LAPORAN

RENCANA TUGAS MANDIRI (RTM) Ke-4

MATA KULIAH BIG DATA KELAS B

"QUERY STATISTIK DESKRIPTIF MENGGUNAKAN HIVE & QUERY"

DOSEN PENGAMPU

Tresna Maulana Fahrudin S.ST., M.T. (NIP. 199305012022031007)

NAMA PENYUSUN

Fadlila Agustina (21083010050)

UNIVERSITAS PEMBANGUNAN NASIONAL

"VETERAN" JAWA TIMUR

TAHUN 2023

SOAL

1. Silakan lakukan analisis pada dataset NOAA menggunakan Hive untuk menjawab pertanyaan:
   a. Statistika deskriptif (suhu maksimum, minimum, rata-rata, varian, deviasi standar, dan persentil) yang dikelompokkkan berdasarkan masing-masing tahun.
   b. Persentase perubahan rata-rata suhu di antara 2 tahun, misalnya antara tahun 1902-1903
   c. Selanjutnya, buatlah 3 pertanyaan tambahan analisis berdasarkan dataset NOAA tersebut (3 kolom) dan jawablah menggunakan sintaks query serta tampilkan hasilnya

2. Unduh dataset dummy - Saham dan Harga Sembako (kerjakan keduanya) yang dapat diakses di https://drive.google.com/drive/folders/182b5TikHcqCe2vAzgfabaNNjAfG5Qh6s?usp=sharing. Lalu analisislah menggunakan bentuk-bentuk Xquery transformation yang sesuai

JAWABAN

1. Analisis pada dataset NOAA menggunakan Hive
   a. Statistika deskriptif
   - Maximum

     Input:

```
SELECT tahun, MAX(suhu) AS suhu_max
FROM suhutemp
GROUP BY tahun;
```

     Query di atas mengambil data suhu maksimum dari tiap tahun pada tabel **suhutemp** dan mengelompokannya berdasarkan tahun.

     Output:

```
OK
1901    999
1902    328
1903    999
1904    294
1905    328
1906    294
1907    999
1908    378
1909    999
1910    999
1911    999
1912    411
1913    999
1914    999
1915    999
1916    289
1917    478
1918    999
1919    999
1920    344
1921    999
1922    999
1923    394
1924    456
1925    378
1926    999
1927    999
1928    999
1929    999
1930    999
1931    999
1932    999
Time taken: 38.949 seconds, Fetched: 32 row(s)
```

   - Minimum

     Input:

```
SELECT tahun, MIN(suhu) AS suhu_min
FROM suhutemp
GROUP BY tahun;
```

     Query di atas mengambil data suhu minimum dari tiap tahun pada tabel **suhutemp** dan mengelompokannya berdasarkan tahun.

     Output:

```
OK
1901    0
1902    0
1903    0
1904    0
1905    0
1906    0
1907    0
1908    0
1909    0
1910    0
1911    0
1912    0
1913    0
1914    0
1915    0
1916    0
1917    0
1918    0
1919    0
1920    0
1921    0
1922    0
1923    0
1924    0
1925    0
1926    0
1927    0
1928    0
1929    0
1930    0
1931    0
1932    0
Time taken: 47.221 seconds, Fetched: 32 row(s)
```

- Rata-rata

Input:

```
SELECT tahun, AVG(suhu) AS rata_rata_suhu
FROM suhutemp
GROUP BY tahun;
```

Query di atas mengambil data rata-rata suhu dari tiap tahun pada tabel **suhutemp** dan mengelompokannya berdasarkan tahun.

Output:

```
OK
1901    93.8994668697639
1902    73.78385376999239
1903    78.52105584375954
1904    74.4364934670313
1905    75.15333028501753
1906    83.6172816952868
1907    89.30807248764415
1908    84.8540622627183
1909    91.83727380795139
1910    78.11803921568628
1911    86.5815352425788
1912    92.43987506507028
1913    85.17100753941055
1914    80.78346861781182
1915    91.46176705909247
1916    62.619004111466424
1917    94.20697541452259
1918    86.09712837837837
1919    87.77162268095114
1920    78.20605002908668
1921    84.81994269340974
1922    89.40480274442538
1923    80.92046493404727
```

```
1924    89.57061923583663
1925    87.4282168517309
1926    100.65950269853508
1927    111.78934446354039
1928    117.86021212945336
1929    131.9264331407987
1930    147.69661222020568
1931    160.65097112536066
1932    167.94986493146183
Time taken: 42.104 seconds, Fetched: 32 row(s)
```

- Varian

Input:

```
SELECT tahun, VARIANCE(suhu) AS varian_suhu
FROM suhutemp
GROUP BY tahun;
```

Query di atas mengambil data variansi suhu dari tiap tahun pada tabel **suhutemp** dan mengelompokannya berdasarkan tahun.

Output:

```
OK
1901    4763.648994384471
1902    2952.782069839996
1903    9037.802806575155
1904    2813.311904325309
1905    3649.533264705588
1906    4165.583705727371
1907    4402.526544756256
1908    4209.395103088001
1909    7828.16583295886
1910    4405.390903344865
1911    6511.908057117325
1912    5485.1690815044385
1913    5929.42917542993
1914    4895.375316058272
1915    4994.930673343384
1916    2821.9215392548913
1917    5113.103187478508
1918    6278.702559321321
1919    4649.827645211168
1920    3998.41932349021
1921    4025.389585096988
1922    9609.395425761575
1923    3350.742139629734
1924    4091.520111737748
1925    4591.046414782489
1926    13678.42964755529
1927    24760.95881599549
1928    32968.09223525558
1929    28532.437138069305
1930    35426.33148434418
1931    44478.2234253197
1932    59660.99342985473
Time taken: 42.151 seconds, Fetched: 32 row(s)
```

- **Deviasi Standar**

Input:

```
SELECT tahun, STDDEV(suhu) AS deviasi_standar_suhu
FROM suhutemp
GROUP BY tahun;
```

Query di atas mengambil data deviasi standar suhu dari tiap tahun pada tabel **suhutemp** dan mengelompokannya berdasarkan tahun.

Output:

```
OK
1901    69.01919294214089
1902    54.339507449368696
1903    95.06735931209595
1904    53.040662744024125
1905    60.41136701569985
1906    64.54133331228424
1907    66.35153762164262
1908    64.87985128749912
1909    88.47692260108768
1910    66.37311883093084
1911    80.69639432538064
1912    74.06192734127595
1913    77.00278680301078
1914    69.96695874524112
1915    70.67482347585585
1916    53.12176144721569
1917    71.50596609709226
1918    79.23826448958432
1919    68.1896447065914
1920    63.233055623544004
1921    63.445957988645645
1922    98.02752381735232
1923    57.8855952688554
1924    63.96499129787909
1925    67.75726097461798
1926    116.95481882998789
1927    157.35615277451177
1928    181.57117677444177
1929    168.9154733530037
1930    188.2188393449077
1931    210.89860934894688
1932    244.25599978271717
Time taken: 49.194 seconds, Fetched: 32 row(s)
```

- **Persentil**

Input:

```
SELECT tahun, PERCENTILE(suhu, 0.75) AS persentil_75
FROM suhutemp
GROUP BY tahun;
```

Query di atas mengambil data persentil ke-75 suhu dari tiap tahun pada tabel **suhutemp** dan mengelompokannya berdasarkan tahun.

