-- 1.Retrieve the names and contact numbers of all participants?

```
SELECT Name
,ContactNumber
FROM Participants;
```

-- 2.Find the total amount donated for the festival?

```
SELECT SUM(Amount) AS TOTAL_AMOUNT
FROM Donations;
```

-- 3.List the events organized by participants under the age of 30?

```
SELECT E.EventName
,P.Name
,P.Age
FROM Events AS E
JOIN Participants AS P ON P.ParticipantID = E.OrganizerID
WHERE P.Age < 30;
```

-- 4.Calculate the average rating for each event along with the event name?

```
SELECT E.EventName
,AVG(F.Rating) AS Average_rating
FROM Events AS E
JOIN Feedback AS F USING(EventID)
GROUP BY E.EventName
ORDER BY AVG(F.Rating) DESC;
```

-- 5.Retrieve the names and details of the idols that cost more than $150 ?

```
SELECT * FROM Idols
WHERE Price > 150;
```

-- 6.Calculate the total amount donated by each participant?

```
SELECT P.Name
,SUM(D.Amount) AS TOTAL_AMOUNT
FROM Donations AS D
JOIN Participants AS P ON D.DonorID = P.ParticipantID
GROUP BY P.Name
ORDER BY SUM(D.Amount) DESC;
```

-- 7.Find the event with the most registrations?

```
SELECT E.EventID
,E.EventName
FROM Events AS E
JOIN Registration AS R USING(EventID)
GROUP BY  E.EventID
,E.EventName
HAVING MAX(R.RegistrationID)
LIMIT 1;
```

```
-- 8.List the top 3 idols by weight?

SELECT IdolID
,IdolName
,MAX(weight) AS Weight
FROM Idols
GROUP BY IdolID
,IdolName
ORDER BY MAX(weight) DESC
LIMIT 3;

-- 9.Find the youngest and oldest participants?

WITH RankAge AS
(
SELECT Name
,Age
,ROW_NUMBER() OVER(ORDER BY Age ASC) AS Youngest_Rank
,ROW_NUMBER() OVER(ORDER BY Age DESC) AS Oldest_Rank
FROM Participants

)

SELECT Youngest.Name AS Youngest_Participant
,Youngest.Age AS Young_Age
,Oldest.Name AS Oldest_Participant
,Oldest.Age AS Old_Age
FROM RankAge AS Youngest
JOIN RankAge AS Oldest
ON Youngest.Youngest_Rank = 1 AND Oldest.Oldest_Rank = 1;


-- 10.Find the events that had the highest and lowest average ratings based on feedback ?

WITH EventAverageRatings AS
(
SELECT e.EventName
,AVG(f.Rating) AS Average_Rating
FROM Events AS e
JOIN Feedback AS f USING(EventID)
GROUP BY e.EventName
)
SELECT HighRated.EventName AS Highest_Rated_Event
,HighRated.Average_Rating AS Highest_Average_Rating
,LowRated.EventName AS Lowest_Rated_Event
,LowRated.Average_Rating AS Lowest_Average_Rating
FROM

(
SELECT EventName
,Average_Rating
,DENSE_RANK() OVER(ORDER BY Average_Rating DESC) AS Highest_Rank
FROM EventAverageRatings
)   AS HighRated
```

```
INNER JOIN

(
SELECT EventName
,Average_Rating
,DENSE_RANK() OVER(ORDER BY Average_Rating ASC) AS Lowest_Rank
FROM EventAverageRatings
)   AS LowRated

ON Highest_Rank = 1 AND Lowest_Rank = 1;


-- 11. Calculate the difference in days between the event date and the registration date for each
registration?


SELECT EventID
,EventDate
,RegistrationDate
,DATEDIFF(EventDate
,RegistrationDate) AS Date_Difference
FROM Events
JOIN Registration USING(EventID)
ORDER BY EventID;

-- 12.Calculate the total number of donations made in each month of the year?

WITH MonthlyDonations AS
(
SELECT MONTH(DonationDate) AS Donation_Month
,COUNT(DonationID) AS Total_Donations
FROM Donations
GROUP BY Donation_Month
)
SELECT Donation_Month
,Total_Donations
FROM MonthlyDonations;

-- 13.Calculate the total number of registrations for each event and rank them?

WITH EventRegistrationCount AS
(
SELECT COUNT(R.RegistrationID) AS TOTAL_REGISTRATIONS
,E.EventName AS EVENTS
FROM Registration AS R
JOIN Events AS E USING(EventID)
GROUP BY E.EventName
)

SELECT  EVENTS
,TOTAL_REGISTRATIONS
,RANK() OVER(ORDER BY TOTAL_REGISTRATIONS DESC) AS RANKING
FROM EventRegistrationCount;
```

-- 14.Retrieve the previous and next participants (based on participant ID) for each participant?

```
SELECT ParticipantID
,Name
,LAG(ParticipantID)OVER(ORDER BY(ParticipantID)) AS Previous_Participant
,LEAD(ParticipantID)OVER(ORDER BY(ParticipantID)) AS Next_Participant
FROM Participants
GROUP BY ParticipantID,Name;
```

-- 15.Find the participants who registered for events and provide their contact numbers along with a flag indicating whether their contact number contains the area code "555." ?

```
WITH ContactsWithAreaCode AS

(
SELECT Name AS Participants_Name
,ContactNumber
,CASE
WHEN ContactNumber LIKE "%555%" THEN "YES"
ELSE "NO"
END AS HasAreaCode555
FROM Participants
JOIN Registration USING(ParticipantID)
)
SELECT Participants_Name
,ContactNumber
,HasAreaCode555
FROM ContactsWithAreaCode;
```