1.What does SQL stand for?

a) Structured Question Language

b) Structured Query Language

c) Sequential Query Language

d) Standard Query Language

2.Which SQL command is used to retrieve data from a database?

a) INSERT

b) UPDATE

c) SELECT

d) DELETE

3.Which clause is used to filter records in SQL?

a) WHERE

b) ORDER BY

c) GROUP BY

d) HAVING

4.Which SQL keyword is used to remove duplicate records from a result set?

a) DELETE

b) DISTINCT

c) UNIQUE

d) REMOVE

5.Which SQL clause is used to sort the results of a query?

a) ORDER BY

b) GROUP BY

c) SORT

d) HAVING

6.What will happen if the WHERE clause is omitted from an SQL DELETE statement?

a) It will delete all records

b) It will delete only the first record

c) It will throw an error

d) It will delete no records

7.What is the default sorting order of the ORDER BY clause?

a) DESC

b) ASC

c) RANDOM

d) NONE

8.Which of the following statements is used to remove a table in SQL?

a) DELETE TABLE

b) DROP TABLE

c) REMOVE TABLE

d) TRUNCATE TABLE

9.Which SQL statement is used to add a new record to a table?

a) ADD

b) INSERT INTO

c) APPEND

d) UPDATE

10.What is the purpose of the SQL GROUP BY clause?

a) To filter data

b) To aggregate data

c) To sort data

d) To delete data

11.Which SQL function is used to count the number of records?

a) COUNT()

b) NUMBER()

c) TOTAL()

d) SUM()

12.What is the purpose of the HAVING clause in SQL?

a) To filter grouped records

b) To filter rows before grouping

c) To sort data

d) To delete duplicate records

13.Which SQL statement is used to change existing records in a table?

a) MODIFY

b) UPDATE

c) CHANGE

d) EDIT

14.What is the function of the SQL INNER JOIN?

a) Returns all records from both tables

b) Returns matching records from both tables

c) Returns records from the first table only

d) Returns records from the second table only

15.Which SQL command is used to remove all records from a table but keeps its structure?

a) DELETE

b) DROP

c) TRUNCATE

d) REMOVE

16.What does the SQL UNION operator do?

a) Joins tables

b) Combines result sets without duplicates

c) Filters data

d) Sorts data17.Which wildcard is used in SQL for matching a single character?

a) %

b) _

c) *

d) #

18.What is the result of the following SQL statement?

SELECT COUNT(*) FROM Customers WHERE Age > 30;

a) Total number of customers

b) Number of customers older than 30

c) Total number of age values

d) Average age of customers

19.Which operator is used to select values within a given range in SQL?

a) BETWEEN

b) IN

c) LIKE

d) RANGE

20.What does the SQL NOT NULL constraint do?

a) Ensures a column has unique values

b) Prevents a column from having NULL values

c) Prevents duplicate records

d) Allows NULL values in a column

21. Which SQL statement creates a new table?

a) ADD TABLE

b) MAKE TABLE

c) CREATE TABLE

d) NEW TABLE

22. In SQL, what is a primary key?

a) A unique identifier for a record

b) A foreign key in another table

c) A required column

d) A duplicate key

23. What is the use of the SQL ALTER TABLE statement?

a) To add, modify, or delete columns

b) To delete the table

c) To create a new table

d) To retrieve data

24. What is the difference between DROP and TRUNCATE?

a) DROP removes table structure, TRUNCATE does not

b) TRUNCATE removes table structure, DROP does not

c) Both are the same

d) DROP deletes all records but keeps table

25. What type of JOIN returns all records when there is a match in one of the tables?

a) INNER JOIN

b) LEFT JOIN

c) RIGHT JOIN

d) FULL JOIN

What will be the output of:

26.SELECT 10 / 0;

26. a) 10

b) 0

c) Error

d) NULL

27. Which SQL keyword is used to define a default value for a column?

a) DEFAULT

b) SET

c) VALUE

d) ASSIGN

28. What does the SQL COALESCE function do?

a) Replaces NULL values

b) Counts NULL values

c) Deletes NULL values

d) Sorts NULL values29. In SQL, what is a foreign key?

