**Chryselys Data Engineer Interview Guide – Experienced 3+**

**Interview Process Breakdown**

**Round 1: Fundamental and Practical Knowledge**

 **Focus Areas**: SQL, Python, Data Warehousing, and basic problem-solving.

 **Objective**: Evaluate the candidate’s ability to work with foundational data engineering concepts and their problem-solving approach.

**Round 2: Advanced Tools and Scenarios**

 **Focus Areas**: Big Data tools (Hive, Sqoop, Spark), cloud services (AWS, Delta

Lake), and programming concepts (Scala).

 **Objective**: Assess the candidate's expertise in using modern data engineering tools, understanding advanced concepts, and their ability to solve real-world scenarios.

**Detailed Insights on Each Round**

**Round 1: Fundamental and Practical Knowledge**

1. **Introduce Yourself**

 The interviewer started with the classic “Tell me about yourself” question, aiming to gauge the candidate’s communication skills and professional background.

 **Tip**: Use this opportunity to highlight your relevant experience, recent projects, and technical expertise.

2. **Recent Projects and Challenges**

**Question**: “Explain the recent projects you have worked on.”

**Follow-ups**:

What challenges did you face during these projects?

What strategies did you use to monitor and troubleshoot failed pipelines?

**Insight**: The interviewer was looking for practical experience and a systematic approach to handling issues.

3. **Data Warehousing Concepts**

**Question**: “What is a Data Warehouse, and can you explain its Tier-1 and

Tier-2 architecture?”

**Tip**: Emphasize the structured nature of data warehouses and explain tiered architectures in simple terms.

4. **OLTP vs OLAP**

**Question**: “What is the difference between OLTP and OLAP?”

**Example Answer**: OLTP systems handle transactional data with frequent, small operations, while OLAP systems focus on analytical queries over large datasets.

5. **Join Operations**

**Scenario**: Analyze the output of various joins (LEFT, RIGHT, INNER, CROSS, FULL OUTER) on the following tables:

**Table 1**:

**Col**

a a

a

**Table 2**:

**Col**

a a a a a

**Tip**: Understand the nuances of join operations and focus on edge cases like duplicates.

6. **SQL Query**

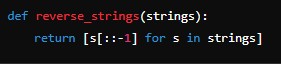
**Question**: “Write a query to get the names of all employees who are managers with five or more direct reports.”

**Insight**: Use GROUP BY and HAVING to handle such queries efficiently.

7. **Python Problem**

**Question**: Write a Python function to reverse all strings in a list.

**Example**:

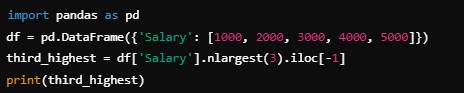


8. **Pandas Problem**

**Question**: Write code to find the third-highest salary in a dataset using

Pandas.

**Solution**:



**Round 2: Advanced Tools and Scenarios**

1. **Sqoop Command**

**Question**: Write a Sqoop command to import all relational tables from a

MySQL database into HDFS.

**Solution**:

sqoop import-all-tables --connect jdbc:mysql://<host>:<port>/<database> -- username <user> --password <password> --target-dir /hdfs/target/path

2. **Scheduling Spark Jobs in Databricks**

**Question**: How would you schedule Spark jobs using Databricks?

**Insight**: Explain using Databricks’ job scheduling interface, specifying cluster settings and cron expressions.

3. **Hive Basics**

**Question**: Explain Hive, its purpose, and its default metadata storage.

**Follow-up**: Why does Hive use Derby by default, and what alternatives are used in production?

**Tip**: Highlight the scalability of production databases like MySQL or

PostgreSQL for metadata storage.

4. **Data Lake vs Data Warehouse**

**Question**: Explain the differences between a Data Lake and a Data

Warehouse.

**Focus**: Talk about schema-on-read vs schema-on-write and use cases for both.

5. **AWS Concepts**

**Question**: Describe an AWS EC2 instance and how IAM roles/policies enhance security.

**Follow-up**: Discuss S3’s advantages, including scalability and durability.

6. **Delta Lake**

**Question**: What file format does Delta Lake use, and why is it beneficial?

**Insight**: Delta Lake uses Parquet format, offering ACID transactions and scalability.

7. **Scala Currying**

**Question**: What is currying in Scala?

**Example**: Currying transforms a function with multiple parameters into a series of functions, each taking one parameter.

8. **Higher-Order Functions in Scala**

**Question**: Write a higher-order function to filter values greater than a threshold in a list.

**Solution**:

def filterThreshold(threshold: Int, values: List[Int]): List[Int] = {

values.filter(\_ > threshold)

}