Jio Institute

GROUP 4
1.Yuvraj Singh Srinet 25PGAI0019
2.Prajwal Wagh 25PGAI0109
3.Piyush Sunil Borse 25PGAI0026
4.Prateek Majumder 25PGAI0027
5.Bhawana Thawarani 25PGAI0137

Databases and Data Warehouses Assignment

1. John is a database administrator for a small retail company. The company stores informationabout its products in a database table named Products. The table has the following structure:

ProductID	ProductName	Category	Price	Stock
1	Laptop	Electronics	900	20
2	Smartphone	Electronics	700	50
3	Desk Chair	Furniture	150	15
4	Coffee Table	Furniture	120	10
5	Headphones	Electronics	100	35
6	Monitor	Electronics	200	25
7	Sofa	Furniture	500	5
8	Bookshelf	Furniture	80	8

The Productstable contains columns for the product ID, product name, category, price, and stock. John needs to perform several queries to extract specific information for his report.

Questions:

- Select Query: Write an SQL query to select the names and prices of all products in the Electronicscategory.

```
-> SELECT product_name, price
    FROM Products
    WHERE category = 'Electronics';
```

- Where and Limit Query: Write an SQL query to find the top 3 most expensive products inthe Furniturecategory.

```
-> SELECT product_name
   FROM Products
   WHERE category = 'Furniture'
   ORDER BY price desc
   LIMIT 3;
```

- Sort Query: Write an SQL query to list all products, sorted by their stock in descendingorder

```
-> SELECT product_name
   FROM Products
   ORDER BY Stock desc;
```

2. Imagine you are a data analyst working for a small retail company. The company has a database that stores sales data in a table called sales. This table records information about each sale transaction made by the company. Below is a simplified version of the salestable:

sale_id	product_name	quantity	price_per_unit	sale_date
1	Laptop	2	1000	2024-06-01
2	Mouse	5	20	2024-06-02
3	Keyboard	3	50	2024-06-03
4	Monitor	1	200	2024-06-04
5	Laptop	1	1000	2024-06-05
6	Mouse	10	20	2024-06-06
7	Laptop	3	1000	2024-06-07
8	Keyboard	2	50	2024-06-08
9	Monitor	2	200	2024-06-09
10	Laptop	1	1000	2024-06-10

Using this table, answer the following questions:

Questions:

- Count the Total Number of Sales Transactions: Write an SQL query to count the totalnumber of sales transactions recorded in the salestable.
  -> SELECT COUNT(*)
  FROM Sales;

- Calculate the Average Quantity Sold per Transaction: Write an SQL query to calculatethe average quantity of products sold per transaction.
  -> SELECT AVG(quantity)
  FROM Sales;

- Find the Maximum and Minimum Price per Unit: Write an SQL query to find themaximum and minimum price per unit of all products sold.
  -> SELECT MAX(price_per_unit), MIN(price_per_unit)
  FROM Sales;

3. John is a data analyst at a retail company. His job involves analyzing sales data to help the company make better business decisions. The company's sales data is stored in a database, which John can query using SQL (Structured Query Language).

John is working with a table named Salesthat contains the following data:

SaleID	Product	Quantity	Price	Date
1	Laptop	2	1200	2024-06-01
2	Mouse	5	20	2024-06-03
3	Keyboard	3	45	2024-06-05
4	Monitor	1	300	2024-06-07
5	Laptop	1	1200	2024-06-09

John needs to extract specific information from this table using SQL queries. He often uses logical and bitwise operators in his queries to filter and manipulate the data.

Questions:

- Using Logical Operators Write an SQL query to find all sales of Laptopthat were soldon or after 2024-06-01and the quantity sold is greater than or equal to 1.
  -> SELECT *

```
FROM Sales
WHERE Product = 'Laptop'
AND Date >= '2024-06-01'
AND quantity > 1;
```

- Using Bitwise Operators Write an SQL query to check if the SaleIDis odd. (Hint: Usethe bitwise AND operator).

```
-> SELECT *
   FROM Sales
  WHERE SaleID & 1 = 1;
```

- Combined Query Write an SQL query to find the total revenue from all sales of Laptopwhere the quantity sold is greater than 1. (Revenue is calculated as Quantity * Price).

```
-> SELECT SUM(Quantity*Price) as Revenue
   FROM Sales
   WHERE Product = 'Laptop'
   AND Quantity>1;
```

4. John is a database administrator at a retail company. The company stores its sales data in aSQL database. Recently, the management has asked John to prepare a few reports to analyzethe sales performance. The sales data is stored in a table called Sales with the following structure:

SaleID	ProductName	Quantity	SaleDate	TotalAmount	CustomerID	Region
1	Laptop	2	2024-06-01	2000.00	101	North
2	Smartphone	5	2024-06-02	2500.00	102	South
3	Tablet	1	2024-06-03	500.00	103	East
4	Headphones	10	2024-06-04	1000.00	104	West
5	Laptop	1	2024-06-05	1000.00	105	North

Questions:

- Write a SQL query to retrieve the details of all sales where the product name containsthe string 'top'.

```
-> SELECT *
   FROM Sales
   WHERE ProductName LIKE '%top%';
```

- Write a SQL query to calculate the total sales amount for sales made in the 'North'region.

```
-> SELECT SUM(TotalAmount) as Total_Sales_North_Region
   FROM Sales
   WHERE Region = 'North';
```

- Write a SQL query to identify the sales where the quantity sold is greater than 1 and thetotal amount is greater than $500.

```
-> SELECT *
   FROM Sales
   WHERE TotalAmount > 500
   AND Quantity>1
```

5. John works as a database administrator for a company that sells various products online. Hisjob involves managing and querying the company's database to generate reports for the sales and inventory departments. Recently, John was asked to create some reports based on customer orders and product details. Below is the information from two tables in the company'sdatabase:

Customers Table:

CustomerID	CustomerName	Country
1	Alice	USA
2	Bob	Canada
3	Charlie	UK
4	David	USA

Orders Table:

OrderID	CustomerID	ProductName	OrderDate
101	1	Laptop	2023-05-01
102	2	Smartphone	2023-05-03
103	1	Tablet	2023-05-04
104	3	Headphones	2023-05-05
105	4	Monitor	2023-05-07
106	5	Keyboard	2023-05-09

Questions:

- Inner Join: Write an SQL query to list all the orders along with the customer names whoplaced those orders.

```
-> SELECT o.OrderID, o.CustomerID, o.ProductName, o.OrderDate,
     c.CustomerName
   FROM Orders o inner join Customers c
   ON o.CustomerID = c.CustomerID
```

- Left Join: Write an SQL query to list all customers and their orders, including thosecustomers who have not placed any orders.

```
-> SELECT c.CustomerName, o.OrderID, o.ProductName, o.OrderDate
   FROM Customers c left  join Orders o
   ON c.CustomerID = o.CustomerID
```

- Right Join: Write an SQL query to list all orders and the customers who placed them, including those orders which do not have a corresponding customer in the Customerstable.

```
-> SELECT c.CustomerName, o.OrderID, o.ProductName, o.OrderDate
   FROM Customers c right  join Orders o
   ON c.CustomerID = o.CustomerID
```

6. Company XYZ is a growing e-commerce business that specializes in selling electronic gadgets. The company maintains a database to manage information about its products andsales. Below is an example of two tables from their database: Productsand Sales.

