BDTT

Week 5

Lakehouse in Databricks

& Spark SQL

2025

1. Learning Outcomes

- To learn what is the Lakehouse concept and its relationship with the Databricks platform
- To learn databases, tables and views in Databricks
- To learn how SQL queries can be written and executed in Spark SQL

2. The Concept of Data Storage and Processing Architectures

Data warehouses, data lakes, and lakehouses all belong to a **broader category of data storage and processing architectures**, which define how organisations collect, store, manage, and analyse data to support decision-making, analytics, and AI-driven applications. These architectures provide the foundation for modern data ecosystems, determining how data is structured, accessed, and utilised across different industries.

At their core, these architectures serve as centralised data repositories that enable businesses to integrate information from various sources, ensuring accessibility, reliability, and performance. They differ in how they handle data structure, governance, scalability, and processing efficiency, but they all share the goal of organizing and optimizing data for analytical and operational use. Over time, these systems have evolved, moving from rigid, structured environments to more flexible and scalable solutions, allowing organisations to balance performance, cost-efficiency, and real-time insights.

Much like different types of vehicles in transportation, where some are built for speed (sports cars), others for heavy loads (trucks), and some for all-terrain versatility (SUVs), data architectures vary in their approach to handling structured, semi-structured, and unstructured data. However, they all belong to the same broader category of **data management frameworks**, designed to store, process, and extract value from data efficiently.

3. What is a Lakehouse

Lakehouse technology is a modern data architecture that **combines the best features of data lakes and data warehouses**, offering a unified platform for handling structured, semi-structured, and unstructured data. **Traditional data lakes** store vast amounts of raw data but often suffer from data quality and governance issues, while data warehouses provide high-performance analytics but struggle with scalability and flexibility. The lakehouse model bridges this gap by supporting open data formats, transactional capabilities, and governance mechanisms, enabling efficient data management and real-time analytics without the need for complex data movement between separate systems.

One of the key advantages of lakehouse technology is its ability to support multiple workloads, including business intelligence (BI), machine learning (ML), and real-time streaming analytics, all from a single platform. By leveraging a scalable storage layer with structured metadata and schema enforcement, lakehouses ensure data consistency while maintaining the cost-effectiveness and flexibility of a data lake. Additionally, they integrate with modern data processing frameworks such as Apache Spark, Delta Lake, and Iceberg, enhancing data reliability and performance. As organisations increasingly deal with vast and diverse datasets, lakehouse technology is becoming a preferred solution for achieving a balance between performance, scalability, and governance in data management.

4. Lakehouse vs other technologies

Let’s think of data storage and processing architectures/technologies like different ways to manage a big collection of books in a library. The goal is the same, storing information so it can be easily accessed, but the approach varies.

Data Warehouse, Like a Well-Organised Library:

A data warehouse is like a highly organised library, where every book is neatly categorised, labelled, and placed on a specific shelf. Before a book is added, it must be cleaned, formatted, and structured according to strict rules. This makes it easy and fast to find exactly what you need, but the system is rigid and costly, and adding new types of books (data) takes a lot of effort.

Best for: Business reports, financial analytics, dashboards.
Limitations: Expensive, slow to adapt to new data types.

Examples: Snowflake, Google BigQuery, Amazon Redshift.

Data Lake, Like a Giant Storage Room

A data lake is like a huge, messy storage room where you can dump books, magazines, handwritten notes, and even audio recordings. Nothing needs to be sorted or labelled upfront, you just throw everything in. While this makes it cheap and flexible, finding a specific book later is a nightmare unless you have the right tools (advanced search and filtering).

Best for: Storing raw, large-scale data for AI, machine learning, and big data projects.
Limitations: Can become a "data swamp" if not managed properly, slow performance for analytics.

Examples: Amazon S3, Azure Data Lake, Google Cloud Storage.

Lakehouse: Like a Smart, and Modern Library

A lakehouse is the best of both worlds, a smart library where you can store all kinds of books (structured and unstructured data) without losing organisation and accessibility. It allows you to quickly find books like in a warehouse but also gives the flexibility of a lake to store any type of content. It achieves this by adding structure and reliability to the raw data, making it easier to use for reporting, AI, and advanced analytics, all in one place.

Best for: A unified data system for analytics, AI, and real-time decision-making.
Limitations: Newer technology, still evolving.

Examples: Databricks Lakehouse, Apache Iceberg, Delta Lake.

5. Relationship of Lakehouse with Databricks Platform

The lakehouse architecture is closely associated with the Databricks platform, which pioneered this concept to unify data lakes and data warehouses into a single, scalable system. Databricks introduced Delta Lake, an open-source storage layer that brings ACID (Atomicity, Consistency, Isolation, Durability) transactions, schema enforcement, and time travel capabilities to traditional data lakes, effectively transforming them into

reliable and high-performance lakehouses. By integrating Delta Lake with Apache Spark and other data processing frameworks, Databricks provides a robust foundation for big data analytics, AI, and machine learning, enabling organisations to manage and process large datasets efficiently.

Databricks extends the lakehouse concept through its Lakehouse Platform, which offers end-to-end data engineering, analytics, and machine learning workflows within a unified environment. This eliminates the need for separate data pipelines between data lakes and warehouses, reducing complexity and cost. Moreover, Databricks supports multi-cloud deployments, enabling businesses to leverage its lakehouse architecture across AWS, Azure, and Google Cloud. With its focus on collaborative analytics, governance, and open-source standards, Databricks has become a leading choice for organisations looking to implement scalable, high-performance, and cost-effective data solutions using the lakehouse paradigm.

6. What is Spark SQL

Imagine you have huge amounts of data, too big for Excel or even a traditional database. You need a way to organise, filter, and analyse it quickly. That's where Spark SQL comes in.

Spark SQL is like a supercharged version of SQL (Structured Query Language) that works on big data. It lets you write familiar SQL queries but runs them on massive datasets spread across multiple computers (clusters). This makes data processing much faster and scalable, whether you're dealing with millions or billions of records.

Why Use Spark SQL?

- **Easy to Use:** If you know SQL, you can use Spark SQL right away.
  Works with Big Data: Can handle huge datasets that normal databases struggle with.
- **Super Fast:** Uses Spark's powerful computing engine to process data quickly.
- **Flexible:** Works with structured data (tables), semi-structured data (JSON, CSV), and databases.
- **Works with Multiple Languages:** You can run queries in Python, Java, Scala, and R.

How Does Spark SQL Work?

- **Data is Loaded**: You can bring in data from files (CSV, JSON, Parquet), databases (MySQL, PostgreSQL), or data lakes (Amazon S3, Azure Blob Storage).
- **Run SQL Queries**: Just like in a normal database, you can use SELECT, WHERE, JOIN, GROUP BY to filter and analyse data.

```
spark.sql("SELECT name, age FROM people WHERE age > 30")
```

- **Spark Processes It Fast:** Unlike traditional databases, Spark splits the work across multiple computers, so even huge queries run quickly.
- **Get the Results**: You can save the cleaned data, use it in reports, or even train AI models with it.

Where is Spark SQL Used?

- **Business Analytics**: Finding sales trends, customer behaviour, and performance insights.
- **E-commerce & Retail**: Tracking customer purchases and recommendations.
- **Big Data Processing**: Cleaning, organising, and preparing huge datasets.
- **Machine Learning & AI**: Pre-processing data for AI models.

Why is Spark SQL Popular?

Because it's easy to learn, super fast, and built for big data! Whether you're working with millions of customer records, analysing website traffic, or preparing data for AI, Spark SQL is a powerful tool that helps businesses and data teams process and understand data efficiently.

Spark SQL vs Traditional SQL?

Feature	Traditional SQL (Databases)	Spark SQL (Big Data)
Data Size	Works on small to medium datasets	Handles massive datasets (billions of rows)
Speed	Slower for big data	Super fast (runs queries in parallel)
Scalability	Limited to one server	Scales across many computers
Flexibility	Works only with structured data	Works with structured + semi-structured data
Use Case	Small company reports	Big data analytics, AI, and cloud computing

7. How Spark SQL and Databricks Are Connected

Think of Spark SQL as a powerful engine that helps you run SQL queries on massive amounts of data. Now, imagine Databricks as a modern, high-tech car that is built around this engine, making it easier to use, faster, and more efficient. Let's briefly review the Spark SQL and Databricks concept.

What is Spark SQL?

- It's a tool that lets you run SQL queries on big data just like you would in a normal database, but much faster.
- It can handle huge datasets spread across multiple computers.
- It works with structured data (tables) and semi-structured data (JSON, CSV, etc.).

What is Databricks?

- It’s a cloud-based platform built on top of Apache Spark.
- It makes using Spark SQL much easier by providing a friendly interface, automation, and performance improvements.
- Think of it as "Spark SQL on steroids" – faster, more organised, and optimised for large-scale business use.

How Do They Work Together?

Spark SQL is the Engine, Databricks is the Car

- Spark SQL processes the data, but Databricks makes it easier to use, manage, and scale.

Databricks Adds Superpowers to Spark SQL

- Databricks optimises Spark SQL, making it faster and more efficient.
- It provides a simple interface where users can write SQL queries easily.
- It integrates with cloud storage (AWS, Azure, Google Cloud) for seamless big data processing.

Databricks is Like a Managed Service for Spark SQL

- Instead of setting up and managing Apache Spark manually, Databricks does it for you.
- It handles infrastructure, performance tuning, and security.

Why Use Databricks with Spark SQL?

- **Faster Queries:** Databricks makes Spark SQL run much faster.
- **Easier to Use:** No need for complicated setups; Databricks provides a simple interface.
- **Scales Automatically:** Handles large datasets with ease, without manual tuning.
- **Better Collaboration:** Teams can share and work on data together in one place.

Do You Need Databricks for Spark SQL?

No, you can use Spark SQL without Databricks, but it will be harder to set up, maintain, and optimise. Databricks just makes everything easier, faster, and more efficient, especially for businesses, AI projects, and large-scale analytics.

Relation of Spark SQL and Databricks

Feature	Spark SQL (Engine)	Databricks (Car)
Purpose	Runs SQL on big data	Makes using Spark SQL easier and faster
Setup	Needs manual configuration	Ready to use with built-in optimisations
Speed	Fast, but needs tuning	Super fast with automatic optimisations
User-Friendly?	Requires technical expertise	Easier to use, even for non-tech users
Best For	Developers & data engineers	Businesses, analysts & AI teams