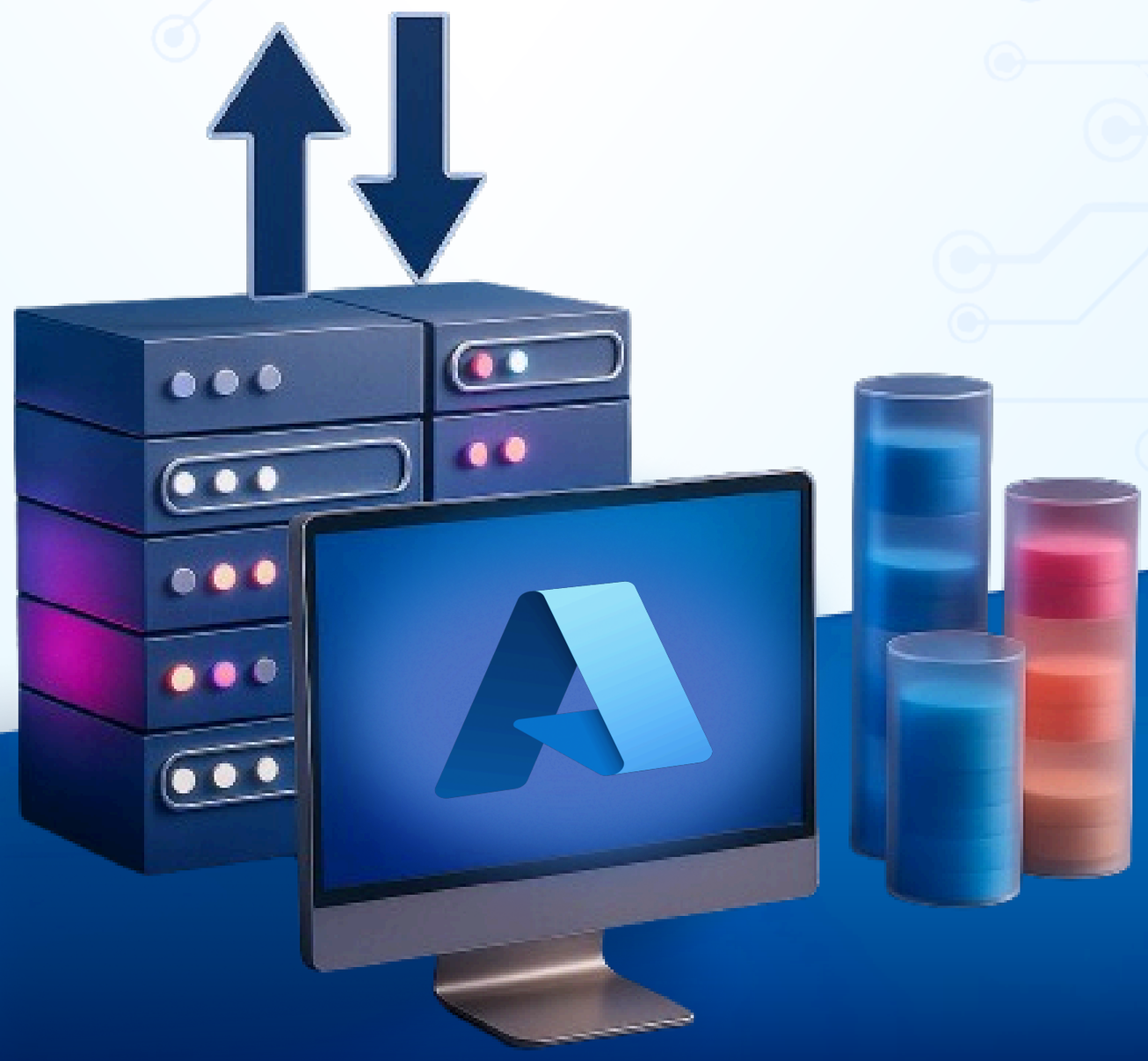
BOSSCODER ACADEMY

TOP 52

AZURE DATA ENGINEER

Interview Questions



For Data Engineer

Disclaimer

System Design is the most asked topic in tech interviews.

So, make sure you prepare it thoroughly.

Take the help of this doc and ace your System Design Interviews.

Key Answers:

QUESTION-1

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.

QUESTION-2

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 or TIMESTAMP AS OF '2023-01-15' .

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

QUESTION-3

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.

QUESTION-4

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.**

QUESTION-5

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.

QUESTION-6

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.

QUESTION-7

Azure Key Vault:

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

QUESTION-8

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.

QUESTION-9

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.

Want further explanations, detailed examples, or help structuring your answers? 😊

💡 These are excellent questions to enhance your preparation for Azure Data Engineer interviews. Here's a breakdown of concise, yet impactful answers:

QUESTION- 10

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.

QUESTION- 11

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.

QUESTION-12

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.

QUESTION-13

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.

QUESTION-14

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.

QUESTION-15

What is a Surrogate Key?

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

QUESTION-16

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.

QUESTION-17

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.

QUESTION-18

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.

QUESTION-19

Spark Optimization Techniques:

- **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.

Need examples for any of these? Or a deeper dive into any topic? 🚀

Here’s a solid overview of answers to these questions, tailored to help you shine in interviews! 🌟

QUESTION- 20

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.

