into Function Values (253,'11-Mar-2020','Corporate Event','Buffet',1266,'All Drink Charged to Guest',14,270,500,'Karma','Bangor',1900);

Insert Into Function Values (254,'02-Feb-2020','Retirement','Sit Down Meal',544,'All Drink Charged to Guest',19,230,340,'The Venue','Portadown',1325);

Insert Into Function Values (255,'01-Dec-2021','Corporate Event','Sit Down Meal',1918,'Free Bar',21,100,50,'Caribou','Drogheda',2000);

Insert Into Function Values (256,'06-Feb-2021','Retirement','Buffet',1811,'All Drink Charged to Guest',9,90,325,'Blue Note','Athboy',1900);

Insert Into Function Values (257,'11-Jun-2022','Corporate Event','Sit Down Meal',1686,'Free Bar',8,300,430,'Pogo','Raheny',2000);

Insert Into Function Values (258,'26-Oct-2020','Corporate Event','Buffet',1395,'No Alcohol',13,10,250,'Diceys','Greystones',1800);

Insert Into Function Values (259,'03-Feb-2020','Wedding','Sit Down Meal',1714,'Free Bar',11,60,315,'Pulse','Ashbourne',1750);

Insert Into Function Values (260,'22-Oct-2021','Birthday','Finger Food',1413,'Free Bar',5,60,220,'Blue Note','Athboy',1900);

Insert Into Function Values (261,'07-Oct-2020','Birthday','Buffet',727,'Free Bar',21,120,620,'Portal','Belleek',1650);

Insert Into Function Values (262,'22-Jun-2022','Corporate Event','Finger Food',1485,'Free Bar',21,40,430,'Copperjacks','Longford',1500);

Insert Into Function Values (263,'04-Mar-2020','Birthday','Sit Down Meal',918,'All Drink Charged to Guest',19,150,315,'Caribou','Drogheda',2000);

Insert Into Function Values (264,'29-May-2020','Retirement','Sandwiches',868,'All Drink Charged to Guest',12,150,75,'Pulse','Ashbourne',1750);

Insert Into Function Values (265,'23-Sep-2020','Birthday','Sit Down Meal',565,'No Alcohol',21,120,325,'Big Apple','Listowel',800);

```
Insert Into  Function Values (266,'07-Mar-2022','Wedding','Sandwiches',1791,'All Drink Charged to Guest',22,240,325,'Swish','Carndonagh',2100);

Insert Into  Function Values (267,'04-Jan-2022','Wedding','Sandwiches',359,'Free Bar',13,50,315,'Diceys','Greystones',1800);

Insert Into  Function Values (268,'09-Sep-2020','Wedding','Sandwiches',1537,'No Alcohol',11,20,500,'Stringfellows','Downpatrick',1600);

Insert Into  Function Values (269,'17-Sep-2021','Retirement','Sit Down Meal',619,'No Alcohol',17,190,180,'Diceys','Greystones',1800);

Insert Into  Function Values (270,'14-May-2022','Birthday','Buffet',1570,'Free Bar',9,160,50,'Stringfellows','Downpatrick',1600);

commit;
```

SQL Queries

RANK AND DENSE_RANK FUNCTIONS

1. Show the Jockey ID and venue where each jockey has performed only for weddings
2. Display the overall number of guests who attended the event
3. Based on the number of guests who attended the event, list the ranking number/position for each jockey relative to others
4. Sort the jockeys by jockey number

```
SELECT
      Disc_Jockey_Id,
      Venue_Name,
      SUM(Num_Guests),
      RANK() OVER (ORDER BY SUM(Num_Guests) DESC) AS rank,
      DENSE_RANK() OVER (ORDER BY SUM(Num_Guests) DESC) AS dense_rank

FROM
      Function

WHERE
      Function_Type = 'Wedding'

AND
      Num_Guests IS NOT NULL

GROUP BY
      Disc_Jockey_Id,
      Venue_Name

ORDER BY
      Disc_Jockey_Id;
```

Run SQL Command Line

```
ML_SQL>SELECT Disc_Jockey_Id, Venue_Name, SUM(Num_Guests),
  2     RANK() OVER (ORDER BY SUM(Num_Guests) DESC) AS rank,
  3  DENSE_RANK() OVER (ORDER BY SUM(Num_Guests) DESC) AS dense_rank
  4  FROM Function
  5  WHERE Function_Type = 'Wedding'
  6  AND Num_Guests IS NOT NULL
  7  GROUP BY Disc_Jockey_Id, Venue_Name
  8  ORDER BY Disc_Jockey_Id;

DISC_JOCKEY_ID VENUE_NAME    SUM(NUM_GUESTS)       RANK DENSE_RANK
-------------- ------------- --------------- ---------- ----------
            20 Tribes                   1566          4          4
            20 Stringfellows            1537          6          6
            20 Diceys                   1192          9          9
            50 Diceys                    359         14         14
            60 Pulse                    1714          2          2
            70 Ginos                    1586          3          3
           120 Blazers                   283         16         16
           130 Stringfellows             722         13         13
           150 Ginos                    1201          8          8
           180 Pogo                      330         15         15
           230 Ginos                     749         12         12
           240 Swish                    1791          1          1
           250 Boogie                    852         10         10
           250 The Buzz                  818         11         11
           250 Taboo                    1531          7          7
           300 Diceys                   1546          5          5

16 rows selected.

ML_SQL>
```

MOVING AVERAGE

1. Show the venue , month and year where each jockey has performed in 2020
2. Display the overall fee paid for all the jockeys at a venue for a month
3. Compute the moving average of the jockey fee charged for a venue between the current month and the previous three months in 2020
4. Sort the venue by name

```
SELECT
      Venue_Name,
      EXTRACT(MONTH FROM Function_Date) AS month,
      EXTRACT(YEAR FROM Function_Date) AS year,
      SUM(Disc_Jockey_Fee) AS jockey_fee,
      AVG(SUM(Disc_Jockey_Fee)) OVER
                 (ORDER BY EXTRACT(MONTH FROM Function_Date)
                  ROWS BETWEEN 3 PRECEDING AND CURRENT ROW) AS moving_average

FROM
      FUNCTION

WHERE
      EXTRACT(YEAR FROM Function_Date) = 2020

GROUP BY
      Venue_Name,
      EXTRACT(YEAR FROM Function_Date),
      EXTRACT(MONTH FROM Function_Date)

ORDER BY
      Venue_Name;
```

Run SQL Command Line

```
ML_SQL>SELECT
  2     Venue_Name,
  3     EXTRACT(MONTH FROM Function_Date) AS month,
  4     EXTRACT(YEAR FROM Function_Date) AS year,
  5     SUM(Disc_Jockey_Fee) AS jockey_fee,
  6     AVG(SUM(Disc_Jockey_Fee)) OVER
  7             (ORDER BY EXTRACT(MONTH FROM Function_Date)
  8              ROWS BETWEEN 3 PRECEDING AND CURRENT ROW) AS moving_average
  9  FROM
 10     FUNCTION
 11  WHERE
 12     EXTRACT(YEAR FROM Function_Date) = 2020
 13  GROUP BY
 14     Venue_Name,
 15     EXTRACT(YEAR FROM Function_Date),
 16     EXTRACT(MONTH FROM Function_Date)
 17  ORDER BY
 18     EXTRACT(YEAR FROM Function_Date);

VENUE_NAME         MONTH       YEAR JOCKEY_FEE MOVING_AVERAGE
------------- ---------- ---------- ---------- --------------
Blazers                2       2020        670            670
Karma                 11       2020        430            350
Pulse                  2       2020        315            345
The Venue              2       2020        340         343.75
Tribes                 2       2020        450         288.75
Big Apple              3       2020        430         383.75
Caribou                3       2020        315         383.75
Karma                  3       2020        500         423.75
Tribes                 3       2020        180         356.25
Pogo                   4       2020        620         403.75
Portal                 5       2020        300            400
Pulse                  5       2020         75         293.75
Taboo                  5       2020        620         403.75
Pogo                   7       2020        175          292.5
The Buzz               7       2020        325         298.75
Big Apple              8       2020        500            405
The Island             8       2020        605         401.25
Big Apple              9       2020        325         438.75
Karma                  9       2020        605         508.75
Stringfellows          9       2020        500         508.75
Diceys                10       2020        250            420
Jacks                 10       2020        250         401.25
Portal                10       2020        620            405
Jacks                 11       2020        100            305
Boogie                 2       2020         50            360

25 rows selected.

ML_SQL>
```

