1.What is SQL?

Structured Query Language is a database tool which is used to create and access database to support software application.

2.What is Database

A place where we can store and retrieve the application data.

3.What DBMS?

A software which facilitates the storing and retrieval operation on the database.

Note- SQL is not case sensitive language

4.What is RDBMS?:

A DBMS bases on relation model is called relational database management system or RDBMS

RDBMS users' relational structures to store data. A relation is also called a table.

A relation contains tuples(rows) and attributes(columns)

Ex: *student(RollNo,Name,City) and marks relation(RollNo,Makrs)*

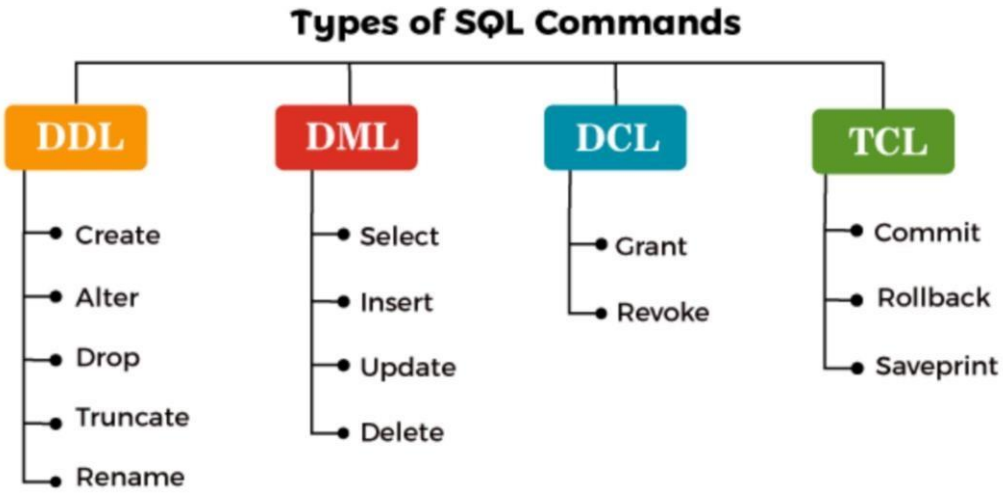
Relation are set using a common column between tables

in above example RollNo is common column.

5.Diffrent DBMS Application?

Oracle,Sql Server,My-SQL,SQL Lite,Access.

6.What are different types of statements supported by SQL?



1.DDL(Data Defination Language) It is used to define the database structure such as table. It includes tables such as create, Alter, drop and truncate.

1.Create:

The CREATE TABLE statement is used to create a new table in a database.

Syntax:

```
CREATE TABLE table_name (
    column1 datatype,
    column2 datatype,
    column3 datatype,
   ....
);
```

Example:

```
CREATE  TABLE
      Employee  ( EmpId
      int, FirstName
      char(20), Address
      char(20), City
      varchar(20)
);
```

Alter:

The ALTER TABLE statement is used to add, delete, or modify columns in an existing table.

The ALTER TABLE statement is also used to add and drop various constraints on an existing table.

ALTER TABLE - ADD Column

To add a column in a table, use the following syntax:

Syntax:

ALTER TABLE table_name
ADD column_name datatype;

Example:

ALTER TABLE Employee
ADD Email char(20);

ALTER TABLE - DROP COLUMN

To delete a column in a table, use the following syntax.

Syntax:

ALTER TABLE table_name
DROP COLUMN column_name;

Example:

ALTER TABLE Customers DROP
COLUMN Email;

ALTER TABLE - ALTER/MODIFY COLUMN

To change the data type of a column in a table, use the following syntax

Syntax:

ALTER TABLE table_name

ALTER COLUMN column_name datatype;

Example:

ALTER TABLE
Employee ADD
EmpId varchar(20);

CREATE DATABASE

The CREATE DATABASE statement is used to create a new SQL database.

Syntax:

CREATE DATABASE databasename;

Example:

CREATE DATABASE Test;

DROP

Drop Database:

The DROP DATABASE statement is used to drop an existing SQL database.

Syntax:

DROP DATABASE *databasename*;

Example:

DROP DATABASE testDB;

1. DROP TABLE

The DROP TABLE command deletes a table in the database.

