

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

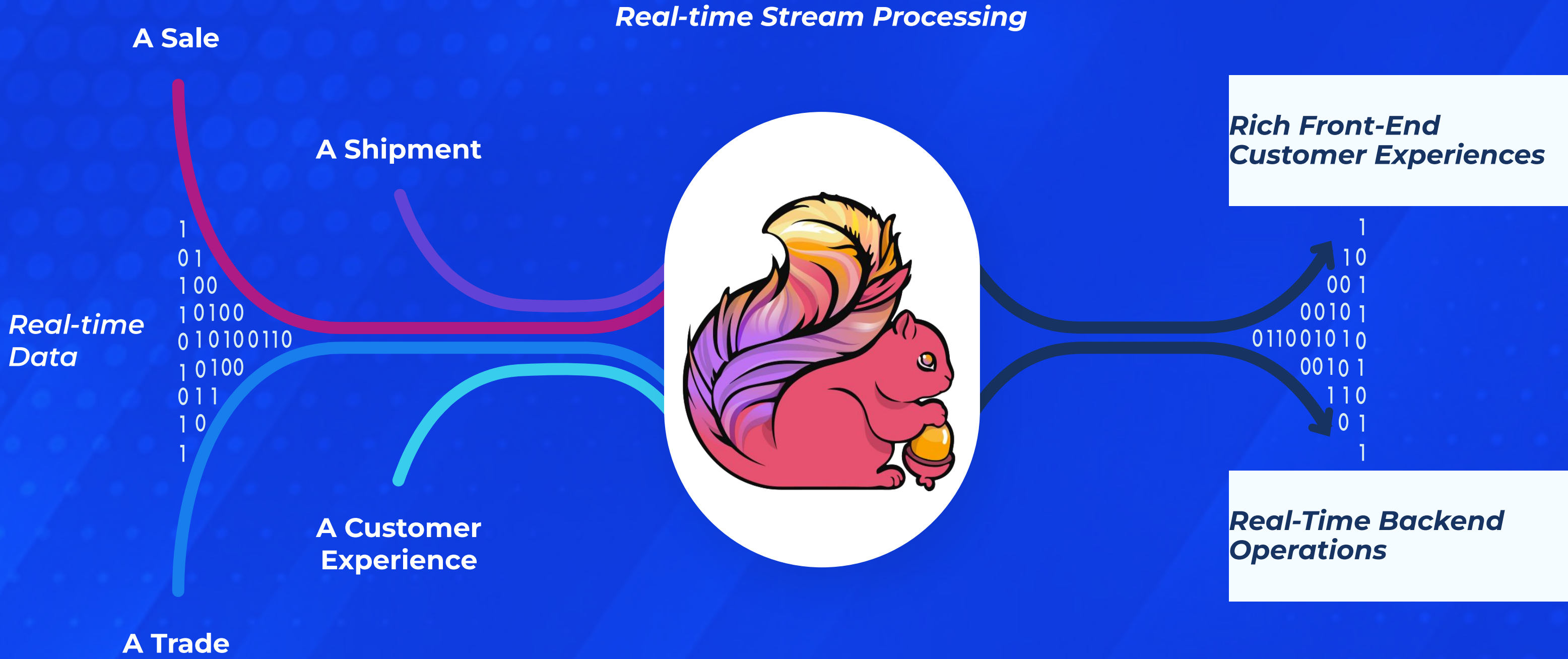


# Today's consumers expect real-time services





# 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

```
{  
  "device_id": "01:B8:4R:7Y",  
  "temp": 34.5,  
  "humidity": 0.45,  
  "motion": "true"  
}
```

```
{  
  "cust_id": 0011223344,  
  "loan_type": "housing",  
  "status": "Y"  
}
```

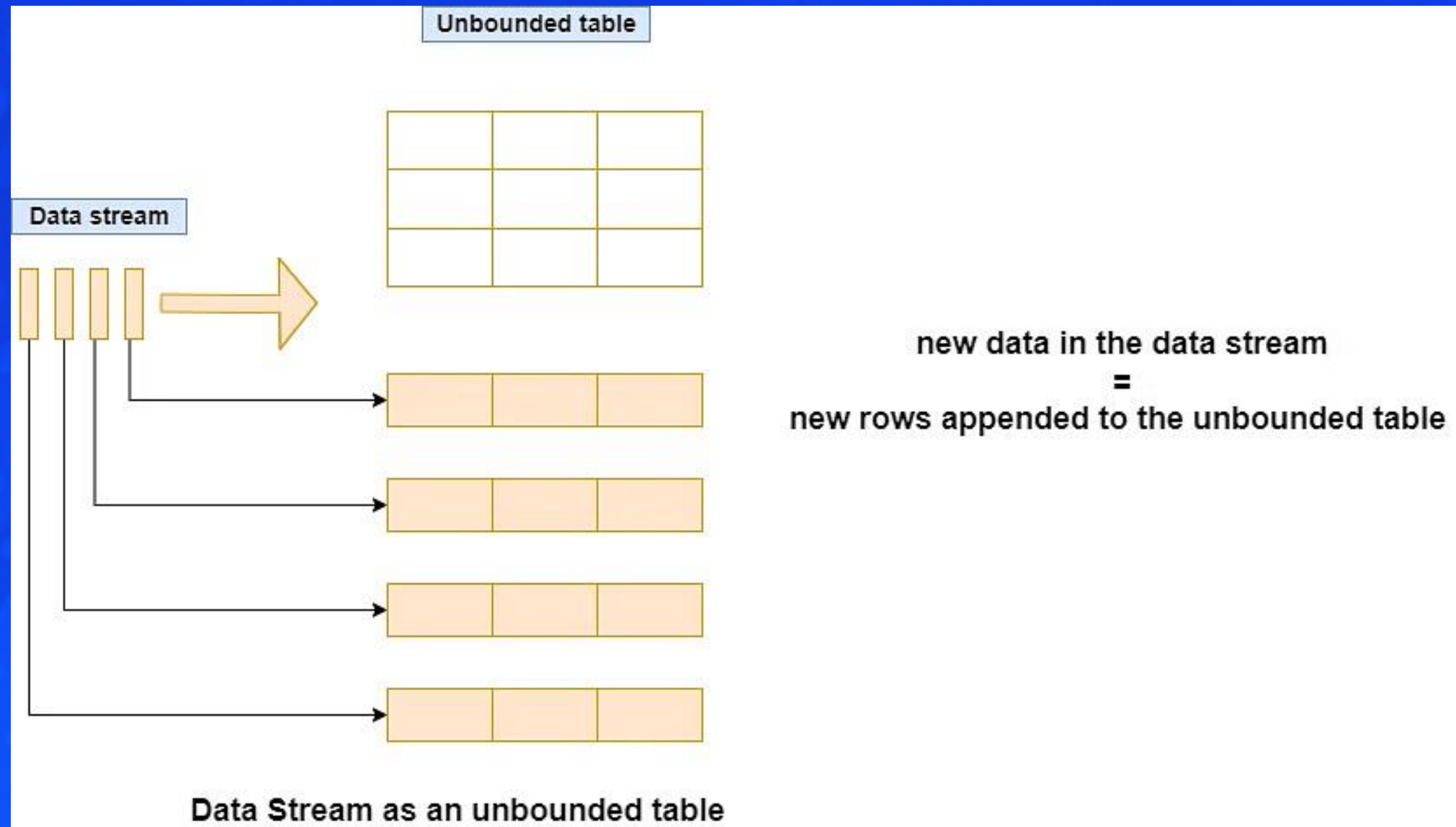
```
{  
  "eventTime":1572559200000,  
  "eventType":"view",  
  "productId":"1003461",  
  "categoryId":"2053013555631882655",  
  "categoryCode":"electronics.smartphone",  
  "brand":"xiaomi",  
  "price":489.07,  
  "userid":"520088904",  
  "userSession":"4d3b30da-a5e4-49df-b1a8-ba5943f1dd33"  
}
```



