
# NoSQL I7-7700 Performance Out Of The Box: Hbase, Couchbase, Mongodb

This project examines the import and SQL performance of 1K, 10K, 100K, 1M, 10M, and 100M documents using out of the box settings for Hbase, Couchbase, and MongoDB. Tests using the Setquery benchmark are run under RHEL V7 in real SSD environments. The following graph illustrates the elapsed times for CSV import and SQL queries on an I7-4700. Detailed response times for the I7-4700 and I7-7700 are provided in Appendix A.

| | hbase | cbase | mongo | hbase | cbase | mongo |
|---|---|---|---|---|---|---|
| # documents | import | import | import | sql | sql | sql |
| 1K | 8 | 55 | 52 | 411 | 2 | 0.227 |
| 10K | 13 | 73 | 54 | 421 | 16 | 0.744 |
| 100K | 25 | 81 | 63 | 453 | 151 | 7 |
| 1M | 89 | 96 | 78 | 840 | 1529 | 73 |
| 10M | 804 | 540 | 613 | 5,100 | 21,854 | 828 |
| 100M | 7,920 | 5,844 | 10,920 | 49,320 | | 12,243 |

**Figure 1 – Import and Query Elapsed Seconds**

Primary index creation is included in the Couchbase import times, creating the default index and 11 secondary indices are included in the Mongodb import times. Couchbase query elapsed times for the 100,000,000 document collection average over an hour and are not shown in Figure 1.

**IMPORTING DATA**

Following is the Vertica DDL describing the document:

```
CREATE TABLE SQ1
( KPART     INTEGER  NOT NULL,
  KSEQ      INTEGER  NOT NULL,
  K1B       INTEGER  NOT NULL,
  K500M     INTEGER  NOT NULL,
  K100M     INTEGER  NOT NULL,
  K10M      INTEGER  NOT NULL,
  K5M       INTEGER  NOT NULL,
  K2M       INTEGER  NOT NULL,
  K1M       INTEGER  NOT NULL,
  K500K     INTEGER  NOT NULL,
  K250K     INTEGER  NOT NULL,
  K100K     INTEGER  NOT NULL,
  K40K      INTEGER  NOT NULL,
  K10K      INTEGER  NOT NULL,
  K1K       INTEGER  NOT NULL,
  K100      INTEGER  NOT NULL,
  K25       INTEGER  NOT NULL,
  K10       INTEGER  NOT NULL,
  K5        INTEGER  NOT NULL,
  K4        INTEGER  NOT NULL,
  K2        INTEGER  NOT NULL,
  SORTBIN   INTEGER  NOT NULL,
  SORTPACK DECIMAL (8, 0)  NOT NULL,
  SORTCHAR CHARACTER (8)  NOT NULL,
  S4        CHARACTER (30)  NOT NULL,
  S5        CHARACTER (30)  NOT NULL,
  S6        CHARACTER (30)  NOT NULL,
  S7        CHARACTER (20)  NOT NULL,
  S8        CHARACTER (22)  NOT NULL)
  partition by KPART;
alter table SQ1
       add constraint pk primary key (KPART, KSEQ);
```

**Figure 2 – Document DDL (Vertica)**

The following graphs illustrate disk (SSD) utilization while importing 10,000,000 documents. Click on the hyperlink below the graph to retrieve the entire NMON spreadsheet analysis.

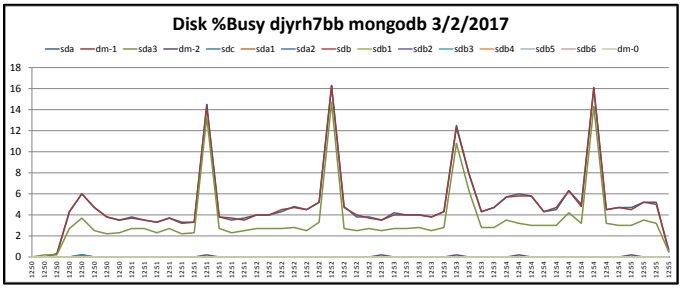


**Figure 3 – Mongodb 10M Document (Import Disk Busy SSD)**

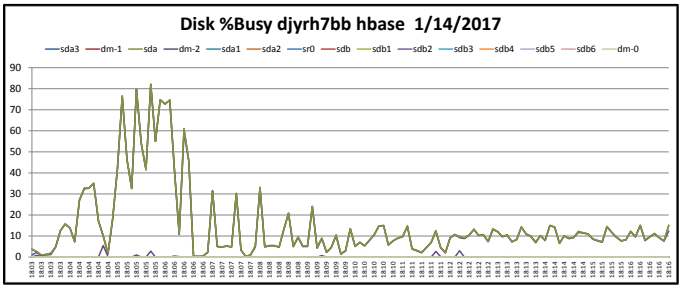


**Figure 4 – Hbase 10M Document Import Disk Busy (SSD)**

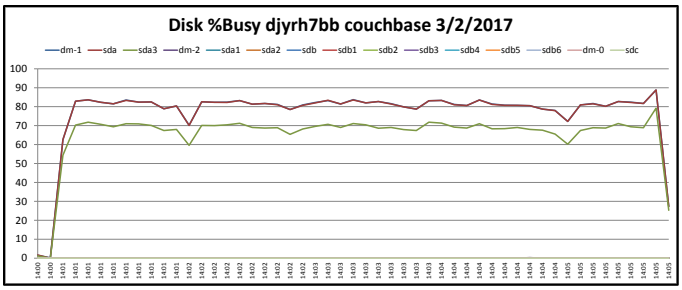


**Figure 5 – Couchbase 10M Document Import Disk Busy (SSD)**

The following graphs illustrate CPU utilization while importing 10,000,000 documents in the REAL/SSD setups. Couchbase is again the busiest for CPU as well as DISK. The memory utilizations are the same: all available RAM is consumed.

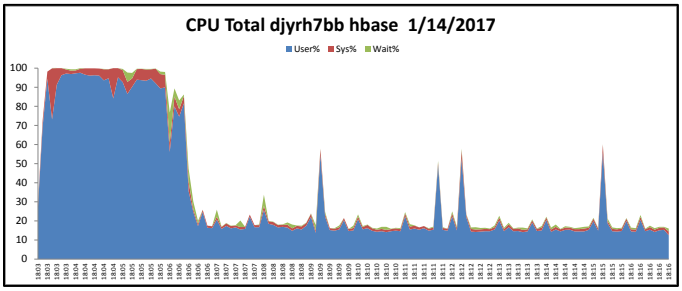


**Figure 6 – Hbase 10M Document Import CPU Busy (SSD)**

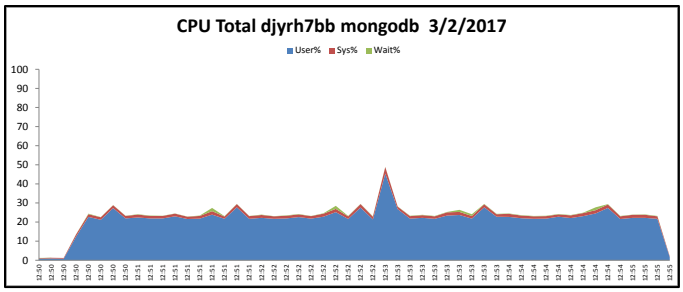


**Figure 7 – Mongodb 10M Document Import CPU Busy (SSD)**

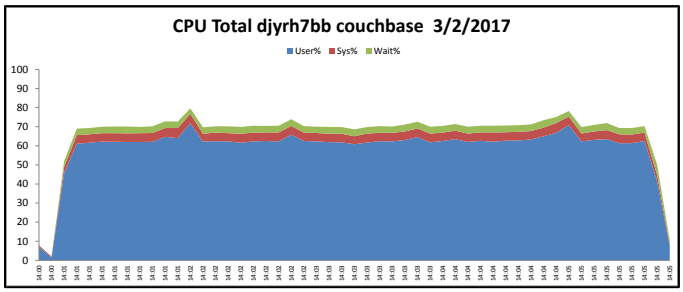


**Figure 8 – Couchbase 10M Document Import CPU Busy (SSD)**

The test data generation program (genhbase.cpp) is in Appendix C. For data and column distribution see http://davidjyoung.com/cmg/pcfverticaonaws.pdf page 15.

The following table illustrates VIOPERF numbers for three disks: The internal 2TB SSD, the external usb3 SSD and the external usb3 2TB rbs volume.

| VIOPERF | write mb/s | read/write mb/s | read mb/s | seeks/s |
|---|---|---|---|---|
| internal SSD | 506 | 227/227 | 523 | 44389 |
| usb3 SSD | 236 | 102/102 | 255 | 6092 |
| usb3 rbs | 31 | 39/39 | 104 | 842 |

**Figure 9 – VIOPERF disk performance**

Part of the reason for the pronounced CPU and DISK utilization in the out-of-the-box Couchbase environment is smaller blocksizes (~25K) compared to out-of-the-box Hbase and Mongodb (~250K) environments, as shown in the following graphs.

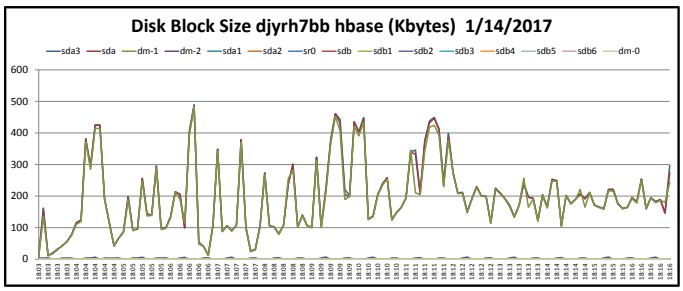


**Figure 10 – Hbase 10M Document Import blksize (SSD)**

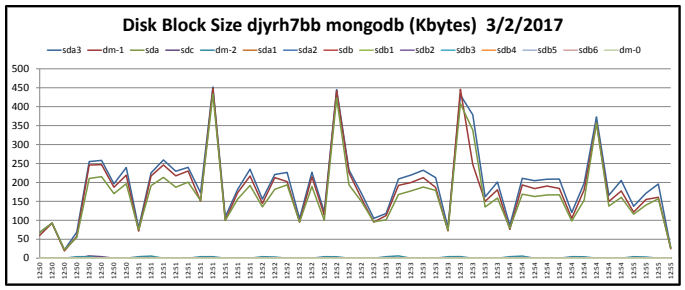


**Figure 11 – Mongodb 10M Document Import blksize (SSD)**

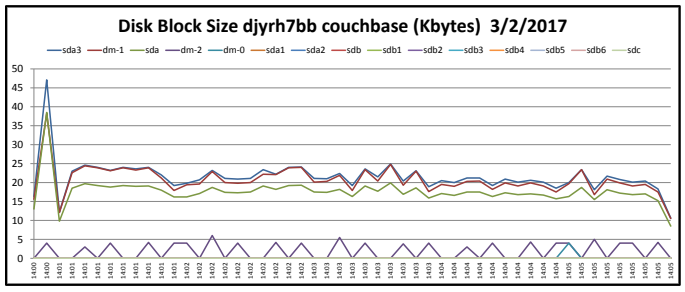


**Figure 12 – Couchbase 10M Document Import blksize (SSD)**

The following link has an excellent high level description of Mongodb, Couchbase, and Hbase: http://db-engines.com/en/system/Couchbase%3BHBase%3BMongoDB

**HBASE SQL**

Hortonworks HDP 2.5 was freshly installed for these experiments. Following is the content of the first two records in the csv import file: The first record is the standard header with column names, the second record is data.

```
hbase_row_key,kpart,kseq,k1b,k500m,k100m,k10m,k5m,k2
m,k1m,k500k,k250k,k100k,k40k,k10k,k1k,k100,k25,k10,k
5,k4,k2,sortbin,sortpack,sortchar,s4,s5,s6,s7,s8

0,1,0,956273372,478136686,95627337,9562733,4781366,1
912546,956273,478136,239068,95627,38250,9562,956,95,
23,9,4,3,2,12345678,12345678,12345678,123456789012345
6789012345678901,12345678901234567890123456789012345
6789012345678901234567890,12345678901234567890123456
789012345678901234567890,12345678901234567890,123456
789012345678901
```

**Figure 13 – CSV Header and Document Records**

The values for hbase_row_key and kseq are the same. Following is the table definition with one column family per field, placing each field in its own file. Multiple column families marginally increase disk space, CPU, and ELAPSED import times, but also significantly reduce CPU and ELAPSED times for big SQL queries, as will be discussed. Hbase is not built for query processing, but it can be done with minimal effort: multiple column families improve query performance.

```
create
'sql1k','colfam01','colfam02','colfam03','colfam04',
'colfam05','colfam06','colfam07','colfam08','colfam0
9','colfam10','colfam11','colfam12','colfam13','colf
am14','colfam15','colfam16','colfam17','colfam18','c
olfam19','colfam20','colfam21','colfam22','colfam23'
,'colfam24','colfam25','colfam26','colfam27','colfam
28','colfam29'
```

**Figure 14 – Hbase table DDL, one column family per field**

The ImportTsv utility is then invoked to Map/Reduce the csv data into an incremental load format, which is subsequently loaded.

```
hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -
Dimporttsv.separator=, -
Dimporttsv.bulk.output=hdfs://djyrh7bb:/tmp/sq11kbul
k   -
Dimporttsv.columns="HBASE_ROW_KEY,colfam01:kpart,col
fam02:kseq,colfam03:k1b,colfam04:k500m,colfam05:k100
m,colfam06:k10m,colfam07:k5m,colfam08:k2m,colfam09:k
1m,colfam10:k500k,colfam11:k250k,colfam12:k100k,colf
am13:k40k,colfam14:k10k,colfam15:k1k,colfam16:k100,c
olfam17:k25,colfam18:k10,colfam19:k5,colfam20:k4,col
fam21:k2,colfam22:sortbin,colfam23:sortpack,colfam24
:sortchar,colfam25:s4,colfam26:s5,colfam27:s6,colfam
28:s7,colfam29:s8" sq11k
hdfs://djyrh7bb:/tmp/genhbase.1K.p01.20170114.csv

hbase
org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFi
les hdfs://djyrh7bb:/tmp/sq11kbulk sq11k
```

**Figure 15 – Hbase bulk load**

The following statement creates an external reference from Hive to the Hbase table, which allows the processing of standard SQL statements. For these out-of-the-box HBASE tests all fields default to string format, the HIVE external table reflects the string format.

```
CREATE EXTERNAL TABLE sq11k(key string,kpart
string,kseq string,k1b string,k500m string,k100m
string,k10m string,k5m string,k2m string,k1m
string,k500k string,k250k string,k100k string,k40k
string,k10k string,k1k string,k100 string,k25
string,k10 string,k5 string,k4 string,k2
string,sortbin string,sortpack string,sortchar
string,s4 string,s5 string,s6 string,s7 string,s8
string) STORED BY
'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ("hbase.columns.mapping" =
"colfam01:kpart,colfam02:kseq,colfam03:k1b,colfam04:
k500m,colfam05:k100m,colfam06:k10m,colfam07:k5m,colf
am08:k2m,colfam09:k1m,colfam10:k500k,colfam11:k250k,
colfam12:k100k,colfam13:k40k,colfam14:k10k,colfam15:
k1k,colfam16:k100,colfam17:k25,colfam18:k10,colfam19
:k5,colfam20:k4,colfam21:k2,colfam22:sortbin,colfam2
3:sortpack,colfam24:sortchar,colfam25:s4,colfam26:s5
,colfam27:s6,colfam28:s7,colfam29:s8")
TBLPROPERTIES("hbase.table.name" = "sq11k");
```

**Figure 16 – Hive Create External Table**

Following are Hbase Shell scan statements against 100M document tables, one with multiple column families (sq100m) which takes 64 seconds, the other with a single column family (sq100mb) which takes 786 seconds. These results are on the SSD, imagine what the Round Brown and Spinning disk times would be.


**Figure 17 – Hbase 100M Document Scans**

The following table contains Hadoop Job counters for the 1M, 10M, and 100M document imports: Notice the garbage collection times.

| NUMBER OF DOCUMENTS | 1,000,000 | 10,000,000 | 100,000,000 |
|---|---|---|---|
| elapsed milliseconds | 89,000 | 804,000 | 7,920,000 |
| CPU_MILLISECONDS | 171,390 | 1,810,050 | 17,800,990 |
| GC_TIME_MILLIS | 6,004 | 169,488 | 1,831,750 |
| HDFS_BYTES_READ | 292,403,530 | 2,934,144,946 | 29,442,687,681 |
| HDFS_BYTES_WRITTEN | 1,616,611,032 | 15,171,707,030 | 161,815,938,228 |
| COMMITTED_HEAP_BYTES | 6,713,690,624 | 38,330,695,680 | 378,324,123,648 |
| VIRTUAL_MEMORY_BYTES | 16,691,380,224 | 87,476,477,932 | 823,630,876,672 |

**Figure 18 – Hbase Hadoop Job Counters**

All 71 SETQUERY queries (Appendix C) were then run from the Hive CLI, the following table shows elapsed times for csv import and SQL:

| # documents | import elapsed seconds | SQL elapsed seconds |
|---|---|---|
| 1,000 | 8 | 411 |
| 10,000 | 13 | 421 |
| 100,000 | 25 | 453 |
| 1,000,000 | 89 | 840 |
| 10,000,000 | 804 | 5100 |
| 100,000,000 | 7,920 | 49,320 |

**Figure 19 – Hbase SQL Elapsed Seconds**

The following settings were changed from their defaults for the HBASE measurements:

- hbase.client.write.buffer 8,388,608
- hbase.client.scanner.caching 50,000
- dfs.blocksize 134,217,728
- hbase.scan.cache 10,000

**Couchbase and Mongodb**

The last 10 of the 71 Setquery queries are self-referencing joins on non-primary keys: This is not supported in Couchbase or Mongodb. These queries have been dropped from the Couchbase and Mongodb tests temporarily, workarounds are in progress.

In addition to the I7-4700 measurements, Couchbase and Mongo are also measured on an I7-7700. For this workload and implementation, Couchbase queries take considerably more elapsed time and hardware resources than Mongo queries, your mileage may vary. This area continues to be under investigation; any updates will be promptly reported. Only 5 Couchbase 100M Document queries are included, due to elapsed times of over an hour per query.

| | m 4700 | m 7700 | cb 4700 | cb 7700 |
|---|---|---|---|---|
| 1K | 0.23 | 0.06 | 2.49 | 1.70 |
| 10K | 0.74 | 0.60 | 16.08 | 10.96 |
| 100K | 6.70 | 6.04 | 151 | 103 |
| 1M | 73 | 62 | 1529 | 1029 |
| 10M | 828 | 617 | 21854 | 15338 |
| 100M | 12243 | 9705 | | |

**Figure 20 – SQL Elapsed Times I7-4700 & I7-7700**

The following shows CPU time and Cycles Per Instruction for the I7-7700 as reported by Vtune Advanced Hotspots for all 61 queries in mongo/secondary indexes, couchbase/primary index, and couchbase/TO_NUMBER secondary indexes setups.

| | m cpu | m cpi | cb cpu | cb cpi | cbnum cpu | cbnum cpi |
|---|---|---|---|---|---|---|
| 1K | 8 | 0.768 | 33 | 0.871 | 24.8 | 0.875 |
| 10K | 8 | 0.768 | 74 | 0.865 | 42.4 | 0.891 |
| 100K | 18 | 0.877 | 510 | 0.853 | 173 | 0.896 |
| 1M | 104 | 0.996 | 4849 | 0.839 | 1374 | 0.896 |
| 10M | 1101 | 0.981 | | | | |
| 100M | 15809 | 0.701 | | | | |

**Figure 21 – Vtune CPU and Cycles Per Instruction I7-7700**

SQL response and CPU usage improve substantially using the TO_NUMBER indices, as shown in the "cbnum cpu" column preceding. Detailed response times for the TO_NUMBER indices (I7-4700) are shown upcoming in Figure 32.

https://forums.couchbase.com/t/tonumber-type-conversion-function-performance-issues/8321/9

The big jump in CPU for the Mongo 100M document test is due mainly to Snappy:InternalUncompress (3,474 seconds of CPU). The larger collections are too big to fit in memory, causing increased disk I/O and CPU (file decompression) per SQL compared to smaller collections.

**COUCHBASE**

The Couchbase install was easy and quick; see the following https://developer.couchbase.com/documentation/server/4.6/install/install-linux.html The Couchbase Enterprise Edition Cluster Overview, illustrating RAM and Disk allocations is shown in Figure 22. Total disk space for the six collections with primary index is approximately 65 gigabytes.

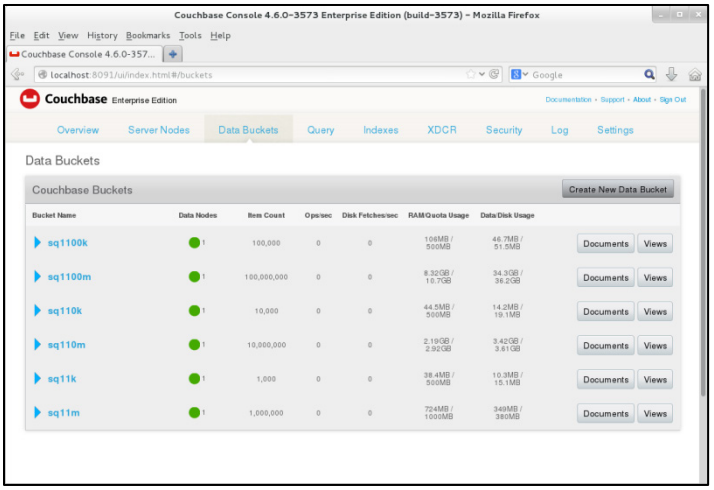


**Figure 22 – Couchbase Enterprise Cluster Overview**

The following control statement was used to import data into Couchbase, specifying bucket sq11m, csv filename, and 8 threads (-t 8) for importing.

```
[root@localhost bin]# ./cbimport csv -c
couchbase://127.0.0.1 -u Administrator -p password
-b sq11m -d
file:///home/setquery/genhbase.1M.p01.20170228.csv -
g key::%hbase_row_key% -t 8
```

**Figure 23 – Couchbase Cbimport Control Statement**


**Figure 24 – Couchbase Import 100M Documents Statistics**

The Couchbase schema is shown following, just like HBASE the fields are all string format:

```
sq1m
        hbase_row_key (string)
        k10 (string)
        k100 (string)
        .
        .
        .

        sortpack (string)
```

**Figure 25 – Couchbase Document Schema**

**COUCHBASE INDEXES**

A primary index is created before SQL processing:

```
CREATE PRIMARY INDEX `sq1100m-primary-index` ON
`sq1100m` USING GSI;
```

**Figure 26 – Couchbase Create Primary Index**

**SQL**

The Couchbase N1QL language is very similar to standard SQL, changes are basically cosmetic for this workload. Following are some examples.