a) A key in the same table

b) A key linking two tables

c) A key with only numeric values

d) A unique key

30. Which SQL clause is used to group rows sharing the same values?

a) GROUP BY

b) HAVING

c) ORDER BY

d) DISTINCT

31. What is an SQL subquery?

a) A query inside another query

b) A query that modifies table structure

c) A query that deletes records

d) A query that runs automatically

32. What is the difference between SQL and NoSQL databases?

a) SQL is schema-less, NoSQL has a fixed schema

b) SQL uses structured tables, NoSQL uses flexible formats

c) SQL does not support relationships, NoSQL does

d) SQL does not use queries, NoSQL does

33. How do you prevent SQL injection attacks?

a) Use raw SQL queries

b) Allow all user inputs

c) Use prepared statements and parameterized queries

d) Store passwords in plain text

34. What is an index in SQL used for?

a) To speed up query performance

b) To store table relationships

c) To delete data faster

d) To create backup copies

35. What is normalization in SQL?

a) Storing data redundantly for faster retrieval

b) Organizing data to reduce redundancy and improve integrity

c) A way to increase the size of a database

d) A process to randomly arrange data

36. What is denormalization?

a) Increasing redundancy for better performance

b) Removing duplicate recordsc) Converting data into relational format

d) Splitting a table into multiple smaller tables

37. Which of the following describes the ACID properties of a database?

a) Atomicity, Consistency, Isolation, Durability

b) Accuracy, Control, Integrity, Data

c) Aggregate, Count, Insert, Delete

d) Auto-commit, Consistency, Isolation, Dependency

38. What is a stored procedure in SQL?

a) A precompiled set of SQL statements

b) A type of query used only for reports

c) A temporary table storing results

d) A tool for database backups

39. What is a view in SQL?

a) A virtual table based on a query

b) A way to change the primary key of a table

c) A backup of a table

d) A table that stores indexes

40. What is the difference between a clustered and a non-clustered index?

a) A clustered index sorts and stores data physically, non-clustered does not

b) A clustered index is faster than a non-clustered index

c) A non-clustered index modifies the structure of a table

d) Clustered index is only used in primary keys

41. How can you optimize SQL queries?

a) Avoid using indexes

b) Use SELECT * in queries

c) Minimize the use of joins and subqueries

d) Use large text columns for filtering

42. What is the difference between UNION and UNION ALL?

a) UNION removes duplicates, UNION ALL includes duplicates

b) UNION ALL removes duplicates, UNION includes duplicates

c) UNION is used for joining tables, UNION ALL is not

d) UNION ALL is faster than UNION in all cases

43. What is a trigger in SQL?

a) A stored procedure executed automatically on a specific event

b) A primary key of a table

c) A type of foreign key

d) A manual SQL command

44. What is the purpose of the EXISTS clause?

a) To check if a subquery returns any records

b) To delete duplicate recordsc) To sort data in ascending order

d) To create a new table

45. What are common aggregate functions in SQL?

a) COUNT, SUM, AVG, MIN, MAX

b) SELECT, UPDATE, DELETE, INSERT

c) INNER JOIN, LEFT JOIN, RIGHT JOIN

d) CHAR, VARCHAR, INT, FLOAT

46. What is the difference between CHAR and VARCHAR?

a) CHAR has a fixed length, VARCHAR has a variable length

b) CHAR stores numbers, VARCHAR stores text

c) CHAR is for foreign keys, VARCHAR is for primary keys

d) CHAR is always faster than VARCHAR

47. How does the CASE statement work in SQL?

a) It is used for conditional logic inside queries

b) It deletes records

c) It is used to create a new table

d) It is an alternative to the WHERE clause

48. What is a CTE (Common Table Expression)?

a) A temporary result set used in queries

b) A type of stored procedure

c) A function that replaces NULL values

d) A way to define a foreign key

49. What is the difference between RANK() and DENSE_RANK()?

a) RANK() skips ranks when there are ties, DENSE_RANK() does not

b) RANK() gives sequential ranking without gaps

c) DENSE_RANK() assigns the same rank to all rows

d) Both functions return the same result

50. How do you handle NULL values in SQL queries?

a) Using IS NULL or IS NOT NULL

b) Using = NULL in conditions

c) Using DELETE to remove NULL values

d) Using ORDER BY NULL

51. What does the LEFT JOIN return?

a) Only matching records from both tables

b) All records from the left table and matching records from the right table

c) All records from the right table and matching records from the left table

d) Only unmatched records from both tables

52. What is the main purpose of the FOREIGN KEY constraint?

a) To enforce a relationship between two tables

b) To create a unique key for each recordc) To store duplicate records

d) To delete data permanently

53. What does the NVL() function do in SQL?

a) Replaces NULL values with a specified value

b) Counts the number of NULL values

c) Deletes NULL values

d) Sorts NULL values in descending order

54. What is the difference between DELETE and TRUNCATE?

a) DELETE removes specific rows, TRUNCATE removes all rows

b) DELETE is faster than TRUNCATE

c) TRUNCATE can be rolled back, DELETE cannot

d) TRUNCATE keeps records but clears the table structure

55. What does the RIGHT JOIN return?

a) Only matching records from both tables

b) All records from the right table and matching records from the left table

c) All records from the left table and matching records from the right table

d) Only unmatched records from both tables

56. How do you get the current date in SQL?

a) GETDATE()

b) CURDATE()

c) CURRENT_DATE

d) All of the above

57. What is the purpose of the DISTINCT keyword?

a) To remove duplicate rows from a query result

b) To filter records based on a condition

c) To delete records

d) To create a new column

58. Which SQL function is used to find the highest value in a column?

a) MAX()

b) MIN()

c) HIGH()

d) TOP()

