#FABCONSQLCON2026

FABCON | SQLCON

Microsoft Fabric COMMUNITY CONFERENCE

Microsoft SQL COMMUNITY CONFERENCE

ATLANTA MARCH 16 - 20, 2026

Own Your Workflow: Build Notebooks Locally & Deploy Them to Fabric

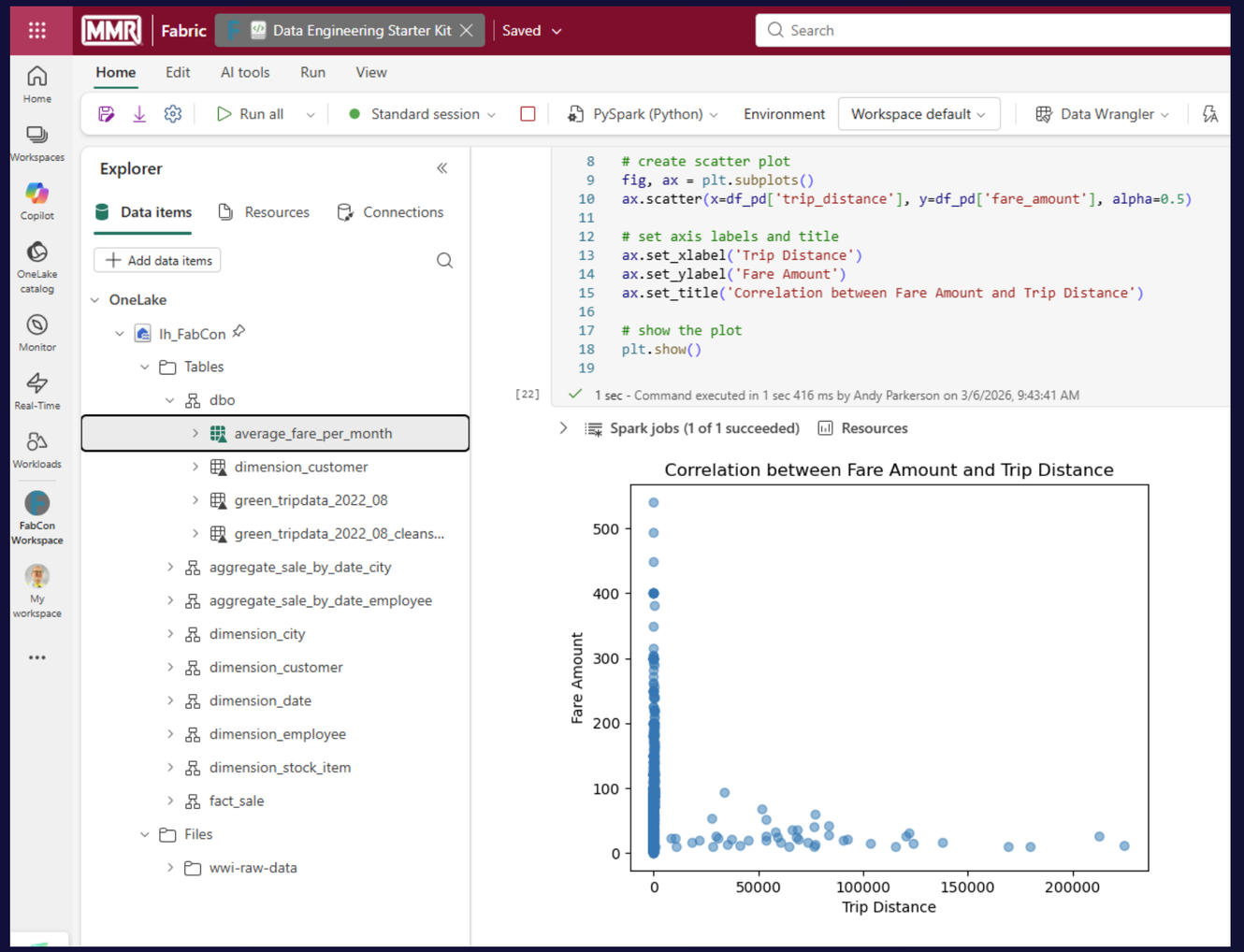
- Andy Parkerson
  Data Engineer
  MMR Group
  andyparkerson@gmail.com
  linkedin.com/in/andyparkerson

Microsoft Fabric

- Power BI
- Jupyter Notebooks
- Pipelines
- OneLake storage
- Delta Lake
- Warehouse
- Eventhouse

Fabric Notebooks

- Connect to Data
- Code
  - Python
  - PySpark
  - Spark SQL
- Markdown
- Visualizations



Why Jupyter Notebooks?

- Julia
- Python
- R

Ju – Pyt – eR

Why local-first?

