SQL Views – Research

1. Types of Views in SQL Server

View Type	What It Is	Key Differences	Real-Life Use Cases	Limitations & Performance
Standard View	A virtual table defined by a query; does not store data, only references base tables	Lightweight, dynamic; no physical storage	University system: a view combining Students and Courses for enrollment reports	Performance depends on base query; cannot always be indexed
Indexed View	A view with a clustered index; stores results physically (materialized view)	Improves performance for aggregations and joins; consumes storage	Banking: pre-aggregated balances for millions of accounts	Slower on frequent updates; strict rules (must be deterministic, schema-bound)
Partitioned View (Union View)	A view that combines horizontally partitioned tables across servers	Enables distributed queries across multiple tables/databases	E-commerce: sales data split by region, unified for global reporting	Complex to maintain; limited DML support; performance depends on partitioning

2. DML Operations on Views

- **Allowed Views:**
  - **Standard Views**: DML possible if the view references a single base table and avoids aggregates, DISTINCT, GROUP BY, UNION, etc.
  - **Indexed Views**: DML allowed but must follow strict rules (deterministic functions, schema binding).
  - **Partitioned Views**: DML possible only if all participating tables meet partitioning rules (same schema, constraints).
- **Restrictions:**
  - Cannot update views with joins, aggregates, or computed columns.
  - Triggers may be required to handle complex updates.
- **Real-Life Example:**
  - **HR System**: Updating an *EmployeeView* that exposes only active employees allows HR staff to update salaries without touching inactive records.
  - **E-commerce Orders**: Updating an *OrderSummaryView* lets customer service adjust shipping details directly.

3. How Views Simplify Complex Queries

- **Benefit:** Views encapsulate JOIN-heavy queries, reducing repetition and improving readability.
- **Scenario:** Call center agents frequently need account summaries. Instead of writing long JOIN queries, they query a prebuilt view.

Example: Banking System

```
sql
-- Create a view joining Customer and Account
CREATE VIEW CustomerAccountSummary AS
SELECT
   c.CustomerID,
   c.Name,
   a.AccountID,
   a.AccountType,
   a.Balance
FROM Customer c
JOIN Account a ON c.CustomerID = a.CustomerID;
-- Usage: Instead of repeating the JOIN
SELECT * FROM CustomerAccountSummary WHERE Balance > 1000;
```

This reduces query complexity and ensures consistency across teams.

Sources:

Views - SQL Server | Microsoft Learn

https://learn.microsoft.com/en-us/sql/relational-databases/views/views?view=sql-server-ver17&utm_source=copilot.com

SQL Server Views

https://www.tutorialsteacher.com/sqlserver/views?utm_source=copilot.com

Indexed Views in SQL Server: A Production DBA's Complete Guide

https://dzone.com/articles/indexed-views-in-sql-server?utm_source=copilot.com