Professional DBA Plan for SummitStyle Retail

Mutungi, Vincent

DSCI 723 – Data Management and Warehousing
2025 Spring

24th April 2025

Table of Contents

Table of Figures

Executive Summary

The DBA plan provides a comprehensive system for SummitStyle Retail's database, serving as the foundation for managing sales, customer, and product data. Built on Microsoft SQL Server, it ensures security, efficiency, and scalability to support business growth. The plan includes:

- **User Roles and Access Controls**: Defined roles with specific permissions to ensure secure and appropriate data access.
- **Proactive Maintenance Plan**: Automated tasks like backups and integrity checks to maintain database reliability.
- **ETL and Data Warehousing**: A process to transform transactional data into a data warehouse for analytics.
- **Comprehensive Data Dictionary**: Detailed documentation of database tables for clarity and usability.

This structure enables SummitStyle Retail to leverage data for informed decision-making, streamline operations, and prepare for future expansion.

User Roles and Access Controls

The user roles and access controls section outlines a role-based access control (RBAC) system to secure and optimize SummitStyle Retail's database operations. Six roles are defined to align with the company's workforce needs, ensuring each user has only the necessary permissions:

- **Business Intelligence Analysts**: Granted read-only access to sales and product data, allowing them to query data for insights without modifying records.
- **Data Managers**: Responsible for data quality, with permissions to update and delete records in designated areas like customer and order data.
- **Database Administrators (DBAs)**: Hold full administrative privileges, including creating tables and managing database configurations.
- **SQL Developers**: Authorized to create and modify stored procedures and views, focusing on tool development without direct data manipulation.
- **Department Heads**: Provided access to summary reports and views for high-level decision-making, avoiding raw data complexity.
- **Data Entry Employees**: Permitted to insert and update customer and order information in specific tables, with restrictions on deletions to prevent errors.

This RBAC system is implemented in SQL Server by creating roles and assigning permissions, followed by mapping users (e.g., Alex to BI_Analyst, Sam to Data_Entry) to these roles. The approach ensures a secure, efficient database environment, akin to a well-organized team where each member has a clear role, supporting SummitStyle's operational integrity and growth.

Code snippet sample:

```
-- Create database roles
CREATE ROLE BI_Analyst_DW;
CREATE ROLE Data_Manager_DW;
CREATE ROLE DBA_DW;
CREATE ROLE SQL_Developer_DW;
CREATE ROLE Department_Head_DW;
CREATE ROLE Data_Entry_DW;
GO
-- Grant permissions for BI_Analyst_DW (read-only access to all data)
GRANT SELECT ON SCHEMA::Fact TO BI_Analyst_DW;
GRANT SELECT ON SCHEMA::Dim TO BI_Analyst_DW;
GO
-- Grant permissions for Data_Manager_DW (update dimension tables)
GRANT SELECT, UPDATE ON Dim.CustomerDimension TO Data_Manager_DW;
GRANT SELECT, UPDATE ON Dim.ProductDimension TO Data_Manager_DW;
GRANT SELECT, UPDATE ON Dim.StoreDimension TO Data_Manager_DW;
GRANT SELECT ON Fact.SalesFact TO Data_Manager_DW; -- Read-only for facts
GO
-- Grant permissions for DBA_DW (full control)
GRANT CONTROL ON DATABASE::SummitStyleDW TO DBA_DW;
GO
-- Grant permissions for SQL_Developer_DW (create and alter stored procedures and
views)
GRANT CREATE PROCEDURE, CREATE VIEW TO SQL_Developer_DW;
GRANT ALTER ON SCHEMA::Fact TO SQL_Developer_DW;
GRANT ALTER ON SCHEMA::Dim TO SQL_Developer_DW;
GO
-- Grant permissions for Department_Head_DW (read-only access to summary view)
-- Create a sample view for department heads
CREATE VIEW Fact.SummarySalesByRegion AS
SELECT s.Region, SUM(f.SalesAmount) AS TotalSales
FROM Fact.SalesFact f
JOIN Dim.StoreDimension s ON f.StoreKey = s.StoreKey
GROUP BY s.Region;
GO

GRANT SELECT ON Fact.SummarySalesByRegion TO Department_Head_DW;
GO
-- Grant permissions for Data_Entry_DW (limited access, e.g., insert into staging tables)
```

```
-- Create a sample staging table for data entry (optional, as data entry is less common in DW)
CREATE TABLE Dim.CustomerStaging (
    CustomerID INT,
    CustomerName VARCHAR(100),
    Email VARCHAR(100),
    LoyaltyStatus VARCHAR(50),
    JoinDate DATE
);
GO

GRANT SELECT, INSERT, UPDATE ON Dim.CustomerStaging TO Data_Entry_DW;
DENY DELETE ON Dim.CustomerStaging TO Data_Entry_DW;
GO
```