REPORTING ON A SUM

1. Show the venue , month and year where each jockey has performed in 2020
2. Compute the total monthly fee of all the venues and fee paid for each jockey
3. Compute the fee paid for each jockey every month at each venue for the first quarter of the year 2021
4. Sort the output by month, name of the venue and identification number of jockey

```
SELECT
      EXTRACT(MONTH FROM Function_Date) AS month,
      Venue_Name,
      Disc_Jockey_Id,
      SUM(SUM(Disc_Jockey_Fee)) OVER
                 (PARTITION BY EXTRACT(MONTH FROM Function_Date))
                 AS total_monthly_fee,
      SUM(SUM(Disc_Jockey_Fee)) OVER
                 (PARTITION BY Disc_Jockey_Id)
                 AS total_jockey_fees
FROM
      FUNCTION

WHERE
      EXTRACT(YEAR FROM Function_Date) = 2021

AND
      EXTRACT(MONTH FROM Function_Date) <= 4

GROUP BY
      EXTRACT(MONTH FROM Function_Date),
      Venue_Name,
      Disc_Jockey_Id

ORDER BY
      EXTRACT(MONTH FROM Function_Date),
      Venue_Name,
      Disc_Jockey_Id;
```

Run SQL Command Line

```
ML_SQL>SELECT
  2     EXTRACT(MONTH FROM Function_Date) AS month,
  3     Venue_Name,
  4     Disc_Jockey_Id,
  5     SUM(SUM(Disc_Jockey_Fee)) OVER
  6             (PARTITION BY EXTRACT(MONTH FROM Function_Date))
  7                     AS total_monthly_fee,
  8     SUM(SUM(Disc_Jockey_Fee)) OVER
  9             (PARTITION BY Disc_Jockey_Id)
 10                     AS total_jockey_fees
 11  FROM
 12     FUNCTION
 13  WHERE
 14     EXTRACT(YEAR FROM Function_Date) = 2021
 15  AND
 16     EXTRACT(MONTH FROM Function_Date) <= 4
 17  GROUP BY
 18     EXTRACT(MONTH FROM Function_Date),
 19     Venue_Name,
 20     Disc_Jockey_Id
 21  ORDER BY
 22     EXTRACT(MONTH FROM Function_Date),
 23     Venue_Name,
 24     Disc_Jockey_Id;

     MONTH VENUE_NAME    DISC_JOCKEY_ID TOTAL_MONTHLY_FEE TOTAL_JOCKEY_FEES
---------- ------------- -------------- ----------------- -----------------
         1 Blue Note                 60               840               620
         1 The Venue                280               840               220
         2 Blue Note                 90               890               325
         2 The Buzz                 100               890                75
         2 The Island               120               890               490
         3 Copperjacks              190               250               175
         3 Diceys                   130               250                75
         4 Copperjacks              260              1100               805
         4 Diceys                    20              1100               220
         4 Jacks                    230              1100                75

10 rows selected.

ML_SQL>
```

RATIO OF A SUM

1. Display the name of each venue and the number of guests who attended every event at a venue in 2020
2. Compute the total number of guests who attended all the events at a venue
3. Compute the ratio of number of guests who attended a particular event to the number of guests who attended all the events at a venue
4. Sort the output by name of the venue and number of guests

```
SELECT
      Venue_Name,
      Num_Guests,
      SUM(SUM(Num_Guests)) OVER
                 (PARTITION BY Venue_Name)
                 AS total_number_guests
      RATIO_TO_REPORT(SUM(Num_Guests)) OVER
                 (PARTITION BY Venue_Name)
                  AS Num_Guests_Ratio
FROM
      FUNCTION

WHERE
      EXTRACT(YEAR FROM Function_Date) = 2020

GROUP BY
      Venue_Name,
      Num_Guests

ORDER BY
      Venue_Name,
      Num_Guests;
```

Run SQL Command Line

```
ML_SQL>SELECT
  2     Venue_Name,
  3     Num_Guests,
  4     SUM(SUM(Num_Guests)) OVER (PARTITION BY Venue_Name)
  5             AS total_number_guests,
  6     RATIO_TO_REPORT(SUM(Num_Guests)) OVER
  7             (PARTITION BY Venue_Name)
  8              AS Num_Guests_Ratio
  9  FROM FUNCTION
 10  WHERE EXTRACT(YEAR FROM Function_Date) = 2020
 11  GROUP BY Venue_Name, Num_Guests
 12  ORDER BY Venue_Name, Num_Guests;

VENUE_NAME    NUM_GUESTS TOTAL_NUMBER_GUESTS NUM_GUESTS_RATIO
------------- ---------- ------------------- ----------------
Big Apple            479                1784       .268497758
Big Apple            565                1784       .316704036
Big Apple            740                1784       .414798206
Blazers             1687                1687                1
Boogie              1396                1396                1
Caribou              918                 918                1
Diceys              1395                1395                1
Jacks               1209                3006       .402195609
Jacks               1797                3006       .597804391
Karma               1037                4064       .255167323
Karma               1266                4064       .311515748
Karma               1761                4064       .433316929
Pogo                 330                2213       .149118843
Pogo                1883                2213       .850881157
Portal               727                2362       .307790008
Portal              1635                2362       .692209992
Pulse                868                2582       .336173509
Pulse               1714                2582       .663826491
Stringfellows       1537                1537                1
Taboo                779                 779                1
The Buzz            1050                1050                1
The Island          1814                1814                1
The Venue            544                 544                1
Tribes               666                1761       .378194208
Tribes              1095                1761       .621805792

25 rows selected.

ML_SQL>
```

LISTAGG() FUNCTION

1. Display the list of names of venues of the first 5 records and sort them based on the capacity in ascending order
2. Display the highest capacity among the 5 venues

```
SELECT
      LISTAGG(Venue_Name, ‘, ’) WITHIN GROUP
                (ORDER BY Venue_Name, Venue_Capacity)
                AS Venue_List,
      MAX(Venue_Capacity)
                AS Highest_Capacity
FROM
      FUNCTION
WHERE
      Function_Id <= 205;
```

Run SQL Command Line

```
ML_SQL>SELECT
  2     LISTAGG(Venue_Name, ', ') WITHIN GROUP
  3             (ORDER BY Venue_Capacity)
  4                      AS Venue_List,
  5     MAX(Venue_Capacity)
  6                      AS Highest_Capacity
  7  FROM
  8     FUNCTION
  9  WHERE
 10     Function_Id <= 205;

VENUE_LIST
--------------------------------------------------------
HIGHEST_CAPACITY
----------------
Tribes, Stringfellows, Blazers, Caribou, The Buzz
            2225


ML_SQL>
```

