Chapter 18: Streams

Explanation on Streams in the SnowFlake SnowPro Core Certification.

Streams in SnowFlake are perhaps one of the most complex topics of the SnowFlake SnowPro Core certification, although once we understand them, all the questions will seem very simple. Let's go for it!

INTRODUCTION TO STREAMS

Streams are SnowFlake objects that record data manipulation language (DML) changes made to tables (they don't work with views), including INSERTS, UPDATES, and DELETES, as well as metadata about each change. Thanks to this, actions can be taken using the changed data.

Streams don't contain table data; they only store offsets. When created, a table stream logically takes an initial snapshot of every row in the source table by initializing a point in time (called an offset) as the current transactional version of the table. The stream then records the DML changes after this snapshot was taken.
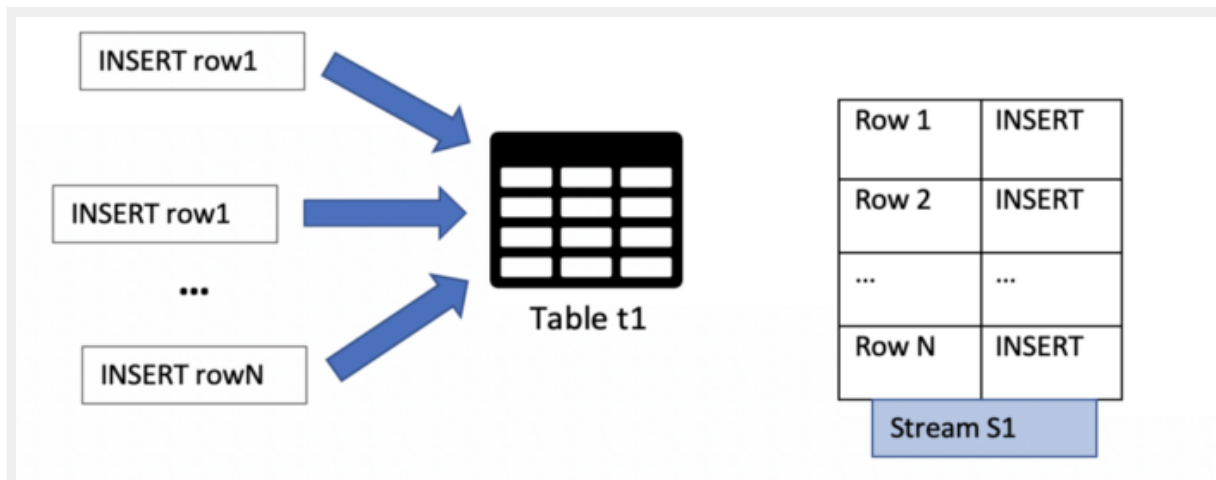
Streams in a SnowFlake table

We have three different types of Streams:

- **Standard** → *Tracks all DML changes to the source table, including inserts, updates, and deletes.*
- **Append Only** → *Tracks row inserts only*
- **Insert Only** → *Tracks row inserts only. The difference with the previous one is that this one is only supported on EXTERNAL TABLES.*
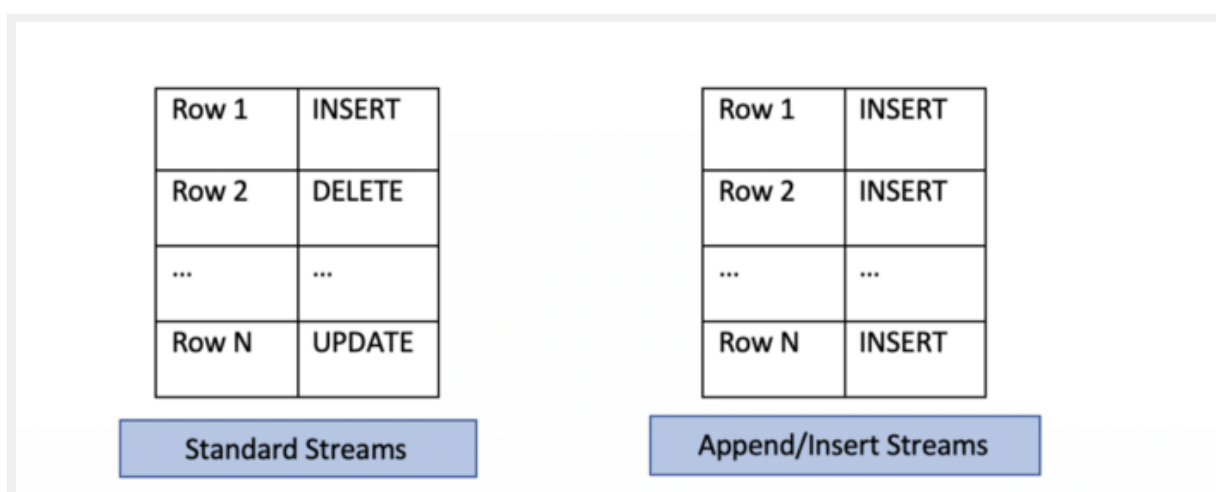
Standard vs. Append only Streams.