For this workload and implementation, all fields are strings, hence the numeric values need to be quoted, else the query will not return what is expected. The SQL at top is correct, the one at the bottom is incorrect.

```
cbq> select count(*) from sq11k where k2 = "2";
{
    "results":
            "$1": 291
    "status": "success",
    }
cbq> select count(*) from sq11k where k2 = 2;
{
    "results":
            "$1": 0
    "status": "success",
```

**Figure 27 – Couchbase N1QL example 1**

Likewise the SQL at the top returns what is expected.

```
cbq> select sum(TO_NUMBER(k1k)) from sq11k where k10
= "3";
{
    "results":
            "$1": 31032
    "status": "success",

cbq> select sum(k1k) from sq11k where k10 = "3";
{
    "results":
            "$1": null
    "status": "success",
```

**Figure 28 – Couchbase N1QL example 2**

In an attempt to improve response times, the following indices were created on the 1K, 10K, 100K, 1M, and 10M document buckets:

```
CREATE INDEX `k250k-num` ON
`sq11knum`(TO_NUMBER(`k250k`));
CREATE INDEX `k100k-num` ON
`sq11knum`(TO_NUMBER(`k100k`));
CREATE INDEX `k10k-num` ON
`sq11knum`(TO_NUMBER(`k10k`));
CREATE INDEX `k1k-num` ON
`sq11knum`(TO_NUMBER(`k1k`));
CREATE INDEX `k100-num` ON
`sq11knum`(TO_NUMBER(`k100`));
CREATE INDEX `k25-num` ON
`sq11knum`(TO_NUMBER(`k25`));
CREATE INDEX `k10-num` ON
`sq11knum`(TO_NUMBER(`k10`));
CREATE INDEX `k5-num` ON
`sq11knum`(TO_NUMBER(`k5`));
CREATE INDEX `k4-num` ON
`sq11knum`(TO_NUMBER(`k4`));
CREATE INDEX `k2-num` ON
`sq11knum`(TO_NUMBER(`k2`));
CREATE INDEX `kseq-num` ON
`sq11knum`(TO_NUMBER(`kseq`));
```

**Figure 29 – Couchbase TO_NUMBER Indices DDL**

The SQL statements were adjusted also, see following:

- `q101 select count(*) from sq11knum where TO_NUMBER(k250k) = 2;`
- `q201 select count(*) from sq11knum where TO_NUMBER(k2) = 2 and TO_NUMBER(kseq) = 3;`
- `q301 select sum(TO_NUMBER(k1k)) from sq11knum where TO_NUMBER(k100k) = 3;`
- `q401 select kseq, k500k from sq11knum where TO_NUMBER(k2) = 1 and TO_NUMBER(k100) > 80 and TO_NUMBER(k10k) between 2000 and 3000;`
- `q501 select k2, k100, count(*) from sq11knum group by k2, k100;`

**Figure 30 – Couchbase TO_NUMBER Indices SQL**

Elapsed index create times and disk space.

| **seconds** | 1k | 10k | 100k | 1m | 10m |
|---|---|---|---|---|---|
| k250k | 1.81 | 1.83 | 2.9 | 33.26 | 476 |
| k100k | 2.64 | 2.93 | 3.14 | 33.71 | 461 |
| k10k | 2.61 | 2.63 | 3.29 | 33.35 | 499 |
| k1k | 2.61 | 2.62 | 3.13 | 32.68 | 472 |
| k100 | 2.62 | 2.63 | 3.04 | 31.73 | 463 |
| k25 | 2.67 | 2.61 | 3.1 | 31.08 | 453 |
| k10 | 2.62 | 2.63 | 3.07 | 30.08 | 450 |
| k5 | 2.63 | 2.63 | 3.11 | 30.94 | 454 |
| k4 | 2.62 | 2.62 | 3.1 | 31.21 | 458 |
| k2 | 2.61 | 2.62 | 3.11 | 30.78 | 445 |
| kseq | 2.64 | 2.65 | 3.08 | 30.35 | 445 |
| primary | 2.62 | 2.63 | 2.78 | 17.79 | 225 |
| total | 30.7 | 31.03 | 36.85 | 366.96 | 5301 |
| | | | | | |
| **megabytes** | 1k | 10k | 100k | 1m | 10m |
| data | 16 | 20 | 54 | 404 | 3800 |
| k250k | 0.34 | 2.5 | 24 | 345 | 3000 |
| k100k | 0.34 | 2.5 | 24 | 354 | 3000 |
| k10k | 0.34 | 2.6 | 24 | 334 | 2900 |
| k1k | 0.37 | 2.5 | 23 | 341 | 2800 |
| k100 | 0.35 | 2.5 | 24 | 321 | 3000 |
| k25 | 0.32 | 2.4 | 22 | 322 | 2700 |
| k10 | 0.34 | 2.4 | 22 | 321 | 2800 |
| k5 | 0.34 | 2.3 | 22 | 324 | 2800 |
| k4 | 0.33 | 2.3 | 22 | 320 | 2900 |
| k2 | 0.33 | 2.2 | 21 | 324 | 2700 |
| kseq | 0.3 | 2.3 | 22 | 333 | 2900 |
| primary | 0.16 | 0.91 | 7.8 | 114 | 1200 |
| total | 19.86 | 47.41 | 311.8 | 4157 | 36500 |

**Figure 31 – Couchbase TO_NUMBER Indices Statistics**

| Query | 1K | 1knum | 10K | 10knum | 100K | 100knum | 1M | 1mnum |
|---|---|---|---|---|---|---|---|---|
| q101 | 129 | 6 | 198 | 4 | 1,960 | 8 | 18,540 | 5 |
| q102 | 28 | 4 | 184 | 6 | 1,760 | 3 | 17,970 | 4 |
| q103 | 30 | 3 | 212 | 4 | 1,700 | 4 | 17,980 | 5 |
| q104 | 31 | 5 | 183 | 3 | 1,780 | 4 | 18,180 | 14 |
| q105 | 29 | 5 | 207 | 4 | 1,850 | 9 | 17,900 | 47 |
| q106 | 31 | 5 | 190 | 5 | 1,810 | 22 | 18,250 | 156 |
| q107 | 30 | 5 | 180 | 9 | 1,730 | 35 | 18,060 | 370 |
| q108 | 30 | 4 | 188 | 9 | 1,780 | 66 | 18,710 | 659 |
| q109 | 30 | 4 | 189 | 13 | 1,940 | 77 | 18,700 | 830 |
| q110 | 28 | 6 | 189 | 12 | 1,760 | 92 | 18,660 | 968 |
| q201 | 35 | 7 | 191 | 6 | 1,920 | 10 | 19,430 | 6 |
| q202 | 30 | 6 | 189 | 5 | 1,810 | 5 | 18,940 | 8 |
| q203 | 34 | 4 | 199 | 4 | 1,830 | 5 | 19,050 | 18 |
| q204 | 34 | 7 | 201 | 6 | 1,840 | 12 | 19,210 | 48 |
| q205 | 33 | 7 | 193 | 14 | 1,820 | 68 | 19,180 | 475 |
| q206 | 32 | 9 | 194 | 23 | 2,120 | 189 | 19,030 | 1,701 |
| q207 | 32 | 11 | 212 | 38 | 1,830 | 456 | 18,920 | 4,869 |
| q208 | 34 | 14 | 195 | 107 | 1,830 | 959 | 19,290 | 9,450 |
| q209 | 39 | 15 | 190 | 115 | 1,830 | 825 | 19,240 | 8,363 |
| q210 | 38 | 23 | 207 | 127 | 2,090 | 1,492 | 20,220 | 15,036 |
| q211 | 33 | 19 | 192 | 184 | 1,930 | 1,409 | 19,540 | 14,889 |
| q212 | 36 | 21 | 201 | 158 | 1,830 | 1,535 | 19,510 | 15,744 |
| q213 | 33 | 22 | 332 | 155 | 1,880 | 1,503 | 19,390 | 15,533 |
| q214 | 35 | 20 | 194 | 160 | 2,070 | 1,718 | 19,460 | 15,440 |
| q215 | 33 | 23 | 205 | 162 | 1,850 | 1,665 | 19,570 | 16,264 |
| q216 | 33 | 26 | 198 | 202 | 1,920 | 1,659 | 19,480 | 19,059 |
| q217 | 37 | 19 | 202 | 151 | 2,060 | 1,515 | 19,680 | 15,587 |
| q218 | 34 | 24 | 216 | 163 | 1,830 | 1,587 | 19,470 | 16,046 |
| q301 | 30 | 6 | 180 | 4 | 1,710 | 5 | 17,990 | 5,023 |
| q302 | 31 | 6 | 176 | 6 | 1,700 | 14 | 17,960 | 11,276 |
| q303 | 27 | 6 | 184 | 4 | 1,930 | 5 | 18,190 | 13,106 |
| q304 | 30 | 6 | 176 | 8 | 1,750 | 37 | 18,000 | 433 |
| q305 | 31 | 8 | 189 | 24 | 1,770 | 213 | 18,340 | 1,844 |
| q306 | 29 | 10 | 183 | 44 | 1,740 | 471 | 18,380 | 4,876 |
| q307 | 32 | 15 | 187 | 113 | 1,980 | 943 | 19,220 | 9,745 |
| q308 | 32 | 15 | 188 | 122 | 1,830 | 1,086 | 19,050 | 12,100 |
| q309 | 62 | 6 | 495 | 8 | 4,650 | 9 | 47,520 | 11 |
| q310 | 64 | 8 | 489 | 7 | 4,830 | 13 | 48,050 | 37 |
| q311 | 66 | 13 | 504 | 7 | 4,800 | 22 | 48,600 | 82 |
| q312 | 60 | 7 | 484 | 17 | 4,740 | 142 | 47,530 | 3,424 |
| q313 | 64 | 10 | 496 | 39 | 4,860 | 868 | 49,510 | 8,068 |
| q314 | 60 | 18 | 486 | 134 | 4,780 | 1,543 | 47,290 | 13,288 |
| q315 | 63 | 23 | 503 | 373 | 4,970 | 3,194 | 50,120 | 13,710 |
| q316 | 65 | 28 | 501 | 271 | 4,860 | 4,227 | 48,500 | 13,094 |
| q401 | 37 | 11 | 213 | 56 | 1,990 | 817 | 20,470 | 6,905 |
| q402 | 40 | 9 | 216 | 68 | 2,020 | 550 | 21,060 | 6,893 |
| q403 | 42 | 8 | 288 | 44 | 2,800 | 509 | 27,750 | 5,546 |
| q404 | 47 | 10 | 305 | 43 | 2,770 | 701 | 28,500 | 5,317 |
| q405 | 42 | 10 | 268 | 42 | 2,570 | 615 | 26,430 | 4,876 |
| q406 | 38 | 9 | 301 | 52 | 2,080 | 626 | 21,210 | 5,621 |
| q407 | 34 | 9 | 208 | 62 | 2,040 | 692 | 19,720 | 7,015 |
| q408 | 35 | 8 | 220 | 48 | 1,990 | 556 | 20,880 | 5,964 |
| q409 | 48 | 26 | 475 | 58 | 3,580 | 964 | 36,370 | 10,262 |
| q410 | 52 | 12 | 379 | 52 | 3,740 | 1,283 | 37,080 | 10,897 |
| q411 | 50 | 14 | 392 | 125 | 3,630 | 796 | 37,010 | 14,001 |
| q412 | 53 | 13 | 383 | 137 | 3,830 | 1,418 | 37,580 | 11,369 |
| q413 | 50 | 7 | 361 | 8 | 3,420 | 6,788 | 34,680 | 6,396 |
| q414 | 51 | 6 | 322 | 5 | 2,900 | 8,596 | 29,730 | 5,140 |
| q501 | 39 | 39 | 213 | 228 | 2,180 | 2,163 | 21,000 | 22,617 |
| q502 | 38 | 41 | 266 | 238 | 2,280 | 2,283 | 22,750 | 24,116 |
| q503 | 41 | 38 | 216 | 216 | 2,050 | 2,132 | 20,650 | 22,541 |
| total ms | 2,494 | 780 | 16,078 | 4,517 | 151,130 | 61,212 | 1,528,680 | 457,186 |
| total sec | 2.49 | 1 | 16.08 | 5 | 151.13 | 61 | 1,528.68 | 457 |

**Figure 32 – I7-4700 SQL Response TO_NUMBER Indices**

Figure 32 compares response times using the primary index versus using the TO_NUMBER indices. Response times are significantly better using the TO_NUMBER indices versus using the primary index for 1K, 10K, 100K, and 1M document collections. Index timeout scans began about transaction Q209 when using the TO_NUMBER indices for the 10M document collection. Increasing the timeout scan threshold allowed the queries to finish, but their response times were greater than those achieved using the primary index alone.

**Single and Multiple users – I7-4700 vs. I7-7700**

The following tables summarize one and eight user performance: sq1100k for Couchbase, sq110m for Mongo. ETR is the number of transactions per hour. ITR factors in leftover CPU to project a theoretical maximum transaction rate.

| | couchbase 1 user sq1100k | couchbase 8 user sq1100k | mongodb 1 user sq110m | mongodb 8 user sq110m |
|---|---|---|---|---|
| **start** | 14:01:01 | 14:05:00 | 15:10:00 | 15:24:00 |
| **stop** | 14:03:41 | 14:18:08 | 15:21:00 | 15:39:25 |
| **elapsed** | 0:02:40 | 0:13:08 | 0:11:00 | 0:15:25 |
| **elapsed** | 160 | 788 | 660 | 925 |
| **tran** | 61 | 488 | 61 | 488 |
| **etr/hour** | 1373 | 2229 | 333 | 1899 |
| **cpu busy** | 0.472 | 0.9 | 0.113 | 0.932 |
| **itr/hour** | 2908 | 2477 | 2944 | 2038 |

**Figure 33 – I7-4700 Single and Multiple users**

| | couchbase 1 user sq1100k | couchbase 8 user sq1100k | mongodb 1 user sq110m | mongodb 8 user sq110m |
|---|---|---|---|---|
| **start** | 7:30:00 | 7:37:00 | 8:11:33 | 8:25:00 |
| **stop** | 7:31:44 | 7:45:05 | 8:21:50 | 8:35:45 |
| **elapsed** | 0:01:44 | 0:08:05 | 0:10:17 | 0:10:45 |
| **elapsed** | 104 | 485 | 617 | 645 |
| **tran** | 61 | 488 | 61 | 488 |
| **etr/hour** | 2112 | 3622 | 356 | 2724 |
| **cpu busy** | 0.444 | 0.883 | 0.116 | 0.93 |
| **itr/hour** | 4756 | 4102 | 3068 | 2929 |
| **7700/4700** | 1.64 | 1.66 | 1.04 | 1.44 |

**Figure 34 – I7-7700 Single and Multiple users**

In a nutshell, for this workload the ASUS-G11CD-K (I7-7700@3.60GHZ, DDR4@2.133GHZ) performs between 44% and 66% more work, at roughly 2/3 the response time of the ASUS-G750JM (I7-4700HQ@2.40GHZ, DDR3@0.799GHZ).

| | vcpuperf | bogomips | gb single | gb multiple | coremark |
|---|---|---|---|---|---|
| **4700** | 9.18 | 4789 | 3620 | 11722 | 17497 |
| **7700** | 7.5 | 7197 | 4952 | 16446 | 21953 |
| **7700/4700** | 0.82 | 1.50 | 1.37 | 1.40 | 1.25 |
| **1/(7700/4700)** | 1.22 | | | | |

**Figure 35 – CPU Kernels I7-4700 vs I7-7700**

**Mongodb**

The Mongodb and Compass installs were also uneventful, see https://docs.mongodb.com/manual/administration/install-on-linux/ and https://docs.mongodb.com/compass/master/install/# The Community edition was installed and tested. The following control statement was used to import data into Mongodb:

```
mongoimport -d mydb -c things100M --type csv --file
genhbase.100M.p01.20170228.csv --headerline -j 16
```

**Figure 36 – MongoDB Import 16 concurrent insert workers**

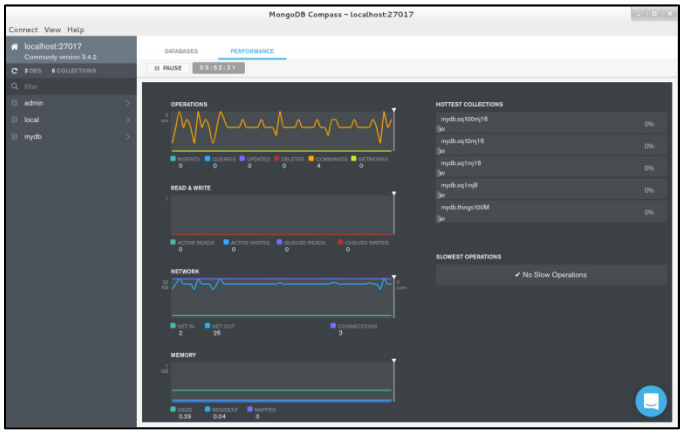


**Figure 37 – MongoDB Performance Statistics**

The MongoDB databases and storage sizes are shown in Figure 38, details for the mydb database are shown in Figure 39. Total space for the six collections with secondary indexes is approximately 23 gigabytes.

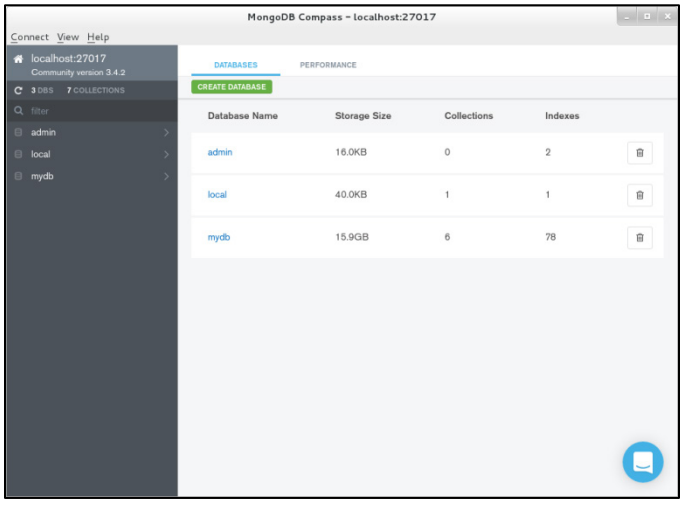


**Figure 38 – MongoDB Databases**

The MongoDB scripts are packaged into a .js file (see Appendix B), and invoked from the command line as follows:

```
mongo < sq11k.mongo.queries.js > sq11k.results.txt
```

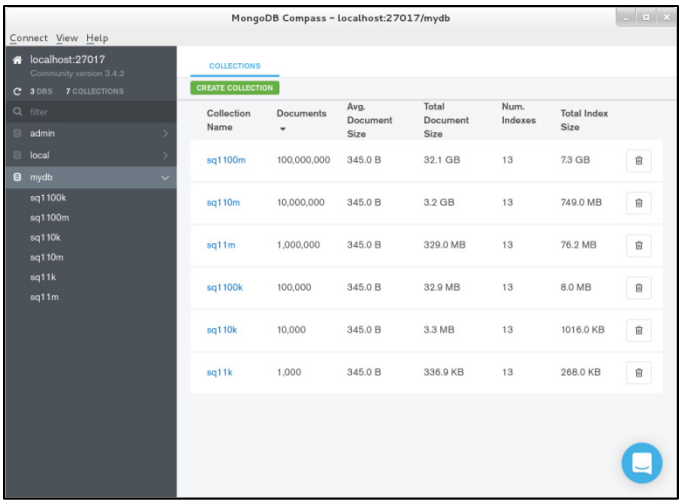


**Figure 39 – MongoDB mydb Collections**

Column statistics and field types are shown in Figure 29. MongoDB out-of-the-box chooses to store all the Document fields as numeric, either Int32 or Double.