Output:

```
OK
1901    144.0
1902    111.0
1903    122.0
1904    117.0
1905    122.0
1906    128.0
1907    133.0
1908    128.0
1909    133.0
1910    122.0
1911    128.0
1912    144.0
1913    128.0
1914    117.0
1915    133.0
1916    89.0
1917    144.0
1918    122.0
1919    128.0
1920    122.0
1921    128.0
1922    128.0
1923    122.0
```

```
1924    133.0
1925    133.0
1926    139.0
1927    128.0
1928    128.0
1929    144.0
1930    156.0
1931    178.0
1932    161.0
Time taken: 43.242 seconds, Fetched: 32 row(s)
```

b. Persentase perubahan rata-rata suhu di antara 2 tahun

Input:

```
SELECT ((rata_rata_suhu_tahun_2 - rata_rata_suhu_tahun_1) /
rata_rata_suhu_tahun_1) * 100 AS persentase_perubahan
FROM
    (SELECT AVG(suhu) AS rata_rata_suhu_tahun_1
     FROM suhutemp
     WHERE tahun = 1926) t1,
    (SELECT AVG(suhu) AS rata_rata_suhu_tahun_2
     FROM suhutemp
     WHERE tahun = 1927) t2;
```

Query di atas menghitung persentase perubahan rata-rata suhu antara tahun 1926 dan 1927 pada tabel **suhutemp** dengan mengambil rata-rata suhu tahun 1 dan tahun 2, kemudian dihitung selisihnya, dibagi dengan rata-rata suhu tahun 1, dan dikalikan 100.

Output:

```
OK
11.056921072159524
Time taken: 135.576 seconds, Fetched: 1 row(s)
```

c. 3 pertanyaan tambahan analisis

- Berapa jumlah rata-rata suhu tertinggi dalam tiap dekade mulai dari tahun 1910 – 1930?

  Input:

  ```
  SELECT CONCAT_WS('-', CAST(FLOOR(tahun/10)*10 AS STRING),
  CAST(FLOOR(tahun/10)*10 + 9 AS STRING)) AS dekade, MAX(rata_rata_suhu) AS
  max_rata_rata_suhu
  FROM (
    SELECT tahun, AVG(suhu) AS rata_rata_suhu
    FROM suhutemp
    WHERE tahun BETWEEN 1910 AND 1930
    GROUP BY tahun
  ) AS suhu_per_tahun
  GROUP BY FLOOR(tahun/10)
  ORDER BY dekade ASC;
  ```

  Query di atas mengambil data maksimum rata-rata suhu dari tiap dekade antara tahun 1910 sampai 1930 pada tabel **suhutemp** dengan mengelompokkan rata-rata suhu tiap tahun dalam rentang waktu dekade, dan menyusunnya berdasarkan dekade secara urut dan naik.

  Output:

  ```
  OK
  1910-1919       94.20697541452259
  1920-1929       131.9264331407987
  1930-1939       147.69661222020568
  Time taken: 130.481 seconds, Fetched: 3 row(s)
  ```

- Berapa jumlah hari dengan suhu di atas 30 derajat celcius pada tahun 1915?

  Input:

```
SELECT COUNT(*) AS jumlah_hari
FROM suhutemp
WHERE tahun = 1915 AND suhu > 30;
```

  Query di atas mengambil data jumlah hari pada tabel **suhutemp** yang suhu hariannya di atas 30 derajat celcius pada tahun 1915.

  Output:

```
OK
6648
Time taken: 45.913 seconds, Fetched: 1 row(s)
```

- Berapa jumlah hari dengan perubahan suhu lebih dari 10 derajat celcius antara 2 hari berturut-turut pada tahun 1925?

  Input:

```
SELECT COUNT(*) AS jumlah_hari
FROM (
  SELECT *,
         LAG(suhu) OVER (PARTITION BY tahun ORDER BY suhu) AS
suhu_sebelumnya
  FROM suhutemp
  WHERE tahun = 1925
) AS t
WHERE ABS(suhu - suhu_sebelumnya) > 10;
```

  Query di atas mengambil data jumlah hari pada tabel **suhutemp** pada tahun 1925, dimana suhu hariannya berubah lebih dari 10 derajat celcius dibandingkan dengan suhu harian sebelumnya pada tiap tahun tersebut.

  Output:

```
OK
4
Time taken: 85.7 seconds, Fetched: 1 row(s)
```

2. Analisis menggunakan bentuk-bentuk Xquery transformation

a) Ganjil

- **XQuery Sederhana**

  1. Langkah pertama yang dilakukan adalah mengubah file trading1.log dan trading2.log menjadi trading1.txt dan trading2.txt
  2. Lalu buka kedua file txt menggunakan fungsi nano dan isinya seperti di bawah ini.

```
 GNU nano 2.0.9            File: trading1.txt

2021-10-28T06:00:00, indra, 100, SMN_GRSK
2021-10-28T06:00:00, brian, 102, PTR_GRSK
2021-10-28T06:00:00, fakarich, 98, SMN_GRSK
2021-10-28T06:00:00, doni, 86, PTR_GRSK
2021-10-28T06:00:00, indra, 108, PTR_GRSK
2021-10-28T06:00:00, brian, 45, SMN_GRSK
2021-10-28T06:00:00, indra, 108, PTR_GRSK
2021-10-28T06:00:00, doni, 77, SMN_GRSK
2021-10-28T06:00:00, doni, 67, PTR_GRSK
2021-10-28T06:00:00, fakarich, 101, PTR_GRSK
2021-10-28T06:00:00, fakarich, 45, WLMR_GRSK
2021-10-28T06:00:00, doni, 56, PTR_GRSK
2021-10-28T06:00:00, doni, 73, SMN_GRSK
2021-10-28T06:00:00, indra, 82, PTR_GRSK
2021-10-28T06:00:00, brian, 90, SMN_GRSK
2021-10-28T06:00:00, brian, 34, WLMR_GRSK
2021-10-28T06:00:00, indra, 109, SMN_GRSK
2021-10-28T06:00:00, indra, 99, WLMR_GRSK
2021-10-28T06:00:00, fakarich, 110, SMN_GRSK
                          [ Read 20 lines ]
```

```
 GNU nano 2.0.9            File: trading2.txt

2021-11-10T06:00:00, brian, 66, SMN_GRSK
2021-11-10T06:00:00, fakarich, 98, PTR_GRSK
2021-11-10T06:00:00, doni, 93, SMN_GRSK
2021-11-10T06:00:00, indra, 82, PTR_GRSK
2021-11-10T06:00:00, indra, 103, PTR_GRSK
2021-11-10T06:00:00, fakarich, 64, SMN_GRSK
2021-11-10T06:00:00, doni, 91, PTR_GRSK
2021-11-10T06:00:00, indra, 71, SMN_GRSK
2021-11-10T06:00:00, indra, 63, PTR_GRSK
2021-11-10T06:00:00, doni, 62, PTR_GRSK
2021-11-10T06:00:00, doni, 32, WLMR_GRSK
2021-11-10T06:00:00, indra, 33, PTR_GRSK
2021-11-10T06:00:00, indra, 75, SMN_GRSK
2021-11-10T06:00:00, fakarich, 87, PTR_GRSK
2021-11-10T06:00:00, doni, 95, SMN_GRSK
2021-11-10T06:00:00, doni, 23, WLMR_GRSK
2021-11-10T06:00:00, brian, 104, SMN_GRSK
2021-11-10T06:00:00, brian, 76, WLMR_GRSK
2021-11-10T06:00:00, doni, 119, SMN_GRSK
                          [ Read 20 lines ]
```

3. Selanjutnya simpan data di HDFS dengan menggunakan syntax di bawah ini.

```
[oracle@bigdatalite ~]$ hdfs dfs -copyFromLocal trading1.txt
[oracle@bigdatalite ~]$ hdfs dfs -copyFromLocal trading2.txt
```

4. Untuk menggabungkan kedua file tersebut, buat file .xq terlebih dahulu dan isi dengan syntax di bawah ini, lalu save **ctrl+s** dan exit **ctrl+x**.

```
 GNU nano 2.0.9            File: trading.xq                    Modified

import module "oxh:text";
for $line in text:collection ("trading*.txt")
return text:put($line || ",in class")
```

5. Setelah itu ketik perintah ini.

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar trading.xq -output ./my
out1a -print
```

6. Maka akan muncul output.

