OSA CON 24

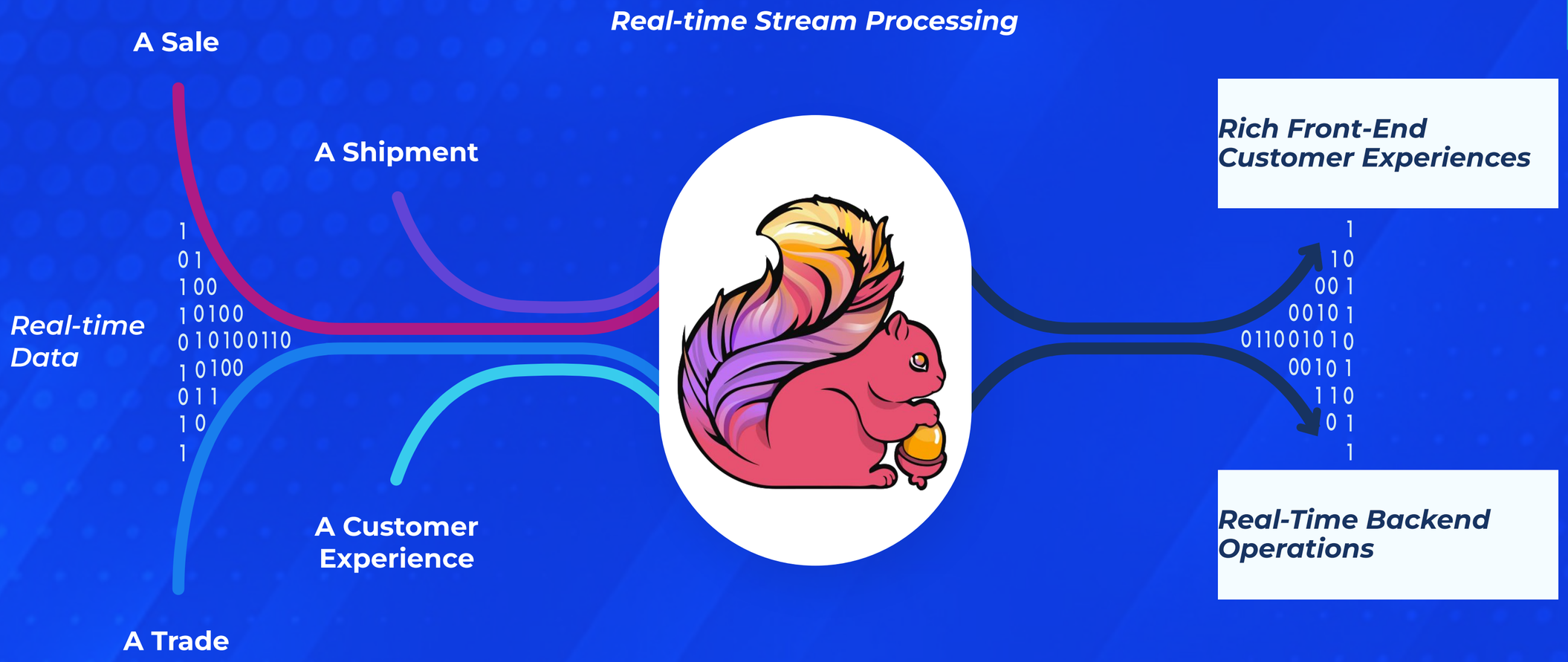
Flink for a non-JVM user, an introduction to pyflink

DIPTIMAN RAICHAUDHURI
Staff Developer Advocate - Confluent
https://www.linkedin.com/in/diptimanrc/

November 19-21, 2024

Today’s consumers expect real-time services

Real-time services rely on data stream processing



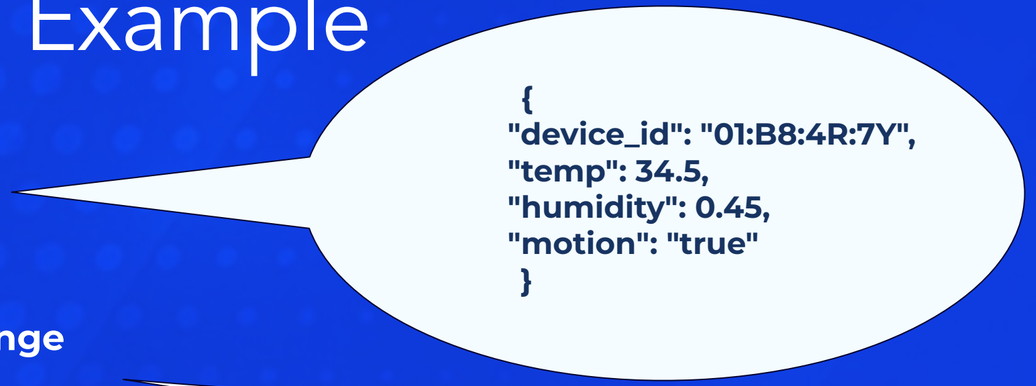
What is Data Stream Processing ?