Apart from that, each stream contains the following columns:

- **METADATA$ACTION** → Indicates the DML operation (INSERT, DELETE) recorded.
- **METADATA$ISUPDATE** → Indicates whether the operation was part of an UPDATE statement.
- **METADATA$ROW_ID** → Unique and immutable ID for the row.

We will see these columns in the following examples.

Stream Examples

We will explain these examples step by step, showing the state of the stream at every moment.

First example: Create a stream from a table, add a new row and update the new row.
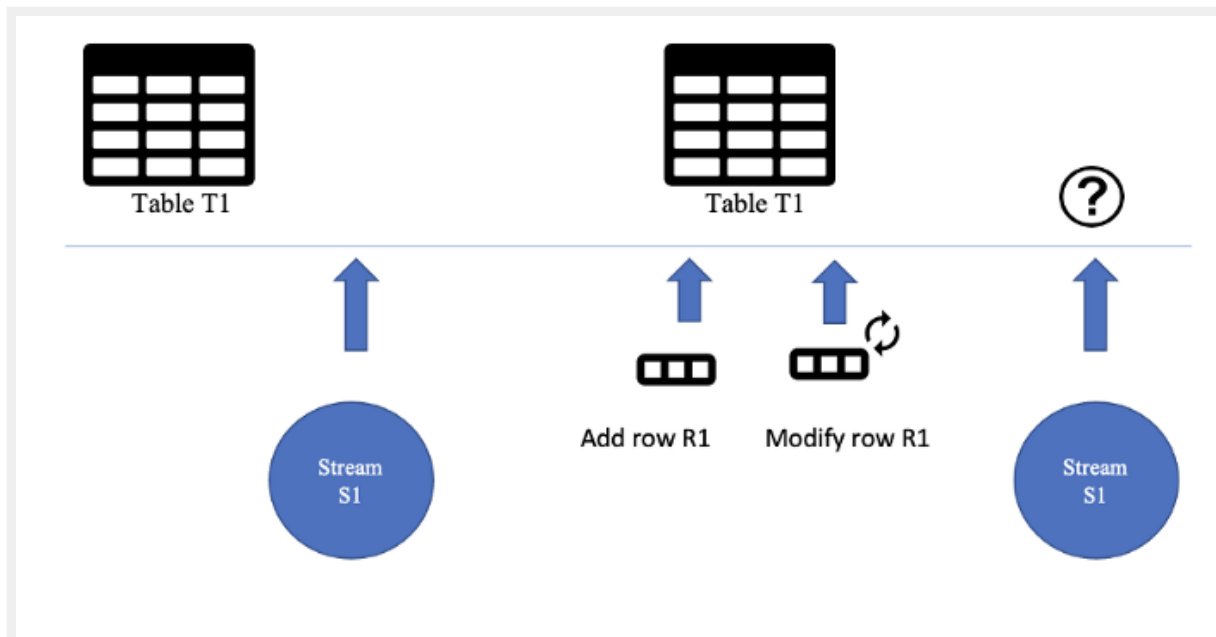
Streams in SnowFlake: **Example 1.**

1) **Create a Stream from a table** → The stream will be empty, it doesn't have any row.

```
+--------+---------+-----------+--+
| ROW_ID |  ACTION |  ISUPDATE |  |
+--------+---------+-----------+--+
|        |         |           |  |
+--------+---------+-----------+--+
```