```
23/03/25 11:22:18 INFO hadoop.xquery: Finished executing "trading.xq". Output pa
th: "hdfs://bigdatalite.localdomain:8020/user/oracle/myout1a"
 2021-10-28T06:00:00, indra, 100, SMN_GRSK,in class
 2021-10-28T06:00:00, brian, 102, PTR_GRSK,in class
 2021-10-28T06:00:00, fakarich, 98, SMN_GRSK,in class
 2021-10-28T06:00:00, doni, 86, PTR_GRSK,in class
 2021-10-28T06:00:00, indra, 108, PTR_GRSK,in class
 2021-10-28T06:00:00, brian, 45, SMN_GRSK,in class
 2021-10-28T06:00:00, indra, 108, PTR_GRSK,in class
 2021-10-28T06:00:00, doni, 77, SMN_GRSK,in class
 2021-10-28T06:00:00, doni, 67, PTR_GRSK,in class
 2021-10-28T06:00:00, fakarich, 101, PTR_GRSK,in class
 2021-10-28T06:00:00, fakarich, 45, WLMR_GRSK,in class
 2021-10-28T06:00:00, doni, 56, PTR_GRSK,in class
 2021-10-28T06:00:00, doni, 73, SMN_GRSK,in class
 2021-10-28T06:00:00, indra, 82, PTR_GRSK,in class
 2021-10-28T06:00:00, brian, 90, SMN_GRSK,in class
 2021-10-28T06:00:00, brian, 34, WLMR_GRSK,in class
 2021-10-28T06:00:00, indra, 109, SMN_GRSK,in class
 2021-10-28T06:00:00, indra, 99, WLMR_GRSK,in class
 2021-10-28T06:00:00, fakarich, 110, SMN_GRSK,in class
 2021-10-28T06:00:00, fakarich, 88, WLMR_GRSK,in class
 2021-11-10T06:00:00, brian, 66, SMN_GRSK,in class
 2021-11-10T06:00:00, fakarich, 98, PTR_GRSK,in class
```

- **XQuery Basic Filtering**
  1. Membuat direktori bernama **/mydata** pada hadoop filesystem.
     ```
     [oracle@bigdatalite ~]$ hadoop fs -mkdir -p /user/oracle/mydata
     ```
  2. Copy file trading1.log dan trading2.log ke dalam direktori yang sudah dibuat sebelumnya.
     ```
     [oracle@bigdatalite ~]$ hdfs dfs -copyFromLocal trading1.log /user/oracle/mydata
     [oracle@bigdatalite ~]$ hdfs dfs -copyFromLocal trading2.log /user/oracle/mydata
     ```
  3. Selanjutnya buat file baru **basicfilter.xq** untuk mengetahui berapa kali brian mengunjungi halaman dan isi file dengan syntax di bawah ini.
     ```
       GNU nano 2.0.9                File: basicfilter.xq

     import module "oxh:text";
     for $line in
     text:collection("mydata/trading*.log")
     let $split := fn:tokenize($line, "\s*,\s*")
     where $split[2] eq "brian"
     return text:put($line)
     ```
  4. Untuk menjalankan file XQuery di atas, dapat menggunakan syntax ini.
     ```
     [oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar basicfilter.xq -output ./mydata/myoutbas
     icfilter -print
     ```
  5. Jika berhasil, akan menghasilkan output.
     ```
     23/03/25 11:50:26 INFO hadoop.xquery: Finished executing "basicfilter.xq". Output path: "hdfs://b
     igdatalite.localdomain:8020/user/oracle/mydata/myoutbasicfilter"
      2021-10-28T06:00:00, brian, 102, PTR_GRSK
      2021-10-28T06:00:00, brian, 45, SMN_GRSK
      2021-10-28T06:00:00, brian, 90, SMN_GRSK
      2021-10-28T06:00:00, brian, 34, WLMR_GRSK
      2021-11-10T06:00:00, brian, 66, SMN_GRSK
      2021-11-10T06:00:00, brian, 104, SMN_GRSK
      2021-11-10T06:00:00, brian, 76, WLMR_GRSK
     ```

- **Group by and Aggregation**
  1. Membuat file XQuery dengan nama **groupaggregation.xq** menggunakan perintah nano. Lalu save dan exit. XQuery ini digunakan untuk mengetahui berapa banyak kunjungan user tiap hari berdasarkan tanggal.

```
 GNU nano 2.0.9                File: groupaggregation.xq

import module "oxh:text";
for $line in text:collection("mydata/trading*.log")
let $split := fn:tokenize($line, "\s*,\s*")
let $time := xs:dateTime($split[1])
let $day := xs:date($time)
group by $day
return text:put($day || " => " || fn:count($line))
```

2. Untuk menjalankan file tersebut dapat menggunakan syntax di bawah ini.

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar groupaggregation.xq -output ./mydata/myo
utgroupaggregation -print
```

3. Jika berhasil, maka akan muncul output seperti ini.

```
23/03/25 12:03:36 INFO hadoop.xquery: Finished executing "groupaggregation.xq". Output path: "hdf
s://bigdatalite.localdomain:8020/user/oracle/mydata/myoutgroupaggregation"
2021-10-28 => 20
2021-11-10 => 20
```

- **Inner Joins**

1. Copy file trader.txt ke dalam direktori hadoop file system.

```
[oracle@bigdatalite ~]$ hdfs dfs -copyFromLocal trader.txt /user/oracle/mydata
```

2. Membuat file XQuery bernama **innerjoin1.xq** dan **innerjoin2.xq** menggunakan perintah nano dan isi file dengan syntax di bawah ini.

```
 GNU nano 2.0.9                File: innerjoin1.xq

import module "oxh:text";
for $traderLine in text:collection("mydata/trader.txt")
let $userSplit := fn:tokenize($traderLine, "\s*:\s*")
let $traderId := $userSplit[1]


for $tradingLine in text:collection("mydata/trading*.log")
let $tradingSplit := fn:tokenize($tradingLine,"\s*,\s*")
let $tradingTraderId := $tradingSplit[2]
let $tradingPage := concat($tradingSplit[2],"-",$tradingSplit[4])
let $tradingSaham := xs:integer($tradingSplit[3][.castable as xs:integer])

where $tradingTraderId eq $traderId and $tradingSaham gt 70
group by $tradingPage
return text:put($tradingPage || " " || fn:count($tradingLine))
```

```
 GNU nano 2.0.9                File: innerjoin2.xq

import module "oxh:text";
for $userLine in text:collection("mydata/trader.txt")
let $userSplit := fn:tokenize($userLine, "\s*:\s*")
let $userId := $userSplit[1]

for $visitLine in text:collection("mydata/trading*.log")
[$userId eq fn:tokenize(., "\s*,\s*")[2]]

group by $userId
return text:put($userId || " " || fn:count($visitLine))
```