{Data} Stream processing

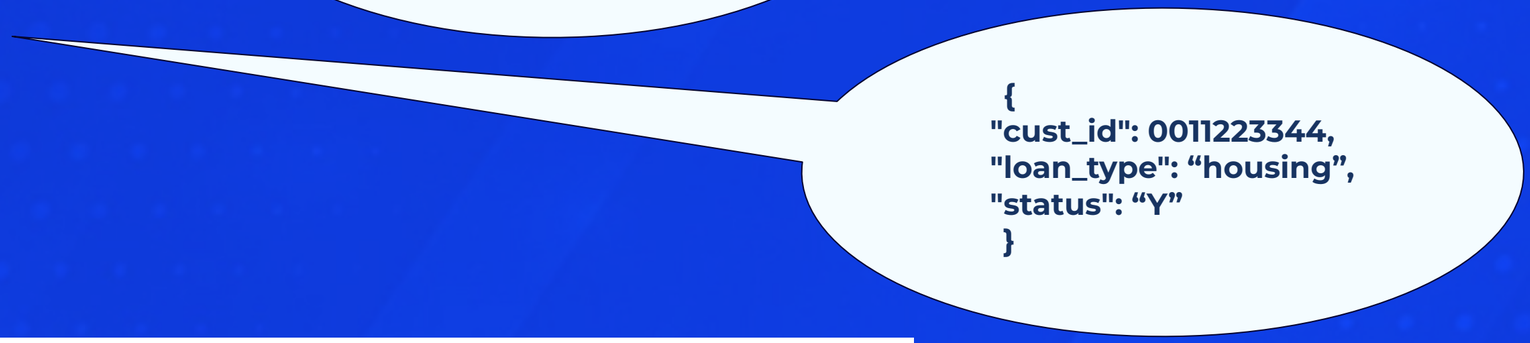
Data Stream at a minimum

Data Stream - Example

- **Internet of Things**

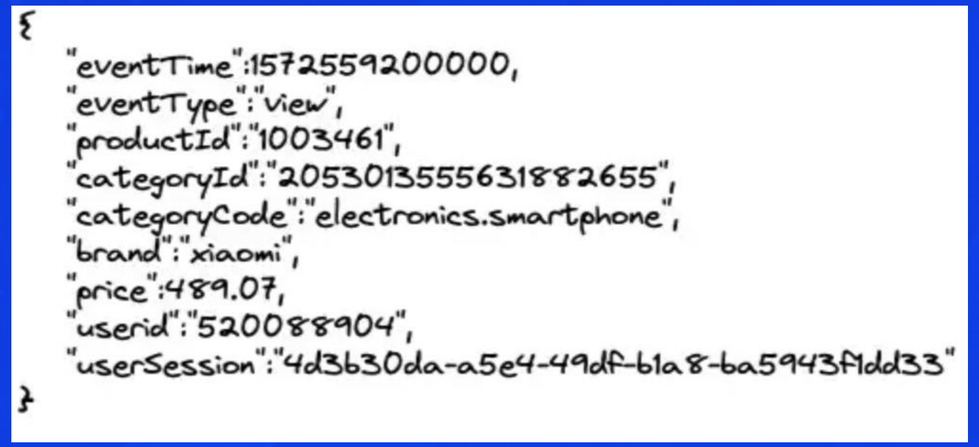


- **Business process change**

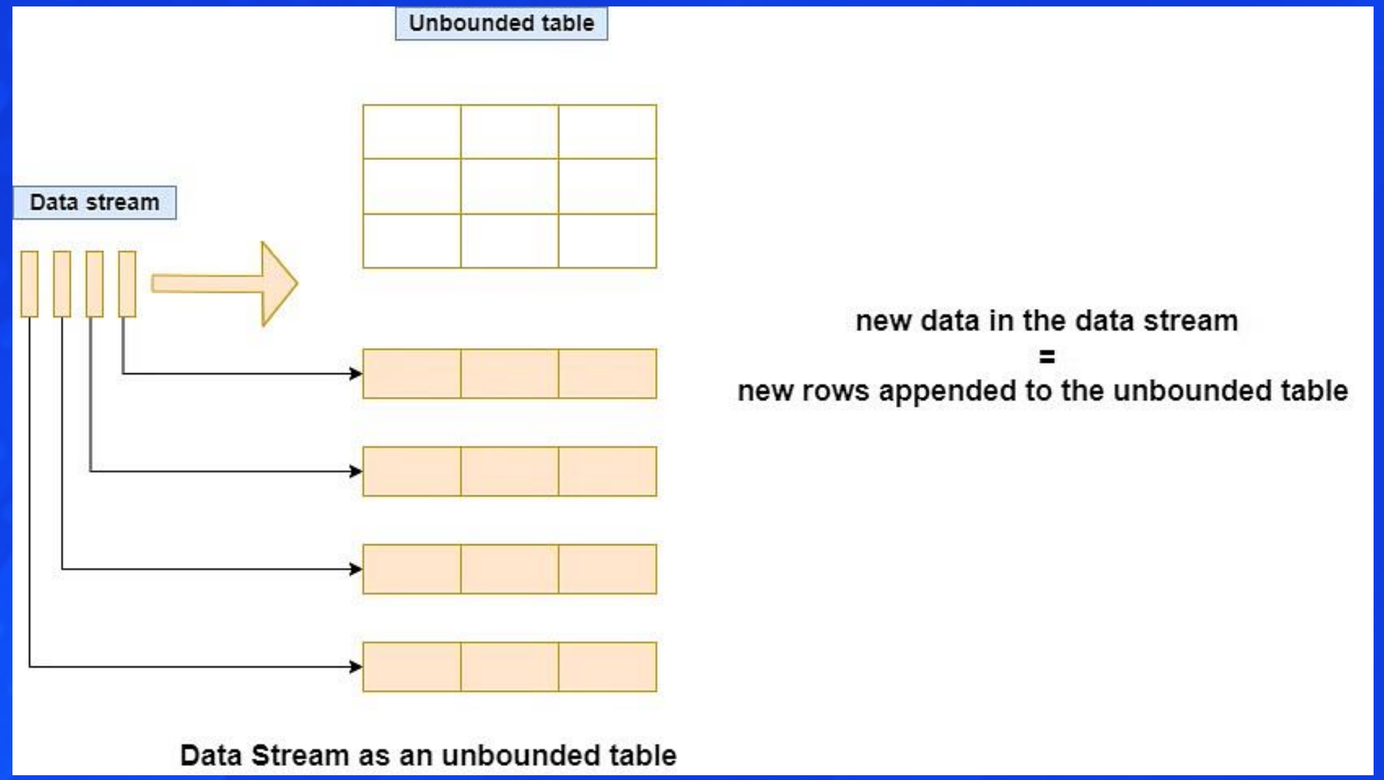


- **User Interaction**