QUESTION- 21

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).

QUESTION- 22

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';
```

QUESTION-23

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.

QUESTION-24

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.

QUESTION-25

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.

QUESTION-26

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() .

QUESTION-27

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.

QUESTION-28

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")
```

QUESTION-29

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.

QUESTION-30

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.

QUESTION- 31

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);

Need any of these elaborated further or some live coding examples? 🚀

Here's a breakdown of these advanced Azure Data Engineering topics to keep your prep on point! 🚀

QUESTION-32

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);

QUESTION-33

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).

QUESTION-34

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.

QUESTION-35

Using Logic Apps in Your Project:

- Automates workflows between services like ADF, Synapse, and notifications.
- Example Use Case:
  - Trigger data pipelines based on events (e.g., file upload).
  - Send failure alerts via email or Teams.

QUESTION-36

What is Data Skewness?

- **Definition:** Uneven distribution of data across partitions, leading to performance bottlenecks.
- **Mitigation:**
  - Use **salting** techniques (adding random keys).
  - Optimize partitioning with balanced keys.

QUESTION-37

What is Fault Tolerance and Its Use in Real-Time Applications?

- **Definition:** The ability of a system to recover from failures.
- **Real-Time Use:**
  - Spark achieves fault tolerance by storing lineage and recomputing lost partitions.
  - In ADF, retry policies handle transient failures.

QUESTION-38

Converting RDD to DataFrame & Vice Versa:

- **RDD to DataFrame:**

- from pyspark.sql import SparkSession

df = rdd.toDF(schema=["col1", "col2"])

- **DataFrame to RDD:**

rdd = df.rdd

QUESTION-39

Encryption Techniques:

- **At Rest:** Encrypt data in storage using Azure Storage Service

Encryption (SSE).

- **In Transit:** Use TLS/SSL for secure data transfer.

- **Column-Level Encryption:** Secure sensitive data fields (e.g., PII).

QUESTION-40

How Does Auto Loader Work?

- A feature in Databricks for **incremental file processing** from cloud storage.
- **Working:**
  - Tracks metadata using checkpointing.
  - Processes new files automatically.
- Example:
- df = spark.readStream.format("cloudFiles") \
- .option("cloudFiles.format", "json") \ .load("path")

QUESTION-41

Explain Lazy Evaluation in PySpark:

Definition: Transformations are not executed immediately but only when an action (e.g., count , collect) is triggered.

- **Benefits:**
  - Optimizes execution by combining transformations into a single stage.
  - Reduces unnecessary computations.

QUESTION- 42

What is DAG in Spark?

- **DAG (Directed Acyclic Graph):**
  - A sequence of computations where each node represents a transformation and edges represent dependencies.
  - Spark breaks the execution into stages using DAG, ensuring fault tolerance and optimized execution.
- **Significance:**
  - Tracks lineage for fault recovery.
  - Optimizes execution by combining transformations.

Want any topic expanded with examples or real-world scenarios? Let me know! 🚀

Here's a detailed explanation of these additional Spark and PySpark-related questions:

QUESTION-43

Significance of Catalyst Optimizer in PySpark?

- **What It Is:** A query optimization engine in Spark SQL.
- **Functions:**
  - Converts logical plans into optimized physical plans.
  - Pushes predicates (filter operations) early to minimize I/O.
- **Benefits:** Better performance with optimized execution plans.

QUESTION-44

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;
```

QUESTION-45

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")

QUESTION-46

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'])

QUESTION-47

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"))
```

QUESTION-48

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"))
```

QUESTION-49

Code to Read a Parquet File:

```
df = spark.read.parquet("path/to/file.parquet")
```

QUESTION-50

Code to Add a Column to a Parquet File:

```
from pyspark.sql.functions import lit

df = spark.read.parquet("path/to/file.parquet")

df = df.withColumn("new_column", lit("value"))

df.write.parquet("path/to/updated_file.parquet")
```

QUESTION-51

Different Approaches to Creating RDD in PySpark:

- **From a Collection:**

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

- **From a File:**

```
rdd = spark.sparkContext.textFile("path/to/file.txt")
```

QUESTION-49

Code to Read a Parquet File:

```
df = spark.read.parquet("path/to/file.parquet")
```

QUESTION-52

Different Approaches to Creating DataFrame in PySpark:

- **From RDD:**

- from pyspark.sql import Row

- rdd = spark.sparkContext.parallelize([Row(name="Alice", age=25), Row(name="Bob", age=30)])

df = rdd.toDF()

- **From a File:**

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

- **From a List/Dictionary:**

- data = [("Alice", 25), ("Bob", 30)]

df = spark.createDataFrame(data, schema=["name", "age"])

BOSSCODER ACADEMY

WHY BOSSCODER?

2200+ Alumni placed at Top Product-based companies.

More than **136% hike** for every 2 out of 3 Working Professional.

Average Package of **24LPA**.

The syllabus is most **up-to-date** and the list of problems provided covers **all important topics.**

Lavanya
Meta

Course is very well **structured** and **streamlined** to crack any **MAANG** company .

Rahul
Google

EXPLORE MORE