**Retract vs. Upsert in Flink Dynamic Tables**

In Apache Flink, **dynamic tables** evolve based on streaming data. Flink uses **changelog messages** to handle updates, and two primary modes for handling updates are **Retract Mode** and **Upsert Mode**.

**1. Retract Mode**

Retract mode is used when Flink needs to **delete or update** previously emitted rows. It works by sending two messages:

* **RETRACT** – Removes an old row.
* **INSERT** – Inserts the new row.

**Example: Counting Orders per Product**

Consider the following **Flink SQL query** that counts the number of orders per product\_id:

SELECT product\_id, COUNT(\*) AS order\_count

FROM orders

GROUP BY product\_id;

**Input Stream (orders Table)**

| **product\_id** | **order\_id** |
| --- | --- |
| A | 1 |
| A | 2 |
| B | 3 |
| A | 4 |

**Output Changelog (Retract Mode)**

| **product\_id** | **order\_count** | **Changelog Message** |
| --- | --- | --- |
| A | 1 | INSERT |
| A | 2 | RETRACT (1), INSERT (2) |
| B | 1 | INSERT |
| A | 3 | RETRACT (2), INSERT (3) |

**When to Use Retract Mode?**

* When computing **aggregations** (SUM, COUNT, AVG).
* When handling **updates** where previous results need to be corrected.
* When dealing with **stream-table joins** that require modifications.

**2. Upsert Mode**

Upsert mode optimizes updates by using a **primary key** and sending only:

* **INSERT/UPDATE (Upsert)** – If the key exists, the row is updated; otherwise, a new row is inserted.
* **DELETE (Optional)** – If a row is removed, a delete message is sent.

**Example: Tracking Latest Order Price per Product**

Consider a **Flink SQL query** that keeps the latest price for each product\_id:

SELECT product\_id, MAX(price) AS latest\_price

FROM orders

GROUP BY product\_id;

**Input Stream (orders Table)**

| **product\_id** | **price** |
| --- | --- |
| A | 100 |
| A | 120 |
| B | 200 |
| A | 130 |

**Output Changelog (Upsert Mode)**

| **product\_id** | **latest\_price** | **Changelog Message** |
| --- | --- | --- |
| A | 100 | INSERT |
| A | 120 | UPSERT (Overwrite 100 → 120) |
| B | 200 | INSERT |
| A | 130 | UPSERT (Overwrite 120 → 130) |

**When to Use Upsert Mode?**

* When dealing with **primary key-based updates**.
* When working with **deduplicated or latest-state data** (e.g., latest stock price, latest sensor reading).
* When writing to **external updatable sinks** like databases.

**Key Differences Between Retract and Upsert Mode**

| **Feature** | **Retract Mode** | **Upsert Mode** |
| --- | --- | --- |
| Update Mechanism | Retract old value, then insert new value | Directly upserts (insert or update) |
| Changelog Messages | INSERT, RETRACT | INSERT, UPSERT, DELETE |
| Required Primary Key | No | Yes |
| Used in Aggregations | Yes | No |
| Suitable for External Databases | No (because of RETRACT messages) | Yes (supports upserts) |