Products Table:

ProductID	ProductName	Category	Price
1	Smartphone	Mobile	300
2	Laptop	Computer	800
3	Tablet	Mobile	200
4	Headphones	Accessory	50

Sales Table:

SaleID	ProductID	Quantity	SaleDate
101	1	2	2024-01-15
102	3	1	2024-01-16
103	2	3	2024-01-17
104	4	5	2024-01-18

Questions:

- Cross Join Question: Write an SQL query to produce a cross join of the Products tablewith itself, listing each possible pair of products. The result should include the ProductIDand ProductNamefrom both instances of the Productstable.

```
-> SELECT p1.ProductID, p1.ProductName, p2.ProductID, p2.ProductName
   FROM Products A
   CROSS JOIN Products B
```

- Self Join Question: Write an SQL query using a self join on the Salestable to find pairsof sales that happened on the same day. The result should include the SaleID of both sales and the SaleDate.

```
-> SELECT s1.SaleID as Sale_id1, s2.SaleID as Sale_id2, s1.SaleDate
   FROM Sales1 s1
   JOIN Sales s2
   ON s1.SaleDate == s2.SaleDate
   AND s1.SaleID != s2.SaleID
```

- General SQL Query: Write an SQL query to find the total quantity of products sold foreach category. The result should include the Categoryand the total Quantity.

```
-> SELECT p.Category, SUM(s.Quantity) as TotalQuantity
```

```
FROM Products p
JOIN Sales s
ON p.ProductID== s.ProductID
GROUP BY p.Category
```

7. You work as a data analyst at a retail company that sells products both online and in physicalstores. The company collects data on sales transactions, including information about products, customers, and the stores where the sales occur. The sales data is stored in a database with the following tables:

Tables

Products

ProductID	ProductName	Category	Price
1	Laptop	Electronics	1000
2	Smartphone	Electronics	700
3	Headphones	Accessories	100
4	T-shirt	Apparel	20
5	Jeans	Apparel	50

Customers

CustomerID	CustomerName	Gender	Age
1	John Doe	Male	28
2	Jane Smith	Female	34
3	Emily Johnson	Female	22
4	Michael Brown	Male	45
5	Sarah Davis	Female	29

Stores

StoreID	StoreName	Location
1	Downtown	City Center
2	Uptown	Suburbs
3	East Side	East End
4	West Side	West End

Sales

SaleID	ProductID	CustomerID	StoreID	Date	Quantity	TotalAmount
1	1	1	1	2024-01-15	1	1000
2	2	2	2	2024-01-16	2	1400
3	3	3	3	2024-01-17	1	100
4	4	4	1	2024-01-18	3	60
5	5	5	2	2024-01-19	2	100

Questions:

- Group By and Aggregate Functions: Write an SQL query to find the total sales amountfor each product category. The result should include the category and the total sales amount.

```
-> SELECT p.Category, SUM(s.Quantity) as TotalQuantity
   FROM Products p
   JOIN Sales s
   ON p.ProductID== s.ProductID
   GROUP BY p.Category
```

- HAVING Clause: Write an SQL query to find the stores that have sold more than 1,000 dollars in total sales. The result should include the store name and the total sales amount.

```
-> SELECT st.storeName, SUM(sa.Quantity*sa.TotalAmount) as TotalSales
   FROM Stores st
   JOIN Sales sa
   ON st.SoreID = sa.StoreID
   GROUP BY st.StoreID
   HAVING SUM(sa.Quantity*sa.TotalAmount) > 1000
```

- Subqueries and Nested Queries: Write an SQL query to find the names of customerswho bought products from the "Electronics" category. The result should include the customer names only.

```
-> SELECT CustomerName
   FROM Customers
   WHERE CustomerID
   IN
   (     SELECT sa.CustomerID
         FROM Sales sa
         JOIN Products p
         ON sa.ProductID = p.ProductID
         WHERE p.Category = 'Electronics'
   )
```

8. Managing a Library Database

The city library has a database to keep track of the books, members, and borrowing records. The database consists of several tables, but for simplicity, we'll focus on the Books and Memberstables. Here's a brief description of each:

- Books: This table contains information about each book in the library.
- Members: This table contains information about each library member.

The structure of the tables is as follows:

Books Table

BookID	Title	Author	YearPublished	Genre
1	To Kill a Mockingbird	Harper Lee	1960	Fiction
2	1984	George Orwell	1949	Dystopian
3	The Great Gatsby	F. Scott Fitzgerald	1925	Fiction
4	The Catcher in the Rye	J.D. Salinger	1951	Fiction
5	Moby Dick	Herman Melville	1851	Adventure

Members Table

MemberID	Name	JoinDate	MembershipType
1	John Doe	2021-01-15	Regular
2	Jane Smith	2022-03-22	Premium
3	Emily Johnson	2021-07-09	Regular
4	Michael Brown	2022-05-18	Premium
5	Sarah Davis	2021-12-30	Regular

Questions:

- INSERT INTO: The library has acquired a new book, "The Hobbit" by J.R.R. Tolkien, published in 1937, and it is classified under the "Fantasy" genre. Write the SQL query toinsert this new book into the Bookstable.

```
-> INSERT INTO Books (Title, Author, YearPublication, Genre)
   VALUES ('The Hobbit', 'J.R.R. Tolkien', 1937, 'Fantasy');
```

- UPDATE: The member John Doe has upgraded his membership to "Premium" on the date "2023-06-01". Write the SQL query to update his membership type and join date inthe Memberstable.

```
-> UPDATE Members
   SET MembershipType = 'Premium', JoinDate = '2023-06-01'
   WHERE Name = 'John Doe';
```

- DELETE: The library is discarding old books and has decided to remove the book "MobyDick" from the Bookstable. Write the SQL query to delete this book from the table.

```
-> DELETE FROM Books
   WHERE BookTitle = 'Moby Dick';
```

9. ABC Retail Inc. is a company that sells a wide variety of products through its online store. The company maintains a database to manage its sales and customer information. Below is asimplified version of their Salestable:

SaleID	ProductName	Category	Quantity	PricePerUnit	SaleDate	CustomerID
1	Laptop	Electronics	2	800	2024-01-15	101
2	Headphones	Electronics	1	150	2024-02-03	102
3	Desk Chair	Furniture	5	120	2024-02-10	103
4	Monitor	Electronics	3	300	2024-03-05	104
5	Coffee Table	Furniture	2	200	2024-03-20	105
6	Notebook	Stationery	10	5	2024-03-25	106
7	Lamp	Furniture	4	50	2024-04-01	107

Questions:

- Calculate Total Sales per Category: Write an SQL query to calculate the total sales (sumof Quantity * PricePerUnit) for each category. Your result should include only those categories where the total sales are greater than $500.

```
->  SELECT Category, SUM(Quantity*PricePerUnit) as TotalSales
    FROM Sales
    GROUP BY category
    HAVING SUM(Quantity*PricePerUnit) > 500
```

- Classify Sales as High or Low: Write an SQL query using a CASE statement to classifyeach sale as 'High' if the total sale amount (Quantity * PricePerUnit) is greater than $1000, otherwise classify it as 'Low'. The result should include SaleID, ProductName, TotalSaleAmount, and SaleClassification.

```
->  SELECT Category, SUM(Quantity*PricePerUnit) as TotalSales
    FROM Sales
    GROUP BY category
    HAVING SUM(Quantity*PricePerUnit) > 500
```

- Monthly Sales Summary: Write an SQL query to summarize the total quantity sold and the total revenue for each month. The result should include YearMonth, TotalQuantitySold, and TotalRevenue.

```
-> SELECT MONTH(SaleDate) as YearMonth,
   SUM(Quantity) as TotalQuantitySold,
   SUM(Quantity*PricePerUnit) as TotalRevenue
   FROM Sales
   GROUP BY MONTH(SaleDate)
   HAVING SUM(Quantity*PricePerUnit) > 500
```

10. In a university setting, managing student information efficiently is crucial for ensuring smoothacademic operations. The university's database administrators are tasked with creating and maintaining a robust database system that adheres to normalization principles and ensures dataintegrity through various constraints.

Consider the following table structure for managing student information:

Table: Students
Column
StudentID
Name
Department
AdmissionYear
AdvisorID

Table: Advisors
Column
AdvisorID
Name
Department

Questions:

- Explain why it is important to define a primary key (PK) for the Studentstable. How does a primary key ensure data integrity?
  -> A primary key is a unique identifier for each record in a database table. it ensures that each student is uniquely identified which prevents duplicates, it creates indexes for each record.
  Other tables can refer primary key of a table through Foreign Key

- Describe the concept of Foreign Key (FK) with respect to the Studentsand Advisors tables. Provide an example scenario where a Foreign Key constraint would be beneficial.
  ->A foreign key is a column or a set of columns in one table that uniquely identifies a row of another table. It establishes a link between the data in the two tables.

- Discuss the significance of database normalization in the context of managing student information. Explain briefly what each of the following normalization forms (1NF, 2NF,3NF) aims to achieve.
  -> Database normalization is the process of structuring a relational database to reduce data redundancy and improve data integrity. It involves organizing the columns and tables of a database to ensure that dependencies are properly enforced by database integrity constraints.

11. In an online store specializing in electronics, managing orders efficiently is crucial to ensurecustomer satisfaction and operational smoothness. The store maintains a database to handle orders, which involves storing customer details, order information, and inventory management. Here's a simplified schema of the database tables:

Table: Customers

customer_id	customer_name	email	phone
1	John Doe	john@example.com	123-456-7890
2	Jane Smith	jane@example.com	987-654-3210
...	...	...	...

Table: Orders

order_id	customer_id	order_date	total_amount
101	1	2024-06-20 10:15:00	250.00
102	2	2024-06-21 15:30:00	150.00
...	...	...	...

Table: Order_Items

order_item_id	order_id	product_id	quantity	unit_price
1	101	1	2	100.00
2	101	2	1	50.00
...	...	...	...	...

Questions:

- Explain the concept of transactions in SQL and give an example related to the scenario.
  -> A transaction in SQL is a sequence of one or more SQL operations executed as a single unit. It follows the ACID properties (Atomicity, Consistency, Isolation, Durability).

  Ex.
  BEGIN TRANSACTION;

```
UPDATE Inventory SET Quantity = Quantity - 1 WHERE ProductID = 1;
INSERT INTO Orders (OrderID, ProductID, Quantity) VALUES (1001, 1, 1);

COMMIT;
```

- Describe the concept of locking mechanisms in databases and how they relate totransaction isolation levels.

-> Locking mechanisms prevent concurrent transactions from interfering with each other. Locks can be placed on rows, pages, or tables.

- Discuss the ACID properties of transactions and their significance in maintaining dataintegrity.

-> **Atomicity**: Ensures that all operations within a transaction are completed; if not, the transaction is aborted.

Consistency: Ensures that the database moves from one valid state to another.

Isolation: Ensures that concurrent transactions do not affect each other.

Durability: Ensures that once a transaction is committed, it remains so, even in the event of a system failure.

12. You are working as a data analyst at a retail company that manages multiple stores acrossdifferent regions. The company wants to analyze sales data to optimize inventory managementand marketing strategies. As part of your role, you are tasked with querying a database to retrieve relevant information.

Table Data Example:

Consider the following table named sales_datathat stores information about sales transactions:

transaction_id	store_id	product_id	sale_date	quant
1	101	1	2023-05-01	5
2	102	2	2023-05-02	3
3	101	3	2023-05-03	2
4	103	1	2023-05-04	4
5	102	2	2023-05-05	1

Questions:

- Query to Calculate Total Sales Revenue:

Write an SQL query to calculate the total revenue generated from sales for a specificstore in the year 2023. Assume you need to retrieve this information for Store ID 102.

```
-> SELECT SUM(quantity*unit_price) as TotalRevenue
   FROM sales_data
   WHERE YEAR(sale_date) = 2023
   AND store_id
```

- Create Procedure to Update Sales Data:
  Create an SQL procedure named update_sales_quantity that updates the quantity of a specific product sold based on its transaction ID. The procedure should take two parameters: transaction_idand new_quantity.

```
-> CREATE PROCEDURE update_sales_quantity
   (IN transaction_id INT, IN new_quantity INT)
   BEGIN
      UPDATE sales_data
      SET quantity = new_quantity
      WHERE transaction_id = transaction_id;
   END;
```

- Create Function to Calculate Average Unit Price:
  Create an SQL function named avg_unit_pricethat calculates the average unit price of products sold by a specific store. The function should take one parameter: store_id.

```
CREATE FUNCTION avg_unit_price (store_id INT) RETURNS D   DECIMAL(10, 2)
BEGIN
   DECLARE avg_price DECIMAL(10, 2);
   SELECT AVG(unit_price) INTO avg_price
   FROM sales_data
   WHERE store_id = store_id;
   RETURN avg_price;
END;
```

13. In a modern workplace, managing employee data securely is crucial for organizational efficiency and data protection. Companies often rely on robust database systems to handle sensitive information about their employees, such as personal details, salary information, and performance reviews. This passage explores how database management systems (DBMS) playa pivotal role in securely storing and managing employee data.

Consider the following simplified table schema for an employee database:

Table: employees
emp_id (int, PK)
emp_name (varchar(50))
emp_email (varchar(100))
emp_salary (decimal(10,2))

Questions:

- Roles and Permissions
  In the context of database management, explain what roles and permissions are and how they ensure data security.
  Roles and permissions in a database are mechanisms to control access and ensure data security. Roles are collections of permissions that can be assigned to users or groups, while permissions define specific access rights (e.g., read, write, delete) to database objects.