LAG() and LEAD() FUNCTIONS

1. Display the month and the number of guests who attended all the events in a month
2. Display the number of the guests who attended all the events in the previous month and next month in the year 2021
3. Sort the results by month

```
SELECT
      EXTRACT(MONTH FROM Function_Date) AS month,
      SUM(Num_Guests) AS Num_Guests,
      LAG(SUM(Num_Guests), 1) OVER
                (ORDER BY EXTRACT(MONTH FROM Function_Date))
                AS Previous_Month_Guests,
      LEAD(SUM(Num_Guests), 1) OVER
                (ORDER BY EXTRACT(MONTH FROM Function_Date))
                AS Next_Month_Guests

FROM
      FUNCTION

WHERE
      EXTRACT(YEAR FROM Function_Date) = 2021

GROUP BY
      EXTRACT(MONTH FROM Function_Date)

ORDER BY
      EXTRACT(MONTH FROM Function_Date);
```

Run SQL Command Line

```
ML_SQL>SELECT
  2     EXTRACT(MONTH FROM Function_Date) AS month,
  3     SUM(Num_Guests) AS Num_Guests,
  4     LAG(SUM(Num_Guests), 1) OVER
  5  (ORDER BY EXTRACT(MONTH FROM Function_Date))
  6                    AS Previous_Month_Guests,
  7     LEAD(SUM(Num_Guests), 1) OVER
  8  (ORDER BY EXTRACT(MONTH FROM Function_Date))
  9                    AS Next_Month_Guests
 10  FROM
 11  FUNCTION
 12  WHERE
 13  EXTRACT(YEAR FROM Function_Date) = 2021
 14  GROUP BY
 15  EXTRACT(MONTH FROM Function_Date)
 16  ORDER BY
 17  EXTRACT(MONTH FROM Function_Date);

     MONTH NUM_GUESTS PREVIOUS_MONTH_GUESTS NEXT_MONTH_GUESTS
---------- ---------- --------------------- -----------------
         1       2212                                    3815
         2       3815                  2212              2646
         3       2646                  3815              3378
         4       3378                  2646              1826
         5       1826                  3378              1201
         6       1201                  1826               341
         7        341                  1201               818
         8        818                   341              1341
         9       1341                   818              1413
        10       1413                  1341              1419
        11       1419                  1413              4641
        12       4641                  1419

12 rows selected.

ML_SQL>
```

FIRST() and LAST() FUNCTIONS

1. Based on the number of guests who attended all the events at all the venues in all the locations, compute the busiest and least busiest months in 2020

```
SELECT
      MIN(EXTRACT(MONTH FROM Function_Date)) KEEP
            (DENSE_RANK FIRST ORDER BY SUM(Num_Guests))
            AS Most_Busiest_Month,
      MIN(EXTRACT(MONTH FROM Function_Date)) KEEP
            (DENSE_RANK LAST ORDER BY SUM(Num_Guests))
            AS Least_Busiest_Month

FROM
      FUNCTION

WHERE
      EXTRACT(YEAR FROM Function_Date) = 2020

GROUP BY
      EXTRACT(MONTH FROM Function_Date);
```

Run SQL Command Line

```
ML_SQL>SELECT
  2     MIN(EXTRACT(MONTH FROM Function_Date)) KEEP
  3             (DENSE_RANK FIRST ORDER BY SUM(Num_Guests))
  4             AS Most_Busiest_Month,
  5     MIN(EXTRACT(MONTH FROM Function_Date)) KEEP
  6             (DENSE_RANK LAST ORDER BY SUM(Num_Guests))
  7             AS Least_Busiest_Month
  8  FROM
  9     FUNCTION
 10  WHERE
 11     EXTRACT(YEAR FROM Function_Date) = 2020
 12  GROUP BY
 13     EXTRACT(MONTH FROM Function_Date);

MOST_BUSIEST_MONTH LEAST_BUSIEST_MONTH
------------------ -------------------
                 7                   2

ML_SQL>
```

CUMULATIVE SUM FUNCTION

1. Display the name of each jockey whose type of music is pop and who performed only in the year 2021
2. Display the fee charged by each jockey at every event
3. Compute and display the cumulative sum of fee charged by each of the jockeys at all the events

```
SELECT
      D.Fname, D.Music_Type,
      F.Disc_Jockey_Fee,
      SUM(F.Disc_Jockey_Fee) OVER
            (ORDER BY D.Fname ROWS BETWEEN UNBOUNDED
             PRECEDING AND CURRENT ROW) AS Cumulative_Total

FROM
      FUNCTION F, DISC_JOCKEY D

WHERE
      D.Music_Type = 'Pop'

AND
      EXTRACT(MONTH FROM Function_Date) = 1

AND
      EXTRACT(YEAR FROM Function_Date) = 2021;
```

Select Run SQL Command Line

```
ML_SQL>SELECT
  2     D.Fname, D.Music_Type,
  3     F.Disc_Jockey_Fee,
  4     SUM(F.Disc_Jockey_Fee) OVER
  5             (ORDER BY D.Fname ROWS BETWEEN UNBOUNDED
  6              PRECEDING AND CURRENT ROW) AS Cumulative_Total
  7  FROM
  8  FUNCTION F, DISC_JOCKEY D
  9  WHERE
 10  D.Music_Type = 'Pop'
 11  AND
 12  EXTRACT(MONTH FROM Function_Date) = 1
 13  AND
 14  EXTRACT(YEAR FROM Function_Date) = 2021;

FNAME      MUSIC_TYPE   DISC_JOCKEY_FEE CUMULATIVE_TOTAL
---------- ------------ --------------- ----------------
Brett      Pop                      220              220
Brett      Pop                      620              840
Crawford   Pop                      620             1460
Crawford   Pop                      220             1680
Shannon    Pop                      220             1900
Shannon    Pop                      620             2520

6 rows selected.

ML_SQL>
```

PERCENT RANK FUNCTION

1. Display the name of all the venues at all the locations
2. Display the total number of guests who attended all the events in each venue
3. Calculate the cumulative distribution of number of guests who attended all the events at a venue to the group of all the events at all the venues
4. Compute the relative rank percentile of each venue based on the total number of guests who attended all the events at the venue.

```
SELECT
      F.Venue_Name,
      SUM(F. Num_Guests),
      CUME_DIST() OVER
            (ORDER BY SUM(F. Num_Guests) DESC) AS Cum_Dist,
      PERCENT_RANK() OVER
            (ORDER BY SUM(F. F. Num_Guests) DESC) AS Percent_Rank

FROM
      FUNCTION F

GROUP BY
      F.Venue_Name

ORDER BY
      F.Venue_Name;
```

Run SQL Command Line

```
ML_SQL>SELECT
  2     F.Venue_Name,
  3     SUM(F. Num_Guests),
  4     CUME_DIST() OVER
  5             (ORDER BY SUM(F. Num_Guests) DESC) AS Cum_Dist,
  6     PERCENT_RANK() OVER
  7             (ORDER BY SUM(F. Num_Guests) DESC) AS Percent_Rank
  8  FROM
  9     FUNCTION F
 10  GROUP BY
 11     F.Venue_Name
 12  ORDER BY
 13  F.Venue_Name;

VENUE_NAME    SUM(F.NUM_GUESTS)   CUM_DIST PERCENT_RANK
------------- ----------------- ---------- ------------
Big Apple                  1784          1            1
Blazers                    3957         .5   .473684211
Blue Note                  5063         .2   .157894737
Boogie                     2248        .95   .947368421
Caribou                    4401         .4   .368421053
Copperjacks                3009         .7   .684210526
Diceys                     6881         .1   .052631579
Ginos                      3536        .65   .631578947
Jacks                      4544        .35   .315789474
Karma                      4995        .25   .210526316
Pogo                       3899        .55   .526315789
Portal                     3636         .6   .578947368
Pulse                      2582         .8   .789473684
Stringfellows              8098        .05            0
Swish                      2968        .75   .736842105
Taboo                      2310         .9   .894736842
The Buzz                   4109        .45   .421052632
The Island                 5271        .15   .105263158
The Venue                  2532        .85   .842105263
Tribes                     4785         .3   .263157895

20 rows selected.

ML_SQL>
```

