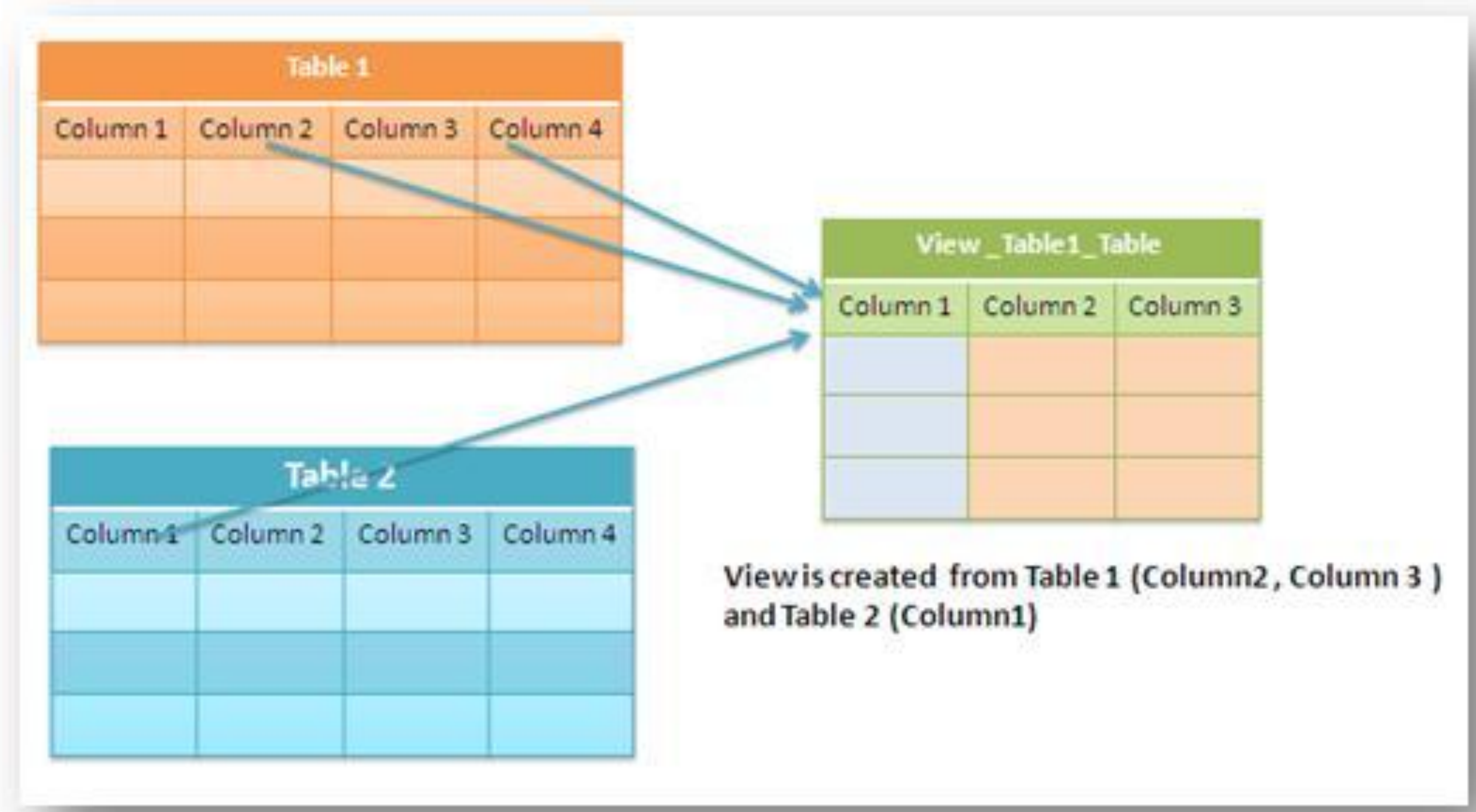
Database Development

Topics for today:

- ❑ Views
- ❑ Store Procedure

View – T-SQL



View is created from Table 1 (Column2 , Column 3) and Table 2 (Column1)

HOUSE OF TECHNOLOGY

Views

1. Tables store real data.
2. Views do not store real data.
3. Views must have underlying tables to provide data.
4. Each view is based on a single SELECT statement to control what data to collect from tables, and how data should be represented.
5. View's columns can be mapped directly to columns in underlying tables.
6. View's columns can be created expressions based multiple columns in underlying tables.
7. Views can be used in same way as tables in queries.