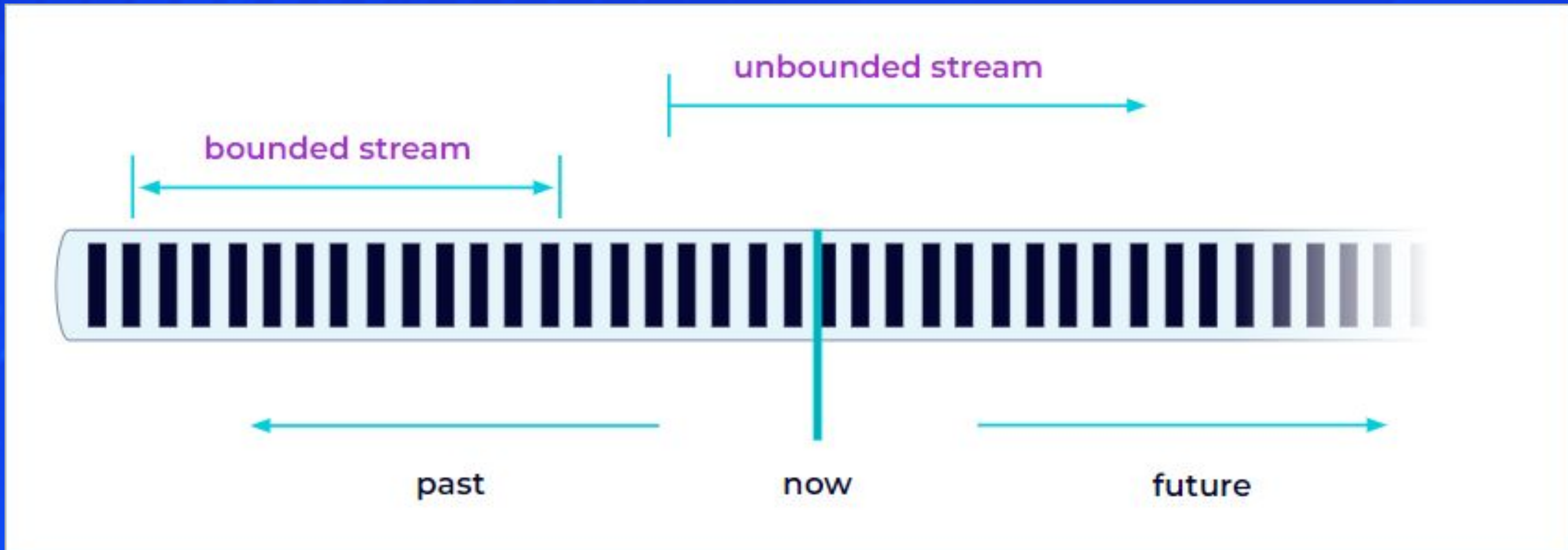
# Data Stream as an unbounded table



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

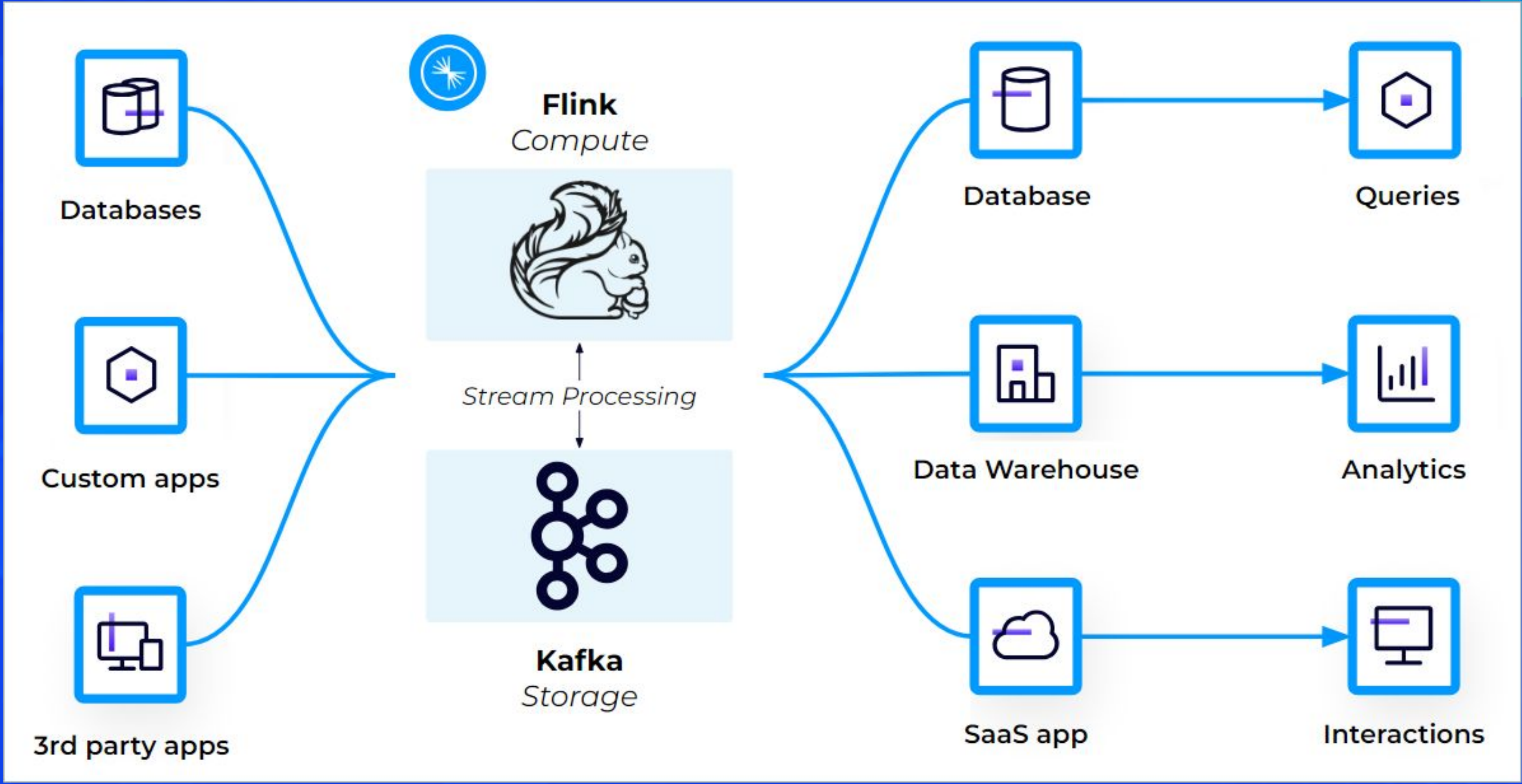


