Queries and Time

Flink SQL Training

https://github.com/ververica/sql-training

Temporal Conditions are Common in Stream Processing

- Smoothing and aggregating data based on time
  - *"Compute average over the last 1 minute"*
- Enriching streaming data with data from other sources
  - *"Join with most recent exchange rate"*
- Stream monitoring, pattern matching, and alerting
  - *"Emit an alert if 3 unsuccessful attempts occurred within 5 minutes"*
- Common types of data
  - User interactions: clicks, (mobile) apps
  - Logs: applications, machines, network hardware
  - Transactions: credit cards, ad serving
  - Sensors: mobile phones, cars, IoT

Characteristics of Queries with Temporal Conditions

- Input tables are append-only. Rows are not updated after they were inserted.
  - Table schema often defines an event-time (or processing-time) attribute
- The query consists of row-at-a-time and temporal operators
  - Filter, Projection, Windowed aggregations, Interval join, Temporal-table join, Pattern matching
  - "Traditional" stream processing operations
  - Temporal operators track time progress and compute final results
- Query output is append-only. Emitted result rows are never updated

Event & Processing Time Attributes

How Is Time Handled in Flink SQL?

- Flink SQL supports Event and Processing Time

- Tables may include *Time Attributes*
  - Time Attributes provide access to event or processing time
  - Time Attributes are (mostly) treated as regular attributes
  - Time Attributes have special type extended from SQL TIMESTAMP

- Time Attributes are declared with the table schema

Event Time Attribute

- Event time attribute is of type TIMESTAMP(3) and carries an actual timestamp
- Watermarks are meta-records and generated based on observed timestamps
  - Operators use watermarks to track the progress of time and reason about data completeness
- Event time attribute can be used like a regular TIMESTAMP attribute
  - Loses its event-time property when it is modified or when operators change the order of rows

```
CREATE TABLE clicks (
  user VARCHAR,
  url VARCHAR,
  cTime TIMESTAMP(3),
  WATERMARK FOR cTime AS cTime – INTERVAL ‘2’ MINUTE
)
```

Processing Time Attribute

- Processing time attributes are virtual and do not hold data
  - Local machine time is queried when the attribute is accessed

- A processing time attribute can be used like a regular TIMESTAMP
  - Loses its processing time property when it is modified

```
CREATE TABLE clicks (
  user VARCHAR,
  url VARCHAR,
  cTime AS PROCTIME() // is of type TIMESTAMP(3)
)
```

Temporal Operators

Temporal Operators

- Temporal operators associate records with each other based on temporal conditions
  - GROUP BY window aggregation
  - OVER window aggregation
  - Time-windowed join
  - Join with a temporal table (enrichment join)
  - Pattern matching (MATCH_RECOGNIZE)

  } Discussed in a later session

- Operators track progress in time to decide when input is complete
  - Operators emit final result rows that will not be updated
  - Operators discard state (records and results) that are no longer needed

- Temporal operators require time attributes in specific clauses

Temporal Aggregation

Temporal Aggregation

- Flink SQL supports two types of temporal aggregations
  - GROUP BY window aggregation
  - OVER window aggregation

- The clicks table is used for the following examples
  - cTime (clickTime) is an event time attribute.

user	cTime	url
Mary	12:00:00	./home
Bob	12:00:00	./cart
Mary	12:00:05	./prod?id=1
…	…	…

GROUP BY Window Aggregation

Compute number of clicks per hour and user

Time attribute

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

Time attribute

GROUP BY Window Aggregation

clicks

user	cTime	url
Mary	12:00:00	./home
Bob	12:00:00	./cart
Mary	12:02:00	./prod?id=2
Mary	12:55:00	./home
Bob	13:01:00	./prod?id=4
Liz	13:30:00	./cart
Liz	13:59:00	./home
Mary	14:00:00	./prod?id=1
Liz	14:02:00	./prod?id=8
Bob	14:30:00	./prod?id=7
Bob	14:40:00	./home