- SQL Query Creation
  Using the provided table schema for employees, write an SQL query to retrieve thenames and email addresses of all employees from the database.
  SELECT emp_name, emp_email
  FROM employees;

- Data Encryption
  Discuss the importance of data encryption in safeguarding sensitive employee information stored in a database.
  Data encryption is crucial for safeguarding sensitive employee information stored in a database. It converts data into a code to prevent unauthorized access, ensuring that even if the data is intercepted or accessed unlawfully, it remains unreadable without the decryption key.

14. You are working as a data analyst for a retail company that has a large database of sales transactions. The database includes information about products, customers, and transactions. Your task is to extract meaningful insights from this data to help the company improve its salesstrategy. Below is an example of the salestable in the database:

sale_id	customer_id	product_id	sale_date	quantity	sale_amount
1	101	501	2023-01-15	2	50.00
2	102	502	2023-01-16	1	30.00
3	101	503	2023-01-17	5	100.00
4	103	501	2023-02-20	3	75.00
5	104	504	2023-02-25	1	20.00
6	105	501	2023-03-05	4	100.00
7	101	502	2023-03-10	2	60.00
8	102	503	2023-03-15	3	90.00
9	104	501	2023-04-01	1	25.00
10	105	504	2023-04-05	2	40.00

Questions:

- Identify High-Spending Customers: Write an SQL query to find the customers who havespent more than $100 in total. Display their customer_id and the total amount they have spent, sorted by the total amount in descending order.

```
SELECT  Customer_id, SUM(quantity*sale_amount) as TotalAmountSpent
FROM Sales
GROUP BY Customer_id
HAVING SUM(quantity*sale_amount) > 100
ORDER BY TotalAmountSpent DESC
```

- Product Sales within a Date Range: Write an SQL query to list all product_ids that were sold between 2023-02-01and 2023-03-31. Each product should be listed only once. Display the product_idand the number of times each product was sold, sorted by the number of times sold in descending order.

```
SELECT product_id,SUM(quantity) as NumberOfTimesSold
FROM sales
WHERE sale_date BETWEEN '2023-02-01' AND '2023-03-31'
GROUP BY product_id
ORDER BY NumberOfTimesSold DESC
```

- Top Selling Product: Write an SQL query to identify the product that has been sold the most in terms of quantity. Display the product_idand the total quantity sold. Limit theresult to one product.

```
SELECT product_id,SUM(quantity) as TotalQuantitySold
FROM sales
GROUP BY product_id
ORDER BY NumberOfTimesSold DESC
LIMIT 1
```

15. You are a data analyst at a retail company that specializes in electronics. The company hasbeen gathering data on its sales transactions and product inventory to optimize its operations. You have been given access to the Salesand Productstables in the company's database togenerate insights for the upcoming quarterly report.

`Sales` Table

SaleID	ProductID	Quantity	SaleDate
1	101	2	2024-01-10
2	102	1	2024-01-12
3	101	1	2024-01-15
4	103	3	2024-02-01
5	104	4	2024-02-10
6	101	5	2024-02-15
7	105	2	2024-03-01
8	103	1	2024-03-05

`Products` Table

ProductID	ProductName	Category
101	Smartphone	Electronics
102	Laptop	Electronics
103	Tablet	Electronics
104	Smartwatch	Electronics
105	Headphones	Accessories

Questions:

- Calculate the total revenue generated from sales of each product. List the product names along with their respective revenues.

```
SELECT p.ProductName,SUM(s.SaleAmount) as TotalSales
FROM Product p
JOIN Sales s
ON p.ProductID = s.ProductID
GROUP BY s.ProductID
```

- Identify the month in which the highest number of products were sold. Provide the monthand the total quantity of products sold.

```
SELECT MONTH(SaleDate) as Month , SUM(Quantity) as TotalQuantity
FROM Sales
GROUP BY Month
ORDER BY TotalQuantity DESC
LIMIT 1
```

- Find the average sale amount for transactions involving more than 2 products.

```
SELECT AVG(SaleAmount) as avgSales
FROM Sales
WHERE Quantity > 2
```

16. The XYZ Retail Company manages a large inventory of products in its database. Each product is identified by a unique ProductID and categorized by various attributes including Category, Price, StockQuantity, SupplierID, and DiscontinuedStatus. The database is used totrack product availability, supplier information, and manage orders efficiently.

ProductID	Category	Price	StockQuantity	SupplierID
101	Electronics	299.99	50	201
102	Home & Kitchen	79.99	0	202
103	Electronics	199.99	30	201
104	Sports	49.99	100	203
105	Home & Kitchen	99.99	20	202
106	Electronics	89.99	5	204

Questions:

- Complex Filtering with Logical Operators
  Write a SQL query to find all products in the 'Electronics' category that are not discontinued and have a stock quantity greater than 10.

```
SELECT *
FROM products
WHERE Category =  "Electronics"
AND DiscontinuedStatus = 0
AND StockQuantity > 10
```

- Utilizing Bitwise Operators
  The DiscontinuedStatus column uses a bit to represent if the product is discontinued (1) or not (0). Write a SQL query that lists the ProductIDand Category of all discontinued products using a bitwise operator.

```
SELECT ProductID, Category
FROM products
WHERE DiscontinuedStatus & 1 = 1
```

- Advanced Combination Query
  Write a SQL query to find all products supplied by SupplierID 202 or SupplierID 204 that are either discontinued or have a price less than $100. Use both logical and bitwise operators in your query.

```
SELECT *
FROM products
WHERE (SupplierID = 202 OR SupplierID 204)
AND (DiscontinuedStatus & 1 = 1 OR Price<100)
```

17. The operations team at a mid-sized e-commerce company is tasked with optimizing the inventory management system. The company sells various products, and each product has anassociated category, price, and stock quantity. The operations team needs to generate reportsand queries to help make decisions about restocking and pricing strategies.

