**A black text on a white background

Description automatically generated**   
   
  
   
  
 **Masterclass**

This documentation provides a complete walkthrough of concepts, configurations, and implementations covered in the Databricks Masterclass. It includes guidance on working with DBFS, Azure integrations, Delta Lake, streaming with Auto Loader, and workflow orchestration.

**Key Takeaways ✅**

**📁 Databricks File System (DBFS)**

* DBFS is a distributed file system within Databricks.
* It provides an abstraction layer so files can be accessed using path semantics (e.g., **/FileStore/raw.file.csv**).
* DBFS connects to Delta Lake using secure URLs.

**🔐 Accessing Azure Data Lake via Service Principal**

To access Azure Data Lake:

* Register an application in Azure (App Registration in Entra ID).
* Assign the "**Storage Blob Data Contributor**" role to the application.
* Collect these details:
  + Application\_ID
  + Tenant\_ID
  + Secret Value

**🛠️ Databricks Utilities (dbutils)**

File system utilities:

dbutils.fs.ls("abfss://source@storageaccount.dfs.core.windows.net/")

Widgets for parameterization:

dbutils.widgets.text("p\_name", "girald")

dbutils.widgets.get("p\_name")

**🔑 Secure Access with Azure Key Vault**

**Steps to Integrate Key Vault**

1. **Create a Key Vault** in Azure and configure access policies.
2. Assign yourself the **Key Vault Administrator role**.
3. Create a secret (e.g., **app-secret**).
4. In Databricks, go to **#secrets/createScope/** to create a new scope.
5. Link to Azure Key Vault using:
   * **DNS Name**: Found under Key Vault → Properties
   * **Resource ID**: Also found in Properties

Access secrets in Databricks:

dbutils.secret.list(scope="giraldscope")

dbutils.secret.get(scope="giraldscope", key="app-secret")

**📥 Reading Data with PySpark**

df\_sales = spark.read.format("csv") \

.option("header", True) \

.option("inferSchema", True) \

.load("abfss://source@storagename.dfs.core.windows.net/")

**🔄 Transformations with PySpark**

Import necessary libraries:

from pyspark.sql.functions import \*

from pyspark.sql.types import \*

Split column:

df\_sales.withColumn("testdata", split(col("testcolumn"), " ")).display()

Add column/constant:

df\_sales.withColumn("testdata", lit("constant\_value"))

Cast column type:

df\_sales.withColumn("testdata", col("testdata").cast(StringType())).display()

**💾 Delta Lake Essentials**

* Delta Lake is a transactional storage format.
* Stores logs in .json and .crc files.

Run another notebook:

%run "/folder/other\_notebook"

Write data in Delta format:

df\_sales.write.format("delta") \

.mode("append") \

.option("path", "abfss://folder@storage.dfs.core.windows.net/") \

.save()

**🔁 Delta Lake Advanced Features**

View history:

DESCRIBE HISTORY table\_name;

Restore to previous version: TIME TRAVEL

RESTORE TABLE table\_name TO VERSION AS OF 1;

Remove old files default 7days : VACCUM

VACUUM table\_name;

Optimize and Z-Order:

OPTIMIZE table\_name;

OPTIMIZE table\_name ZORDER BY (id);

**🌊 Streaming with Auto Loader**

Define a streaming DataFrame:

df = spark.readStream.format("cloudFiles") \

.option("cloudFiles.format", "parquet") \

.option("cloudFiles.schemaLocation", "abfss://aldestination@lake.dfs.core.windows.net/checkpoint") \

.load("abfss://alsource@lake.dfs.core.windows.net")

Write streaming data to Delta:

df.writeStream.format("delta") \

.option("checkpointLocation", "abfss://aldestination@lake.dfs.core.windows.net/checkpoint") \

.option("mergeSchema", "true") \

.trigger(processingTime="5 seconds") \

.start("abfss://aldestination@lake.dfs.core.windows.net/data")

**🧩 Workflows in Databricks**

Databricks workflows allow you to:

* Automate tasks and pipelines
* Chain notebooks and define dependencies
* Schedule jobs and pass parameters between tasks

📘 **End of Documentation** – This concludes the Databricks Masterclass.

**❤️ I love data fam!**

Forever grateful for this growing community and always excited to keep learning and building!