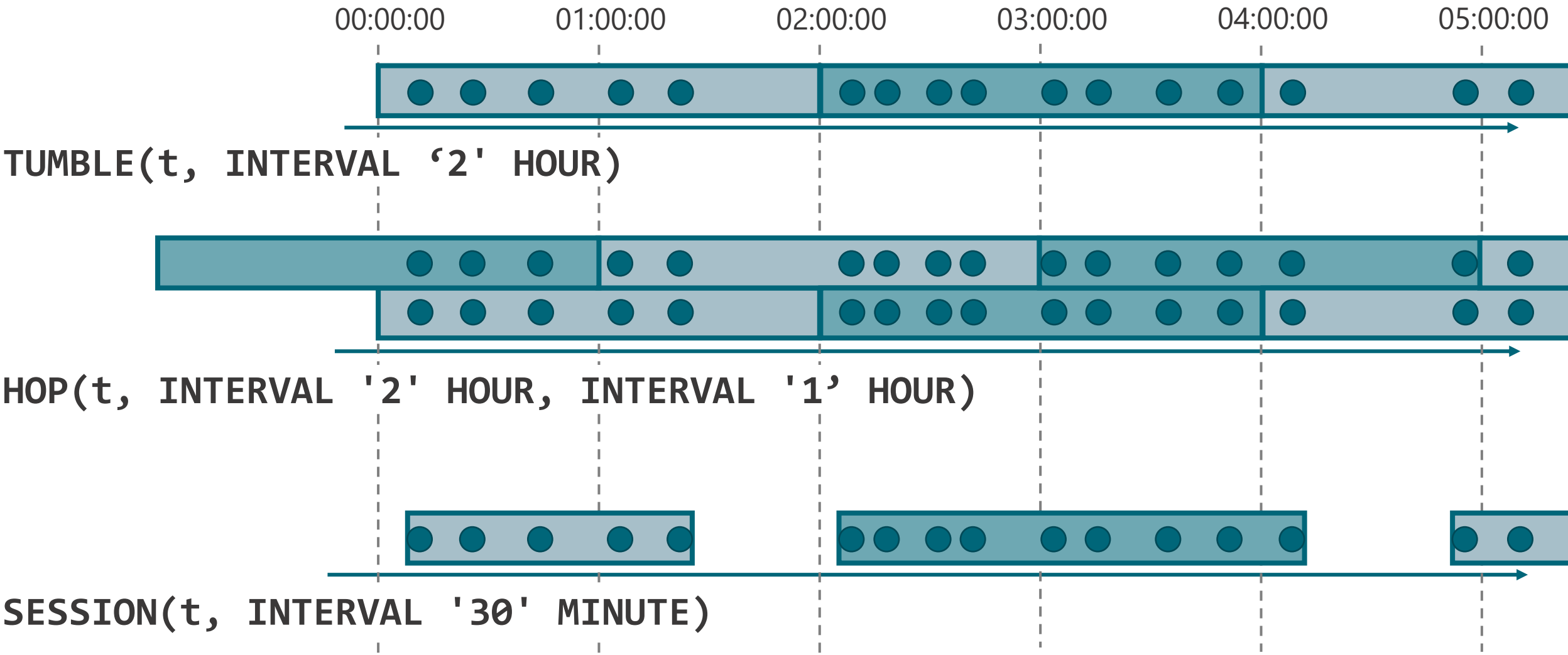
```
SELECT
 user,
 TUMBLE_END(
   cTime,
   INTERVAL '1' HOURS)
  AS endT,
  COUNT(url) AS cnt
FROM clicks
GROUP BY
 user,
 TUMBLE(
  cTime,
  INTERVAL '1' HOURS)
```

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

Rows are appended to result table.

