Design For SELECT, FILTER, and Order Query

- for ComScore Code Challenge Section 2-1

Author: Ran Zhao

Email: kevin2robinli@yahoo.com

Contents

1. Summary

1.1 Development Environment and input/output

The design is to use Java programming language to implement the requirement. The development jdk version **jdk1.8.0_0131**.

The IDE is Eclipse. Version control tool is Git and code repository is Github.

The input is an args command. For example, $s TITLE,REV,DATE $f DATE=2014-04-01.

The output is ".txt" file which contains the queried information. Naming for the output file is {query_type}_{dateMillisecond}.txt
The output path is resource/query_out/{inputSampleName}/…

1.2 Design

The query begins from a args command input. Firstly, the command needs to be parsed. The key word "s", "f", "o" will be extracted, also the queried column, ordered column, filter criteria will parsed.

Based on the parsing result, a service invoker is called to invoke SELECT, FILTER, ORDER query service to get the corresponding output.

For each type of query like SELECT, FILTER, ORDER, different algorithms are applied to process the command and generate data.

After the output is generated, service invoker calls the OUTPUT service to write the output into the query output folder

2. Command

2.1 Command Types

There are five types of command the tool is designed to support

2.1.1 SELECT

This type of command only has “SELECT” query. It can get some or all columns from the data store. The command format like:

```
$s TITLE,REV,DATE
```

“$s” is the key word stands for “SELECT”. The selected Columns are after it.

2.1.2 FILTER

This type of command has “SELECT” and “FILTER”query. It first queries the column from data store, then apply the filter criteria to refine the queried result. Format like:

```
$s TITLE,REV,DATE $f DATE=2014-04-01"
```

“$s” is the key word stands for “SELECT”. The selected Columns are after it. “$f” is the key word stands for “FILTER”, the criteria after it will be the condition to re-catch the result from the selected result.

2.1.3 ORDER BY DATE

This type of command has “SELECT” and “ORDER”query. It first queries the column from data store, then order the select query by Date ascendant, format like:

```
$s TITLE,REV,DATE $o DATE
```

“$s” is the key word stands for “SELECT”. The selected Columns are after it. “$o” is the key word stands for “ORDER”, the column DATE after it make the selected query order by time.

2.1.4 ORDER BY NAME

This type of command has “SELECT” and “ORDER”query. It first queries the column from data store, then order the select query by Name ascendant, format like:

```
$s TITLE,REV,DATE $o TITLE
```

“$s” is the key word stands for “SELECT”. The selected Columns are after it. “$o” is the key word stands for “ORDER”, the column TITLE after it make the selected query order by title name ascendant.

2.1.5 ORDER BY DATE,NAME

This type of command has “SELECT” and “ORDER”query. It queries the column from data store, then order the select query by DATE first, then by Name ascendant, format like:

$s TITLE,REV,DATE $o DATE, TITLE

“$s” is the key word stands for “SELECT”. The selected Columns are after it. “$o” is the key word stands for “ORDER”, the column DATE,TITLE after it make the selected query order by time first, then the time ordered result will be ordered by title name ascendant if data DATE column has same value.

For example, below records

Star trek|warner bros|2014-04-02|
Batman |warner bros|2014-04-03|
Iron man|warner bros|2014-04-02|

first ordered by Date

Star trek|warner bros|2014-04-02|
Iron man|warner bros|2014-04-02|
Batman |warner bros|2014-04-03|

And for 1st and 2nd row, they have the same date, so order them by TITLE again, result as :

Iron man|warner bros|2014-04-02|
Star trek|warner bros|2014-04-02|
Batman |warner bros|2014-04-03|

2.2 Command Parsing

The design is parsing command into two parts : type and column/condition

The type includes “$s”, “$f”, “o”

The column/condition is to store the operation action. For example,

For “$s”, “o” columns DATE, TITLE are put together as second part. For “f”, DATE=2014-12-10 is put as second part.

The data structure to store the parsing result is HASHMAP. Key of HASHMAP is the type. The VALUE is the second part of command.

3. Query

3.1 SELELCT

For SELECT only query, the design as below:

- Read from data store files based on the queried columns. For each column, data is stored in List<String>

- Get List<string> for all columns, get the same-position element in each list to consist a row. Rows together is the output.

3.2 FILTER

First it does the same thing in 3.1 SELECT, after that, apply the filter criteria in each row, if the value in the specific column matches the criteria, this row will be selected.

When the criteria is applied on all rows, all selected rows it the output.

3.3 Order by DATE

First it does the same thing in 3.1 SELECT, after that, steps are:

- Get DATE from all rows.
- Convert the DATE data from String -> Date
- Create a HASHMAP, key is the Integer index, value is the Date from last step.
- Sort the HASHMAP by it's value date, get the Keyset of HASHMAP. The Keyset is the list of index.
- Apply the index to re-order the rows in all SELECTED result.
- Get the output

3.4 Order by NAME

First it does the same thing in 3.1 SELECT, after that, steps are:

- Get NAME COLUMN from all rows.
- Create a HASHMAP, key is the Integer index, value is the Name Column value from last step.