- **Microservice output**



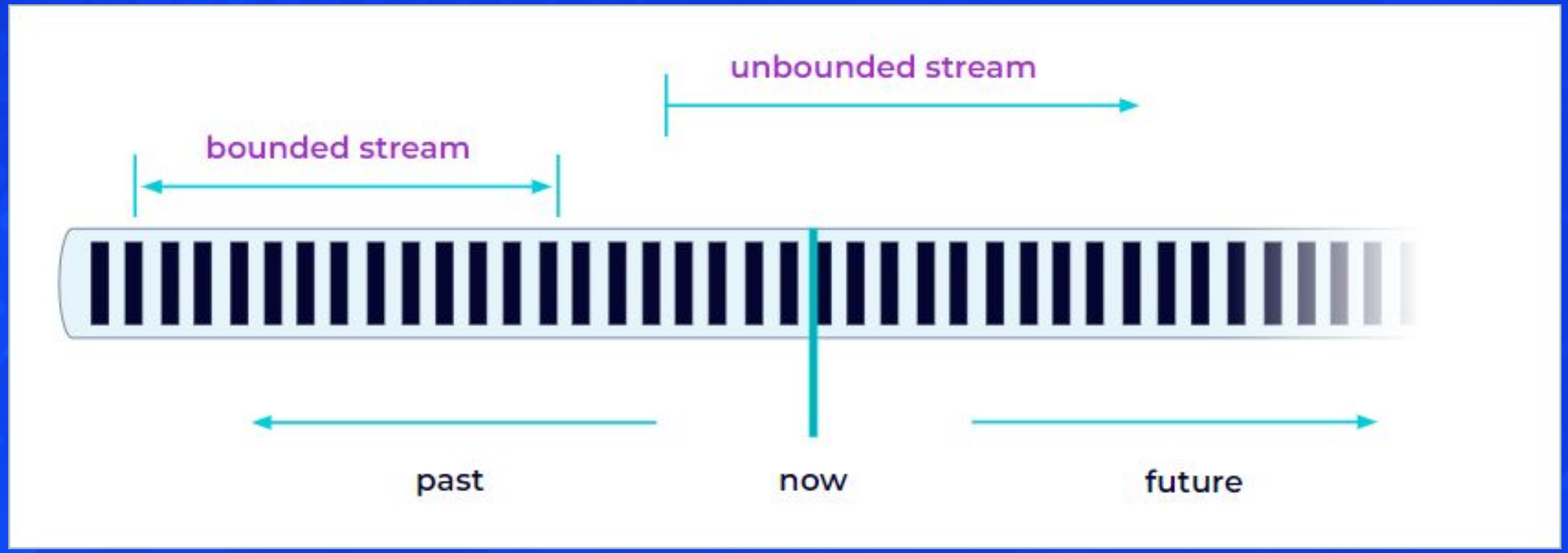
Data Stream as an unbounded table



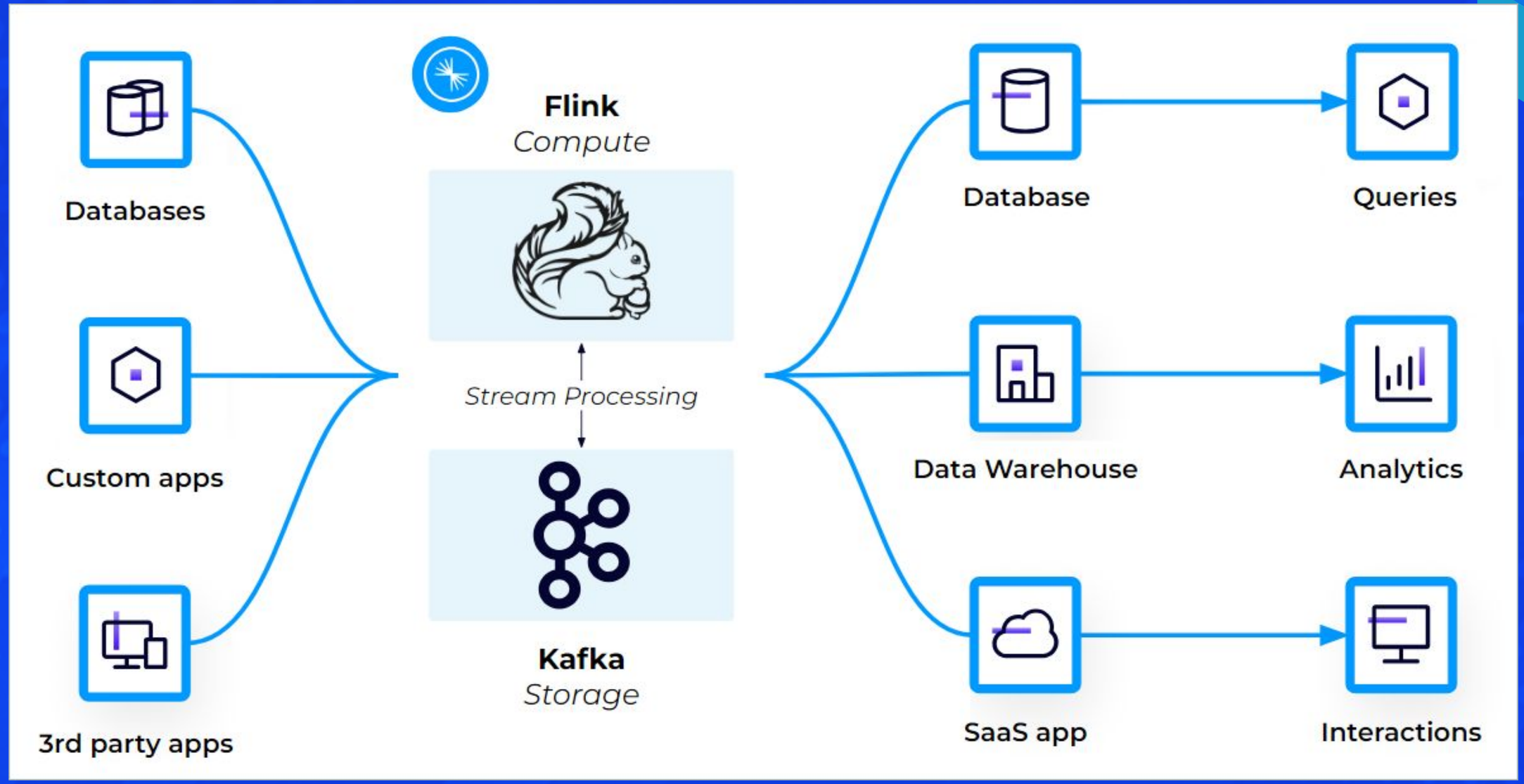
Data Stream as an unbounded table

- **Data ingested through an unbounded context.**
- **Data ingested perpetually, till the data producers stop**

