**SQL Assignment 4**

**Lokashwar Nallagatla**

1. Explain different types of views. Demonstrate with suitable examples.

Answer:

In SQL, views are virtual tables that are defined by a query. They present the data in a specific way without storing it physically. There are two main types of views: System Defined Views and User Defined Views.

System Defined Views

System Defined Views are built-in views provided by SQL Server to help users access metadata about the database. These views are part of the system catalog and provide information about database objects such as tables, columns, indexes, and more. They are stored in the `sys` schema.

Examples of System Defined Views

1. sys.tables: This view returns a row for each table object in the current database.

SELECT \* FROM sys.tables;

2. sys.columns: This view returns a row for each column of all the tables and views in the database.

SELECT \* FROM sys.columns WHERE object\_id = OBJECT\_ID('your\_table\_name');

3. sys.indexes: This view returns a row for each index and table in the database.

SELECT \* FROM sys.indexes WHERE object\_id = OBJECT\_ID('your\_table\_name');

User Defined Views

User Defined Views are created by users to simplify complex queries, encapsulate logic, and present data in a specific way. They are defined using the `CREATE VIEW` statement and can be based on one or more tables.

Example of a User Defined View

1. Creating the View

CREATE VIEW EmployeeDepartments AS

SELECT

e.EmployeeID,

e.EmployeeName,

d.DepartmentName

FROM

Employees e

JOIN

Departments d ON e.DepartmentID = d.DepartmentID;

2. Using the View

SELECT \* FROM EmployeeDepartments;

This view simplifies the process of getting employee names along with their department names, without needing to join the tables every time.

1. What is the difference between function and stored procedure? Write syntax for creating functions and stored procedures.

Answer:

1. - Functions: Used to perform calculations, return a single value, or return a table. They are often used in SQL statements where expressions are allowed, such as SELECT, WHERE, and JOIN clauses.

- Stored Procedures: Used to execute a series of SQL statements, which may include control-of-flow statements like loops and conditionals. They are used for more complex operations, including data modification.

2. - Functions: Must return a value (scalar value, table, or table row).

- Stored Procedures: May return zero or more values, including output parameters and result sets, but are not required to return a value.

3. - Functions: Can be used inline in SQL statements.

- Stored Procedures: Cannot be used directly in SQL statements. They are executed using the `EXEC` or `EXECUTE` command.

4. - Functions: Only accept input parameters.

- Stored Procedures\*\*: Accept input, output, and input-output (INOUT) parameters.

5. - Functions: Limited error handling.

- Stored Procedures: Robust error handling with the ability to use TRY...CATCH blocks.

6. - Functions: Cannot manage transactions.

- Stored Procedures: Can begin, commit, or roll back transactions.

Syntax for Creating Functions:

CREATE FUNCTION dbo.GetEmployeesByDepartment (@DepartmentID INT)

RETURNS TABLE

AS

RETURN

(

SELECT EmployeeID, EmployeeName, DepartmentID

FROM Employees

WHERE DepartmentID = @DepartmentID

);

Syntax for Creating Stored Procedures:

CREATE PROCEDURE dbo.AddNewEmployee

@EmployeeName NVARCHAR(50),

@DateOfBirth DATE,

@DepartmentID INT

AS

BEGIN

INSERT INTO Employees (EmployeeName, DateOfBirth, DepartmentID)

VALUES (@EmployeeName, @DateOfBirth, @DepartmentID);

-- Optional: Return the ID of the newly inserted employee

SELECT SCOPE\_IDENTITY() AS NewEmployeeID;

END;

1. What is an index in SQL? What are the different types of indexes in SQL?

Answer:

An index in SQL is a database object that improves the speed of data retrieval operations on a table at the cost of additional storage space and potentially slower write operations (inserts, updates, and deletes). Indexes are created on columns in a table to allow quick lookup of rows based on the values in these columns.

Types of Indexes in SQL:

1. Clustered Index:

- A clustered index determines the physical order of data in a table. There can be only one clustered index per table because the data rows themselves can be sorted in only one order.

- Example:

CREATE CLUSTERED INDEX idx\_employee\_id

ON Employees(EmployeeID);

2. Non-Clustered Index:

- A non-clustered index creates a separate structure from the data rows. The index contains pointers to the data rows, allowing for multiple non-clustered indexes per table.

- Example:

CREATE NONCLUSTERED INDEX idx\_employee\_name

ON Employees(EmployeeName);

3. Unique Index:

- A unique index ensures that the values in the indexed column(s) are unique. Both clustered and non-clustered indexes can be unique.

- Example:

CREATE UNIQUE INDEX idx\_employee\_email

ON Employees(Email);

1. Showcase an example of exception handling in SQL stored procedure.

Answer:

DECLARE

name VARCHAR2(20);

ans1 VARCHAR2(3);

ans2 VARCHAR2(3);

ans3 VARCHAR2(3);

suffix NUMBER := 1;

BEGIN

FOR i IN 1..10 LOOP

BEGIN

SAVEPOINT start\_transaction;

DELETE FROM results WHERE answer1 = 'NO';

INSERT INTO results VALUES (name, ans1, ans2, ans3);

COMMIT;

EXIT;

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

ROLLBACK TO start\_transaction;

suffix := suffix + 1;

name := name || TO\_CHAR(suffix);

END;

END LOOP;

END;

1. Create a SQL function to split strings into rows on a given character?

Input String: Stephen;peter;berry;Olivier;caroline;

SELECT COLUMN\_VALUE

FROM TABLE(split\_string('Stephen; peter; berry; Olivier; caroline;', ';'));

1. What is a temporary and a variable table? Write suitable syntax to create temporary tables and variable tables.

Temporary Tables

A temporary table is a table that is created temporarily to store data for a short period of time. It is used to store intermediate results, and it is dropped automatically when the session is closed or the transaction is committed or rolled back. Temporary tables are useful when you need to perform complex operations that require intermediate results, but you don't want to store those results permanently.

Syntax to create a temporary table:

CREATE TABLE #temp\_table (

column1 datatype,

column2 datatype,

...

);

Variable Tables (Table Variables)

A variable table, also known as a table variable, is a variable that can store a table-like structure. It is a user-defined variable that can be used to store data temporarily, similar to a temporary table. However, unlike temporary tables, variable tables are not stored in the database and do not require a separate create statement.

Syntax to create a variable table:

DECLARE @temp\_table TABLE (

column1 datatype,

column2 datatype,

...

);