Basic Maintenance Plan

The maintenance plan is designed to keep SummitStyle Retail's database reliable and performant through three recurring tasks:

- **Database Backups**: Daily full backups protect against data loss from hardware failures or other issues, enabling full database restoration if needed.
- **Updating Statistics**: Weekly updates to statistics ensure the SQL Server query optimizer uses accurate data distribution information, enhancing query performance.
- **Checking Database Integrity**: Monthly integrity checks via DBCC CHECKDB detect and resolve database corruption early, preventing data loss.

These tasks are automated using SQL Server Agent, which schedules jobs and monitors their execution. Automation reduces manual effort, minimizes errors, and ensures consistent database health, allowing SummitStyle to focus on business operations without worrying about underlying system reliability.

ETL/Data Warehousing Considerations

The ETL (Extract, Transform, Load) process integrates data from the transactional database (SummitStyleDB) into the data warehouse (SummitStyleDW) to support business intelligence. The data warehouse features one fact table (Fact.SalesFact) and four dimension tables (Dim.DateDimension, Dim.CustomerDimension, Dim.ProductDimension, Dim.StoreDimension).

- **Extraction**: Data is extracted nightly from SummitStyleDB tables (Sales.SalesOrder, Sales.SalesOrderLine, Production.Product, Sales.Customer, Sales.Store, HR.Employee),

capturing operational details like orders and customer information with minimal system impact.

- **Transformation**: Data is processed to fit the data warehouse's star schema. For Fact.SalesFact, sales line totals and quantities are mapped, linked to dimension keys. Dim.CustomerDimension combines customer names and retains loyalty data. Dim.ProductDimension copies product details, Dim.StoreDimension integrates store and manager information, and Dim.DateDimension generates time attributes for 2023. Data quality measures include deduplication and referential integrity validation.
- **Loading**: Transformed data is inserted into SummitStyleDW, with incremental updates to dimension tables and appends to Fact.SalesFact. Indexes optimize query performance.

Nightly ETL schedules balance data freshness and system load. Data quality checks and error logging ensure reliability, while the star schema's design supports efficient queries. Scalability options, like partitioning Fact.SalesFact, prepare for future growth. This framework enables SummitStyle to analyze sales trends, customer behavior, and regional performance, transforming raw data into strategic insights.

Data Dictionary

1. Sales.SalesOrder

Description: Stores' details of each sales transaction, including customer, store, and order total, to track retail purchases.

Table Name	Column Name	Data Type	Description
Sales.SalesOrder	SalesOrderID	INT	Primary key; unique identifier for each sales order.
	CustomerID	INT	Foreign key; references Sales.Customer.CustomerID; identifies the customer.
	OrderDate	DATETIME	Date and time the order was placed.
	StoreID	INT	Foreign key; references Sales.Store.StoreID; identifies the store location.
	TotalAmount	DECIMAL(10,2)	Total amount of the order, summing all line items.
	OrderStatus	VARCHAR(20)	Status of the order (e.g., Pending, Completed).

2. Sales.SalesOrderLine

Description: Captures individual line items within each sales order, detailing products and quantities sold.

Table Name	Column Name	Data Type	Description
Sales.SalesOrderLine	SalesOrderLineID	INT	Primary key; unique identifier for each order line item (auto-incremented).
	SalesOrderID	INT	Foreign key; references Sales.SalesOrder.SalesOrderID; links to the order.
	ProductID	INT	Foreign key; references Production.Product.ProductID; identifies the product.
	Quantity	INT	Number of units sold in this line item.
	UnitPrice	DECIMAL(10,2)	Price per unit at the time of sale.
	LineTotal	DECIMAL(10,2)	Total for this line item (Quantity * UnitPrice).

3. Production.Product

Description: Maintains the product catalog, including names, categories, and prices, for inventory management.

Table Name	Column Name	Data Type	Description
Production.Product	ProductID	INT	Primary key; unique identifier for each product.
	ProductName	VARCHAR(100)	Name of the product (e.g., Blue T-Shirt, Wireless Headphones).
	CategoryID	INT	Identifier for the product category (e.g., Clothing, Electronics).
	UnitPrice	DECIMAL(10,2)	Standard price per unit of the product.
	ActiveFlag	BIT	Indicates if the product is active (1 = Active, 0 = Inactive).