59. What does the LIKE operator do in SQL?

a) Matches a specified pattern in a column

b) Joins two tables

c) Sorts the results

d) Removes duplicates

60. Which SQL statement is used to rename a column?

a) MODIFY COLUMN

b) CHANGE COLUMN

c) ALTER TABLE … RENAME COLUMN

d) RENAME TABLE

1.What does SQL stand for?

a) Structured Question Language

b) Structured Query Language

c) Sequential Query Language

d) Standard Query Language

2.Which SQL command is used to retrieve data from a database?

a) INSERT

b) UPDATE

c) SELECT

d) DELETE

3.Which clause is used to filter records in SQL?

a) WHERE

b) ORDER BY

c) GROUP BY

d) HAVING

4.Which SQL keyword is used to remove duplicate records from a result set?

a) DELETE

b) DISTINCT

c) UNIQUE

d) REMOVE

5.Which SQL clause is used to sort the results of a query?

a) ORDER BY

b) GROUP BY

c) SORT

d) HAVING

6.What will happen if the WHERE clause is omitted from an SQL DELETE statement?

a) It will delete all records

b) It will delete only the first record

c) It will throw an error

d) It will delete no records

7.What is the default sorting order of the ORDER BY clause?

a) DESC

b) ASC

c) RANDOM

d) NONE

8.Which of the following statements is used to remove a table in SQL?

a) DELETE TABLE

b) DROP TABLE

c) REMOVE TABLE

d) TRUNCATE TABLE

9.Which SQL statement is used to add a new record to a table?

a) ADD

b) INSERT INTO

c) APPEND

d) UPDATE

10.What is the purpose of the SQL GROUP BY clause?

a) To filter data

b) To aggregate data

c) To sort data

d) To delete data

11.Which SQL function is used to count the number of records?

a) COUNT()

b) NUMBER()

c) TOTAL()

d) SUM()

12.What is the purpose of the HAVING clause in SQL?

a) To filter grouped records

b) To filter rows before grouping

c) To sort data

d) To delete duplicate records

13.Which SQL statement is used to change existing records in a table?

a) MODIFY

b) UPDATE

c) CHANGE

d) EDIT

14.What is the function of the SQL INNER JOIN?

a) Returns all records from both tables

b) Returns matching records from both tables

c) Returns records from the first table only

d) Returns records from the second table only

15.Which SQL command is used to remove all records from a table but keeps its structure?

a) DELETE

b) DROP

c) TRUNCATE

d) REMOVE

16.What does the SQL UNION operator do?

a) Joins tables

b) Combines result sets without duplicates

c) Filters data

d) Sorts data17.Which wildcard is used in SQL for matching a single character?

a) %

b) _

c) *

d) #

18.What is the result of the following SQL statement?

SELECT COUNT(*) FROM Customers WHERE Age > 30;

a) Total number of customers

b) Number of customers older than 30

c) Total number of age values

d) Average age of customers

19.Which operator is used to select values within a given range in SQL?

a) BETWEEN

b) IN

c) LIKE

d) RANGE

20.What does the SQL NOT NULL constraint do?

a) Ensures a column has unique values

b) Prevents a column from having NULL values

c) Prevents duplicate records

d) Allows NULL values in a column

21. Which SQL statement creates a new table?

a) ADD TABLE

b) MAKE TABLE

c) CREATE TABLE

d) NEW TABLE

22. In SQL, what is a primary key?

a) A unique identifier for a record

b) A foreign key in another table

c) A required column

d) A duplicate key

23. What is the use of the SQL ALTER TABLE statement?

a) To add, modify, or delete columns

b) To delete the table

c) To create a new table

d) To retrieve data

24. What is the difference between DROP and TRUNCATE?

a) DROP removes table structure, TRUNCATE does not

b) TRUNCATE removes table structure, DROP does not

c) Both are the same

d) DROP deletes all records but keeps table

25. What type of JOIN returns all records when there is a match in one of the tables?

a) INNER JOIN

b) LEFT JOIN

c) RIGHT JOIN

d) FULL JOIN

What will be the output of:

26.SELECT 10 / 0;

26. a) 10

b) 0

c) Error

d) NULL

27. Which SQL keyword is used to define a default value for a column?

a) DEFAULT

b) SET

c) VALUE

d) ASSIGN

28. What does the SQL COALESCE function do?

a) Replaces NULL values

b) Counts NULL values

c) Deletes NULL values

d) Sorts NULL values29. In SQL, what is a foreign key?