2) **Add a row in the table** → As a new DML operation (INSERT) is done, SnowFlake will add a new entry in the stream. In this case, the action is "INSERT".

```
+--------+---------+-----------+
| ROW_ID |  ACTION |  ISUPDATE |
+--------+---------+-----------+
| #1234  |  INSERT |  FALSE    |
+--------+---------+-----------+
```

3) Update the row in the table → The stream will remain like in the previous case. Why is ISUPDATE false if the row has been updated? We need to see how the stream was at the beginning. When we first created it, the row didn't exist, so from the point of view of the stream, it's just a new row, not an updated one since it didn't exist at the beginning.

```
+--------+---------+-----------+
| ROW_ID |  ACTION |  ISUPDATE |
+--------+---------+-----------+
| #1234  |  INSERT |  FALSE    |
+--------+---------+-----------+
```

Second example: Add a new row in the table, create a stream from a table, and then modify the stream.

**1) Add a row in the table** → No stream has been created.

2) Create the stream

```
+--------+---------+-----------+--+
| ROW_ID |  ACTION |  ISUPDATE |  |
+--------+---------+-----------+--+
|        |         |           |  |
+--------+---------+-----------+--+
```

3) **Update the row in the table** → In this case, the row already existed. If we modify it, it's not a new row, it's just a modification from the table before creating the stream, so ISUPDATE will be true in this case.

```
+--------+---------+-----------+
| ROW_ID |  ACTION |  ISUPDATE |
+--------+---------+-----------+
| #1234  |  INSERT |  TRUE     |
+--------+---------+-----------+
```

Third example: Create a stream from a table, add a new row and delete the same row.
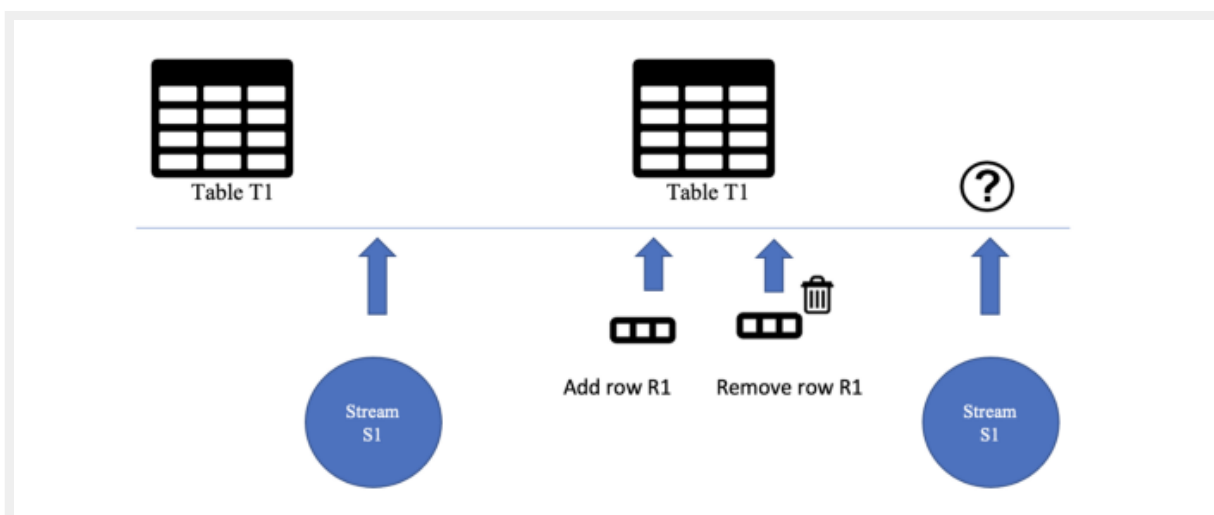
Streams in SnowFlake: Example 3.

