Assignment 5:

1- Table Creation:

create table Sales(

id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY;

ProductName varchar2(2000) not null,

Category varchar2(2000),

QuantitySold number(38,0),

PricePerUnit number(38,10),

DaleDate date

);

2- Sample data insertion:

insert into Sales (ProductName, Category, QuantitySold, PricePerUnit, DaleDate) values ('Laptop','Electronics','2','1000.00',TO\_DATE('2024-01-05', 'YYYY-MM-DD'));

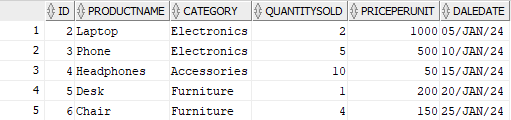
insert into Sales (ProductName, Category, QuantitySold, PricePerUnit, DaleDate) values ('Phone','Electronics','5','500.00',TO\_DATE('2024-01-10', 'YYYY-MM-DD'));

insert into Sales (ProductName, Category, QuantitySold, PricePerUnit, DaleDate) values ('Headphones','Accessories','10','50.00',TO\_DATE('2024-01-15', 'YYYY-MM-DD'));

insert into Sales (ProductName, Category, QuantitySold, PricePerUnit, DaleDate) values ('Desk','Furniture','1','200.00',TO\_DATE('2024-01-20', 'YYYY-MM-DD'));

insert into Sales (ProductName, Category, QuantitySold, PricePerUnit, DaleDate) values ('Chair','Furniture','4','150.00',TO\_DATE('2024-01-25', 'YYYY-MM-DD'));

* Result:



3- Applying provided query statements:

* Retrieve all sales transactions where the product category is "Electronics".

select \* from Sales where Category = 'Electronics';

Result:

A screenshot of a computer

Description automatically generated

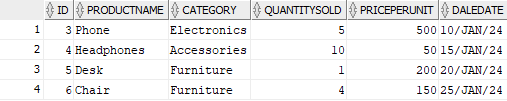
* Retrieve all sales transactions that occurred between January 10, 2024, and January 25, 2024.

select \* from Sales

where DaleDate >= TO\_DATE('2024-01-10', 'YYYY-MM-DD')

AND DaleDate <= TO\_DATE('2024-01-25', 'YYYY-MM-DD');

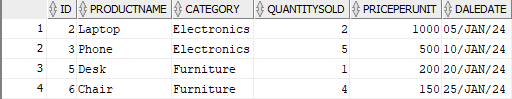
Result:



* Retrieve all sales where the price per unit is greater than $100.

select \* from Sales where PricePerUnit > 100;

Result:



* Retrieve all sales where the quantity sold is less than or equal to 3.

select \* from Sales where QuantitySold <= 3;

Result:



* Retrieve all sales of "Furniture" products where the quantity sold is greater than 2.

select \* from Sales where Category = 'Furniture' AND QuantitySold > 2;

Result:



* Retrieve all sales where the product name starts with the letter "L" (using pattern matching with LIKE).

select \* from Sales where ProductName like 'L%';

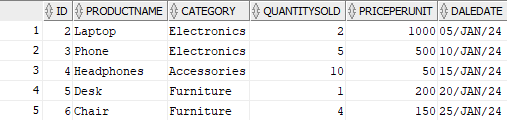
Result:



* Retrieve all sales where the category is not null.

select \* from Sales where Category is not null;

Result:



* Retrieve all sales that occurred on January 20, 2024.

select \* from Sales where DaleDate = TO\_DATE('2024-01-20', 'YYYY-MM-DD');

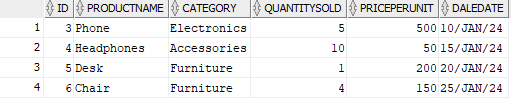
Result:



* Retrieve all sales where the price per unit is between $50 and $500.

select \* from Sales where PricePerUnit between 50 AND 500;

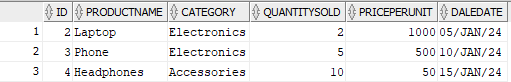
Result:



* Retrieve all sales where either the category is "Electronics" or the quantity sold is greater than 4.

select \* from Sales where Category = 'Electronics' OR QuantitySold > 4;

Result:



* Find the total revenue generated for each product category, but only for categories where the total revenue is greater than $1000.

select Category, sum(QuantitySold\*PricePerUnit) as TotalRevenue

from Sales Group by Category

having sum(QuantitySold\*PricePerUnit) > 1000;

Result:



* For each product, find the total quantity sold, but only for products where more than 3 units have been sold.

select ProductName, sum(QuantitySold)

from Sales Group by ProductName

having sum(QuantitySold) > 3;

Result:

A screenshot of a computer

Description automatically generated

* Find the average price per unit for each category, but only for categories where the average price is greater than $200.

select Category, avg(PricePerUnit)

from Sales Group by Category

having avg(PricePerUnit) > 200;

Result:



* Retrieve the maximum quantity sold for each product category, but only for categories that have sold at least 10 units in total.

select Category, max(QuantitySold)

from Sales Group by Category

having sum(QuantitySold) >= 10;

Result:



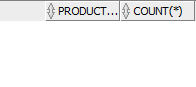
* Find the number of sales transactions for each product, but only include products that have had more than 2 sales transactions.

select ProductName, count(\*)

from Sales Group by ProductName

having count(\*) > 2;

Result:



* Find the minimum price per unit for each product category, but only for categories that have sold more than 5 units.

select Category, min(PricePerUnit)

from Sales Group by Category

having sum(QuantitySold) > 5;

Result:



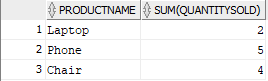
* Calculate the total quantity sold for each product, but only for products where the total revenue generated (QuantitySold \* PricePerUnit) exceeds $500.

select ProductName, sum(QuantitySold)

from Sales Group by ProductName

having sum(QuantitySold\*PricePerUnit) > 500;

Result:

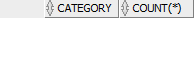


* Retrieve the total number of sales transactions for each category, but only for categories that have more than 2 distinct products sold.

select Category, count(\*)

from Sales Group by Category

having count(distinct ProductName) > 2;

Result:

* Find the average quantity sold for each product, but only for products where the total quantity sold exceeds 8 units.

select ProductName, avg(QuantitySold)

from Sales Group by ProductName

having sum(QuantitySold) > 8;

Result:



* For each category, retrieve the total number of products sold and the total revenue, but only for categories where the total number of products sold is greater than 6.

select Category, sum(QuantitySold), sum(QuantitySold \* PricePerUnit)

from Sales Group by Category

having sum(QuantitySold) > 6;

Result:

