SQL UDFs, JSON Handling & Advanced Delta Lake Features

Agenda

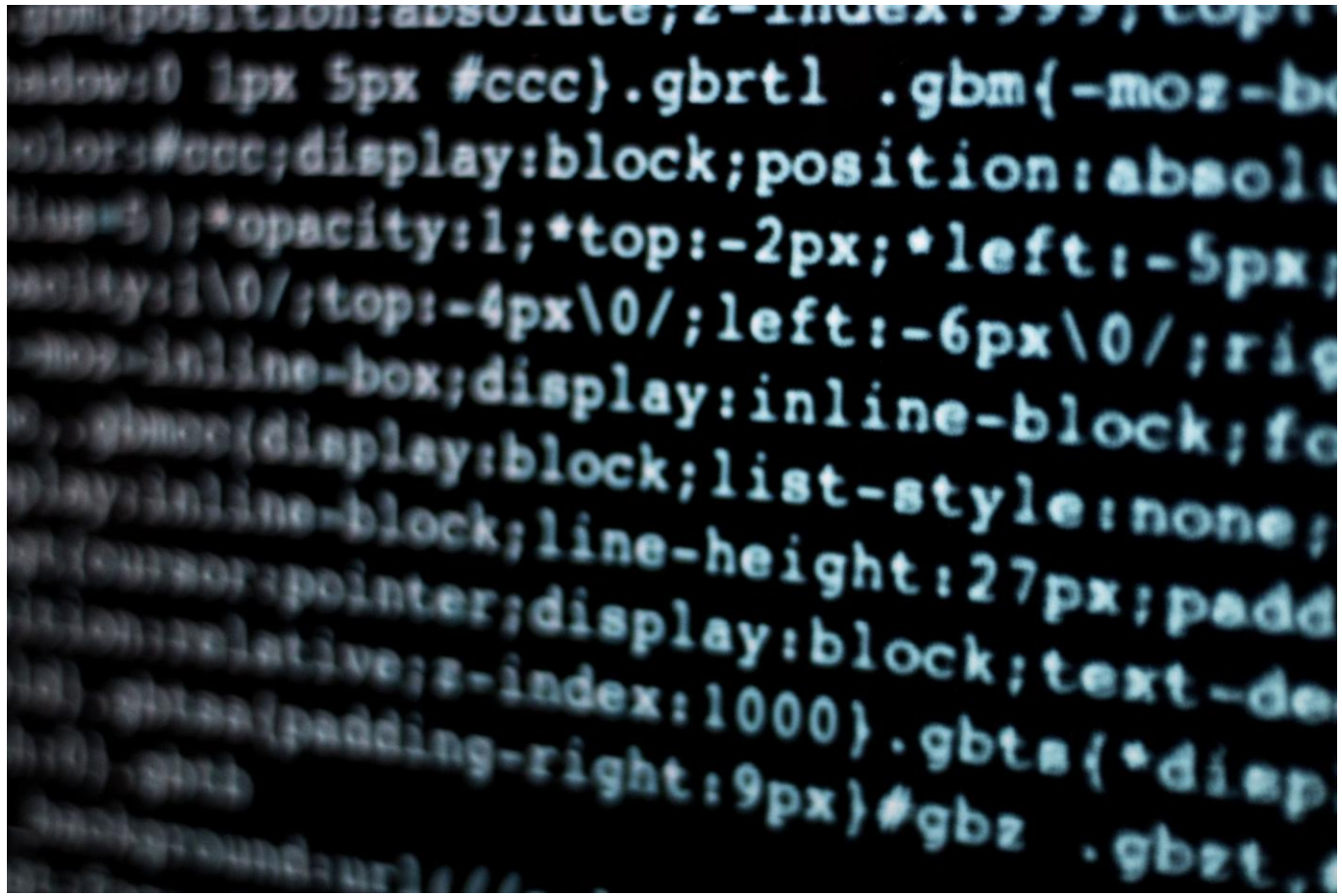
- Introduction
  - Title Slide
  - Agenda
- SQL UDFs
  - What are SQL UDFs?
  - Creating SQL UDFs in Databricks
- Handling JSON Data
- Advanced Delta Lake Features
- Hands-On Labs and Demo

What are SQL UDFs?

- User-Defined Functions (UDFs)
  - Allow custom logic inside SQL queries
  - Useful when built-in functions don’t meet specific needs
- Types of UDFs
  - Scalar: returns a single value
  - Table-valued: returns a table

Creating SQL UDFs in Databricks

- Syntax to create a SQL scalar UDF
  - CREATE OR REPLACE FUNCTION my_upper(str STRING) RETURNS STRING RETURN UPPER(str);
- Use in queries
  - SELECT my_upper(CustomerName) FROM sales_delta;
- Benefits of UDFs
  - Simplify repetitive transformations
  - Improve query readability



Handling JSON Data

- JSON for Semi-Structured Data
  - Common format for semi-structured data
- Delta Lake JSON Query Support
  - Use from_json() to parse JSON
  - Use get_json_object() to extract JSON fields
- Schema Definition
  - Define schema to parse JSON strings
  - Convert JSON strings into structured columns

Advanced Delta Lake Features

- Schema Evolution
  - Add or modify columns without downtime
- Time Travel
  - Query past table versions for auditing or rollback
- Optimize & Z-Ordering
  - Improve query performance by clustering data

Hands-On Labs and Demo

- Create and use SQL UDFs for string manipulation
  - Hands-on practice with SQL UDFs
- Parse and query JSON columns in Delta tables
  - Learn to handle JSON data in Delta tables
- Use schema evolution to add columns to a Delta table
  - Understand schema evolution in Delta tables
- Query older versions using time travel
  - Explore time travel queries in Delta tables
- Optimize Delta tables with Z-ordering
  - Improve query performance with Z-ordering