NTILE() FUNCTION

1. Display the jockey ID along with the combined first and last names of all the jockeys who performed only for birthday events at a venue whose capacity ranges between 500 and 2000
2. Compute the total sum of fee charged by each jockey for all the birthday events
3. Categorize the jockeys into 5 groups based on the total fee charged by each of the jockeys
4. Sort the output by jockey ID and combined first and last names of jockeys

```
SELECT
      D.Disc_Jockey_Id,
      CONCAT(CONCAT(D.Fname, ‘ ’), D.Sname) AS Name,
      SUM(F. Disc_Jockey_Fee),
      NTILE(5) OVER
            (ORDER BY SUM(F. Disc_Jockey_Fee) DESC) AS ntile

FROM
      FUNCTION F, DISC_JOCKEY D

WHERE
      F. Disc_Jockey_Id = D. Disc_Jockey_Id

AND
      F.Function_Type = ‘Birthday’

AND
      F.Venue_Capacity BETWEEN 500 AND 2000

GROUP BY
      D.Disc_Jockey_Id,
      CONCAT(CONCAT(D.Fname, ‘ ’), D.Sname)

ORDER BY
      D.Disc_Jockey_Id,
      CONCAT(CONCAT(D.Fname, ‘ ’), D.Sname);
```

Run SQL Command Line

```
ML_SQL>SELECT
  2  D.Disc_Jockey_Id,
  3  CONCAT(CONCAT(D.Fname, ' '), D.Sname) AS Name,
  4  SUM(F. Disc_Jockey_Fee),
  5  NTILE(5) OVER
  6  (ORDER BY SUM(F. Disc_Jockey_Fee) DESC) AS ntile
  7  FROM
  8  FUNCTION F, DISC_JOCKEY D
  9  WHERE
 10  F. Disc_Jockey_Id = D. Disc_Jockey_Id
 11  AND
 12  F.Function_Type = 'Birthday'
 13  AND
 14  F.Venue_Capacity BETWEEN 500 AND 2000
 15  GROUP BY
 16  D.Disc_Jockey_Id,
 17  CONCAT(CONCAT(D.Fname, ' '), D.Sname)
 18  ORDER BY
 19  D.Disc_Jockey_Id,
 20  CONCAT(CONCAT(D.Fname, ' '), D.Sname);

DISC_JOCKEY_ID NAME                    SUM(F.DISC_JOCKEY_FEE)      NTILE
-------------- --------------------- ---------------------- ----------
            50 Huntington Hambridge                     670          1
            60 Mattheus Brahm                           220          3
           120 Shaun Gormally                          1375          1
           150 Emmerich Plackstone                      495          2
           160 Druci Downgate                            50          4
           190 Domeniga Loisi                           430          2
           260 Feliza Gonning                           315          3
           270 Chane Veel                                50          5

8 rows selected.

ML_SQL>
```

HYPOTHETICAL RANK FUNCTION

1. Rank the total capacity of all the venues in all the locations based on a capacity of 1500 guests.

```
SELECT
      RANK(1500) WITHIN GROUP
            (ORDER BY SUM(Venue_Capacity) DESC) AS Rank,
      PERCENT_RANK(1500) WITHIN GROUP
            (ORDER BY SUM(Venue_Capacity) DESC) AS Percent_Rank

FROM
      FUNCTION

GROUP BY
      Venue_Capacity;
```

Run SQL Command Line

```
ML_SQL>SELECT
  2     RANK(1500) WITHIN GROUP
  3             (ORDER BY SUM(Venue_Capacity) DESC) AS Rank,
  4     PERCENT_RANK(1500) WITHIN GROUP
  5             (ORDER BY SUM(Venue_Capacity) DESC) AS Percent_Rank
  6  FROM
  7     FUNCTION
  8  GROUP BY
  9     Venue_Capacity;

      RANK PERCENT_RANK
---------- ------------
        17            1

ML_SQL>
```

PIVOT CLAUSE

1. Display the jockey IDs which have the following values – 30, 50, 120, 140, 170, 230 and 300.
2. Display the trend in total fee charged by these jockeys for all the events at all the venues only for the first 6 months in the year 2021

```
SELECT *
      FROM (
              SELECT
                 EXTRACT(MONTH FROM Function_Date) AS month,
                 Disc_Jockey_Id,
                 Disc_Jockey_Fee

              FROM
                 Function

              WHERE
                 EXTRACT(YEAR FROM Function_Date) = 2020

              AND
                 Disc_Jockey_Id IN (30, 50, 120, 140, 170, 230, 300)
            )

PIVOT (
      SUM(Disc_Jockey_Fee) FOR month IN
                                 (1 AS JAN,
                                  2 AS FEB,
                                  3 AS MAR,
                                  4 AS APR,
                                  5 AS MAY,
                                  6 AS JUN)
      )

ORDER BY
      Disc_Jockey_Id;
```

Run SQL Command Line

```
ML_SQL>SELECT *
  2     FROM (
  3                SELECT
  4                     EXTRACT(MONTH FROM Function_Date) AS month,
  5                     Disc_Jockey_Id,
  6                     Disc_Jockey_Fee
  7                FROM
  8                     Function
  9                WHERE
 10                     EXTRACT(YEAR FROM Function_Date) = 2020
 11                AND
 12                     Disc_Jockey_Id IN (30, 50, 120, 140, 170, 230, 300)
 13             )
 14   PIVOT (
 15     SUM(Disc_Jockey_Fee) FOR month IN
 16                                          (1 AS JAN,
 17                                           2 AS FEB,
 18                                           3 AS MAR,
 19                                           4 AS APR,
 20                                           5 AS MAY,
 21                                           6 AS JUN)
 22   )
 23   ORDER BY
 24     Disc_Jockey_Id;

DISC_JOCKEY_ID        JAN        FEB        MAR        APR        MAY        JUN
-------------- ---------- ---------- ---------- ---------- ---------- ----------
            50                   670
           120                              430
           140                                         620
           230                   340
           300                                                    300

ML_SQL>
```

MODEL CLAUSE

1. Display the identification number of an event, first name of the jockey, type of event, and the part of day when the event was started. Transform the numerical column of the start time of event into categories such as 'Morning', 'Afternoon', 'Evening', and 'Night'.
2. Display only the events whose IDs range between 215 and 225
3. Display the fee charged during these events as 500

```
SELECT *

FROM
      FUNCTION F,
      DISC_JOCKEY D

WHERE
      F.Disc_Jockey_Id = D.Disc_Jockey_Id

AND
      F.Function_Id BETWEEN 215 AND 225

MODEL
      DIMENSION BY (F.Function_Id)
      MEASURES (D.Fname,
                F.Function_Type,
                 CASE
                      WHEN F.Start_Time BETWEEN 6 AND 11
                            THEN 'Morning'
                      WHEN F.Start_Time BETWEEN 6 AND 11
                            THEN 'Afternoon'
                      WHEN F.Start_Time BETWEEN 6 AND 11
                            THEN 'Evening'
                      ELSE 'Night'
                            END AS Timing,
                   500 AS Fee)
RULES();
```

