

# TPC-H em MongoDB e PostgreSQL

---

fabiohenrique@alunos.utfpr.edu.br

2025-11-15

## 1. Introdução

Para executar as 3 primeiras queries TPC-H no MongoDB eu tive que seguir o passo a passo do tpch-kit no seguinte link: <https://github.com/gregrahn/tpch-kit> depois escrever um script para exportar os dados de tbl para csv para importar as tabelas do tpc-h no mongodb e converter as queries PostgreSQL para mql.

### 1.1. Como instalar o tpch-kit


Primeiro e necessário clonar o repositório do tpch-kit:

```
1 git clone https://github.com/gregrahn/tpch-kit.git
2 cd tpch-kit/dbgen
```



Eu preferi buildar para PostgreSQL então tive que ajustar algumas flags de compilação no arquivo Makefile:

```
CFLAGS = -std=gnu99 -g -DDBNAME=\"dss\" -  
1 D$(MACHINE) -D$(DATABASE) -D$(WORKLOAD) -  
DRNG_TEST -D_FILE_OFFSET_BITS=64
```



Para compilar as ferramentas para gerar a base de dados:

```
1 make MACHINE=LINUX DATABASE=POSTGRESQL
```




## 1.2. Gerando tabelas

```
1 ./dbgen -s 1  
2 mkdir -p tpls && cd tpls && mv ../*.tbl .
```




Para subir um PostgreSQL via docker:

```
docker run --name postgres-db -e  
1 POSTGRES_PASSWORD=1234 -d -p 5432:5432  
postgres
```



Para carregar a DDL e os arquivos gerados via dbgen para o PostgreSQL:

```
1 export PGPASSWORD="1234"  
2 psql -U postgres -h localhost -c "DROP DATABASE IF  
EXISTS tpc"  
3 psql -U postgres -h localhost -c "CREATE DATABASE  
tpc"  
4 psql -U postgres -h localhost -d tpc -f dss.ddl  
5 # editar arquivo para ser compativel com PostgreSQL  
6 psql -U postgres -h localhost -d tpc -f dss.ri  
7  
8 psql -U postgres -h localhost -d tpc -c "\copy  
region FROM 'region.tbl' WITH (FORMAT csv, DELIMITER  
'|')";
```



```

9 psql -U postgres -h localhost -d tpc -c "\copy
nation FROM 'nation.tbl' WITH (FORMAT csv, DELIMITER
'|')";

10 psql -U postgres -h localhost -d tpc -c "\copy
customer FROM 'customer.tbl' WITH (FORMAT csv,
DELIMITER '|')";

11 psql -U postgres -h localhost -d tpc -c "\copy
supplier FROM 'supplier.tbl' WITH (FORMAT csv,
DELIMITER '|')";

12 psql -U postgres -h localhost -d tpc -c "\copy part
FROM 'part.tbl' WITH (FORMAT csv, DELIMITER '|')";

13 psql -U postgres -h localhost -d tpc -c "\copy
partsupp FROM 'partsupp.tbl' WITH (FORMAT csv,
DELIMITER '|')";

14 psql -U postgres -h localhost -d tpc -c "\copy
orders FROM 'orders.tbl' WITH (FORMAT csv, DELIMITER
'|')";

15 psql -U postgres -h localhost -d tpc -c "\copy
lineitem FROM 'lineitem.tbl' WITH (FORMAT csv,
DELIMITER '|')";

```

Convertendo os arquivos de tbl para csv com awk:

```

for f in *.tbl; do awk 'BEGIN {FS="|"}
1 OFS=","} {$1=$1; print}' "$f" >
"${f/.tbl/.csv}"; done

```



Para carregar os arquivos .csv para o MongoDB e preciso o mongoimport:

```

1 mongoimport --db tpch --collection region --
type csv --file region.csv --headerline
2 mongoimport --db tpch --collection nation --type csv
--file nation.csv --headerline
3 mongoimport --db tpch --collection customer --type
csv --file customer.csv --headerline
4 mongoimport --db tpch --collection supplier --type
csv --file supplier.csv --headerline