3. Untuk menjalankan file di atas dapat menggunakan syntax seperti gambar di bawah ini.

Innerjoin1

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar innerjoin1.xq -output ./mydata/myoutinne
rjoin1 -print
```

Innerjoin2

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar innerjoin2.xq -output ./mydata/myoutinne
rjoin2 -print
```

4. Dan jika berhasil, maka akan muncul output.

Innerjoin1

```
23/03/25 12:44:34 INFO hadoop.xquery: Finished executing "innerjoin1.xq". Output path: "hdfs://bi
gdatalite.localdomain:8020/user/oracle/mydata/myoutinnerjoin1"
brian-PTR_GRSK 1
brian-SMN_GRSK 2
brian-WLMR_GRSK 1
doni-PTR_GRSK 2
doni-SMN_GRSK 5
fakarich-PTR_GRSK 3
fakarich-SMN_GRSK 2
fakarich-WLMR_GRSK 1
indra-PTR_GRSK 5
indra-SMN_GRSK 4
indra-WLMR_GRSK 2
```

Innerjoin2

```
23/03/25 12:50:10 INFO hadoop.xquery: Finished executing "innerjoin2.xq". Output path: "hdfs://bi
gdatalite.localdomain:8020/user/oracle/mydata/myoutinnerjoin2"
brian 7
doni 12
fakarich 8
indra 13
```

- **Left Outer Joins**

1. Buat file XQuery bernama **outerjoin.xq** menggunakan perintah nano dan isi file tersebut dengan syntax di bawah ini. XQuery ini berguna untuk menampilkan dan menghitung user yang mengakses suatu halaman maupun yang tidak.

```
  GNU nano 2.0.9                  File: outerjoin.xq

import module "oxh:text";

for $userLine in text:collection("mydata/trader.txt")
let $userSplit := fn:tokenize($userLine, "\s*:\s*")
let $userId := $userSplit[1]

for $visitLine allowing empty in
text:collection("mydata/trading*.log")
[$userId eq fn:tokenize(., "\s*,\s*")[2]]

group by $userId
return text:put($userId || " " || fn:count($visitLine))
```

2. Untuk menjalankan file di atas, dapat menjalankan perintah di bawah ini.

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar outerjoin.xq -output ./mydata/myoutouter
join -print
```

3. Jika berhasil, maka akan muncul output.

```
23/03/25 13:01:22 INFO hadoop.xquery: Finished executing "outerjoin.xq". Output path: "hdfs://big
datalite.localdomain:8020/user/oracle/mydata/myoutouterjoin"
brian 7
doni 12
fakarich 8
indra 13
```

- **Semijoins**

1. Buat file XQuery dengan nama **semijoin.xq** menggunakan perintah nano dan isi file dengan syntax di bawah ini. File ini berguna untuk menampilkan user siapa saja yang pernah mengakses halaman berdasarkan kecocokan antara id user dan data kunjungan user.

```
 GNU nano 2.0.9                  File: semijoin.xq

import module "oxh:text";
for $userLine in text:collection("mydata/trader.txt")
let $userId := fn:tokenize($userLine, "\s*:\s*")[1]

where some $visitLine in text:collection("mydata/trading*.log")
satisfies $userId eq fn:tokenize($visitLine, "\s*,\s*")[2]

return text:put($userId)
```

2. Untuk menjalankan file di atas, dapat menggunakan syntax seperti biasanya.

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar semijoin.xq -output ./mydata/myoutsemijo
in -print
```

3. Jika berhasil, maka akan muncul output.

```
23/03/25 13:10:27 INFO hadoop.xquery: Finished executing "semijoin.xq". Output path: "hdfs://bigd
atalite.localdomain:8020/user/oracle/mydata/myoutsemijoin"
brian
doni
fakarich
indra
```

- **Multiple Outputs**

1. Membuat file XQuery dengan nama **multiple.xq** menggunakan perintah nano dan isi file tersebut dengan syntax di bawah ini. File ini berguna untuk melacak user yang memiliki visitcode 401.

```
 GNU nano 2.0.9                File: multiple.xq

import module "oxh:text";
for $visitLine in text:collection("mydata/trading*.log")
let $visitCode := xs:integer(fn:tokenize($visitLine, "\s*,\s*")[4])
return if ($visitCode eq 401) then text:trace($visitLine) else text:put($visitLine)
```

2. Untuk menjalankan file tersebut, dapat menggunakan syntax seperti biasanya.

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar multiple.xq -output ./mydata/myoutmultip
le -print
```

3. Jika berhasil, maka akan muncul output.

```
23/03/25 13:40:42 INFO hadoop.xquery: Finished executing "multiple.xq". Output path: "hdfs://bigd
atalite.localdomain:8020/user/oracle/mydata/myoutmultiple"
 2021-10-28T06:00:00, indra, 100, SMN_GRSK
 2021-10-28T06:00:00, brian, 102, PTR_GRSK
 2021-10-28T06:00:00, fakarich, 98, SMN_GRSK
 2021-10-28T06:00:00, doni, 86, PTR_GRSK
 2021-10-28T06:00:00, indra, 108, PTR_GRSK
 2021-10-28T06:00:00, brian, 45, SMN_GRSK
 2021-10-28T06:00:00, indra, 108, PTR_GRSK
 2021-10-28T06:00:00, doni, 77, SMN_GRSK
 2021-10-28T06:00:00, doni, 67, PTR_GRSK
 2021-10-28T06:00:00, fakarich, 101, PTR_GRSK
 2021-10-28T06:00:00, fakarich, 45, WLMR_GRSK
 2021-10-28T06:00:00, doni, 56, PTR_GRSK
 2021-10-28T06:00:00, doni, 73, SMN_GRSK
 2021-10-28T06:00:00, indra, 82, PTR_GRSK
 2021-10-28T06:00:00, brian, 90, SMN_GRSK
 2021-10-28T06:00:00, brian, 34, WLMR_GRSK
 2021-10-28T06:00:00, indra, 109, SMN_GRSK
 2021-10-28T06:00:00, indra, 99, WLMR_GRSK
 2021-10-28T06:00:00, fakarich, 110, SMN_GRSK
 2021-10-28T06:00:00, fakarich, 88, WLMR_GRSK
 2021-11-10T06:00:00, brian, 66, SMN_GRSK
 2021-11-10T06:00:00, fakarich, 98, PTR_GRSK
```

- **Accessing Auxiliary Input Data**

1. Buat file XQuery bernama **accessing.xq** menggunakan perintah nano dan isi file tersebut dengan syntax di bawah ini.

