H2 Database Implementation : Assignment 3

Implementing Spread Aggregate Function

Kavya Kotian
Siddarth Sargunaraj
Neelabh Kumar
Prashanth Kumar Purshotam

June 26, 2020

Introduction

In this assignment we are trying to implement *Spread*, an aggregate function which returns the range in which the values of the column lie. This takes inspiration from our NoSql database *InfluxDB* which is a time series database and helps figure out the maximum variation of the values in the given column. *Spread* is a method to calculate difference between the *Maximum element* and *Minimum element* in the specified column.

Implementation

To implement a new Aggregate function *Spread*, we need to implement the interface *Aggregate Function* that is provided by *h2 database*.We have created an implementation of the *Aggregate Function* called *CustomAggregateFunction*. Our implementation overides all methods in the *Aggregate function* like :

- default void init(Connection conn) throws SQLException
- int getType(int[] inputTypes) throws SQLException;
- void add(Object value) throws SQLException;
- Object getResult() throws SQLException

CustomAggregateFunction

Our Custom Aggregate Function implements the *Spread* function which gives the difference between the min and max element in the column. It implements *AggregateFunction* interface from *org.h2.api* package. This package gives us interfaces to build Custom functions.
The following methods are implemented in our code:

Global variables

- min : Records the minimum element of the column
- max : Records the minimum element of the column
- count : Is 0 when empty and 1 otherwise.

default void init(Connection conn)

Creates a new object of the CustomAggregateFunction class with every invocation. Parameter *Connection* helps establish connection with the database.

int getType(int[] inputTypes)

This method returns the type of the data as an integer *id* value corresponding to the SQL type.*inputTypes* is an array of *ids* corresponding to the SQL type of all the columns i.e the given data used to aggregate.
Spread function handles one column at a time, hence we ensure that the size of *inputTypes* array is 1. If the condition is satisfied, we return the integer *id* of the column.

void add(Object value)

This method helps iterate over the entire column. This method is called once for each row i.e internally we are iterating over the column and passing one value at a time. This method helps us to set *min*, *max* and *count* variables.
When the method is called and *count* is 0 , and no value has been iterated over yet and we are at the first row, the method sets the *min* and *max* values to the value of the first iteration, and increments count by 1.
When the *count* is not equal to 0, we try to update our min and max values in every iteration.

Object getResult()

Gives us the result of our function.

$Spread = max - min$

Queries using new aggregate function SPREAD

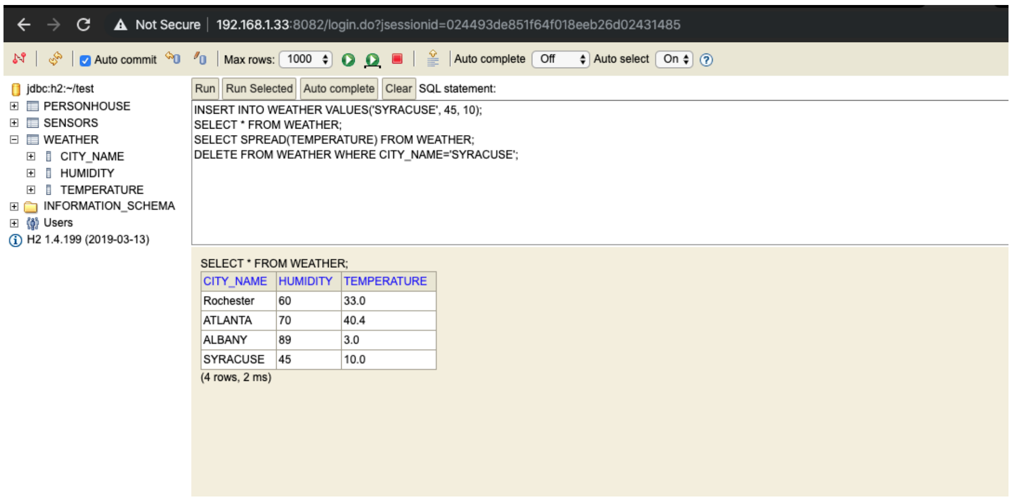
To demonstrate our new aggregate function we first create a table and insert some values. In our example we create a WEATHER table.

```
CREATE TABLE WEATHER (
    CITY_NAME VARCHAR(25),
    HUMIDITY INT,
    TEMPERATURE FLOAT
    );
```

Example 1:

Let us insert a few values into the table. For example,

```
INSERT INTO WEATHER VALUES('SYRACUSE', 45, 10);
```



```
INSERT INTO WEATHER VALUES('SYRACUSE', 45, 10);
SELECT * FROM WEATHER;
SELECT SPREAD(TEMPERATURE) FROM WEATHER;
DELETE FROM WEATHER WHERE CITY_NAME='SYRACUSE';
```

SELECT * FROM WEATHER;

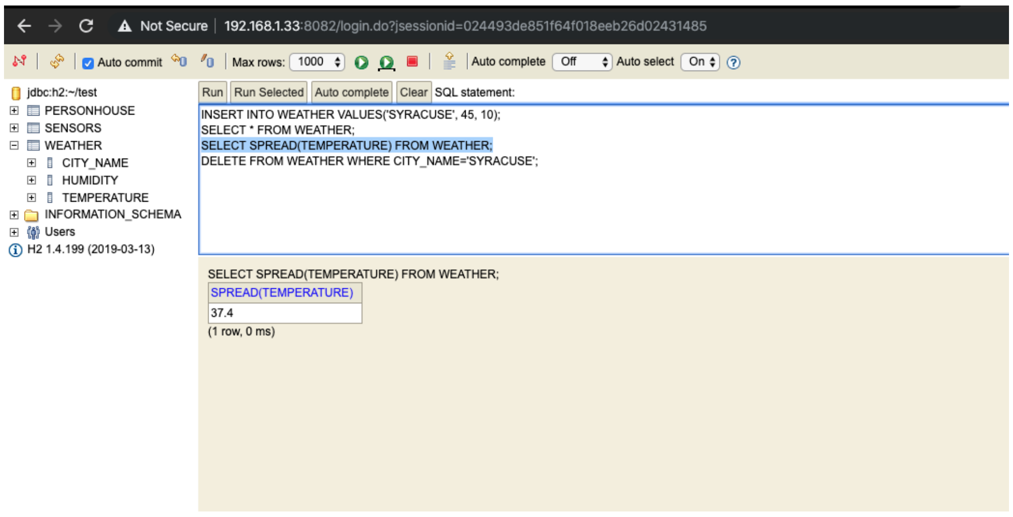
CITY_NAME	HUMIDITY	TEMPERATURE
Rochester	60	33.0
ATLANTA	70	40.4
ALBANY	89	3.0
SYRACUSE	45	10.0

(4 rows, 2 ms)

Table after creation and insertion of values

We then find the SPREAD of the temperature using the following statement

```
SELECT SPREAD(TEMPERATURE) FROM WEATHER
```



```
INSERT INTO WEATHER VALUES('SYRACUSE', 45, 10);
SELECT * FROM WEATHER;
SELECT SPREAD(TEMPERATURE) FROM WEATHER;
DELETE FROM WEATHER WHERE CITY_NAME='SYRACUSE';
```

```
SELECT SPREAD(TEMPERATURE) FROM WEATHER;
SPREAD(TEMPERATURE)
37.4
(1 row, 0 ms)
```

SPREAD of the temperature

The spread is calculated by subtracting the minimum value from the maximum value. From the table, we see that:

Maximum value = 40.4

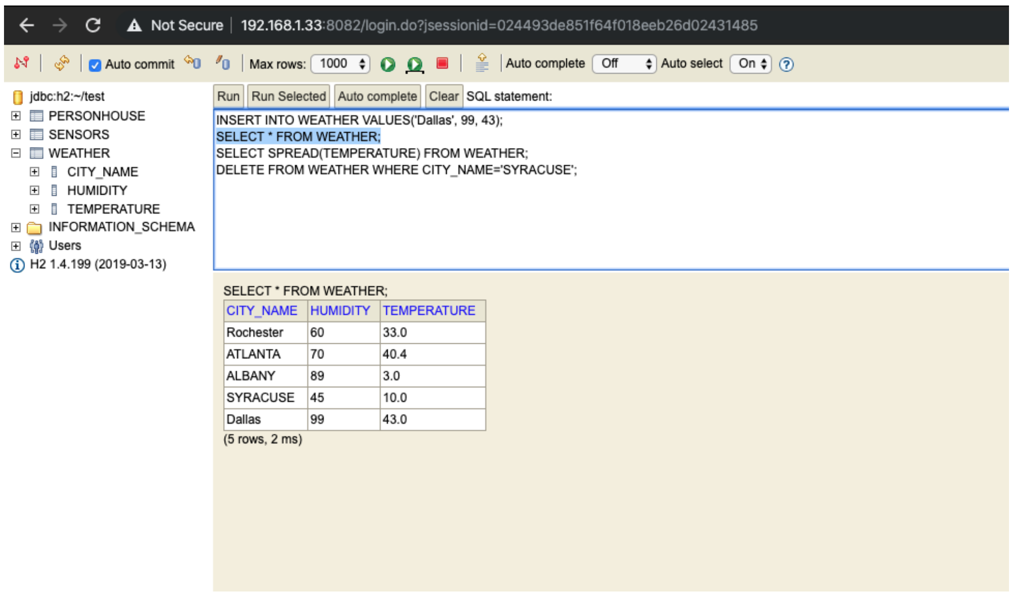
Minimum value = 3.0

∴ Spread = 40.4 - 3.0 = 37.4

Example 2:

Let us insert a few values into the table. For example,

```
INSERT INTO WEATHER VALUES('Dallas', 99, 43);
```



```
INSERT INTO WEATHER VALUES('Dallas', 99, 43);
SELECT * FROM WEATHER;
SELECT SPREAD(TEMPERATURE) FROM WEATHER;
DELETE FROM WEATHER WHERE CITY_NAME='SYRACUSE';
```

SELECT * FROM WEATHER;

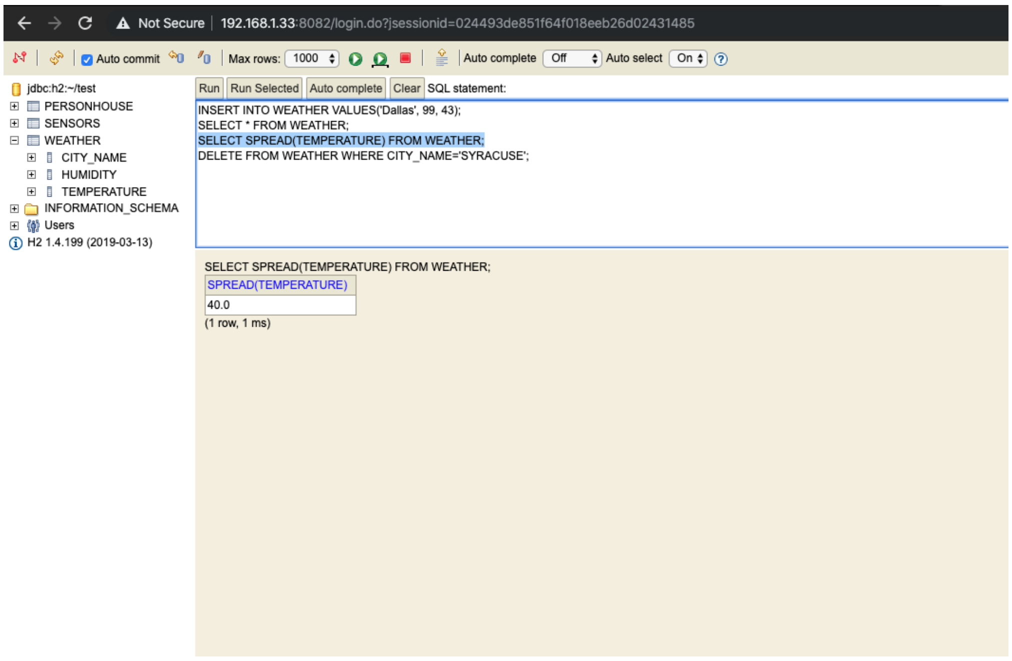
CITY_NAME	HUMIDITY	TEMPERATURE
Rochester	60	33.0
ATLANTA	70	40.4
ALBANY	89	3.0
SYRACUSE	45	10.0
Dallas	99	43.0

(5 rows, 2 ms)

Table after creation and insertion of values

We then find the SPREAD of the temperature using the following statement

```
SELECT SPREAD(TEMPERATURE) FROM WEATHER
```



```
INSERT INTO WEATHER VALUES('Dallas', 99, 43);
SELECT * FROM WEATHER;
SELECT SPREAD(TEMPERATURE) FROM WEATHER;
DELETE FROM WEATHER WHERE CITY_NAME='SYRACUSE';
```

```
SELECT SPREAD(TEMPERATURE) FROM WEATHER;
SPREAD(TEMPERATURE)
40.0
(1 row, 1 ms)
```

SPREAD of the temperature

The spread is calculated by subtracting the minimum value from the maximum value. From the table, we see that:

Maximum value = 43.0

Minimum value = 3.0

∴ Spread = 43.0 - 3.0 = 40.0