MySQL vs MongoDB

When to Use Which Technology

Peter Zaitsev
CEO
Percona University, Ghent
June 22nd, 2017
In This Presentation

Very brief discussion on merits of MySQL and MongoDB
# Why MySQL and MongoDB?

## Most Popular Open Source SQL and NoSQL Engines

<table>
<thead>
<tr>
<th>Rank</th>
<th>DBMS</th>
<th>Database Model</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Oracle +</td>
<td>Relational DBMS</td>
<td>1354.31</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-47.68</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-107.71</td>
</tr>
<tr>
<td>2</td>
<td>MySQL +</td>
<td>Relational DBMS</td>
<td>1340.03</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-24.59</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-31.80</td>
</tr>
<tr>
<td>3</td>
<td>Microsoft SQL Server +</td>
<td>Relational DBMS</td>
<td>1213.80</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+9.03</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+70.98</td>
</tr>
<tr>
<td>4</td>
<td>PostgreSQL +</td>
<td>Relational DBMS</td>
<td>365.91</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+4.14</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+58.30</td>
</tr>
<tr>
<td>5</td>
<td>MongoDB +</td>
<td>Document store</td>
<td>331.58</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+6.16</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+11.36</td>
</tr>
<tr>
<td>6</td>
<td>DB2 +</td>
<td>Relational DBMS</td>
<td>188.84</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+2.18</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+2.88</td>
</tr>
<tr>
<td>7</td>
<td>Microsoft Access</td>
<td>Relational DBMS</td>
<td>129.87</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+1.69</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-1.70</td>
</tr>
<tr>
<td>8</td>
<td>Cassandra +</td>
<td>Wide column store</td>
<td>123.11</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-3.07</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-11.39</td>
</tr>
<tr>
<td>9</td>
<td>Redis +</td>
<td>Key-value store</td>
<td>117.45</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+3.09</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+9.21</td>
</tr>
<tr>
<td>10</td>
<td>SQLite</td>
<td>Relational DBMS</td>
<td>116.07</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+2.27</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>+8.81</td>
</tr>
</tbody>
</table>
Why MySQL and MongoDB?

Two Technologies Percona Provides Solutions For
Full Disclosure

I know MySQL Much better than MongoDB... which will impact my bias
MySQL

Relational Database First and Foremost

Full SQL Support, Transactions, ACID

Designed for a Single Server first

Scale-Out as Afterthought
MongoDB

Designed for “Web Scale”

Scalability, Cloud, Multiple Machines

Replication and Sharding part of initial design

Only features which can scale
Q1: What do you know and love?

Both MySQL and MongoDB are very capable. Your experience and preference matter.
Q2: Which data model fits better?

**Relational**
- MySQL Obvious Choice

**Document Based**
- MongoDB Obvious choice
- MySQL has Document Store starting 5.7
Q3: How Data is Used

Data belongs to single application

- JSON model more expressive for application data structures
- Schema designed for specific access paths

Data shared by multiple applications

- Relational structure easier to share
- Can be more flexible in how data is accessed
Q4: Transactions

Need full Transactions

• MySQL can be better choice
• One of the main benefits of MySQL Document Store

Do not need Transactions

• MongoDB can be great choice
• Can do Atomic Document Updates
Q5: JOINs

Advanced JOINs and other SQL features

- MySQL much more powerful
- $lookup and $graphLookup features in MongoDB aggregation framework

Mainly simple lookups with filters/sorting

- MongoDB and MySQL both do these very well
Q6: Scale

Single Server is Good Enough

• MySQL works great
• Well optimized for Many cores; large memory; fast storage

Need Massive Scale out

• Automated shading in MongoDB is much better
• Replication in MongoDB is easier to use
• Solutions like Vitess try to make it less painful for MySQL
Q7: Large Scale Aggregation

**MongoDB**
- has built in aggregation framework for parallel processing
- BI Connector and ToroDB for SQL access
- Replicate to Hadoop

**MySQL**
- Executes every query single threaded
- MariaDB ColumnStore (InfiniDB reborn)
- ClickHouse
- Replicate to Hadoop
MySQL and MongoDB compared