Run SQL Command Line

```
ML_SQL>SELECT *
  2  FROM
  3     FUNCTION F,
  4     DISC_JOCKEY D
  5  WHERE
  6     F.Disc_Jockey_Id = D.Disc_Jockey_Id
  7  AND
  8     F.Function_Id BETWEEN 215 AND 225
  9  MODEL
 10  DIMENSION BY (F.Function_Id)
 11  MEASURES (D.Fname,
 12         F.Function_Type,
 13         CASE
 14                     WHEN F.Start_Time BETWEEN 6 AND 11
 15                             THEN 'Morning'
 16                     WHEN F.Start_Time BETWEEN 6 AND 11
 17                             THEN 'Afternoon'
 18                     WHEN F.Start_Time BETWEEN 6 AND 11
 19                             THEN 'Evening'
 20                     ELSE 'Night'
 21                             END AS Timing,
 22          500 AS Fee)
 23  RULES();

FUNCTION_ID FNAME      FUNCTION_TYPE   TIMING           FEE
----------- ---------- --------------- --------- ----------
        223 Aleece     Wedding         Night            500
        224 Kingsley   Birthday        Night            500
        216 Irwinn     Corporate Event Morning          500
        222 Jojo       Corporate Event Night            500
        221 Charity    Corporate Event Morning          500
        215 Charity    Corporate Event Morning          500
        219 Mellie     Retirement      Night            500
        220 Emmerich   Wedding         Night            500
        225 Neddie     Wedding         Night            500
        218 Neddie     Wedding         Night            500
        217 Feliza     Corporate Event Night            500

11 rows selected.

ML_SQL>
```

MODEL CLAUSE WITH RULES

1. Display the identification number of an event, first name of the jockey, and fee charged by each jockey at each of the events.
2. Display only the events whose IDs range between 100 and 150, the jockeys whose IDs range between 200 and 270, and the county of the jockeys which has 'ord' in its name
3. Create a new column for commission and assign the value 50 to all these jockeys
4. Create another column to compute the total fee by adding the commission to fee charged by these jockeys for each of the events

```
SELECT *
FROM
      Function F,
      Disc_Jockey D
WHERE
      F.Disc_Jockey_Id = D.Disc_Jockey_Id
AND
      F.Disc_Jockey_Id BETWEEN 100 AND 150
AND
      F.Function_Id BETWEEN 200 AND 270
AND
      D.County LIKE '%ord%'
MODEL
      PARTITION BY (F.Function_Id)
      DIMENSION BY (D.Fname)
      MEASURES (F.Disc_Jockey_Fee,
                50 AS Commission,
                0 AS Total_Fee)
      RULES(Total_Fee[ANY] = Disc_Jockey_Fee[cv()] +
                             Commission[cv()]
      );
```

Run SQL Command Line

```
ML_SQL>SELECT *
  2  FROM
  3     FUNCTION F,
  4     DISC_JOCKEY D
  5  WHERE
  6     F.Disc_Jockey_Id = D.Disc_Jockey_Id
  7  AND
  8     F.Disc_Jockey_Id BETWEEN 100 AND 150
  9  AND
 10     F.Function_Id BETWEEN 200 AND 270
 11  AND
 12     D.County LIKE '%ord%'
 13  MODEL
 14     PARTITION BY (F.Function_Id)
 15     DIMENSION BY (D.Fname)
 16     MEASURES (F.Disc_Jockey_Fee,
 17                 50 AS Commission,
 18                 0 AS Total_Fee)
 19  RULES(Total_Fee[ANY] = Disc_Jockey_Fee[cv()] + Commission[cv()]
 20     );

FUNCTION_ID FNAME      DISC_JOCKEY_FEE COMMISSION  TOTAL_FEE
----------- ---------- --------------- ---------- ----------
        263 Emmerich              315         50        365
        220 Emmerich               50         50        100
        229 Emmerich              315         50        365
        251 Emmerich              120         50        170
        231 Emmerich              180         50        230
        264 Emmerich               75         50        125
        243 Emmerich              250         50        300

7 rows selected.

ML_SQL>
```

PIVOT CLAUSE FOR MULTIPLE COLUMNS

1. Display the jockey IDs of the jockeys who were born between 1980 and 1990.
2. Display the average and sum of fee of each of the jockeys for May and June, and who conducts shows only during the weekends

```
SELECT *
     FROM (
             SELECT
               EXTRACT(MONTH FROM F.Function_Date) AS Month,
               D.Disc_Jockey_Id,
               F.Disc_Jockey_Fee

             FROM
               Function F, Disc_Jockey D

             WHERE

               F.Disc_Jockey_Id = D.Disc_Jockey_Id


             AND
               EXTRACT(MONTH FROM F.Function_Date) IN (5, 6)

             AND
               D.Availability IN ('Weekends')

             AND
               EXTRACT(YEAR FROM D.Date_Of_Birth)
                               BETWEEN 1980 AND 1990
           )
PIVOT (
        SUM(Disc_Jockey_Fee) AS Sum_Amount,
        AVG(Disc_Jockey_Fee) AS Avg_Amount
               FOR (Month) IN
                               (5 AS MAY,
                               6 AS JUN)
     )
ORDER BY
     Disc_Jockey_Id;
```

Run SQL Command Line

```
ML_SQL>SELECT *
  2     FROM (
  3                SELECT
  4                     EXTRACT(MONTH FROM F.Function_Date) AS Month,
  5                     D.Disc_Jockey_Id,
  6                     F.Disc_Jockey_Fee
  7                        FROM
  8                     Function F, Disc_Jockey D
  9                        WHERE
 10                     F.Disc_Jockey_Id = D.Disc_Jockey_Id
 11                        AND
 12                     EXTRACT(MONTH FROM F.Function_Date) IN (5, 6)
 13                        AND
 14                     D.Availability IN ('Weekends')
 15                        AND
 16                     EXTRACT(YEAR FROM D.Date_Of_Birth)
 17                               BETWEEN 1980 AND 1990
 18              )
 19  PIVOT (
 20        SUM(Disc_Jockey_Fee) AS Sum_Amount,
 21        AVG(Disc_Jockey_Fee) AS Avg_Amount
 22                     FOR (Month) IN
 23                                          (5 AS MAY,
 24                                             6 AS JUN)
 25     )
 26  ORDER BY
 27     Disc_Jockey_Id;

DISC_JOCKEY_ID MAY_SUM_AMOUNT MAY_AVG_AMOUNT JUN_SUM_AMOUNT JUN_AVG_AMOUNT
-------------- -------------- -------------- -------------- --------------
            40                                         430            430
           120                                         220            220
           130            620            620
           160             50             50
           300            300            300            430            430

ML_SQL>
```

FETCHING CONSECUTIVE RECORDS

1. Find the type of events which were organized consecutively for 4 times in all the venues across all the locations in 2020 and 2021

```
WITH PARAMS (N) AS (SELECT 4 FROM DUAL) -- SET N = 4
SELECT DISTINCT
      Function_Type
FROM (
      SELECT
            Function_Id, Function_Type, n,
            MIN(Function_Type) OVER
                  (ORDER BY Function_Id ROWS BETWEEN n - 1 PRECEDING AND
                                                     CURRENT ROW) AS Min,
            MAX(Function_Type) OVER
                  (ORDER BY Function_Id ROWS BETWEEN n - 1 PRECEDING AND
                                                     CURRENT ROW) AS Max,
            COUNT(*) OVER
                  (ORDER BY Function_Id ROWS BETWEEN n - 1 PRECEDING AND
                                                      CURRENT ROW) AS Cnt
      FROM
            Function
      CROSS JOIN
            PARAMS
        ) X
WHERE
      Min = Max AND Cnt = N
```