The product data is stored in a SQL database table named Products. The table has thefollowing schema:

```sql
CREATE TABLE Products (
    ProductID INT PRIMARY KEY,
    ProductName VARCHAR(100),
    Category VARCHAR(50),
    Price DECIMAL(10, 2),
    StockQuantity INT,
    Discontinued BIT
);
```

ProductID	ProductName	Category	Price	StockQuantity	Discontinued
1	Wireless Mouse	Electronics	25.99	100	0
2	Mechanical Keyboard	Electronics	75.50	50	0
3	Office Chair	Furniture	150.00	20	0
4	Standing Desk	Furniture	300.00	10	0
5	LED Monitor	Electronics	200.00	5	1
6	Water Bottle	Kitchen	10.00	200	0
7	Coffee Maker	Kitchen	50.00	30	1
8	Desk Lamp	Electronics	40.00	75	0
9	Bookshelf	Furniture	80.00	15	0
10	Blender	Kitchen	60.00	25	0

Questions:

- Logical and Bitwise Operators: Write an SQL query to find products in the Electronicscategory with a price less than $100 and a stock quantity greater than 50, but only if they are not discontinued. Your query should use both logical (AND, OR,etc.) and bitwise operators.

```sql
SELECT *
FROM Products
WHERE Category = Electronics
AND Price < 100
AND StockQuantity > 50
AND Discontinued = 0
```

- Advanced Filtering: Write an SQL query to find products that belong to either the Kitchen or Furniture category and have a stock quantity between 10 and 50, inclusive. Additionally, ensure that the results do not include any discontinued products.Explain how the BETWEENoperator is used in your query.

```
SELECT *
FROM Products
WHERE (Category = 'Kitchen'
OR Category = 'Furniture')
AND StockQuantity BETWEEN 10 AND 50
AND Discontinued = 0
```

BETWEEN selects values in given range, it includes both start and end in this example our range would be [10,11,12,.....,50] will include all products which has staock quantity in this range.

- Aggregation and Conditionals: Write an SQL query to calculate the total stock quantity and the average price of products for each category. Ensure that discontinued products are excluded from this calculation. Explain how the GROUP BY clause helps in generating the required output.

```
SELECT SUM(StockQuantity) as TotalStockQuantity,
AVG(Price) as AveragePrice
FROM Products
GROUP BY Category
AND Discontinued = 0
```

Group By clause helps to perform operations based on certain grouping criteria, as in this example, GROUP BY will group all products with their categories and we get to use aggregate functions on top of them.

18. You are a data analyst at a retail company that sells a variety of products online. Your company maintains a database to store information about products, sales, customers, andsuppliers. The company is expanding its data analysis capabilities and you are tasked withgenerating insightful queries from the database.

ProductID	ProductName	Category	Price	SupplierID
1	Smartphone	Electronics	699	101
2	Laptop	Electronics	999	102
3	Coffee Maker	Home	49	103
4	Blender	Home	99	103
5	Desk	Furniture	199	104

Table: Sales

SaleID	ProductID	CustomerID	Quantity	SaleDate
1	1	201	2	2024-01-15
2	2	202	1	2024-01-20
3	3	203	5	2024-01-25
4	1	204	1	2024-02-10
5	4	201	3	2024-02-15

CustomerID	CustomerName	Country
201	Alice	USA
202	Bob	Canada
203	Charlie	USA
204	Dave	UK

Table: Suppliers

SupplierID	SupplierName	ContactNumber
101	TechCorp	123-456-7890
102	GadgetWorks	234-567-8901
103	HomeGoods Inc.	345-678-9012
104	OfficeSupplies Co	456-789-0123

Questions:

- Using SQL Operators, filter and list the names and prices of products that belong to either the 'Electronics' or 'Furniture' category and have a price greater than $500.

```
SELECT ProductName, Price
FROM Products
WHERE Category = "Electronics"
AND Category = "Furniture"
AND Price > 500
```

- Write an SQL query to calculate the total sales quantity for each product and display theproduct name along with the total quantity sold. Sort the results in descending order of total quantity sold.

```
SELECT p.ProductName, SUM(s.Quantity) as TotalQuantity
FROM Products p
JOIN Sales s
ON p.ProductID = s.ProductID
ORDER BY TotalQuantity DESC
```

- Identify the names of customers who have purchased all products from the 'Home' category. Use SQL operators and joins to achieve this.

```
SELECT c.CustomerName
FROM Customers c
JOIN Sales s
ON c.CustomerID = s.CustomerID
JOIN Products p
ON s.ProductID = p.ProductID
WHERE p.Category = "Home"
```

19. You are working as a data analyst for a retail company. The company has a database thatstores information about products, sales transactions, and customers. Your task is to analyze the data to gain insights and improve business operations. Below is a sample of the tables in the database:

Table: `Products`

ProductID	ProductName	Category	Price	StockQuantity
1	Laptop	Electronics	1000.00	50
2	Smartphone	Electronics	500.00	150
3	Coffee Maker	Home & Kitchen	80.00	200
4	Blender	Home & Kitchen	60.00	300
5	Television	Electronics	800.00	30

Table: `Sales`

SaleID	ProductID	CustomerID	SaleDate	Quantity	TotalAmount
1	1	100	2023-01-10	1	1000.00
2	2	101	2023-01-15	2	1000.00
3	3	102	2023-01-20	1	80.00
4	4	103	2023-02-05	3	180.00
5	1	104	2023-02-10	2	2000.00

Table: `Customers`

CustomerID	FirstName	LastName	Email	Age	LoyaltyPoints
100	John	Doe	john.doe@example.com	28	150
101	Jane	Smith	jane.smith@example.com	34	200
102	Alice	Johnson	alice.j@example.com	45	120
103	Bob	Brown	bob.brown@example.com	23	80
104	Carol	Miller	carol.miller@example.com	37	250

Questions:

- String Operations: Write an SQL query to concatenate the first name and last name of customers and create a new column called FullName. Additionally, filter the results to include only those customers whose email addresses contain the domain "example.com".

```
SELECT CONCAT(FirstName,' ',LastName) as FullName
FROM Customers
WHERE Email LIKE '%example.com'
```

- Bitwise Operator: Assume that the LoyaltyPoints field in the Customers table is stored as a binary number. Write an SQL query to find all customers whose LoyaltyPointswhen ANDed with the binary number 100000(32 in decimal) result ina non-zero value.

```
SELECT *

FROM customers

WHERE LoyaltyPoints & 32 != 0;
```

- Aggregate Functions and Grouping: Write an SQL query to calculate the total revenue generated by each product category. The result should display the Categoryand theTotalRevenue.

```
SELECT p.Category , SUM(s.quantity*TotalAmount) as TotalRevenue
FROM Products p
LEFT JOIN Sales s
ON p.ProductID = s.ProductID
GROUP BY p.Category
```

20. In recent years, retail stores have increasingly relied on data analysis to improve customersatisfaction and operational efficiency. One such store, ShopEase, collects customer feedbackthrough surveys and records the responses in a database. This feedback data is critical for understanding customer preferences, identifying areas for improvement, and making data- driven decisions.