# Data Stream - Timeline



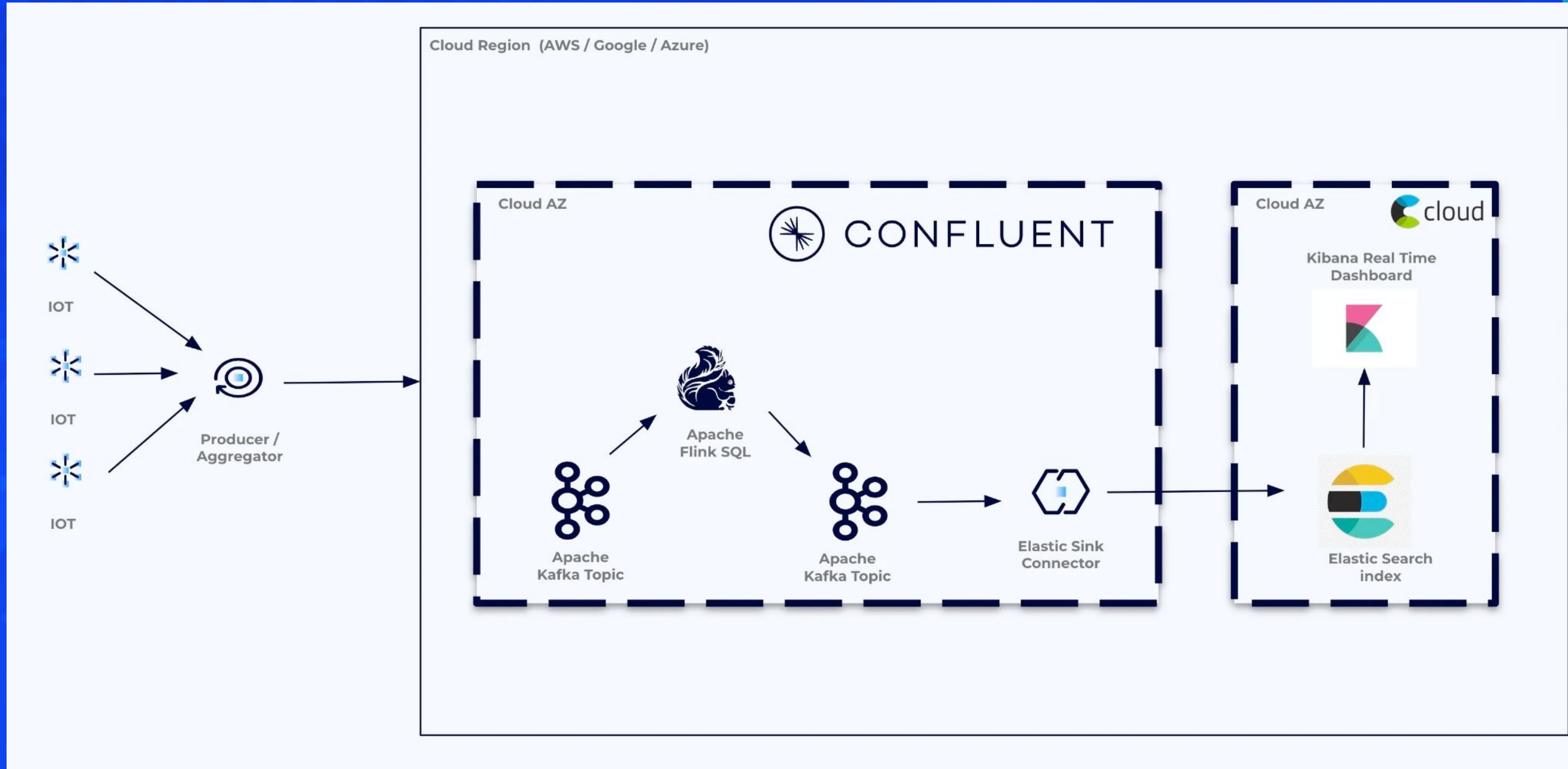


# Data Streaming Platform - Components



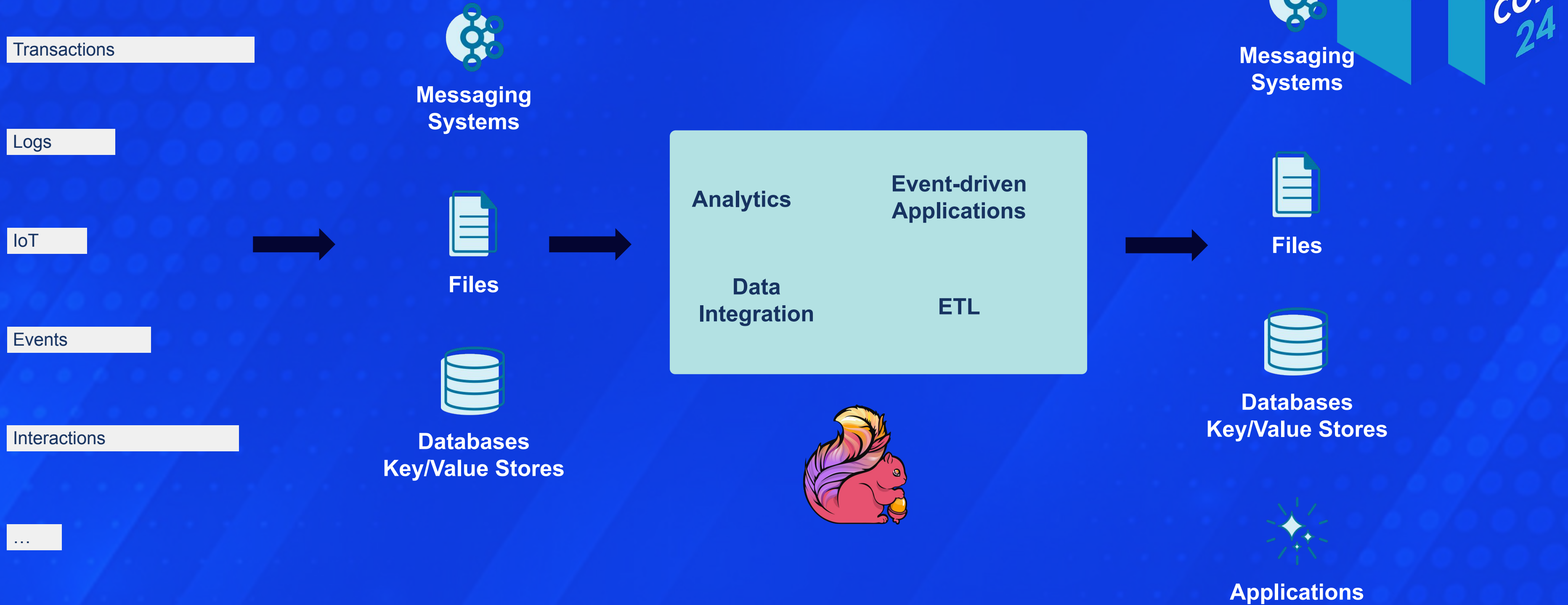