4. Sales.Customer

Description: Stores customer information, including contact details and loyalty status, for sales and marketing.

Table Name	Column Name	Data Type	Description
Sales.Customer	CustomerID	INT	Primary key; unique identifier for each customer.
	FirstName	VARCHAR(50)	Customer's first name.
	LastName	VARCHAR(50)	Customer's last name.
	Email	VARCHAR(100)	Customer's email address for contact and marketing.
	JoinDate	DATE	Date the customer joined the loyalty program or registered.
	LoyaltyStatus	VARCHAR(20)	Customer's loyalty tier (e.g., Gold, Silver, Regular).

5. HR.Employee

Description: Tracks employee records, including roles and store assignments, for workforce management.

Table Name	Column Name	Data Type	Description
HR.Employee	EmployeeID	INT	Primary key; unique identifier for each employee.
	FirstName	VARCHAR(50)	Employee's first name.
	LastName	VARCHAR(50)	Employee's last name.
	Position	VARCHAR(50)	Employee's role (e.g., Manager, Sales Associate).
	StoreID	INT	Foreign key; references Sales.Store.StoreID; identifies the employee's store.

6. Sales.Store

Description: Contains details of retail store locations, including regions and managers, for operational tracking.

Table Name	Column Name	Data Type	Description
Sales.Store	StoreID	INT	Primary key; unique identifier for each store.
	StoreName	VARCHAR(100)	Name of the store (e.g., Downtown NYC).
	Region	VARCHAR(50)	Geographic region of the store (e.g., Northeast, Midwest).

Table Name	Column Name	Data Type	Description
	OpenDate	DATE	Date the store opened.
	ManagerID	INT	Foreign key; references HR Employee.EmployeeID; identifies the store manager.

7. Fact.SalesFact

Description: Central fact table storing sales metrics, linking to dimensions for business analytics.

Table Name	Column Name	Data Type	Description
Fact.SalesFact	SalesFactID	BIGINT	Primary key; unique identifier for each fact record (auto-incremented).
	OrderID	INT	References Sales.SalesOrder.SalesOrderID; links to the original order.
	CustomerKey	INT	Foreign key; references Dim.CustomerDimension.CustomerKey; identifies the customer.
	ProductKey	INT	Foreign key; references Dim.ProductDimension.ProductKey; identifies the product.
	StoreKey	INT	Foreign key; references Dim.StoreDimension.StoreKey; identifies the store.
	DateKey	INT	Foreign key; references Dim.DateDimension.DateKey; identifies the sale date.
	SalesAmount	DECIMAL(10,2)	Total revenue from the sale (from Sales.SalesOrderLine.LineTotal).
	Quantity	INT	Number of items sold (from Sales.SalesOrderLine.Quantity).

8. Dim.DateDimension

Description: Provides time-based context for sales analysis, enabling breakdowns by date, month, or quarter.

Table Name	Column Name	Data Type	Description
Dim.DateDimension	DateKey	INT	Primary key; unique identifier for each date (e.g., 20230101 for Jan 1, 2023).
	FullDate	DATE	The actual date.
	DayOfMonth	INT	Day of the month (1–31).
	Month	INT	Month number (1–12).
	MonthName	VARCHAR(10)	Name of the month (e.g., January).
	Quarter	INT	Quarter of the year (1–4).
	Year	INT	Year of the date.
	DayOfWeek	VARCHAR(10)	Day of the week (e.g., Monday).

9. Dim.CustomerDimension

Description: Enables customer-based sales analysis, including by loyalty status or join date.

Table Name	Column Name	Data Type	Description
Dim.CustomerDimension	CustomerKey	INT	Primary key; surrogate key for the customer in the data warehouse.
	CustomerID	INT	Original CustomerID from Sales.Customer; links to transactional data.
	CustomerName	VARCHAR(100)	Concatenated FirstName and LastName from Sales.Customer.
	Email	VARCHAR(100)	Customer's email address.
	Address	VARCHAR(200)	Customer's address (NULL if not provided).
	LoyaltyStatus	VARCHAR(50)	Customer's loyalty tier (e.g., Gold, Silver, Regular).
	JoinDate	DATE	Date the customer joined.

10.Dim.ProductDimension

Description: Supports product-based sales analysis, such as by category or price range.

Table Name	Column Name	Data Type	Description
Dim.ProductDimension	ProductKey	INT	Primary key; surrogate key for the product in the data warehouse.