a) A key in the same table

b) A key linking two tables

c) A key with only numeric values

d) A unique key

30. Which SQL clause is used to group rows sharing the same values?

a) GROUP BY

b) HAVING

c) ORDER BY

d) DISTINCT

31. What is an SQL subquery?

a) A query inside another query

b) A query that modifies table structure

c) A query that deletes records

d) A query that runs automatically

32. What is the difference between SQL and NoSQL databases?

a) SQL is schema-less, NoSQL has a fixed schema

b) SQL uses structured tables, NoSQL uses flexible formats

c) SQL does not support relationships, NoSQL does

d) SQL does not use queries, NoSQL does

33. How do you prevent SQL injection attacks?

a) Use raw SQL queries

b) Allow all user inputs

c) Use prepared statements and parameterized queries

d) Store passwords in plain text

34. What is an index in SQL used for?

a) To speed up query performance

b) To store table relationships

c) To delete data faster

d) To create backup copies

35. What is normalization in SQL?

a) Storing data redundantly for faster retrieval

b) Organizing data to reduce redundancy and improve integrity

c) A way to increase the size of a database

d) A process to randomly arrange data

36. What is denormalization?

a) Increasing redundancy for better performance

b) Removing duplicate recordsc) Converting data into relational format

d) Splitting a table into multiple smaller tables

37. Which of the following describes the ACID properties of a database?

a) Atomicity, Consistency, Isolation, Durability

b) Accuracy, Control, Integrity, Data

c) Aggregate, Count, Insert, Delete

d) Auto-commit, Consistency, Isolation, Dependency

38. What is a stored procedure in SQL?

a) A precompiled set of SQL statements

b) A type of query used only for reports

c) A temporary table storing results

d) A tool for database backups

39. What is a view in SQL?

a) A virtual table based on a query

b) A way to change the primary key of a table

c) A backup of a table

d) A table that stores indexes

40. What is the difference between a clustered and a non-clustered index?

a) A clustered index sorts and stores data physically, non-clustered does not

b) A clustered index is faster than a non-clustered index

c) A non-clustered index modifies the structure of a table

d) Clustered index is only used in primary keys

41. How can you optimize SQL queries?

a) Avoid using indexes

b) Use SELECT * in queries

c) Minimize the use of joins and subqueries

d) Use large text columns for filtering

42. What is the difference between UNION and UNION ALL?

a) UNION removes duplicates, UNION ALL includes duplicates

b) UNION ALL removes duplicates, UNION includes duplicates

c) UNION is used for joining tables, UNION ALL is not

d) UNION ALL is faster than UNION in all cases

43. What is a trigger in SQL?

a) A stored procedure executed automatically on a specific event

b) A primary key of a table

c) A type of foreign key

d) A manual SQL command

44. What is the purpose of the EXISTS clause?

a) To check if a subquery returns any records

b) To delete duplicate recordsc) To sort data in ascending order

d) To create a new table

45. What are common aggregate functions in SQL?

a) COUNT, SUM, AVG, MIN, MAX

b) SELECT, UPDATE, DELETE, INSERT

c) INNER JOIN, LEFT JOIN, RIGHT JOIN

d) CHAR, VARCHAR, INT, FLOAT

46. What is the difference between CHAR and VARCHAR?

a) CHAR has a fixed length, VARCHAR has a variable length

b) CHAR stores numbers, VARCHAR stores text

c) CHAR is for foreign keys, VARCHAR is for primary keys

d) CHAR is always faster than VARCHAR

47. How does the CASE statement work in SQL?

a) It is used for conditional logic inside queries

b) It deletes records

c) It is used to create a new table

d) It is an alternative to the WHERE clause

48. What is a CTE (Common Table Expression)?

a) A temporary result set used in queries

b) A type of stored procedure

c) A function that replaces NULL values

d) A way to define a foreign key

49. What is the difference between RANK() and DENSE_RANK()?

a) RANK() skips ranks when there are ties, DENSE_RANK() does not

b) RANK() gives sequential ranking without gaps

c) DENSE_RANK() assigns the same rank to all rows

d) Both functions return the same result

50. How do you handle NULL values in SQL queries?

a) Using IS NULL or IS NOT NULL

b) Using = NULL in conditions

c) Using DELETE to remove NULL values

d) Using ORDER BY NULL

51. What does the LEFT JOIN return?

a) Only matching records from both tables

b) All records from the left table and matching records from the right table

c) All records from the right table and matching records from the left table

d) Only unmatched records from both tables

52. What is the main purpose of the FOREIGN KEY constraint?

a) To enforce a relationship between two tables

b) To create a unique key for each recordc) To store duplicate records

d) To delete data permanently

53. What does the NVL() function do in SQL?

a) Replaces NULL values with a specified value

b) Counts the number of NULL values

c) Deletes NULL values

d) Sorts NULL values in descending order

54. What is the difference between DELETE and TRUNCATE?

a) DELETE removes specific rows, TRUNCATE removes all rows

b) DELETE is faster than TRUNCATE

c) TRUNCATE can be rolled back, DELETE cannot

d) TRUNCATE keeps records but clears the table structure

55. What does the RIGHT JOIN return?

a) Only matching records from both tables

b) All records from the right table and matching records from the left table

c) All records from the left table and matching records from the right table

d) Only unmatched records from both tables

56. How do you get the current date in SQL?

a) GETDATE()