# A real world example



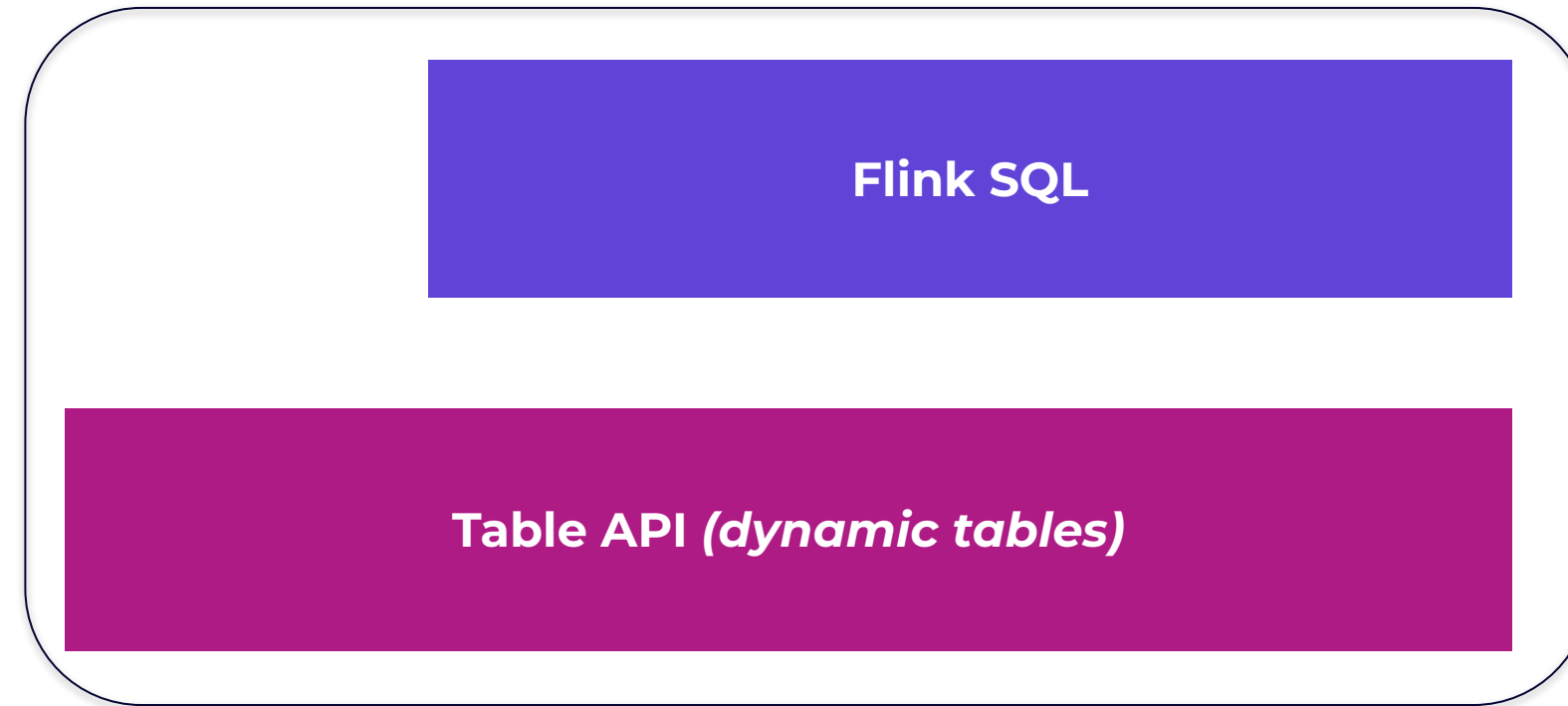


# Where does Flink fit in the overall picture ?

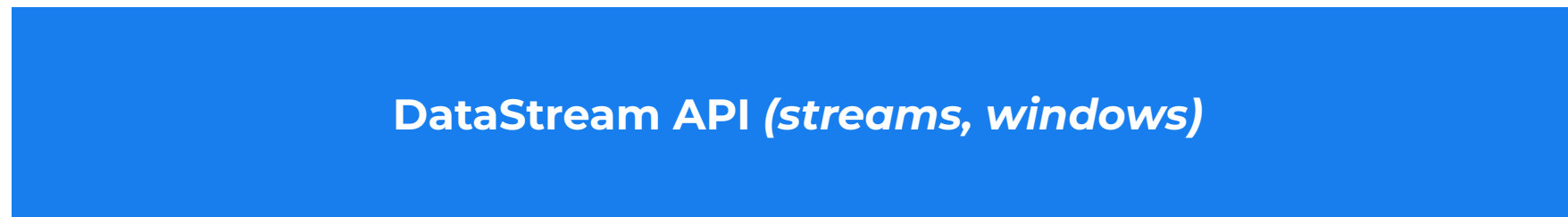