Syntax:

DROP table tablename;

Example:

DROP table Employee;

TRUNCATE TABLE

The TRUNCATE TABLE command deletes the data inside a table, but not the table itself

Syntax:

Truncate table *tablename*;

Example:

Truncate table *Employee*;

Rename :

This command is used to rename the column and table name

1. Renaming the column:

By using this command, we can existing rename the column name

Syntax:

exec sp_rename 'Table_Name.Column_Name','NewColumn_Name','Column';

2. Renaming the Table:

By using this command, we can rename the existing table name

Syntax:

exec sp_rename 'Old_tableName','New_tableName';

2.DML(Data Manuplation Language):

DML is short name of Data Manipulation Language which deals with data manipulation and includes most common SQL statements such SELECT, INSERT, UPDATE, DELETE, etc., and it is used to store, modify, retrieve, delete and update data in a database.

1. SELECT:

retrieve data from a database

Syntax:

Select * from *tablename*;

Example:

Select * from Employee;

2. Insert:

insert data into a table.

Syntax:

INSERT INTO *table_name* (*column1*, *column2*, *column3*, ...)
VALUES (*value1*, *value2*, *value3*, ...);

Or

INSERT INTO table_name
VALUES (value1, value2, value3, ...);

Example:

INSERT INTO Customers(CustomerName, ContactNumber)
VALUES (‘Sachin’,867678788);

OR
INSERT INTO Customers VALUES (‘Sachin’,867678788);

UPDATE :

The UPDATE statement is used to modify the existing records in a table.

Syntax:

UPDATE *table_name*
SET *column1* = *value1*, *column2* = *value2*, ...
WHERE *condition*;

Example:

UPDATE Customers
SET Name = 'Sachin', City= ‘Sangli’
WHERE CustomerID = 1;

DELETE

The DELETE statement is used to delete existing records in a table.

Syntax:

DELETE FROM *table_name* WHERE *condition*; --delete specific record

Or

DELETE FROM *table_name*; ---delete all records

Example:

DELETE FROM *Employee* WHERE *EmpID=1*; --Specific record

Or

DELETE FROM *Employee;---ALL RECORDS IS DELETED*

DCL(Data Control Language);

DCL is short name of Data Control Language which includes commands such as GRANT and mostly concerned with rights, permissions and other controls of the database system.

- GRANT - allow users access privileges to the database
- REVOKE - withdraw users access privileges given by using the GRANT command

TCL:

TCL is short name of Transaction Control Language which deals with a transaction within a database.

- *COMMIT - commits a Transaction*
- *ROLLBACK - rollback a transaction in case of any error occurs*

1.BEGIN TRANSACTION:

To start the transaction

Syntax:

BEGIN TRANSACTION

<SQL STATEMENT>

2. COMMIT:

TO make the transaction is Permanent by explicitly or user.Once the operation is commit then we cannot rollback.

Syntax:

BEGIN TRANSACTION

<SQL STATEMENT>

Commit;

3. Rollback: *To cancel the transaction.*

Syntax:

BEGIN TRANSACTION

<SQL STATEMENT>

Rollback;

Aggregate Functions in sql:

In database management an aggregate function is a function where the values of multiple rows are grouped together as input on certain criteria to form a single value of more significant meaning.

Various Aggregate Functions:

1. Min()
2. Max()
3. Count()
4. Sum()
5. Avg()

1.Min():

The MIN() function returns the smallest value of the selected column.

Syntax:

SELECT MIN(*column_name*)
FROM *table_name*
WHERE *condition*;

2.Max():

The MAX() function returns the largest value of the selected column.

Syntax:

SELECT MAX(*column_name*)
FROM *table_name*
WHERE *condition*;

3.Count():
The COUNT() function returns the number of rows that matches a specified criterion.
Syntax:
SELECT COUNT(*column_name*)
FROM *table_name*
WHERE *condition*;

4.Sum():
The SUM() function returns the total sum of a numeric column.
Syntax:
SELECT SUM(*column_name*)
FROM *table_name*
WHERE *condition*;

5.Avg():
The AVG() function returns the average value of a numeric column.
Syntax:
SELECT AVG(*column_name*)
FROM *table_name*
WHERE *condition*;

SQL LIKE Operator:

The LIKE operator is used in a WHERE clause to search for a specified pattern in a column.

