HCL Data Engineer Interview Guide – Experienced 3+

Round 1: Technical

1. Introduction (Example Answer)

Thank you for the opportunity. My name is X and I currently work as a Data Engineer at ZS Associates. I have three years of experience, with 2.5 years at ZS. My primary expertise lies in designing and implementing scalable data pipelines using technologies like Spark, Hive, Hadoop, and Python. I also specialize in optimizing data processing workflows and ensuring robust CI/CD practices with Azure DevOps. I hold a B.Tech degree from COEP Pune, class of 2020.

2. Explain Your Project Architecture

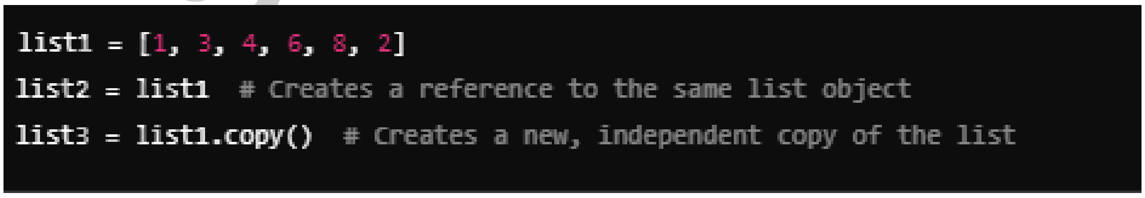
Overview:
I am working on a data analytics platform for a healthcare domain client. The project's primary objective is to track doctor prescribing patterns and optimize the marketing strategy for pharmaceutical products. The goal is to process large datasets, perform complex transformations, and generate insights for the client's sales and marketing teams.

Tech Stack:
"We use data ingestion pipelines built with Spark on AWS S3 for storage and Python for data transformation. Hive serves as our data warehouse, with Airflow orchestrating and automating our data workflows. CI/CD is managed using Azure DevOps."

Data Flow:

- Data is received from vendors in raw format and stored in S3 buckets.
- Spark jobs process and clean the data, applying transformation rules and business logic.
- The output is loaded into Hive tables for further querying and analytics.
- Airflow schedules the Spark jobs to ensure efficient automation and error handling.

3. Difference Between list1 = list2 and list1.copy()



```
list1 = [1, 3, 4, 6, 8, 2]
list2 = list1  # Creates a reference to the same list object
list3 = list1.copy()  # Creates a new, independent copy of the list
```

Explanation:

- list2 = list1: Both variables point to the same object, so changes in list2 will affect list1.
- list3 = list1.copy(): Creates a shallow copy, meaning changes in list3 do not affect list1.

4. Dictionary Sorting Based on Values

```python
data = {'apple': 10, 'banana': 5, 'cherry': 7}
sorted_dict = dict(sorted(data.items(), key=lambda item: item[1]))
print(sorted_dict)
# Output: {'banana': 5, 'cherry': 7, 'apple': 10}
```

5. PySpark Code to Extract Data from a CSV and Create a Table

```python
from pyspark.sql import SparkSession

spark = SparkSession.builder.appName("CSV to Table").getOrCreate()

df = spark.read.csv("path_to_file.csv", header=True, inferSchema=True)
df.createOrReplaceTempView("temp_table")
spark.sql("CREATE TABLE my_table AS SELECT * FROM temp_table")
```

6. How to Monitor Spark Jobs

- Use Spark UI for detailed stage-level insights into task execution times, shuffle operations, and memory usage.
- Ganglia and Grafana can be used for cluster-level monitoring.
- Log aggregation services like Splunk or ELK can help analyze Spark logs for job errors or performance bottlenecks.

Round 2: Managerial

1. Explain Your Project Architecture

(Refer to the detailed architecture explanation from Round 1.)

2. How to Handle Data Using AWS S3

- **Read**: spark.read.csv("s3a://bucket_name/file.csv")
- **Write**: df.write.csv("s3a://bucket_name/output/")
- **Delete**: aws s3 rm s3://bucket_name/file --recursive
- **Rename**: Requires a **copy and delete** operation, as S3 does not support renaming directly.

3. Data Volume

In my current role, I handle data pipelines that process 10 TB of data daily, performing ETL on streaming and batch data sources. We utilize Spark for distributed computation to ensure scalability.

4. Cluster Configuration

Our cluster comprises 10 m5.4xlarge instances with 128 GB of memory and 32 cores per node. The master node manages scheduling and task distribution.

5. Fact Table and Star Schema

- **Fact Table**: Stores quantitative metrics like sales or transactions.
- **Star Schema**: A central fact table is connected to multiple dimension tables, enabling easy navigation of relationships.

6. SQL Query to Find House with Avg(score) > 70

SELECT house FROM student GROUP BY house HAVING AVG(score) > 70;

7. Difference Between List and Tuple

- **List**: Mutable, dynamic in size.
- **Tuple**: Immutable, faster access due to fixed structure.

8. MapReduce Architecture

MapReduce consists of a Map phase, which processes input data, and a Reduce phase, which aggregates the output from Map tasks.

9. Handling Production Deployment

We use Azure DevOps for CI/CD pipelines, automating Spark job deployments. Jobs are tested in staging before being promoted to production.

10. Spark Submit Properties

Key properties:

- --num-executors: Number of executors.
- --executor-memory: Memory per executor.
- --conf spark.sql.shuffle.partitions=200: Controls shuffle partitions.

11. Partitioning vs. Bucketing

- **Partitioning** divides data into folders based on column values (e.g., year).
- **Bucketing** creates fixed-size data buckets, improving shuffle performance.

12. Spark SQL vs. Hive Performance

- Spark SQL leverages in-memory computation for faster processing.
- Hive queries rely on MapReduce, making Spark SQL significantly faster.

Glassdoor HCL Review –

https://www.glassdoor.co.in/Reviews/HCLTech-Reviews-E553909.htm

HCL Careers –

https://www.hcltech.com/careers

Subscribe to my YouTube Channel for Free Data Engineering Content –

https://www.youtube.com/@shubhamwadekar27

Connect with me here –

https://bento.me/shubhamwadekar

Checkout more Interview Preparation Material on –

https://topmate.io/shubham_wadekar