Gartner Data Engineer Interview Guide – Experienced 3+

Gartner Round 1: Technical

1. **Databases drivers/connectors**

   This question focuses on understanding different types of database connectors (JDBC, ODBC, Python libraries like psycopg2 for PostgreSQL or pyodbc for SQL Server) and their use cases in connecting and querying databases. A deep understanding of how these drivers work in data pipelines, integrations, and tools (like Apache Kafka or Spark) is essential.

2. **Lambda functions in Python**

   Lambda functions in Python are anonymous functions defined using the lambda keyword. They're used for short, throwaway functions, particularly in cases like map-reduce operations or when functions are passed as arguments.

3. **CTE in SQL**

   CTEs (Common Table Expressions) in SQL are temporary result sets that can be referenced within a SELECT, INSERT, UPDATE, or DELETE statement. CTEs improve readability and organization in complex queries. Example:

   ```
   WITH cte AS (SELECT column1 FROM table WHERE condition)
   SELECT * FROM cte;
   ```

4. **Union vs Union All in SQL**

   UNION combines the results of two queries and removes duplicates, while UNION ALL combines the results of two queries, keeping all records, including duplicates.

   - UNION: Slower, because it needs to eliminate duplicates.
   - UNION ALL: Faster, as no de-duplication occurs.

5. **Number of rows using joins (left, right, full outer, inner)**

   - For each type of join:
     - **Inner Join**: Returns only matching records from both tables.
     - **Left Join**: Returns all records from the left table, and matching records from the right table.
     - **Right Join**: Returns all records from the right table, and matching records from the left table.
     - **Full Outer Join**: Returns all records when there is a match in either table. Non-matching records will have NULLs in the columns of the other table.
     - **Example SQL**:

   ```
   SELECT * FROM table_a
   INNER JOIN table_b ON table_a.column_a = table_b.column_b;
   ```

6. **PostgreSQL vs Snowflake, Duplicate Record Error**

   PostgreSQL and Snowflake have different architectures and performance optimizations. Snowflake is a cloud-native data warehouse, offering scalable storage and compute resources, while PostgreSQL is an open-source RDBMS that may require more configuration for performance optimization.

   For duplicate record error handling, Snowflake's error message might indicate a unique constraint violation, whereas PostgreSQL might raise a duplicate key value violates unique constraint error.

7. **Python Dictionary from List**

   Task: Create a dictionary with list elements as keys and their occurrences as values.

```
nums = [1, 2, 3, 4, 2, 3, 4, 4]
result = {x: nums.count(x) for x in set(nums)}
print(result)  # Output: {1: 1, 2: 2, 3: 2, 4: 3}
```

8. **Latest Rule ID and Status Query**

   Solution uses ROW_NUMBER() to get the latest record per rule:

   SELECT rule_id, rule_status

   FROM rule

   QUALIFY ROW_NUMBER() OVER (PARTITION BY rule_id ORDER BY to_timestamp(time_created DESC)) = 1;

9. **Average Salary by Department, Max Average Salary, and List of Employees for Max Salary Department**

   DataFrame approach:

```
data = {
    "employee_id": [1, 2, 3, 4, 5, 6, 7],
    "name": ['Alice', 'Bob', 'Carol', 'David', 'Eve', 'Frank', 'Grace'],
    "department": ['HR', 'IT', 'IT', 'HR', 'Marketing', 'Marketing', 'IT'],
    "salary": [60000, 80000, 85000, 65000, 70000, 72000, 95000]
}
df = pd.DataFrame(data)
avg_salary = df.groupby('department')['salary'].mean()
max_avg_dept = avg_salary.idxmax()
employees_max_avg_salary = df[df['department'] == max_avg_dept]
```

Gartner Round 2: Technical

1. **Tell Me About Yourself Apart from CV**

   Discuss additional skills, projects, or hobbies relevant to the role that might not be on your CV. Example: open-source contributions, personal projects, or leadership roles in your field.

2. **How to Optimize Data Ingestion**

   Considerations for optimizing data ingestion include:

   - Using batch processing vs real-time processing depending on the use case.
   - Choosing the right data format (Parquet, Avro, or ORC) for storage efficiency.
   - Using partitioning and indexing to improve query performance.
   - Parallel data loading and optimized connectors.

3. **Making Data Ingestion Fast with 5 Tables of 50k Records**

   Data ingestion can be made faster by:

   - Using distributed systems like Spark or Kafka for parallel processing.
   - Utilizing data compression to reduce the transfer time.
   - Staging data in batches to avoid memory overload.
   - Implementing efficient ETL pipelines with minimal transformations during ingestion.

4. **Which Library Would You Use and Why?**

   For optimized data ingestion in Python, you might use:

   - pandas for smaller datasets.
   - PySpark for larger datasets to leverage distributed computing.
   - dask for out-of-core computations on large datasets.

5. **How Can Spark Help in Optimizing Ingestion?**

   Spark enables distributed data processing and can efficiently handle large-scale data ingestion tasks by parallelizing them across clusters. It supports fault tolerance, which is vital for ingesting large amounts of data.