There are two wildcards often used in conjunction with the LIKE operator:

- The percent sign (%) represents zero, one, or multiple characters
- The underscore sign (_) represents one, single character

LIKE Syntax:
SELECT *column1, column2, ..*
FROM *table_name*
WHERE *columnN* LIKE *pattern*;
Example:

LIKE Operator	Description
WHERE CustomerName LIKE 'a%'	Finds any values that starts with "a"
WHERE CustomerName LIKE '%a'	Finds any values that ends with "a"
WHERE CustomerName LIKE '%or%'	Finds any values that have "or" in any position
WHERE CustomerName LIKE '_r%'	Finds any values that have "r" in the second position
WHERE CustomerName LIKE 'a__%'	Finds any values that starts with "a" and are at least 3 characters in length
WHERE ContactName LIKE 'a%o'	Finds any values that starts with "a" and ends with "o"

Constraint:

SQL constraints are used to specify rules for data in a table. Constraints can be specified when the table is created with the CREATE TABLE statement, or after the table is created with the ALTER TABLE statement.

Syntax:

CREATE TABLE *table_name* **(**
column1 datatype **constraint,**
column2 datatype **constraint,**
column3 datatype **constraint,**
....
);

Constraint in sql:

- *NOT NULL - Ensures that a column cannot have a NULL value*
- *UNIQUE - Ensures that all values in a column are different*
- *PRIMARY KEY - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table*
- *FOREIGN KEY - Prevents actions that would destroy links between tables*
- *CHECK - Ensures that the values in a column satisfies a specific condition*
- *DEFAULT - Sets a default value for a column if no value is specified*
-
- **Data Types in sql: (Refer w3schols website)**
- **1.String data type**
- **2.Numeric data type.**
- **3.Date data type**

What is Null: *value represent missing unknown data.*

IS NULL: *This is used to select only the records with NULL values in the column*

Syntax:

```
Select column_name from table_name
Where column_name is null
```

IS NOT NULL: *This is used to select only the records with no Null Values in the column*

Syntax:

Select column_name from table_name
Where column_name is not null

FOREIGN key:

To make the relationship between two or more than tables we use FOREIGN key

The FOREIGN KEY constraint is used to prevent actions that would destroy links between tables.

A FOREIGN KEY is a field (or collection of fields) in one table, that refers to the PRIMARY KEY in another table.

The table with the foreign key is called the child table, and the table with the primary key is called the referenced or parent table.

Note: *it accepts only those record which is present in parent table*

Condition:

1. One table should contain Primary key and another table contains foreign key

2. Common column name or different column name in both table

3.The common column datatype must be same in both table

Points:

1.Foregin key in one table points to a primary key in another table.

2.A foreign key can have a different name than the primary key it comes from

3.*The primary key used by a foreign key I also known as parent key. The table where the primary key is from is known as parent table*

4.*The foreign key can be used to make sure that the row in one table have corresponding row in another table*

4.Foregin key value can be null, even though primary key value cant

Syntax:

CREATE TABLE TableName (

Column_name1 Datatype

Constraint, Column_name2

Datatype Constraint,

…
Column_name datatype FOREIGN KEY REFERENCES Parent_tablename(Column_name)
);

Alias Name:

alias are used to temporarily rename a Column or table name

1.temporarily Column rename:

Syntax:

select Column_name as aliasname from table_name

2. temporarily table rename:

Syntax:

select column_name from table_name as alias_name;

Set Operators:

To combine two or more than two select statement and display as single unit of values

Types of Set Operators:

1.Union

2.Unionall

3.Intersect

4.Except

Set Operators Condition:

1 Number of columns should same with in both select statement

2.Order of the columns should be same

3.Datatypes of the columns must be match

Syntax:

Select statement

Set Operator

Select statement;

1. Union:

To combined the result of two or more than two select statements as as single unit of values without duplicate values

Syntax:

Select statement

Union

Select statement;

2. UnionAll:

To combined the result of two or more than two select statements as as single unit of values with duplicate values

Syntax:

Select statement

Unionall

Select statement;

3. **Intersect:** *it returns common value from both tables.*

Syntax:

Select statement

Intersect

Select statement;

4. Except:

Except operators is used for is used to Returns all values from the left-hand side table which are not found in the right-hand table.

Syntax:

Select Statement

Except

Select statement;

Distinct:

The SELECT DISTINCT statement is used to return only distinct (different) values.

