**E-Commerce Database**

1. **Retrieve the names and contact details of customers in a specific geographic location.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

C.customerID,

C.FirstName,

C.LastName,

C.EmailAddress

FROM

sales S JOIN customers C

ON S.customer\_ID = C.customerID

JOIN territories T

ON S.territory\_ID = T.territory\_id

WHERE

T.country = 'United States'

**2. Show the product names and prices sorted in descending order of price.**

SELECT

ProductName,

ProductPrice

FROM

Products

ORDER BY 2 DESC

**3. Select all sales product records where the quantity sold is greater than 500.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

Product\_ID,

SUM(OrderQuantity) AS Total\_Quantity

FROM

sales

GROUP BY 1

HAVING SUM(OrderQuantity) > 500

ORDER BY 2 DESC

**4. Find customers who have made purchases in the last month.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

DISTINCT C.FirstName

FROM sales S JOIN customers C

ON S.customer\_ID = C.customerID

WHERE

OrderDate >= (SELECT MAX(OrderDate) - INTERVAL '1 month' FROM sales)

**5. Display the top 5 products with the highest sales quantity.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

ProductName,

SUM(OrderQuantity) AS Total\_Quantity

FROM

sales S JOIN products P

ON S.product\_ID = P.product\_ID

GROUP BY 1

ORDER BY 2 DESC

LIMIT 5

**6. List the products with sales quantities between 5 and 10.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

),

product\_rank as

(SELECT

ProductName,

SUM(OrderQuantity) AS Total\_Quantity,

ROW\_NUMBER() over (ORDER BY SUM(OrderQuantity) DESC) as ranks

FROM

sales S JOIN products P

ON S.product\_ID = P.product\_ID

GROUP BY 1

)

SELECT

ProductName,

Total\_Quantity

FROM

product\_rank

WHERE

ranks between 5 and 10

**7. Retrieve the 10 most recent sales records.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT \* FROM SALES

ORDER BY OrderNumber DESC

LIMIT 10

**8. Calculate the total number of customers.**

SELECT COUNT(CustomerID) FROM Customers

**9. Find the average price of products in each category.**

SELECT

CategoryName,

ROUND(AVG(ProductPrice),2) as TotalPrice

FROM

products P JOIN product\_subcategories SC

ON P.subCategory\_id = SC.subcategoryID

JOIN product\_categories C

ON SC.categoryID = C.productCategoryID

GROUP BY 1

ORDER BY 2 DESC

**10. Determine the total revenue generated from sales.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

ROUND(SUM(ProductPrice \* OrderQuantity),2) as Total\_Revenue

FROM

sales S JOIN products P

ON S.product\_ID = P.product\_ID

**11. Identify the territory with the highest total sales.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

Territory\_id,

Country,

Region,

ROUND(SUM(ProductPrice \* OrderQuantity),2) as Total\_Revenue

FROM

sales S JOIN products P

ON S.product\_ID = P.product\_ID

JOIN Territories T

ON S.territory\_ID = T.territory\_id

GROUP BY 1

ORDER BY 4 DESC

**12. Compute the average quantity of items sold per sale.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

ROUND(AVG(OrderQuantity),2)

FROM

SALES

**13. Retrieve the names of customers who made purchases, along with the product names they bought.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

CONCAT(C.FirstName,' ',C.LastName) as CustomerName,

P.productName

FROM

sales S JOIN products P

ON S.product\_ID = P.product\_ID

JOIN Customers C

ON S.customer\_ID = C.customerID

**14. List all sales records with customer names, product names, and territory details.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

CONCAT(C.FirstName,' ',C.LastName) as CustomerName,

P.ProductName,

T.Region,

T.country

FROM

sales S JOIN products P

ON S.product\_ID = P.product\_ID

JOIN Customers C

ON S.customer\_ID = C.customerID

JOIN Territories T

ON S.territory\_ID = T.territory\_id

**15. Find the products with no associated sales records.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

DISTINCT P."ProductName"