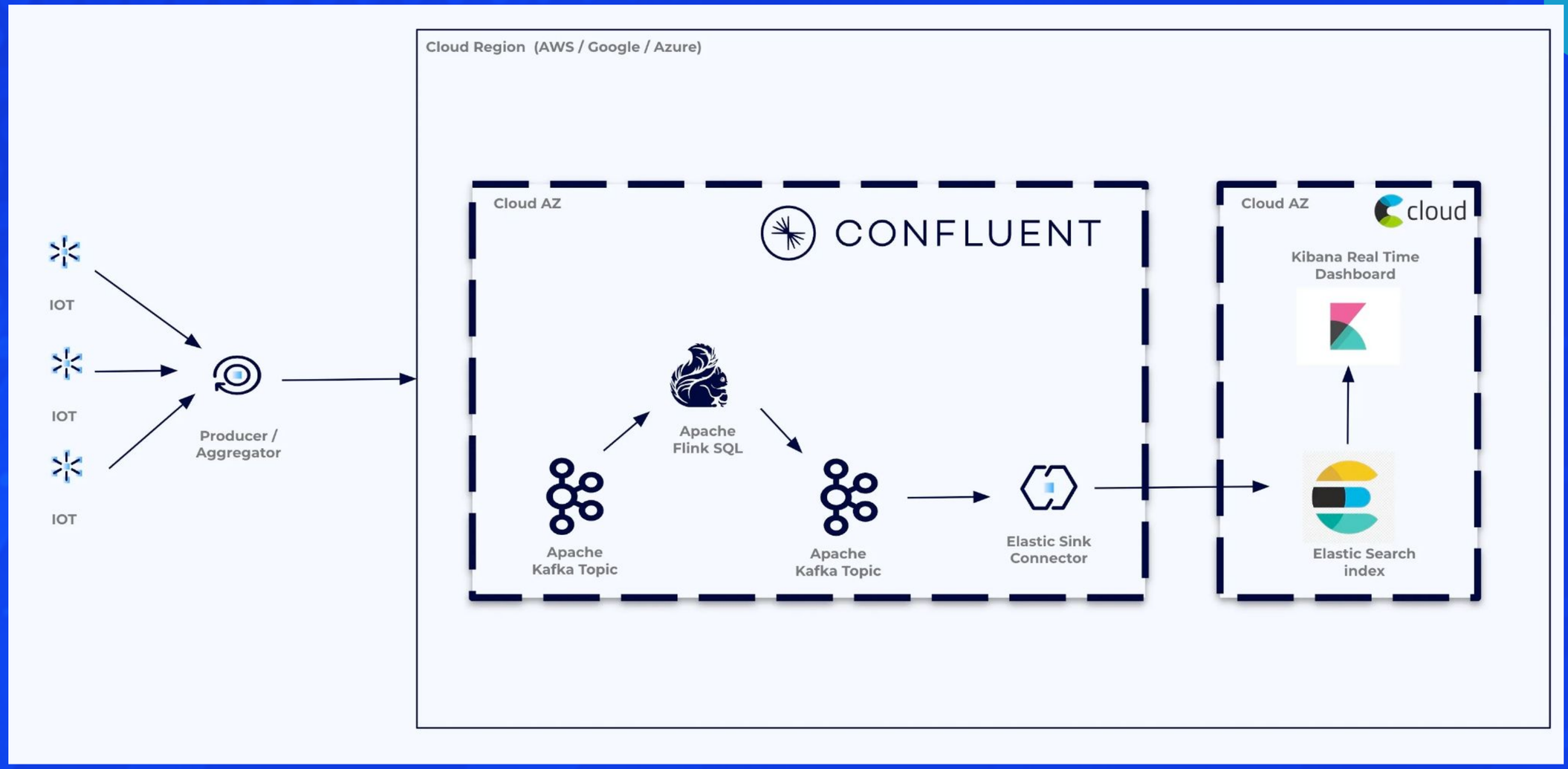
Data Stream - Timeline



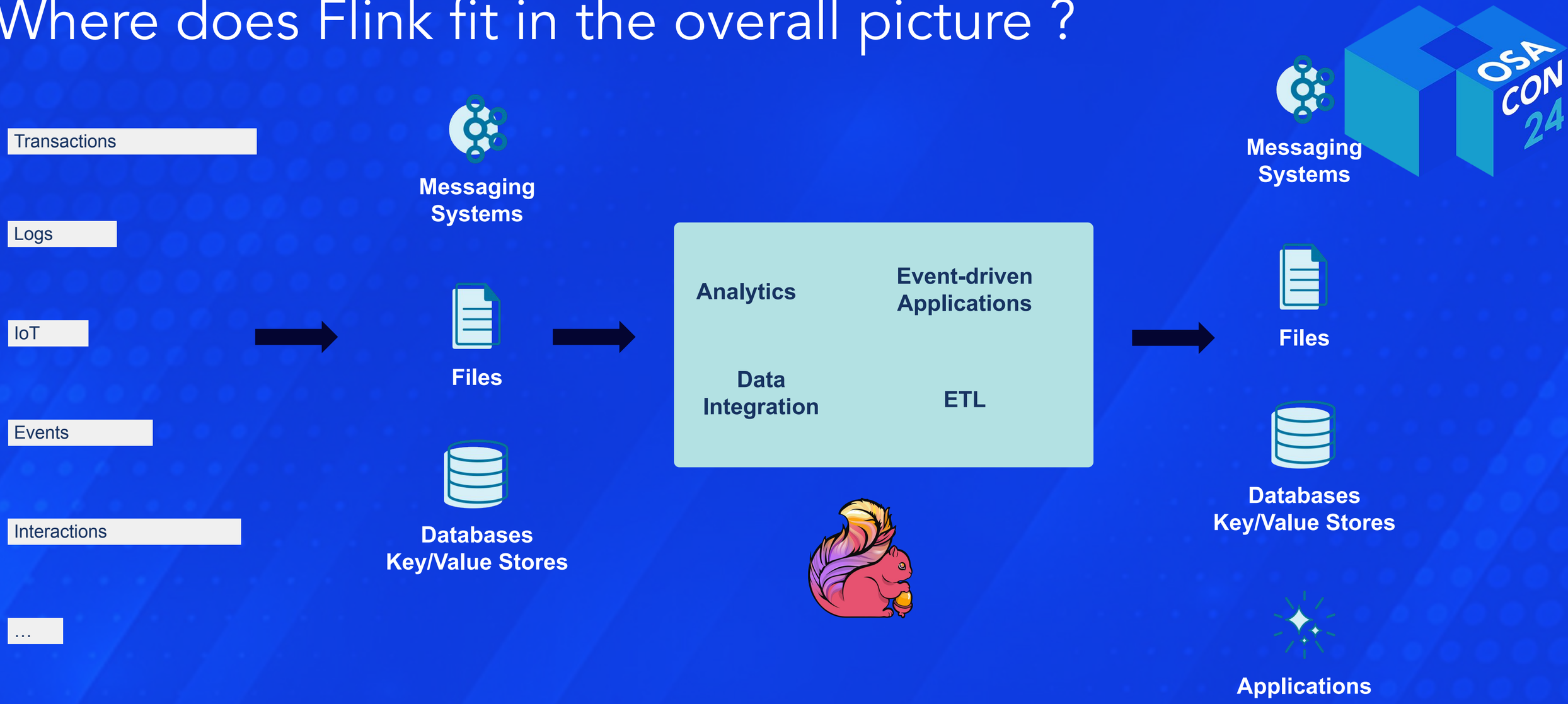
Data Streaming Platform - Components



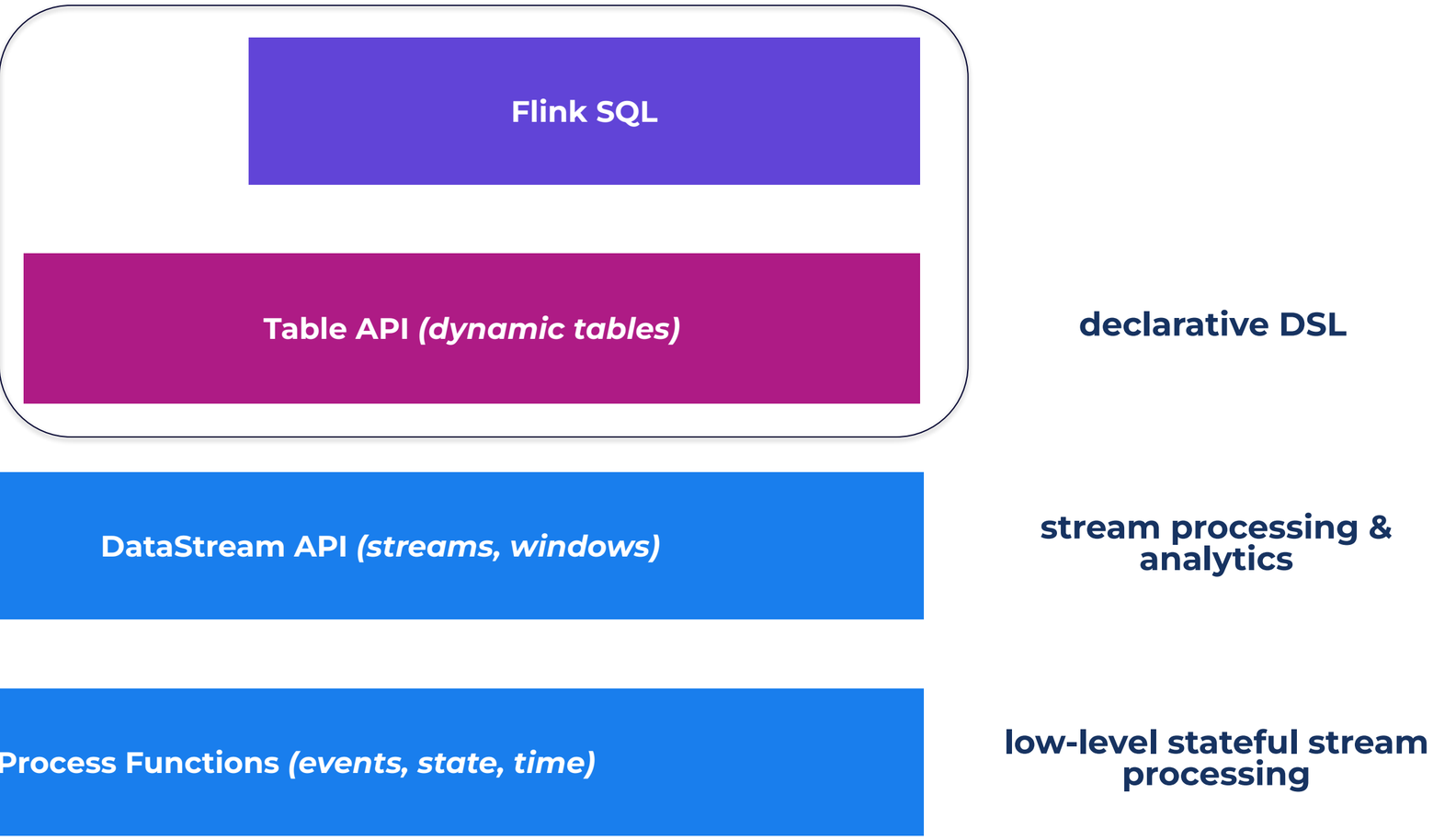
A real world example



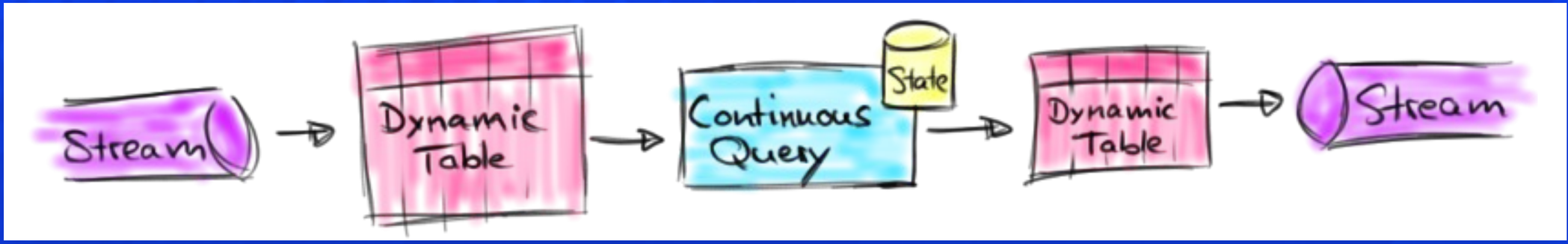
Where does Flink fit in the overall picture ?