```



```

5 mongoimport --db tpch --collection part --type csv --
  file part.csv --headerline
6 mongoimport --db tpch --collection partsupp --type
  csv --file partsupp.csv --headerline
7 mongoimport --db tpch --collection orders --type csv
  --file orders.csv --headerline
8 mongoimport --db tpch --collection lineitem --type
  csv --file lineitem.csv \
9 --fields
  "L_ORDERKEY,L_PARTKEY,L_SUPPKEY,L_LINENUMBER,L_QUANTITY"

```

### 1.2.1. MongoDB Compass

Agora com todas as tabelas carregadas como collections, o MongoDB ficará com essa configuração:

The screenshot shows the MongoDB Compass interface for a database named 'tpch'. The 'Collections' tab is selected, displaying a list of collections with their respective statistics. The collections are: customer, lineitem, orders, part, partsupp, region, and supplier. Each collection entry shows its storage size, document count, average document size, number of indexes, and total index size.


Collection Name	Storage size	Documents	Avg. document size	Indexes	Total index size
customer	22.42 MB	100 K	291,000 B	1	1.02 MB
lineitem	489.28 MB	4.14	368,000 B	1	46.42 MB
orders	125.02 MB	1.91	285,000 B	1	16.10 MB
part	7.68 MB	200 K	310,000 B	1	1.91 MB
partsupp	25.31 MB	600 K	224,000 B	1	6.22 MB
region	25.48 KB	4	244,000 B	1	25.48 KB
supplier	1.25 MB	10 K	291,000 B	1	94.61 KB

## 1.3. As 3 primeiras queries TPC-H


Primeiro vamos converter as queries de PostgreSQL para MongoDB, então vamos testar em ambos os bancos.

### 1.3.1. Consultas para o PostgreSQL

1.sql

```
1  explain analyze select  SQL
2    l_returnflag,
3    l_linestatus,
4    sum(l_quantity) as sum_qty,
5    sum(l_extendedprice) as sum_base_price,
6    sum(l_extendedprice * (1 - l_discount)) as
   sum_disc_price,
7    sum(l_extendedprice * (1 - l_discount) * (1 +
   l_tax)) as sum_charge,
8    avg(l_quantity) as avg_qty,
9    avg(l_extendedprice) as avg_price,
10   avg(l_discount) as avg_disc,
11   count(*) as count_order
12 from
13   lineitem
14 where
15   l_shipdate <= date '1998-12-01' - interval ':1
   day'
16 group by
17   l_returnflag,
18   l_linestatus
19 order by
20   l_returnflag,
21   l_linestatus;
```

2.sql

```
1  explain analyze select  SQL
2    s_acctbal,
3    s_name,
4    n_name,
5    p_partkey,
```

```

6      p_mfgr,
7      s_address,
8      s_phone,
9      s_comment
10  from
11      part,
12      supplier,
13      partsupp,
14      nation,
15      region
16  where
17      p_partkey = ps_partkey
18      and s_suppkey = ps_suppkey
19      and p_size = 15
20      and p_type like '%STEEL'
21      and s_nationkey = n_nationkey
22      and n_regionkey = r_regionkey
23      and r_name = 'EUROPE'
24      and ps_supplycost = (
25          select
26              min(ps_supplycost)
27          from
28              partsupp,
29              supplier,
30              nation,
31              region
32          where
33              p_partkey = ps_partkey
34              and s_suppkey = ps_suppkey
35              and s_nationkey = n_nationkey
36              and n_regionkey = r_regionkey
37              and r_name = 'EUROPE'
38      )
39  order by

```

```

40      s_acctbal desc,
41      n_name,
42      s_name,
43      p_partkey;

```

### 3.sql

```

1  select
2      l_orderkey,
3      sum(l_extendedprice * (1 - l_discount)) as
         revenue,
4      o_orderdate,
5      o_shippriority
6  from
7      customer,
8      orders,
9      lineitem
10 where
11     c_mktsegment = ':1'
12     and c_custkey = o_custkey
13     and l_orderkey = o_orderkey
14     and o_orderdate < date ':2'
15     and l_shipdate > date ':2'
16 group by
17     l_orderkey,
18     o_orderdate,
19     o_shippriority
20 order by
21     revenue desc,
22     o_orderdate;