b) CURDATE()

c) CURRENT_DATE

d) All of the above

57. What is the purpose of the DISTINCT keyword?

a) To remove duplicate rows from a query result

b) To filter records based on a condition

c) To delete records

d) To create a new column

58. Which SQL function is used to find the highest value in a column?

a) MAX()

b) MIN()

c) HIGH()

d) TOP()

59. What does the LIKE operator do in SQL?

a) Matches a specified pattern in a column

b) Joins two tables

c) Sorts the results

d) Removes duplicates

60. Which SQL statement is used to rename a column?

a) MODIFY COLUMN

b) CHANGE COLUMN

c) ALTER TABLE … RENAME COLUMN

d) RENAME TABLE

Test 02

```
1)SELECT department, COUNT(employee_id)
FROM employees;
```

```
Select department,count(employee_id)
From employee
Group by depertment;
```

```
2)SELECT department, COUNT(*)
FROM employees
WHERE COUNT(*) > 5
GROUP BY department;
```

```
 SELECT department, COUNT(*)
FROM employees
GROUP BY department
HAVING COUNT(*) > 5;
```

```
3)SELECT e.name, d.department_name
FROM employees e
JOIN departments d;
```

```
SELECT e.name, d.department_name
FROM employees e
JOIN departments d ON e.department_id = d.department_id;
```

```
4)INSERT INTO students (id, name, age)
VALUES (1, 'John');
```

```
INSERT INTO VALUES (id,name,age)
INSERT INTO VALUES (01,’JOHN’,28);
```

```
5)SELECT name, salary, salary * 0.1 AS bonus
FROM employees
```

```
 WHERE bonus > 50000;
SELECT name,salary,salary *0.1 as bonus
From employee
Having salary *0.1>5000;
```

```
6)SELECT name, age
FROM students
ORDER BY marks DESC;
SELECT name,age,mark
From students
Order by mark desc;
```

```
7)SELECT name, age
FROM students
WHERE age > (SELECT age FROM students);
SELECT name,age
From employee
Where age>(select max(age)from students);
```

```
8)SELECT DISTINCT name, COUNT(*)
FROM students
GROUP BY age;
SELECT DISTINCT name,age
From employee;
```

```
9)DELETE FROM students
WHERE students.name = teachers.name;
Delete students from employee student
```

```
John teachers on s.name =t.name;
```

```
10)SELECT name, COUNT(*)

FROM employees;

SELECT name COUNT (*)

From employee

Group by name;
```

1. Rank employees based on salary in the IT department

```
  SELECT Employee_ID, Name, Department, Salary,
  RANK() OVER (ORDER BY Salary DESC) AS Rank
   FROM employees
   WHERE Department = 'IT';
```

2. Find customers who never placed an order

```
SELECT c.Customer_ID, c.Name, COUNT(o.Order_ID) AS Orders_Placed
FROM customers c
LEFT JOIN orders o ON c.Customer_ID = o.Customer_ID
GROUP BY c.Customer_ID, c.Name
HAVING COUNT(o.Order_ID) = 0;
```

3. Get the second-highest salary

```
SELECT MAX(Salary) AS Salary
FROM employees
WHERE Salary < (SELECT MAX(Salary) FROM employees);
```

4. Calculate the running total of sales

```
    SELECT Order_ID, Customer_ID, Sale_Amount,
     SUM(Sale_Amount) OVER (ORDER BY Order_ID) AS Running_Total
      FROM sales;
```

5. Find products ordered more than 10 times

```
SELECT Product_Name, COUNT(*) AS Order_Count
FROM order_items
GROUP BY Product_Name
HAVING COUNT(*) > 10;
```

6. Retrieve users who logged in on consecutive days

```
SELECT User_ID, Login_Date
FROM (
   SELECT User_ID, Login_Date,
        LEAD(Login_Date) OVER (PARTITION BY User_ID ORDER BY Login_Date) AS Next_Login
   FROM user_logins
) AS SubqueryWHERE DATEDIFF(Next_Login, Login_Date) = 1;
```

7. Get the highest order value customer per city

```
SELECT City, Customer_ID, MAX(Order_Value) AS Order_Value

FROM customers c

JOIN orders o ON c.Customer_ID = o.Customer_ID

GROUP BY City, Customer_ID;
```

8. Find employees earning the same as their manager

```
SELECT e.Name AS Employee, e.Salary, m.Name AS Manager

FROM employees e

JOIN employees m ON e.Manager_ID = m.Employee_ID

WHERE e.Salary = m.Salary;
```

