```
SELECT * FROM campusx.ipl;
```

TOP 5 batsman in respective Teams.

```
SELECT * FROM (SELECT BattingTeam,batter,SUM(batsman_run) AS 'total_runs',
DENSE_RANK() OVER(PARTITION BY BattingTeam ORDER BY SUM(batsman_run) DESC) AS
'rank_within_team'
FROM ipl
GROUP BY BattingTeam,batter) t
WHERE t.rank_within_team < 6
ORDER BY t.BattingTeam,t.rank_within_team;
```

How many Total run/Total Avg Runs/Total Running Avg Runs scored by VK till his 50th ,100th match?

```
SELECT * FROM (SELECT
CONCAT("Match-",CAST(ROW_NUMBER() OVER(ORDER BY ID) AS CHAR)) AS 'match_no',
SUM(batsman_run) AS 'runs_scored',
SUM(SUM(batsman_run)) OVER w AS 'career_runs',
AVG(SUM(batsman_run)) OVER w AS 'career_avg',
AVG(SUM(batsman_run)) OVER(ROWS BETWEEN 9 PRECEDING AND CURRENT ROW) AS
'rolling_avg'
FROM ipl
WHERE batter = 'V Kohli'
GROUP BY ID
WINDOW w AS (ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)) t
```

For particular Restaurant which one is the IMP Food.

```
SELECT f_name,
(total_value/SUM(total_value) OVER())*100 AS 'percent_of_total'
FROM (SELECT f_id,SUM(amount) AS 'total_value' FROM orders t1
JOIN order_details t2
ON t1.order_id = t2.order_id
WHERE r_id = 5
GROUP BY f_id) t
JOIN food t3
ON t.f_id = t3.f_id
ORDER BY (total_value/SUM(total_value) OVER())*100 DESC
```

How much month by month % Change in views?

```
SELECT YEAR(Date),QUARTER(Date),SUM(views) AS 'views',
((SUM(views) - LAG(SUM(views)) OVER(ORDER BY
YEAR(Date),QUARTER(Date)))/LAG(SUM(views)) OVER(ORDER BY
YEAR(Date),QUARTER(Date)))*100 AS 'Percent_change'
FROM youtube_views
GROUP BY YEAR(Date),QUARTER(Date)
ORDER BY YEAR(Date),QUARTER(Date);
```

```
SELECT *,
((Views - LAG(Views,7) OVER(ORDER BY Date))/LAG(Views,7) OVER(ORDER BY Date))*100 AS
'weekly_percent_change'
FROM youtube_views;
```

Find 50 percentile of marks for students by branch.

```
SELECT *,
PERCENTILE_DISC(0.5) WITHIN GROUP(ORDER BY marks) OVER(PARTITION BY branch) AS
'median_marks',
PERCENTILE_CONT(0.5) WITHIN GROUP(ORDER BY marks) OVER(PARTITION BY branch)
AS 'median_marks_cont'
FROM marks;
```

To remove Outlier

```
SELECT * FROM (SELECT *,
PERCENTILE_CONT(0.25) WITHIN GROUP(ORDER BY marks) OVER() AS 'Q1',
PERCENTILE_CONT(0.75) WITHIN GROUP(ORDER BY marks) OVER() AS 'Q3'
FROM marks) t
WHERE t.marks <= t.Q1 - (1.5*(t.Q3 - t.Q1));
```

Creating N-Bucket as per marks

```
SELECT *,
NTILE(3) OVER(ORDER BY marks DESC) AS 'buckets'
FROM marks;
```

Creating N-Bucket as per Price

```
SELECT brand_name,model,price,
CASE
     WHEN bucket = 1 THEN 'budget'
  WHEN bucket = 2 THEN 'mid-range'
  WHEN bucket = 3 THEN 'premium'
END AS 'phone_type'
FROM (SELECT brand_name,model,price,
NTILE(3) OVER(PARTITION BY brand_name ORDER BY price) AS 'bucket'
FROM smartphones) t;
```

```
SELECT * FROM (SELECT *,
CUME_DIST() OVER(ORDER BY marks) AS 'Percentile_Score'
FROM marks) t
WHERE t.Percentile_Score > 0.90;
```

Partition By more than 2 columns.

```
SELECT * FROM (SELECT source,destination,airline,AVG(price) AS 'avg_fare',
DENSE_RANK() OVER(PARTITION BY source,destination ORDER BY AVG(price)) AS 'rank'
FROM flights
GROUP BY source,destination,airline) t
WHERE t.rank < 2
```