6. **Exception Handling in Data Ingestion**

   Essential practices:

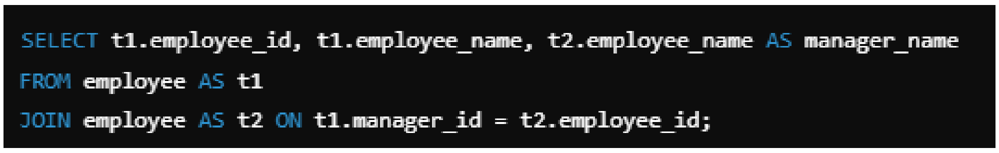
   - Implementing retries in case of failures.
   - Using dead-letter queues for records that cannot be processed.
   - Logging errors for debugging and analysis.

7. **Normalization and Disadvantages**

   Normalization reduces redundancy and improves data integrity, but it can lead to complex queries and performance issues due to the need for multiple joins. Denormalization is often used for performance optimization in OLAP systems.

8. **Self Join Query Example**

   The task involves self-joining the employee table to get the manager's name for each employee:



```sql
SELECT t1.employee_id, t1.employee_name, t2.employee_name AS manager_name
FROM employee AS t1
JOIN employee AS t2 ON t1.manager_id = t2.employee_id;
```

9. **ETL Experience**

   Discuss any ETL pipelines you have designed or worked on, focusing on the tools (e.g., Apache Nifi, Airflow, Spark), architectures, and optimizations.

10. **Native vs Cloud Database Systems**

    Discuss pros and cons of native vs cloud databases, and specifically compare PostgreSQL/MySQL vs Snowflake/Redis, focusing on scalability, performance, and cost-efficiency in cloud environments.

Gartner Round 2: Technical

Same interviewer same position, interviewed again after 3 months

1. **Design an End-to-End ETL Pipeline**

   Walk through the stages of an ETL pipeline, such as:

   - Extracting data from various sources (APIs, databases, flat files).
   - Transforming data (cleaning, normalization, aggregation).
   - Loading data into a data warehouse or data lake.

2. **Dealing with Failed Large File Processing**

   If a file takes hours and fails at the final 10%, you should:

   - Check the logs to identify the failure point.
   - Retry the job from the last successful checkpoint (using tools like Apache Airflow or DBT).
   - Use incremental loading to avoid reprocessing the entire file.

3. **Compression Algorithms: Gzip vs Snappy**

   Discuss when to use each:

   - Gzip for better compression ratios and slower processing.
   - Snappy for faster compression and decompression with real-time systems, though it may result in larger file sizes.

4. **ETL Process Flags and Segregation of Steps**

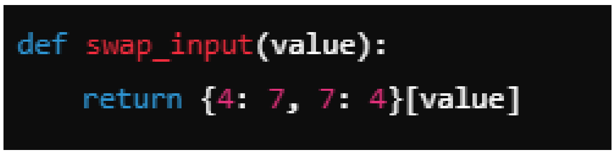
   Flags in ETL can help identify the status of each record (e.g., "Processed", "Failed"). Segregating the steps ensures that failures in one stage do not impact others.

5. **Explain This Code: [f(2) for f in [lambda x: x * i for i in range(5)]]**

   Discuss the behavior of lambda functions and closures. All lambdas in the list will use the last value of i, which will be 4, and thus all the lambda functions will return 2 * 4.

6. **Swap Function Without If-Else**

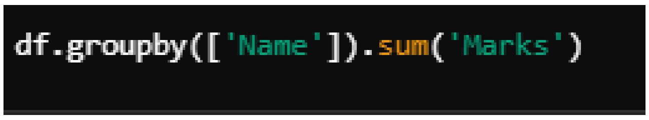
   The function uses a dictionary to map values:



```python
def swap_input(value):
    return {4: 7, 7: 4}[value]
```

7. **SQL Query for Sum of Marks Grouped by Student**

   Grouping data by the Name and summing the marks:



```python
df.groupby(['Name']).sum('Marks')
```

Gartner Round 3: Managerial + Technical (Aug 26, 2024)

1. **High-Level System Design of a Netflix-like App**
   - Focus on components like:
     - Frontend (UI/UX).
     - Backend (Microservices, APIs).
     - Database (NoSQL like Cassandra, PostgreSQL for metadata).
     - Data pipelines for content recommendations.

2. **Handling Fluctuations in Active Users**

   Discuss load balancing, auto-scaling, and caching strategies to manage fluctuations in active users.

3. **Convincing Stakeholders to Move from Google Drive to Amazon S3**

   Highlight the benefits of S3 (scalability, security, cost efficiency) compared to Google Drive, along with S3's integration with AWS services.

4. **Explaining Your Job to a Kid**

   I help build systems that let different parts of an app talk to each other and send information, kind of like making sure everyone has the right information to work together.

5. **Disagreement with a Co-Worker**

   Emphasize collaboration, explaining both approaches, and working together to find a middle ground or presenting the best approach to the manager.

6. **Lack of Communication from Stakeholders**

   Handle by ensuring frequent check-ins, clarifying expectations, and using project management tools to track progress.

7. **Strategy for Documentation**

   Maintain clear, concise, and up-to-date documentation on systems, processes, and code to ensure knowledge sharing and easier maintenance.

8. **Peer Code Review and Team Lead Review**

   Code reviews are essential for catching bugs early, ensuring code quality, and sharing knowledge within the team.

Glassdoor Gartner Review –

https://www.glassdoor.co.in/Reviews/Gartner-Reviews-E2465.htm

Gartner Careers –

https://jobs.gartner.com/jobs/

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