```sql
/* Que1 Find the sales across all product
Find the total Sales for Each Product
Additionally provide details such as Orderid, orderdate */

select orderid,
orderdate,
productid,
sales,
sum(sales) over() as Total_sales,
sum(sales) over(Partition by productid) as Total_sales_by_prod
from sales.orders;

/* Que2 Rank each order based on their sales from highest to lowest
Additionally Provide details such as orderid,orderdate */

select orderid,
orderdate,
sales,
rank() over(order by sales desc) as Ranking
from sales.Orders;

/* Que3- Find the total Number of Orders
Find the Total number of orders for each customer
Additionally provide details such as orderid,orderdate */

select
orderid,orderdate,sales,customerid,
count(*) over() as Total_orders,
count(*) over(partition by customerid) as orders_by_customer
from sales.Orders;


/* Que-4 Find the Percentage Contribution of each product's sales
to the total sales in decreasing order*/
select orderid,productid,sales,
sum(sales) over() as Totalsales,
round((cast(sales as float)/sum(sales) over())*100,2) as percentage_of_total
from sales.Orders
order by round((cast(sales as float)/sum(sales) over())*100,2) desc ;

/* Que5 Find The average sales across all orders
find the average sales for each Product
Additionally provide details such as orderid,orderdate */

select orderid,orderdate,productid,sales,
avg(sales) over() as avg_sales,
avg(sales) over(partition by Productid) as Avg_sales_By_product
from sales.orders;
```

```sql
/* Que6 Find the average Scores of Customers.
Additionally Provide details such as Customerid and last Name */

select Customerid,lastname,score,
avg(coalesce(score,0)) over() as avg_score
from sales.Customers;

/* NOTE at the time of finding avg if null present in data this give wrong result
-- For  finding average we use coalesce(score,0) this coalesce put zero in the
  place of null */

/* Que7 Find all orders where sales are higher the average sales across all orders
   */

select orderid,productid,customerid,sales from sales.orders
where sales>(select
avg(coalesce(sales,0))
from sales.Orders);

/* Que 8 Find the Highest & Lowest sales across all orders
and the highest and lowest sales for each product */

select orderid,productid,orderdate,sales,
max(sales) over() max_sales,
min(sales) over() min_sales,
max(sales) over(partition by productid) as product_wise_max_sales,
min(sales) over(partition by productid) as product_wise_min_sales
from sales.orders;


/* Que 9 Find the deviation of each sales from the Minimum and maximum sales
  amount */

select orderid,orderdate,sales,
max(sales) over() as max_sales,
min(sales) over() as min_sales,
max(sales) over() - sales as Deviation_from_max,
sales - min(sales) over() as Deviation_from_min
from sales.orders;

/* Que 10 Calculate the moving average of sales for each product over time */

select orderid,productid,orderdate,sales,
avg(sales) over(partition by productid) as avg_sales,
avg(sales) over(partition by productid order by orderdate) as moving_avg
from sales.orders;

/* Que 11 Calculate the moving average of sales for each product over time
```

```sql
and Calculate the moving average of sales for each product over time, including
  only the next order */

select orderid,productid,orderdate,sales,
avg(sales) over(partition by productid) as avg_sales,
avg(sales) over(partition by productid order by orderdate) as moving_avg,
avg(sales) over(partition by productid order by orderdate rows between current row
   and 1 following) rolling_avg
from sales.orders;
```