Run SQL Command Line

```
ML_SQL>WITH PARAMS (N) AS (SELECT 4 FROM DUAL) -- SET N = 4
  2  SELECT DISTINCT
  3     Function_Type
  4  FROM (
  5     SELECT
  6                     Function_Id, Function_Type, n,
  7                     MIN(Function_Type) OVER
  8                     (ORDER BY Function_Id ROWS BETWEEN n - 1 PRECEDING AND
  9                                                 CURRENT ROW) AS Min,
 10                     MAX(Function_Type) OVER
 11                              (ORDER BY Function_Id ROWS BETWEEN n - 1 PRECEDING AND
 12                                                 CURRENT ROW) AS Max,
 13                     COUNT(*) OVER
 14                              (ORDER BY Function_Id ROWS BETWEEN n - 1 PRECEDING AND
 15                                                 CURRENT ROW) AS Cnt
 16     FROM
 17             Function
 18     CROSS JOIN
 19             PARAMS
 20       ) X
 21  WHERE
 22     Min = Max AND Cnt = N;

FUNCTION_TYPE
---------------
Corporate Event
Retirement

ML_SQL>_
```

CASE GROUPING AND ROLLUP FUNCTIONS

1. Display the name of the venue, type of the event and type of bar for only the venues whose name starts with the letter B or D
2. Group the venues based on the type of event and type of bar
3. Compute the total fee charged by all the jockeys at each venue for each type of event and each type of bar
4. Display the total fee charged by all the jockeys at each venue for all the type of events and all the type of bars

```
SELECT
      CASE GROUPING(Venue_Name)
            WHEN 1 THEN 'All Venues'
            ELSE Venue_Name
      END AS Venue,
      CASE GROUPING(Function_Type)
            WHEN 1 THEN 'All Events'
            ELSE Function_Type
      END AS Event,
      CASE GROUPING(Bar_Details)
            WHEN 1 THEN 'All Bar Types'
            ELSE Bar_Details
      END AS Type_of_Bar,
      Sum(Disc_Jockey_Fee)
FROM
      Function
WHERE
      Venue_Name LIKE 'B%' OR
      Venue_Name LIKE 'D%'
GROUP BY
      ROLLUP(Venue_Name, Function_Type, Bar_Details)
ORDER BY
      Venue_Name, Function_Type, Bar_Details;
```

Run SQL Command Line

```
ML_SQL>SELECT
  2     CASE GROUPING(Venue_Name)
  3             WHEN 1 THEN 'All Venues'
  4             ELSE Venue_Name
  5     END AS Venue,
  6     CASE GROUPING(Function_Type)
  7             WHEN 1 THEN 'All Events'
  8             ELSE Function_Type
  9     END AS Event,
 10     CASE GROUPING(Bar_Details)
 11             WHEN 1 THEN 'All Bar Types'
 12             ELSE Bar_Details
 13     END AS Type_of_Bar,
 14     Sum(Disc_Jockey_Fee)
 15  FROM
 16     Function
 17  WHERE
 18     Venue_Name LIKE 'B%' OR
 19     Venue_Name LIKE 'D%'
 20  GROUP BY
 21     ROLLUP(Venue_Name, Function_Type, Bar_Details)
 22  ORDER BY
 23     Venue_Name, Function_Type, Bar_Details;

VENUE         EVENT           TYPE_OF_BAR                SUM(DISC_JOCKEY_FEE)
------------- --------------- -------------------------- --------------------
Big Apple     Birthday        Free Bar                                    430
Big Apple     Birthday        No Alcohol                                  325
Big Apple     Birthday        All Bar Types                               755
Big Apple     Retirement      No Alcohol                                  500
Big Apple     Retirement      All Bar Types                               500
Big Apple     All Events      All Bar Types                              1255
Blazers       Birthday        Free Bar                                    670
Blazers       Birthday        All Bar Types                               670
Blazers       Retirement      Free Bar                                    340
Blazers       Retirement      All Bar Types                               340
Blazers       Wedding         All Drink Charged to Guest                  220
Blazers       Wedding         All Bar Types                               220
Blazers       All Events      All Bar Types                              1230
Blue Note     Birthday        Free Bar                                    220
Blue Note     Birthday        All Bar Types                               220
Blue Note     Retirement      All Drink Charged to Guest                  325
Blue Note     Retirement      Free Bar                                    620
Blue Note     Retirement      All Bar Types                               945
Blue Note     All Events      All Bar Types                              1165
Boogie        Birthday        All Drink Charged to Guest                   50
Boogie        Birthday        All Bar Types                                50
Boogie        Wedding         No Alcohol                                  250
Boogie        Wedding         All Bar Types                               250
Boogie        All Events      All Bar Types                               300
Diceys        Corporate Event No Alcohol                                  325
Diceys        Corporate Event All Bar Types                               325
Diceys        Retirement      No Alcohol                                  180
Diceys        Retirement      All Bar Types                               180
Diceys        Wedding         Free Bar                                    315
Diceys        Wedding         No Alcohol                                  395
Diceys        Wedding         All Bar Types                               710
Diceys        All Events      All Bar Types                              1215
All Venues    All Events      All Bar Types                              5165

33 rows selected.

ML_SQL>
```

ROW NUMBER FUNCTION

1. Display the name of the venue and the type of food that was served at the venue only for those events where the food served includes 'Tea', 'Buffet', and 'Sandwiches' in their menu
2. Display the maximum count of number of times a particular type of food was served in all the venues, and the maximum count of venues where this particular food was served
3. Sort the output by the maximum count

```
SELECT
      Type_of_Food,
      SUM(CNT) AS TOTAL_COUNT,
            MAX(CASE WHEN SEQNUM = 1 THEN Venue_Name END) AS Venue_WITH_MAX,
            MAX(CASE WHEN SEQNUM = 1 THEN CNT END) AS CNT_AT_MAX
FROM (
      SELECT
            SUBSTR(Food_Details, 1, 10) AS Type_of_Food,
            Venue_Name, COUNT(*) AS CNT,
                  ROW_NUMBER() OVER
                  (PARTITION BY SUBSTR(Food_Details, 1, 10)
                  ORDER BY COUNT(*) DESC) AS SEQNUM
             FROM
                  Function
             WHERE
                  Food_Details LIKE '%Tea%' OR
                  Food_Details LIKE '%Buffet%' OR
                  Food_Details LIKE '%Sandwiches%'
             GROUP BY
                  SUBSTR(Food_Details, 1, 10),
                  Venue_Name
```

```
    ) T
GROUP BY
    Type_of_Food
ORDER BY
    SUM(CNT) DESC;
```

Run SQL Command Line

```
ML_SQL>SELECT
  2     Type_of_Food,
  3     SUM(CNT) AS TOTAL_COUNT,
  4         MAX(CASE WHEN SEQNUM = 1 THEN Venue_Name END) AS Venue_WITH_MAX,
  5         MAX(CASE WHEN SEQNUM = 1 THEN CNT END) AS CNT_AT_MAX
  6  FROM (
  7     SELECT
  8             SUBSTR(Food_Details, 1, 10) AS Type_of_Food,
  9             Venue_Name, COUNT(*) AS CNT,
 10             ROW_NUMBER() OVER
 11             (PARTITION BY SUBSTR(Food_Details, 1, 10)
 12             ORDER BY COUNT(*) DESC) AS SEQNUM
 13        FROM
 14             Function
 15        WHERE
 16             Food_Details LIKE '%Tea%' OR
 17             Food_Details LIKE '%Buffet%' OR
 18             Food_Details LIKE '%Sandwiches%'
 19        GROUP BY
 20             SUBSTR(Food_Details, 1, 10),
 21             Venue_Name
 22        ) T
 23  GROUP BY
 24     Type_of_Food
 25  ORDER BY
 26     SUM(CNT) DESC;

TYPE_OF_FOOD                               TOTAL_COUNT VENUE_WITH_MA CNT_AT_MAX
------------------------------------------ ----------- ------------- ----------
Sandwiches                                          19 Diceys                 3
Buffet                                              14 Karma                  2
Tea                                                 11 The Island             2

ML_SQL>
```