Inside a table, a column often contains many duplicate values; and sometimes you only want to list the different (distinct) values.

Syntax:

SELECT DISTINCT *column1*, *column2*, .
FROM *table_name*;

Clause in SQL:

1.Where clause

2.Top Clause

3.Order by Clause

4. Group by clause

5.Having clause

1.Where clause:

The WHERE clause is used to filter records.

It is used to extract only those records that fulfill a specified condition.

Syntax:

SELECT *column1*, *column2, ...*
FROM *table_name*
WHERE *condition*;

2.Top Clause

The SELECT TOP clause is used to specify the number of records to return.

The SELECT TOP clause is useful on large tables with thousands of records. Returning a large number of records can impact performance.

Syntax:

SELECT TOP number|percent column_name(s)
FROM table_name
WHERE condition*;*

3.Order by Clause

The ORDER BY keyword is used to sort the result-set in ascending or descending order.

The ORDER BY keyword sorts the records in ascending order by default. To sort the records in descending order, use the DESC keyword.

Syntax

SELECT column1, column2, ...
FROM table_name
ORDER BY column1, column2, ... *ASC|DESC;*

5.Group by clause

Group by clause form a group the basis of similar type of data and produce separate result. The GROUP BY statement is often used with aggregate functions (COUNT(), MAX(), MIN(), SUM(), AVG()) to group the result-set by one or more columns.

Or

Group by clause is used to grouping similar data based on the columns

Note: *group by clause perform operations in buffer not in database.*

Syntax:

```
SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
ORDER BY column_name(s);
```

5.Having clause

Having clause is used to filtering records after grouping data and having clause can be used along with group by clause.The HAVING clause was added to SQL because the WHERE keyword cannot be used with aggregate functions.

Syntax:

SELECT column_name(s)
FROM table_name
WHERE condition
GROUP BY column_name(s)
HAVING condition
ORDER BY column_name(s);

Difference between Where and Having Clause:

WHERE Clause	HAVING Clause
WHERE Clause is used to filter the records from the table based on the specified condition.	HAVING Clause is used to filter record from the groups based on the specified condition.
WHERE Clause implements in row operations	HAVING Clause implements in column operation
WHERE Clause can be used without GROUP BY Clause	HAVING Clause cannot be used without GROUP BY Clause
WHERE Clause can be used with SELECT, UPDATE, DELETE statement.	HAVING Clause can only be used with SELECT statement.

Joins:

Join statement is used to combine data or rows from two or more tables based on a common field between them. Different types of Joins are as follows:

1.INNER JOIN

2.LEFT OUTER JOIN

3.RIGHT OUTER JOIN

4.FULL OUTER JOIN

5.CROSS JOIN

6.EQUI JOIN

7.NON EQUI JOIN

8.SELF JOIN

Note: *In case of inner join, left outer join, right outer join, full outer join condition is*

1.We retrieve the data from multiple table bases on equality condition

2.Common column

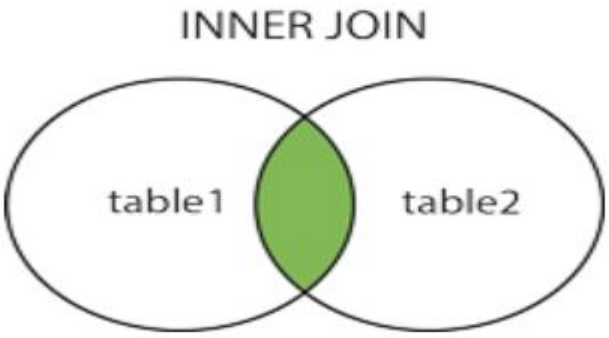
3.the common column data type must be same.

1.INNER JOIN:

The INNER JOIN keyword selects records that have matching values in both tables.it is also called as normal join.