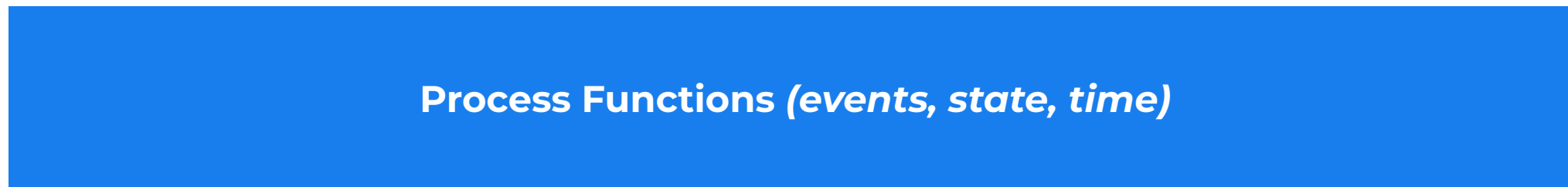
# Flink's APIs



**declarative DSL**

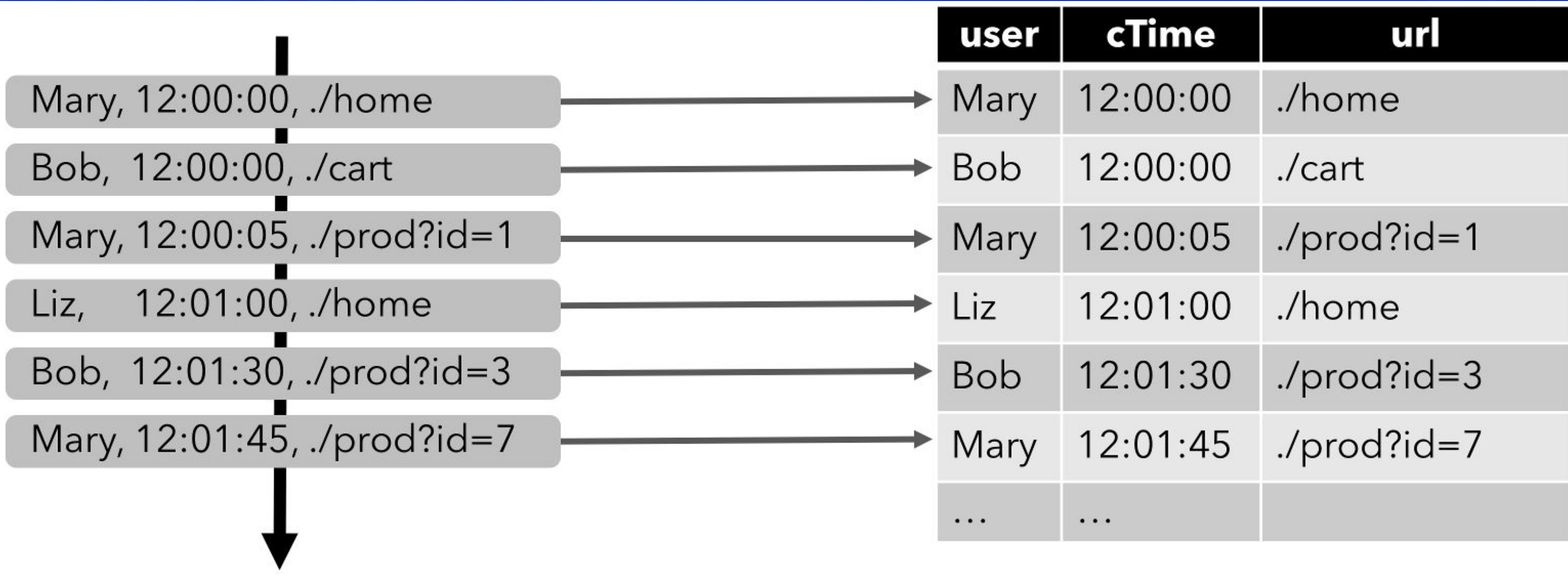
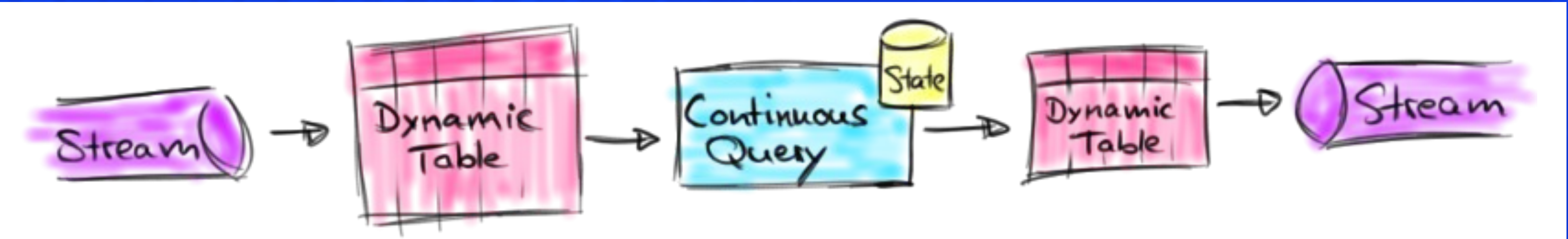