FROM

sales S right join products P

ON S.product\_ID = P.product\_ID

WHERE

S.product\_ID is null

**16. Identify customers who have not made any purchases.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

DISTINCT CONCAT(C.FirstName,' ',C.LastName) as CustomerName

FROM

SALES S right JOIN Customers C

ON S.customer\_ID = C.customerID

WHERE

S.customer\_ID is null

**17. Find products with prices higher than the average price.**

SELECT

DISTINCT ProductName

FROM

Products

WHERE

ProductPrice>(SELECT AVG(ProductPrice) FROM Products)

ORDER BY

ProductPrice DESC

**18. Identify products with sales quantities greater than the average quantity.**

WITH sales AS

(

SELECT \*

FROM

Sales\_2015

UNION

SELECT \*

FROM

Sales\_2016

UNION

SELECT \*

FROM

Sales\_2017

),

cte2 AS

(

SELECT

P.ProductName,

SUM(S.OrderQuantity) AS sales\_quantity

FROM

Products P

JOIN

SALES S

ON

P.Product\_ID=S.Product\_ID

GROUP BY 1

)

SELECT

DISTINCT ProductName

FROM

cte2

WHERE

sales\_quantity > (SELECT AVG(sales\_quantity) FROM cte2)

**19. Display average price, average sales and total revenue generated by each category.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

CategoryName,

ROUND(AVG(ProductPrice),2)as Avg\_Price,

ROUND(AVG(OrderQuantity),2)as Avg\_Order\_Quantity,

ROUND(SUM(ProductPrice \* OrderQuantity),2) as Revenue

FROM

sales S JOIN products P

ON S.product\_ID = P.product\_ID

JOIN Product\_Subcategories SC

ON P.subCategory\_id = SC.subcategoryID

JOIN product\_categories C

ON SC.categoryID = C.productCategoryID

GROUP BY 1

**20. Find the number of sales made in each month.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

TO\_CHAR(OrderDate,'yyyy-mm') AS Month,

ROUND(SUM(ProductPrice \* OrderQuantity),2) as Revenue

FROM

sales S JOIN products P

ON S.product\_ID = P.product\_ID

GROUP BY 1

ORDER BY 1 DESC

**21. Find the customers with the highest total purchase amount.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

CONCAT(C.FirstName,' ',C.LastName) as CustomerName,

ROUND(SUM(ProductPrice \* OrderQuantity),2) as Revenue

FROM

sales S JOIN products P

ON S.product\_ID = P.product\_ID

JOIN Customers C

ON S.customer\_ID = C.customerID

GROUP BY 1

ORDER BY 2 DESC

**22. Calculate the total sales revenue for each month and territory.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

TO\_CHAR(OrderDate,'yyyy-mm') AS Month,

Country,

Region,

ROUND(SUM(ProductPrice \* OrderQuantity),2) as Total\_Revenue

FROM

sales S JOIN products P

ON S.product\_ID = P.product\_ID

JOIN Territories T

ON S.territory\_ID = T.territory\_id

GROUP BY 1,2,3

ORDER BY 4 DESC

**23. Find the month with the highest sales and the product sold during that month.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

),

max\_month as

(SELECT

TO\_CHAR(OrderDate,'yyyy-mm') as m\_month,

ROUND(SUM(ProductPrice \* OrderQuantity),2) as Total\_Revenue

FROM

sales S JOIN products P

ON S.product\_ID = P.product\_ID

GROUP BY 1)

, max\_sale\_month as (

SELECT

\*

FROM

SALES

WHERE

TO\_CHAR(OrderDate,'yyyy-mm') =

(SELECT m\_month FROM max\_month WHERE

Total\_Revenue = (SELECT MAX(Total\_Revenue) FROM max\_month)))