CORR FUNCTION

1. Display the jockey id, start time of the event at which the jockey performed, and the fee charged by the jockey during an event organized in the year 2020 and 2021
2. Compute the correlation between the start time and fee charged by each jockey during each event for which the jockey with id as 100 has performed.

```
SELECT
       Disc_Jockey_Id, Start_Time, Disc_Jockey_Fee,
       CORR(Start_Time, Disc_Jockey_Fee)
            OVER(PARTITION BY Disc_Jockey_Id) Correlation

FROM
      FUNCTION

WHERE
      EXTRACT(YEAR FROM Function_Date) = 2020 OR
      EXTRACT(YEAR FROM Function_Date) = 2021

AND
      Disc_Jockey_Id IN (100)

ORDER BY
      Food_Details,
      Correlation;
```

Run SQL Command Line

```
ML_SQL>SELECT
  2       Disc_Jockey_Id, Start_Time, Disc_Jockey_Fee,
  3       CORR(Start_Time, Disc_Jockey_Fee)
  4              OVER(PARTITION BY Disc_Jockey_Id) Correlation
  5   FROM
  6      FUNCTION
  7   WHERE
  8      EXTRACT(YEAR FROM Function_Date) = 2020 OR
  9      EXTRACT(YEAR FROM Function_Date) = 2021
 10   AND
 11      Disc_Jockey_Id IN (100)
 12   ORDER BY
 13      Food_Details,
 14      Correlation;

DISC_JOCKEY_ID START_TIME DISC_JOCKEY_FEE CORRELATION
-------------- ---------- --------------- -----------
           100         19              75  -.94563984
           120         21             620  .164100003
           150         23             180  .546108245
           130         10             620           1
           270         14             500           1
           270          4              50           1
           190         12             430
            10         13             250
           120          2             430  .164100003
           130          7             605           1
           300         22             300           1
           140         19             620
            60          2             605          -1
           100         24              50  -.94563984
           150         12              75  .546108245
           150         18             250  .546108245
           300         23             500           1
            20         11             500
            40          8             450
           180          3             175
            60         11             315          -1
           100         21              50  -.94563984
           100          7             100  -.94563984
           120         21             325  .164100003
           150         19             315  .546108245
           230         19             340           1
           230          5             325           1
            50         16             670

28 rows selected.

ML_SQL>
```

LIST AGGREGATE FUNCTION FOR CATEGORIES

1. List the different venues and their location where the jockeys performed in the year 2021
2. Exclude the aggregated list for the venues located at 'Greystones', 'Longford', 'Drumcliff', 'Ballybofey', and 'Downpatrick'.

```
SELECT
      DISTINCT D.Fname “Jockey”, F.Venue_Location, F.Venue_Name,
      LISTAGG(Venue_Name, ‘, ’) WITHIN GROUP
                 (ORDER BY D.Fname , Venue_Name)
                 OVER (PARTITION BY D.Fname) AS Venue_List,

FROM
      FUNCTION F, Disc_Jockey D

WHERE
      F.Disc_Jockey_Id = D.Disc_Jockey_Id

AND
      EXTRACT(YEAR FROM F.Function_Date) = 2021

AND
      F.Venue_Location IN ('Greystones', 'Longford', 'Drumcliff',
                           'Ballybofey','Downpatrick' )

ORDER BY
      D.Fname;
```

Run SQL Command Line

```
ML_SQL>SELECT
  2     DISTINCT D.Fname "Jockey", F.Venue_Location,
  3     LISTAGG(Venue_Name, ', ') WITHIN GROUP
  4             (ORDER BY D.Fname , Venue_Name)
  5              OVER (PARTITION BY D.Fname) AS Venue_List
  6  FROM
  7     FUNCTION F, Disc_Jockey D
  8  WHERE
  9     F.Disc_Jockey_Id = D.Disc_Jockey_Id
 10  AND
 11     EXTRACT(YEAR FROM F.Function_Date) = 2021
 12  AND
 13     F.Venue_Location NOT IN ('Greystones', 'Longford', 'Drumcliff',
 14                                       'Ballybofey','Downpatrick')
 15  ORDER BY
 16     D.Fname;

Jockey     VENUE_LOCAT
---------- -----------
VENUE_LIST
------------------------------------------------------------
Bancroft   Portadown
The Venue

Crawford   Athboy
Blue Note

Edithe     Mountrath
Tribes

Emmerich   Ballinrobe
Ginos

Harald     Moyne
Jacks

Jojo       Carndonagh
Caribou, Swish

Jojo       Drogheda
Caribou, Swish

Mattheus   Athboy
Blue Note, Blue Note

Shaun      Dungiven
The Island


9 rows selected.

ML_SQL>_
```

Nth VALUE FUNCTION

1. Find the second highest occurrence of a venue based on the number of guests who attended the event in the year 2021; whose function IDs are between 70 and 170
2. Display the Jockey, Venue of the event and also the number of guests who attended the event
3. Filter the records only for jockeys whose email address has ‘gov’ or ‘com’ and who was born in the year 1988.
4. Filter the records for number of guests between 500 and 2000 and the name of the location of venue that starts with the letter ‘L’

```
SELECT
      F.Venue_Name, D.Fname, F.Num_Guests,
      Nth_Value(F.Venue_Name, 2) OVER
            (PARTITION BY D.Fname
             ORDER BY F.Num_Guests
             ROWS BETWEEN UNBOUNDED PRECEDING AND
             UNBOUNDED FOLLOWING) AS Second_Highest_Venue
FROM
      FUNCTION F, Disc_Jockey D
WHERE
      F.Disc_Jockey_Id = D.Disc_Jockey_Id
AND
      EXTRACT(YEAR FROM F.Function_Date) = 2021
AND
      F.Function_Id BETWEEN 70 AND 170
AND
      D.Email LIKE '%gov%' OR
      D.Email LIKE '%com%'
AND
      EXTRACT(YEAR FROM D.Date_Of_Birth) = 1988
AND
      F.Num_Guests BETWEEN 500 AND 2000
AND
      F.Venue_Location LIKE 'L%'
GROUP BY
```

```
        D.Fname,
        F.Venue_Name,
        F.Num_Guests
ORDER BY
        D.Fname,
        F.Venue_Name,
        F.Num_Guests;
```