**stream processing & analytics**



**low-level stateful stream processing**

# Everything revolves around a Flink Dynamic Table



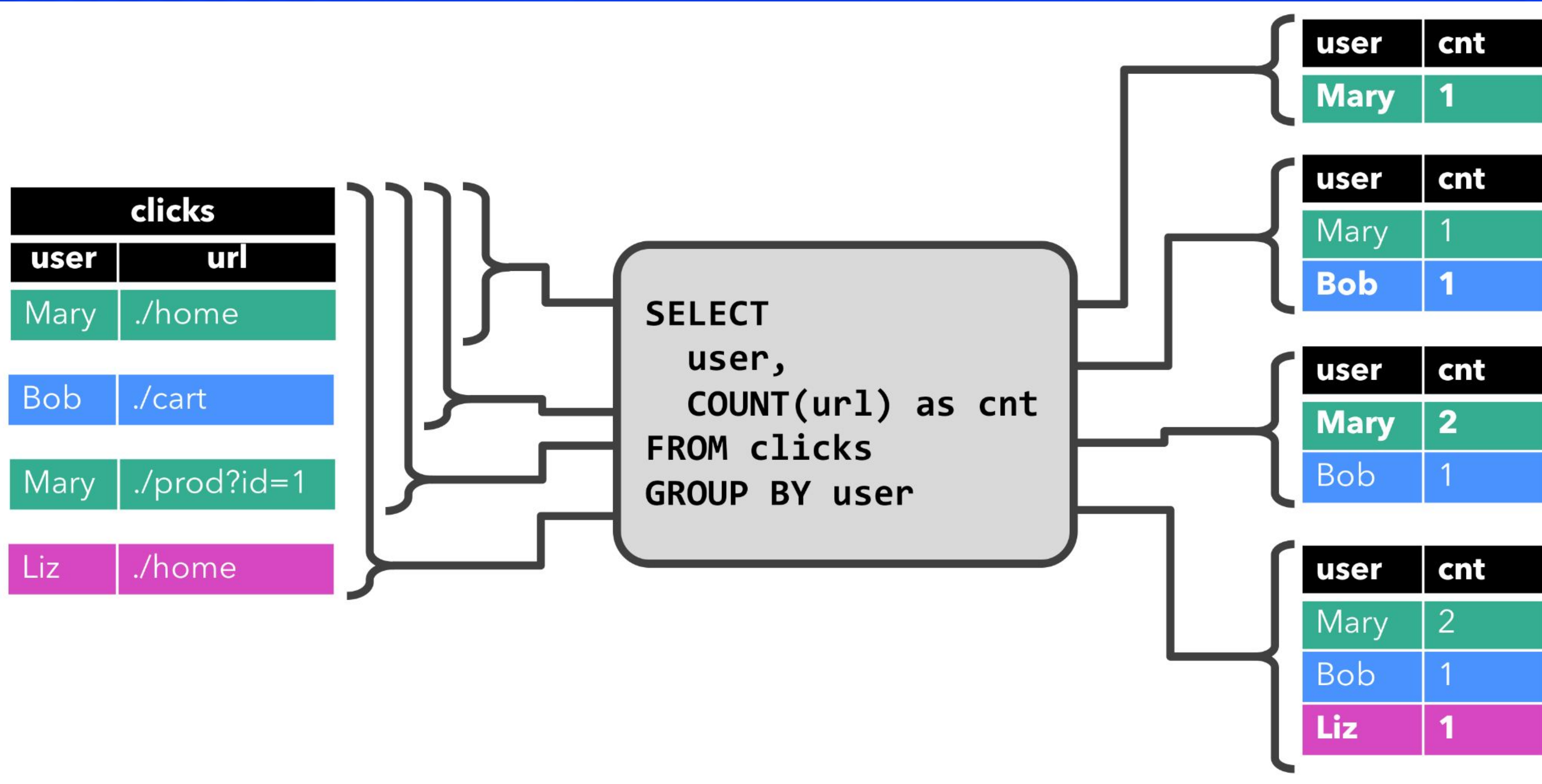


# What is a Flink Dynamic Table

- **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

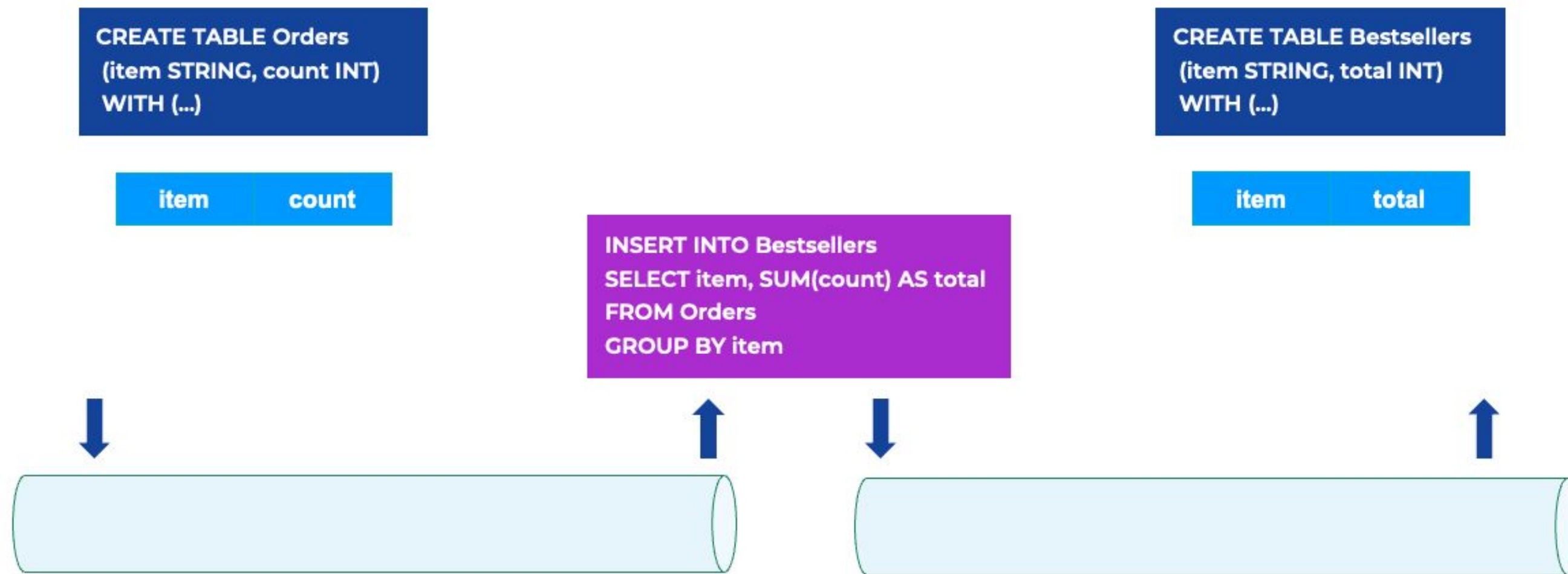




# Flink Kafka - Stream Table Duality



## Stream-Table duality – Append-only table



## Stream-Table duality – Updating table

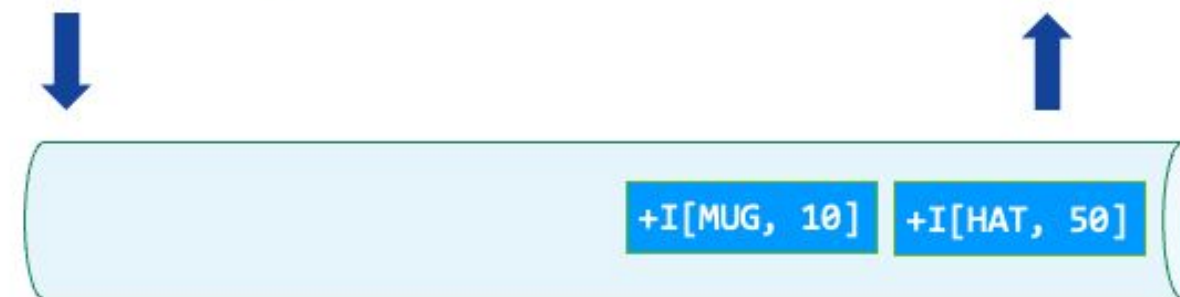
```
CREATE TABLE Orders  
(item STRING, count INT)  
WITH (...)
```

item	count
HAT	50
MUG	10

```
CREATE TABLE Bestsellers  
(item STRING, total INT)  
WITH (...)
```

item	total
HAT	50
MUG	10

```
INSERT INTO Bestsellers  
SELECT item, SUM(count) AS total  
FROM Orders  
GROUP BY item
```





