Otto Friedrich University Bamberg

Data Streams and Complex Event Processing

Assignment - 01

Supervisor:
Prof. Dr. Daniela Nicklas
Mobile Software Systems

Submitted By :
Manasa Reddy Bandari - 1942692
Akshay Sharma - 1943984
Kummari Daniel Raj - 1973448

Nov 15, 2018

Contents

1 Exercise 1: Answer the following questions.

1.1 What is the difference between DSMS and DBMS?

Data stream management system (DSMS) is basically used to manage the continuous data streams whereas the Database management system (DBMS) is generally used for static data.
Below are some key differences in between two similar technologies.

DSMS	DBMS
DSMS is based on sequential access. (CQL)	DBMS is based on random access.(SQL)
It supports volatile data.	It supports non-volatile data.
Real time requirements.	Not specific to time requirements.
Update rate is pretty high in DSMS.	Update rate is low compared to DSMS
Supports continuous queries.	Supports one time queries.
Restricted on main memory.	Can have unlimited memory (Secondary Storage)

1.2 What are the steps that a query goes through before its deployment?

Generally below are the 5 steps which takes places when the query actually gets deployed.

- Query will be created according to the requirements.
- Logical query plan will be generated from the query.
- Check for the optimization of query plan for efficient use.
- Transform the logical plan into actual real plan.
- Execute the real or the phyiscal plan.

1.3 How are queries processed in DSMS different to those in DBMS?

As DSMS is entirely based on the stream data so it supports continuous query language (CQL) whereas DBMS has SQL has the main language.

DSMS has data based on time stamps but DBMS is not restriced to time. Different stream windows can be defined in DSMS in order to control the data flow whereas in DBMS , tables plays the role.

DSMS usually returns the unbounded data and DBMS is bounded by size of the tables.

When using JOIN, in DBMS we'll always get the accurate output but in DSMS , output will not be accurate but close to that.

1.4 Provide three use cases, where DSMS can be used; describe the data processed and the queries performed on them in each use case.

- **Satellite Network** : Monitoring of the weather with the help of the sensors. Use of multiple sources of streams to produce the whole layout and activities happening on earth from the universe perspective.
- **Google Maps** : Tracking of an individual to provide proper route from initial to final point. This is done taking the geo location coordinates of an individual at specified time intervals.
- **Network Traffic Management** : It tracks the data or the packets sent on a network and calculate the amount of traffice being generated or sent.

2 Exercise 2: Query design.

2.1 Query 1: Output a stream of readings for all books in the room number 036.

We will perform the JOIN operation between the readers stream and database. After that add the book information from the previous operation and then select room number 36.

```
SELECT bookID FROM (
SELECT * FROM Books AS BS, (
SELECT * FROM RFID_stream AS RS, Room_DB AS RDB
WHERE RS.Reader_id = RDB.Reader_id) AS B
WHERE BS.tag_ID = B.tag_ID) AS C
WHERE C.Room_number = 36;
```

2.2 Query 2: Get a stream of all books that left one room to another (taken by a person from one room to another) in the last 30 minutes.

We have to rename the attributes having the same name after joining the RFID with itself using 30 minutes range. After that you , we can select data with different timestamps.

```
SELECT tag_ID FROM (
SELECT A.Reader_ID AS Reader_ID_1, B.Reader_ID AS Reader_ID_2,
A.tag_ID, A.timestamp AS timestamp_1,B.timestamp AS
   timestamp_2 FROM
RFID_stream AS A [RANGE 30 MINUTES], RFID_stream AS B [RANGE
   30 MINUTES]
WHERE A.tag_ID = B.tag_ID)
WHERE Reader_ID_1 != Reader_ID_2 AND timestamp_1 !=
   timestamp_2;
```

2.3 Query 3: Get a stream of checked out books within the last 12 hours.

We only have to define the window of 12 hours for checked out books only.

```
SELECT bookID
FROM Books [RANGE 12 HOURS]
WHERE checked_out = true;
```

2.4 Query 4: Return a result stream of books that were taken off shelve to a reading room and back within the last 12 hours.

We have rename the attributes having same streams after joining RFID reader stream with itself using a 12 hours window. And then select data with same and different Reader_ID to make sure books were taken off the shelves respectively.

```
SELECT tag_ID FROM (
SELECT A.Reader_ID AS Reader_ID_1, B.Reader_ID AS Reader_ID_2,
A.tag_ID, A.timestamp AS timestamp_1, B.timestamp AS
   timestamp_2
FROM RFID_stream AS A [RANGE 12 HOURS],
RFID_stream AS B [RANGE 12 HOURS]
WHERE A.tag_ID = B.tag_ID)
WHERE timestamp_1 != timestamp_2 AND Reader_ID_1 = Reader_ID_1
   ;
```

2.5 Query 5: Return a result stream of books that were taken out of the library without being checked out.

Join operation needs to be implemented between books and RFID readers stream. After another join is there between result of the previous operation and the rooms database. Select the data where it satisfies the requirements.

```
SELECT bookID FROM (
SELECT * FROM Room_DB AS R, (
SELECT * FROM RFID_stream AS RS, Books AS B
WHERE RS.tag_ID = B.tag_ID) AS C
WHERE R.Reader_ID = C.Reader_ID)
WHERE Room_number = 2 AND checked_out = false;
```

3 Exercise 3: Develop queries.

3.1 Reports descriptions of auctions made in the last 10 minutes by luitpold Martucci.

```
SELECT x.id, x.name, y.description, y.itemname
FROM person [SIZE 10 MINUTES TIME] AS x,
auction [SIZE 10 MINUTES TIME] AS y
WHERE x.id = y.seller
AND x.name = 'Luitpold Martucci';
```

3.2 Find all items in the auctions stream that have an initial bid of more than 120 dollars.

```
SELECT x.itemname, x.seller,
x.initialbid FROM auction [UNBOUNDED] AS x
WHERE x.initialbid >= 120 ;
```

3.3 Return the number of bids received for each auction.

```
SELECT COUNT(bidder) AS bids_count, auction
FROM bid [UNBOUNDED]
GROUP BY auction;
```

3.4 Find auctions which had bids 2 minutes before their expiry date.

```
SELECT x.itemname, x.expires, y.datetime
FROM auction [UNBOUNDED] AS x, bid [UNBOUNDED] AS y
WHERE a.id = b.auction
AND (x.expires - y.datetime) <= 120000
```

3.5 Return the number of bids received for each auction coming from each state.

```
SELECT b.auction, p.state, COUNT(b.bidder)
FROM bid [UNBOUNDED] AS b , person [UNBOUNDED] AS p
WHERE b.bidder = p.id
GROUP BY b.auction, p.state ;
```