Dylan Schechtel

22 February 2022

Foundations of Databases and SQL Programming

Assignment 06

<https://github.com/dschechtel-uw/DBFoundations>

Views

# Introduction

This writeup will consist of an introduction to SQL views and their standard use cases. It will also explore two other SQL abstraction layers, functions and stored procedures, and the ways they are similar to and distinct from views.

# SQL Views

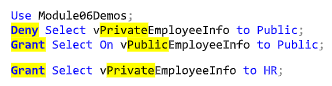
Views are a type of abstraction layer that is leveraged in SQL databases. A view is essentially a saved Select statement that is given a name and stored in the database (<https://www.tutorialspoint.com/sql/sql-using-views.htm>, 2020) (External site). While SQL views can be used to save the simplest of Select statements, they commonly store more complex statements, making it easier to use the statements repeatedly throughout a SQL script without having to write the complex code multiple times (Module06Notes). When a view is invoked, the saved SQL statement calls data forward and displays it in a result set as a temporary table. Tables generated by views are not saved in the database, but can be used in much the same way as stored tables. Views are created by prefacing a Select statement with CREATE VIEW [view name] AS (Figure 1). After the statement is run, the new view will be stored in the database and can be leveraged in future code.

Graphical user interface, text, application

Description automatically generated

***Fig 1: CREATE VIEW AS statement***

Views can be employed in a database to restrict access to specific data, to aggregate and summarize data for reporting purposes, and to make interacting with a database more intuitive for users who may have less functional knowledge of SQL. Views can be leveraged to display or hide certain datapoints based on user permissions. If a table contained sensitive or personal data (e.g., employee SSN, address, salary), it would be prudent to limit the number of users with the ability to access and manipulate that data. This can be done by using the DENY and GRANT statements to disable or enable access to a table or view. In the example depicted in Figure 2, two views have been created to house employee data. One view, vPrivateEmployeeInfo, includes sensitive information about employees that should only be available to the HR department. The other view, vPublicEmployeeInfo, only includes datapoints that are not sensitive and can be shared with non-HR individuals with the Public role.



***Fig 2: Setting permissions on views***

Views can also be used to save complex Select statements that are leveraged frequently for reporting purposes. As explained in the Module06 notes, “Any view that is used to extract data for reporting purposes is called a ‘Reporting View.’” If the same data is regularly analyzed and reported on, it is more time-effective (and often more user friendly) to save the report’s Select statement as a view, rather than rewriting the code every time the report is needed.

Views are also useful tools in making a database more intuitive and navigable for users with less SQL knowledge. In a sense, a view allows for the pre-aggregation and organization of data into a useable virtual table for other database users. Instead of having to write their own complex Select statements to query data, users can select data from a virtual table that has been created for their specific purposes. Views can create a simplified “version” of tables for users to interact with without engaging heavily with the underlying SQL and schema. Often, views display data from multiple underlying tables in one single virtual table, which removes the need for users to write their own (or even understand how to use) SQL joins.

# Views, Functions, and Stored Procedures

Functions and stored procedures are two other abstraction layers utilized in databases in addition to views. The three tools are similar in their general purpose, but they do have distinct differences in the way they are built and the scope of actions they can perform. As explained above, a view is a saved Select statement that represents a virtual table of data. While the virtual table itself isn’t stored in the database, the saved view can be leveraged to call forward the virtual table as needed. A view’s underlying Select statement can be simple or complex, but the result set of the statement will always be displayed in a table format.

Functions are similar to views in that they are saved Select statements used to perform an action or return data. Like views, they can be leveraged to return a set of values in a table, but they can also be used to return single scalar value as an expression (Module06Notes). Functions are especially unique from views because they can accept parameters to filter and change the results of the query they execute. Parameters have a similar impact in a function as a Where clause does in a Select statement.

Stored procedures are similar to both views and functions because they, too, are SQL statements that are stored to the database and can be called readily and easily. Like functions, specifically, stored procedures allow for the use of parameters to filter their results. However, stored procedures are distinct from the other two abstraction layers because they are able to store Insert, Update, and Delete statements in addition to Select statements. The SQL statements stored in views and functions are not generally used to change data (although it *is* possible to alter table data using a view). Stored procedures are generally invoked to execute a specific action or series of actions in the database, and this necessitates the use of a larger set of SQL commands and clauses. Stored procedures *can* be written to return table in a data, producing a result set like that of a view or function, but they can also be used to perform more complex actions on the data itself.

# Conclusion

SQL views, functions, and stored procedures all have similar uses in a database. They are all abstraction layers used to save SQL statements to the database, which can be invoked repeatedly and as needed. They also provide a way for database users to interact with tables and data without needing an advanced knowledge of SQL. However, there are differences between the three abstraction layers that make them distinct from one another. Views are most commonly leveraged for reporting purposes, to grant and deny access to specific data, and to make working with a database more intuitive for users. When invoked, views return data in a table. Functions, too, can return data in a table, but they can also return individual values and can incorporate parameters into their saved SQL code to filter the results they produce. Stored procedures save SQL statements to the database and can be used to perform a wider array of actions within the database. In addition to selecting data, stored procedures can also execute insert, update, and delete statements when invoked.