```
  GNU nano 2.0.9                File: accessing.xq

import module "oxh:text";
for $visitLine in text:collection("mydata/trading*.log")
let $visitUserId := fn:tokenize($visitLine, "\s*,\s*")[2]

for $userLine in fn:unparsed-text-lines("trader.txt")
let $userSplit := fn:tokenize($userLine, "\s*:\s*")
let $userId := $userSplit[1]

where $userId eq $visitUserId

group by $userId
return text:put($userId || " " || fn:count($visitLine))
```

2. Untuk menjalankan file tersebut, dapat menggunakan syntax seperti di bawah ini.

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar multiple.xq -output ./mydata/myoutmultip
le -print
```

3. Jika berhasil, maka akan muncul output.

```
23/03/25 13:49:55 INFO hadoop.xquery: Finished executing "accessing.xq". Output path: "hdfs://big
datalite.localdomain:8020/user/oracle/mydata/myoutaccessing"
brian 7
doni 12
fakarich 8
indra 13
```

- **Calling a Custom Java Function from XQuery**

1. Buat file XQuery bernama **calling.xq** menggunakan perintah nano da nisi file tersebut dengan syntax di bawah ini.

```
 GNU nano 2.0.9                  File: calling.xq

import module "oxh:text";

declare %ora-java:binding("java.lang.String#format")
function local:string-format($pattern as xs:string, $data as xs:anyAtomicType*) as
xs:string external;

for $line in text:collection("mydata/trader*.txt")
let $split := fn:tokenize($line, "\s*:\s*")
return text:put(local:string-format("%s,%s,%s", $split))
```

2. Untuk menjalankan file tersebut, dapat menggunakan syntax seperti biasanya.

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar calling.xq -output ./mydata/myoutcalling
 -print
```

3. Jika berhasil, maka akan muncul output.

```
23/03/25 13:58:03 INFO hadoop.xquery: Finished executing "calling.xq". Output path: "hdfs://bigda
talite.localdomain:8020/user/oracle/mydata/myoutcalling"
indra,26,Indra Kenz
fakarich,31,Fakar Suhartami
doni,24,Doni Salmanan
brian,27,Brian Edgar Nababan
```

- **Using User-defined XQuery Library Modules and XML Schemas**

1. Buat file XQuery bernama **mytools1.xq** dan **mytools2.xq** menggunakan perintah nano da nisi file tersebut dengan syntax di bawah ini.

Mytools1.xq

```
 GNU nano 2.0.9                  File: mytools1.xq

module namespace mytools = "urn:mytools";
declare %ora-java:binding("java.lang.String#format")
function mytools:string-format($pattern as xs:string, $data as xs:anyAtomicType*) as
xs:string external;
```

Mytools2.xq

```
 GNU nano 2.0.9                  File: mytools2.xq

import module namespace mytools = "urn:mytools" at "mytools1.xq";
import module "oxh:text";

for $line in text:collection("mydata/trader*.txt")
let $split := fn:tokenize($line, "\s*:\s*")
return text:put(mytools:string-format("%s,%s,%s", $split))
```

2. Buat direktori bernama **mytools** dan copy file **mytools1.xq** dan **mytools2.xq** ke dalam direktori tersebut.

```
[oracle@bigdatalite ~]$ mkdir mytools
[oracle@bigdatalite ~]$ mv mytools1.xq mytools2.xq mytools
[oracle@bigdatalite ~]$ cd mytools
[oracle@bigdatalite mytools]$ ls
mytools1.xq  mytools2.xq
```

3. Untuk menjalankan file-file tersebut, dapat menggunakan syntax di bawah ini dan tetap berada pada direktori **mytools**.

```
[oracle@bigdatalite mytools]$ hadoop jar $OXH_HOME/lib/oxh.jar -files mytools1.xq mytools2.xq -ou
tput ./mydata/mydata-out -print
```

4. Jika berhasil, maka akan muncul output.

```
23/03/25 14:10:05 INFO hadoop.xquery: Finished executing "mytools2.xq". Output path: "hdfs://bigd
atalite.localdomain:8020/user/oracle/mydata/mydata-out"
indra,26,Indra Kenz
fakarich,31,Fakar Suhartami
doni,24,Doni Salmanan
brian,27,Brian Edgar Nababan
```

Secara keseluruhan, data XQuery dan output tersimpan di dalam hadoop file system.

/user/oracle/mydata/ Go!

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	oracle	oracle	0 B	Sat Mar 25 14:10:03 -0400 2023	0	0 B	mydata-out
drwxr-xr-x	oracle	oracle	0 B	Sat Mar 25 13:49:53 -0400 2023	0	0 B	myoutaccessing
drwxr-xr-x	oracle	oracle	0 B	Sat Mar 25 11:50:23 -0400 2023	0	0 B	myoutbasicfilter
drwxr-xr-x	oracle	oracle	0 B	Sat Mar 25 13:58:00 -0400 2023	0	0 B	myoutcalling
drwxr-xr-x	oracle	oracle	0 B	Sat Mar 25 12:03:34 -0400 2023	0	0 B	myoutgroupaggregation
drwxr-xr-x	oracle	oracle	0 B	Sat Mar 25 12:44:32 -0400 2023	0	0 B	myoutinnerjoin1
drwxr-xr-x	oracle	oracle	0 B	Sat Mar 25 12:50:08 -0400 2023	0	0 B	myoutinnerjoin2
drwxr-xr-x	oracle	oracle	0 B	Sat Mar 25 13:40:40 -0400 2023	0	0 B	myoutmultiple
drwxr-xr-x	oracle	oracle	0 B	Sat Mar 25 13:01:20 -0400 2023	0	0 B	myoutouterjoin
drwxr-xr-x	oracle	oracle	0 B	Sat Mar 25 13:10:24 -0400 2023	0	0 B	myoutsemijoin
-rw-r--r--	oracle	oracle	98 B	Sat Mar 25 12:08:25 -0400 2023	1	64 MB	trader.txt
-rw-r--r--	oracle	oracle	860 B	Sat Mar 25 11:34:45 -0400 2023	1	64 MB	trading1.log
-rw-r--r--	oracle	oracle	848 B	Sat Mar 25 11:35:11 -0400 2023	1	64 MB	trading2.log

b) Genap

➢ **XQuery Sederhana**

1. Langkah pertama yang dilakukan adalah mengubah file bahanpokok1.log dan bahanpokok2.log menjadi bahanpokok1.txt dan bahanpokok2.txt
2. Lalu buka kedua file txt menggunakan fungsi nano dan isinya seperti di bawah ini.

```
  GNU nano 2.0.9                File: bahanpokok1.txt

2021-09-28T06:00:00, jatim, 24000, telur
 2021-09-28T06:00:00, jateng, 50600, minyak_goreng
 2021-09-28T06:00:00, jabar, 7642, pertalite
 2021-09-28T06:00:00, sumbar, 12560, pertamax
 2021-09-28T06:00:00, sumut, 7633, pertalite
 2021-09-28T06:00:00, sumbar, 12450, pertamax
 2021-09-28T06:00:00, jateng, 24060, telur
 2021-09-28T06:00:00, jatim, 50300, minyak_goreng
 2021-09-28T06:00:00, jatim, 7649, pertalite
 2021-09-28T06:00:00, jateng, 12460, pertamax
 2021-09-28T06:00:00, jabar, 23450, telur
 2021-09-28T06:00:00, jatim, 50200, minyak_goreng
 2021-09-28T06:00:00, jabar, 7620, pertalite
 2021-09-28T06:00:00, jabar, 12520, pertamax
 2021-09-28T06:00:00, jabar, 24020, telur
 2021-09-28T06:00:00, sumbar, 50300, minyak_goreng
 2021-09-28T06:00:00, sumut, 7630, pertalite
 2021-09-28T06:00:00, jatim, 12600, pertamax
 2021-09-28T06:00:00, sumut, 24500, telur
                                [ Read 20 lines ]
```