The database consists of a table named CustomerFeedbackwith the following structure:

FeedbackID	CustomerName	FeedbackDate	FeedbackText	Rating
1	John Doe	2024-01-15	The service was excellent	5
2	Jane Smith	2024-02-01	Product quality is good but expensive	3
3	Emily Jones	2024-02-15	Not happy with the product quality	2
4	Michael Brown	2024-03-10	Great prices but slow service	4
5	Sarah White	2024-03-25	Excellent quality and fast delivery	5

Questions:

- Identify Positive Feedback: Write an SQL query to select all records where the feedbackcontains the word 'excellent'. This will help the store identify the most positive feedback comments.
```
SELECT *
FROM CustomerFeedback
WHERE FeedBackText LIKE '&excellent&'
```

- Count Feedback by Keyword: Write an SQL query to count how many feedback entries contain the word 'quality'. This will help the store determine the frequency of feedback related to product quality.
```
SELECT Count(*) as QualityWordCount
FROM CustomerFeedback
WHERE FeedBackText LIKE '&quality&'
```

- Feedback Rating Analysis: Write an SQL query to find feedback entries where the customer mentions the word 'service' and the rating is below 4. This will help the storefocus on improving service aspects that customers are not satisfied with.
```
SELECT *
FROM CustomerFeedback
WHERE FeedBackText LIKE '&service&'
AND Rating<4
```

21. You are working as a database analyst for a retail company. The company maintains a database to manage information about its products, orders, and customers. There are threeprimary tables in this database:

Customers Table:

CustomerID	CustomerName	CustomerEmail
1	Alice Smith	alice@example.com
2	Bob Johnson	bob@example.com
3	Charlie Davis	charlie@example.com

Orders Table:

OrderID	CustomerID	OrderDate	OrderTotal
101	1	2024-06-01	150.00
102	2	2024-06-02	200.00
103	1	2024-06-03	50.00

Products Table:

ProductID	ProductName	ProductPrice
201	Laptop	1000.00
202	Mouse	20.00
203	Keyboard	50.00

OrderDetails Table:

OrderID	ProductID	Quantity
101	201	1
101	202	3
102	203	2
103	202	1

Questions:

- Inner Join: Write a query to find the names and email addresses of customers who have placed at least one order, along with the total amount of their orders.

```
SELECT c.CustomerName, c.CustomerEmail
FROM Customers c
JOIN Orders o
ON c.CustomerID = o.CustomerID
WHERE o.OrderID IS NOT NULL
GROUP BY c.CustomerName, c.CustomerEmail
```

- Left Join: Write a query to list all customers, including those who have not placed any orders. For each customer, include the total number of orders they have placed (if any).

```
SELECT c.CustomerName, c.CustomerEmail,      COALESCE(SUM(od.Quantity),
0) AS TotalQuantityOrdered
FROM Customers c
LEFT JOIN Orders o ON c.CustomerID = o.CustomerID
LEFT JOIN OrderDetails od ON o.OrderID = od.OrderID
GROUP BY c.CustomerName, c.CustomerEmail;
```

- Right Join: Write a query to list all products and their respective order quantities, including products that have never been ordered.

```
SELECT p.ProductID, p.ProductName, SUM(od.Quantity) as OrderQuantity
FROM OrderDetails od
RIGHT JOIN Products p
ON od.ProductID = p.ProductID
```

22. TechCorp is a rapidly growing technology company that sells various products. They have adatabase that tracks sales transactions, customer information, and product details. The database includes the following tables:

Sales

sale_id	product_id	customer_id	sale_date	quantity
1	101	201	2023-01-15	2
2	102	202	2023-01-17	1
3	101	203	2023-01-20	1
4	103	201	2023-01-25	5
5	102	204	2023-02-01	3

Customers

customer_id	customer_name	customer_
201	Alice	New York
202	Bob	Los Angele
203	Charlie	Chicago
204	David	Houston

Products

product_id	product_name	product_category
101	Laptop	Electronics
102	Phone	Electronics
103	Tablet	Electronics

Questions:

- Identify the customer who has spent the most money in total on their purchases. Providethe customer name and the total amount spent.

```
SELECT c.customer_name, SUM(s.total_amount) as TotalSpent
FROM Sales s
JOIN Customers c
ON s.customer_id = c.customer_id
GROUP BY s.customer_id
ORDER BY TotalSpent DESC
LIMIT 1
```

- Find the total number of units sold and the total revenue generated for each product category, but only for categories where more than 5 units were sold.

```
SELECT p.product_category, SUM(s.quantity) as TotalQuantity,
SUM(s.total_amount) as TotalRevenue
FROM Products p
JOIN Sales s
ON p.product_id = s.product_id
GROUP BY s.product_id
HAVING TotalQuantity>5
```

- List the products along with their total sales amount, but only for products whose totalsales amount exceeds the average sales amount of all products.

```
SELECT p.product_id, p.product_name,
SUM(s.total_amount) as totalSales
FROM Products p
JOIN Sales s
ON p.product_id = s.product_id
GROUP BY p.product_id
HAVING totalSales >
(SELECT AVG(total_amount) FROM Sales)
```

23. In a multinational retail corporation, the data analytics team is tasked with analyzing sales performance across various regions. The team uses SQL to query and process large datasets from their database. Below is a snapshot of their sales data for the past quarter:

Transaction_ID	Region	Product	Units_Sold
1	North	Laptop	50
2	South	Smartphone	100
3	East	Tablet	80
4	West	Laptop	70
5	North	Smartphone	120
6	South	Tablet	60
7	East	Laptop	55
8	West	Smartphone	90
9	North	Tablet	75
10	South	Laptop	65
11	East	Smartphone	110
12	West	Tablet	85

Questions:

- Query 1 (GROUP BY and HAVING): The management wants to identify regions wherethe average revenue per unit sold exceeds $500. Write an SQL query to find these regions along with the average revenue per unit sold.

```
SELECT Region,
AVG(Revenue/Units_Sold) as averageRevenuePerUnit
FROM Sales_data
```

```
GROUP BY Region
HAVING averageRevenuePerUnit>500
```

- Query 2 (Self Join): There is a need to compare the sales performance of each region with every other region in terms of total units sold. Construct an SQL query using a self-join to list all possible pairs of regions along with their total units sold.

```
SELECT A.Region as region1, B.Region AS region2,
A.total_units_sold as total_units_sold1,
B.total_units_sold as total_units_sold2
FROM
(SELECT region, SUM(quantity) AS total_units_sold
FROM sales_data GROUP BY region) AS A
JOIN
(SELECT region, SUM(quantity) AS total_units_sold
FROM sales_data GROUP BY region) AS B
ON A.region != B.region;
```

- Query 3 (Full Outer Join): The company is planning to launch a new product and wants to understand which regions have not contributed any sales in the last quarter. Create an SQL query using a full outer join to list all regions that have not recorded any sales, along with a count of products sold (which should show as NULL for these regions).

```
SELECT Region, SUM(Units_Sold) as TotalUnitsSold
FROM sales_data
GROUP BY region
```

24. XYZ Corp is a multinational company that deals with the sales of electronic gadgets across various regions. The company has a database that stores sales data for different products. Each product sale is recorded with details such as the product name, the quantity sold, the saleamount, the region, and the sale date.

