Name: Andy Bigley

Date: 19 November 2025

Course: IT FDN 130 A Au 25: Foundations of Databases & SQL Programming

Assignment06

Github URL: https://github.com/abigley2004/DBFoundations

Introduction

During the sixth week of the course, SQL views were introduced as a way to organize and simplify data retrieval within a relational database. Views encapsulate **SELECT** statements into reusable virtual tables that present data clearly in the abstraction layer. They reduce query repetition and enhance security by hiding underlying table structures. The week also compared views with functions and stored procedures, each serving a distinct role in structuring database logic. Together, these tools support efficient data access and promote modular, maintainable database systems.

Academic Attribution Note:

I completed the coding and Q&A assignments independently after reviewing the course videos and materials. I used ChatGPT only to improve the clarity of my explanations and confirm the accuracy of my SQL concepts. The final work reflects my own understanding of the material.

1. ***Explain when you would use a SQL View.***

A SQL view is useful to simplify how users retrieve data from one or more tables. In a well-designed database, information is stored across multiple related tables, which can make queries long or difficult for users who do not need full control over the underlying structure. A view provides a saved, reusable **SELECT** statement that presents the data as if it were a single table, allowing users to work with organized and meaningful results without writing complex joins or filters themselves. Views are also helpful for improving security, because they expose only the specific columns or rows that users should see while hiding the underlying tables. In addition, views promote consistency by ensuring that everyone retrieves data using the same approved logic.

2. ***Explain the differences and similarities between a View, Function, and Stored Procedure.***

A view, function, and stored procedure are all database objects used to organize and reuse SQL logic, but they differ in how they operate and how they are used within a system.

Object	Differences	Similarities
View	A view is a virtual table that stores a predefined **SELECT** statement. It cannot accept parameters and is generally used to simplify data retrieval. Views are often schemabound to protect the underlying tables.	Like functions and stored procedures, a view encapsulates SQL logic and can be reused to promote consistency and simplify queries.
Function	A function must return a value and cannot modify data. It can accept parameters and is commonly used for calculations, expressions, or returning table-valued results. Functions cannot execute actions such as **INSERT**, **UPDATE**, or **DELETE**. There are two basic types: Those that return a table of values and those that return a single (scalar) value.	Functions, like views and stored procedures, store SQL logic that can be called repeatedly and help keep code organized.
Stored Procedure	A stored procedure can perform complex operations, accept parameters, and modify data using **INSERT**, **UPDATE**, and **DELETE** statements. It does not have to return a value. Procedures can include control-of-flow logic such as IF statements and loops.	Stored procedures also encapsulate SQL logic and promote reusability, similar to views and functions, and they provide a structured way to perform repeated tasks.

To demonstrate how each object is defined and used, the following SQL examples are based on the material covered in this week's homework.

Object	SQL Code and Use Example

View
 	
This view exposes basic product information.
	`CREATE VIEW dbo.vwProductsBasic`
	`WITH SCHEMABINDING`
	`AS`
	`SELECT`
	`    p.ProductID,`
	`    p.ProductName,`
	`    p.CategoryID,`
	`    p.UnitPrice`
	`FROM dbo.Products AS p;`
	`GO`

Object	SQL Code and Use Example

View	*Use:*
	`SELECT`
	`    ProductID,`
	`    ProductName,`
	`    UnitPrice`
	`FROM dbo.vwProductsBasic`
	`ORDER BY UnitPrice;`

Function
 	
This scalar function returns a discounted price for a given product. The number of items and discount rate are passed in as function parameters.
	`CREATE FUNCTION dbo.fnGetProductDiscountedPrice`
	`(`
	`    @ProductID    INT,`
	`    @DiscountRate DECIMAL(4,2)  -- e.g., 0.10 for 10%`
	`)`
	`RETURNS MONEY`
	`AS`
	`BEGIN`
	`    DECLARE @BasePrice MONEY;`
	
	`    SELECT @BasePrice = p.UnitPrice`
	`    FROM dbo.Products AS p`
	`    WHERE p.ProductID = @ProductID;`
	
	`    RETURN @BasePrice * (1 - @DiscountRate);`
	`END;`
	`GO`
	
	Use:
	`SELECT dbo.fnGetProductDiscountedPrice(1, 0.10) AS DiscountedPrice;`
	
	Use as a query:
	`SELECT`
	`    p.ProductID,`
	`    p.ProductName,`
	`    p.UnitPrice,`
	
	`dbo.fnGetProductDiscountedPrice(p.ProductID, 0.15) AS PriceWith15PercentOff`
	`FROM dbo.Products AS p;`

Object	SQL Code and Use Example

Stored Procedure
 	
This stored procedure returns products with a minimum unit price as a parameterized query over product unit price.
	`CREATE PROCEDURE dbo.uspGetProductsByMinPrice`
	`(`
	`    @MinPrice MONEY`
	`)`
	`AS`
	`BEGIN`
	`    SET NOCOUNT ON;`
	
	`    SELECT`
	`        p.ProductID,`
	`        p.ProductName,`
	`        p.CategoryID,`
	`        p.UnitPrice`
	`    FROM dbo.Products AS p`
	`    WHERE p.UnitPrice >= @MinPrice`
	`    ORDER BY p.UnitPrice DESC;`
	`END;`
	`GO`
	
	Use:
	`EXEC dbo.uspGetProductsByMinPrice @MinPrice = 20.00;`

Together, these objects help developers simplify queries, enforce consistent logic, and organize database operations in a maintainable way.

Summary

By the end of the lesson, the material highlighted how views streamline complex queries and promote consistent reporting across applications and users. Creating simple and multi-table schemabound views emphasized the importance of explicit column selection, stable table relationships, and thoughtful result ordering. Examining functions and stored procedures alongside views showed how each object contributes to an effective abstraction layer: views simplify data retrieval, functions provide reusable calculations or table-valued results, and stored procedures handle more advanced operations. Together, these techniques improve clarity, security, and maintainability in database systems and provide a strong foundation for more advanced SQL topics.