The following is a snip from a JSON file containing 1,000 sample documents, showing the two numeric field types (Int32(k10), Double(s4)) used in the schema, along with sampled field content:

```
"name": "k10",
      "path": "k10",
          "name": "Int32",
          "bsonType": "Int32",
.
.

      "name": "s4",
        {
          "name": "Double",
          "bsonType": "Double",
```

**Figure 40 – MongoDB OOTB Schema**

**INDEXES**

Indexes were created on the following fields.

```
K250K
K100K
K40K
K10K
K1K
K100
K25
K10
K5
K4
K2
KSEQ unique
KPART,KSEQ unique
```

**Figure 41 – MongoDB Secondary Indexes**

**SQL**

The following nomenclature examples show the HBASE/Vertica SQL, the MongoDB SQL, and the Couchbase SQL.

```
q101
SELECT COUNT(*) FROM SQ1 WHERE K250K = 2;
print("q101");
db.things.find({ k250k: 2 }).count();
# q101
select count(*) from sq11k where k250k = "2";
```

**Figure 42 – Q101 Nomenclature**

```
q210
SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND KSEQ <> 3;
print("q210")
db.things.find({ k2: 2, kseq: { $ne: 3} }).count();
# q210
select count(*) from sq11k where k2 = "2" and kseq
<> "3";
```

**Figure 43 – Q210 Nomenclature**

```
q301
SELECT SUM(K1K) FROM SQ1 WHERE K100K = 3;
print("q301")
db.things.aggregate({ $match: {     $and: [
{ k100k: 3}     ] } }, { $group: { _id : null, sum :
{ $sum: "$k1k" } } });
# q301
select sum(TO_NUMBER(k1k)) from sq11k where k100k =
"3";
```

**Figure 44 – Q301 Nomenclature**

```
q403
SELECT KSEQ, K500K FROM SQ1
WHERE K10K BETWEEN 2000 AND 3000
AND K5 = 3 AND
(K25 = 11 OR K25 = 9);
print("q403")
db.things.find(     { k5: 3,  k10k: { $gt: 2000},
k10k: { $lt: 3000},
     $and: [     { $or : [ {k25: 11} , {  k25: 9 }
]}   ]},
     { kseq: 1, k500k: 1, _id: 0 } )
# q403
select kseq, k500k from sq11k where k10k between
"2000" and "3000"
and k5 ="3"
and (k25 = "11" or k25 = "9");
```

**Figure 45 – Q403 Nomenclature**

```
q501
SELECT K2, K100, COUNT(*) FROM SQ1 GROUP BY K2,
K100;
print("q501")
db.things.group(
   {
     key: { k2: 1, k100: 1 },
     reduce: function( curr, result ) {
                result.count ++;
             },
     initial: {  count : 0}
   }
)
# q501
select k2, k100, count(*) from sq11k group by k2,
k100;
```

**Figure 46 – Q501 Nomenclature**

Mongodb log verbosity is set to 2, which provides some additional query information in the mongod.log such as query planning, scoring, and plan selection, as well as keys examined, docs examined, and query response time (protocol:op_command 4146ms in the following snippet from the log).

```
2017-03-31T11:59:33.911-0700 D COMMAND  [conn4] run
command mydb.$cmd { count: "sq110m", query: { k2:
2.0, k100: 3.0 }, fields: {} }
2017-03-31T11:59:33.911-0700 D QUERY    [conn4]
Relevant index 0 is kp: { k100: 1 } name: 'k100_1'
io: { v: 2, key: { k100: 1 }, name: "k100_1", ns:
"mydb.sq110m" }
2017-03-31T11:59:33.911-0700 D QUERY    [conn4]
Relevant index 1 is kp: { k2: 1 } name: 'k2_1' io: {
v: 2, key: { k2: 1 }, name: "k2_1", ns:
"mydb.sq110m" }
2017-03-31T11:59:38.058-0700 D QUERY    [conn4]
Scoring query plan: IXSCAN { k2: 1 } planHitEOF=0
2017-03-31T11:59:38.058-0700 D QUERY    [conn4]
score(1) = baseScore(1) + productivity((0
advanced)/(87716 works) = 0) + tieBreakers(1.14004e-
06 noFetchBonus + 1.14004e-06 noSortBonus +
1.14004e-06 noIxisectBonus = 3.42013e-06)
2017-03-31T11:59:38.058-0700 D QUERY    [conn4]
Scoring query plan: IXSCAN { k100: 1 } planHitEOF=1
2017-03-31T11:59:38.058-0700 D QUERY    [conn4]
score(1) = baseScore(1) + productivity((0
advanced)/(87716 works) = 0) + tieBreakers(1.14004e-
06 noFetchBonus + 1.14004e-06 noSortBonus +
1.14004e-06 noIxisectBonus = 3.42013e-06)
2017-03-31T11:59:38.058-0700 D QUERY    [conn4]
Scoring query plan: IXSCAN { k100: 1 }, IXSCAN { k2:
1 } planHitEOF=0
2017-03-31T11:59:38.058-0700 D QUERY    [conn4]
score(1) = baseScore(1) + productivity((0
advanced)/(87716 works) = 0) + tieBreakers(1.14004e-
06 noFetchBonus + 1.14004e-06 noSortBonus + 0
noIxisectBonus = 2.28009e-06)
2017-03-31T11:59:38.058-0700 D QUERY    [conn4]
Winning plan: IXSCAN { k100: 1 }
2017-03-31T11:59:38.058-0700 I COMMAND  [conn4]
command mydb.sq110m appName: "MongoDB Shell"
command: count { count: "sq110m", query: { k2: 2.0,
k100: 3.0 }, fields: {} } planSummary: IXSCAN {
k100: 1 } keysExamined:87715 docsExamined:87715
fromMultiPlanner:1 numYields:2074 reslen:29 locks:{
Global: { acquireCount: { r: 4150 } }, Database: {
acquireCount: { r: 2075 } }, Collection: {
acquireCount: { r: 2075 } } } protocol:op_command
4146ms
```

**Figure 47 – MongoDB Log Verbosity 2**

**Summary**

This project examined the import and SQL performance of 1K, 10K, 100K, 1M, 10M, and 100M documents using out of the box settings for Hbase, Couchbase, and MongoDB. Tests using the Setquery benchmark are run under RHEL V7 in real SSD environments, on an I7-4700HQ and an I7-7700.

**Software/Hardware Configuration**

The benchmark machine is an ASUS G750JM (i7-4700HQ @2.40GHZ, gbcpu 3620/11722, intelgpu, nvidiagpu) with 32GB of RAM (DDR3@0.799GHZ), a 2TB internal Samsung 850 PRO solid state drive, and a 1TB usb 3.0 external Samsung Portable SSD T3. An ASUS G11CD-K I7-7700 @ 3.60GHZ was also measured (gbcpu 4952/16446), with a 2TB Samsung 850 PRO and 32 GB of RAM (DDR4@2.133GHZ). RHEL70 was installed on the 2TB Internal SSD (Server with GUI), no LVM.

**References**

Jim Gray (Ed.): The Benchmark Handbook for Database and Transaction Systems (2nd Edition). Morgan Kaufmann 1993, ISBN 1-55860-292-5

## Appendix A – Detailed SQL Response Times

| | 4700 | 7700 | 4700 | 7700 | 4700 | 7700 |
|---|---|---|---|---|---|---|
| Query | 1K | 1K | 10K | 10K | 100K | 100K |
| q101 | 0 | 0 | 0 | 1 | 0 | 2 |
| q102 | 0 | 0 | 0 | 0 | 0 | 0 |
| q103 | 0 | 0 | 0 | 0 | 0 | 0 |
| q104 | 0 | 0 | 0 | 0 | 0 | 0 |
| q105 | 0 | 0 | 0 | 0 | 0 | 0 |
| q106 | 0 | 0 | 0 | 0 | 1 | 1 |
| q107 | 0 | 0 | 0 | 7 | 2 | 73 |
| q108 | 0 | 0 | 1 | 0 | 7 | 4 |
| q109 | 0 | 0 | 2 | 1 | 6 | 5 |
| q110 | 0 | 0 | 1 | 1 | 7 | 6 |
| q201 | 0 | 0 | 0 | 0 | 0 | 0 |
| q202 | 0 | 0 | 0 | 0 | 0 | 0 |
| q203 | 0 | 0 | 0 | 0 | 1 | 0 |
| q204 | 0 | 0 | 1 | 0 | 14 | 0 |
| q205 | 0 | 0 | 3 | 0 | 38 | 2 |
| q206 | 0 | 0 | 2 | 0 | 19 | 8 |
| q207 | 0 | 0 | 3 | 2 | 13 | 22 |
| q208 | 0 | 0 | 5 | 3 | 22 | 38 |
| q209 | 0 | 0 | 3 | 2 | 27 | 20 |
| q210 | 0 | 0 | 3 | 2 | 26 | 19 |
| q211 | 0 | 0 | 4 | 2 | 30 | 22 |
| q212 | 0 | 0 | 3 | 2 | 30 | 22 |
| q213 | 0 | 0 | 3 | 2 | 31 | 22 |
| q214 | 0 | 0 | 3 | 2 | 30 | 23 |
| q215 | 0 | 0 | 3 | 2 | 31 | 23 |
| q216 | 0 | 0 | 3 | 2 | 32 | 22 |
| q217 | 0 | 0 | 3 | 2 | 32 | 24 |
| q218 | 0 | 0 | 3 | 2 | 32 | 25 |
| q301 | 0 | 0 | 0 | 0 | 0 | 0 |
| q302 | 0 | 0 | 0 | 0 | 0 | 0 |
| q303 | 0 | 0 | 0 | 0 | 0 | 0 |
| q304 | 0 | 0 | 0 | 0 | 2 | 1 |
| q305 | 0 | 0 | 0 | 0 | 7 | 3 |
| q306 | 0 | 0 | 1 | 4 | 14 | 41 |
| q307 | 0 | 0 | 2 | 1 | 26 | 17 |
| q308 | 0 | 0 | 3 | 2 | 30 | 20 |
| q309 | 0 | 0 | 0 | 0 | 0 | 0 |
| q310 | 0 | 0 | 0 | 0 | 0 | 0 |
| q311 | 0 | 0 | 0 | 0 | 0 | 0 |
| q312 | 0 | 0 | 0 | 0 | 3 | 1 |
| q313 | 0 | 0 | 1 | 0 | 9 | 6 |
| q314 | 0 | 1 | 2 | 9 | 20 | 52 |
| q315 | 0 | 0 | 4 | 2 | 37 | 34 |
| q316 | 0 | 0 | 5 | 3 | 44 | 32 |
| q401 | 1 | 0 | 13 | 8 | 57 | 92 |
| q402 | 0 | 0 | 7 | 5 | 22 | 58 |
| q403 | 0 | 0 | 4 | 2 | 12 | 28 |
| q404 | 0 | 0 | 4 | 3 | 12 | 29 |
| q405 | 0 | 0 | 3 | 2 | 11 | 27 |
| q406 | 1 | 0 | 9 | 6 | 24 | 71 |
| q407 | 0 | 0 | 4 | 7 | 14 | 129 |
| q408 | 0 | 0 | 4 | 4 | 12 | 45 |
| q409 | 0 | 0 | 6 | 4 | 13 | 51 |
| q410 | 0 | 0 | 6 | 4 | 12 | 48 |
| q411 | 0 | 0 | 6 | 4 | 12 | 47 |
| q412 | 0 | 0 | 6 | 4 | 12 | 47 |
| q413 | 0 | 0 | 8 | 3 | 12 | 37 |
| q414 | 0 | 0 | 0 | 0 | 1 | 1 |
| q501 | 181 | 27 | 200 | 162 | 1,957 | 1,599 |
| q502 | 22 | 18 | 200 | 160 | 1,941 | 1,554 |
| q503 | 22 | 17 | 197 | 162 | 1,957 | 1,586 |
| total ms | 227 | 63 | 744 | 596 | 6,704 | 6,039 |
| total sec | 0.23 | 0.06 | 0.74 | 0.60 | 6.70 | 6.04 |
| | | 0.278 | | 0.801 | | 0.901 |

**Figure 48 – MongoDB Response Times**

| | 4700 | 7700 | 4700 | 7700 | 4700 | 7700 |
|---|---|---|---|---|---|---|
| Query | 1M | 1M | 10M | 10M | 100M | 100M |
| q101 | 0 | 2 | 2 | 0 | 2 | 2 |
| q102 | 0 | 0 | 0 | 0 | 0 | 0 |
| q103 | 0 | 0 | 0 | 0 | 3 | 2 |
| q104 | 0 | 0 | 3 | 2 | 24 | 18 |
| q105 | 0 | 2 | 24 | 18 | 231 | 175 |
| q106 | 11 | 8 | 98 | 72 | 974 | 724 |
| q107 | 25 | 707 | 245 | 7327 | 2,397 | 69,522 |
| q108 | 50 | 36 | 479 | 354 | 4,678 | 3,516 |
| q109 | 61 | 45 | 585 | 447 | 5,841 | 4,329 |
| q110 | 77 | 57 | 758 | 568 | 7,575 | 5,697 |
| q201 | 0 | 0 | 1 | 2 | 1 | 2 |
| q202 | 3 | 0 | 25 | 0 | 234 | 16 |
| q203 | 15 | 0 | 143 | 3 | 1,498 | 156 |
| q204 | 138 | 2 | 1,256 | 26 | 13,169 | 1,546 |
| q205 | 416 | 22 | 4,146 | 226 | 43,568 | 11,731 |
| q206 | 279 | 83 | 2,740 | 853 | 40,275 | 30,267 |
| q207 | 274 | 235 | 2,739 | 2,391 | 58,201 | 45,198 |
| q208 | 499 | 389 | 5,054 | 3,991 | 88,132 | 69,968 |
| q209 | 280 | 209 | 2,643 | 2,037 | 46,575 | 41,868 |
| q210 | 257 | 194 | 2,539 | 1,930 | 46,779 | 41,507 |
| q211 | 305 | 227 | 2,964 | 2,262 | 50,346 | 44,851 |
| q212 | 299 | 233 | 2,999 | 2,308 | 51,000 | 45,406 |
| q213 | 305 | 235 | 3,064 | 2,323 | 53,114 | 45,007 |
| q214 | 315 | 232 | 3,070 | 2,362 | 52,105 | 45,723 |
| q215 | 339 | 241 | 3,126 | 2,407 | 52,974 | 45,626 |
| q216 | 320 | 235 | 3,155 | 2,383 | 52,450 | 45,647 |
| q217 | 342 | 249 | 3,170 | 2,433 | 52,185 | 46,511 |
| q218 | 319 | 244 | 3,187 | 2,479 | 52,676 | 46,603 |
| q301 | 0 | 0 | 0 | 0 | 22 | 13 |
| q302 | 0 | 0 | 2 | 1 | 177 | 124 |
| q303 | 0 | 0 | 7 | 4 | 66 | 45 |
| q304 | 18 | 11 | 180 | 119 | 12,563 | 11,774 |
| q305 | 60 | 40 | 588 | 406 | 25,393 | 25,455 |
| q306 | 129 | 428 | 1,270 | 4,233 | 35,073 | 65,485 |
| q307 | 240 | 171 | 2,388 | 1,760 | 47,248 | 39,178 |
| q308 | 288 | 215 | 2,892 | 2,257 | 51,080 | 44,620 |
| q309 | 0 | 0 | 1 | 0 | 24 | 15 |
| q310 | 0 | 0 | 4 | 3 | 201 | 143 |
| q311 | 3 | 2 | 24 | 17 | 1,664 | 1,273 |
| q312 | 20 | 14 | 200 | 138 | 12,117 | 11,472 |
| q313 | 70 | 51 | 656 | 493 | 25,293 | 23,037 |
| q314 | 156 | 134 | 1,474 | 212 | 38,654 | 184 |
| q315 | 292 | 230 | 2,808 | 2,180 | 48,735 | 42,884 |
| q316 | 362 | 288 | 3,412 | 2,848 | 55,947 | 48,969 |
| q401 | 1,484 | 1,042 | 16,297 | 11,582 | 980,894 | 736,018 |
| q402 | 971 | 675 | 100,650 | 7,414 | 677,642 | 501,485 |
| q403 | 435 | 301 | 4,657 | 3,281 | 219,477 | 161,910 |
| q404 | 400 | 310 | 3,989 | 3,104 | 111,614 | 76,018 |
| q405 | 391 | 276 | 3,915 | 2,890 | 180,469 | 141,830 |
| q406 | 1,195 | 823 | 12,790 | 8,933 | 712,146 | 518,158 |
| q407 | 487 | 1544 | 5,178 | 17,603 | 308,477 | 136,517 |
| q408 | 622 | 533 | 6,620 | 5,961 | 378,471 | 383,292 |
| q409 | 733 | 522 | 7,775 | 5,672 | 340,976 | 260,479 |
| q410 | 710 | 518 | 7,650 | 5,677 | 347,690 | 267,782 |
| q411 | 711 | 509 | 7,680 | 5,772 | 322,291 | 242,218 |
| q412 | 702 | 501 | 7,365 | 5,594 | 328,180 | 229,424 |
| q413 | 716 | 402 | 7,470 | 4,292 | 333,146 | 216,818 |
| q414 | 31 | 17 | 323 | 179 | 17,951 | 10,091 |
| q501 | 19,025 | 16,089 | 191,975 | 159,948 | 1,965,896 | 1,613,738 |
| q502 | 19,103 | 15,999 | 190,096 | 157,836 | 1,944,303 | 1,601,949 |
| q503 | 18,937 | 15,997 | 189,910 | 157,036 | 1,942,539 | 1,600,901 |
| total ms | 73,220 | 61,529 | 828,461 | 616,649 | 12,243,426 | 9,704,917 |
| total sec | 73.22 | 61.53 | 828 | 617 | 12,243 | 9,705 |
| | | 0.840 | | 0.744 | | 0.793 |

**Figure 49 – MongoDB Response Times**

| | 4700 | 7700 | 4700 | 7700 | 4700 | 7700 |
|---|---|---|---|---|---|---|
| Query | 1K | 1K | 10K | 10K | 100K | 100K |
| q101 | 129 | 22 | 198 | 129 | 1,960 | 1,196 |
| q102 | 28 | 21 | 184 | 133 | 1,760 | 1,169 |
| q103 | 30 | 21 | 212 | 126 | 1,700 | 1,205 |
| q104 | 31 | 20 | 183 | 130 | 1,780 | 1,235 |
| q105 | 29 | 22 | 207 | 133 | 1,850 | 1,151 |
| q106 | 31 | 24 | 190 | 144 | 1,810 | 1,194 |
| q107 | 30 | 20 | 180 | 131 | 1,730 | 1,181 |
| q108 | 30 | 22 | 188 | 132 | 1,780 | 1,219 |
| q109 | 30 | 27 | 189 | 135 | 1,940 | 1,230 |
| q110 | 28 | 21 | 189 | 128 | 1,760 | 1,221 |
| q201 | 35 | 25 | 191 | 138 | 1,920 | 1,308 |
| q202 | 30 | 29 | 189 | 136 | 1,810 | 1,260 |
| q203 | 34 | 22 | 199 | 153 | 1,830 | 1,260 |
| q204 | 34 | 22 | 201 | 134 | 1,840 | 1,259 |
| q205 | 33 | 22 | 193 | 141 | 1,820 | 1,255 |
| q206 | 32 | 24 | 194 | 147 | 2,120 | 1,258 |
| q207 | 32 | 23 | 212 | 133 | 1,830 | 1,236 |
| q208 | 34 | 24 | 195 | 137 | 1,830 | 1,296 |
| q209 | 39 | 25 | 190 | 139 | 1,830 | 1,323 |
| q210 | 38 | 26 | 207 | 144 | 2,090 | 1,331 |
| q211 | 33 | 23 | 192 | 139 | 1,930 | 1,269 |
| q212 | 36 | 23 | 201 | 134 | 1,830 | 1,284 |
| q213 | 33 | 23 | 332 | 128 | 1,880 | 1,280 |
| q214 | 35 | 24 | 194 | 134 | 2,070 | 1,289 |
| q215 | 33 | 23 | 205 | 134 | 1,850 | 1,295 |
| q216 | 33 | 24 | 198 | 138 | 1,920 | 1,275 |
| q217 | 37 | 25 | 202 | 153 | 2,060 | 1,328 |
| q218 | 34 | 23 | 216 | 130 | 1,830 | 1,293 |
| q301 | 30 | 20 | 180 | 134 | 1,710 | 1,171 |
| q302 | 31 | 20 | 176 | 123 | 1,700 | 1,177 |
| q303 | 27 | 22 | 184 | 123 | 1,930 | 1,192 |
| q304 | 30 | 21 | 176 | 133 | 1,750 | 1,172 |
| q305 | 31 | 21 | 189 | 128 | 1,770 | 1,200 |
| q306 | 29 | 22 | 183 | 123 | 1,740 | 1,227 |
| q307 | 32 | 27 | 187 | 131 | 1,980 | 1,270 |
| q308 | 32 | 22 | 188 | 137 | 1,830 | 1,255 |
| q309 | 62 | 43 | 495 | 358 | 4,650 | 3,292 |
| q310 | 64 | 43 | 489 | 341 | 4,830 | 3,339 |
| q311 | 66 | 43 | 504 | 347 | 4,800 | 3,386 |
| q312 | 60 | 47 | 484 | 335 | 4,740 | 3,261 |
| q313 | 64 | 44 | 496 | 352 | 4,860 | 3,433 |
| q314 | 60 | 43 | 486 | 348 | 4,780 | 3,284 |
| q315 | 63 | 47 | 503 | 354 | 4,970 | 3,479 |
| q316 | 65 | 45 | 501 | 342 | 4,860 | 3,377 |
| q401 | 37 | 24 | 213 | 150 | 1,990 | 1,375 |
| q402 | 40 | 25 | 216 | 154 | 2,020 | 1,394 |
| q403 | 42 | 30 | 288 | 193 | 2,800 | 1,887 |
| q404 | 47 | 34 | 305 | 202 | 2,770 | 1,928 |
| q405 | 42 | 29 | 268 | 195 | 2,570 | 1,782 |
| q406 | 38 | 26 | 301 | 151 | 2,080 | 1,447 |
| q407 | 34 | 24 | 208 | 157 | 2,040 | 1,294 |
| q408 | 35 | 25 | 220 | 150 | 1,990 | 1,397 |
| q409 | 48 | 35 | 475 | 268 | 3,580 | 2,476 |
| q410 | 52 | 39 | 379 | 267 | 3,740 | 2,538 |
| q411 | 50 | 39 | 392 | 268 | 3,630 | 2,575 |
| q412 | 53 | 36 | 383 | 266 | 3,830 | 2,577 |
| q413 | 50 | 35 | 361 | 249 | 3,420 | 2,500 |
| q414 | 51 | 38 | 322 | 212 | 2,900 | 1,999 |
| q501 | 39 | 33 | 213 | 147 | 2,180 | 1,382 |
| q502 | 38 | 27 | 266 | 160 | 2,280 | 1,524 |
| q503 | 41 | 25 | 216 | 150 | 2,050 | 1,344 |
| total ms | 2,494 | 1,704 | 16,078 | 10,961 | 151,130 | 103,034 |
| total sec | 2.49 | 1.70 | 16.08 | 10.96 | 151 | 103 |
| | | 0.683 | | 0.682 | | 0.682 |