9. Find the most ordered product in each category

```
SELECT p.Category, p.Product_Name, COUNT(*) AS Orders

FROM order_items oi

JOIN products p ON oi.Product_ID = p.Product_ID

GROUP BY p.Category, p.Product_Name

HAVING COUNT(*) = (

    SELECT MAX(Order_Count)

    FROM (

        SELECT Product_Name, COUNT(*) AS Order_Count

        FROM order_items

        GROUP BY Product_Name

    ) AS Subquery

);
```

10. Retrieve customers who placed orders in every month of 2024

```
SELECT Customer_Name

FROM customers c

JOIN orders o ON c.Customer_ID = o.Customer_ID

WHERE YEAR(o.Order_Date) = 2024

GROUP BY c.Customer_Name
```

```
HAVING COUNT(DISTINCT MONTH(o.Order_Date)) = 12;
```

11. Find the highest-paid employee per department

```
SELECT Department, Name AS Employee, Salary
FROM employees e
WHERE Salary = (
    SELECT MAX(Salary)
    FROM employees
    WHERE Department = e.Department
);
```

12. Count customers who placed multiple orders

```
SELECT Customer_ID, COUNT(Order_ID) AS Order_Count
FROM orders
GROUP BY Customer_ID
HAVING COUNT(Order_ID) > 1;
```

13. Find the monthly growth rate of sales

```
    SELECT YEAR(Order_Date) AS Year, MONTH(Order_Date) AS Month,
   SUM(Sale_Amount) AS Sales,
  (SUM(Sale_Amount) - LAG(SUM(Sale_Amount)) OVER (ORDER BY YEAR(Order_Date), MONTH(Order_Date)))
       / LAG(SUM(Sale_Amount)) OVER (ORDER BY YEAR(Order_Date), MONTH(Order_Date)) * 100 AS Growth_Percent
FROM sales
GROUP BY YEAR(Order_Date), MONTH(Order_Date);
```

14. Get the top 3 products in terms of revenue

```
SELECT Product_Name, SUM(Order_Value) AS Revenue
FROM order_items oi
JOIN products p ON oi.Product_ID = p.Product_ID
GROUP BY Product_Name
ORDER BY Revenue DESC
LIMIT 3;
```

15. Find the first and last order date for each customer

```
SELECT Customer_ID,

  MIN(Order_Date) AS First_Order,

  MAX(Order_Date) AS Last_Order

FROM orders

GROUP BY Customer_ID;

WHERE TIMESTAMPDIFF(YEAR, Join_Date, CURDATE()) > 5;
```

16. Find employees who have worked for more than 5 years

```
SELECT Name AS Employee, Join_Date, TIMESTAMPDIFF(YEAR, Join_Date, CURDATE()) AS Years_Worked

FROM employees

WHERE TIMESTAMPDIFF(YEAR, Join_Date, CURDATE()) > 5;
```

17. Detect duplicate records in a table

```
SELECT Email, COUNT(*) AS Count

FROM users

GROUP BY Email

HAVING COUNT(*) > 1;
```

18. Calculate the cumulative sum of orders per customer

```
SELECT Customer_ID, Order_ID, Order_Value,

    SUM(Order_Value) OVER (PARTITION BY Customer_ID ORDER BY Order_ID) AS Cumulative_Sum
```

19. Find the average order value per category

```
SELECT Category, AVG(Order_Value) AS Avg_Order_Value

FROM products p

JOIN orders o ON p.Product_ID = o.Product_ID

GROUP BY Category;
```

20. Retrieve employees who earn more than the average salary

```
SELECT Name AS Employee, Salary

FROM employees

WHERE Salary > (SELECT AVG(Salary) FROM employees);
```

TEST 4

1.Scenario;

1. Movie Streaming Platform

You are designing a database for a movie streaming platform like Netflix. The platform allows users

to watch movies, rate them, and subscribe to different plans.

Requirements:

Users can create accounts, and each user has a subscription plan.

Movies belong to different genres and can have multiple actors.

Users can rate movies and add them to their watchlist.

Subscription plans determine what content a user can access.

Challenge:

Design the SQL database schema (tables, primary keys, foreign keys) for this system.