Table Name	Column Name	Data Type	Description
	ProductID	INT	Original ProductID from Production.Product; links to transactional data.
	ProductName	VARCHAR(100)	Name of the product.
	CategoryID	INT	Product category identifier.
	UnitPrice	DECIMAL(10,2)	Standard price per unit.
	ActiveFlag	BIT	Indicates if the product is active (1 = Active, 0 = Inactive).

11.Dim.StoreDimension

Description: Facilitates store-based sales analysis, including by region or manager.

Table Name	Column Name	Data Type	Description
Dim.StoreDimension	StoreKey	INT	Primary key; surrogate key for the store in the data warehouse.
	StoreID	INT	Original StoreID from Sales.Store; links to transactional data.
	StoreName	VARCHAR(100)	Name of the store.
	Region	VARCHAR(50)	Geographic region of the store.
	OpenDate	DATE	Date the store opened.
	ManagerName	VARCHAR(100)	Concatenated FirstName and LastName of the store manager (NULL if none).

Appendix

1. Appendix A: Example Queries

The following SQL queries illustrate how to interact with the SummitStyle Retail database:

Query 1: Top 5 Selling Products (Transactional Database)

```
SELECT p.ProductID, p.ProductName, SUM(sol.Quantity) AS TotalQuantitySold,
SUM(sol.LineTotal) AS TotalRevenue
FROM Sales.SalesOrderLine sol
JOIN Production.Product p ON sol.ProductID = p.ProductID
GROUP BY p.ProductID, p.ProductName
ORDER BY TotalRevenue DESC
TOP 5;
```

Description: This query identifies the top 5 products by total revenue in the transactional database, using Sales.SalesOrderLine and Production.Product. It helps SummitStyle Retail prioritize inventory and marketing for high-performing products.

Query 2: Recent Orders by Store (Transactional Database)

```
SELECT s.StoreID, s.StoreName, so.SalesOrderID, so.OrderDate, so.TotalAmount
FROM Sales.SalesOrder so
JOIN Sales.Store s ON so.StoreID = s.StoreID
WHERE so.OrderDate >= DATEADD(DAY, -7, GETDATE())
ORDER BY so.OrderDate DESC;
```

Description: This query retrieves sales orders from the past week, joining Sales.SalesOrder and Sales.Store in the transactional database. It supports store managers in monitoring recent sales activity.

Query 3: Sales by Customer Loyalty Status (Data Warehouse)

```
SELECT c.LoyaltyStatus, COUNT(f.SalesFactID) AS NumberOfSales,
SUM(f.SalesAmount) AS TotalSalesAmount
FROM Fact.SalesFact f
JOIN Dim.CustomerDimension c ON f.CustomerKey = c.CustomerKey
GROUP BY c.LoyaltyStatus
ORDER BY TotalSalesAmount DESC;
```

Description: This query analyzes sales by customer loyalty status (Gold, Silver, Regular) using Fact.SalesFact and Dim.CustomerDimension in the data warehouse. It helps identify which loyalty tiers drive the most revenue.

Query 4: Monthly Sales Trends by Region (Data Warehouse)

```
SELECT d.Year, d.MonthName, s.Region, SUM(f.SalesAmount) AS TotalSalesAmount,
SUM(f.Quantity) AS TotalQuantitySold
FROM Fact.SalesFact f
```

```
JOIN Dim.DateDimension d ON f.DateKey = d.DateKey
JOIN Dim.StoreDimension s ON f.StoreKey = s.StoreKey
GROUP BY d.Year, d.MonthName, s.Region
ORDER BY d.Year, d.Month, s.Region;
```

Description: This query aggregates sales by month and store region, using Fact.SalesFact, Dim.DateDimension, and Dim.StoreDimension in the data warehouse. It enables SummitStyle to track regional sales trends over time.

Query 5: Top Product Categories by Sales (Data Warehouse)

```
SELECT p.CategoryID, COUNT(f.SalesFactID) AS NumberOfSales,
   SUM(f.SalesAmount) AS TotalSalesAmount
FROM Fact.SalesFact f
JOIN Dim.ProductDimension p ON f.ProductKey = p.ProductKey
GROUP BY p.CategoryID
ORDER BY TotalSalesAmount DESC;
```

Description: This query ranks product categories by total sales revenue, using Fact.SalesFact and Dim.ProductDimension in the data warehouse. It helps SummitStyle optimize product category strategies.