**Figure 50 – Couchbase Response Times**

| | 4700 | 7700 | 4700 | 7700 | 4700 | 7700 |
|---|---|---|---|---|---|---|
| Query | 1M | 1M | 10M | 10M | 100M | 100M |
| q101 | 18,540 | 12,041 | 316,600 | 221,000 | 3,657,000 | 2,514,000 |
| q102 | 17,970 | 11,716 | 315,800 | 219,000 | | |
| q103 | 17,980 | 11,926 | 324,400 | 216,000 | | |
| q104 | 18,180 | 12,153 | 314,300 | 218,000 | | |
| q105 | 17,900 | 13,522 | 310,400 | 216,000 | | |
| q106 | 18,250 | 13,001 | 315,800 | 218,000 | | |
| q107 | 18,060 | 12,092 | 312,700 | 218,000 | | |
| q108 | 18,710 | 12,463 | 319,000 | 221,000 | | |
| q109 | 18,700 | 12,821 | 316,800 | 220,000 | | |
| q110 | 18,660 | 12,216 | 315,500 | 220,000 | | |
| q201 | 19,430 | 12,608 | 352,800 | 230,000 | 3,830,000 | 2,661,000 |
| q202 | 18,940 | 12,364 | 357,100 | 226,000 | | |
| q203 | 19,050 | 12,406 | 360,800 | 225,000 | | |
| q204 | 19,210 | 12,434 | 355,400 | 226,000 | | |
| q205 | 19,180 | 12,399 | 321,900 | 225,000 | | |
| q206 | 19,030 | 12,490 | 324,900 | 228,000 | | |
| q207 | 18,920 | 12,376 | 320,700 | 224,000 | | |
| q208 | 19,290 | 12,553 | 328,100 | 229,000 | | |
| q209 | 19,240 | 12,802 | 327,300 | 229,000 | | |
| q210 | 20,220 | 13,108 | 330,900 | 232,000 | | |
| q211 | 19,540 | 13,121 | 325,000 | 227,000 | | |
| q212 | 19,510 | 12,652 | 325,200 | 227,000 | | |
| q213 | 19,390 | 12,762 | 325,900 | 228,000 | | |
| q214 | 19,460 | 12,601 | 323,500 | 227,000 | | |
| q215 | 19,570 | 12,855 | 327,400 | 229,000 | | |
| q216 | 19,480 | 12,638 | 324,300 | 226,000 | | |
| q217 | 19,680 | 12,857 | 328,900 | 230,000 | | |
| q218 | 19,470 | 12,762 | 326,700 | 229,000 | | |
| q301 | 17,990 | 11,805 | 312,540 | 216,000 | 3,612,000 | 2,490,000 |
| q302 | 17,960 | 11,715 | 311,000 | 217,000 | | |
| q303 | 18,190 | 11,827 | 312,200 | 217,000 | | |
| q304 | 18,000 | 11,651 | 310,500 | 216,000 | | |
| q305 | 18,340 | 12,039 | 314,800 | 219,000 | | |
| q306 | 18,380 | 11,749 | 313,000 | 218,000 | | |
| q307 | 19,220 | 12,439 | 320,000 | 223,000 | | |
| q308 | 19,050 | 12,507 | 320,800 | 223,000 | | |
| q309 | 47,520 | 32,744 | 521,900 | 367,000 | | |
| q310 | 48,050 | 33,079 | 524,400 | 370,000 | | |
| q311 | 48,600 | 33,468 | 531,200 | 374,000 | | |
| q312 | 47,530 | 33,532 | 518,300 | 366,000 | | |
| q313 | 49,510 | 34,071 | 539,100 | 380,000 | | |
| q314 | 47,290 | 32,602 | 519,900 | 366,000 | | |
| q315 | 50,120 | 34,700 | 548,600 | 388,000 | | |
| q316 | 48,500 | 33,490 | 530,900 | 375,000 | | |
| q401 | 20,470 | 13,570 | 332,000 | 235,000 | 3,874,000 | 2,684,000 |
| q402 | 21,060 | 14,094 | 337,000 | 235,000 | | |
| q403 | 27,750 | 18,779 | 344,000 | 242,000 | | |
| q404 | 28,500 | 19,355 | 347,000 | 244,000 | | |
| q405 | 26,430 | 17,855 | 342,400 | 240,000 | | |
| q406 | 21,210 | 14,443 | 335,400 | 236,000 | | |
| q407 | 19,720 | 13,031 | 331,900 | 233,000 | | |
| q408 | 20,880 | 13,896 | 335,100 | 235,000 | | |
| q409 | 36,370 | 24,959 | 350,000 | 289,000 | | |
| q410 | 37,080 | 25,386 | 355,000 | 293,000 | | |
| q411 | 37,010 | 25,510 | 356,000 | 294,000 | | |
| q412 | 37,580 | 25,858 | 421,900 | 297,000 | | |
| q413 | 34,680 | 23,897 | 394,500 | 279,000 | | |
| q414 | 29,730 | 20,390 | 355,300 | 251,000 | | |
| q501 | 21,000 | 13,674 | 338,000 | 238,000 | 3,320,000 | 2,845,000 |
| q502 | 22,750 | 15,159 | 341,000 | 239,000 | | |
| q503 | 20,650 | 13,565 | 340,000 | 239,000 | | |
| total ms | 1,528,680 | 1,028,578 | 21,853,740 | 15,338,000 | 18,297,700 | 13,201,700 |
| total sec | 1,529 | 1,029 | 21,854 | 15,338 | 18,298 | 13,202 |
| | | 0.673 | | 0.702 | | 0.721 |

**Figure 51 – Couchbase Response Times**

## Appendix B - SQL

### HBASE/Vertica Queries

```
q101 SELECT COUNT(*) FROM SQ1 WHERE K250K = 2;
q102 SELECT COUNT(*) FROM SQ1 WHERE K100K = 2;
q103 SELECT COUNT(*) FROM SQ1 WHERE K10K = 2;
q104 SELECT COUNT(*) FROM SQ1 WHERE K1K = 2;
q105 SELECT COUNT(*) FROM SQ1 WHERE K100 = 2;
q106 SELECT COUNT(*) FROM SQ1 WHERE K25 = 2;
q107 SELECT COUNT(*) FROM SQ1 WHERE K10 = 2;
q108 SELECT COUNT(*) FROM SQ1 WHERE K5 = 2;
q109 SELECT COUNT(*) FROM SQ1 WHERE K4 = 2;
q110 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2;
q201 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND
KSEQ=3;
q202 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K100K
= 3;
q203 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K10K
= 3;
q204 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K1K =
3;
q205 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K100
= 3;
q206 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K25 =
3;
q207 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K10 =
3;
q208 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K5 =
3;
q209 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K4 =
3;
q210 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND KSEQ
<> 3;
q211 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K100K
<> 3;
q212 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K10K
<> 3;
q213 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K1K
<> 3;
q214 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K100
<> 3;
q215 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K25
<> 3;
q216 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K10
<> 3;
q217 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K5 <>
3;
q218 SELECT COUNT(*) FROM SQ1 WHERE K2 = 2 AND K4 <>
3;
q301 SELECT SUM(K1K) FROM SQ1 WHERE K100K = 3;
q302 SELECT SUM(K1K) FROM SQ1 WHERE K10K = 3;
q303 SELECT SUM(K1K) FROM SQ1 WHERE K1K = 3;
q304 SELECT SUM(K1K) FROM SQ1 WHERE K100 = 3;
q305 SELECT SUM(K1K) FROM SQ1 WHERE K25 = 3;
q306 SELECT SUM(K1K) FROM SQ1 WHERE K10 = 3;
q307 SELECT SUM(K1K) FROM SQ1 WHERE K5 = 3;
q308 SELECT SUM(K1K) FROM SQ1 WHERE K4 = 3;
q309 SELECT SUM(K1K) FROM SQ1
WHERE (KSEQ BETWEEN 0 AND 25000 OR
KSEQ BETWEEN 50000 AND 75000 OR
KSEQ BETWEEN 100000 AND 125000 OR
      KSEQ BETWEEN 150000 AND 175000 OR
      KSEQ BETWEEN 200000 AND 225000)
      AND K100K = 3;
q310 SELECT SUM(K1K) FROM SQ1
WHERE (KSEQ BETWEEN 0 AND 25000 OR
KSEQ BETWEEN 50000 AND 75000 OR
KSEQ BETWEEN 100000 AND 125000 OR
KSEQ BETWEEN 150000 AND 175000 OR
KSEQ BETWEEN 200000 AND 225000)
AND K10K = 3;
q311 SELECT SUM(K1K) FROM SQ1
WHERE (KSEQ BETWEEN 0 AND 25000 OR
KSEQ BETWEEN 50000 AND 75000 OR
KSEQ BETWEEN 100000 AND 125000 OR
KSEQ BETWEEN 150000 AND 175000 OR
KSEQ BETWEEN 200000 AND 225000)
AND K1K = 3;
q312 SELECT SUM(K1K) FROM SQ1
WHERE (KSEQ BETWEEN 0 AND 25000 OR
KSEQ BETWEEN 50000 AND 75000 OR
KSEQ BETWEEN 100000 AND 125000 OR
KSEQ BETWEEN 150000 AND 175000 OR
KSEQ BETWEEN 200000 AND 225000)
AND K100 = 3;
q313 SELECT SUM(K1K) FROM SQ1
WHERE (KSEQ BETWEEN 0 AND 25000 OR
KSEQ BETWEEN 50000 AND 75000 OR
KSEQ BETWEEN 100000 AND 125000 OR
KSEQ BETWEEN 150000 AND 175000 OR
KSEQ BETWEEN 200000 AND 225000)
AND K25 = 3;
q314 SELECT SUM(K1K) FROM SQ1
WHERE (KSEQ BETWEEN 0 AND 25000 OR
KSEQ BETWEEN 50000 AND 75000 OR
KSEQ BETWEEN 100000 AND 125000 OR
KSEQ BETWEEN 150000 AND 175000 OR
KSEQ BETWEEN 200000 AND 225000)
AND K10 = 3;
q315 SELECT SUM(K1K) FROM SQ1
WHERE (KSEQ BETWEEN 0 AND 25000 OR
KSEQ BETWEEN 50000 AND 75000 OR
KSEQ BETWEEN 100000 AND 125000 OR
KSEQ BETWEEN 150000 AND 175000 OR
KSEQ BETWEEN 200000 AND 225000)
AND K5 = 3;
q316 SELECT SUM(K1K) FROM SQ1
WHERE (KSEQ BETWEEN 0 AND 25000 OR
KSEQ BETWEEN 50000 AND 75000 OR
KSEQ BETWEEN 100000 AND 125000 OR
KSEQ BETWEEN 150000 AND 175000 OR
KSEQ BETWEEN 200000 AND 225000)
AND K4 = 3;
q401 SELECT KSEQ, K500K FROM SQ1
WHERE K2 = 1 AND
K100 > 80 AND
K10K BETWEEN 2000 AND 3000;
q402 SELECT KSEQ, K500K FROM SQ1
WHERE  K100 > 80 AND
K10K BETWEEN 2000 AND 3000
AND K5 = 3;
q403 SELECT KSEQ, K500K FROM SQ1
WHERE K10K BETWEEN 2000 AND 3000
AND K5 = 3 AND
(K25 = 11 OR K25 = 9);
q404 SELECT KSEQ, K500K FROM SQ1
WHERE  K5 = 3 AND
(K25 = 11 OR K25 = 9)
AND K4 = 3;
q405 SELECT KSEQ, K500K FROM SQ1
WHERE (K25 = 11 OR K25 = 9) AND
K4 = 3 AND
K100 < 41;
q406 SELECT KSEQ, K500K FROM SQ1
WHERE K4 = 3 AND
K100 < 41 AND
K1K BETWEEN 850 AND 950;
q407 SELECT KSEQ, K500K FROM SQ1
WHERE K100 < 41 AND
K1K BETWEEN 850 AND 950 AND
K10 = 7;
q408 SELECT KSEQ, K500K FROM SQ1
WHERE K1K BETWEEN 850 AND 950 AND
K10 = 7 AND
K25 BETWEEN 3 AND 4;
q409 SELECT KSEQ, K500K FROM SQ1
WHERE K2 = 1 AND
K100 > 80  AND
K10K BETWEEN 2000 AND 3000 AND
K5 = 3 AND
(K25 = 11 OR K25 = 9);
q410 SELECT KSEQ, K500K FROM SQ1
WHERE K100 > 80 AND
K10K BETWEEN 2000 AND 3000 AND
K5 = 3 AND
(K25 = 11 OR K25 = 9)
AND K4 = 3;
```

```
q411 SELECT KSEQ, K500K FROM SQ1
WHERE K10K BETWEEN 2000 AND 3000 AND
K5 = 3 AND
(K25 = 11 OR K25 = 9) AND
K4 = 3 AND
K100 < 41;
q412 SELECT KSEQ, K500K FROM SQ1
WHERE K5 = 3 AND
(K25 = 11 OR K25 = 9) AND
K4 = 3 AND
K100 < 41 AND
K1K BETWEEN 850 AND 950;
q413 SELECT KSEQ, K500K FROM SQ1
WHERE (K25 = 11 OR K25 = 9) AND
K4 = 3 AND K100 < 41 AND
K1K BETWEEN 850 AND 950 AND
K10 = 7;
q414 SELECT KSEQ, K500K FROM SQ1
WHERE K4 = 3 AND
K100 < 41 AND
K1K BETWEEN 850 AND 950 AND
K10 = 7 AND
K1K BETWEEN 3 AND 4;
q501 SELECT K2, K100, COUNT(*) FROM SQ1 GROUP BY K2,
K100;
q502 SELECT K4, K25, COUNT(*) FROM SQ1 GROUP BY K4,
K25;
q503 SELECT K10, K25, COUNT(*) FROM SQ1 GROUP BY
K10, K25;
q601 SELECT COUNT(*) FROM SQ1 T1, SQ1 T2
 WHERE T1.K100K = 49 AND T1.K250K = T2.K500K;
q602 SELECT COUNT(*) FROM SQ1 T1, SQ1 T2
 WHERE T1.K40K = 49  AND T1.K250K = T2.K500K;
q603 SELECT COUNT(*) FROM SQ1 T1, SQ1 T2
 WHERE T1.K10K = 49  AND T1.K250K = T2.K500K;
q604 SELECT COUNT(*) FROM SQ1 T1, SQ1 T2
 WHERE T1.K1K = 49 AND T1.K250K = T2.K500K;
q605 SELECT COUNT(*) FROM SQ1 T1, SQ1 T2
 WHERE T1.K100 = 49 AND T1.K250K = T2.K500K;
q606 SELECT COUNT(*) FROM SQ1 T1, SQ1 T2
 WHERE T1.K100K = 49 AND T1.K250K = T2.K500K AND
T2.K25 =19;
q607 SELECT COUNT(*) FROM SQ1 T1, SQ1 T2
 WHERE T1.K40K = 49 AND T1.K250K = T2.K500K AND
T2.K25 =19;
q608 SELECT COUNT(*) FROM SQ1 T1, SQ1 T2
 WHERE T1.K10K = 49 AND T1.K250K = T2.K500K AND
T2.K25 =19;
q609 SELECT COUNT(*) FROM SQ1 T1, SQ1 T2
 WHERE T1.K1K = 49 AND T1.K250K = T2.K500K AND
T2.K25 =19;
q610 SELECT COUNT(*) FROM SQ1 T1, SQ1 T2
 WHERE T1.K100 = 49 AND T1.K250K = T2.K500K AND
T2.K25 =19;
```

### Couchbase Queries

```
# q101
select count(*) from sq11k where k250k = "2";
# q102
select count(*) from sq11k where k100k = "2";
# q103
select count(*) from sq11k where k10k = "2";
# q104
select count(*) from sq11k where k1k = "2";
# q105
select count(*) from sq11k where k100 = "2";
# q106
select count(*) from sq11k where k25 = "2";
# q107
select count(*) from sq11k where k10 = "2";
# q108
select count(*) from sq11k where k5 = "2";
# q109
select count(*) from sq11k where k4 = "2";
# q110
select count(*) from sq11k where k2 = "2";
# q201
select count(*) from sq11k where k2 = "2" and kseq =
"3";
# q202
select count(*) from sq11k where k2 = "2" and k100k
= "3";
# q203
select count(*) from sq11k where k2 = "2" and k10k =
"3";
# q204
select count(*) from sq11k where k2 = "2" and k1k =
"3";
# q205
select count(*) from sq11k where k2 = "2" and k100 =
"3";
# q206
select count(*) from sq11k where k2 = "2" and k25 =
"3";
# q207
select count(*) from sq11k where k2 = "2" and k10 =
"3";
# q208
select count(*) from sq11k where k2 = "2" and k5 =
"3";
# q209
select count(*) from sq11k where k2 = "2" and k4 =
"3";
# q210
select count(*) from sq11k where k2 = "2" and kseq
<> "3";
# q211
select count(*) from sq11k where k2 = "2" and k100k
<> "3";
# q212
select count(*) from sq11k where k2 = "2" and k10k
<> "3";
# q213
select count(*) from sq11k where k2 = "2" and k1k <>
"3";
# q214
select count(*) from sq11k where k2 = "2" and k100
<> "3";
# q215
select count(*) from sq11k where k2 = "2" and k25 <>
"3";
# q216
select count(*) from sq11k where k2 = "2" and k10 <>
"3";
# q217
select count(*) from sq11k where k2 = "2" and k5 <>
"3";
# q218
select count(*) from sq11k where k2 = "2" and k4 <>
"3";
# q301
select sum(TO_NUMBER(k1k)) from sq11k where k100k =
"3";
# q302
select sum(TO_NUMBER(k1k)) from sq11k where k10k =
"3";
# q303
select sum(TO_NUMBER(k1k)) from sq11k where k1k =
"3";
# q304
select sum(TO_NUMBER(k1k)) from sq11k where k100 =
"3";
# q305
select sum(TO_NUMBER(k1k)) from sq11k where k25 =
"3";
# q306
select sum(TO_NUMBER(k1k)) from sq11k where k10 =
"3";
# q307
select sum(TO_NUMBER(k1k)) from sq11k where k5 =
"3";
# q308
select sum(TO_NUMBER(k1k)) from sq11k where k4 =
"3";
# q309
```

```
select sum(TO_NUMBER(k1k)) FROM sq11k
WHERE (kseq BETWEEN "0" AND "25000" OR
kseq BETWEEN "50000" AND "75000" OR
kseq BETWEEN "100000" AND "125000" OR
      kseq BETWEEN "150000" AND "175000" OR
      kseq BETWEEN "200000" AND "225000")
      AND k100k = "3";
# q310
select sum(TO_NUMBER(k1k)) FROM sq11k
WHERE (kseq BETWEEN "0" AND "25000" OR
kseq BETWEEN "50000" AND "75000" OR
kseq BETWEEN "100000" AND "125000" OR
      kseq BETWEEN "150000" AND "175000" OR
      kseq BETWEEN "200000" AND "225000")
      AND k10k = "3";
# q311
select sum(TO_NUMBER(k1k)) FROM sq11k
WHERE (kseq BETWEEN "0" AND "25000" OR
kseq BETWEEN "50000" AND "75000" OR
kseq BETWEEN "100000" AND "125000" OR
      kseq BETWEEN "150000" AND "175000" OR
      kseq BETWEEN "200000" AND "225000")
      AND k1k = "3";
# q312
select sum(TO_NUMBER(k1k)) FROM sq11k
WHERE (kseq BETWEEN "0" AND "25000" OR
kseq BETWEEN "50000" AND "75000" OR
kseq BETWEEN "100000" AND "125000" OR
      kseq BETWEEN "150000" AND "175000" OR
      kseq BETWEEN "200000" AND "225000")
      AND k100 = "3";
# q313
select sum(TO_NUMBER(k1k)) FROM sq11k
WHERE (kseq BETWEEN "0" AND "25000" OR
kseq BETWEEN "50000" AND "75000" OR
kseq BETWEEN "100000" AND "125000" OR
      kseq BETWEEN "150000" AND "175000" OR
      kseq BETWEEN "200000" AND "225000")
      AND k25 = "3";
# q314
select sum(TO_NUMBER(k1k)) FROM sq11k
WHERE (kseq BETWEEN "0" AND "25000" OR
kseq BETWEEN "50000" AND "75000" OR
kseq BETWEEN "100000" AND "125000" OR
      kseq BETWEEN "150000" AND "175000" OR
      kseq BETWEEN "200000" AND "225000")
      AND k10 = "3";
# q315
select sum(TO_NUMBER(k1k)) FROM sq11k
WHERE (kseq BETWEEN "0" AND "25000" OR
kseq BETWEEN "50000" AND "75000" OR
kseq BETWEEN "100000" AND "125000" OR
      kseq BETWEEN "150000" AND "175000" OR
      kseq BETWEEN "200000" AND "225000")
      AND k5 = "3";
# q316
select sum(TO_NUMBER(k1k)) FROM sq11k
WHERE (kseq BETWEEN "0" AND "25000" OR
kseq BETWEEN "50000" AND "75000" OR
kseq BETWEEN "100000" AND "125000" OR
      kseq BETWEEN "150000" AND "175000" OR
      kseq BETWEEN "200000" AND "225000")
      AND k1k = "3";
# q401
select kseq, k500k from sq11k where k2 = "1" and
k100 > "80" and
k10k between "2000" and "3000";
# q402
select kseq, k500k from sq11k where k100 > "80" and
k10k between "2000" and "3000"
and k5 ="3";
# q403
select kseq, k500k from sq11k where k10k between
"2000" and "3000"
and k5 ="3"
and (k25 = "11" or k25 = "9");
# q404
select kseq, k500k from sq11k where k5 = "3"
and (k25 = "11" or k25 = "9")
and k4 = "3";
# q405
select kseq, k500k from sq11k
where (k25 = "11" or k25 = "9") and
k4 = "3" and
k100 < "41";
# q406
SELECT kseq, k500k FROM sq11k
WHERE k4 = "3" AND
k100 < "41" AND
k1k BETWEEN "850" AND "950";
# q407
SELECT kseq, k500k FROM sq11k
WHERE k100 < "41" AND
k1k BETWEEN "850" AND "950" AND
k10 = "7";
# q408
SELECT kseq, k500k FROM sq11k
WHERE k1k BETWEEN "850" AND "950" AND
k10 = "7" AND
k25 BETWEEN "3" AND "4";
# q409
SELECT kseq, k500k FROM sq11k
WHERE k2 = "1" AND
k100 > "80"  AND
k10k BETWEEN "2000" AND "3000" AND
k5 = "3" AND
(k25 = "11" OR k25 = "9");
# q410
SELECT kseq, k500k FROM sq11k
WHERE k100 > "80" AND
k10k BETWEEN "2000" AND "3000" AND
k5 = "3" AND
(k25 = "11" OR k25 = "9")
AND k4 = "3";
# q411
SELECT kseq, k500k FROM sq11k
WHERE k10k BETWEEN "2000" AND "3000" AND
k5 = "3" AND
(k25 = "11" OR k25 = "9") AND
k4 = "3" AND
k100 < "41";
# q412
SELECT kseq, k500k FROM sq11k
WHERE k5 = "3" AND
(k25 = "11" OR k25 = "9") AND
k4 = "3" AND
k100 < "41" AND
k1k BETWEEN "850" AND "950";
# q413
SELECT kseq, k500k FROM sq11k
WHERE (k25 = "11" OR k25 = "9") AND
k4 = "3" AND k100 < "41" AND
k1k BETWEEN "850" AND "950" AND
k10 = "7";
# q414
SELECT kseq, k500k FROM sq11k
WHERE k4 = "3" AND
k100 < "41" AND
k1k BETWEEN "850" AND "950" AND
k10 = "7" AND
k1k BETWEEN "3" AND "4";
# q501
select k2, k100, count(*) from sq11k group by k2,
k100;
# q502
select k4, k25, count(*) from sq11k group by k4,
k25;
# q503
select k10, k25, count(*) from sq11k group by k10,
k25;
```