Syntax:

SELECT column_name(s)
FROM table1
INNER JOIN table2
ON table1.column_name = table2.column_name*;*

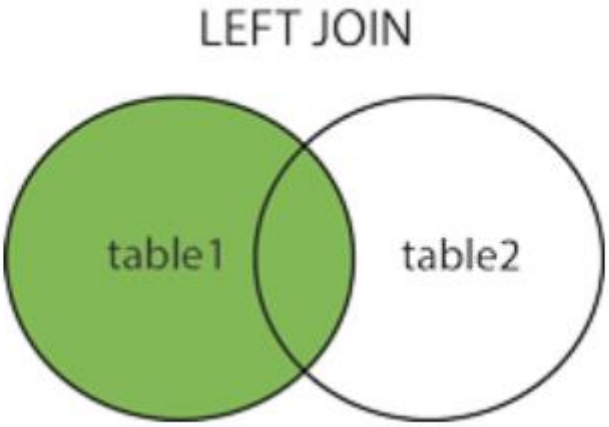


2.Left outer Join or Left Join:

The LEFT JOIN keyword returns all records from the left table (table1), and the matching records from the right table (table2). The result is 0 records from the right side, if there is no match.

Syntax:

SELECT column_name(s)
FROM table1
LEFT JOIN table2
ON table1.column_name = table2.column_name*;*

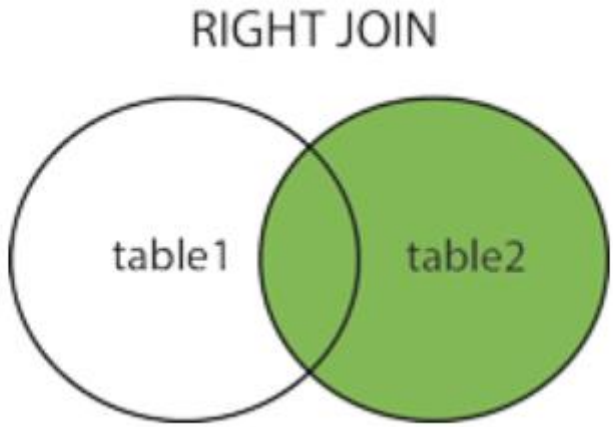


3.Right outer Join or Right Join:

The RIGHT JOIN keyword returns all records from the right table (table2), and the matching records from the left table (table1). The result is 0 records from the left side, if there is no match.

Syntax:

SELECT *column_name(s)*
FROM *table1*
RIGHT JOIN *table2*
ON *table1.column_name = table2.column_name*

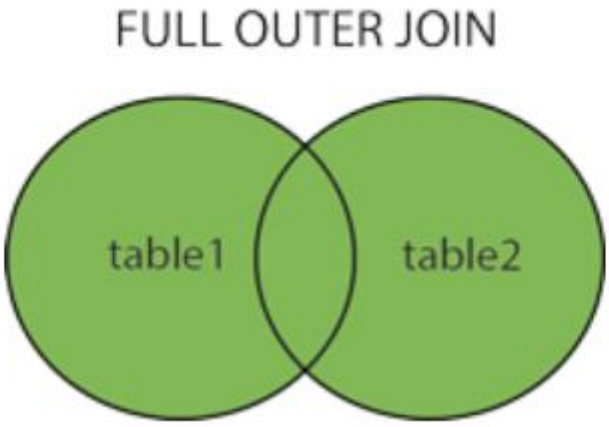


4.Full Outer Join:

The FULL OUTER JOIN keyword returns all records when there is a match in left (table1) or right (table2) table records. It display matching and unmatching both records

Syntax:

SELECT column_name(s)
FROM table1
FULL OUTER JOIN table2
ON table1.column_name = table2.column_name
WHERE condition*;*



5.Cross Join:

*A cross join is a type of join that returns the Cartesian product of rows from the tables in the join. In other words, it combines each row from the first table with each row from the second table. Product of rows that means M*N rows*

1.To join two more than two tables' data without join condition

2.No requirement of common column

Syntax:

Select column_name from table1 ***cross join*** *table2*

6.Equi join

Equi join we used for retrieving data from multiple table bases on and equality condition.

We can not use following operators in Equi Join.

<, >,<=, >=,!<, !>,!=

Syntax:

*Select * from Table1,Table2 where Table1.Column_name=Table2.Column_name*

7.Non Equi Join:

Non Equi Join we used for retrieving data from multiple table bases on any condition except equal condition.

Note: we cannot use = and we can use <,>,<=,>=,!<,!>,between, and

Syntax:

*Select * from Table1,Table2 where Table1.Column_name>Table2.Column_name*

8.Self Join:

1.Self-join is used for Joining a table data by itself is called as self-join.

2.Self join can be used with alias name of the table

3.Self Join can be implemented for single table only.

