🔥 Azure Data Engineer Interview Questions

Complete Guide with Key Answers & Code Examples

Core Azure Data Factory & Pipeline Concepts

1️⃣

Parameters and Variables in ADF

- **Parameters:** Used to pass values at runtime to pipelines. They are defined at the pipeline level and cannot be changed during execution.
- **Variables:** Used to store values within the pipeline and can change during execution. Variables are updated using Set Variable or Append Variable activities.

2️⃣

Time Travel in Your Project

- Time travel is a Delta Lake feature that allows querying historical data (snapshots).
- **Example:**

```
SELECT * FROM table_name VERSION AS OF 5
SELECT * FROM table_name TIMESTAMP AS OF '2023-01-15'
```

- **Use Case:** Debugging, auditing, or recreating datasets for ML models.

3️⃣

Resume Pipeline from Failed Activity

- Enable checkpointing or activity retry in ADF. Use a failure path with logic to resume execution by using the Get Metadata activity to evaluate where the pipeline failed.

4️⃣

Pipelines You've Worked With

- **Example:** ETL pipelines to ingest and transform raw data from Azure Data Lake using Data Flows and Spark jobs.
- Mention specifics like copy data activities, data validation, and orchestration of transformations.

5️⃣

Partition vs. Bucketing

- **Partitioning:** Divides the data into directories based on keys (e.g., year, month).
- **Bucketing:** Hashes data into fixed-sized buckets, optimizing joins and aggregations.

Data Architecture & Security

6️⃣

Medallion Architecture

- A data architecture that separates data into three layers:

  Bronze: Raw ingested data

Silver: Cleaned and transformed data

Gold: Business-level aggregates and insights

7️⃣

Azure Key Vault

- Securely stores secrets, keys, and certificates.
- Use Managed Identity in ADF to access Key Vault without hardcoding credentials.

8️⃣

Unity Catalog vs. Hive Metastore

- **Unity Catalog:** Centralized data governance and access control for all your Databricks workspaces.
- **Hive Metastore:** Manages metadata for Hive and Spark tables, but lacks robust access control.

PySpark & Data Processing

9️⃣

Joins in PySpark

- **Inner Join:** Matches rows from both datasets based on a condition.
- **Left/Right Join:** Keeps all rows from the left/right and matches with the right/left dataset.
- **Full Outer Join:** Includes all rows from both datasets.
- **Cross Join:** Cartesian product of both datasets.

🔟

How to Implement Parallel Processing in ADF?

- Use For Each Activity with the Batch Count property set for parallelism.
- Enable concurrent execution in pipeline settings.
- Use partitioned datasets for parallel reads/writes to optimize execution.

Advanced Spark Concepts

1️⃣1️⃣

Difference Between Narrow and Wide Transformations

- **Narrow:** Data is processed within the same partition (e.g., map, filter). Minimal shuffling.
- **Wide:** Data is shuffled across partitions (e.g., groupBy, join). Higher computational cost.

1️⃣2️⃣

What is SCD? Explain SCD1, SCD2, SCD3

- SCD (Slowly Changing Dimensions) handles historical changes in dimension data.
- **SCD1:** Overwrites old data with new data.
- **SCD2:** Maintains history by adding new rows for changes (e.g., adding an Effective_Date).
- **SCD3:** Adds new columns to store historical data for specific attributes.

1️⃣3️⃣

Cluster Options in Databricks

- **Standard Cluster:** For general-purpose workloads.
- **High-Concurrency Cluster:** Optimized for multiple concurrent users.
- **Single Node Cluster:** For lightweight testing and debugging.
- **Jobs Cluster:** Automatically created for specific jobs and deleted afterward.

1️⃣4️⃣

Difference Between Managed and External Tables

- **Managed Tables:** Databricks manages the data and metadata (stored in default storage).
- **External Tables:** Data is stored outside Databricks, and only metadata is managed in the metastore.

1️⃣5️⃣

What is a Surrogate Key?