```
  GNU nano 2.0.9                File: bahanpokok2.txt

2021-11-28T06:00:00, jateng, 50600, minyak_goreng
2021-11-28T06:00:00, pertalite, 7632, pertalite
2021-11-28T06:00:00, jabar, 7642, pertalite
2021-11-28T06:00:00, sumbar, 2455, telur
2021-11-28T06:00:00, sumut, 50400, minyak_goreng
2021-11-28T06:00:00, sumbar, 7630, pertalite
2021-11-28T06:00:00, jatim, 24040, telur
2021-11-28T06:00:00, jatim, 50300, minyak_goreng
2021-11-28T06:00:00, jatim, 24050, telur
2021-11-28T06:00:00, jateng, 50300, minyak_goreng
2021-11-28T06:00:00, jabar, 23430, telur
2021-11-28T06:00:00, jateng, 50100, minyak_goreng
2021-11-28T06:00:00, jatim, 7600, pertalite
2021-11-28T06:00:00, sumut, 12510, pertamax
2021-11-28T06:00:00, jatim, 24010, telur
2021-11-28T06:00:00, sumbar, 50300, minyak_goreng
2021-11-28T06:00:00, sumut, 7630, pertalite
2021-11-28T06:00:00, sumbar, 12500, pertamax
2021-11-28T06:00:00, sumut, 24500, telur
                                [ Read 20 lines ]
```

3. Selanjutnya simpan data di HDFS dengan menggunakan syntax di bawah ini.

```
[oracle@bigdatalite ~]$ hdfs dfs -copyFromLocal bahanpokok1.txt
[oracle@bigdatalite ~]$ hdfs dfs -copyFromLocal bahanpokok2.txt
```

4. Untuk menggabungkan kedua file tersebut, buat file .xq terlebih dahulu menggunakan perintah nano dan isi dengan syntax di bawah ini, lalu save **ctrl+s** dan exit **ctrl+x**.

```
  GNU nano 2.0.9                File: bahanpokok.xq

import module "oxh:text";
for $line in text:collection ("bahanpokok*.txt")
return text:put($line || ",in class")
```

5. Membuat direktori **mydata2** pada hadoop file system untuk menyimpan hasil output.

```
[oracle@bigdatalite ~]$ hadoop fs -mkdir -p /user/mydata2
```

6. Setelah itu ketik perintah ini.

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar bahanpokok.xq -output ./mydata2/myou
tbahanpokok -print
```

7. Maka akan muncul output.

```
23/03/26 10:55:20 INFO hadoop.xquery: Finished executing "bahanpokok.xq". Output path: "hdfs:
//bigdatalite.localdomain:8020/user/oracle/mydata2/myoutbahanpokok"
 2021-11-28T06:00:00, jateng, 50600, minyak_goreng,in class
 2021-11-28T06:00:00, pertalite, 7632, pertalite,in class
 2021-11-28T06:00:00, jabar, 7642, pertalite,in class
 2021-11-28T06:00:00, sumbar, 2455, telur,in class
 2021-11-28T06:00:00, sumut, 50400, minyak_goreng,in class
 2021-11-28T06:00:00, sumbar, 7630, pertalite,in class
 2021-11-28T06:00:00, jatim, 24040, telur,in class
 2021-11-28T06:00:00, jatim, 50300, minyak_goreng,in class
 2021-11-28T06:00:00, jatim, 24050, telur,in class
 2021-11-28T06:00:00, jateng, 50300, minyak_goreng,in class
 2021-11-28T06:00:00, jabar, 23430, telur,in class
 2021-11-28T06:00:00, jateng, 50100, minyak_goreng,in class
 2021-11-28T06:00:00, jatim, 7600, pertalite,in class
 2021-11-28T06:00:00, sumut, 12510, pertamax,in class
 2021-11-28T06:00:00, jatim, 24010, telur,in class
 2021-11-28T06:00:00, sumbar, 50300, minyak_goreng,in class
 2021-11-28T06:00:00, sumut, 7630, pertalite,in class
 2021-11-28T06:00:00, sumbar, 12500, pertamax,in class
 2021-11-28T06:00:00, sumut, 24500, telur,in class
 2021-11-28T06:00:00, sumut, 50100, minyak_goreng,in class
 2021-09-28T06:00:00, jatim, 24000, telur,in class
 2021-09-28T06:00:00, jateng, 50600, minyak_goreng,in class
```

- **XQuery Basic Filtering**

1. Copy file bahanpokok1.log dan bahanpokok2.log ke dalam direktori yang sudah dibuat sebelumnya.

```
[oracle@bigdatalite ~]$ hdfs dfs -copyFromLocal bahanpokok1.log /user/oracle/mydata2
[oracle@bigdatalite ~]$ hdfs dfs -copyFromLocal bahanpokok2.log /user/oracle/mydata2
```

2. Selanjutnya buat file baru **basicfilter2.xq** dan isi file dengan syntax di bawah ini.

```
  GNU nano 2.0.9                File: basicfilter2.xq

import module "oxh:text";
for $line in
text:collection("mydata2/bahanpokok*.log")
let $split := fn:tokenize($line, "\s*,\s*")
where $split[2] eq "jabar"
return text:put($line)
```

3. Untuk menjalankan file XQuery di atas, dapat menggunakan syntax ini.

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar basicfilter2.xq -output ./mydata2/my
outbasicfilter -print
```

4. Jika berhasil, akan menghasilkan output.

```
23/03/26 11:17:58 INFO hadoop.xquery: Finished executing "basicfilter2.xq". Output path: "hdf
s://bigdatalite.localdomain:8020/user/oracle/mydata2/myoutbasicfilter"
 2021-11-28T06:00:00, jabar, 7642, pertalite
 2021-11-28T06:00:00, jabar, 23430, telur
 2021-09-28T06:00:00, jabar, 7642, pertalite
 2021-09-28T06:00:00, jabar, 23450, telur
 2021-09-28T06:00:00, jabar, 7620, pertalite
 2021-09-28T06:00:00, jabar, 12520, pertamax
 2021-09-28T06:00:00, jabar, 24020, telur
```

- **Group by and Aggregation**

1. Membuat file XQuery dengan nama **groupaggregation2.xq** menggunakan perintah nano. Lalu save dan exit.

