Ashley Lane

11/19/2025

IT FDN 130 A Au 25

Assjgnment06

https://github.com/AshleySmashley83/DBFoundations

Room with a View

Introduction

In this paper I focus on SQL `VIEW`s and briefly touch on Functions and Stored Procedures.

`CREATE VIEW`

A SQL `VIEW` is a way to save a query that presents a virtual table. You can use a view for reporting purposes that general users can access (by setting permissions) without connecting directly to tables in the database. Views are useful to form complex queries, so users can run a simple SELECT on the view.

Provide a stable API to data: let callers use the same column set while you change underlying tables.

- Simplify permission management: grant access to a view instead of the underlying tables to limit visible columns/rows.
- Implement row- or column-level security: filters inside the view can enforce data visibility rules.
- Reuse common logic across queries: central place for repeated business logic.
- Create a convenient read-only abstraction: even when base schema is complex.
- Use as a basis for reporting or BI tools that expect a table-like object.

Also consider materialized views (or indexed views) when the underlying query is expensive and you want stored results for faster reads; materialized views require refresh management.

```sql
SELECT c.CategoryName, p.ProductName, p.UnitPrice
FROM Categories AS C
    INNER JOIN Products AS P
    ON C.CategoryID = P.CategoryID
ORDER BY C.CategoryName, P.ProductName;
```

Figure 1: Simple inner join

```sql
SELECT c.CategoryName, p.ProductName, i.InventoryDate, i.[Count],
    CONCAT (e.EmployeeFirstName, ' ', e.EmployeeLastName) AS Employee
FROM Categories AS C
    INNER JOIN Products AS P
    ON C.CategoryID = P.CategoryID
    INNER JOIN Inventories AS I
    ON P.ProductID = I.ProductID
    INNER JOIN Employees AS E
    ON I.EmployeeID = E.EmployeeID
WHERE p.ProductID IN
    (SELECT ProductID
    FROM Products
    WHERE ProductName IN ('Chai', 'Chang'))
ORDER BY i.InventoryDate, c.CategoryName, p.ProductName;
```

Figure 2: More complex inner join

CREATE FUNCTION

A SELF JOIN is simply joining a table back to itself. This could be used when you need to reference or combine other rows in the same table. To do this, you use table aliases which create a virtual copy of the table then call out the relationships needed.

Stored Procedure

A SELF JOIN is simply joining a table back to itself. This could be used when you need to reference or combine other rows in the same table. To do this, you use table aliases which create a virtual copy of the table then call out the relationships needed.

Summary

Use views to present simplified, reusable, and secure read-only (or sometimes updatable) projections of data.

Use functions when you need parameterized logic that can be embedded in queries (scalar or table-valued).

Use stored procedures for procedural operations, side effects, transaction control, and batch or administrative tasks.

Choose based on your needs: inline use and composability (functions), table-like reusable abstraction (views), or procedural side-effectful work (stored procedures).