## Couchbase TO_NUMBER Queries

```
# q101
select count(*) from sq11knum where TO_NUMBER(k250k)
= 2;
# q102
select count(*) from sq11knum where TO_NUMBER(k100k)
= 2;
# q103
select count(*) from sq11knum where TO_NUMBER(k10k)
= 2;
# q104
select count(*) from sq11knum where TO_NUMBER(k1k) =
2;
# q105
select count(*) from sq11knum where TO_NUMBER(k100)
= 2;
# q106
select count(*) from sq11knum where TO_NUMBER(k25) =
2;
# q107
select count(*) from sq11knum where TO_NUMBER(k10) =
2;
# q108
select count(*) from sq11knum where TO_NUMBER(k5) =
2;
# q109
select count(*) from sq11knum where TO_NUMBER(k4) =
2;
# q110
select count(*) from sq11knum where TO_NUMBER(k2) =
2;
# q201
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(kseq) = 3;
# q202
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k100k) = 3;
# q203
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k10k) = 3;
# q204
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k1k) = 3;
# q205
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k100) = 3;
# q206
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k25) = 3;
# q207
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k10) = 3;
# q208
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k5) = 3;
# q209
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k4) = 3;
# q210
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(kseq) <> 3;
# q211
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k100k) <> 3;
# q212
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k10k) <> 3;
# q213
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k1k) <> 3;
# q214
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k100) <> 3;
# q215
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k25) <> 3;
# q216
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k10) <> 3;
# q217
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k5) <> 3;
# q218
select count(*) from sq11knum where TO_NUMBER(k2) =
2 and TO_NUMBER(k4) <> 3;
# q301
select sum(TO_NUMBER(k1k)) from sq11knum where
TO_NUMBER(k100k) = 3;
# q302
select sum(TO_NUMBER(k1k)) from sq11knum where
TO_NUMBER(k10k) = 3;
# q303
select sum(TO_NUMBER(k1k)) from sq11knum where
TO_NUMBER(k1k) = 3;
# q304
select sum(TO_NUMBER(k1k)) from sq11knum where
TO_NUMBER(k100) = 3;
# q305
select sum(TO_NUMBER(k1k)) from sq11knum where
TO_NUMBER(k25) = 3;
# q306
select sum(TO_NUMBER(k1k)) from sq11knum where
TO_NUMBER(k10) = 3;
# q307
select sum(TO_NUMBER(k1k)) from sq11knum where
TO_NUMBER(k5) = 3;
# q308
select sum(TO_NUMBER(k1k)) from sq11knum where
TO_NUMBER(k4) = 3;
# q309
select sum(TO_NUMBER(k1k)) FROM sq11knum
WHERE (TO_NUMBER(kseq) BETWEEN 0 AND 25000 OR
TO_NUMBER(kseq) BETWEEN 50000 AND 75000 OR
TO_NUMBER(kseq) BETWEEN 100000 AND 125000 OR
      TO_NUMBER(kseq) BETWEEN 150000 AND 175000 OR
      TO_NUMBER(kseq) BETWEEN 200000 AND 225000)
      AND TO_NUMBER(k100k) = 3;
# q310
select sum(TO_NUMBER(k1k)) FROM sq11knum
WHERE (TO_NUMBER(kseq) BETWEEN 0 AND 25000 OR
TO_NUMBER(kseq) BETWEEN 50000 AND 75000 OR
TO_NUMBER(kseq) BETWEEN 100000 AND 125000 OR
      TO_NUMBER(kseq) BETWEEN 150000 AND 175000 OR
      TO_NUMBER(kseq) BETWEEN 200000 AND 225000)
      AND TO_NUMBER(k10k) = 3;
# q311
select sum(TO_NUMBER(k1k)) FROM sq11knum
WHERE (TO_NUMBER(kseq) BETWEEN 0 AND 25000 OR
TO_NUMBER(kseq) BETWEEN 50000 AND 75000 OR
TO_NUMBER(kseq) BETWEEN 100000 AND 125000 OR
      TO_NUMBER(kseq) BETWEEN 150000 AND 175000 OR
      TO_NUMBER(kseq) BETWEEN 200000 AND 225000)
      AND TO_NUMBER(k1k) = 3;
# q312
select sum(TO_NUMBER(k1k)) FROM sq11knum
WHERE (TO_NUMBER(kseq) BETWEEN 0 AND 25000 OR
TO_NUMBER(kseq) BETWEEN 50000 AND 75000 OR
TO_NUMBER(kseq) BETWEEN 100000 AND 125000 OR
      TO_NUMBER(kseq) BETWEEN 150000 AND 175000 OR
      TO_NUMBER(kseq) BETWEEN 200000 AND 225000)
      AND TO_NUMBER(k100) = 3;
# q313
select sum(TO_NUMBER(k1k)) FROM sq11knum
WHERE (TO_NUMBER(kseq) BETWEEN 0 AND 25000 OR
TO_NUMBER(kseq) BETWEEN 50000 AND 75000 OR
TO_NUMBER(kseq) BETWEEN 100000 AND 125000 OR
      TO_NUMBER(kseq) BETWEEN 150000 AND 175000 OR
      TO_NUMBER(kseq) BETWEEN 200000 AND 225000)
      AND TO_NUMBER(k25) = 3;
# q314
select sum(TO_NUMBER(k1k)) FROM sq11knum
WHERE (TO_NUMBER(kseq) BETWEEN 0 AND 25000 OR
TO_NUMBER(kseq) BETWEEN 50000 AND 75000 OR
TO_NUMBER(kseq) BETWEEN 100000 AND 125000 OR
      TO_NUMBER(kseq) BETWEEN 150000 AND 175000 OR
```

```
      TO_NUMBER(kseq) BETWEEN 200000 AND 225000)
      AND TO_NUMBER(k10) = 3;
# q315
select sum(TO_NUMBER(k1k)) FROM sq11knum
WHERE (TO_NUMBER(kseq) BETWEEN 0 AND 25000 OR
TO_NUMBER(kseq) BETWEEN 50000 AND 75000 OR
TO_NUMBER(kseq) BETWEEN 100000 AND 125000 OR
      TO_NUMBER(kseq) BETWEEN 150000 AND 175000 OR
      TO_NUMBER(kseq) BETWEEN 200000 AND 225000)
      AND TO_NUMBER(k5) = 3;
# q316
select sum(TO_NUMBER(k1k)) FROM sq11knum
WHERE (TO_NUMBER(kseq) BETWEEN 0 AND 25000 OR
TO_NUMBER(kseq) BETWEEN 50000 AND 75000 OR
TO_NUMBER(kseq) BETWEEN 100000 AND 125000 OR
      TO_NUMBER(kseq) BETWEEN 150000 AND 175000 OR
      TO_NUMBER(kseq) BETWEEN 200000 AND 225000)
      AND TO_NUMBER(k4) = 3;
# q401
select kseq, k500k from sq11knum where TO_NUMBER(k2)
= 1 and
TO_NUMBER(k100) > 80 and
TO_NUMBER(k10k) between 2000 and 3000;
# q402
select kseq, k500k from sq11knum where
TO_NUMBER(k100) > 80 and
TO_NUMBER(k10k) between 2000 and 3000
and TO_NUMBER(k5) =3;
# q403
select kseq, k500k from sq11knum where
TO_NUMBER(k10k) between 2000 and 3000
and TO_NUMBER(k5) =3
and (TO_NUMBER(k25) = 11 or TO_NUMBER(k25) = 9);
# q404
select kseq, k500k from sq11knum where TO_NUMBER(k5)
= 3
and (TO_NUMBER(k25) = 11 or TO_NUMBER(k25) = 9)
and TO_NUMBER(k4) = 3;
# q405
select kseq, k500k from sq11knum
where (TO_NUMBER(k25) = 11 or TO_NUMBER(k25) = 9)
and
TO_NUMBER(k4) = 3 and
TO_NUMBER(k100) < 41;
# q406
SELECT kseq, k500k FROM sq11knum
WHERE TO_NUMBER(k4) = 3 AND
TO_NUMBER(k100) < 41 AND
TO_NUMBER(k1k) BETWEEN 850 AND 950;
# q407
SELECT kseq, k500k FROM sq11knum
WHERE TO_NUMBER(k100) < 41 AND
TO_NUMBER(k1k) BETWEEN 850 AND 950 AND
TO_NUMBER(k10) = 7;
# q408
SELECT kseq, k500k FROM sq11knum
WHERE TO_NUMBER(k1k) BETWEEN 850 AND 950 AND
TO_NUMBER(k10) = 7 AND
TO_NUMBER(k25) BETWEEN 3 AND 4;
# q409
SELECT kseq, k500k FROM sq11knum
WHERE TO_NUMBER(k2) = 1 AND
TO_NUMBER(k100) > 80  AND
TO_NUMBER(k10k) BETWEEN 2000 AND 3000 AND
TO_NUMBER(k5) = 3 AND
(TO_NUMBER(k25) = 11 OR TO_NUMBER(k25) = 9);
# q410
SELECT kseq, k500k FROM sq11knum
WHERE TO_NUMBER(k100) > 80 AND
TO_NUMBER(k10k) BETWEEN 2000 AND 3000 AND
TO_NUMBER(k5) = 3 AND
(TO_NUMBER(k25) = 11 OR TO_NUMBER(k25) = 9)
AND TO_NUMBER(k4) = 3;
# q411
SELECT kseq, k500k FROM sq11knum
WHERE TO_NUMBER(k10k) BETWEEN 2000 AND 3000 AND
TO_NUMBER(k5) = 3 AND
(TO_NUMBER(k25) = 11 OR TO_NUMBER(k25) = 9) AND
TO_NUMBER(k4) = 3 AND
TO_NUMBER(k100) < 41;
# q412
SELECT kseq, k500k FROM sq11knum
WHERE TO_NUMBER(k5) = 3 AND
(TO_NUMBER(k25) = 11 OR TO_NUMBER(k25) = 9) AND
TO_NUMBER(k4) = 3 AND
TO_NUMBER(k100) < 41 AND
TO_NUMBER(k1k) BETWEEN 850 AND 950;
# q413
SELECT kseq, k500k FROM sq11knum
WHERE (TO_NUMBER(k25) = 11 OR TO_NUMBER(k25) = 9)
AND
TO_NUMBER(k4) = 3 AND TO_NUMBER(k100) < 41 AND
TO_NUMBER(k1k) BETWEEN 850 AND 950 AND
TO_NUMBER(k10) = "7";
# q414
SELECT kseq, k500k FROM sq11knum
WHERE TO_NUMBER(k4) = 3 AND
TO_NUMBER(k100) < 41 AND
TO_NUMBER(k1k) BETWEEN 850 AND 950 AND
TO_NUMBER(k10) = 7 AND
TO_NUMBER(k1k) BETWEEN 3 AND 4;
# q501
select k2, k100, count(*) from sq11knum group by k2,
k100;
# q502
select k4, k25, count(*) from sq11knum group by k4,
k25;
# q503
select k10, k25, count(*) from sq11knum group by
k10, k25;
```

### MongoDB Queries

```
use mydb;
var newDate = new Date();print("STARTED",newDate);
print("q101");
db.things.find({ k250k: 2 }).count();
print("q102");
db.things.find({ k100k: 2 }).count();
print("q103");
db.things.find({ k10k: 2 }).count();
print("q104")
db.things.find({ k1k: 2 }).count();
print("q105")
db.things.find({ k100: 2 }).count();
print("q106")
db.things.find({ k25: 2 }).count();
print("q107")
db.things.find({ k10: 2 }).count();
print("q108")
db.things.find({ k5: 2 }).count();
print("q109")
db.things.find({ k4: 2 }).count();
print("q110")
db.things.find({ k2: 2 }).count();
print("q201")
db.things.find({ k2: 2, kseq: 3 }).count();
print("q202")
db.things.find({ k2: 2, k100k: 3 }).count();
print("q203")
db.things.find({ k2: 2, k10k: 3 }).count();
print("q204")
db.things.find({ k2: 2, k1k: 3 }).count();
print("q205")
db.things.find({ k2: 2, k100: 3 }).count();
print("q206")
db.things.find({ k2: 2, k25: 3 }).count();
print("q207")
db.things.find({ k2: 2, k10: 3 }).count();
print("q208")
db.things.find({ k2: 2, k5: 3 }).count();
print("q209")
db.things.find({ k2: 2, k4: 3 }).count();
print("q210")
db.things.find({ k2: 2, kseq: { $ne: 3} }).count();
print("q211")
db.things.find({ k2: 2, k100k: { $ne: 3} }).count();
```

```
print("q212")
db.things.find({ k2: 2, k10k: { $ne: 3} }).count();
print("q213")
db.things.find({ k2: 2, k1k: { $ne: 3} }).count();
print("q214")
db.things.find({ k2: 2, k100: { $ne: 3} }).count();
print("q215")
db.things.find({ k2: 2, k25: { $ne: 3} }).count();
print("q216")
db.things.find({ k2: 2, k10: { $ne: 3} }).count();
print("q217")
db.things.find({ k2: 2, k5: { $ne: 3} }).count();
print("q218")
db.things.find({ k2: 2, k4: { $ne: 3} }).count();
print("q301")
db.things.aggregate({ $match: {     $and: [
{ k100k: 3}     ] } }, { $group: { _id : null, sum :
{ $sum: "$k1k" } } });
print("q302")
db.things.aggregate({ $match: {     $and: [
{ k10k: 3}     ] } }, { $group: { _id : null, sum :
{ $sum: "$k1k" } } });
print("q303")
db.things.aggregate({ $match: {     $and: [
{ k1k: 3}     ] } }, { $group: { _id : null, sum : {
$sum: "$k1k" } } });
print("q304")
db.things.aggregate({ $match: {     $and: [
{ k100: 3}     ] } }, { $group: { _id : null, sum :
{ $sum: "$k1k" } } });
print("q305")
db.things.aggregate({ $match: {     $and: [
{ k25: 3}     ] } }, { $group: { _id : null, sum : {
$sum: "$k1k" } } });
print("q306")
db.things.aggregate({ $match: {     $and: [
{ k10: 3}     ] } }, { $group: { _id : null, sum : {
$sum: "$k1k" } } });
print("q307")
db.things.aggregate({ $match: {     $and: [
{ k5: 3}     ] } }, { $group: { _id : null, sum : {
$sum: "$k1k" } } });
print("q308")
db.things.aggregate({ $match: {     $and: [
{ k4: 3}     ] } }, { $group: { _id : null, sum : {
$sum: "$k1k" } } });
print("q309")
db.things.aggregate({ $match: { k100k: 3,
    $or: [
{ $and : [ { kseq: { $gt: 0 } },  { kseq: { $lt:
25000 } } ]},
{ $and : [ { kseq: { $gt: 50000 } },  { kseq: { $lt:
75000 } } ]},
{ $and : [ { kseq: { $gt: 100000 } },  { kseq: {
$lt:  125000 } } ]},
{ $and : [ { kseq: { $gt: 150000 } },  { kseq: {
$lt:  175000 } } ]},
{ $and : [ { kseq: { $gt: 200000 } },  { kseq: {
$lt:  225000 } } ]}
]
} },
{ $group: { _id : null, sum : { $sum: "$k1k" } } });
print("q310")
db.things.aggregate({ $match: { k10k: 3,
    $or: [
{ $and : [ { kseq: { $gt: 0 } },  { kseq: { $lt:
25000 } } ]},
{ $and : [ { kseq: { $gt: 50000 } },  { kseq: { $lt:
75000 } } ]},
{ $and : [ { kseq: { $gt: 100000 } },  { kseq: {
$lt:  125000 } } ]},
{ $and : [ { kseq: { $gt: 150000 } },  { kseq: {
$lt:  175000 } } ]},
{ $and : [ { kseq: { $gt: 200000 } },  { kseq: {
$lt:  225000 } } ]}
]
} },
{ $group: { _id : null, sum : { $sum: "$k1k" } } });
print("q311")
db.things.aggregate({ $match: { k1k: 3,
    $or: [
{ $and : [ { kseq: { $gt: 0 } },  { kseq: { $lt:
25000 } } ]},
{ $and : [ { kseq: { $gt: 50000 } },  { kseq: { $lt:
75000 } } ]},
{ $and : [ { kseq: { $gt: 100000 } },  { kseq: {
$lt:  125000 } } ]},
{ $and : [ { kseq: { $gt: 150000 } },  { kseq: {
$lt:  175000 } } ]},
{ $and : [ { kseq: { $gt: 200000 } },  { kseq: {
$lt:  225000 } } ]}
]
} },
{ $group: { _id : null, sum : { $sum: "$k1k" } } });
print("q312")
db.things.aggregate({ $match: { k100: 3,
    $or: [
{ $and : [ { kseq: { $gt: 0 } },  { kseq: { $lt:
25000 } } ]},
{ $and : [ { kseq: { $gt: 50000 } },  { kseq: { $lt:
75000 } } ]},
{ $and : [ { kseq: { $gt: 100000 } },  { kseq: {
$lt:  125000 } } ]},
{ $and : [ { kseq: { $gt: 150000 } },  { kseq: {
$lt:  175000 } } ]},
{ $and : [ { kseq: { $gt: 200000 } },  { kseq: {
$lt:  225000 } } ]}
]
} },
{ $group: { _id : null, sum : { $sum: "$k1k" } } });
print("q313")
db.things.aggregate({ $match: { k25: 3,
    $or: [
{ $and : [ { kseq: { $gt: 0 } },  { kseq: { $lt:
25000 } } ]},
{ $and : [ { kseq: { $gt: 50000 } },  { kseq: { $lt:
75000 } } ]},
{ $and : [ { kseq: { $gt: 100000 } },  { kseq: {
$lt:  125000 } } ]},
{ $and : [ { kseq: { $gt: 150000 } },  { kseq: {
$lt:  175000 } } ]},
{ $and : [ { kseq: { $gt: 200000 } },  { kseq: {
$lt:  225000 } } ]}
]
} },
{ $group: { _id : null, sum : { $sum: "$k1k" } } });
print("q314")
db.things.aggregate({ $match: { k10: 3,
    $or: [
{ $and : [ { kseq: { $gt: 0 } },  { kseq: { $lt:
25000 } } ]},
{ $and : [ { kseq: { $gt: 50000 } },  { kseq: { $lt:
75000 } } ]},
{ $and : [ { kseq: { $gt: 100000 } },  { kseq: {
$lt:  125000 } } ]},
{ $and : [ { kseq: { $gt: 150000 } },  { kseq: {
$lt:  175000 } } ]},
{ $and : [ { kseq: { $gt: 200000 } },  { kseq: {
$lt:  225000 } } ]}
]
} },
{ $group: { _id : null, sum : { $sum: "$k1k" } } });
print("q315")
db.things.aggregate({ $match: { k5: 3,
    $or: [
{ $and : [ { kseq: { $gt: 0 } },  { kseq: { $lt:
25000 } } ]},
{ $and : [ { kseq: { $gt: 50000 } },  { kseq: { $lt:
75000 } } ]},
{ $and : [ { kseq: { $gt: 100000 } },  { kseq: {
$lt:  125000 } } ]},
{ $and : [ { kseq: { $gt: 150000 } },  { kseq: {
$lt:  175000 } } ]},
{ $and : [ { kseq: { $gt: 200000 } },  { kseq: {
$lt:  225000 } } ]}
]
```

```
} },
{ $group: { _id : null, sum : { $sum: "$k1k" } } });
print("q316")
db.things.aggregate({ $match: { k4: 3,
    $or: [
{ $and : [ { kseq: { $gt: 0 } },  { kseq: { $lt:
25000 } } ]},
{ $and : [ { kseq: { $gt: 50000 } },  { kseq: { $lt:
75000 } } ]},
{ $and : [ { kseq: { $gt: 100000 } },  { kseq: {
$lt:  125000 } } ]},
{ $and : [ { kseq: { $gt: 150000 } },  { kseq: {
$lt:  175000 } } ]},
{ $and : [ { kseq: { $gt: 200000 } },  { kseq: {
$lt:  225000 } } ]}
]
} },
{ $group: { _id : null, sum : { $sum: "$k1k" } } });
print("q401")
db.things.find(
    { k2: 1, k100: { $gt: 80}, k10k: { $gt: 2000},
k10k: { $lt: 3000}},

    { kseq: 1, k500k: 1, _id: 0 }
)
print("q402")
db.things.find(
    { k5: 3, k100: { $gt: 80}, k10k: { $gt: 2000},
k10k: { $lt: 3000}},

    { kseq: 1, k500k: 1, _id: 0 }
)
print("q403")

db.things.find(     { k5: 3,  k10k: { $gt: 2000},
k10k: { $lt: 3000},
     $and: [     { $or : [ {k25: 11} , {  k25: 9 }
]}   ]},
     { kseq: 1, k500k: 1, _id: 0 } )
print("q404")
db.things.find(     { k5: 3,  k4: 3,
     $and: [     { $or : [ {k25: 11} , {  k25: 9 }
]}   ]},
     { kseq: 1, k500k: 1, _id: 0 } )
print("q405")
db.things.find(     { k4: 3,  k100: { $lt: 41},
     $and: [     { $or : [ {k25: 11} , {  k25: 9 }
]}   ]},
     { kseq: 1, k500k: 1, _id: 0 } )
print("q406")
db.things.find(     { k4: 3,  k100: { $lt: 41}, k1k:
{ $gt: 850}, k1k: { $lt: 950}},
     { kseq: 1, k500k: 1, _id: 0 } )
print("q407")
db.things.find(     { k10: 7,  k100: { $lt: 41},
k1k: { $gt: 850}, k1k: { $lt: 950}},
     { kseq: 1, k500k: 1, _id: 0 } )
print("q408")
db.things.find(     { k10: 7,  k100: { $lt: 41},
k1k: { $gt: 850}, k1k: { $lt: 950}, k25: { $gt: 3},
k25: { $lt: 4}},
     { kseq: 1, k500k: 1, _id: 0 } )
print("q409")
db.things.find(     { k2: 1,  k100: { $gt: 80},
k10k: { $gt: 2000}, k10k: { $lt: 3000}, k5: 3,
     $and: [     { $or : [ {k25: 11} , {  k25: 9 }
]}   ]},
     { kseq: 1, k500k: 1, _id: 0 } )
print("q410")
db.things.find(     { k4: 3,  k100: { $gt: 80},
k10k: { $gt: 2000}, k10k: { $lt: 3000}, k5: 3,
     $and: [     { $or : [ {k25: 11} , {  k25: 9 }
]}   ]},
     { kseq: 1, k500k: 1, _id: 0 } )
print("q411")
db.things.find(     { k4: 3,  k100: { $lt: 41},
k10k: { $gt: 2000}, k10k: { $lt: 3000}, k5: 3,
     $and: [     { $or : [ {k25: 11} , {  k25: 9 }
]}   ]},
     { kseq: 1, k500k: 1, _id: 0 } )
print("q412")
db.things.find(     { k4: 3,  k100: { $lt: 41}, k1k:
{ $gt: 850}, k1k: { $lt: 950}, k5: 3,
     $and: [     { $or : [ {k25: 11} , {  k25: 9 }
]}   ]},
     { kseq: 1, k500k: 1, _id: 0 } )
print("q413")
db.things.find(     { k4: 3,  k100: { $lt: 41}, k1k:
{ $gt: 850}, k1k: { $lt: 950}, k10: 7,
     $and: [     { $or : [ {k25: 11} , {  k25: 9 }
]}   ]},
     { kseq: 1, k500k: 1, _id: 0 } )
print("q414")
db.things.find(     { k10: 7,  k100: { $lt: 41},
k1k: { $gt: 850}, k1k: { $lt: 950}, k4: 3, k25: {
$lt: 4}, k1k: { $gt: 3}, k1k: { $lt: 4}},
     { kseq: 1, k500k: 1, _id: 0 } )
print("q501")
db.things.group(
   {
     key: { k2: 1, k100: 1 },

     reduce: function( curr, result ) {
                result.count ++;
            },
     initial: {  count : 0}
   }
)

print("q502")
db.things.group(
   {
     key: { k4: 1, k25: 1 },

     reduce: function( curr, result ) {
                result.count ++;
            },
     initial: {  count : 0}
   }
)

print("q503")
db.things.group(
   {
     key: { k10: 1, k25: 1 },

     reduce: function( curr, result ) {
                result.count ++;
            },
     initial: {  count : 0}
   }
)

var newDate = new Date();print("ENDED",newDate);
```