GROUP BY Windows



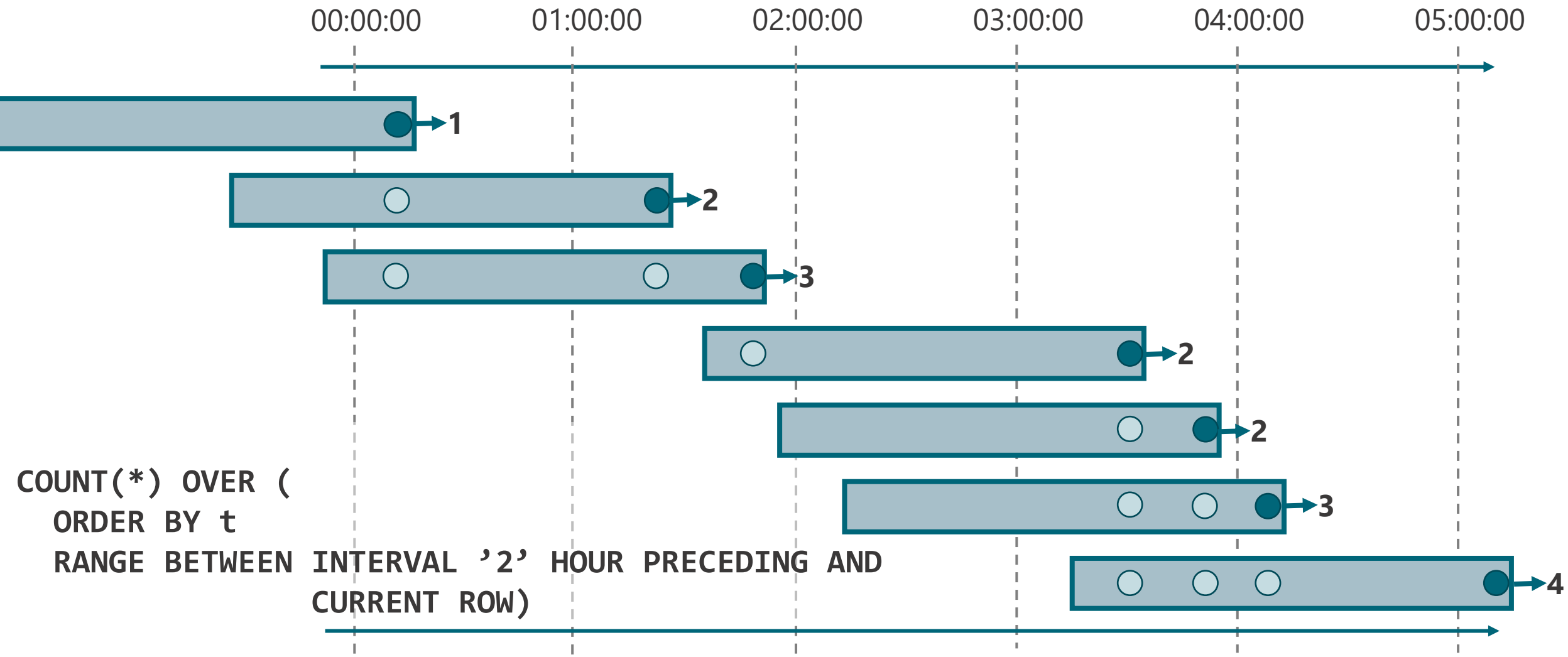
OVER Window Aggregation

Compute for each click
how often its URL was clicked in the previous 2 hours

```
SELECT
  user,
  url,
  COUNT(*) OVER w
FROM clicks
WINDOW w AS (
  PARTITION BY url
  ORDER BY cTime
  RANGE BETWEEN INTERVAL '2' HOUR PRECEDING AND CURRENT ROW)
```

Time attribute

OVER Windows




```
COUNT(*) OVER (
  ORDER BY t
  RANGE BETWEEN INTERVAL '2' HOUR PRECEDING AND
                CURRENT ROW)
```

Validity of Time Attributes

Validity of Event-Time Attributes

- Attributes lose their event-time property when watermark alignment cannot be guaranteed anymore
  - Event-time attributes become regular TIMESTAMPs
  - Cannot be used in temporal operations anymore

- Modifying a time attribute

```
SELECT FLOOR(cTime TO MINUTE) FROM clicks
```

- Operators that do not preserve the timestamp order in their output
  - Joins without a temporal join condition (discussed later in detail)
  - Non-windowed aggregations

```
SELECT cTime, COUNT(*) FROM clicks GROUP BY cTime
```

Validity of Processing-Time Attributes

- Attributes lose their processing-time property when they are modified
  - A processing-time attribute is materialized when it is accessed and becomes regular a TIMESTAMP
  - Cannot be used in temporal operations anymore

```
SELECT FLOOR(cTime TO MINUTE) FROM clicks
```

Summary

Summary

- Many streaming SQL use cases require temporal operators
  - Input and output are append-only tables
  - Queries use record-at-a-time and temporal operators
- Time attributes are defined in the table schema
  - Event-time and/or processing-time attributes are supported
- Temporal operations associate input rows based on their time attributes
  - Produce final results, i.e., emitted rows are never updated
  - Automatically manage their state

Hands On Exercises

Queries and Time

Continue with the hands-on exercises in
"Queries and Time"

https://github.com/ververica/sql-training/wiki/Queries-and-Time

We are here to help!

ververica

www.ververica.com

@VervericaData