SOLUTION:

```
CREATE TABLE Users (
 user_id INT PRIMARY KEY AUTO_INCREMENT,
 username VARCHAR(50) UNIQUE NOT NULL,
 email VARCHAR(100) UNIQUE NOT NULL,
 password_hash VARCHAR(255) NOT NULL,
 subscription_id INT,
 FOREIGN KEY (subscription_id) REFERENCES Subscriptions(subscription_id)
);
CREATE TABLE Subscriptions (
 subscription_id INT PRIMARY KEY AUTO_INCREMENT,
 plan_name VARCHAR(50) UNIQUE NOT NULL,
 price DECIMAL(10,2) NOT NULL,
 content_access VARCHAR(255) NOT NULL -- Defines access level
);
```

```
CREATE TABLE Movies (
 movie_id INT PRIMARY KEY AUTO_INCREMENT,
 title VARCHAR(255) NOT NULL,
 release_year YEAR,
 duration INT, -- Duration in minutes
 description TEXT
);
CREATE TABLE Genres (
 genre_id INT PRIMARY KEY AUTO_INCREMENT,
 genre_name VARCHAR(50) UNIQUE NOT NULL
);
CREATE TABLE MovieGenres (
 movie_id INT,
 genre_id INT,
 PRIMARY KEY (movie_id, genre_id),
 FOREIGN KEY (movie_id) REFERENCES Movies(movie_id) ON DELETE CASCADE,
 FOREIGN KEY (genre_id) REFERENCES Genres(genre_id) ON DELETE CASCADE
);
CREATE TABLE Actors (
 actor_id INT PRIMARY KEY AUTO_INCREMENT,
 actor_name VARCHAR(100) NOT NULL
);
CREATE TABLE MovieActors (
 movie_id INT,
 actor_id INT,
 PRIMARY KEY (movie_id, actor_id),
 FOREIGN KEY (movie_id) REFERENCES Movies(movie_id) ON DELETE CASCADE,
 FOREIGN KEY (actor_id) REFERENCES Actors(actor_id) ON DELETE CASCADE
);
CREATE TABLE Ratings (
 rating_id INT PRIMARY KEY AUTO_INCREMENT,
 user_id INT,
 movie_id INT,
 rating DECIMAL(2,1) CHECK (rating BETWEEN 1.0 AND 5.0),
 review TEXT,
 FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE,
 FOREIGN KEY (movie_id) REFERENCES Movies(movie_id) ON DELETE CASCADE
);
CREATE TABLE Watchlist (
 user_id INT,
 movie_id INT,
 PRIMARY KEY (user_id, movie_id),
 FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE,
 FOREIGN KEY (movie_id) REFERENCES Movies(movie_id) ON DELETE CASCADE
);
```

2.Social Media Platform

Scenario:

You are designing a database for a social media platform like Instagram or Twitter, where users can

post, like, and follow others.

Requirements:

Users can create profiles and follow/unfollow other users.

Users can post text, images, and videos.

Posts can receive likes, comments, and shares.

Users can send direct messages to each other.

A notification system alerts users about new likes, comments, and follows.

SOLUTION:

```
-- Users table
CREATE TABLE Users (
 user_id INT PRIMARY KEY AUTO_INCREMENT,
 name VARCHAR(100) NOT NULL,
 email VARCHAR(100) UNIQUE NOT NULL,
 password_hash VARCHAR(255) NOT NULL
);
-- Followers table (Many-to-Many relationship for following users)
CREATE TABLE Followers (
 follower_id INT,
 following_id INT,
 PRIMARY KEY (follower_id, following_id),
 FOREIGN KEY (follower_id) REFERENCES Users(user_id),
 FOREIGN KEY (following_id) REFERENCES Users(user_id)
);
-- Posts table
CREATE TABLE Posts (
 post_id INT PRIMARY KEY AUTO_INCREMENT,
 user_id INT,
 content TEXT,
 media_url VARCHAR(255),
 created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
 FOREIGN KEY (user_id) REFERENCES Users(user_id)
);
-- Likes table
CREATE TABLE Likes (
 like_id INT PRIMARY KEY AUTO_INCREMENT,
 user_id INT,
 post_id INT,
 created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
 FOREIGN KEY (user_id) REFERENCES Users(user_id),
 FOREIGN KEY (post_id) REFERENCES Posts(post_id)
);
-- Comments table
CREATE TABLE Comments (
 comment_id INT PRIMARY KEY AUTO_INCREMENT,
 user_id INT,
 post_id INT,
 comment_text TEXT NOT NULL,
 created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
 FOREIGN KEY (user_id) REFERENCES Users(user_id),
 FOREIGN KEY (post_id) REFERENCES Posts(post_id)
);
-- Shares table
CREATE TABLE Shares (
 share_id INT PRIMARY KEY AUTO_INCREMENT,
 user_id INT,
 post_id INT,
 created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
 FOREIGN KEY (user_id) REFERENCES Users(user_id),
 FOREIGN KEY (post_id) REFERENCES Posts(post_id)
);
-- Messages table
CREATE TABLE Messages (
 message_id INT PRIMARY KEY AUTO_INCREMENT,
 sender_id INT,
 receiver_id INT,
 message_text TEXT NOT NULL,
 sent_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
 FOREIGN KEY (sender_id) REFERENCES Users(user_id),
 FOREIGN KEY (receiver_id) REFERENCES Users(user_id)
);
-- Notifications table
CREATE TABLE Notifications (
 notification_id INT PRIMARY KEY AUTO_INCREMENT,
 user_id INT,
 notification_text TEXT NOT NULL,
 created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
 FOREIGN KEY (user_id) REFERENCES Users(user_id)
);
```

3. Online Learning Platform

Scenario:

You are designing a database for an e-learning platform like Udemy or Coursera, where students

can enroll in courses, watch lessons, and receive certificates.