Views

SQL CREATE VIEW Syntax

```
CREATE VIEW view_name AS
SELECT column_name(s)
FROM table_name
WHERE condition
```

Views

Advantages:

To hide data complexity.

To protect the data.

Enforcing some simple business rules.

Customizing data.

Disadvantage:

Views can especially degrade the performance if they are based on other views.

Store Procedure

Benefits of Stored Procedures

1. Precompiled execution
2. Reduced client/server traffic
3. Efficient reuse of code and programming abstraction
4. Enhanced security controls

Store Procedure

ID	Product	Warehouse	Quantity
142	Green beans	NY	100
214	Peas	FL	200
825	Corn	NY	140
512	Lima beans	NY	180
491	Tomatoes	FL	80
379	Watermelon	FL	85

Store Procedure

```
SELECT Product, Quantity
FROM Inventory
WHERE Warehouse = 'FL'
```

HOUSE OF TECHNOLOGY

Store Procedure

```
CREATE PROCEDURE sp_GetInventory
@location varchar(10)
AS
SELECT Product, Quantity
FROM Inventory
WHERE Warehouse = @location
```

Store Procedure

```
EXECUTE sp_GetInventory 'FL'
EXECUTE sp_GetInventory 'NY'
```

Store Procedure vs. Functions

1. **Procedure** can return zero or n values whereas **function** can return one value which is mandatory.
2. **Procedure**s can have input/output parameters for it whereas **function**s can have only input parameters.
3. **Procedure** allows select as well as DML statement in it whereas **function** allows only select statement in it.
4. **Function**s can be called from **procedure** whereas **procedure**s cannot be called from **function**.
5. Exception can be handled by try-catch block in a **procedure** whereas try-catch block cannot be used in a **function**.
6. We can go for transaction management in **procedure** whereas we can't go in **function**.
7. **Procedure**s can not be utilized in a select statement whereas **function** can be embedded in a select statement.

Store Procedure

```
CREATE PROCEDURE uspTryCatchTest
AS
BEGIN TRY
   SELECT 1/0
END TRY
BEGIN CATCH
   SELECT ERROR_NUMBER() AS ErrorNumber
    ,ERROR_SEVERITY() AS ErrorSeverity
    ,ERROR_STATE() AS ErrorState
    ,ERROR_PROCEDURE() AS ErrorProcedure
    ,ERROR_LINE() AS ErrorLine
    ,ERROR_MESSAGE() AS ErrorMessage;
END CATCH
```

TASK 11:

Google about **Views in T-SQL** (Create following views by using T-SQL).

1. Create a view for **CHEF** and name is **CHEF_VIEW**. This view should show information about date, name, and license and damage amount columns.
2. Create another view for **DBA** and name is **DBA_VIEW**. This view should show information about total number of reports and total amount of damage amount columns.
3. Create another view **USER** and name it **USER_VIEW**. This view should show information about drive id, name, date and damage-amount columns.
4. Create a new **SCHEMA** named **BILEN** on Car Insurance database. Bind all the above three views with it. In order words change the schema membership of all views from DBO to **BILEN**.
5. Use select statement to view all views after they are binded with **BILEN SCHEMA**. Check the results if it is the same?

TASK 12:

1. Create carinfo.txt file that contain information about 10 cars in such a way that 4 car models are BMW, 3 car models are Volvo and rest of 3 cars models are Suzuki. (with respect to car table in CAR-INSURANCE DB).

2. Use BULK INSERT command to insert all 10 records into car table in CAR-INSURANCE DB.

3. Create a store procedure sp_car_name which takes car model name as a parameter and retrieves the desired result.

TASK 13:

1. Change the name of CAR_INSURANCE DB to CAR_INSULT in task13.sql script.

2. Put all code of task12.sql into a store prcedure sp_carinsult. Execute the store procedure without parameter. Is it possible to run a store procedrue without parameter?

TASK 14:

1. Put USER_VIEW inside a new store procedure sp_userview. Check if it is possible to execute a store procedue that has a view inside its code?

2. Create a VIEW, SPVIEW that retrieves model and year columns from Car table. The condition is that this view emebed sp_car_name to retrieve car model. Check if it is possible to run a store procedure inside view?

Reading:

Chapter 9 Lesson 1: Designing and Implementing Views and Inline Functions (pg.300-307)

Chapter 13 Lesson 1: Designing and Implementing Stored Procedures(pg. 502-514)

Bulk Insert

http://technet.microsoft.com/en-us/library/ms188365.aspx