Nada Yousif

November 16th, 2025

Foundations of Databases and SQL Programming

Assignment06_Writeup

NYousif123/DBFoundations

A Comparative Analysis of SQL Views, Functions, and Stored Procedures

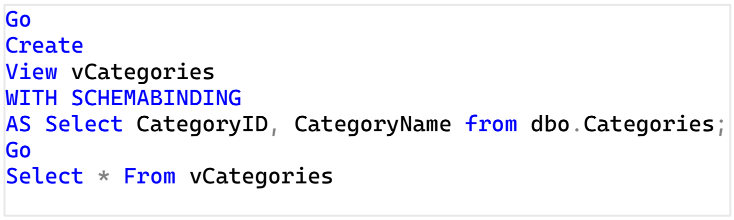
Introduction

In this week's module, we explored SQL Views, Functions, and Stored Procedures, looking at both their similarities and differences, as well as scenarios where each might be most useful. All three simplify complex SQL queries by providing an abstraction layer that streamlines data access and output. Their differences lie in the way they retrieve data and the specific SQL code they employ. Below, I will highlight the key features and purposes of Views, Functions, and Stored Procedures.

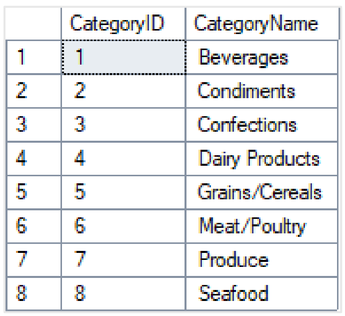
Explain When You Would Use a SQL View.

In SQL, views serve as a type of abstraction layer. Abstraction layers are a component of Relational Database Management Systems that allow for data to be presented in a table format while the specific SQL queries and programming remain "hidden". If changes need to be made to the underlying table in a database, you can update the View so that applications using it remain unaffected (Root, 2025).

Views also allow for Schema-binding, which is a way you can protect a View from being disrupted when the structure of its source tables keep changing.



```
Go
Create
View vCategories
WITH SCHEMABINDING
AS Select CategoryID, CategoryName from dbo.Categories;
Go
Select * From vCategories
```

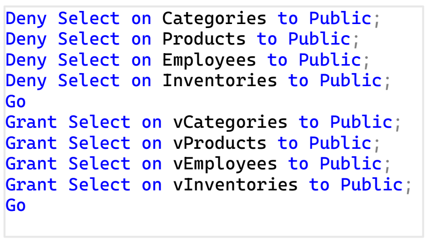


	CategoryID	CategoryName
1	1	Beverages
2	2	Condiments
3	3	Confections
4	4	Dairy Products
5	5	Grains/Cereals
6	6	Meat/Poultry
7	7	Produce
8	8	Seafood

Figure 1.1: SQL code and associated table result for a view for the Categories table

Programmers may want to create a view when they are writing complex SQL queries that they want to protect from other users. Instead of users of a database querying the table containing the complex code, they can query a simplified version of table in the form of a view. The view is presented as if it were a real table.

Views also allow for permissions that deny the public from selecting data from a table but allow them access to select from a view. This added layer of protection makes it easier to control individual user or group access and overall database usage.



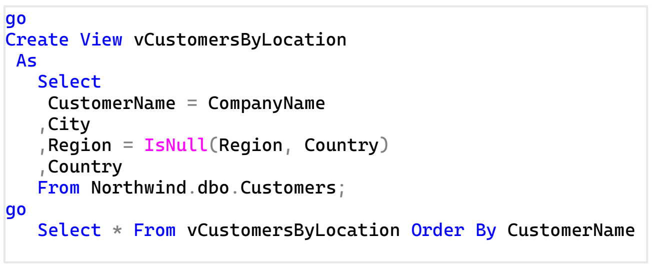
```
Deny Select on Categories to Public;
Deny Select on Products to Public;
Deny Select on Employees to Public;
Deny Select on Inventories to Public;
Go
Grant Select on vCategories to Public;
Grant Select on vProducts to Public;
Grant Select on vEmployees to Public;
Grant Select on vInventories to Public;
Go
```

Figure 1.2: Example of adding permissions in database tables and views

Explain the Differences and Similarities Between a View, Function, and Stored Procedure.

Views, functions, and stored procedures can encapsulate complex SQL queries, providing an abstraction layer that simplifies data access. They can return data without exposing the underlying query logic. While views, functions, and stored procedures differ in the type of data they can return, their real differences lie in their purpose, capabilities, and how they interact with the database

- **View:** A view uses a Select query that creates and returns a simplified table. The new view table must be saved using a different name than the table it is based on. The actual data that the view returns is not saved on a hard drive, rather it is the Select table that is stored (Root, 2025). Views are simpler than functions and stored procedures.



```
go
Create View vCustomersByLocation
 As
   Select
    CustomerName = CompanyName
   ,City
   ,Region = IsNull(Region, Country)
   ,Country
   From Northwind.dbo.Customers;
go
   Select * From vCustomersByLocation Order By CustomerName
```

Figure 2.1: Example of a View from Module 06, Practice Lab 01

- **Function:** Functions in SQL, also called User-Defined Functions (UDFs), are database objects that can return either a single value (scalar) or a table (table-valued). Unlike views, functions can accept parameters before returning a result.

```sql
go
Create or Alter Function dbo.fCustomersByLocation()
 Returns Table
 As
   Return(
   Select
    CustomerName = CompanyName
   ,City
   ,Region = IsNull(Region, Country)
   ,Country
   From Northwind.dbo.Customers
   );
go
Select * From dbo.fCustomersByLocation() Order By CustomerName;
```

Figure 2.2: Example of a Function from Module 6, Practice Lab 02

Stored Procedure: Stored procedures, like views and functions, are database objects that store SQL code. Unlike views and functions, stored procedures can perform data modification operations such as INSERT, UPDATE, and DELETE, in addition to returning query results. For more complex or multi-step operations, stored procedures are often the preferred choice.

```sql
go
Create or Alter Procedure pCustomersByLocation
 As
   Select
    CustomerName = CompanyName
   ,City
   ,Region = IsNull(Region, Country)
   ,Country
   From Northwind.dbo.Customers
   Order By CustomerName;
go
Exec pCustomersByLocation;
```

Figure 2.3: Example of a Stored Procedure from Module 6, Practice Lab 03

Summary

In summary, understanding when and how to use Views, Functions, and Stored Procedures is essential for protecting complex SQL queries while providing simplified, reusable ways for others to access data. Each—View, Function, and Stored Procedure—is unique, relying on different methods to achieve its results.