Requirements:

Instructors can create courses, and each course has multiple lessons.

Students can enroll in courses and track their progress.

Courses belong to different categories (Programming, Business, etc.).

Students receive certificates upon course completion.

Instructors and students can engage in discussions via a Q&A section.

SOLUTION:

```
-- Users table (Both Students and Instructors)
```

```sql
CREATE TABLE Users (
 user_id INT PRIMARY KEY AUTO_INCREMENT,
 name VARCHAR(100) NOT NULL,
 email VARCHAR(100) UNIQUE NOT NULL,
 password_hash VARCHAR(255) NOT NULL,
 role ENUM('Student', 'Instructor') NOT NULL
);
-- Courses table
CREATE TABLE Courses (
 course_id INT PRIMARY KEY AUTO_INCREMENT,
 title VARCHAR(255) NOT NULL,
 description TEXT,
 category VARCHAR(100),
 instructor_id INT,
 FOREIGN KEY (instructor_id) REFERENCES Users(user_id)
);
-- Lessons table
CREATE TABLE Lessons (
 lesson_id INT PRIMARY KEY AUTO_INCREMENT,
 course_id INT,
 title VARCHAR(255) NOT NULL,
 content TEXT,
 video_url VARCHAR(255),
 FOREIGN KEY (course_id) REFERENCES Courses(course_id)
);
-- Enrollments table (Many-to-Many between Users and Courses)
CREATE TABLE Enrollments (
 user_id INT,
 course_id INT,
 enrolled_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
 progress DECIMAL(5,2) DEFAULT 0.00,
 PRIMARY KEY (user_id, course_id),
 FOREIGN KEY (user_id) REFERENCES Users(user_id),
 FOREIGN KEY (course_id) REFERENCES Courses(course_id)
);
-- Certificates table
CREATE TABLE Certificates (
 certificate_id INT PRIMARY KEY AUTO_INCREMENT,
 user_id INT,
 course_id INT,
 issue_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
 FOREIGN KEY (user_id) REFERENCES Users(user_id),
 FOREIGN KEY (course_id) REFERENCES Courses(course_id)
);
-- Q&A table
CREATE TABLE QnA (
 question_id INT PRIMARY KEY AUTO_INCREMENT,
 user_id INT,
 course_id INT,
 question_text TEXT NOT NULL,
 created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
 FOREIGN KEY (user_id) REFERENCES Users(user_id),
 FOREIGN KEY (course_id) REFERENCES Courses(course_id)
);
-- Answers table
CREATE TABLE Answers (
 answer_id INT PRIMARY KEY AUTO_INCREMENT,
 question_id INT,
 user_id INT,
 answer_text TEXT NOT NULL,
 created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
 FOREIGN KEY (question_id) REFERENCES QnA(question_id),
 FOREIGN KEY (user_id) REFERENCES Users(user_id)
```

TEST 5

01) Find the average salary of employees in the Marketing department:

```
SELECT AVG(salary) AS average_salary

FROM employees

WHERE department = 'Marketing';
```

2. Find the names of employees who joined after 2019:

```
SELECT name

FROM employees

WHERE join_date > '2019-12-31';
```

3. Find the employee with the lowest salary:

```
SELECT name, salary

FROM employees

ORDER BY salary ASC

LIMIT 1;
```

4. Find the employees who do not belong to any department:

```
SELECT name

FROM employees

WHERE department IS NULL OR department = '';
```

5. Find the employees who have the same salary as their manager:

```
SELECT e.name

FROM employees e

JOIN employees m ON e.manager_id = m.employee_id

WHERE e.salary = m.salary;
```

6. Find the employee(s) who has/have been with the company for the longest time:

```
SELECT name, join_date

FROM employees

ORDER BY join_date ASC

LIMIT 1;
```

7. *Find the department(s) where the number of employees is less than 5*:

```
SELECT department

FROM employees

GROUP BY department

HAVING COUNT(employee_id) < 5;
```

8. *Find the employees who have duplicate names*:

```
SELECT name

FROM employees

GROUP BY name

HAVING COUNT(name) > 1;
```

9. *Find the department(s) with no employees*:

```
SELECT department

FROM departments

WHERE department NOT IN (SELECT DISTINCT department FROM employees);
```

10. *Find the employees who joined in the year 2020*:

```
SELECT name

FROM employees

WHERE YEAR(join_date) = 2020;
```