SaleID	ProductName	QuantitySold	SaleAmount	Region
1	Laptop	5	2500	North
2	Smartphone	10	3000	South
3	Tablet	3	900	East
4	Laptop	2	1000	West
5	Smartphone	7	2100	North
6	Tablet	6	1800	South
7	Laptop	8	4000	East
8	Smartphone	9	2700	West
9	Tablet	4	1200	North
10	Laptop	10	5000	South

Questions

- Calculate the total sales amount for each product. Include a column that specifies whether the total sales amount is 'High' if it exceeds $10,000, 'Medium' if it is between
$5,000 and $10,000, and 'Low' if it is less than $5,000.

```
SELECT ProductName, SUM(QuantitySold *SaleAmount) as TotalSale
        CASE
                WHEN TotalSale > 10000
                THEN 'High'
                WHEN TotalSale BETWEEN 5000 AND 10000
                THEN 'Medium'
                ELSE 'Low'
        END AS sales_classification
FROM sales_data
GROUP BY ProductName
```

- Find the product that has the highest average sale amount per unit sold across all regions.

```
SELECT ProductName,
AVG(SUM(SaleAmount)/SUM(QuantitySold)) as AvgAmtPerUnitSold
FROM sales_data
GROUP BY ProductName
ORDER BY AvgAmtPerUnitSold DESC
LIMIT 1
```

- Generate a report that shows the total quantity sold for each product in each region. If a product has not been sold in a particular region, include that region with a quantity of 0.

```
SELECT product_name, region, SUM(quantity) AS total_quantity_sold
FROM sales_data
GROUP BY product_name, region
```

25. Company ABC is a mid-sized firm that specializes in software development. The companyhas a diverse team of employees working on various projects. Below is a sample of the employeestable in the company's database:

employee_id	name	department	salary	experience	performance_rating
1	Alice	IT	95000	5	A
2	Bob	HR	60000	3	B
3	Charlie	IT	110000	7	B
4	David	Marketing	80000	4	C
5	Eve	IT	115000	6	A
6	Frank	HR	62000	3	C
7	Grace	Marketing	95000	5	B
8	Heidi	IT	70000	2	B
9	Ivan	HR	67000	4	A
10	Judy	IT	80000	3	C

Questions:

- Performance-Based Salary Bonus Calculation: Create an SQL query to calculate the bonus for each employee based on their performance rating. The bonus rules are as follows:
    - Performance Rating 'A': 10% of salary
    - Performance Rating 'B': 5% of salary
    - Performance Rating 'C': No bonus

  Add a column bonus in the result set to display the calculated bonus.

```
SELECT
   employee_id,
   name,
   salary,
   performance_rating,
   CASE
      WHEN performance_rating = 'A' THEN salary * 0.10
      WHEN performance_rating = 'B' THEN salary * 0.05
      ELSE 0
   END AS bonus
FROM
   employees;
```

- Experience and Department Based Salary Adjustment: Write an SQL query to adjust the salaries of employees. The adjustment rules are:

    - If the employee is in the IT department and has more than 5 years of experience, increase the salary by 8%.
    - If the employee is in the HR department and has less than or equal to 3 years of experience, increase the salary by 5%.
    - Otherwise, increase the salary by 2%.

  Display the employee_id, name, original salary, and the adjusted_salary.

```
SELECT employee_id, salary,
CASE
        WHEN department = 'IT' AND experience>5 THEN salary*1.08
        WHEN department = 'HR' AND experience<=3 THEN salary*1.05
        ELSE salary*1.02
END as adjusted_salary
FROM employees
```

- Advanced Conditional Query for Performance Analysis: Create an SQL query that classifies employees into three categories based on their salary and performance rating:
    - 'High Performer' if the salary is above 90000 and performance rating is 'A'.
    - 'Potential for Improvement' if the salary is between 70000 and 90000 (inclusive) and performance rating is 'B'.
    - 'Needs Improvement' for all others.

  Display the employee_id, name, salary, performance_rating, and the

category.

```
SELECT employee_id,name, salary, performance_rating
CASE
        WHEN salary>90000 AND performance_rating='A'
        THEN 'High Performer'
        WHEN salary BETWEEN 70000 AND 90000
        AND performance_rating='B' THEN 'Potential For Improvement'
        ELSE 'Needs Improvement'
END as category
FROM employees
```

26. XYZ Company maintains a database to manage their employee information, including their working hours and projects. Below is a simplified schema of the relevant tables in the database:

EmployeeID	Name	Department	Position	Salary
1	John Doe	IT	Developer	70000
2	Jane Smith	HR	Manager	80000
3	Alice Brown	IT	Developer	75000
4	Bob White	IT	Analyst	60000
5	Carol Black	Sales	Salesperson	65000

Table: Projects

ProjectID	ProjectName	Budget
101	Website Overhaul	200000
102	HR System Update	100000
103	Market Analysis	150000

Table: WorkHours

EmployeeID	ProjectID	HoursWorked	Date
1	101	6	2024-01-15
1	101	5	2024-01-16
2	102	8	2024-01-15
3	101	7	2024-01-15
4	103	8	2024-01-16
5	103	6	2024-01-15

Questions:

- Create a SQL Procedure: Write a SQL procedure named UpdateEmployeeSalary that takes an EmployeeID and a NewSalary as parameters. The procedure should update the salary of the specified employee. If the EmployeeIDdoes not exist, it should raise an error.

```
CREATE PROCEDURE UpdateEmployeeSalary
(IN EmployeeID INT, IN NewSalary DECIMAL(10, 2))
BEGIN
    DECLARE empCount INT;
    SELECT COUNT(*) INTO empCount
    FROM Employees
    WHERE EmployeeID = EmployeeID;

    IF empCount = 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'EmployeeID does not exist';
    ELSE
        UPDATE Employees
        SET Salary = NewSalary
        WHERE EmployeeID = EmployeeID;
    END IF;
END
```

- Create a SQL Function: Write a SQL function named CalculateTotalHours that takes an EmployeeIDand returns the total number of hours worked by that

employee across all projects. The function should return 0 if the employee has not logged any hours.

```
CREATE FUNCTION calculateTotalHours (EmployeeID int)        RETURNS int
BEGIN
DECLARE HoursWorkedByEmp int
SELECT COALESCE(SUM(w.HoursWorked),0)
INTO HoursWorkedByEmp
FROM WorkHours w JOIN Employee e ON w.employeeID = e.employeeID
RETURN HoursWorkedByEmp;
END;
```

- Complex Query: Write a complex SQL query to find the names of employees who have worked on more than one project. The query should return the Name and the Countof distinct projects they have worked on.

```
SELECT e.Name, Count(ProjectID) as DistinctProjectCnt
FROM Employee e
JOIN WorkHours w
ON e.EmployeeID = w.EmployeeID
GROUP BY w.EmployeeID
Having DistinctProjectCnt > 1
```

27. An online bookstore maintains a database to manage its inventory, customers, and orders.The database consists of the following tables: Books, Customers, Orders, and OrderDetails. The Books table stores information about each book available in the store. The Customers table holds customer details. The Orders table keeps track of each order placed by customers, and the OrderDetails table records the specific books and quantities for each order.