4.When we compare common values within same table that time we use self-join

5.In self-join we can define multiple alias

AUTO INCREMENT Field

Auto-increment allows a unique number to be generated automatically when a new record is inserted into a table.

Often this is the primary key field that we would like to be created automatically every time a new record is inserted.

Syntax:

```
CREATE TABLE Persons (
    Columnname Datatype IDENTITY(1,1) PRIMARY KEY,
    Columnname Datatype Constraint,
    Columnname Datatype  Constraint
    …
);
```

View:

In SQL, a view is a virtual table based on the result-set of an SQL statement.

A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.

You can add SQL statements and functions to a view and present the data as if the data were coming from one single table.

Syntax:

```
CREATE VIEW view_name AS
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

Stored Procedure:

A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again.

So if you have an SQL query that you write over and over again, save it as a stored procedure, and then just call it to execute it.

Syntax:

```
CREATE PROCEDURE procedure_name
AS
sql_statement
GO;
```

Execute a Stored Procedure

EXEC *procedure_name*;

Or

Procedure_name;

Functions In SQL:

Functions are the objects in SQL which take one or more input from user take operations on input and produce one or more output.

Types of Aggregate Functions:

1.SQL Aggregate Functions

In this tutorial, you will learn about the SQL aggregate functions including AVG(), COUNT(), MIN(), MAX(), and SUM().

2.SQL Comparison Functions

This section provides you with the SQL comparison functions including COALESCE, DECODE, and NULLIF.

3.SQL String Functions

This section provides you with some handy SQL string functions that allow you to manipulate string data more effectively.

4.SQL Math Functions

SQL has many mathematical functions that allow you to perform business and engineering calculations.

5.SQL Date Functions

This section covers the most important SQL date functions that allow you to manipulate date and time data effectively.

6.SQL Window Functions

This tutorial shows you how to use the SQL window functions to solve complex query challenges in easy ways.

What Is Normazliation

1.Normalization is a database design technique that organizes tables in am manner that reduces redundancy(useless) abd dependency of data

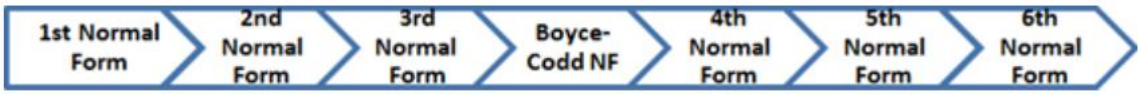
2.To avoid insertion, update, and deletion anomaly we use normalization

3.Normalization divides larger tables into smaller tables and links them using relations i.e Primary key, Foreign key

What is Anomaly: Something that is not according to standard or normal

Types of Normal Form:

Database Normalization can be easily understood with the help of a case study. The normal forms can be divided into 6 forms, and they are explained below -.



Database Normal Forms

- **First Normal Form (1NF):**

This should remove all the duplicate columns from the table. Creation of tables for the related data and identification of unique columns.

- **Second Normal Form (2NF):**

Meeting all requirements of the first normal form. Placing the subsets of data in separate tables and Creation of relationships between the tables using primary keys.

- **Third Normal Form (3NF):**

This should meet all requirements of 2NF. Removing the columns which are not dependent on primary key constraints.

- **Fourth Normal Form (4NF):**

If no database table instance contains two or more, independent and multivalued data describing the relevant entity, then it is in 4th Normal Form.

- **Fifth Normal Form (5NF):.**

A table is in 5th Normal Form only if it is in 4NF and it cannot be decomposed into any number of smaller tables without loss of data.

- **Sixth Normal Form (6NF):.**

6th Normal Form is not standardized, yet however, it is being discussed by database experts for some time. Hopefully, we would have a clear & standardized definition for 6th Normal Form in the near future…

SQL INJECTION:

SQL injection is a code injection technique that might destroy your database.

SQL injection is one of the most common web hacking techniques.

SQL injection is the placement of malicious code in SQL statements, via web page input.

What is Subquery:

A subquery is a query within another query.

This subquery can be in many places, such as in the FROM clause,

the SELECT clause, or a WHERE clause.

It's often used if you need to use the result of

one query as an input into another query.

Important Question interview point of view

1.Find out Nth Highest Salary like 2nd highest,3rd highest,4th Highest or 2nd lowest,3rd lowest,4th Lowest----please Prepare logic and its explanation


Bright
Future
Academy


Bright
Future
Academy