
Joins in SQL

1. **INNER JOIN** – Returns only matching records between tables.
2. **LEFT JOIN (or LEFT OUTER JOIN)** – Returns all records from the left table and matching records from the right table.
3. **RIGHT JOIN (or RIGHT OUTER JOIN)** – Returns all records from the right table and matching records from the left table.
4. **FULL JOIN (or FULL OUTER JOIN)** – Returns all records from both tables, with NULLs where there is no match.
5. **CROSS JOIN** – Returns a Cartesian product of both tables.
6. **SELF JOIN** – A table joins itself.

Difference Between LEFT JOIN and LEFT OUTER JOIN

There is no difference! **LEFT JOIN** and **LEFT OUTER JOIN** are the same. The word **OUTER** is optional.

What is `spark-submit`?

`spark-submit` is a command used to submit Apache Spark applications to a cluster. It allows you to run PySpark, Scala, or Java applications on a local or distributed environment.

Example:

```
spark-submit --master yarn --deploy-mode cluster my_spark_script.py
```

Hive Internal vs. External Tables

Feature	Internal Table	External Table
Storage	Data is managed by Hive	Data remains in external location (HDFS, S3, etc.)
Deletion	`DROP TABLE` deletes both metadata and data	`DROP TABLE` deletes only metadata, data remains
Use Case	When Hive should manage the data	When external tools manage the data

---

What is a Broadcast Join?

A **broadcast join** in Spark optimizes joins by sending a small table to all worker nodes instead of shuffling data across the cluster.

- It is used when one table is significantly smaller than the other.
- Example in PySpark:

```
from pyspark.sql.functions import broadcast

df1 = spark.read.parquet("large_table")
df2 = spark.read.parquet("small_table")

df_joined = df1.join(broadcast(df2), "id")
```

Spark Optimization Techniques

1. **Use Broadcast Joins** – For small tables.
2. **Cache/Persist Data** – To avoid recomputation.
3. **Use Columnar Formats** – Like Parquet instead of CSV.
4. **Optimize File Sizes** – Avoid too many small files in S3/HDFS.
5. **Optimize Partitions** – Use partitioning and bucketing for efficient queries.
6. **Avoid `collect()` and `count()`** – Minimize data movement to the driver.
7. **Use `repartition()` and `coalesce()` wisely** – Balance parallelism vs. shuffle cost.
8. **Optimize Shuffle Operations** – Reduce data movement in joins.

Fixing Your PySpark Code

Your code has issues with `minv, maxv` calculation. Try this:

```
from pyspark.sql.functions import col, min, max

list_values = [(1,), (3,), (5,), (6,), (7,), (9,), (10,)]
df = spark.createDataFrame(list_values, ["nums"])

minv, maxv = df.agg(min(col("nums")), max(col("nums"))).collect()[0]

ndf = spark.range(minv, maxv + 1).toDF("number")

res = ndf.join(df, ndf.number == df.nums, "left_anti")
res.show()
```

Expected Output:

```
+------+
|number|
+------+
|     2|
|     4|
```

```
|     8|
+------+
```

Find Numbers Appearing Consecutively (at least 3 times)

```
SELECT DISTINCT num
FROM (
    SELECT num,
           LAG(num, 1) OVER (ORDER BY id) AS prev_num1,
           LAG(num, 2) OVER (ORDER BY id) AS prev_num2
    FROM emp
) t
WHERE num = prev_num1 AND num = prev_num2;
```

This finds numbers that repeat **at least three times consecutively**.

Interview Questions from Persistent Systems

Here are some L2 interview questions based on your experience:

SQL & Database

1. Difference between `WHERE` and `HAVING` in SQL?
2. How do you optimize SQL queries in Hive?
3. How does partitioning work in Hive?
4. Explain ACID transactions in Hive.
5. Explain bucketing in Hive.

PySpark & Spark Streaming

6. What are transformations and actions in Spark?
7. Difference between `repartition()` and `coalesce()` in Spark?
8. What is the use of `persist()` and `cache()` in Spark?
9. How does a Spark DAG (Directed Acyclic Graph) work?
10. How do you handle skewed data in Spark?

Big Data & Cloud (AWS/Azure)

11. What is the difference between EMR and Glue?
12. How do you optimize S3 storage for big data processing?
13. How does Kafka handle real-time data ingestion?
14. How do you implement checkpointing in Spark Streaming?
15. What is the difference between Snowflake and Redshift?