```
 GNU nano 2.0.9                File: groupaggregation2.xq

import module "oxh:text";
for $line in text:collection("mydata2/bahanpokok*.log")
let $split := fn:tokenize($line, "\s*,\s*")
let $time := xs:dateTime($split[1])
let $day := xs:date($time)
group by $day
return text:put($day || " => " || fn:count($line))
```

2. Untuk menjalankan file tersebut dapat menggunakan syntax di bawah ini.

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar groupaggregation2.xq -output ./mydat
a2/myoutgroupaggregation -print
```

3. Jika berhasil, maka akan muncul output seperti ini.

```
23/03/26 11:29:54 INFO hadoop.xquery: Finished executing "groupaggregation2.xq". Output path:
 "hdfs://bigdatalite.localdomain:8020/user/oracle/mydata2/myoutgroupaggregation"
2021-09-28 => 20
2021-11-28 => 20
```

- **Inner Joins**

1. Copy file provinsi.txt ke dalam direktori hadoop file system.

```
[oracle@bigdatalite ~]$ hdfs dfs -copyFromLocal provinsi.txt /user/oracle/mydata2
```

2. Membuat file XQuery bernama **innerjoins1.xq** dan **innerjoins2.xq** menggunakan perintah nano dan isi file dengan syntax di bawah ini.

```
 GNU nano 2.0.9                File: innerjoins1.xq

import module "oxh:text";
for $traderLine in text:collection("mydata2/provinsi.txt")
let $userSplit := fn:tokenize($traderLine, "\s*:\s*")
let $traderId := $userSplit[1]


for $tradingLine in text:collection("mydata2/bahanpokok*.log")
let $tradingSplit := fn:tokenize($tradingLine,"\s*,\s*")
let $tradingTraderId := $tradingSplit[2]
let $tradingPage := concat($tradingSplit[2],"-",$tradingSplit[4])
let $tradingSaham := xs:integer($tradingSplit[3][.castable as xs:integer])

where $tradingTraderId eq $traderId and $tradingSaham gt 70
group by $tradingPage
return text:put($tradingPage || " " || fn:count($tradingLine))
```

```
 GNU nano 2.0.9                File: innerjoins2.xq

import module "oxh:text";
for $userLine in text:collection("mydata2/provinsi.txt")
let $userSplit := fn:tokenize($userLine, "\s*:\s*")
let $userId := $userSplit[1]

for $visitLine in text:collection("mydata2/bahanpokok*.log")
[$userId eq fn:tokenize(., "\s*,\s*")[2]]

group by $userId
return text:put($userId || " " || fn:count($visitLine))
```

3. Untuk menjalankan file di atas dapat menggunakan syntax seperti gambar di bawah ini.

Innerjoins1

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar innerjoins1.xq -output ./mydata2/myo
utinnerjoins1 -print
```

Innerjoins2

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar innerjoins2.xq -output ./mydata2/myo
utinnerjoins2 -print
```

4. Dan jika berhasil, maka akan muncul output.

Innerjoins1

```
23/03/26 11:43:33 INFO hadoop.xquery: Finished executing "innerjoins1.xq". Output path: "hdfs
://bigdatalite.localdomain:8020/user/oracle/mydata2/myoutinnerjoins1"
jabar-pertalite 3
jabar-pertamax 1
jabar-telur 3
jateng-minyak_goreng 4
jateng-pertamax 1
jateng-telur 1
jatim-minyak_goreng 3
jatim-pertalite 2
jatim-pertamax 1
jatim-telur 4
sumbar-minyak_goreng 3
sumbar-pertalite 1
sumbar-pertamax 3
sumbar-telur 1
sumut-minyak_goreng 2
sumut-pertalite 3
sumut-pertamax 1
sumut-telur 2
```

Innerjoins2

```
23/03/26 11:47:05 INFO hadoop.xquery: Finished executing "innerjoins2.xq". Output path: "hdfs
://bigdatalite.localdomain:8020/user/oracle/mydata2/myoutinnerjoins2"
jabar 7
jateng 6
jatim 10
sumbar 8
sumut 8
```

- **Left Outer Joins**

1. Buat file XQuery bernama **outerjoin2.xq** menggunakan perintah nano dan isi file tersebut dengan syntax di bawah ini.

```
  GNU nano 2.0.9                File: outerjoin2.xq

import module "oxh:text";

for $userLine in text:collection("mydata2/provinsi.txt")
let $userSplit := fn:tokenize($userLine, "\s*:\s*")
let $userId := $userSplit[1]

for $visitLine allowing empty in
text:collection("mydata2/bahanpokok*.log")
[$userId eq fn:tokenize(., "\s*,\s*")[2]]

group by $userId
return text:put($userId || " " || fn:count($visitLine))
```

2. Untuk menjalankan file di atas, dapat menjalankan perintah di bawah ini.

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar outerjoin2.xq -output ./mydata2/myou
touterjoin2 -print
```

3. Jika berhasil, maka akan muncul output.

```
23/03/26 11:54:37 INFO hadoop.xquery: Finished executing "outerjoin2.xq". Output path: "hdfs:
//bigdatalite.localdomain:8020/user/oracle/mydata2/myoutouterjoin2"
jabar 7
jateng 6
jatim 10
sumbar 8
sumut 8
```

- **Semijoins**

1. Buat file XQuery dengan nama **semijoin2.xq** menggunakan perintah nano dan isi file dengan syntax di bawah ini.

```
 GNU nano 2.0.9                File: semijoin2.xq

import module "oxh:text";
for $userLine in text:collection("mydata2/provinsi.txt")
let $userId := fn:tokenize($userLine, "\s*:\s*")[1]

where some $visitLine in text:collection("mydata2/bahanpokok*.log")
satisfies $userId eq fn:tokenize($visitLine, "\s*,\s*")[2]

return text:put($userId)
```

2. Untuk menjalankan file di atas, dapat menggunakan syntax seperti biasanya.

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar semijoin2.xq -output ./mydata2/myout
semijoin2 -print
```

3. Jika berhasil, maka akan muncul output.

```
23/03/26 12:01:01 INFO hadoop.xquery: Finished executing "semijoin2.xq". Output path: "hdfs:/
/bigdatalite.localdomain:8020/user/oracle/mydata2/myoutsemijoin2"
jabar
jateng
jatim
sumbar
sumut
```

- **Multiple Outputs**

1. Membuat file XQuery dengan nama **multiple2.xq** menggunakan perintah nano dan isi file tersebut dengan syntax di bawah ini.

```
 GNU nano 2.0.9                File: multiple2.xq

import module "oxh:text";
for $visitLine in text:collection("mydata2/bahanpokok*.log")
let $visitCode := xs:integer(fn:tokenize($visitLine, "\s*,\s*")[3])
return if ($visitCode eq 401) then text:trace($visitLine) else text:put($visitLine)
```

2. Untuk menjalankan file tersebut, dapat menggunakan syntax seperti biasanya.

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar multiple2.xq -output ./mydata2/myout
multiple2 -print
```

3. Jika berhasil, maka akan muncul output.

```
23/03/26 12:15:48 INFO hadoop.xquery: Finished executing "multiple2.xq". Output path: "hdfs:/
/bigdatalite.localdomain:8020/user/oracle/mydata2/myoutmultiple2"
 2021-11-28T06:00:00, jateng, 50600, minyak_goreng
 2021-11-28T06:00:00, pertalite, 7632, pertalite
 2021-11-28T06:00:00, jabar, 7642, pertalite
 2021-11-28T06:00:00, sumbar, 2455, telur
 2021-11-28T06:00:00, sumut, 50400, minyak_goreng
 2021-11-28T06:00:00, sumbar, 7630, pertalite
 2021-11-28T06:00:00, jatim, 24040, telur
 2021-11-28T06:00:00, jatim, 50300, minyak_goreng
 2021-11-28T06:00:00, jatim, 24050, telur
 2021-11-28T06:00:00, jateng, 50300, minyak_goreng
 2021-11-28T06:00:00, jabar, 23430, telur
 2021-11-28T06:00:00, jateng, 50100, minyak_goreng
 2021-11-28T06:00:00, jatim, 7600, pertalite
 2021-11-28T06:00:00, sumut, 12510, pertamax
 2021-11-28T06:00:00, jatim, 24010, telur
 2021-11-28T06:00:00, sumbar, 50300, minyak_goreng
 2021-11-28T06:00:00, sumut, 7630, pertalite
 2021-11-28T06:00:00, sumbar, 12500, pertamax
 2021-11-28T06:00:00, sumut, 24500, telur
 2021-11-28T06:00:00, sumut, 50100, minyak_goreng
 2021-09-28T06:00:00, jatim, 24000, telur
 2021-09-28T06:00:00, jateng, 50600, minyak_goreng
```

- **Accessing Auxiliary Input Data**

1. Buat file XQuery bernama **accessing2.xq** menggunakan perintah nano dan isi file tersebut dengan syntax di bawah ini.