```

## 1.3.2. Consultas para o MongoDB:

### 1.mql

```

1  db.lineitem.aggregate([
2      {
3          $match: {
4              L_SHIPDATE: { $lte: "1998-09-02" }
5          }
6      },
7      {
8          $group: {
9              _id: {
10                 l_returnflag: "$L_RETURNFLAG",
11                 l_linestatus: "$L_LINESTATUS"
12             },
13             sum_qty:      { $sum:
14                 "$L_QUANTITY" },
15             sum_base_price: { $sum:
16                 "$L_EXTENDEDPRI" },
17             // sum(l_extendedprice * (1 -
18                 l_discount))
19             sum_disc_price: { $sum: { $multiply:
20                 [ "$L_EXTENDEDPRI", { $subtract: [ 1,
21                 "$L_DISCOUNT" ] } ] } },
22             // sum(l_extendedprice * (1 -
23                 l_discount) * (1 + l_tax))
24             sum_charge:    { $sum: { $multiply:
25                 [ "$L_EXTENDEDPRI", { $subtract: [ 1,
26                 "$L_DISCOUNT" ] } ], { $add: [ 1,
27                 "$L_TAX" ] } ] } },
28             avg_qty:      { $avg:
29                 "$L_QUANTITY" },
30             avg_price:    { $avg:
31                 "$L_EXTENDEDPRI" },
32             avg_disc:     { $avg:
33                 "$L_DISCOUNT" },
34             count_order:  { $sum: 1 } //
35             count(*)
36         }
37     },
38 ])
```



```

25     {
26         $project: {
27             _id: 0, // Oculta o campo _id padrão
28             l_returnflag: "$_id.l_returnflag",
29             l_linestatus: "$_id.l_linestatus",
30             sum_qty: "$sum_qty",
31             sum_base_price: "$sum_base_price",
32             sum_disc_price: "$sum_disc_price",
33             sum_charge: "$sum_charge",
34             avg_qty: "$avg_qty",
35             avg_price: "$avg_price",
36             avg_disc: "$avg_disc",
37             count_order: "$count_order"
38         }
39     },
40     {
41         $sort: {
42             l_returnflag: 1,
43             l_linestatus: 1
44         }
45     }
46 ])
47 .explain("executionStats")

```

## 2.mql

```

1  db.part.aggregate([
2      {
3          $match: {
4              p_size: 15,
5              p_type: { $regex: "STEEL$" }
6          }
7      },
8      {

```

mql

```

9      $lookup: {
10        from: "partsupp",
11        localField: "p_partkey",
12        foreignField: "ps_partkey",
13        as: "part_supplier_info"
14      }
15    },
16    { $unwind: "$part_supplier_info" },
17    {
18      $lookup: {
19        from: "supplier",
20        localField: "part_supplier_info.ps_suppkey",
21        foreignField: "s_suppkey",
22        as: "supplier_info"
23      }
24    },
25    { $unwind: "$supplier_info" },
26    {
27      $lookup: {
28        from: "nation",
29        localField: "supplier_info.s_nationkey",
30        foreignField: "n_nationkey",
31        as: "nation_info"
32      }
33    },
34    { $unwind: "$nation_info" },
35    {
36      $lookup: {
37        from: "region",
38        localField: "nation_info.n_regionkey",
39        foreignField: "r_regionkey",
40        as: "region_info"
41      }
42    },

```

```

43   { $unwind: "$region_info" },
44   {
45     $match: {
46       "region_info.r_name": "EUROPE"
47     }
48   },
49   {
50     $group: {
51       _id: "$p_partkey",
52       min_cost: { $min:
53         "$part_supplier_info.ps_supplycost" },
54       data: { $push: "$$ROOT" }
55     },
56     { $unwind: "$data" },
57     {
58       $match: {
59         $expr: {
60           $eq: [
61             "$data.part_supplier_info.ps_supplycost",
62             "$min_cost"
63           ]
64         }
65       }
66     },
67     {
68       $project: {
69         _id: 0,
70         s_acctbal: "$data.supplier_info.s_acctbal",
71         s_name: "$data.supplier_info.s_name",
72         n_name: "$data.nation_info.n_name",
73         p_partkey: "$data.p_partkey",
74         p_mfgr: "$data.p_mfgr",
75         s_address: "$data.supplier_info.s_address",