SELECT

DISTINCT ProductName

FROM

max\_sale\_month S JOIN Products P

ON S.product\_ID = P.product\_ID

**24. Find the sales records for the last quarter.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

\*

FROM

sales S JOIN customers C

ON S.customer\_ID = C.customerID

WHERE

OrderDate >= (SELECT MAX(OrderDate) - INTERVAL '3 months' FROM sales)

**25. Identify the busiest day of the week for sales.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

TO\_CHAR(OrderDate,'Day') AS weekday,

ROUND(SUM(ProductPrice \* OrderQuantity),2) as Total\_Revenue

FROM

sales S JOIN products P

ON S.product\_ID = P.product\_ID

GROUP BY 1

ORDER BY 2 DESC

**26. Find out first sale date and last sale date of all products.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

DISTINCT Product\_ID,

FIRST\_VALUE(OrderDate) OVER (PARTITION BY Product\_ID ORDER BY OrderDate) as first\_sale\_date,

FIRST\_VALUE(OrderDate) OVER (PARTITION BY Product\_ID ORDER BY OrderDate DESC) as last\_sale\_date

FROM

sales

**27. Figure out order quantity trends of products.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

)

SELECT

Product\_ID,

OrderDate,

LAG(SUM(OrderQuantity)) OVER (PARTITION BY Product\_ID ORDER BY OrderDate) as prev\_sale\_qty,

SUM(OrderQuantity) as current\_quantity,

LEAD(SUM(OrderQuantity)) OVER (PARTITION BY Product\_ID ORDER BY OrderDate) as next\_sale\_qty

FROM

sales

GROUP BY

Product\_ID, OrderDate

**28. Monthly revenue growth.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

),

monthly\_revenue as

(SELECT

TO\_CHAR(OrderDate,'yyyy-mm') as month,

LAG(SUM(ProductPrice \* OrderQuantity)) OVER (ORDER BY TO\_CHAR(OrderDate,'yyyy-mm')) as Prev\_Month\_Revenue,

ROUND(SUM(ProductPrice \* OrderQuantity),2) as Revenue,

LEAD(SUM(ProductPrice \* OrderQuantity)) OVER (ORDER BY TO\_CHAR(OrderDate,'yyyy-mm')) as Next\_Month\_Revenue

FROM

sales S JOIN products P

ON S.product\_ID = P.product\_ID

GROUP BY 1

ORDER BY 1 ASC)

SELECT

month,

ROUND(Prev\_Month\_Revenue,2) as Prev\_Month\_Revenue,

Revenue,

ROUND(((Revenue / Prev\_Month\_Revenue) - 1) \* 100 , 2) as revenue\_growth

FROM

monthly\_revenue

**29. List out top 5% highest contributing customers.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

),

per\_rank\_cust AS

(SELECT

CONCAT(C.FirstName,' ',C.LastName) as CustomerName,

ROUND(SUM(ProductPrice \* OrderQuantity),2) as Revenue,

NTILE(100) OVER (ORDER BY SUM(ProductPrice \* OrderQuantity) DESC) as per\_rank

FROM

sales S JOIN products P

ON S.product\_ID = P.product\_ID

JOIN Customers C

ON S.customer\_ID = C.customerID

GROUP BY 1)

SELECT

CustomerName,

Revenue

FROM

per\_rank\_cust

WHERE

per\_rank < 5

**30. Summation of currently monthly revenue and previous monthly revenue.**

WITH sales AS

(

SELECT \* FROM sales\_2015

UNION

SELECT \* FROM sales\_2016

UNION

SELECT \* FROM sales\_2017

),

monthly\_rev as

(SELECT

TO\_CHAR(OrderDate,'yyyy-mm') as month,

ROUND(SUM(ProductPrice \* OrderQuantity),2) as Revenue

FROM

sales S JOIN products P

ON S.product\_ID = P.product\_ID

GROUP BY 1

ORDER BY 1 ASC

)

SELECT

\*,

SUM(Revenue) OVER (ORDER BY month ROWS BETWEEN 1 PRECEDING AND CURRENT ROW)

FROM

monthly\_rev