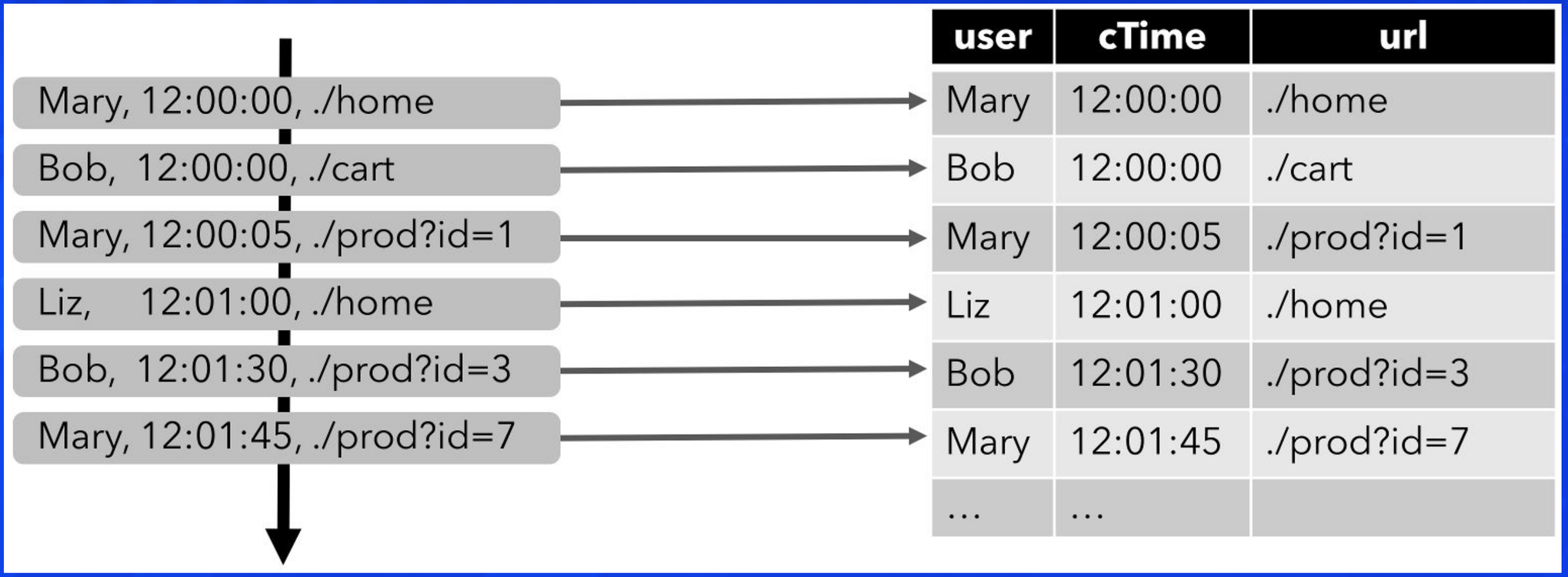


Flink's APIs


Flink SQL
Table API (dynamic tables)
declarative DSL
DataStream API (streams, windows)
stream processing & analytics
Process Functions (events, state, time)
low-level stateful stream processing

Everything revolves around a Flink Dynamic Table



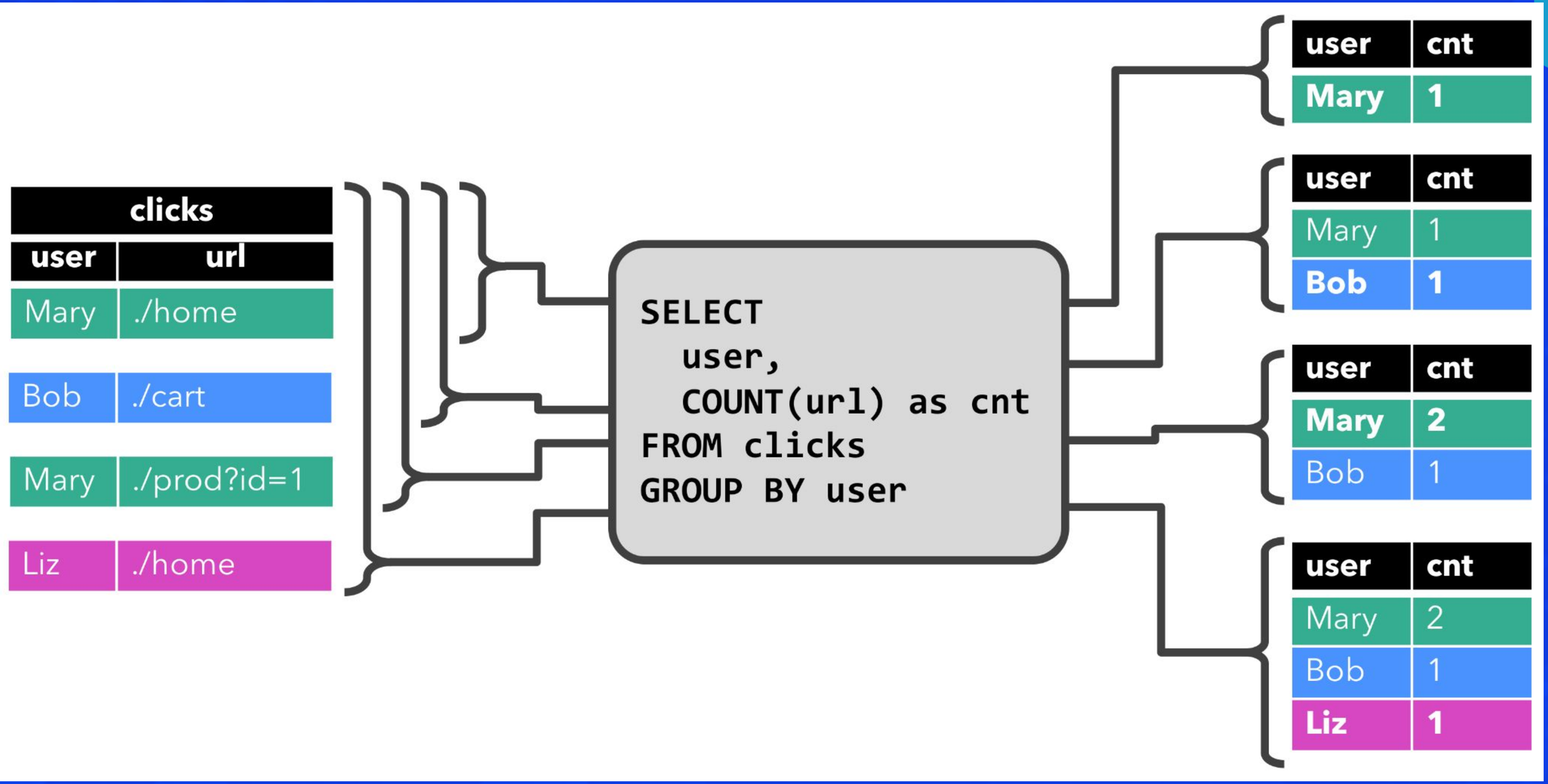


user	cTime	url
Mary	12:00:00	./home
Bob	12:00:00	./cart
Mary	12:00:05	./prod?id=1
Liz	12:01:00	./home
Bob	12:01:30	./prod?id=3
Mary	12:01:45	./prod?id=7
…	…	

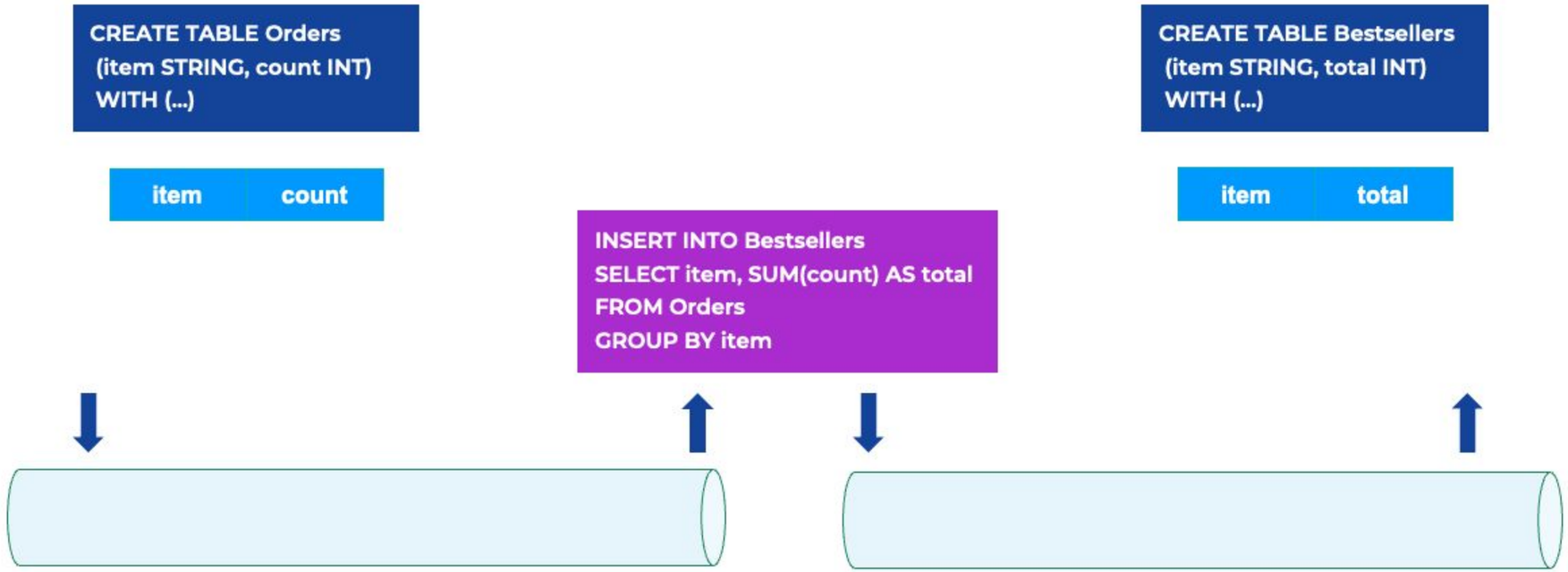
What is a Flink Dynamic Table

- **Dynamic tables change over time**
- **Querying dynamic tables yields a Continuous Query**
- **A continuous query never terminates and produces dynamic results -> another dynamic table**