# Flink Kafka - Stream Table Duality

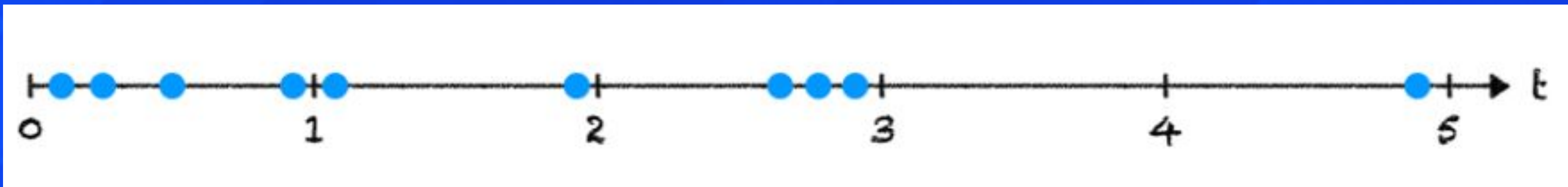


- **Append only stream - INSERT**
- **Retract Stream - INSERT:Add + DELETE:Retract + UPDATE:Retract**
- **Upsert Stream - UPSERT + DELETE**
  - **Main diff with Retract - changes are encoded with a single message and hence more efficient**
-

# Flink Window Functions

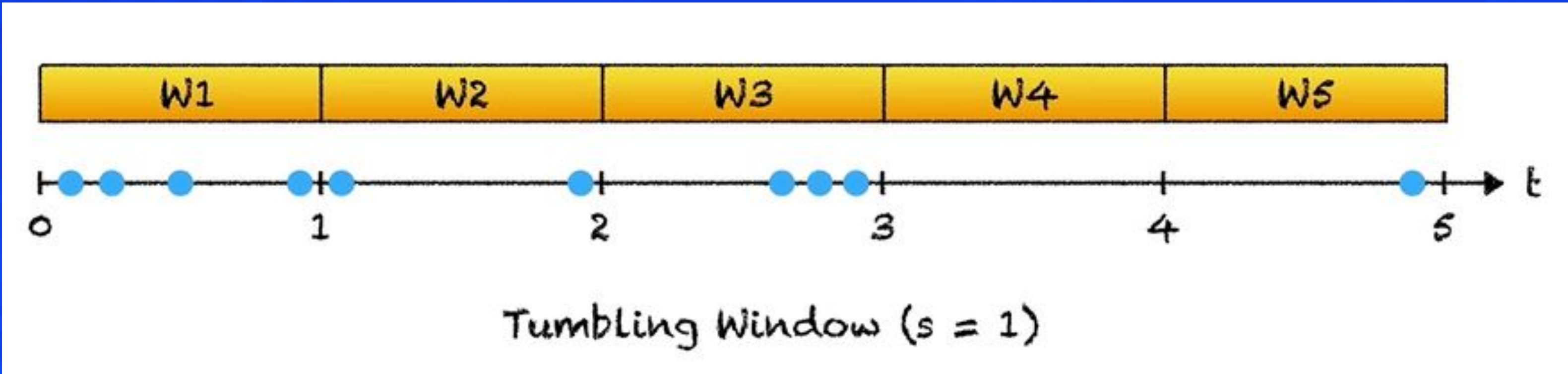


- Data events over time



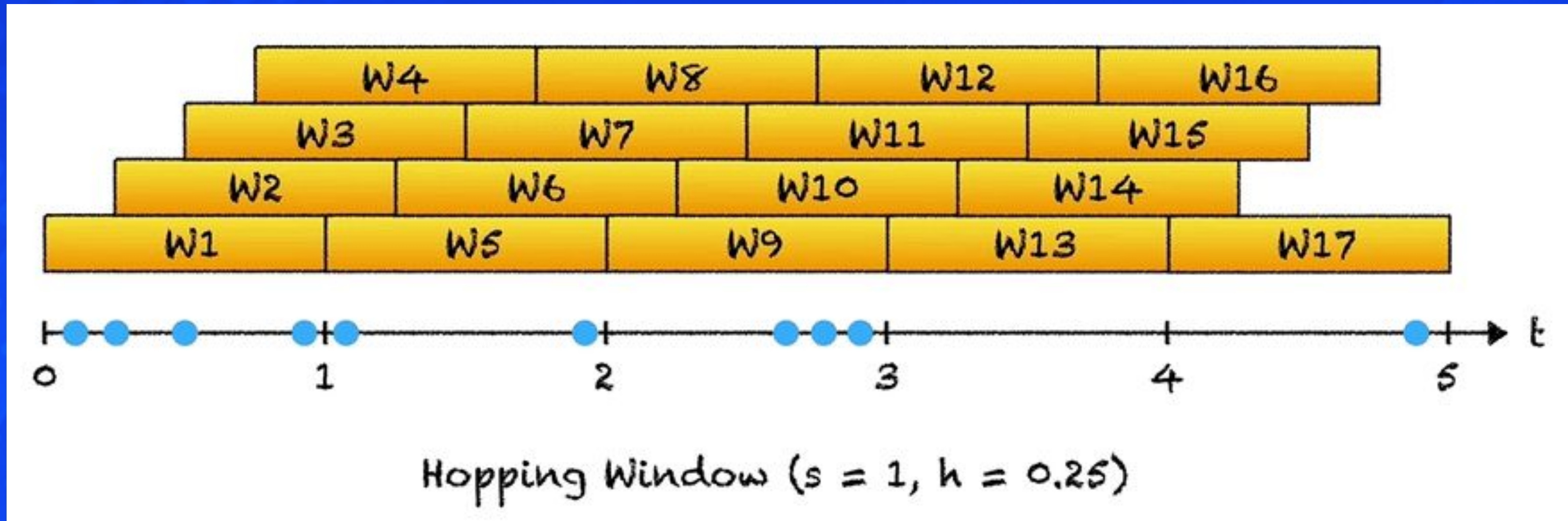


# Flink Tumbling Window





# Flink Hopping Window





# Flink Dynamic Table - Windowing Example



clicks		
user	cTime	url
Mary	12:00:00	./home
Bob	12:00:00	./cart
Mary	12:02:00	./prod?id=1
Mary	12:55:00	./prod?id=4
Bob	13:01:00	./prod?id=5
Liz	13:30:00	./home
Liz	13:59:00	./prod?id=7
Mary	14:00:00	./cart
Liz	14:02:00	./home
Bob	14:30:00	./prod?id=3
Bob	14:40:00	./home

```
SELECT
  user,
  TUMBLE_END(
    cTime,
    INTERVAL '1' HOURS)
  AS endT,
  COUNT(url) AS cnt
FROM clicks
GROUP BY
  user,
  TUMBLE(
    cTime,
    INTERVAL '1' HOURS)
```

result		
user	endT	cnt
Mary	13:00:00	3
Bob	13:00:00	1
Bob	14:00:00	1
Liz	14:00:00	2
Mary	15:00:00	1
Bob	15:00:00	2
Liz	15:00:00	1