- A unique identifier for a record, not derived from application data.
- **Example:** Auto-increment ID in databases.

Optimization & Performance

1️⃣6️⃣

Spark Optimization Techniques

- Cache/persist frequently used data.
- Use broadcast joins for smaller datasets.
- Partition data effectively.
- Enable predicate pushdown for filters.
- Avoid wide transformations where possible.

1️⃣7️⃣

Why is Databricks Better Than Dataflow?

- **Flexibility:** Databricks supports more complex workloads (e.g., ML, streaming).
- **Notebook Interface:** Collaborative development environment.
- **Performance:** Databricks uses Apache Spark with optimizations like Delta Lake.
- Dataflow is simpler for straightforward ETL use cases.

1️⃣8️⃣

Difference Between Data Lake and Delta Lake

- **Data Lake:** Stores raw, unstructured data. No ACID compliance.
- **Delta Lake:** Built on top of a data lake with ACID transactions, time travel, and schema enforcement.

1️⃣9️⃣

Explain Spark Architecture

- **Driver:** Coordinates execution, maintains DAG, and schedules tasks.
- **Executors:** Run tasks assigned by the driver. Each executor has its memory and cache.
- **Cluster Manager:** (e.g., YARN, Kubernetes) Allocates resources to the driver and executors.

Advanced Data Processing

2️⃣0️⃣

Difference Between groupByKey and reduceByKey

- **groupByKey:** Groups all key-value pairs by key and shuffles all data. More memory-intensive.

- **reduceByKey:** Combines values at the mapper side before shuffling, reducing network traffic. Preferred for better performance.

2️⃣1️⃣

Why is MapReduce Not Widely Used Now? Similarities Between Spark and MapReduce?

- **Why not MapReduce:**

  High latency due to disk I/O for intermediate results

  Complex to code compared to Spark
- **Similarities:**

  Both process large-scale data using distributed computing

  Use key-value pairs for transformations
- **Spark Advantages:**

  In-memory computation, faster execution, rich APIs (Python, Scala)

2️⃣2️⃣

What is Delta Lake? Key Features and Creating Delta Tables

- Delta Lake: A storage layer on top of Data Lake offering ACID compliance and reliability.
- **Key Features:**

  ACID transactions

  Schema enforcement and evolution

  Time travel and versioning
- **Creating Delta Tables:**

```
CREATE TABLE delta_table USING DELTA LOCATION 'path_to_delta';
```

Azure Synapse & Data Migration

2️⃣3️⃣

Difference Between Serverless Pool and Dedicated SQL Pool

- **Serverless Pool:**

  Pay-per-query model

  Used for ad-hoc queries on data lakes
- **Dedicated SQL Pool:**

  Pre-provisioned resources with fixed cost

  Designed for high-performance data warehousing

2️⃣4️⃣

Prerequisites Before Migration

- Assess source and target environments.
- Ensure schema compatibility.
- Perform data profiling and cleansing.
- Set up network, storage, and permissions.
- Validate data transformation logic.

2️⃣5️⃣

What is a Mount Point in Databricks?

- A mount point is a shortcut to a storage account, enabling easier access.
- **Example:** Mounting an Azure Data Lake Gen2 folder using a dbutils.fs.mount command.

2️⃣6️⃣

How to Optimize Databricks Performance

- Enable Delta Lake optimizations like Z-ordering and OPTIMIZE.
- Use Auto-scaling for clusters.
- Use broadcast joins for smaller datasets.
- Optimize shuffling with correct partitioning.
- Persist reusable datasets in memory with cache().

PySpark Functions & Data Handling

2️⃣7️⃣

Difference Between map and flatMap

- **map:** Transforms each element into another element, 1-to-1 mapping.
- **flatMap:** Can produce 0 or more elements per input, 1-to-n mapping.