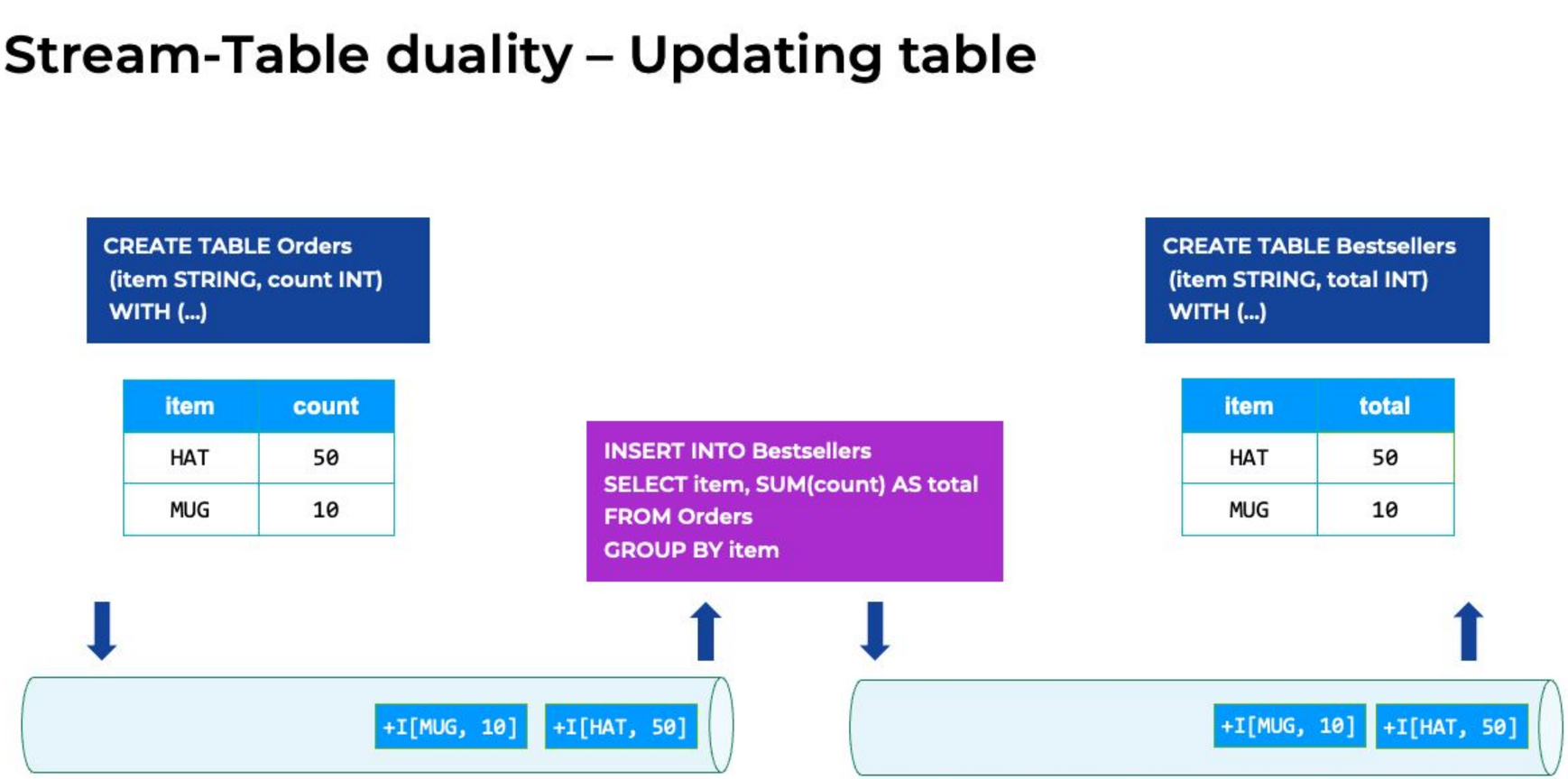
Aggregation on a Flink Dynamic Table



Flink Kafka - Stream Table Duality



Flink Kafka - Stream Table Duality



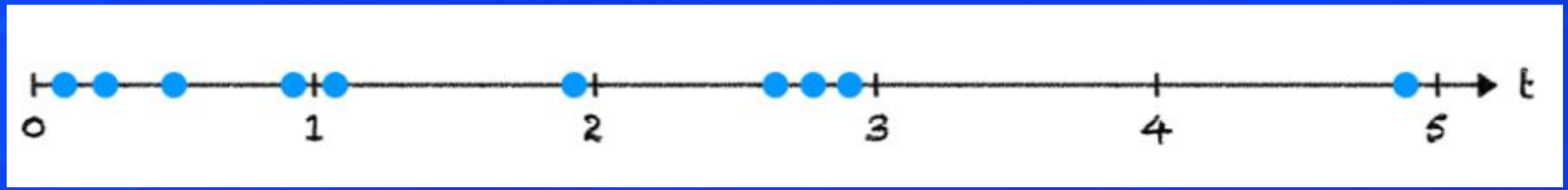
Flink Kafka - Stream Table Duality

- **Append only stream - INSERT**

- **Retract Stream - INSERT:Add + DELETE:Retract + UPDATE:Retract**

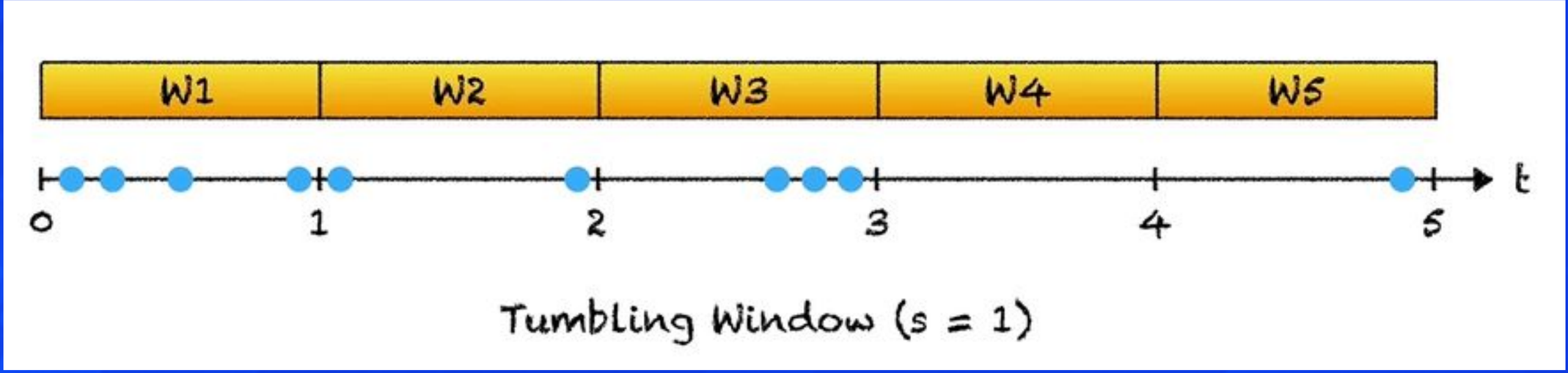
- **Upsert Stream - UPSERT + DELETE**
  - **Main diff with Retract - changes are encoded with a single message and hence more efficient**
-