Books Table:

BookID	Title	Author	Price	Stock
1	SQL Fundamentals	John Smith	29.99	10
2	Advanced SQL	Jane Doe	39.99	5
3	Database Design	Alice Johnson	49.99	2
4	Data Science	Michael Brown	59.99	0

Customers Table:

CustomerID	Name	Email
1	Bob Green	bob@example.com
2	Alice Blue	alice@example.com
3	John White	john@example.com

Orders Table:

OrderID	CustomerID	OrderDate
1	1	2024-06-20
2	2	2024-06-21
3	3	2024-06-22

OrderDetails Table:

OrderDetailID	OrderID	BookID	Quantity
1	1	1	2
2	1	3	1
3	2	2	1
4	3	4	1

Questions:

- Advanced SQL Query Write an SQL query to find the total revenue generated from all orders. The query should calculate the total revenue by summing up the product of the price and quantity for each book in each order.

```
SELECT o.OrderID, SUM(od.Quantity*b.Price) revenueByOrder
FROM Orders o
JOIN OrderDetails od
ON o.OrderID = od.OrderID
JOIN Books b
ON od.BookID = b.BookID
```

- Create a Stored Procedure Create a stored procedure named UpdateStockthat takesBookID and Quantity as parameters. This procedure should decrement the stock ofthe specified book by the given quantity. If the stock falls below zero, the procedure should raise an error.

```
CREATE PROCEDURE UpdateStock (IN BookID INT, IN Quantity INT)
BEGIN
  DECLARE NewStock INT;
  UPDATE Books
  SET Stock = Stock - DecrementQuantity
  WHERE BookID = BookID;
  SELECT Stock INTO NewStock FROM Books WHERE BookID = BookID;
  IF NewStock < 0 THEN
  SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Insufficient Stock!';
  END IF;
  END;
```

- Create a User-Defined Function Create a user-defined function named GetCustomerOrderCountthat takes a CustomerIDas input and returns the total number of orders placed by that customer.

```
CREATE FUNCTION GetCustomerOrderCount

(CustomerID INT)

RETURNS INT

BEGIN

  DECLARE OrderCount INT;

  SELECT COUNT(*) AS OrderCount FROM Orders WHERE CustomerID =
CustomerID;

  RETURN OrderCount;

END;
```

28. ABC Corporation is a technology company that provides a wide range of services and products. The company keeps track of its product sales, customer interactions, and employeeinformation in a relational database. The database contains several tables, including Products, Sales, Customers, and Employees.

Table: Products

ProductID	ProductName	Category	Price
1	Smartphone	Electronics	700
2	Laptop	Electronics	1200
3	Office Chair	Furniture	150
4	Desk	Furniture	300
5	Smartwatch	Electronics	250

Table: Sales

SaleID	ProductID	CustomerID	SaleDate	Quantity	TotalAmount
101	1	1001	2023-01-15	2	1400
102	2	1002	2023-02-20	1	1200
103	3	1003	2023-03-18	3	450
104	1	1001	2023-04-10	1	700
105	4	1004	2023-05-25	2	600
106	5	1005	2023-06-30	1	250

Table: Customers

CustomerID	CustomerName	City
1001	John Doe	New York
1002	Jane Smith	Los Angeles
1003	Bob Johnson	Chicago
1004	Alice Brown	Houston
1005	Charlie Davis	Phoenix

Table: Employees

EmployeeID	EmployeeName	Department	Salary
201	Michael Scott	Sales	75000
202	Dwight Schrute	Sales	65000
203	Jim Halpert	Sales	72000
204	Pam Beesly	Human Resources	50000
205	Angela Martin	Accounting	55000

Questions:

- Advanced Subquery: Write an SQL query to find the names of customers who havepurchased products from the 'Furniture' category.

```
SELECT c.CustomerName
    FROM Customers c
    WHERE c.CustomerID IN (
      SELECT DISTINCT s.CustomerID
      FROM Sales s
      INNER JOIN Products p ON s.ProductID = p.ProductID
      WHERE p.Category = 'Furniture'
    );
```

- Correlated Subquery: Write an SQL query to find the employees who earn more thanthe average salary of their respective departments.

```
SELECT e.EmployeeName, e.Salary
    FROM Employees e
    WHERE e.Salary > (
      SELECT AVG(Salary)
      FROM Employees
      WHERE DepartmentID = e.DepartmentID
    );
```

- Complex Join and Aggregation: Write an SQL query to find the total sales amount foreach city, including the city name and the total sales amount. Sort the result by the total sales amount in descending order.

```
SELECT c.CityName, SUM(s.SaleAmount) AS TotalSales
    FROM Customers c
    INNER JOIN Sales s ON c.CustomerID = s.CustomerID
    GROUP BY c.CityName
    ORDER BY TotalSales DESC;
```

29. You are a data analyst at TechMart, a multinational technology retailer. Your manager has tasked you with analyzing the sales data to extract meaningful insights that can help in strategic decision-making. The database contains various tables, but for this task, you'll primarily focus on the Sales and Products tables. The Sales table records every sale made in different regions,and the Products table holds information about the products sold.

Table: Customers

CustomerID	CustomerName	City
1001	John Doe	New York
1002	Jane Smith	Los Angeles
1003	Bob Johnson	Chicago
1004	Alice Brown	Houston
1005	Charlie Davis	Phoenix

Table: Employees

EmployeeID	EmployeeName	Department	Salary
201	Michael Scott	Sales	75000
202	Dwight Schrute	Sales	65000
203	Jim Halpert	Sales	72000
204	Pam Beesly	Human Resources	50000
205	Angela Martin	Accounting	55000

Questions:

- Find the total revenue generated in each region.
  Write an SQL query to calculate the total revenue for each region. Revenue is calculatedby multiplying the quantity sold by the unit price for each sale.

```
SELECT Region, SUM(Sales.Quantity * Sales.UnitPrice) AS TotalRevenue
FROM Sales
INNER JOIN Products ON Sales.ProductID = Products.ProductID
GROUP BY Region;
```

- Identify the products that have been sold more than the average quantity of all products.Write an SQL query to list the product_id and product_name of products whose total quantity sold is more than the average quantity sold of all products.

```
SELECT p.ProductID, p.ProductName
```

```
FROM Products p
WHERE p.QuantitySold > (
SELECT AVG(s.Quantity) AS AverageQuantity
FROM Sales s);
```

- List the regions where the sales revenue for 'Electronics' category products is higherthan the overall average sales revenue of all regions.
  Write an SQL query to find the regions where the sales revenue from 'Electronics' products exceeds the average sales revenue across all regions. Ensure your queryconsiders the necessary joins and calculations.

```
SELECT Region, SUM(Sales.Quantity * Sales.UnitPrice) AS ElectronicsRevenue
FROM Sales
INNER JOIN Products ON Sales.ProductID = Products.ProductID
WHERE Products.Category = 'Electronics'
GROUP BY Region
HAVING SUM(Sales.Quantity * Sales.UnitPrice) > (
SELECT AVG(SUM(s.Quantity * s.UnitPrice)) AS AvgSalesRevenue
FROM Sales s
INNER JOIN Products p ON s.ProductID = p.ProductID
GROUP BY Region
);
```