**SQL Assignment 4**

1. Explain different types of views. Demonstrate with suitable examples.
2. What is the difference between function and stored procedure? Write syntax for creating functions and stored procedures.
3. What is an index in SQL? What are the different types of indexes in SQL?
4. Showcase an example of exception handling in SQL stored procedure.
5. Create a SQL function to split strings into rows on a given character?

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

|  |
| --- |
| Stephen |
| Peter |
| Berry |
| Oliver |
| Caroline |

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

Solve

Here’s a detailed response to each part of your SQL assignment:

### 1. Types of Views in SQL

\*\*Views\*\* are virtual tables in SQL that represent the result of a query. They can simplify complex queries, provide security, and encapsulate complex logic. The main types of views include:

- \*\*Simple Views\*\*: These are created from a single table and do not contain functions or GROUP BY clauses. They are updatable.

\*\*Example\*\*:

```sql

CREATE VIEW SimpleView AS

SELECT FirstName, LastName

FROM Employees;

```

- \*\*Complex Views\*\*: These are created from multiple tables and may include joins, functions, and GROUP BY clauses. They are generally not updatable.

\*\*Example\*\*:

```sql

CREATE VIEW ComplexView AS

SELECT e.FirstName, e.LastName, d.DepartmentName

FROM Employees e

JOIN Departments d ON e.DepartmentId = d.DepartmentId;

```

- \*\*Materialized Views\*\*: These are physical copies of the data that can be refreshed periodically. They improve performance for complex queries.

\*\*Example\*\*:

```sql

CREATE MATERIALIZED VIEW SalesSummary AS

SELECT ProductId, SUM(SalesAmount) AS TotalSales

FROM Sales

GROUP BY ProductId;

```

### 2. Difference Between Function and Stored Procedure

- \*\*Function\*\*: A function is a database object that can return a single value or a table. It can be used in SQL expressions.

- \*\*Stored Procedure\*\*: A stored procedure is a set of SQL statements that can perform operations but cannot be used directly in SQL expressions.

\*\*Syntax for Creating a Function\*\*:

```sql

CREATE FUNCTION FunctionName (parameters)

RETURNS data\_type

AS

BEGIN

-- Function logic

RETURN value;

END;

```

\*\*Syntax for Creating a Stored Procedure\*\*:

```sql

CREATE PROCEDURE ProcedureName (parameters)

AS

BEGIN

-- Procedure logic

END;

```

### 3. Index in SQL

An \*\*index\*\* is a database object that improves the speed of data retrieval operations on a database table at the cost of additional space and slower writes.

\*\*Types of Indexes\*\*:

- \*\*Single-column Index\*\*: An index on a single column of a table.

\*\*Example\*\*:

```sql

CREATE INDEX idx\_lastname ON Employees(LastName);

```

- \*\*Composite Index\*\*: An index on multiple columns.

\*\*Example\*\*:

```sql

CREATE INDEX idx\_fullname ON Employees(FirstName, LastName);

```

- \*\*Unique Index\*\*: Ensures that the indexed columns contain unique values.

\*\*Example\*\*:

```sql

CREATE UNIQUE INDEX idx\_email ON Employees(Email);

```

- \*\*Full-text Index\*\*: Allows for full-text searches on textual data.

### 4. Exception Handling in SQL Stored Procedure

You can handle exceptions in SQL Server using `TRY...CATCH` blocks.

\*\*Example\*\*:

```sql

CREATE PROCEDURE SampleProcedure

AS

BEGIN

BEGIN TRY

-- Code that may throw an exception

INSERT INTO Employees (FirstName, LastName)

VALUES ('John', 'Doe');

END TRY

BEGIN CATCH

-- Handling the error

PRINT 'Error occurred: ' + ERROR\_MESSAGE();

END CATCH;

END;

```

### 5. SQL Function to Split Strings into Rows

You can create a function that splits a string based on a delimiter and returns the results as a table.

\*\*Example\*\*:

```sql

CREATE FUNCTION SplitString

(

@InputString NVARCHAR(MAX),

@Delimiter CHAR(1)

)

RETURNS @OutputTable TABLE (Value NVARCHAR(MAX))

AS

BEGIN

DECLARE @Start INT, @End INT

SET @Start = 1

WHILE CHARINDEX(@Delimiter, @InputString, @Start) > 0

BEGIN

SET @End = CHARINDEX(@Delimiter, @InputString, @Start)

INSERT INTO @OutputTable (Value)

VALUES (SUBSTRING(@InputString, @Start, @End - @Start))

SET @Start = @End + 1

END

INSERT INTO @OutputTable (Value)

VALUES (SUBSTRING(@InputString, @Start, LEN(@InputString) - @Start + 1))

RETURN

END;

-- Usage

SELECT \* FROM SplitString('Stephen;peter;berry;Olivier;caroline;', ';');

```

### 6. Temporary and Variable Tables

- \*\*Temporary Tables\*\*: These are used to store temporary data and are deleted automatically when they go out of scope.

\*\*Syntax to Create a Temporary Table\*\*:

```sql

CREATE TABLE #TempTable

(

Id INT,

Name NVARCHAR(50)

);

```

- \*\*Variable Tables\*\*: These are declared in the procedure and used to store data temporarily, similar to variables.

\*\*Syntax to Create a Variable Table\*\*:

```sql

DECLARE @VarTable TABLE

(

Id INT,

Name NVARCHAR(50)

);

```

These explanations and examples should help you with your SQL assignment! If you have any further questions or need additional examples, feel free to ask.