## Appendix C – Test data generator GENHBASE.CPP

```
//================================================
==========================
// Name        : genhbase.cpp
// Author      : djy
// Version     :
// Copyright   : Your copyright notice
// Description : Hello World in C++, Ansi-style
//================================================
==========================

#include <iostream>
#include <stdio.h>
#include <stdlib.h>
using namespace std;
int main() {
cout << "!!!genhbase start!!!" << endl; // prints
!!!genhbase start!!
FILE *stream;
int kseqint = 1; // no need to initialize, the do
loop does
long long krand;
long long k1b;
long long k500m;
long long k100m;
long long k10m;
long long k5m;
long long k2m;
long long k1m;
long long k500k;
long long k250k;
long long k100k;
long long k40k;
long long k10k;
long long k1k;
long long k100;
long long k25;
long long k10;
long long k5;
long long k4;
long long k2;
char sortbin[] = "12345678";
char sortpack[] = "12345678";
char sortchar[] = "12345678";
char s4[] = "12345678901234567890123456789O";
char s5[] = "12345678901234567890123456789O";
char s6[] = "12345678901234567890123456789O";
char s7[] = "1234567890123456789O";
char s8[] = "123456789012345678901";
long long kpart = 1;
srand(1);
/* Open file in text mode: */
if( (stream = fopen( "genhbase.1M.p01.20170302.csv",
"w+t" )) != NULL )
        {
        fprintf(stream,
"hbase_row_key,kpart,kseq,k1b,k500m,k100m,k10m,k5m,k
2m,k1m,k500k,k250k,k100k,k40k,k10k,k1k,k100,k25,k10,
k5,k4,k2,sortbin,sortpack,sortchar,s4,s5,s6,s7,s8\n"
);
        for ( kseqint = 0; kseqint < 1000000;
kseqint++ )
        {
                krand = abs (rand());
                k1b = (krand*.5000000000) +
(krand*.03000000000);
                k500m = (krand*.2500000000) +
(krand*.01500000000);
                k100m = (krand*.0500000000) +
(krand*.00300000000);
                k10m = (krand*.0050000000) +
(krand*.00030000000);
                k5m = (krand*.0025000000) +
(krand*.00015000000);
                k2m = (krand*.0010000000) +
(krand*.00006000000);
                k1m = (krand*.0005000000) +
(krand*.00003000000);
                k500k = (krand*.0002500000) +
(krand*.00001500000);
                k250k = (krand*.0001250000) +
(krand*.00000750000);
                k100k = (krand*.0000500000) +
(krand*.00000300000);
                k40k = (krand*.0000200000) +
(krand*.00000120000);
                k10k = (krand*.0000050000) +
(krand*.00000030000);
                k1k = (krand*.0000005000) +
(krand*.00000003000);
                k100 = (krand*.0000000500) +
(krand*.00000000300);
                k25 = (krand*.0000000125) +
(krand*.00000000060);
                k10 = (krand*.0000000050) +
(krand*.00000000025);
                k5 = (krand*.0000000025) +
(krand*.00000000012);
                k4 = (krand*.0000000020) +
(krand*.00000000010);
                k2 = (krand*.00000000125) +
(krand*.00000000006);
/* Write record */
                fprintf(stream,
"%11d,%lld,%lld,%lld,%lld,%lld,%lld,%lld,%lld,%lld,%
lld,%lld,%lld,%lld,%lld,%lld,%lld,%lld,%lld,%lld,%ll
d,%lld,%s,%s,%s,%s,%s,%s,%s,%s\n",
                                kseqint,
                                kpart,
                                kseqint,
                                k1b,
                                k500m,
                                k100m,
                                k10m,
                                k5m,
                                k2m,
                                k1m,
                                k500k,
                                k250k,
                                k100k,
                                k40k,
                                k10k,
                                k1k,
                                k100,
                                k25,
                                k10,
                                k5,
                                k4,
                                k2,
                                sortbin,
                                sortpack,
                                sortchar,
                                s4,
                                s5,
                                s6,
                                s7,
                                s8
                        );
                }
                /* close file */
                fclose( stream );
        }
        else
                printf( "Problem opening the file\n"
);
                cout << "!!!gendb2 end!!!" << endl;
// prints !!!gendb2 end!!
}
```

| mongodb 10M | | | | | | |
|---|---|---|---|---|---|---|
| Function / Call Stack | Effective Ti | Spin Time | Overhead T | Instruction | CPI Rate | CPU Freque |
| mwait_idle_with_hints | 157.402 | 0 | 0 | 9.72E+09 | 66.2963 | 1.13723 |
| [Outside any known module] | 89.001 | 0 | 0 | 4.62E+11 | 0.737129 | 1.0618 |
| __wt_row_search | 36.8004 | 0 | 0 | 1.02E+11 | 1.45035 | 1.11141 |
| __switch_to | 34.3004 | 0 | 0 | 5.11E+10 | 2.40845 | 0.997085 |
| native_write_msr_safe | 34.1004 | 0 | 0 | 2.95E+10 | 4.37805 | 1.05279 |
| __schedule | 0 | 27.4003 | 0 | 1.01E+11 | 0.971429 | 0.992701 |
| mongo::BSONElement::size | 22.3002 | 0 | 0 | 2.30E+11 | 0.412226 | 1.17937 |
| cpu_startup_entry | 17.7002 | 0 | 0 | 4.07E+10 | 1.85841 | 1.18644 |
| futex_wake_op | 17.1002 | 0 | 0 | 3.89E+10 | 1.63889 | 1.03509 |
| pthread_cond_wait | 0 | 16.3002 | 0 | 2.99E+10 | 2.73494 | 1.39264 |
| _raw_spin_lock_irqsave | 15.3002 | 0 | 0 | 6.66E+10 | 0.854054 | 1.03268 |
| mongo::BSONObj::getField | 14.8002 | 0 | 0 | 1.09E+11 | 0.450331 | 0.918919 |
| _raw_spin_lock | 13.8002 | 0 | 0 | 4E+10 | 1.20721 | 0.971014 |
| resched_task | 12.9001 | 0 | 0 | 1.37E+10 | 4.23684 | 1.24806 |
| __strlen_sse2_pminub | 12.2001 | 0 | 0 | 1.32E+11 | 0.455041 | 1.36885 |
| menu_select | 10.9001 | 0 | 0 | 1.00E+11 | 0.498208 | 1.27523 |
| native_read_tsc | 9.80011 | 0 | 0 | 9.29E+10 | 0.465116 | 1.22449 |
| wake_futex | 9.80011 | 0 | 0 | 1.15E+10 | 3.0625 | 1 |
| _raw_spin_unlock_irqrestore | 9.5001 | 0 | 0 | 7.06E+10 | 0.515306 | 1.06316 |
| [h"] | 9.4001 | 0 | 0 | 3.28E+10 | 1.30769 | 1.26596 |
| task_waking_fair | 9.2001 | 0 | 0 | 3.96E+09 | 11.3636 | 1.3587 |
| futex_wait | 9.0001 | 0 | 0 | 2.81E+10 | 1.26923 | 1.1 |
| finish_task_switch | 8.50009 | 0 | 0 | 1.04E+10 | 3.37931 | 1.15294 |
| hash_futex | 7.80009 | 0 | 0 | 8.93E+10 | 0.407258 | 1.29487 |
| ktime_get | 7.60008 | 0 | 0 | 2.66E+10 | 0.932432 | 0.907895 |
| system_call | 7.60008 | 0 | 0 | 1.44E+10 | 2.25 | 1.18421 |
| __wt_page_in_func | 7.10008 | 0 | 0 | 8.64E+09 | 3.20833 | 1.08451 |
| pthread_cond_signal | 0 | 7.00008 | 0 | 2.23E+10 | 1.17742 | 1.04286 |
| timerqueue_add | 6.60007 | 0 | 0 | 3.42E+10 | 0.536842 | 0.772727 |
| select_task_rq_fair | 6.10007 | 0 | 0 | 4.25E+10 | 0.559322 | 1.08197 |
| __unqueue_futex | 5.90006 | 0 | 0 | 1.33E+10 | 1.67568 | 1.05085 |
| _raw_spin_lock_irq | 5.80006 | 0 | 0 | 1.62E+10 | 1.66667 | 1.2931 |
| system_call_after_swapgs | 5.80006 | 0 | 0 | 2.52E+09 | 8.57143 | 1.03448 |
| futex_wait_queue_me | 5.60006 | 0 | 0 | 1.8E+10 | 1.02 | 0.910714 |
| __hrtimer_start_range_ns | 5.50006 | 0 | 0 | 1.91E+10 | 1.35849 | 1.30909 |
| pick_next_task_fair | 5.50006 | 0 | 0 | 3.53E+10 | 0.693878 | 1.23636 |
| __wt_btcur_next | 5.40006 | 0 | 0 | 3.89E+10 | 0.453704 | 0.907407 |
| __memcpy_ssse3_back | 5.20006 | 0 | 0 | 1.66E+10 | 1.02174 | 0.903846 |
| pthread_mutex_unlock | 0 | 5.00005 | 0 | 1.26E+10 | 1.57143 | 1.1 |
| __pthread_mutex_lock | 0 | 4.70005 | 0 | 1.37E+10 | 1 | 0.808511 |
| mongo::compareElementValues | 4.60005 | 0 | 0 | 4E+10 | 0.261261 | 0.630435 |
| copy_user_enhanced_fast_string | 4.50005 | 0 | 0 | 4.32E+09 | 4.91667 | 1.31111 |
| current_set_polling_and_test | 4.30005 | 0 | 0 | 3.24E+09 | 4.11111 | 0.860465 |
| __audit_syscall_exit | 4.20005 | 0 | 0 | 4.9E+10 | 0.345588 | 1.11905 |
| nr_iowait_cpu | 4.20005 | 0 | 0 | 2.09E+10 | 1.2069 | 1.66667 |

| mongodb 10M | | | | | | |
|---|---|---|---|---|---|---|
| Process / Module / Function / Thread / | Effective Ti | Spin Time | Overhead T | Instruction | CPI Rate | CPU Freque |
| mongod | 474.605 | 50.9006 | 0 | 2.84E+12 | 0.754248 | 1.13187 |
| | 377.204 | 16.9002 | 0 | 9.05E+11 | 1.77119 | 1.12941 |
| gnome-shell | 110.501 | 0.400004 | 0 | 5.73E+11 | 0.730358 | 1.04779 |
| amplxe-gui | 60.2007 | 1.90002 | 0 | 2.01E+11 | 1.31127 | 1.18035 |
| X | 4.10004 | 0 | 0 | 1.94E+10 | 0.925926 | 1.21951 |
| firefox | 1.40002 | 0.400004 | 0 | 7.2E+08 | 8 | 0.888889 |
| gnome-terminal- | 0.90001 | 0 | 0 | 2.52E+09 | 2.14286 | 1.66667 |
| nmon | 0.800009 | 0 | 0 | 7.56E+09 | 1 | 2.625 |
| MongoDB Compass | 0.700008 | 0 | 0 | 2.52E+09 | 1.14286 | 1.14286 |
| avahi-daemon | 0.200002 | 0 | 0 | 3.6E+08 | 0 | 0 |
| udisksd | 0.100001 | 0 | 0 | 0 | 0 | 0 |
| tuned | 0.100001 | 0 | 0 | 0 | 0 | 0 |
| rcu_sched | 0.100001 | 0 | 0 | 7.2E+08 | 0 | 0 |
| MongoDB Compass | 0.100001 | 0 | 0 | 0 | 0 | 0 |
| xfsaild/sda4 | 0.100001 | 0 | 0 | 0 | | 1 |
| mongo | 0.100001 | 0 | 0 | 0 | 0 | 0 |
| amplxe-runss | 0.100001 | 0 | 0 | 0 | 0 | 0 |
| amplxe-runss | 0 | 0 | 0 | 7.2E+08 | 0.5 | |
| bash | 0 | 0 | 0 | 3.6E+08 | 0 | 0 |
| awk | 0 | 0 | 0 | 3.6E+08 | 0 | 0 |
| kswapd0 | 0 | 0 | 0 | 3.6E+08 | 0 | 0 |

| mongodb 100M | | | | | | |
|---|---|---|---|---|---|---|
| Function / Call Stack | Effective Ti | Spin Time | Overhead T | Instruction | CPI Rate | CPU Freque |
| snappy::InternalUncompress<snappy:: | 3473.74 | 0 | 0 | 3.78E+13 | 0.380986 | 1.15171 |
| mwait_idle_with_hints | 1553.02 | 0 | 0 | 1.28E+11 | 49.0169 | 1.12048 |
| [Outside any known module] | 676.307 | 0 | 0 | 3.40E+12 | 0.742712 | 1.03593 |
| native_write_msr_safe | 331.704 | 0 | 0 | 3.02E+11 | 4.68974 | 1.18481 |
| __switch_to | 324.204 | 0 | 0 | 6.07E+11 | 2.19703 | 1.14189 |
| __wt_row_search | 317.503 | 0 | 0 | 1.16E+12 | 1.15571 | 1.16882 |
| __wt_checksum_hw | 294.903 | 0 | 0 | 2.01E+12 | 0.60372 | 1.14479 |
| copy_user_enhanced_fast_string | 271.203 | 0 | 0 | 8.03E+10 | 14.2063 | 1.16814 |
| __wt_cell_unpack_safe.constprop.14 | 263.103 | 0 | 0 | 4.16E+12 | 0.264859 | 1.16192 |
| __schedule | 0 | 247.603 | 0 | 1.03E+12 | 0.995791 | 1.14661 |
| mongo::BSONElement::size | 225.902 | 0 | 0 | 2.34E+12 | 0.390997 | 1.1266 |
| pthread_cond_wait | 0 | 184.002 | 0 | 2.91E+11 | 2.64312 | 1.15924 |
| cpu_startup_entry | 172.302 | 0 | 0 | 3.86E+11 | 1.83287 | 1.13929 |
| _raw_spin_lock_irqsave | 165.602 | 0 | 0 | 7.57E+11 | 0.867808 | 1.10205 |
| resched_task | 162.702 | 0 | 0 | 1.31E+11 | 4.92287 | 1.09834 |
| futex_wake_op | 159.102 | 0 | 0 | 4.25E+11 | 1.52288 | 1.12948 |
| _raw_spin_lock | 147.802 | 0 | 0 | 4.13E+11 | 1.40192 | 1.08796 |
| operator delete | 128.901 | 0 | 0 | 3.71E+11 | 1.46751 | 1.17378 |
| mongo::BSONObj::getField | 128.101 | 0 | 0 | 1.24E+12 | 0.421997 | 1.13817 |
| menu_select | 123.301 | 0 | 0 | 9.49E+11 | 0.516875 | 1.10543 |
| __strlen_sse2_pminub | 120.201 | 0 | 0 | 1.28E+12 | 0.394449 | 1.17055 |
| __wt_page_inmem | 103.801 | 0 | 0 | 1.39E+12 | 0.321955 | 1.19942 |
| __tree_walk_internal | 101.201 | 0 | 0 | 1.95E+11 | 2.14365 | 1.1502 |
| native_read_tsc | 100.901 | 0 | 0 | 9.37E+11 | 0.455803 | 1.17542 |
| __wt_hazard_check | 94.201 | 0 | 0 | 2.15E+11 | 1.7802 | 1.12633 |
| _raw_spin_unlock_irqrestore | 92.001 | 0 | 0 | 6.51E+11 | 0.610957 | 1.2 |
| finish_task_switch | 90.201 | 0 | 0 | 9.04E+10 | 4.03187 | 1.12195 |
| hash_futex | 84.4009 | 0 | 0 | 9.33E+11 | 0.353009 | 1.08412 |
| wake_futex | 84.3009 | 0 | 0 | 1.12E+11 | 3.27331 | 1.20759 |
| task_waking_fair | 83.7009 | 0 | 0 | 5.18E+10 | 6.77083 | 1.16487 |
| futex_wait | 82.1009 | 0 | 0 | 2.97E+11 | 1.13956 | 1.14373 |
| pick_next_task_fair | 78.5009 | 0 | 0 | 3.08E+11 | 0.982456 | 1.07006 |
| __wt_page_out | 77.2008 | 0 | 0 | 2.81E+11 | 1.03969 | 1.05181 |
| _raw_spin_lock_irq | 72.7008 | 0 | 0 | 1.88E+11 | 1.41267 | 1.01238 |
| system_call | 71.8008 | 0 | 0 | 1.29E+11 | 2.29692 | 1.14206 |
| pthread_cond_signal | 0 | 68.6008 | 0 | 2.71E+11 | 1.00532 | 1.10204 |
| ktime_get | 63.1007 | 0 | 0 | 3.01E+11 | 0.8 | 1.05864 |
| __wt_page_in_func | 62.8007 | 0 | 0 | 1.58E+11 | 1.64318 | 1.15127 |
| __hrtimer_start_range_ns | 62.5007 | 0 | 0 | 1.94E+11 | 1.31784 | 1.1344 |
| select_task_rq_fair | 61.9007 | 0 | 0 | 3.29E+11 | 0.71663 | 1.05816 |
| system_call_after_swapgs | 60.2007 | 0 | 0 | 2.05E+10 | 10.2982 | 0.975083 |
| poll_idle | 59.3006 | 0 | 0 | 2.52E+09 | 93.1429 | 1.09949 |
| nr_iowait_cpu | 59.2006 | 0 | 0 | 1.70E+11 | 1.45243 | 1.16047 |
| __find_get_page | 57.4006 | 0 | 0 | 4.64E+10 | 5.3876 | 1.2108 |
| __unqueue_futex | 53.5006 | 0 | 0 | 1.36E+11 | 1.59524 | 1.1271 |