```
  GNU nano 2.0.9                File: accessing2.xq

import module "oxh:text";
for $visitLine in text:collection("mydata2/bahanpokok*.log")
let $visitUserId := fn:tokenize($visitLine, "\s*,\s*")[2]

for $userLine in fn:unparsed-text-lines("provinsi.txt")
let $userSplit := fn:tokenize($userLine, "\s*:\s*")
let $userId := $userSplit[1]

where $userId eq $visitUserId

group by $userId
return text:put($userId || " " || fn:count($visitLine))
```

2. Untuk menjalankan file tersebut, dapat menggunakan syntax seperti di bawah ini.

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar -files provinsi.txt accessing2.xq -o
utput ./mydata2/myoutaccessing2 -print
```

3. Jika berhasil, maka akan muncul output.

```
23/03/26 12:21:39 INFO hadoop.xquery: Finished executing "accessing2.xq". Output path: "hdfs:
//bigdatalite.localdomain:8020/user/oracle/mydata2/myoutaccessing2"
jabar 7
jateng 6
jatim 10
sumbar 8
sumut 8
```

- **Calling a Custom Java Function from XQuery**

1. Buat file XQuery bernama **calling2.xq** menggunakan perintah nano da nisi file tersebut dengan syntax di bawah ini.

```
 GNU nano 2.0.9                File: calling2.xq

import module "oxh:text";

declare %ora-java:binding("java.lang.String#format")
function local:string-format($pattern as xs:string, $data as xs:anyAtomicType*) as
xs:string external;

for $line in text:collection("mydata2/provinsi*.txt")
let $split := fn:tokenize($line, "\s*:\s*")
return text:put(local:string-format("%s,%s,%s", $split))
```

2. Untuk menjalankan file tersebut, dapat menggunakan syntax seperti biasanya.

```
[oracle@bigdatalite ~]$ hadoop jar $OXH_HOME/lib/oxh.jar calling2.xq -output ./mydata2/myoutc
alling2 -print
```

3. Jika berhasil, maka akan muncul output.

```
23/03/26 12:25:32 INFO hadoop.xquery: Finished executing "calling2.xq". Output path: "hdfs://
bigdatalite.localdomain:8020/user/oracle/mydata2/myoutcalling2"
sumut,,Sumatera Utara
sumbar,,Sumatera Barat
jabar,,Jawa Barat
jatim,,Jawa Timur
jateng,,Jawa Tengah
```

- **Using User-defined XQuery Library Modules and XML Schemas**

1. Buat file XQuery bernama **mytool1.xq** dan **mytool2.xq** menggunakan perintah nano da nisi file tersebut dengan syntax di bawah ini.

Mytool1.xq

```
 GNU nano 2.0.9                File: mytool1.xq                                Mo

module namespace mytools = "urn:mytools";
declare %ora-java:binding("java.lang.String#format")
function mytools:string-format($pattern as xs:string, $data as xs:anyAtomicType*) as
xs:string external;
```

Mytool2.xq

```
 GNU nano 2.0.9                File: mytool2.xq

import module namespace mytools = "urn:mytools" at "mytool1.xq";
import module "oxh:text";

for $line in text:collection("mydata2/provinsi*.txt")
let $split := fn:tokenize($line, "\s*:\s*")
return text:put(mytools:string-format("%s,%s,%s", $split))
```

2. Buat direktori bernama **mytools2** dan copy file **mytool1.xq** dan **mytool2.xq** ke dalam direktori tersebut.

```
[oracle@bigdatalite ~]$ mkdir mytools2
[oracle@bigdatalite ~]$ mv mytool1.xq mytool2.xq mytools2
[oracle@bigdatalite ~]$ cd mytools2
[oracle@bigdatalite mytools2]$ ls
mytool1.xq  mytool2.xq
```

3. Untuk menjalankan file-file tersebut, dapat menggunakan syntax di bawah ini dan tetap berada pada direktori **mytools2**.

```
[oracle@bigdatalite mytools2]$ hadoop jar $OXH_HOME/lib/oxh.jar -files mytool1.xq mytool2.xq
-output ./mydata2/myoutmytools2 -print
```

4. Jika berhasil, maka akan muncul output.

```
23/03/26 12:45:35 INFO hadoop.xquery: Finished executing "mytool2.xq". Output path: "hdfs://b
igdatalite.localdomain:8020/user/oracle/mydata2/myoutmytools2"
sumut,,Sumatera Utara
sumbar,,Sumatera Barat
jabar,,Jawa Barat
jatim,,Jawa Timur
jateng,,Jawa Tengah
```

Secara keseluruhan, data XQuery dan output tersimpan di dalam hadoop file system.

/user/oracle/mydata2 Go!

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	oracle	oracle	916 B	Sun Mar 26 11:10:59 -0400 2023	1	64 MB	bahanpokok1.log
-rw-r--r--	oracle	oracle	926 B	Sun Mar 26 11:11:15 -0400 2023	1	64 MB	bahanpokok2.log
drwxr-xr-x	oracle	oracle	0 B	Sun Mar 26 12:21:37 -0400 2023	0	0 B	myoutaccessing2
drwxr-xr-x	oracle	oracle	0 B	Sun Mar 26 11:17:55 -0400 2023	0	0 B	myoutbasicfilter
drwxr-xr-x	oracle	oracle	0 B	Sun Mar 26 12:25:30 -0400 2023	0	0 B	myoutcalling2
drwxr-xr-x	oracle	oracle	0 B	Sun Mar 26 11:29:52 -0400 2023	0	0 B	myoutgroupaggregation
drwxr-xr-x	oracle	oracle	0 B	Sun Mar 26 11:43:30 -0400 2023	0	0 B	myoutinnerjoins1
drwxr-xr-x	oracle	oracle	0 B	Sun Mar 26 11:47:04 -0400 2023	0	0 B	myoutinnerjoins2
drwxr-xr-x	oracle	oracle	0 B	Sun Mar 26 12:10:01 -0400 2023	0	0 B	myoutmultiple
drwxr-xr-x	oracle	oracle	0 B	Sun Mar 26 12:15:45 -0400 2023	0	0 B	myoutmultiple2
drwxr-xr-x	oracle	oracle	0 B	Sun Mar 26 12:45:33 -0400 2023	0	0 B	myoutmytools2
drwxr-xr-x	oracle	oracle	0 B	Sun Mar 26 11:54:36 -0400 2023	0	0 B	myoutouterjoin2
drwxr-xr-x	oracle	oracle	0 B	Sun Mar 26 12:00:58 -0400 2023	0	0 B	myoutsemijoin2
-rw-r--r--	oracle	oracle	100 B	Sun Mar 26 11:31:53 -0400 2023	1	64 MB	provinsi.txt