Run SQL Command Line

```
ML_SQL>SELECT
  2     F.Venue_Name, D.Fname, F.Num_Guests,
  3     Nth_Value(F.Venue_Name, 2) OVER
  4             (PARTITION BY D.Fname
  5              ORDER BY F.Num_Guests
  6              ROWS BETWEEN UNBOUNDED PRECEDING AND
  7              UNBOUNDED FOLLOWING) AS Second_Highest_Venue
  8  FROM
  9     FUNCTION F, Disc_Jockey D
 10  WHERE
 11     F.Disc_Jockey_Id = D.Disc_Jockey_Id
 12  AND
 13     EXTRACT(YEAR FROM F.Function_Date) = 2021
 14  AND
 15     F.Function_Id BETWEEN 70 AND 170
 16  AND
 17     D.Email LIKE '%gov%' OR
 18     D.Email LIKE '%com%'
 19  AND
 20     EXTRACT(YEAR FROM D.Date_Of_Birth) = 1988
 21  AND
 22     F.Num_Guests BETWEEN 500 AND 2000
 23  AND
 24     F.Venue_Location LIKE 'L%'
 25  GROUP BY
 26     D.Fname,
 27     F.Venue_Name,
 28     F.Num_Guests
 29  ORDER BY
 30     D.Fname,
 31     F.Venue_Name,
 32     F.Num_Guests;

VENUE_NAME    FNAME      NUM_GUESTS SECOND_HIGHES
------------- ---------- ---------- -------------
Big Apple     Baxie             565 Big Apple
Big Apple     Baxie             740 Big Apple
Copperjacks   Baxie             876 Big Apple
Copperjacks   Baxie            1485 Big Apple
Big Apple     Druci             565 Big Apple
Big Apple     Druci             740 Big Apple
Copperjacks   Druci             876 Big Apple
Copperjacks   Druci            1485 Big Apple

8 rows selected.

ML_SQL>
```

AGGREGATE FUNCTIONS

1. Display the first name of the jockey, name of the venue where the jockey performed, and the fee charged by the jockey during each event, only for the jockeys whose first name is ‘Emmerich’.
2. Compute the minimum, average, and maximum fee charged by the jockeys during each event at each venue.

```
SELECT
      D.Fname, F.Venue_Name, F.Disc_Jockey_Fee,
      MAX(F.Disc_Jockey_Fee) OVER
            (PARTITION BY D.Fname
             ORDER BY F.Disc_Jockey_Fee) AS Max_Fee,
      AVG(F.Disc_Jockey_Fee) OVER
            (PARTITION BY D.Fname
             ORDER BY F.Disc_Jockey_Fee) AS Avg_Fee,
      MIN(F.Disc_Jockey_Fee) OVER
            (PARTITION BY D.Fname
             ORDER BY F.Disc_Jockey_Fee) AS Min_Fee
FROM
      FUNCTION F, Disc_Jockey D
WHERE
      F.Disc_Jockey_Id = D.Disc_Jockey_Id
AND
      D.Fname IN (‘Emmerich’)
ORDER BY
      D.Fname,
      F.Disc_Jockey_Fee;
```

Run SQL Command Line

```
ML_SQL>SELECT
  2     D.Fname, F.Venue_Name, F.Disc_Jockey_Fee,
  3     MAX(F.Disc_Jockey_Fee) OVER
  4             (PARTITION BY D.Fname
  5              ORDER BY F.Disc_Jockey_Fee) AS Max_Fee,
  6     AVG(F.Disc_Jockey_Fee) OVER
  7             (PARTITION BY D.Fname
  8              ORDER BY F.Disc_Jockey_Fee) AS Avg_Fee,
  9     MIN(F.Disc_Jockey_Fee) OVER
 10             (PARTITION BY D.Fname
 11              ORDER BY F.Disc_Jockey_Fee) AS Min_Fee
 12  FROM
 13     FUNCTION F, Disc_Jockey D
 14  WHERE
 15     F.Disc_Jockey_Id = D.Disc_Jockey_Id
 16  AND
 17     D.Fname IN ('Emmerich')
 18  ORDER BY
 19     D.Fname,
 20     F.Disc_Jockey_Fee;

FNAME      VENUE_NAME    DISC_JOCKEY_FEE    MAX_FEE    AVG_FEE    MIN_FEE
---------- ------------- --------------- ---------- ---------- ----------
Emmerich   Ginos                      50         50         50         50
Emmerich   Pulse                      75         75       62.5         50
Emmerich   The Venue                 120        120 81.6666667         50
Emmerich   Tribes                    180        180     106.25         50
Emmerich   Jacks                     250        250        135         50
Emmerich   Caribou                   315        315 186.428571         50
Emmerich   Stringfellows             315        315 186.428571         50

7 rows selected.

ML_SQL>
```

LINEAR REGRESSION FUNCTION

1. Train a linear regression model to predict the number of guests who would attend an event based on the start time of an event.
2. Predict the number of guests who would attend an event whose start time is 20, 3, 14 and 24
3. Analyze if the input value is outside the range of all the trained input values and categorize the prediction as interpolation or extrapolation
4. Compute the correlation of the predicted value with the input value.

```
SELECT
    ROUND(c + m * ST.Start_Time, 2) Num_of_Guests,
    ST.Start_Time,
    CASE
        WHEN ST.Start_Time BETWEEN EQ.Min_Time AND EQ.Max_Time
        THEN 'INTERPOLATION'
        ELSE 'EXTRAPOLATION'
    END AS ANALYTICS,
    CORRELATION
FROM
        (
        SELECT
               REGR_SLOPE(F.Num_Guests, F.Start_Time) AS m,
               REGR_INTERCEPT(F.Num_Guests, F.Start_Time) AS c,
               MIN(F.Start_Time) AS Min_Time,
               MAX(F.Start_Time) AS Max_Time ,
               ROUND(CORR(F.Num_Guests, F.Start_Time), 3) AS CORRELATION
        FROM
               Function F,
               Disc_Jockey D
        WHERE
               F.Disc_Jockey_Id = D.Disc_Jockey_Id
        ) EQ,
        (
        SELECT
               20 AS Start_Time
        FROM
               DUAL
        UNION ALL
```

```
        SELECT
               3 AS Start_Time
        FROM
               DUAL
        UNION ALL
        SELECT
               14 AS Start_Time
        FROM
               DUAL
        UNION ALL
        SELECT
               24 AS Start_Time
        FROM
               DUAL
        ) ST
ORDER BY Start_Time;
```

Select Run SQL Command Line

```
ML_SQL>SELECT
  2       ROUND(c + m * ST.Start_Time, 2) Num_of_Guests,
  3       ST.Start_Time,
  4       CASE
  5           WHEN ST.Start_Time BETWEEN EQ.Min_Time AND EQ.Max_Time
  6           THEN 'INTERPOLATION'
  7           ELSE 'EXTRAPOLATION'
  8       END AS ANALYTICS,
  9       CORRELATION
 10   FROM
 11           (
 12           SELECT
 13                  REGR_SLOPE(F.Num_Guests, F.Start_Time) AS m,
 14                  REGR_INTERCEPT(F.Num_Guests, F.Start_Time) AS c,
 15                  MIN(F.Start_Time) AS Min_Time,
 16                  MAX(F.Start_Time) AS Max_Time ,
 17                  ROUND(CORR(F.Num_Guests, F.Start_Time), 3) AS CORRELATION
 18           FROM
 19                  Function F,
 20                  Disc_Jockey D
 21           WHERE
 22                  F.Disc_Jockey_Id = D.Disc_Jockey_Id
 23           ) EQ,
 24     (
 25           SELECT
 26                  20 AS Start_Time
 27           FROM
 28                  DUAL
 29           UNION ALL
 30           SELECT
 31                  3 AS Start_Time
 32           FROM
 33                  DUAL
 34          UNION ALL
 35           SELECT
 36                  14 AS Start_Time
 37           FROM
 38                  DUAL
 39          UNION ALL
 40           SELECT
 41                  24 AS Start_Time
 42           FROM
 43                  DUAL
 44           ) ST
 45   ORDER BY Start_Time;

NUM_OF_GUESTS START_TIME ANALYTICS     CORRELATION
------------- ---------- ------------- -----------
      1144.99          3 INTERPOLATION         .01
      1152.73         14 INTERPOLATION         .01
      1156.95         20 INTERPOLATION         .01
      1159.77         24 INTERPOLATION         .01

ML_SQL>
```

YouTube Channel Video Upload Link

https://youtu.be/XNlwvMrzbWY