2️⃣8️⃣

How to Fetch Details from Key Vault

- Use Azure Key Vault Linked Service in ADF or Databricks.
- **In Databricks:**

```
secret_value = dbutils.secrets.get(scope="key_vault_scope",
key="secret_name")
```

2️⃣9️⃣

Applying Indexing on a Databricks Table

- Use Delta Lake Z-order indexing:

```
OPTIMIZE delta_table_name ZORDER BY (column_name);
```

- Helps improve query performance for large datasets.

3️⃣0️⃣

Transferring Data to Azure Synapse

- Use Azure Data Factory for ETL pipelines.
- COPY INTO command in Synapse for fast ingestion from Data Lake.
- Databricks-to-Synapse via JDBC connector or PolyBase.

Advanced Data Engineering Concepts

3️⃣1️⃣

What is Incremental Loading? How to Implement It?

- **Definition:** Loading only new or updated data to a target without reloading the entire dataset.
- **Implementation:**

  Watermarking: Use timestamps or surrogate keys to identify changes

  ADF: Use Lookup + Filter activities

  Delta Lake: Merge using UPSERT logic:

```
MERGE INTO target_table AS target
USING source_table AS source
ON target.id = source.id
WHEN MATCHED THEN UPDATE SET target.col = source.col
WHEN NOT MATCHED THEN INSERT (columns) VALUES (values);
```

3️⃣2️⃣

How Does Z-Ordering Work?

- Z-Ordering: A data layout optimization in Delta Lake that reduces I/O by co-locating similar data on disk.
- **How:**

  Applies a multi-dimensional sort algorithm

  Improves query performance on frequently filtered columns

```
OPTIMIZE table_name ZORDER BY (column1, column2);
```

3️⃣3️⃣

What is Dimension Modeling? Dimension and Fact Tables?

- Dimension Modeling: A design technique for data warehouses to optimize query performance using star or snowflake schemas.
- **Fact Tables:** Store numeric measures (e.g., sales amount).
- **Dimension Tables:** Describe the context of facts (e.g., customer, product).

3️⃣4️⃣

Difference Between a Data Lake and a Data Warehouse

- **Data Lake:**

  Stores raw, unstructured data

  Scalable, cost-effective

  Example: Azure Data Lake
- **Data Warehouse:**

  Stores structured, processed data for analytics

  Schema-on-write

  Example: Azure Synapse

Data Processing & SQL Queries

4️⃣4️⃣

Query to Find the 4th Highest Salary of an Employee

```
SELECT DISTINCT salary
FROM employee
ORDER BY salary DESC
LIMIT 4 OFFSET 3;
```

- **Alternatively, using ROW_NUMBER:**

```
SELECT salary
FROM (
  SELECT salary, ROW_NUMBER() OVER (ORDER BY salary DESC) AS
rank
  FROM employee
) ranked
WHERE rank = 4;
```

4️⃣5️⃣

PySpark Command to Read Data from a File into a DataFrame

```
df = spark.read.csv("path/to/file.csv", header=True,
inferSchema=True)
```

- **Other Formats:**

  JSON: spark.read.json("path")

  Parquet: spark.read.parquet("path")

4️⃣6️⃣

Handling Nulls and Duplicates in PySpark

- **Drop Nulls:**

```
df = df.dropna()
```

- **Fill Nulls:**

```
df = df.fillna({'col1': 'default_value', 'col2': 0})
```

- **Remove Duplicates:**

```
df = df.dropDuplicates(['col1', 'col2'])
```

4️⃣7️⃣

Changing the Date Format for a Date Column

```
from pyspark.sql.functions import date_format
df = df.withColumn("new_date", date_format("date_column",
"yyyy-MM-dd"))
```

4️⃣8️⃣

What is the Explode Function in PySpark?

- Explode: Converts an array or map into multiple rows.
- **Example:**

```
from pyspark.sql.functions import explode
df = df.withColumn("exploded_col", explode("array_col"))
```

RDD & DataFrame Operations

5️⃣1️⃣

Different Approaches to Creating RDD in PySpark

- **From a Collection:**

```
rdd = spark.sparkContext.parallelize([1, 2, 3, 4])
```

- **From a File:**