SELECT \* FROM campusx.ipl;

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;

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

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

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;

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;

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));

SELECT \*,

NTILE(3) OVER(ORDER BY marks DESC) AS 'buckets'

FROM marks;

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;

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