2. Appendix B: Schema Diagram Placeholder

SummitStyle Transactional Database Diagram

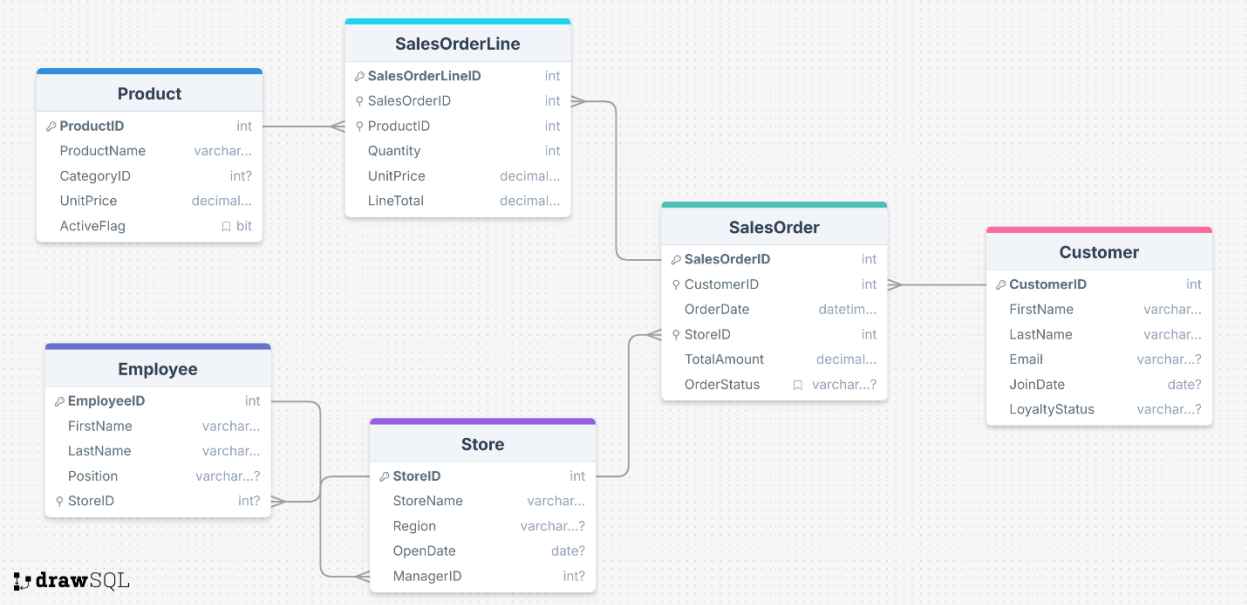

Figure 1: Transactional Database Schema

SummitStyle Database Warehouse Diagram

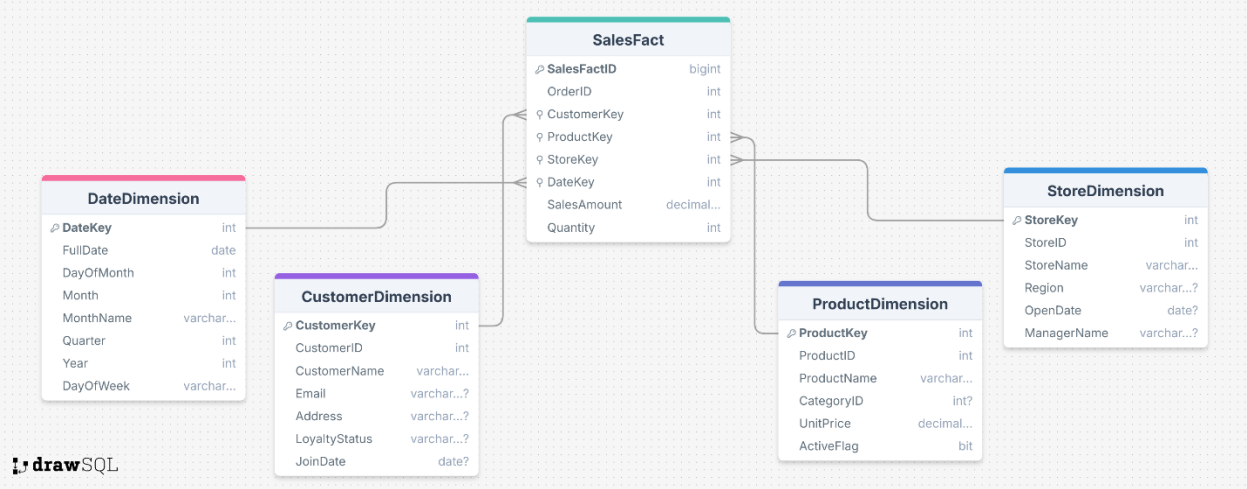

Figure 2: Database Warehouse Schema