- Faster edit–run loop
- Familiar tooling
- Lighter capacity usage
- Use Fabric for scale, governance, scheduling

What we’ll build

- Local VS Code + PySpark + Delta
- A small medallion-style ETL notebook
- Deploy to Fabric and run it there

What we'll need

- Python 3.12+
- Java 17
- PySpark 3.5.1
- delta-spark 3.1.0
- VS Code + Fabric extension (GA)
- Access to a Fabric workspace + capacity
- OneLake File Explorer for easy file moves on Windows

Package Installation

```
> winget install --id=Python.Python.3.12 -e

> $env:PYSPARK_HADOOP_VERSION="3"

> pip install “pyspark==3.5.1”

> pip install “delta-spark==3.1.0”
```

Package Installation - Spark

- https://spark.apache.org/downloads.html
- Spark release 3.5.8
- Download Pre-built for Apache Hadoop 3.3 and later
- Unzip into “C:\Users\your_name\spark\spark_3.5”

Package Installation - Hadoop

- https://github.com/robguilarr/spark-winutils-3.3.1
- Hadoop 3.4.0
- Clone repo or download
- Select files in hadoop-3.4.0-win10-x64/bin directory
- Copy into “C:\Users\your_name\spark\Hadoop\bin”
- Copy hadoop.dll into “C:\Windows\system32” folder (?)

Package Installation - Environment

```
> setx JAVA_HOME “CK\Program Files\Exclipse Adoptium\jpk-17”

> setx SPARK_HOME “C:\Users\your_name\spark\spark-3.5.8-bin-hadoop3\”

> setx HADOOP_HOME “C:\Users\your_name\spark\Hadoop\”
```

Package Installation - VSCode

- https://code.visualstudio.com/download
- https://marketplace.visualstudio.com/items?itemName=fabric.vscode-fabric
  - Microsoft Fabric Extension
- https://marketplace.visualstudio.com/items?itemName=SynapseVSCode.vscode-synapse-remote
  - Fabric Data Engineering Extension
- https://marketplace.visualstudio.com/items?itemName=ms-toolsai.jupyter
  - Jupyter notebook Extension

Demo

Demo: Create a local “lakehouse” with Delta

Create local lakehouse

In this notebook we will create a lakehouse locally, and then do some cool things.

Generate | + Code | + Markdown

```python
from delta import configure_spark_with_delta_pip
from pyspark.sql import SparkSession

# Stop any existing session first (if running inside a notebook)
try:
    spark.stop()
except:
    pass

builder = (
    SparkSession.builder
    .master("local[*]")
    .appName("delta-pip-test")
    # Delta
    .config("spark.sql.extensions", "io.delta.sql.DeltaSparkSessionExtension")
    .config("spark.sql.catalog.spark_catalog", "org.apache.spark.sql.delta.catalog.DeltaCatalog")
    # Local dirs (Windows-friendly)
    .config("spark.sql.warehouse.dir", r"C:\dev\github\FabCon\lakehouse")
    .config("spark.local.dir", r"C:\dev\github\FabCon\spark_tmp")
    .config("spark.driver.host", "localhost")
    .config("spark.driver.bindAddress", "127.0.0.1")
)

spark = configure_spark_with_delta_pip(builder).getOrCreate()
```

✓ 13.8s Python

Demo continued 2/3

```python
# Bronze: ingest CSV
df = spark.read.parquet(r"C:\dev\github\FabCon\lakehouse\Files\green_tripdata_2022-08.parquet")


from pathlib import Path

table_name = "green_tripdata_2022_08"
base = Path(r"C:\dev\github\FabCon\lakehouse\Tables")
base.mkdir(parents=True, exist_ok=True)
table_path = (base / table_name).as_posix()

(
    df.write
      .format("delta")
      .mode("overwrite")
      .option("overwriteSchema", "true")
      .save(table_path)
)

spark.sql(f"DROP TABLE IF EXISTS {table_name}")
spark.sql(f"CREATE TABLE {table_name} USING DELTA LOCATION '{table_path}'")

spark.table(table_name).show(5, truncate=False)
```

✓ 38.2s Python

Demo continued 3/3

Scenario 1: Data cleaning and transformation

In this scenario, the data engineer could perform some data cleaning and transformation tasks to prepare the data for downstream analysis. For example, they could remove any invalid or missing data, convert the data types of some columns, and add some new calculated columns based on the existing data. This could involve using Spark SQL queries, user-defined functions (UDFs), or built-in Spark functions to manipulate the data.

```python
from pyspark.sql.functions import col, when

# Load data from source
df = spark.table("green_tripdata_2022_08")
df_count = df.count()

print(f"Total records before cleansing: {df_count}")

# Remove invalid records
df = df.filter((col("trip_distance") > 0) & (col("fare_amount") > 0))
df_count_after_clearning = df.count()

number_of_deleted_records = df_count - df_count_after_clearning

print(f"Removed {number_of_deleted_records} records")

# # Cleanse data
df = df.withColumn("store_and_fwd_flag", when(col("store_and_fwd_flag") == "Y", True).otherwise(False))
df = df.withColumn("lpep_pickup_datetime", col("lpep_pickup_datetime").cast("timestamp"))
df = df.withColumn("lpep_dropoff_datetime", col("lpep_dropoff_datetime").cast("timestamp"))

# Display cleansed data to destination
display(df)

# Write cleansed data to destination
table_name = "green_tripdata_2022_08_cleansed"
table_path = (base / table_name).as_posix()

(
    df.write
      .format("delta")
      .mode("overwrite")
      .option("overwriteSchema", "true")
      .save(table_path)
)

spark.sql(f"DROP TABLE IF EXISTS {table_name}")
spark.sql(f"CREATE TABLE {table name} USING DELTA LOCATION '{table path}'")
```