Flink Window Functions

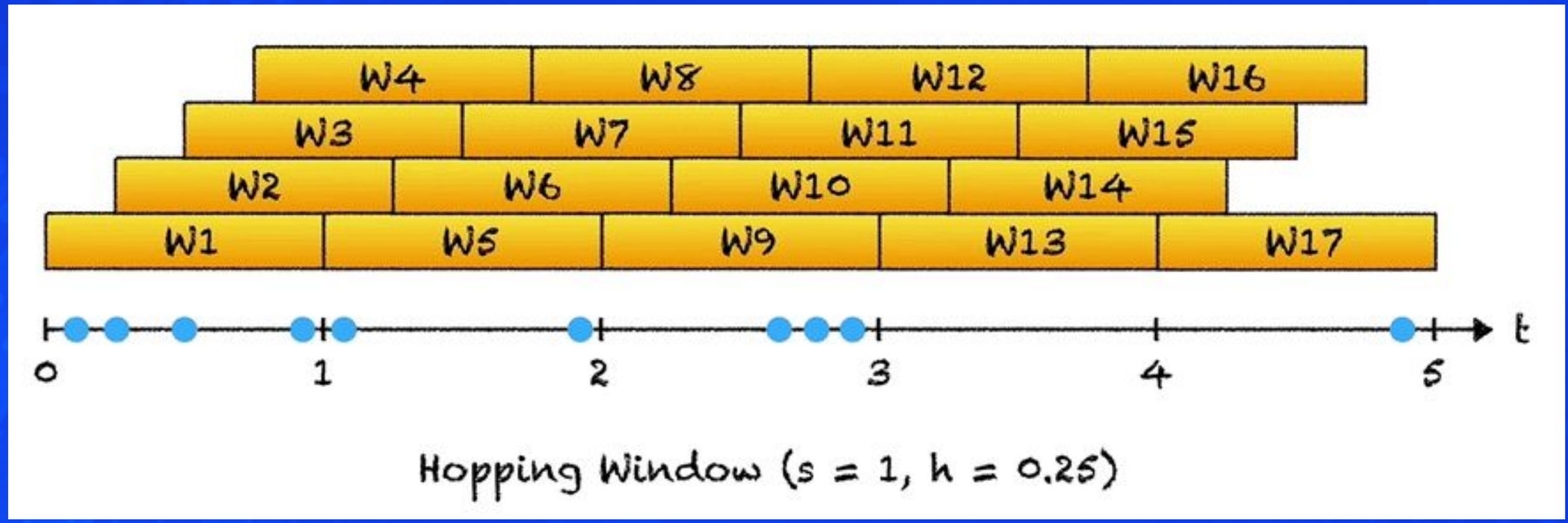


- **Data events over time**

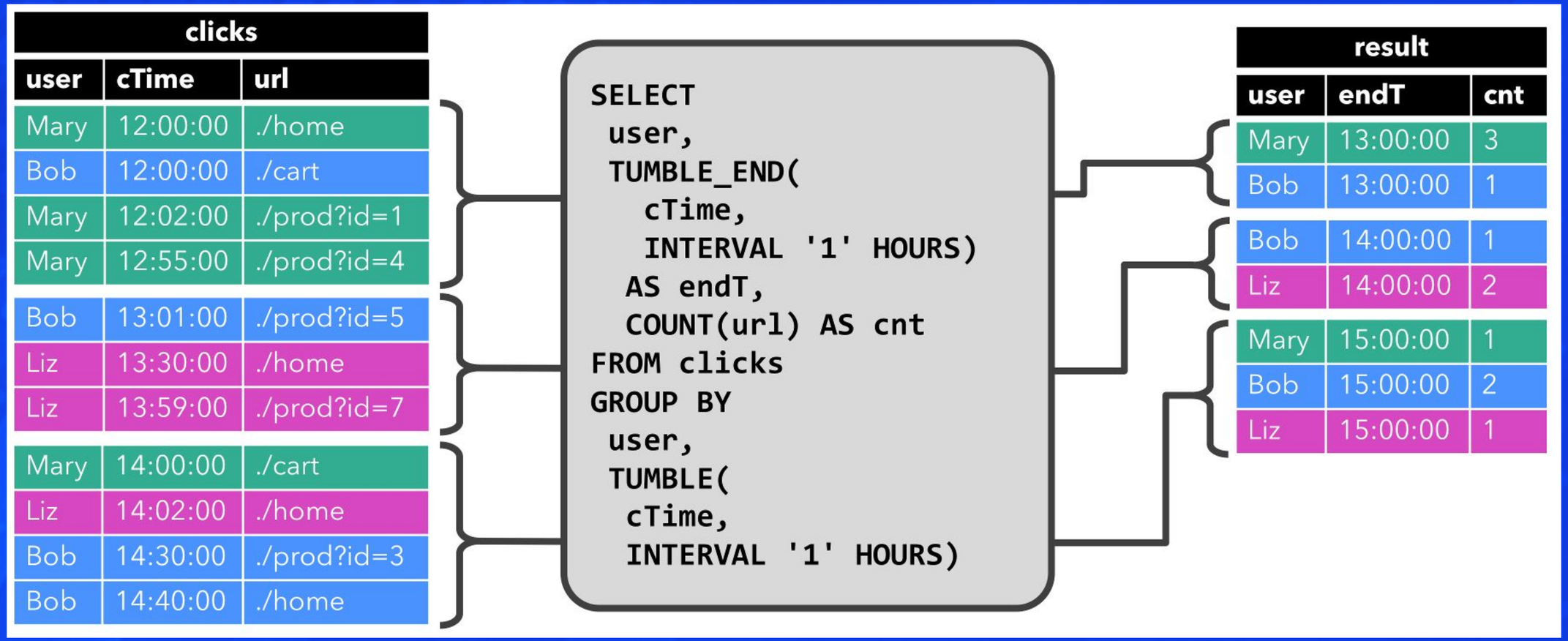
Flink Tumbling Window



Flink Hopping Window



Flink Dynamic Table - Windowing Example



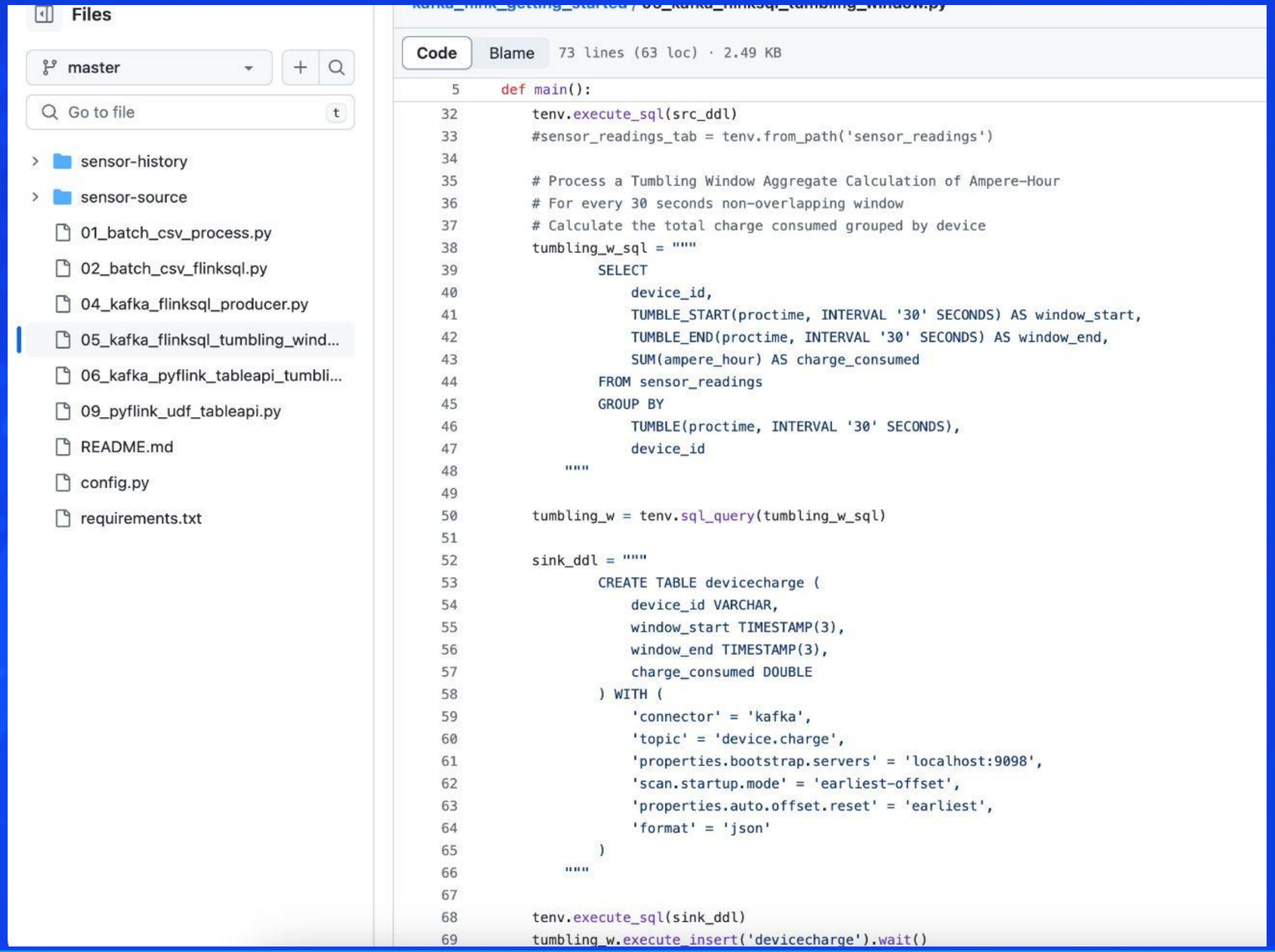
Pyflink Table API in action


```
def main():
    env = StreamExecutionEnvironment.get_execution_environment()
    settings = EnvironmentSettings.in_streaming_mode()
    tenv = StreamTableEnvironment.create(env, settings)

    env.add_jars("file:///D:\\testing_space\\PycharmProjects\\kafka-flink-getting-started\\flink-sql-connector-kafka-3.1.0-1.18.

    src_ddl = """
            CREATE TABLE sensor_readings (
                device_id VARCHAR,
                co DOUBLE,
                humidity DOUBLE,
                motion BOOLEAN,
                temp DOUBLE,
                ampere_hour DOUBLE,
                ts BIGINT,
                proctime AS PROCTIME()
            ) WITH (
                'connector' = 'kafka',
                'topic' = 'sensor.readings',
                'properties.bootstrap.servers' = 'localhost:9098',
                'properties.group.id' = 'device.tumbling.w.sql',
                'scan.startup.mode' = 'earliest-offset',
                'properties.auto.offset.reset' = 'earliest',
                'format' = 'json'
            )
        """

    tenv.execute_sql(src_ddl)
    sensor_readings_tab = tenv.from_path('sensor_readings')
```

