University of Trieste

Data Management for Big Data Course

Academic Year 2022–2023

Data Warehouse case study

Davide Capone[*] Sandro Junior Della Rovere[†] Enrico Stefanel[‡]

Last update on 2023–06–22T14:05:55Z.

1 Introduction

The aim of this project is to study an efficient implementation of a suite of business oriented ad-hoc queries over the public TPC-H benchmark, which can be considered as a Big Data database, that has been implemented in Postgres.

1.1 TPC-H benchmark database

The TPC-H benchmark is a decision support benchmark that can be downloaded from the TPC official website. The data generator lets the user specify a *scale factor* in order to control the size of the resulted database. Our choices was to use a *scale factor* of 10, meaning that the overall database size is approximately 13 GB.

1.1.1 Database statistics

The benchmark is composed by eight tables:

- `CUSTOMER`, with 16 columns and 1 500 000 tuples (312 MB);
- `LINEITEM`, with 32 columns and 59 986 052 tuples (11 GB); the main attributes that are going to be used are:

[*] davide.capone@studenti.units.it

[†] sandrojunior.dellarovere@studenti.units.it

[‡] enrico.stefanel@studenti.units.it

  - ○ l_extendedprice (1 351 462 distinct values, i.e. there is an average of 44 tuples with the same value, that range from 900.91 to 104 949.50),
  - ○ l_discount (11 distinct values, i.e. there is an average of 5 453 277 tuples with the same value, that range from 0.00 to 0.10),
  - ○ l_returnflag (which can assume values A→accepted, R→returned, N→not yet delivered; the percentage of tuples for A and R are almost 25 %, while the percentage of tuples where l_returnflag is N is about 50 %),
  - ○ l_commitdate (2466 distinct values, i.e. there is an average of 24 325 tuples with the same value, that range from 1992-01-31 to 1998-10-31),
  - ○ l_receiptdate (2555 distinct values, i.e. there is an average of 23 478 tuples with the same value, that range from 1992-01-03 to 1998-12-31);

- NATION, with 8 columns and 25 tuples (24 kB);
- ORDERS, with 18 columns and 1 500 000 tuples (2481 kB); the main attributes that are going to be used are:
  - ○ o_orderdate (2406 distinct values, i.e. there is an average of 6234 tuples with the same value, that range from 1992-01-01 to 1998-08-02);
- PART, with 18 columns and 2 000 000 tuples (363 MB); the main attributes that are going to be used are:
  - ○ p_type (150 distinct values, i.e. there is an average of 13 333 tuples with the same value);
- PARTSUPP, 10 columns and with 8 000 000 tuples (1535 MB);
- REGION, 6 columns and with 5 tuples (24 kB);
- SUPPLIER, 14 columns and with 100 000 tuples (20 MB).

Other attributes have been used, but statistics about them have been omitted for lack of usefulness (e.g., keys of the tables, for which the cardinality is exactly the cardinality of the corresponding table).

1.1.2 Database SQL definition

The SQL definition of the tables can be found on the official benchmark download.

2 Set of queries

2.1 Export/import revenue value

```
WITH lineitem_orders AS (
    SELECT
        l_partkey,
        l_suppkey,
        o_orderdate,
        o_custkey,
        l_extendedprice,
        l_discount
    FROM lineitem JOIN orders ON (l_orderkey = o_orderkey)
), customer_location AS (
    SELECT
        c_custkey,
        c_name,
        n_nationkey AS c_nationkey,
        n_name AS c_nationname,
        r_regionkey AS c_regionkey,
        r_name AS c_regionname
    FROM customer
        JOIN nation ON (c_nationkey = n_nationkey)
        JOIN region ON (n_regionkey = r_regionkey)
), supplier_location AS (
    SELECT
        s_suppkey,
        s_name,
        n_nationkey AS s_nationkey,
        n_name AS s_nationname,
        r_regionkey AS s_regionkey,
        r_name AS s_regionname
    FROM supplier
        JOIN nation ON (s_nationkey = n_nationkey)
        JOIN region ON (n_regionkey = r_regionkey)
), query1 AS (
    SELECT
        EXTRACT (YEAR FROM o_orderdate) AS _year,
        EXTRACT (QUARTER FROM o_orderdate) AS _quarter,
        EXTRACT (MONTH FROM o_orderdate) AS _month,
        c_regionname,
        c_nationname,
        c_name,
        s_regionname,
        s_nationname,
        s_name,
        p_type,
        SUM(l_extendedprice * (1 - l_discount)) AS revenue
    FROM lineitem_orders
        JOIN part ON l_partkey = p_partkey
        JOIN supplier_location ON (s_suppkey = l_suppkey)
        JOIN customer_location ON (c_custkey = o_custkey)
    WHERE s_nationkey <> c_nationkey
    GROUP BY
        _year,
        _quarter,
        _month,
        c_regionkey,
        c_regionname,
        c_nationkey,
        c_nationname,
        c_custkey,
        c_name,
        s_regionkey,
        s_regionname,
        s_nationkey,
        s_nationname,
        s_suppkey,
        s_name,
        p_type
)
```