| mongodb 100M | | | | | | |
|---|---|---|---|---|---|---|
| Process / Module / Function / Thread / | Effective Ti | Spin Time | Overhead T | Instruction | CPI Rate | CPU Freque |
| mongod | 10132 | 563.506 | 0 | 7.75E+13 | 0.568469 | 1.14485 |
| | 3846.74 | 133.101 | 0 | 9.28E+12 | 1.74336 | 1.12925 |
| gnome-shell | 824.109 | 1.80002 | 0 | 4.33E+12 | 0.718766 | 1.04625 |
| amplxe-gui | 135.401 | 4.80005 | 0 | 3.80E+11 | 1.59943 | 1.20471 |
| firefox | 44.4005 | 4.50005 | 0 | 1.50E+11 | 1.21875 | 1.03681 |
| X | 37.5004 | 0 | 0 | 1.62E+11 | 0.904656 | 1.088 |
| nmon | 21.2002 | 0 | 0 | 1.27E+11 | 0.65625 | 1.08962 |
| kswapd0 | 19.7002 | 0 | 0 | 8.96E+10 | 0.895582 | 1.13198 |
| MongoDB Compass | 12.9001 | 0 | 0 | 6.91E+10 | 0.765625 | 1.13953 |
| gnome-terminal- | 7.20008 | 0.200002 | 0 | 4.18E+10 | 0.741379 | 1.16216 |
| MongoDB Compass | 6.00007 | 0.200002 | 0 | 1.4E+10 | 1.53846 | 0.967742 |
| amplxe-runss | 1.80002 | 0 | 0 | 6.84E+09 | 1.10526 | 1.16667 |
| nautilus | 1.20001 | 0 | 0 | 7.2E+08 | 7 | 1.16667 |
| avahi-daemon | 1.00001 | 0 | 0 | 2.88E+09 | 1.625 | 1.3 |
| gedit | 0.700008 | 0.100001 | 0 | 2.52E+09 | 0.714286 | 0.625 |
| rhsmcertd-worke | 0.800009 | 0 | 0 | 3.96E+09 | 0.909091 | 1.25 |
| kworker/4:1H | 0.600007 | 0 | 0 | 1.44E+09 | 1.25 | 0.833333 |
| tuned | 0.500005 | 0 | 0 | 0 | | 0.8 |
| udisksd | 0.400004 | 0 | 0 | 3.6E+09 | 0.5 | 1.25 |
| rcuos/1 | 0.400004 | 0 | 0 | 0 | 0 | 0 |
| rcu_sched | 0.400004 | 0 | 0 | 1.44E+09 | 1.5 | 1.5 |
| xfsaild/sda5 | 0.400004 | 0 | 0 | 2.52E+09 | 0.714286 | 1.25 |
| rcuos/2 | 0.400004 | 0 | 0 | 3.6E+08 | 4 | 1 |
| irqbalance | 0.300003 | 0 | 0 | 3.6E+08 | 2 | 0.666667 |
| rngd | 0.300003 | 0 | 0 | 1.08E+09 | 0 | 0 |
| kworker/0:2 | 0.300003 | 0 | 0 | 0 | | 1 |
| gnome-settings- | 0.300003 | 0 | 0 | 3.6E+08 | 1 | 0.333333 |
| xfsaild/sda4 | 0.300003 | 0 | 0 | 1.08E+09 | 2 | 2 |
| goa-daemon | 0.200002 | 0 | 0 | 0 | | 1 |
| ksoftirqd/4 | 0 | 0.200002 | 0 | 3.6E+08 | 0 | 0 |
| MongoDB Compass | 0.200002 | 0 | 0 | 0 | | 1.5 |
| rcuos/0 | 0.200002 | 0 | 0 | 3.6E+08 | 1 | 0.5 |
| rcuos/6 | 0.200002 | 0 | 0 | 0 | 0 | 0 |
| kworker/5:1 | 0.200002 | 0 | 0 | 3.6E+08 | 0 | 0 |
| pulseaudio | 0.200002 | 0 | 0 | 3.6E+08 | 1 | 0.5 |
| ksmtuned | 0.200002 | 0 | 0 | 0 | | 0.5 |
| dbus-daemon | 0.200002 | 0 | 0 | 0 | 0 | 0 |
| ksmtuned | 0.100001 | 0 | 0 | 0 | 0 | 0 |
| rsyslogd | 0.100001 | 0 | 0 | 0 | 0 | 0 |
| awk | 0.100001 | 0 | 0 | 0 | 0 | 0 |
| pgrep | 0.100001 | 0 | 0 | 0 | 0 | 0 |
| pgrep | 0.100001 | 0 | 0 | 0 | 0 | 0 |
| rcuos/3 | 0.100001 | 0 | 0 | 0 | | 1 |
| crond | 0.100001 | 0 | 0 | 0 | 0 | 0 |
| pgrep | 0.100001 | 0 | 0 | 0 | 0 | 0 |

| couchbase 1M pi | | | | | | |
|---|---|---|---|---|---|---|
| Function / Call Stack | Effective Ti | Spin Time | Overhead T | Instruction | CPI Rate | CPU Freque |
| func@0x42fa6e | 283.199 | 0 | 0 | 2.25E+12 | 0.509998 | 1.12571 |
| func@0x49da00 | 144.6 | 0 | 0 | 4.9E+10 | 11.8971 | 1.11895 |
| func@0xac8945 | 133.5 | 0 | 0 | 1.24E+12 | 0.432221 | 1.11536 |
| func@0x410f10 | 89.2997 | 0 | 0 | 7.23E+11 | 0.506222 | 1.13886 |
| func@0x4131e0 | 78.3998 | 0 | 0 | 4.88E+11 | 0.605904 | 1.04719 |
| func@0xac2f46 | 74.2998 | 0 | 0 | 7.10E+11 | 0.403953 | 1.07268 |
| func@0x41151f | 73.7998 | 0 | 0 | 3.54E+11 | 0.840122 | 1.11789 |
| func@0x4317db | 70.7998 | 0 | 0 | 4.90E+11 | 0.567548 | 1.09181 |
| [Outside any known module] | 69.6998 | 0 | 0 | 3.88E+11 | 0.707136 | 1.09469 |
| func@0xac4656 | 66.5998 | 0 | 0 | 4.94E+11 | 0.520408 | 1.07207 |
| func@0x42c3e4 | 62.1998 | 0 | 0 | 1.04E+10 | 25.7241 | 1.19936 |
| func@0xb34f96 | 49.7998 | 0 | 0 | 4.57E+11 | 0.422835 | 1.07831 |
| func@0x42f520 | 43.8999 | 0 | 0 | 3.12E+11 | 0.544931 | 1.07745 |
| func@0x42c2f0 | 41.5999 | 0 | 0 | 1.76E+10 | 8.46939 | 0.997596 |
| func@0x420bc0 | 41.2999 | 0 | 0 | 3.22E+11 | 0.486034 | 1.05327 |
| func@0x42c3ab | 40.5999 | 0 | 0 | 1.35E+11 | 1.168 | 1.07882 |
| func@0x40b85b | 38.4999 | 0 | 0 | 2.68E+11 | 0.561828 | 1.08571 |
| func@0x41ff36 | 38.1999 | 0 | 0 | 2.37E+11 | 0.681335 | 1.17539 |
| copy_user_enhanced_fast_string | 36.1999 | 0 | 0 | 7.2E+10 | 1.95 | 1.07735 |
| mwait_idle_with_hints | 35.9999 | 0 | 0 | 1.8E+09 | 78.2 | 1.08611 |
| func@0x410dab | 33.7999 | 0 | 0 | 2.06E+11 | 0.647469 | 1.09763 |
| _raw_spin_lock_irqsave | 33.7999 | 0 | 0 | 7.56E+10 | 1.9 | 1.18047 |
| func@0x4225d0 | 30.4999 | 0 | 0 | 2.41E+11 | 0.518685 | 1.1377 |
| func@0x420a86 | 29.4999 | 0 | 0 | 2.49E+11 | 0.470418 | 1.10508 |
| fget_light | 28.8999 | 0 | 0 | 5.18E+10 | 2.13889 | 1.06574 |
| func@0x42c370 | 27.0999 | 0 | 0 | 4.57E+10 | 2.70866 | 1.26937 |
| clock_gettime | 26.4999 | 0 | 0 | 7.63E+10 | 1.40566 | 1.12453 |
| func@0x417759 | 25.9999 | 0 | 0 | 7.38E+10 | 1.25366 | 0.988462 |
| func@0x4414b6 | 25.6999 | 0 | 0 | 1.61E+11 | 0.58296 | 1.01167 |
| func@0x40c46b | 25.0999 | 0 | 0 | 1.69E+11 | 0.57234 | 1.07171 |
| func@0x41de66 | 24.1999 | 0 | 0 | 7.85E+10 | 1.33486 | 1.20248 |
| func@0xb2af1b | 24.0999 | 0 | 0 | 2.29E+11 | 0.36478 | 0.962656 |
| __memcpy_ssse3_back | 23.8999 | 0 | 0 | 7.27E+10 | 1.16832 | 0.987448 |
| ipt_do_table | 22.0999 | 0 | 0 | 8.06E+10 | 0.919643 | 0.932127 |
| func@0x42f360 | 21.9999 | 0 | 0 | 6.08E+10 | 1.43787 | 1.10455 |
| func@0x431fe6 | 21.6999 | 0 | 0 | 1.73E+10 | 4.54167 | 1.00461 |
| _raw_spin_lock | 20.9999 | 0 | 0 | 6.08E+10 | 1.30178 | 1.04762 |
| func@0x41b28e | 20.5999 | 0 | 0 | 1.17E+11 | 0.723926 | 1.14563 |
| func@0x42c2c0 | 20.3999 | 0 | 0 | 9.11E+10 | 0.944664 | 1.17157 |
| func@0x441856 | 19.8999 | 0 | 0 | 1.04E+11 | 0.712803 | 1.03518 |
| func@0xb34ba6 | 19.6999 | 0 | 0 | 1.95E+11 | 0.408503 | 1.12183 |
| func@0x4173d6 | 19.2999 | 0 | 0 | 8.39E+10 | 0.83691 | 1.01036 |
| tcp_recvmsg | 18.0999 | 0 | 0 | 3.92E+10 | 1.74312 | 1.04972 |
| func@0x42010b | 17.9999 | 0 | 0 | 1.07E+11 | 0.703704 | 1.16111 |
| tcp_transmit_skb | 17.9999 | 0 | 0 | 6.66E+10 | 1.02162 | 1.05 |

| couchbase 1M pi | | | | | | |
|---|---|---|---|---|---|---|
| Process / Module / Function / Thread / | Effective Ti | Spin Time | Overhead T | Instruction | CPI Rate | CPU Freque |
| cbq-engine | 3190.19 | 4.89999 | 0 | 1.72E+13 | 0.740155 | 1.10475 |
| memcached | 656.598 | 40.5999 | 0 | 1.92E+12 | 1.43682 | 1.09925 |
| indexer | 319.599 | 1.4 | 0 | 2.04E+12 | 0.642908 | 1.1324 |
| beam.smp | 251.399 | 5.69998 | 0 | 7.17E+11 | 1.46362 | 1.13458 |
| | 120.3 | 3.69999 | 0 | 2.32E+11 | 2.16124 | 1.12419 |
| amplxe-gui | 115.5 | 3.09999 | 0 | 2.90E+11 | 1.66129 | 1.12901 |
| gnome-shell | 91.2997 | 0.199999 | 0 | 5.09E+11 | 0.708127 | 1.09508 |
| projector | 14.6 | 0.1 | 0 | 8.21E+10 | 0.688596 | 1.06803 |
| X | 5.29998 | 0 | 0 | 2.23E+10 | 0.983871 | 1.15094 |
| beam.smp | 4.59999 | 0.199999 | 0 | 1.01E+10 | 1.32143 | 0.770833 |
| firefox | 3.49999 | 0 | 0 | 1.26E+10 | 1.42857 | 1.42857 |
| nmon | 2.59999 | 0 | 0 | 1.26E+10 | 0.942857 | 1.26923 |
| godu | 1.5 | 0 | 0 | 6.84E+09 | 1.36842 | 1.73333 |
| gnome-terminal- | 1.2 | 0.1 | 0 | 3.96E+09 | 1.27273 | 1.07692 |
| rcu_sched | 1.2 | 0 | 0 | 2.52E+09 | 0.857143 | 0.5 |
| rcuos/2 | 1.1 | 0 | 0 | 1.8E+09 | 0.8 | 0.363636 |
| rcuos/5 | 1.1 | 0 | 0 | 2.16E+09 | 1 | 0.545455 |
| rcuos/6 | 0.899997 | 0 | 0 | 1.44E+09 | 1.75 | 0.777778 |
| rcuos/1 | 0.599998 | 0 | 0 | 1.44E+09 | 3 | 2 |
| rcuos/3 | 0.599998 | 0 | 0 | 7.2E+08 | 2.5 | 0.833333 |
| goport | 0.499998 | 0 | 0 | 1.08E+09 | 1.66667 | 1 |
| goport | 0.299999 | 0.199999 | 0 | 7.2E+08 | 1.5 | 0.6 |
| goport | 0.399999 | 0 | 0 | 3.6E+08 | 3 | 0.75 |
| rcuos/0 | 0.399999 | 0 | 0 | 1.8E+09 | 1.8 | 2.25 |
| goport | 0.399999 | 0 | 0 | 7.2E+08 | 4 | 2 |
| beam.smp | 0.399999 | 0 | 0 | 7.2E+08 | 3 | 1.5 |
| rcuos/7 | 0.299999 | 0 | 0 | 1.8E+09 | 1.2 | 2 |
| sigar_port | 0.299999 | 0 | 0 | 2.16E+09 | 0.833333 | 1.66667 |
| goxdcr | 0.299999 | 0 | 0 | 1.44E+09 | 2.75 | 3.66667 |
| ksoftirqd/1 | 0.199999 | 0 | 0 | 0 | 0 | 0 |
| rcuos/4 | 0.199999 | 0 | 0 | 2.88E+09 | 0.375 | 1.5 |
| cbq | 0.199999 | 0 | 0 | 2.16E+09 | 0.5 | 1.5 |
| irqbalance | 0.1 | 0 | 0 | 3.6E+08 | 0 | 0 |
| amplxe-gui | 0.1 | 0 | 0 | 0 | 0 | 0 |
| sh | 0.1 | 0 | 0 | 0 | 0 | 0 |
| kthreadd | 0.1 | 0 | 0 | 0 | 0 | 0 |
| pgrep | 0.1 | 0 | 0 | 0 | 0 | 0 |
| amplxe-runss | 0.1 | 0 | 0 | 1.8E+09 | 0.6 | 3 |
| tuned | 0.1 | 0 | 0 | 0 | 0 | 0 |
| moxi | 0.1 | 0 | 0 | 0 | | 1 |
| xfsaild/sda4 | 0.1 | 0 | 0 | 0 | 0 | 0 |
| beam.smp | 0.1 | 0 | 0 | 1.08E+09 | 0.666667 | 2 |
| goa-daemon | 0.1 | 0 | 0 | 0 | 0 | 0 |
| kworker/5:2 | 0.1 | 0 | 0 | 0 | 0 | 0 |
| avahi-daemon | 0.1 | 0 | 0 | 7.2E+08 | 1 | 2 |

| couchbase sq11mnum | | | | | | |
|---|---|---|---|---|---|---|
| Function / Call Stack | Effective Ti | Spin Time | Overhead T | Instruction | CPI Rate | CPU Freque |
| func@0x42fa6e | 111.5 | 0 | 0 | 5.45E+11 | 0.79604 | 1.08161 |
| func@0x49da00 | 35.4999 | 0 | 0 | 1.26E+10 | 11.0286 | 1.08732 |
| [Outside any known module] | 28.3999 | 0 | 0 | 1.50E+11 | 0.72488 | 1.0669 |
| func@0xac8994 | 26.4999 | 0 | 0 | 2.58E+11 | 0.452646 | 1.22642 |
| func@0x4131e0 | 24.1999 | 0 | 0 | 1.30E+11 | 0.720222 | 1.07438 |
| func@0x4317db | 21.8999 | 0 | 0 | 1.19E+11 | 0.706949 | 1.06849 |
| func@0x41151f | 21.2999 | 0 | 0 | 9.29E+10 | 0.895349 | 1.08451 |
| func@0xac2f46 | 18.3999 | 0 | 0 | 1.67E+11 | 0.36933 | 0.929348 |
| func@0x42f520 | 15.1 | 0 | 0 | 8.71E+10 | 0.72314 | 1.15894 |
| func@0xac4656 | 14.6 | 0 | 0 | 1.24E+11 | 0.533333 | 1.26027 |
| func@0x42c3b0 | 13.3 | 0 | 0 | 4.79E+10 | 1.22556 | 1.22556 |
| func@0x4111fe | 12.8 | 0 | 0 | 1.29E+11 | 0.42577 | 1.1875 |
| __memcpy_ssse3_back | 12.1 | 0 | 0 | 2.3E+10 | 1.875 | 0.991736 |
| func@0x410dab | 12.1 | 0 | 0 | 8.06E+10 | 0.696429 | 1.28926 |
| func@0x41b4a6 | 10.4 | 0 | 0 | 5.18E+10 | 0.6875 | 0.951923 |
| func@0x40b85b | 10.2 | 0 | 0 | 7.49E+10 | 0.576923 | 1.17647 |
| func@0x42c3f0 | 9.99997 | 0 | 0 | 7.2E+08 | 61 | 1.22 |
| func@0x420bc0 | 9.89997 | 0 | 0 | 7.78E+10 | 0.509259 | 1.11111 |
| copy_user_enhanced_fast_string | 9.89997 | 0 | 0 | 1.48E+10 | 2.80488 | 1.16162 |
| func@0xb34f96 | 9.69997 | 0 | 0 | 8.46E+10 | 0.442553 | 1.07216 |
| func@0x417759 | 9.39997 | 0 | 0 | 2.27E+10 | 1.47619 | 0.989362 |
| clock_gettime | 8.99997 | 0 | 0 | 2.52E+10 | 1.41429 | 1.1 |
| func@0x41ff36 | 8.89997 | 0 | 0 | 6.66E+10 | 0.589189 | 1.22472 |
| func@0x420a86 | 8.79997 | 0 | 0 | 5.51E+10 | 0.464052 | 0.806818 |
| func@0x4225d0 | 8.79997 | 0 | 0 | 7.34E+10 | 0.480392 | 1.11364 |
| func@0x42c6d0 | 8.39997 | 0 | 0 | 2.59E+10 | 1.59722 | 1.36905 |
| _raw_spin_lock_irqsave | 8.09998 | 0 | 0 | 1.84E+10 | 1.72549 | 1.08642 |
| func@0x42f360 | 7.79998 | 0 | 0 | 1.44E+10 | 2.2 | 1.12821 |
| mwait_idle_with_hints | 7.69998 | 0 | 0 | 1.8E+09 | 17.8 | 1.15584 |
| func@0x415190 | 7.19998 | 0 | 0 | 3.56E+10 | 0.79798 | 1.09722 |
| func@0x40e036 | 7.19998 | 0 | 0 | 1.22E+10 | 2.11765 | 1 |
| func@0x40c46b | 6.89998 | 0 | 0 | 4.07E+10 | 0.637168 | 1.04348 |
| fget_light | 6.79998 | 0 | 0 | 9E+09 | 2.64 | 0.970588 |
| func@0x41de66 | 6.29998 | 0 | 0 | 2.05E+10 | 1.05263 | 0.952381 |
| func@0x4173d6 | 6.19998 | 0 | 0 | 2.34E+10 | 1.04615 | 1.09677 |
| func@0x4414b6 | 6.19998 | 0 | 0 | 4.86E+10 | 0.511111 | 1.1129 |
| func@0x411097 | 5.79998 | 0 | 0 | 4.1E+10 | 0.570175 | 1.12069 |
| func@0x42c370 | 5.79998 | 0 | 0 | 9.72E+09 | 2.51852 | 1.17241 |
| _raw_spin_lock | 5.39998 | 0 | 0 | 1.3E+10 | 1.38889 | 0.925926 |
| ipt_do_table | 5.39998 | 0 | 0 | 2.41E+10 | 0.701493 | 0.87037 |
| process_main | 5.19998 | 0 | 0 | 2.88E+10 | 0.625 | 0.961538 |
| func@0x40eb56 | 5.19998 | 0 | 0 | 2.02E+10 | 0.946429 | 1.01923 |
| func@0x431fe6 | 4.99998 | 0 | 0 | 6.12E+09 | 2.35294 | 0.8 |
| func@0xac4276 | 4.99998 | 0 | 0 | 3.13E+10 | 0.45977 | 0.8 |
| func@0x4117f6 | 4.79999 | 0 | 0 | 3.13E+10 | 0.528736 | 0.958333 |