```

```

76     s_phone: "$data.supplier_info.s_phone",
77     s_comment: "$data.supplier_info.s_comment"
78   }
79 },
80 {
81   $sort: {
82     s_acctbal: -1,
83     n_name: 1,
84     s_name: 1,
85     p_partkey: 1
86   }
87 }
88 ]).explain("executionStats")

```

### 3.mql

```

1  db.orders.aggregate([
2    {
3      $lookup: {
4        from: "customer",
5        localField: "o_custkey",
6        foreignField: "c_custkey",
7        as: "customer_info"
8      }
9    },
10   { $unwind: "$customer_info" },
11   {
12     $match: {
13       "customer_info.c_mktsegment": "BUILDING",
14       o_orderdate: { $lt: new
15         Date("1995-03-15T00:00:00.000Z") }
16     }
17   },
18   {
19     $lookup: {

```

```

19     from: "lineitem",
20     localField: "o_orderkey",
21     foreignField: "l_orderkey",
22     as: "lineitems"
23   }
24 },
25 { $unwind: "$lineitems" },
26 {
27   $match: {
28     "lineitems.l_shipdate": { $gt: new
      Date("1995-03-15T00:00:00.000Z") }
29   }
30 },
31 {
32   $group: {
33     _id: {
34       l_orderkey: "$o_orderkey",
35       o_orderdate: "$o_orderdate",
36       o_shippriority: "$o_shippriority"
37     },
38     revenue: {
39       $sum: {
40         $multiply: [
41           "$lineitems.l_extendedprice",
42           { $subtract: [1,
            "$lineitems.l_discount"] }
43         ]
44       }
45     }
46   }
47 },
48 {
49   $sort: {
50     revenue: -1,
51     "_id.o_orderdate": 1

```

```

52     }
53   },
54   {
55     $project: {
56       _id: 0,
57       l_orderkey: "$_id.l_orderkey",
58       revenue: 1,
59       o_orderdate: "$_id.o_orderdate",
60       o_shippriority: "$_id.o_shippriority"
61     }
62   }
63 }).explain("executionStats")

```

## 2. Resultados

Todos os resultados eu deixei em um repositório publico para facilitar a leitura

<https://github.com/Fabiokleis/tpc-h>

### 2.1. Resultados Postgres

1. Primeira query com `explain analyze`:

[https://github.com/Fabiokleis/tpc-h/blob/main/1\\_postgres\\_result.md](https://github.com/Fabiokleis/tpc-h/blob/main/1_postgres_result.md)

2. Segunda query com `explain analyze`:

[https://github.com/Fabiokleis/tpc-h/blob/main/2\\_postgres\\_result.md](https://github.com/Fabiokleis/tpc-h/blob/main/2_postgres_result.md):

3. Terceira query com `explain analyze`:

[https://github.com/Fabiokleis/tpc-h/blob/main/3\\_postgres\\_result.md](https://github.com/Fabiokleis/tpc-h/blob/main/3_postgres_result.md):

## 2.2. Resultados MongoDB

1. Primeira query com explain executionStats:

[https://github.com/Fabiokleis/tpc-h/blob/main/1\\_mongodb\\_result.json](https://github.com/Fabiokleis/tpc-h/blob/main/1_mongodb_result.json):

2. Segunda query com explain executionStats:

[https://github.com/Fabiokleis/tpc-h/blob/main/2\\_mongodb\\_result.json](https://github.com/Fabiokleis/tpc-h/blob/main/2_mongodb_result.json):

3. Terceira query com explain executionStats:

[https://github.com/Fabiokleis/tpc-h/blob/main/3\\_mongodb\\_result.json](https://github.com/Fabiokleis/tpc-h/blob/main/3_mongodb_result.json):

## 2.3. PostgreSQL vs MongoDB

Query	Planning Time	Total Execution Time (ms)
1.sql	1.830 ms	3498.590 ms
2.sql	15.858 ms	2018.225 ms
3.sql	0.403 ms	1144.707 ms
1.mql	0.0 ms	7031 ms
2.mql	0.0 ms	58 ms
3.mql	0.0 ms	577 ms