2.2 Late delivery

It is asked to retrieve the number of orders where at least one “lineitem” has been received later than the committed date. The aggregation should be performed with the Month → Year roll-up, and the (Customer’s) Nation → Region roll-up.

```
    WITH lineitem_orders AS (
    SELECT
        o_orderkey,
        l_partkey,
        l_suppkey,
        o_orderdate,
        o_custkey,
        l_commitdate,
        l_receiptdate
    FROM lineitem JOIN orders ON (l_orderkey = o_orderkey)
), customer_location AS (
    SELECT
        c_custkey,
        n_nationkey AS c_nationkey,
        n_name AS c_nationname,
        r_regionkey AS c_regionkey,
        r_name AS c_regionname
    FROM customer
        JOIN nation ON (c_nationkey = n_nationkey)
        JOIN region ON (n_regionkey = r_regionkey)
), query2 AS (
SELECT
    EXTRACT(YEAR FROM o_orderdate) AS _year,
    EXTRACT(MONTH FROM o_orderdate) AS _month,
    c_regionname,
    c_nationname,
    COUNT(DISTINCT(o_orderkey)) AS orders_no
FROM lineitem_orders
    JOIN part ON l_partkey = p_partkey
    JOIN customer_location ON (c_custkey = o_custkey)
WHERE
    l_receiptdate > l_commitdate
    -- AND _month = 1
    -- AND p_type = ’PROMO BURNISHED COPPER’
GROUP BY
    _year,
    _month,
    c_regionkey,
    c_regionname,
    c_nationkey,
    c_nationname
)
SELECT * FROM query2;
```

2.3 Returned item loss

It is asked to retrieve the *revenue loss* for customers who might be having problems with the parts that are shipped to them, where a *revenue loss* is defined as

```
SUM(l_extendedprice*(1-l_discount))
```

for all qualifying *lineitems*.

```
WITH lineitem_orders AS (
    SELECT
```

```sql
        o_orderkey,
        l_partkey,
        l_suppkey,
        o_orderdate,
        o_custkey,
        l_extendedprice,
        l_discount,
        l_returnflag,
        l_commitdate,
        l_receiptdate
    FROM lineitem JOIN orders ON (l_orderkey=o_orderkey)
),
query3 AS (
SELECT
    EXTRACT(YEAR FROM o_orderdate) AS _year,
    EXTRACT(QUARTER FROM o_orderdate) AS _quarter,
    EXTRACT(MONTH FROM o_orderdate) AS _month,
    c_name,
    SUM(l_extendedprice*(1-l_discount)) AS returnloss
FROM
    lineitem_orders
    JOIN customer ON (o_custkey=c_custkey)
WHERE
    l_returnflag='R'
    -- AND c_name='Customer#000129976'
    -- AND EXTRACT(QUARTER FROM o_orderdate) = 1
GROUP BY
    _year,
    _quarter,
    _month,
    c_custkey,
    c_name
)
SELECT * FROM query3;
```

Five independent runs of the above query obtained the following execution times: 753 541.744 ms, 672 530.120 ms, 624 276.525 ms, 615 741.447 ms and 634 262.713 ms.

3 Indexes design

. . .

4 Materialisation

...

5 Conclusions

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque mollis odio felis, eu aliquam velit congue at. Morbi et aliquet risus. Sed varius, diam vel semper varius, nibh mi bibendum est, vel molestie magna elit nec elit. Morbi ut facilisis mi. Suspendisse tristique porta purus, ac convallis tellus porta et. Maecenas ac semper justo, ut tincidunt turpis. Proin euismod viverra nisi iaculis accumsan. Suspendisse hendrerit suscipit purus sed viverra. Nullam gravida suscipit sagittis. Donec at pulvinar nunc. Donec consequat nibh elit, ornare posuere nisi feugiat laoreet. Vivamus nec imperdiet justo.

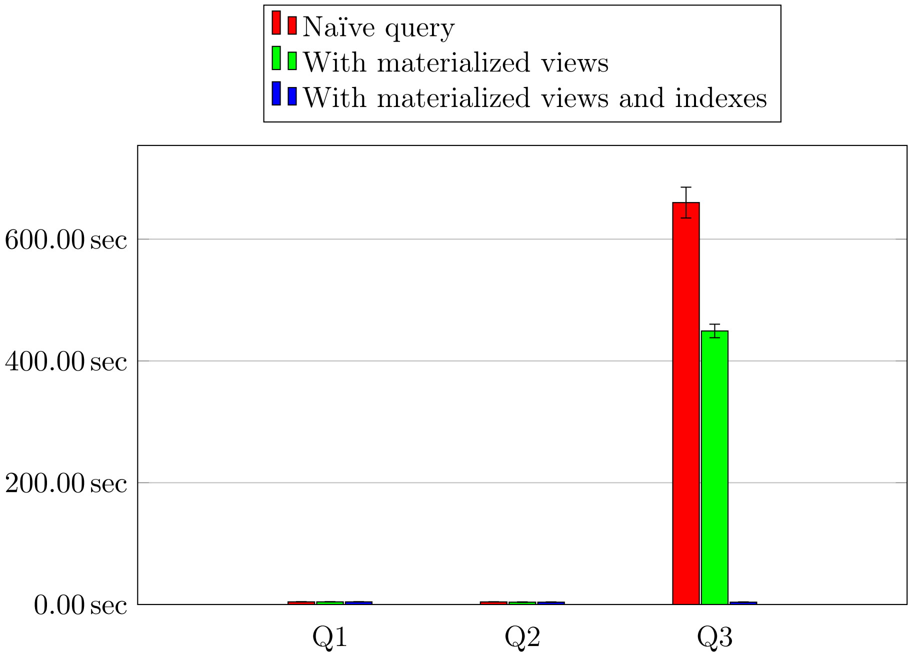

Figure 1: Query timings

Mauris vitae eros lacus. Etiam facilisis orci ac nunc tincidunt, sed vehicula sem ultrices. Sed dignissim velit elit, vel dictum tortor feugiat eget. Nulla nec sagittis quam. Cras diam mauris, suscipit vel urna in, congue laoreet diam. In consequat, nisi non iaculis interdum, risus lacus auctor nisi, in sagittis massa libero in nibh. Phasellus venenatis urna eu nibh maximus ullamcorper. Pellentesque tincidunt, nulla vel ultrices malesuada, justo velit sollicitudin orci, vitae euismod ipsum nisl non magna.