✓ 11.5s Python

Develop in VS Code (Fabric Extension)

- Explore Fabric workspaces
- Clone Git-enabled workspaces
- Edit item definitions
- Manage items
- Extension is Gorgeously Awesome (GA).

Demo 2

Deploy to Fabric — three paths

- A. Git integration (Azure DevOps & GitHub supported)
  - Commit locally → sync in Fabric
  - Notebooks stored as source files (e.g., notebook-content.py)
- B. VS Code extension
  - Create/update items (including notebooks) directly via item definitions
- C. REST API
  - Create/update notebooks
  - Run on demand (service principal supported)

CI/CD options

- Git Integration (Azure DevOps, GitHub) + Deployment Pipelines
  - Treat Git as the source of truth
  - Promote Dev → Test → Prod.
- Community accelerator (fabric-toolbox) for Git-based deployments and automation patterns.
- REST APIs or VS Code extension item definitions for scripted deployments

Common pitfalls & how to avoid them

- Spark/Delta version mismatch
  - Verify the compatibility matrix in Delta docs
  - Start Spark with Delta extensions enabled
- Windows specifics
  - If using manual Spark binaries, set `JAVA_HOME`, `SPARK_HOME`, and (sometimes) `HADOOP_HOME` for winutils
  - Prefer `pip install pyspark` for simplicity
- Notebook type selection
  - Python vs PySpark
  - Use Python for in-memory/smaller data (starter pool ~seconds)
  - Use PySpark for large/parallel workloads
- OneLake File Explorer is preview
  - Be aware of sync behavior, naming limitations, and update prompts; latest installer on Microsoft Download Center
- Environment management
  - Put libraries in a Fabric Environment and attach to the notebook
  - version in Git
  - Publish after update.

Own Your Workflow: Build Notebooks Locally & Deploy Them to Fabric

- Andy Parkerson
  Data Engineer
  MMR Group
  andyparkerson@gmail.com
  linkedin.com/in/andyparkerson

Sound off.
The mic is all yours.
Influence the product roadmap.

Join the Fabric User Panel

Share your feedback directly with our Fabric product group and researchers.

https://aka.ms/JoinFabricUserPanel

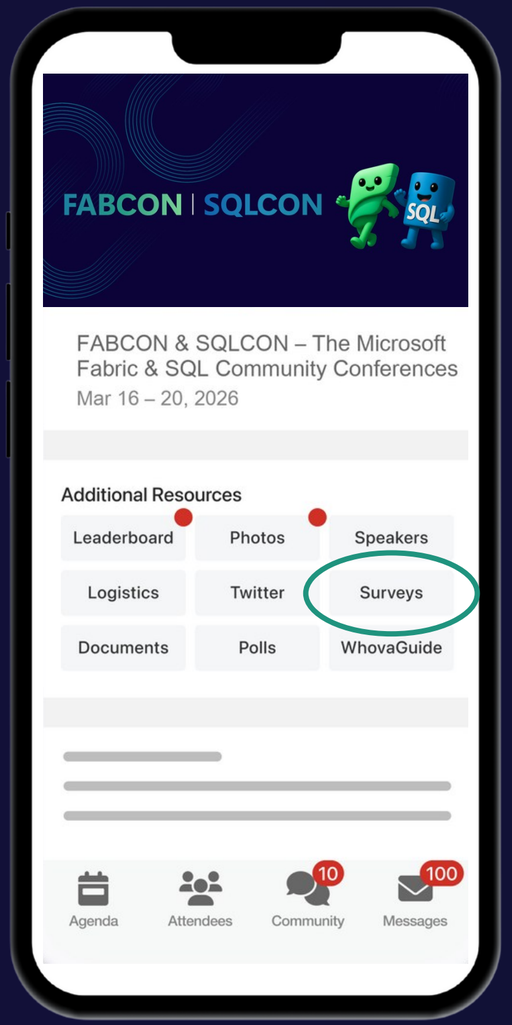
Join the SQL User Panel

Influence our SQL roadmap and ensure it meets your real-life needs

https://aka.ms/JoinSQLUserPanel

How was the session?

Complete Session Surveys in Whova for your chance to WIN PRIZES!



Get Two Fabric Certifications for FREE

Attendees of FABCON can take the Fabric Analytics Engineer or Fabric Data Engineer exam for free. Be part of the 2 fastest growing role-based certifications in Microsoft history.

Request your voucher by March 23, 2026.

https://aka.ms/fabcon/cert100