**🧱 6-Week SQL for Data Engineering (SQL Server Focus)**

**🎯 Overall Goal**

Master the SQL concepts and patterns used in data pipelines:

* Complex transformations
* Data modeling
* Performance tuning
* Window functions
* SCDs, CDCs, ETL logic
* Real-world warehouse architecture

**🗓️ WEEK 1 — Data Engineering SQL Foundations**

**Goal:** Learn how data engineers *think in SQL* — sets, pipelines, transformations — not CRUD.

**🧠 Theory**

* Set-based vs procedural thinking in SQL
* How SQL Server processes queries (logical order of execution)
* Query plans & execution flow: FROM → JOIN → WHERE → GROUP BY → HAVING → SELECT → ORDER BY
* ACID properties and transaction concepts (just enough for analytics context)
* Temporary objects (CTE, temp tables, table variables)

**🧰 Practice**

* Create CTE-based pipelines (step-by-step transformations)
* Build multi-step transformation queries instead of one massive SELECT
* Compare performance of:
  + CTE vs Temp Table
  + Derived table vs Subquery

**⚙️ Assignment**

**Project:** Build a *Data Transformation Chain*  
Load raw sales data → clean → join with products → aggregate → output summary view.

**🗓️ WEEK 2 — Advanced Joins, CASE Logic & Conditional Aggregation**

**Goal:** Learn how to manipulate and merge datasets like a data pipeline.

**🧠 Theory**

* Deep join mechanics in SQL Server (nested loops, hash joins, merge joins)
* Conditional joins (e.g., join only latest record or date ranges)
* Conditional aggregation (CASE WHEN inside SUM/COUNT)
* The importance of null-safe joins
* Anti-joins & semi-joins (NOT EXISTS, EXISTS)

**🧰 Practice**

* Write queries that:
  + Identify customers with multiple purchases but no returns.
  + Join fact and dimension tables conditionally (e.g., current active products only).
  + Use CASE for dynamic grouping (e.g., categorize sales into Low/Med/High).

**⚙️ Assignment**

**Project:** Create a “Business Metrics View” using CASE + Joins to produce KPIs:

* Total sales, average order, churned customers, etc.

**🗓️ WEEK 3 — Window Functions & Analytics**

**Goal:** Master the analytical backbone of SQL — ranking, moving averages, cumulative totals.

**🧠 Theory**

* Windowing logic: OVER(), PARTITION BY, ORDER BY
* Types: Ranking (ROW\_NUMBER, RANK, DENSE\_RANK), Aggregation (SUM, AVG), Navigation (LAG, LEAD)
* Frame clauses: ROWS BETWEEN, cumulative logic
* Use cases: sessionization, time-series, trend analysis
* How SQL Server executes window functions (spooling, partitioning)

**🧰 Practice**

* Calculate rolling 7-day sales averages.
* Identify customer’s first and last purchase dates.
* Create rank-based segments (top 10%, etc.)
* Compare ROW\_NUMBER() vs RANK() in duplicate-handling scenarios.

**⚙️ Assignment**

**Project:** *Customer Lifetime Analysis*  
Use window functions to find:

* Customer retention patterns
* Average order frequency
* Time between purchases

**🗓️ WEEK 4 — Data Modeling, ETL & Warehouse Concepts**

**Goal:** Go from querying to designing — understand how SQL underpins pipelines.

**🧠 Theory**

* OLTP vs OLAP schema design
* Star & Snowflake schema — and how to query them efficiently
* Fact and Dimension tables
* Surrogate vs Natural keys
* Normalization (up to 3NF) vs Denormalization (for warehouses)
* ETL flow: Staging → Cleansing → Transformation → Fact/Dim load
* Using MERGE for UPSERTS

**🧰 Practice**

* Build small star schema (Sales Fact + Customer/Date/Product Dimensions)
* Write ETL queries to populate Fact table:
* MERGE INTO FactSales AS T
* USING StageSales AS S
* ON T.SaleID = S.SaleID
* WHEN MATCHED THEN UPDATE SET ...
* WHEN NOT MATCHED THEN INSERT (...)
* Practice using surrogate keys and surrogate date dimension tables.

**⚙️ Assignment**

**Project:** *Mini Data Warehouse*  
Simulate a daily ETL load:

1. Stage new data
2. Transform and deduplicate
3. Merge into FactSales

**🗓️ WEEK 5 — Optimization, Indexing, and Query Tuning**

**Goal:** Think like the SQL Server engine — performance, indexing, and scalability.

**🧠 Theory**

* How SQL Server optimizes queries (statistics, cost-based optimizer)
* Clustered vs Non-clustered indexes
* Covering indexes
* Execution plan basics (seek vs scan)
* Common performance killers: implicit conversions, scalar functions, cursors
* CTE vs Temp Tables vs Indexed Views trade-offs

**🧰 Practice**

* Compare query times with and without indexes.
* View actual execution plan → analyze operator cost.
* Rewrite subqueries into CTEs for optimization.
* Experiment with query hints (OPTION (RECOMPILE), FORCESEEK) and note impact.

**⚙️ Assignment**

**Project:** *Query Performance Lab*  
Optimize a slow reporting query (multi-joins + aggregations).  
Document tuning steps: added indexes, rewrote logic, reduced cost by X%.

**🗓️ WEEK 6 — Slowly Changing Dimensions, CDC & Advanced ETL Patterns**

**Goal:** Master the change-tracking and history-management side of SQL pipelines.

**🧠 Theory**

* SCD Types (1, 2, 3) and when to use them
* Implementing SCD Type 2 manually in SQL Server
* Change Data Capture (CDC) in SQL Server
* Using Timestamps and Hashing for incremental loads
* Data quality enforcement (constraints, surrogate timestamps)
* Handling late-arriving facts

**🧰 Practice**

* Build SCD Type 2 logic:
* UPDATE DimCustomer
* SET EndDate = GETDATE(), IsCurrent = 0
* WHERE CustomerID = @CustomerID AND IsCurrent = 1
* INSERT INTO DimCustomer (...)
* VALUES (...)
* Create CDC-enabled table and track inserts/updates.
* Implement incremental load from staging to warehouse.

**⚙️ Assignment**

**Project:** *Historical Data Tracking Pipeline*  
Build a mini pipeline that:

* Detects changes in dimension data
* Preserves history (SCD Type 2)
* Loads incremental changes to Fact table

**📚 Optional Theoretical Reinforcements (for deeper mastery)**

* **Books:**
  + *SQL Performance Explained* by Markus Winand
  + *Data Warehouse Toolkit* by Ralph Kimball
  + *Inside Microsoft SQL Server T-SQL Querying* by Itzik Ben-Gan
* **Concepts to Review:**
  + Query optimizer internals
  + Dimensional modeling principles
  + ETL framework architecture

**✅ Final Outcome**

After 6 weeks, you’ll be able to:

* Design fact/dimension schemas
* Write ETL and incremental SQL pipelines
* Tune and index for performance
* Implement SCD Type 2 and CDC logic
* Read and interpret execution plans
* Integrate SQL transformations with Airflow and Python