- Sort the HASHMAP by it's value date, get the Keyset of HASHMAP. The Keyset is the list of index.
- Apply the index to re-order the rows in all SELECTED result.
- Get the output

3.5 Order by DATE, NAME

First it does the same thing in 3.3 ORDER By DATE to get the time-ordered result, after that, steps are:

- Create a new output.
- Loop the time-ordered result, get all rows that next row's date is same as currently row's date
    - If next row's date is different from current row's date. Insert current row into new output. The next row becomes the current row, and loop go forward
    - If next row's date is same as current row. Loop go forward.
- For all rows we get from last step, use HashMap<index, columnToBeOrderd> to keep temp data. Key is the Integer index, value is the date in all rows we just got.
- Sort the HASHMAP by it's value date, get the Keyset of HASHMAP. The Keyset is the list of index.
- Use the index to get all rows from time-ordered result. Insert these rows into new output.
- Repeat the loop starts in step 2 until it reaches the end of time-ordered result.
- Get the output

4. Workflow

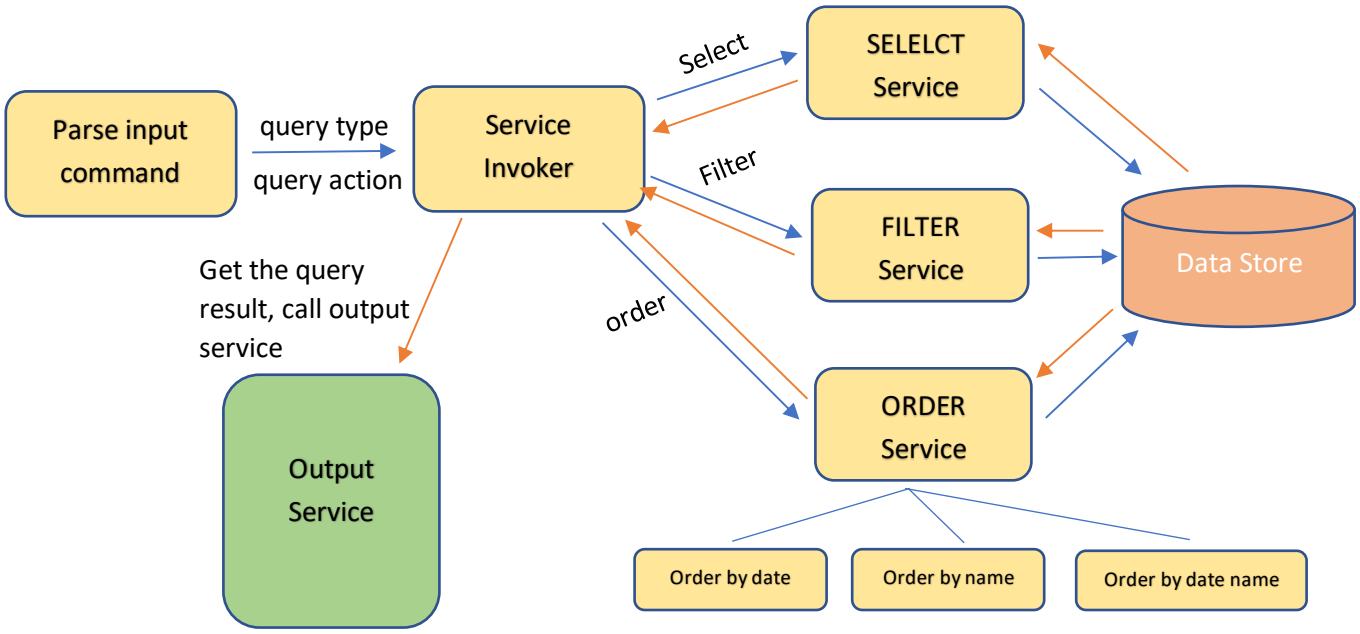
The whole workflow is:

(1) Parse the command, based on different command type, call the query service invoker to invoke SELECT, FILTER, or ORDER service.

(2) For each command, the action is different such as selecting columns, filter by criteria, order by date, order by name, order by date name. The action is in the second part in command parsing result. Use different service implementation to execute action.

(3) After query service complete processing command, invoke OUTPUT service to generate output.

(4) OUTPUT service use Java I/O, BufferWritter to create output file.


Parse input command
query type
query action
Service Invoker
Select
SELELCT Service
Filter
FILTER Service
Data Store
order
ORDER Service
Get the query result, call output service
Output Service
Order by date
Order by name
Order by date name