| couchbase sq11mnum | | | | | | |
|---|---|---|---|---|---|---|
| Process / Function / Thread / Call Stack | Effective Ti | Spin Time | Overhead T | Instruction | CPI Rate | CPU Freque |
| cbq-engine | 907.197 | 0.799998 | 0 | 4.49E+12 | 0.806677 | 1.10705 |
| memcached | 158.6 | 8.19997 | 0 | 4.54E+11 | 1.45678 | 1.10132 |
| indexer | 123.7 | 3.19999 | 0 | 6.40E+11 | 0.796963 | 1.11663 |
| beam.smp | 60.7998 | 1.2 | 0 | 1.67E+11 | 1.48276 | 1.10968 |
| gnome-shell | 33.1999 | 0.1 | 0 | 1.93E+11 | 0.734579 | 1.18018 |
| amplxe-gui | 31.5999 | 1.2 | 0 | 7.67E+10 | 1.84507 | 1.19817 |
| | 28.4999 | 0.799998 | 0 | 4.82E+10 | 2.3806 | 1.08874 |
| projector | 5.49998 | 0 | 0 | 1.94E+10 | 0.611111 | 0.6 |
| X | 2.49999 | 0 | 0 | 6.84E+09 | 1.26316 | 0.96 |
| nmon | 1.1 | 0 | 0 | 4.32E+09 | 0.833333 | 0.909091 |
| firefox | 0.799998 | 0 | 0 | 2.16E+09 | 2 | 1.5 |
| ibus-daemon | 0.499998 | 0.1 | 0 | 3.6E+08 | 0 | 0 |
| beam.smp | 0.499998 | 0 | 0 | 4.68E+09 | 0.230769 | 0.6 |
| gnome-terminal- | 0.499998 | 0 | 0 | 1.8E+09 | 1.2 | 1.2 |
| rcu_sched | 0.399999 | 0 | 0 | 3.6E+08 | 4 | 1 |
| goxdcr | 0.299999 | 0 | 0 | 3.6E+08 | 2 | 0.666667 |
| goport | 0.299999 | 0 | 0 | 0 | | 1 |
| rcuos/5 | 0.299999 | 0 | 0 | 0 | | 0.333333 |
| rcuos/7 | 0.199999 | 0 | 0 | 7.2E+08 | 1 | 1 |
| cbq | 0.199999 | 0 | 0 | 1.8E+09 | 0.2 | 0.5 |
| godu | 0.199999 | 0 | 0 | 1.44E+09 | 0.75 | 1.5 |
| beam.smp | 0.1 | 0 | 0 | 1.08E+09 | 0 | 0 |
| rcuos/0 | 0.1 | 0 | 0 | 7.2E+08 | 1 | 2 |
| df | 0.1 | 0 | 0 | 0 | 0 | 0 |
| kworker/2:0 | 0.1 | 0 | 0 | 0 | | 1 |
| moxi | 0.1 | 0 | 0 | 0 | 0 | 0 |
| rcuos/1 | 0.1 | 0 | 0 | 0 | | 3 |
| rcuos/3 | 0.1 | 0 | 0 | 7.2E+08 | 0 | 0 |
| rcuos/6 | 0.1 | 0 | 0 | 3.6E+08 | 0 | 0 |
| rcuos/4 | 0.1 | 0 | 0 | 3.6E+08 | 2 | 2 |
| goport | 0.1 | 0 | 0 | 0 | 0 | 0 |
| goport | 0.1 | 0 | 0 | 0 | | 1 |
| goport | 0 | 0 | 0 | 3.6E+08 | 1 | |
| sigar_port | 0 | 0 | 0 | 1.08E+09 | 0 | 0 |
| rcuos/2 | 0 | 0 | 0 | 3.6E+08 | 0 | 0 |
| migration/2 | 0 | 0 | 0 | 3.6E+08 | 0 | 0 |
| amplxe-gui | 0 | 0 | 0 | 3.6E+08 | 1 | |

| couchbase sq1100knum | | | | | | |
|---|---|---|---|---|---|---|
| Function / Call Stack | Effective Ti | Spin Time | Overhead T | Instruction | CPI Rate | CPU Freque |
| func@0x42fa6e | 9.69997 | 0 | 0 | 4.68E+10 | 0.769231 | 1.03093 |
| [Outside any known module] | 8.79997 | 0 | 0 | 4.61E+10 | 0.757812 | 1.10227 |
| func@0x49da00 | 7.79998 | 0 | 0 | 1.44E+09 | 24.25 | 1.24359 |
| func@0x42c370 | 4.69999 | 0 | 0 | 2.16E+09 | 5.83333 | 0.744681 |
| func@0x431fe6 | 3.39999 | 0 | 0 | 1.8E+09 | 4.8 | 0.705882 |
| func@0x4131e0 | 3.19999 | 0 | 0 | 1.76E+10 | 0.530612 | 0.8125 |
| func@0xac2f46 | 2.79999 | 0 | 0 | 1.66E+10 | 0.391304 | 0.642857 |
| func@0x41151f | 2.19999 | 0 | 0 | 1.37E+10 | 0.631579 | 1.09091 |
| func@0x410dab | 1.99999 | 0 | 0 | 1.08E+10 | 0.666667 | 1 |
| __memcpy_ssse3_back | 1.99999 | 0 | 0 | 3.24E+09 | 1.77778 | 0.8 |
| func@0xac892f | 1.89999 | 0 | 0 | 3.17E+10 | 0.522727 | 2.42105 |
| func@0x4225d0 | 1.79999 | 0 | 0 | 9.36E+09 | 0.615385 | 0.888889 |
| func@0x41117a | 1.6 | 0 | 0 | 1.62E+10 | 0.466667 | 1.3125 |
| func@0xac4656 | 1.6 | 0 | 0 | 1.3E+10 | 0.555556 | 1.25 |
| process_main | 1.3 | 0 | 0 | 7.92E+09 | 0.681818 | 1.15385 |
| mwait_idle_with_hints | 1.2 | 0 | 0 | 0 | | 1.16667 |
| func@0x42c3ab | 1.2 | 0 | 0 | 7.2E+09 | 1.05 | 1.75 |
| func@0x42c6cb | 1.1 | 0 | 0 | 1.8E+09 | 2.2 | 1 |
| func@0xb34f96 | 1.1 | 0 | 0 | 1.22E+10 | 0.352941 | 1.09091 |
| func@0x417760 | 0.999997 | 0 | 0 | 1.8E+09 | 1.8 | 0.9 |
| func@0x40b85b | 0.999997 | 0 | 0 | 6.48E+09 | 0.722222 | 1.3 |
| clock_gettime | 0.999997 | 0 | 0 | 1.8E+09 | 2.4 | 1.2 |
| fget_light | 0.999997 | 0 | 0 | 1.08E+09 | 3.33333 | 1 |
| func@0x41ff36 | 0.999997 | 0 | 0 | 4.68E+09 | 1.07692 | 1.4 |
| func@0x41b4b7 | 0.999997 | 0 | 0 | 3.6E+09 | 1.1 | 1.1 |
| func@0x41f0e6 | 0.899997 | 0 | 0 | 3.6E+09 | 0.8 | 0.888889 |
| func@0x441856 | 0.899997 | 0 | 0 | 2.88E+09 | 0.625 | 0.555556 |
| func@0x430606 | 0.799998 | 0 | 0 | 1.08E+09 | 3 | 1.125 |
| func@0x411097 | 0.799998 | 0 | 0 | 6.12E+09 | 0.235294 | 0.5 |
| func@0x41de66 | 0.799998 | 0 | 0 | 2.88E+09 | 0.25 | 0.25 |
| func@0x41e85b | 0.799998 | 0 | 0 | 4.32E+09 | 0.25 | 0.375 |
| func@0x42010b | 0.799998 | 0 | 0 | 3.24E+09 | 0.888889 | 1 |
| func@0x4173d6 | 0.799998 | 0 | 0 | 3.24E+09 | 0.444444 | 0.5 |
| copy_user_enhanced_fast_string | 0.699998 | 0 | 0 | 2.52E+09 | 2 | 2 |
| func@0x42c3f0 | 0.699998 | 0 | 0 | 0 | | 1 |
| tcp_sendmsg | 0.699998 | 0 | 0 | 7.2E+08 | 1.5 | 0.428571 |
| func@0x431abc | 0.699998 | 0 | 0 | 5.4E+09 | 0.6 | 1.28571 |
| func@0xac4276 | 0.699998 | 0 | 0 | 2.16E+09 | 1 | 0.857143 |
| func@0x690e1e | 0.699998 | 0 | 0 | 1.8E+09 | 0.2 | 0.142857 |
| _raw_spin_lock_irqsave | 0.599998 | 0 | 0 | 1.44E+09 | 3 | 2 |
| func@0x41dfe6 | 0.599998 | 0 | 0 | 2.16E+09 | 0.5 | 0.5 |
| func@0x42f360 | 0.599998 | 0 | 0 | 2.52E+09 | 0.714286 | 0.833333 |
| func@0xb34ba6 | 0.599998 | 0 | 0 | 5.4E+09 | 0.4 | 1 |
| func@0xac82c6 | 0.599998 | 0 | 0 | 1.8E+09 | 0.6 | 0.5 |
| func@0x40e036 | 0.599998 | 0 | 0 | 1.08E+09 | 1.33333 | 0.666667 |

| couchbase sq1100knum | | | | | | |
|---|---|---|---|---|---|---|
| Process / Module / Function / Thread / | Effective Ti | Spin Time | Overhead T | Instruction | CPI Rate | CPU Freque |
| cbq-engine | 102.3 | 0 | 0 | 5.08E+11 | 0.802128 | 1.10557 |
| indexer | 17.9999 | 0 | 0 | 8.46E+10 | 0.774468 | 1.01111 |
| memcached | 16.7999 | 0.499998 | 0 | 5.11E+10 | 1.43662 | 1.17919 |
| beam.smp | 13.7 | 0.399999 | 0 | 3.38E+10 | 1.74468 | 1.16312 |
| gnome-shell | 11 | 0 | 0 | 5.72E+10 | 0.735849 | 1.06364 |
| | 3.59999 | 0.1 | 0 | 9.72E+09 | 1.40741 | 1.02703 |
| amplxe-gui | 3.59999 | 0.1 | 0 | 7.56E+09 | 1.90476 | 1.08108 |
| X | 0.699998 | 0 | 0 | 5.04E+09 | 0.285714 | 0.571429 |
| projector | 0.699998 | 0 | 0 | 3.24E+09 | 1 | 1.28571 |
| cbq | 0.299999 | 0 | 0 | 2.52E+09 | 0.571429 | 1.33333 |
| godu | 0.1 | 0 | 0 | 7.2E+08 | 0 | 0 |
| gedit | 0.1 | 0 | 0 | 0 | | 1 |
| firefox | 0.1 | 0 | 0 | 7.2E+08 | 0.5 | 1 |
| goxdcr | 0.1 | 0 | 0 | 0 | 0 | 0 |
| beam.smp | 0.1 | 0 | 0 | 3.6E+08 | 0 | 0 |
| nmon | 0.1 | 0 | 0 | 7.2E+08 | 0.5 | 1 |
| ibus-daemon | 0.1 | 0 | 0 | 0 | | 1 |
| rcuos/7 | 0.1 | 0 | 0 | 0 | 0 | 0 |
| rcuos/2 | 0 | 0 | 0 | 3.6E+08 | 0 | 0 |
| beam.smp | 0 | 0 | 0 | 3.6E+08 | 3 | |
| gnome-terminal- | 0 | 0 | 0 | 7.2E+08 | 0.5 | |
| amplxe-gui | 0 | 0 | 0 | 3.6E+08 | 0 | 0 |
| | 171.4998 | | | | | |

| | vertica | cb | cb | mongo | vertica | cb | cb | mongo | vertica | cb | cb | mongo |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | sq1khb | sq11k | sq11knum | sq11k | sq10khb | sq110k | sq110knum | sq110k | sq100khb | sq1100k | q1100knum | sq1100k |
| q101 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q102 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| q103 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 9 | 9 | 9 | 9 |
| q104 | 0 | 0 | 0 | 0 | 7 | 7 | 7 | 7 | 95 | 95 | 95 | 95 |
| q105 | 3 | 3 | 3 | 3 | 81 | 81 | 81 | 81 | 809 | 809 | 809 | 809 |
| q106 | 35 | 35 | 35 | 35 | 360 | 360 | 360 | 360 | 3603 | 3603 | 3603 | 3603 |
| q107 | 95 | 95 | 95 | 95 | 833 | 833 | 833 | 833 | 8873 | 8873 | 8873 | 8873 |
| q108 | 165 | 165 | 165 | 165 | 1749 | 1749 | 1749 | 1749 | 17723 | 17723 | 17723 | 17723 |
| q109 | 239 | 239 | 239 | 239 | 2255 | 2255 | 2255 | 2255 | 22053 | 22053 | 22053 | 22053 |
| q110 | 291 | 291 | 291 | 291 | 2843 | 2843 | 2843 | 2843 | 28861 | 28861 | 28861 | 28861 |
| q201 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| q202 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q203 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q204 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q205 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q206 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q207 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q208 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q209 | 181 | 181 | 181 | 181 | 1722 | 1722 | 1722 | 1722 | 17514 | 17514 | 17514 | 17514 |
| q210 | 290 | 290 | 290 | 290 | 2842 | 2842 | 2842 | 2842 | 28860 | 28860 | 28860 | 28860 |
| q211 | 291 | 291 | 291 | 291 | 2843 | 2843 | 2843 | 2843 | 28861 | 28861 | 28861 | 28861 |
| q212 | 291 | 291 | 291 | 291 | 2843 | 2843 | 2843 | 2843 | 28861 | 28861 | 28861 | 28861 |
| q213 | 291 | 291 | 291 | 291 | 2843 | 2843 | 2843 | 2843 | 28861 | 28861 | 28861 | 28861 |
| q214 | 291 | 291 | 291 | 291 | 2843 | 2843 | 2843 | 2843 | 28861 | 28861 | 28861 | 28861 |
| q215 | 291 | 291 | 291 | 291 | 2843 | 2843 | 2843 | 2843 | 28861 | 28861 | 28861 | 28861 |
| q216 | 291 | 291 | 291 | 291 | 2843 | 2843 | 2843 | 2843 | 28861 | 28861 | 28861 | 28861 |
| q217 | 291 | 291 | 291 | 291 | 2843 | 2843 | 2843 | 2843 | 28861 | 28861 | 28861 | 28861 |
| q218 | 110 | 110 | 110 | 110 | 1121 | 1121 | 1121 | 1121 | 11347 | 11347 | 11347 | 11347 |
| q301 | empty | empty | empty | empty | empty | empty | empty | empty | 0 | empty | 0 | 0 |
| q302 | empty | empty | empty | empty | empty | empty | empty | empty | 0 | empty | 0 | 0 |
| q303 | 6 | 6 | 6 | 6 | 42 | 42 | 42 | 42 | 303 | 303 | 303 | 303 |
| q304 | 178 | 178 | 178 | 178 | 3063 | 3063 | 3063 | 3063 | 28963 | 28963 | 28963 | 28963 |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| q305 | 5228 | 5228 | 5228 | 5228 | 52344 | 52344 | 52344 | 52344 | 49588 | 49588 | 49588 | 49588 |
| q306 | 31032 | 31032 | 31032 | 31032 | 337227 | 337227 | 337227 | 337227 | 3210783 | 3210783 | 3210783 | 3210783 |
| q307 | 139139 | 139139 | 139139 | 139139 | 1259006 | 1259006 | 1259006 | 1259006 | 12538015 | 12538015 | 12538015 | 12538015 |
| q308 | 202895 | 202895 | 202895 | 202895 | 1909482 | 1909482 | 1909482 | 1909482 | 19494037 | 19494037 | 19494037 | 19494037 |
| q309 | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q310 | empty | empty | empty | empty | empty | empty | empty | empty | 0 | 0 | 0 | 0 |
| q311 | 6 | 3 | 6 | 6 | 42 | 12 | 42 | 42 | 156 | 126 | 156 | 156 |
| q312 | 178 | 76 | 178 | 178 | 3063 | 1225 | 3063 | 3063 | 14246 | 12408 | 14246 | 14246 |
| q313 | 5228 | 1850 | 5228 | 5228 | 52344 | 22576 | 52344 | 52344 | 244821 | 215053 | 244821 | 244821 |
| q314 | 31032 | 10612 | 31032 | 31032 | 337227 | 138495 | 337227 | 337227 | 1624866 | 1426134 | 1624866 | 1624866 |
| q315 | 139139 | 64224 | 139139 | 139139 | 1259006 | 561977 | 1259006 | 1259006 | 6215314 | 5518285 | 6215314 | 6213935 |
| q316 | 202895 | 3 | 202895 | 201939 | 1909482 | 12 | 1909482 | 1908526 | 9704528 | 126 | 9704528 | 9703572 |
| q401 | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q402 | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q403 | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q404 | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q405 | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q406 | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q407 | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q408 | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q409 | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q410 | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q411 | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q412 | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q413 | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q414 | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q501 | | | | | | | | | | | | |
| q502 | | | | | | | | | | | | |
| q503 | | | | | | | | | | | | |

| | vertica | cb | cb | mongo | vertica | cb | mongo | vertica | mongo |
|---|---|---|---|---|---|---|---|---|---|
| | sq1mhb | sq11m | sq11mnum | sq11m | sq10mhb | sq110m | sq110m | sq100mhb | sq1100m |
| q101 | 3 | 3 | 3 | 3 | 30 | 30 | 30 | 363 | 363 |
| q102 | 11 | 11 | 11 | 11 | 88 | 88 | 88 | 907 | 907 |
| q103 | 90 | 90 | 90 | 90 | 877 | 877 | 877 | 8669 | 8669 |
| q104 | 902 | 902 | 902 | 902 | 8789 | 8789 | 8789 | 87404 | 87404 |
| q105 | 8949 | 8949 | 8949 | 8949 | 88098 | 88098 | 88098 | 878297 | 878297 |
| q106 | 35727 | 35727 | 35727 | 35727 | 356511 | 356511 | 356511 | 355607 | 355607 |
| q107 | 88716 | 88716 | 88716 | 88716 | 886882 | 886882 | 886882 | 8867793 | 8867793 |
| q108 | 178162 | 178162 | 178162 | 178162 | 1778112 | 1778112 | 1778112 | 17777031 | 17777031 |
| q109 | 221969 | 221969 | 221969 | 221969 | 2218878 | 2218878 | 2218878 | 22177138 | 22177138 |
| q110 | 288909 | 288909 | 288909 | 288909 | 2891278 | 2891278 | 2891278 | 28907042 | 28907042 |
| q201 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| q202 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q203 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q204 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q205 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q206 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q207 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q208 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q209 | 175883 | 175883 | 175883 | 175883 | 176170 | 176170 | 1761670 | 17605441 | 17605441 |
| q210 | 288908 | 288908 | 288908 | 288908 | 2891277 | 2891277 | 2891277 | 28907041 | 28907041 |
| q211 | 288909 | 288909 | 288909 | 288909 | 2891278 | 2891278 | 2891278 | 28907402 | 28907402 |
| q212 | 288909 | 288909 | 288909 | 288909 | 2891278 | 2891278 | 2891278 | 28907402 | 28907402 |
| q213 | 288909 | 288909 | 288909 | 288909 | 2891278 | 2891278 | 2891278 | 28907402 | 28907402 |
| q214 | 288909 | 288909 | 288909 | 288909 | 2891278 | 2891278 | 2891278 | 28907402 | 28907402 |
| q215 | 288909 | 288909 | 288909 | 288909 | 2891278 | 2891278 | 2891278 | 28907402 | 28907402 |
| q216 | 288909 | 288909 | 288909 | 288909 | 2891278 | 2891278 | 2891278 | 28907402 | 28907402 |
| q217 | 288909 | 288909 | 288909 | 288909 | 2891278 | 2891278 | 2891278 | 28907402 | 28907402 |
| q218 | 113026 | 113026 | 113026 | 113026 | 1129608 | 1129608 | 1129608 | 11301601 | 11301601 |
| q301 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q302 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q303 | 2730 | 2730 | 2730 | 2730 | 26373 | 26373 | 26373 | 262635 | 262635 |
| q304 | 301781 | 301781 | 301781 | 301781 | 3025338 | 3025338 | 3025338 | 30245233 | 30245233 |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| q305 | 5002310 | 5002310 | 5002310 | 5002310 | 50041893 | 50041893 | 50041893 | 501950867 | 501950867 |
| q306 | 31298629 | 31298629 | 31298629 | 31298629 | 313209671 | 313209671 | 313209671 | 3129619227 | 3129619227 |
| q307 | 125693021 | 125693021 | 125693021 | 125693021 | 1256980895 | 1256980895 | 1256980895 | 12574604989 | 12574604989 |
| q308 | 195670828 | 195670828 | 195670828 | 195670828 | 1958326064 | 1958326064 | 1958326064 | 19577428992 | 19577428992 |
| q309 | emptty | 0 | emptty | empty | empty | 0 | empty | empty | empty |
| q310 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| q311 | 372 | 1164 | 372 | 372 | 372 | 11877 | 372 | 372 | 372 |
| q312 | 36789 | 131976 | 36789 | 36789 | 36789 | 1342162 | 36789 | 36789 | 36789 |
| q313 | 614585 | 2208653 | 614585 | 614585 | 614583 | 22183460 | 614585 | 614585 | 614585 |
| q314 | 3975418 | 13961046 | 3975418 | 3975418 | 3975418 | 139387281 | 3975418 | 3975418 | 3975418 |
| q315 | 15695319 | 55773960 | 15695319 | 15692667 | 15695319 | 558561508 | 15692667 | 15695319 | 15692667 |
| q316 | 24338972 | 1164 | 24338972 | 24336254 | 24338972 | 11877 | 24336254 | 24338972 | 24336254 |
| q401 | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q402 | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q403 | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q404 | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q405 | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q406 | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q407 | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q408 | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q409 | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q410 | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q411 | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q412 | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q413 | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q414 | empty | empty | empty | empty | empty | empty | empty | empty | empty |
| q501 | | | | | | | | | |
| q502 | | | | | | | | | |
| q503 | | | | | | | | | |