1) **Create a Stream from a table** → The stream will be empty, it doesn't have any row.

```
+--------+---------+-----------+--+
| ROW_ID |  ACTION |  ISUPDATE |  |
+--------+---------+-----------+--+
|        |         |           |  |
+--------+---------+-----------+--+
```

2) **Add a row in the table** → As a new DML operation is done, SnowFlake will add a new entry in the stream. In this case, the action is "INSERT".

```
+--------+---------+-----------+
| ROW_ID |  ACTION |  ISUPDATE |
+--------+---------+-----------+
| #1234  |  INSERT |  FALSE    |
+--------+---------+-----------+
```

3) **Delete the row in the table** → The stream won't contain any information. Comparing this to the state of the table before creating the stream, it will be the same one. You just add a new row that you delete, so it will be exactly the same table state. So the stream will be empty. This is important to understand.

```
+--------+---------+-----------+--+
| ROW_ID |  ACTION |  ISUPDATE |  |
+--------+---------+-----------+--+
|        |         |           |  |
+--------+---------+-----------+--+
```

Fourth example: Add a new row in the table, create the stream and delete the same row.
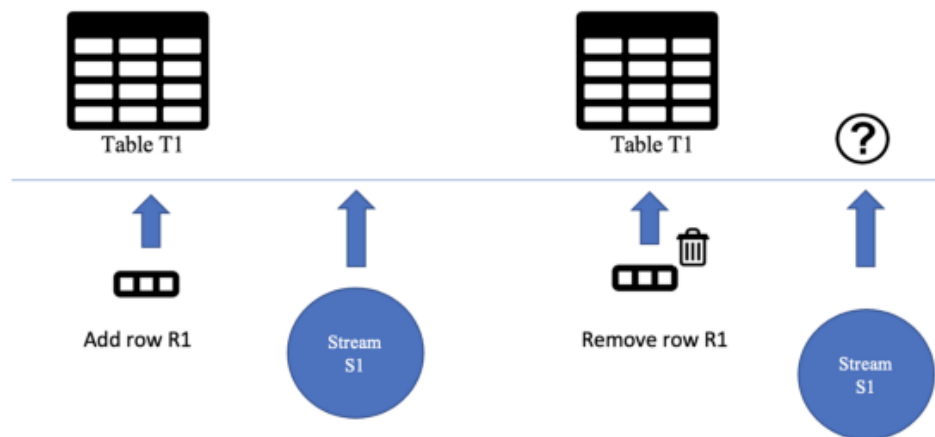
Streams in SnowFlake: Example 4.

1) **Add a row to the table**→ The stream has not been created yet.

2) **Create a stream from the table.**

```
+--------+---------+-----------+--+
| ROW_ID |  ACTION |  ISUPDATE |  |
+--------+---------+-----------+--+
|        |         |           |  |
+--------+---------+-----------+--+
```

3) **Delete the row in the table** → In this case, there is going to be a difference between the moment that you created the stream and this moment because the table will contain one less row. That's why the stream will contain this information.

```
+--------+---------+-----------+
| ROW_ID |  ACTION |  ISUPDATE |
+--------+---------+-----------+
| #1234  | DELETE  |  FALSE    |
+--------+---------+-----------+
```

All the examples show that the stream only contains the last DML action on a row.

TYPICAL EXAM QUESTIONS

1. Which object in SnowFlake stores DML change made to tables and metadata about each change?

1. Tables
2. Pipes
3. Table Streams
4. Account Streams

Solution: 3. This is the definition of streams, also known as Table Streams.

2. What statements are true about streams?

1. A Stream itself does NOT contain any table data
2. A stream only stores the offset for the source table
3. The hidden columns used by a stream consume storage

Solution: 1, 2, 3.

3. Which are the additional columns that the streams create?

1. METADATA$ACTION
2. METADATA$ISREAD

3. METADATA$ISUPDATE

4. METADATA$ROW_ID

5. METADATA$COLUMN_ID

Solution: 1, 3, 4.

4. What different types of streams exist in SnowFlake?

1. Standard

2. Append-only

3. Update-only

4. Insert-only

Solution: 1, 2, 4.

5. Do streams ensure exactly-once semantics for new or changed data in a table?

1. True

2. False

Solution: 1.

6. Are Insert-only streams only supported on external tables?

1. True
2. False

Solution: 1.

7. A stream has been created on a table. A row is inserted in the table, and it's updated later. Will the stream capture both events?

1. True
2. False

Solution: 2. This is the _first example_ that we've seen in this chapter.

8. Can we use streams in materialized views?

1. True
2. False

Solution: 2. As we explained in the chapter about tables and views, materialized views don't support streams at this moment.

9. Can multiple streams be created for the same table?

1. True
2. False

Solution: *1. Multiple streams can be created for the same table and consumed by different tasks.*

Thanks for Reading!