Lab7:Stream Processing

- docker-compose up
- docker exec -it ksqldb-cli ksql http://ksqldb-server:8088
- **Create Custom Type** `season_length`

  CREATE TYPE season_length AS STRUCT<season_id INT, episode_count INT>;
- Since changes are infinit and we must to store every single change we do Stream for it and movies as tables

```
CREATE TABLE titles (
    id INT PRIMARY KEY,
    title VARCHAR
) WITH (
    KAFKA_TOPIC='titles',
    VALUE_FORMAT='JSON',
    PARTITIONS=4
);
```

```
CREATE STREAM production_changes (
    rowkey VARCHAR KEY,
    title_id INT,
    change_type VARCHAR,
    before season_length,
    after season_length,
    created_at TIMESTAMP
) WITH (
    KAFKA_TOPIC='production_changes',
    VALUE_FORMAT='JSON',
    TIMESTAMP='created_at',
    PARTITIONS=4
);
```

- then we insert values:

```
INSERT INTO production_changes (
    rowkey, title_id, change_type, before, after, created_at
) VALUES (
    'key1', 1, 'season_length',
    STRUCT(season_id := 1, episode_count := 12),
    STRUCT(season_id := 1, episode_count := 8),
    '2021-02-08T11:30:00' -- Correct timestamp format
);


INSERT INTO titles (id, title) VALUES (1, 'Stranger Things');
```

- Query 5 :

  first run: SET 'auto.offset.reset' = 'earliest';

  then:

```
SELECT * FROM production_changes
WHERE created_at < '2023-04-14T12:00:00'
EMIT CHANGES;
```

- Query6:

```
SELECT * FROM production_changes
WHERE change_type LIKE 'season%'
EMIT CHANGES;
```

- 7)

```
CREATE STREAM season_length_changes
WITH (
    KAFKA_TOPIC='season_length_changes',
    VALUE_FORMAT='JSON',
    PARTITIONS=4,
    REPLICAS=1
) AS SELECT
    title_id AS KEY,
    ROWKEY,
```

```
    title_id,
    COALESCE(after->season_id, before->season_id) AS season_id,
    before->episode_count AS old_episode_count,
    after->episode_count AS new_episode_count,
    created_at
FROM production_changes
WHERE change_type = 'season_length'
EMIT CHANGES;
```

- 8)

```
SELECT s.ROWKEY, s.title_id, t.title
FROM season_length_changes s
LEFT JOIN titles t ON s.title_id = t.id
EMIT CHANGES;
```

9. `season_length_changes_enriched`

   to show: `DESCRIBE season_length_changes_enriched;`

```
CREATE STREAM season_length_changes_enriched
WITH (
    TIMESTAMP='created_at'
) AS SELECT
    s.ROWKEY,
    s.title_id,
    s.season_id,
    s.old_episode_count,
    s.new_episode_count,
    s.created_at,
    t.title AS title
FROM season_length_changes s
LEFT JOIN titles t ON s.title_id = t.id
EMIT CHANGES;
```

10. `season_length_change_counts`

    window means we aggregate all the data of an hour to be in one hour like each hour we store all the data of it.

```
CREATE TABLE season_length_change_counts
WITH (
    KAFKA_TOPIC='season_length_change_counts',
    VALUE_FORMAT='JSON',
    KEY_FORMAT='JSON'
) AS SELECT
    title_id,
    title,
    COUNT(*) AS change_count,
    LATEST_BY_OFFSET(new_episode_count) AS latest_episode_count,
    WINDOWSTART AS window_start,
    WINDOWEND AS window_end
FROM season_length_changes_enriched
WINDOW TUMBLING (SIZE 1 HOUR)
GROUP BY title_id, title
EMIT CHANGES;
```