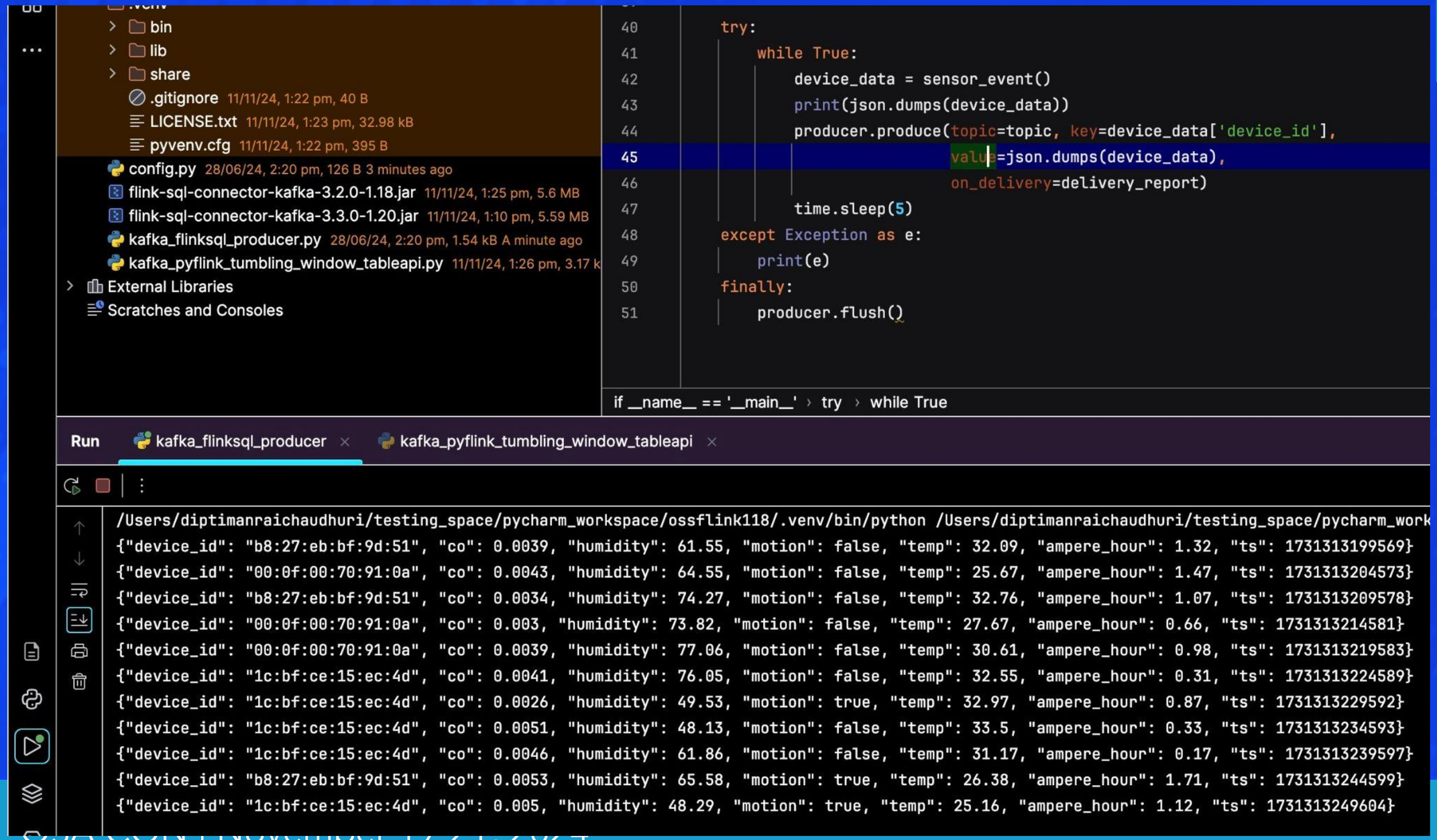
Pyflink Table API - Tumbling Window


```
# Define a Tumbling Window Aggregate Calculation of ampere-hour sensor readings
# - For every 30 seconds non-overlapping window
# - Sum of charge consumed by each device
tumbling_w = sensor_readings_tab.window(Tumble.over(lit(30).seconds)
                                        .on(sensor_readings_tab.proctime)
                                        .alias('w')) \
             .group_by(col('w'), sensor_readings_tab.device_id) \
             .select(sensor_readings_tab.device_id,
                 col('w').start.alias('window_start'),
                 col('w').end.alias('window_end'),
                 sensor_readings_tab.ampere_hour.sum.alias('charge_consumed'))
```

Pyflink Table API to Flink SQL



Data Streaming Source



Pyflink Table API - Tumbling Window 30 sec interval

server-3-start.sh

Terminal | server-1-start.sh | server-2-start.sh | server-3-start.sh | console-consumer.sh

```
diptimanraichaudhuri@QY4TFWR5QP scripts_kraft % /bin/zsh /Users/diptimanraichaudhuri/testing_space/intellij_workpace/oss-kraft-370/scripts_kraft/console-consumer.sh
{"device_id":"1c:bf:ce:15:ec:4d","window_start":"2024-11-11 13:51:00","window_end":"2024-11-11 13:51:30","charge_consumed":3.1199999999999997}
{"device_id":"00:0f:00:70:91:0a","window_start":"2024-11-11 13:51:00","window_end":"2024-11-11 13:51:30","charge_consumed":6.550000000000001}
{"device_id":"b8:27:eb:bf:9d:51","window_start":"2024-11-11 13:51:00","window_end":"2024-11-11 13:51:30","charge_consumed":6.129999999999999}

{"device_id":"b8:27:eb:bf:9d:51","window_start":"2024-11-11 13:51:30","window_end":"2024-11-11 13:52:00","charge_consumed":0.99}
{"device_id":"1c:bf:ce:15:ec:4d","window_start":"2024-11-11 13:51:30","window_end":"2024-11-11 13:52:00","charge_consumed":1.03}
{"device_id":"00:0f:00:70:91:0a","window_start":"2024-11-11 13:51:30","window_end":"2024-11-11 13:52:00","charge_consumed":3.15}
```

Thank You + Q&A

Reach Me At :

- https://www.linkedin.com/in/diptimanrc/

My Blog

- https://diptimanrc.medium.com/