# 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-3.1.0-1.18.0.jar")

    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



```
Files
master
Go to file
sensor-history
sensor-source
01_batch_csv_process.py
02_batch_csv_flinksql.py
04_kafka_flinksql_producer.py
05_kafka_flinksql_tumbling_wind...
06_kafka_pyflink_tableapi_tumbli...
09_pyflink_udf_tableapi.py
README.md
config.py
requirements.txt

Code Blame 73 lines (63 loc) · 2.49 KB
5 def main():
32     tenv.execute_sql(src_ddl)
33     #sensor_readings_tab = tenv.from_path('sensor_readings')
34
35     # Process a Tumbling Window Aggregate Calculation of Ampere-Hour
36     # For every 30 seconds non-overlapping window
37     # Calculate the total charge consumed grouped by device
38     tumbling_w_sql = """
39         SELECT
40             device_id,
41             TUMBLE_START(proctime, INTERVAL '30' SECONDS) AS window_start,
42             TUMBLE_END(proctime, INTERVAL '30' SECONDS) AS window_end,
43             SUM(ampere_hour) AS charge_consumed
44         FROM sensor_readings
45         GROUP BY
46             TUMBLE(proctime, INTERVAL '30' SECONDS),
47             device_id
48     """
49
50     tumbling_w = tenv.sql_query(tumbling_w_sql)
51
52     sink_ddl = """
53         CREATE TABLE devicecharge (
54             device_id VARCHAR,
55             window_start TIMESTAMP(3),
56             window_end TIMESTAMP(3),
57             charge_consumed DOUBLE
58         ) WITH (
59             'connector' = 'kafka',
60             'topic' = 'device.charge',
61             'properties.bootstrap.servers' = 'localhost:9098',
62             'scan.startup.mode' = 'earliest-offset',
63             'properties.auto.offset.reset' = 'earliest',
64             'format' = 'json'
65         )
66     """
67
68     tenv.execute_sql(sink_ddl)
69     tumbling_w.execute_insert('devicecharge').wait()
```





# Data Streaming Source



```
...
> bin
> lib
> share
.gitignore 11/11/24, 1:22 pm, 40 B
LICENSE.txt 11/11/24, 1:23 pm, 32.98 kB
pyenv.cfg 11/11/24, 1:22 pm, 395 B
config.py 28/06/24, 2:20 pm, 126 B 3 minutes ago
flink-sql-connector-kafka-3.2.0-1.18.jar 11/11/24, 1:25 pm, 5.6 MB
flink-sql-connector-kafka-3.3.0-1.20.jar 11/11/24, 1:10 pm, 5.59 MB
kafka_flinksqL_producer.py 28/06/24, 2:20 pm, 1.54 kB A minute ago
kafka_pyflink_tumbling_window_tableapi.py 11/11/24, 1:26 pm, 3.17 kB
> External Libraries
Scratches and Consoles

40     try:
41         while True:
42             device_data = sensor_event()
43             print(json.dumps(device_data))
44             producer.produce(topic=topic, key=device_data['device_id'],
45                               value=json.dumps(device_data),
46                               on_delivery=delivery_report)
47             time.sleep(5)
48         except Exception as e:
49             print(e)
50         finally:
51             producer.flush()

if __name__ == '__main__':
    try:
        while True:
```

Run kafka\_flinksqL\_producer x kafka\_pyflink\_tumbling\_window\_tableapi x

```
/Users/diptimanraichaudhuri/testing_space/pycharm_workspace/ossflink118/.venv/bin/python /Users/diptimanraichaudhuri/testing_space/pycharm_work
{"device_id": "b8:27:eb:bf:9d:51", "co": 0.0039, "humidity": 61.55, "motion": false, "temp": 32.09, "ampere_hour": 1.32, "ts": 1731313199569}
{"device_id": "00:0f:00:70:91:0a", "co": 0.0043, "humidity": 64.55, "motion": false, "temp": 25.67, "ampere_hour": 1.47, "ts": 1731313204573}
{"device_id": "b8:27:eb:bf:9d:51", "co": 0.0034, "humidity": 74.27, "motion": false, "temp": 32.76, "ampere_hour": 1.07, "ts": 1731313209578}
{"device_id": "00:0f:00:70:91:0a", "co": 0.003, "humidity": 73.82, "motion": false, "temp": 27.67, "ampere_hour": 0.66, "ts": 1731313214581}
{"device_id": "00:0f:00:70:91:0a", "co": 0.0039, "humidity": 77.06, "motion": false, "temp": 30.61, "ampere_hour": 0.98, "ts": 1731313219583}
{"device_id": "1c:bf:ce:15:ec:4d", "co": 0.0041, "humidity": 76.05, "motion": false, "temp": 32.55, "ampere_hour": 0.31, "ts": 1731313224589}
{"device_id": "1c:bf:ce:15:ec:4d", "co": 0.0026, "humidity": 49.53, "motion": true, "temp": 32.97, "ampere_hour": 0.87, "ts": 1731313229592}
{"device_id": "1c:bf:ce:15:ec:4d", "co": 0.0051, "humidity": 48.13, "motion": false, "temp": 33.5, "ampere_hour": 0.33, "ts": 1731313234593}
{"device_id": "1c:bf:ce:15:ec:4d", "co": 0.0046, "humidity": 61.86, "motion": false, "temp": 31.17, "ampere_hour": 0.17, "ts": 1731313239597}
{"device_id": "b8:27:eb:bf:9d:51", "co": 0.0053, "humidity": 65.58, "motion": true, "temp": 26.38, "ampere_hour": 1.71, "ts": 1731313244599}
{"device_id": "1c:bf:ce:15:ec:4d", "co": 0.005, "humidity": 48.29, "motion": true, "temp": 25.16, "ampere_hour": 1.12, "ts": 1731313249604}
```



# Pyflink Table API - Tumbling Window 30 sec interval



```
server-3-start sh
Terminal server-1-start.sh x server-2-start.sh x server-3-start.sh x console-consumer.sh x + v
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":6.5500000000000001}
{"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/>