Courtesy of Alexander Rubin
From MySQL to MongoDB

**MySQL**

```sql
mysql> select * from zips limit 1

+----------------+----------------+-------------------+------------------+
| country_code   | postal_code    | place_name        | admin_name1      |
|----------------+----------------+-------------------+------------------|
| US             | 34050          | FPO               | AA               |
| postal_code    | place_name     |                  | admin_name1      |
| 34050          | FPO            |                  | AA               |
| place_name     | admin_name1    |                  |                  |
| FPO            | AA             |                  |                  |
| admin_name1    |                |                  |                  |
| AA             |                |                  |                  |
| admin_code1    | admin_name2    | admin_name3      |                  |
| AA             | Erie           |                  |                  |
| admin_code2    | admin_code3    |                  |                  |
| 029            |                |                  |                  |
| admin_code3    |                  |                  |                  |
|                | latitude       | longitude        | accuracy         |
|                | 41.03750000    | -111.67890000    |                  |
| accuracy       |                |                  |                  |
```

1 row in set (0.00 sec)

**MongoDB**

```
MongoDB shell version: 3.0.8
connecting to: zips
> db.zips.find().limit(1).pretty()
{
   "_id" : "01001",
   "city" : "AGAWAM",
   "loc" : [
      -72.622739,
      42.070206
   ],
   "pop" : 15338,
   "state" : "MA"
}
```
### SQL to MongoDB Mapping Chart

[https://docs.mongodb.org/manual/reference/sql-comparison/](https://docs.mongodb.org/manual/reference/sql-comparison/)

<table>
<thead>
<tr>
<th>MySQL</th>
<th>MongoDB</th>
</tr>
</thead>
</table>
| **CREATE TABLE** users (  
  id MEDIUMINT **NOT NULL**  
  AUTO_INCREMENT,  
  user_id Varchar(30),  
  age Number,  
  status char(1),  
  **PRIMARY KEY** (id)  
) | db.users.insert( {  
  user_id: "abc123",  
  age: 55,  
  status: "A"  
} )  
(no schema) |
### SQL to MongoDB Mapping Chart

**MySQL**

**SELECT** *

**FROM** users

**WHERE** status = "A"

**AND** age = 50

**MongoDB**

```
db.users.find(
    { status: "A",
      age: 50 }
)
```
<table>
<thead>
<tr>
<th>MySQL</th>
<th>MongoDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/my.cnf</td>
<td>/etc/mongod.conf</td>
</tr>
<tr>
<td></td>
<td># Where and how to store data.</td>
</tr>
<tr>
<td></td>
<td>storage:</td>
</tr>
<tr>
<td></td>
<td>dbPath: /datawt</td>
</tr>
<tr>
<td></td>
<td>journal:</td>
</tr>
<tr>
<td></td>
<td>enabled: true</td>
</tr>
<tr>
<td></td>
<td>engine: wiredTiger</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>/usr/bin/mongod -f /etc/mongod.conf</td>
</tr>
</tbody>
</table>
### Where are my databases/tables?

**MySQL**

<table>
<thead>
<tr>
<th>Databases</th>
<th></th>
</tr>
</thead>
</table>
| mysql> show databases; | +----------------------+
| | | Database | |
| | +----------------------+ |
| | | information_schema | |
| ... | |

mysql> use zips
Database changed

<table>
<thead>
<tr>
<th>Tables</th>
<th></th>
</tr>
</thead>
</table>
| mysql> show tables; | +------------------+
| | | Tables_in_zips | |
| | +------------------+ |
| | | zips | |
| +------------------+ |

**MongoDB**

<table>
<thead>
<tr>
<th>Databases</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt; show dbs;</td>
<td></td>
</tr>
<tr>
<td>admin</td>
<td>0.000GB</td>
</tr>
<tr>
<td>local</td>
<td>0.000GB</td>
</tr>
<tr>
<td>osm</td>
<td>13.528GB</td>
</tr>
<tr>
<td>test</td>
<td>0.000GB</td>
</tr>
<tr>
<td>zips</td>
<td>0.002GB</td>
</tr>
</tbody>
</table>

> use zips
switched to db zips

<table>
<thead>
<tr>
<th>Collections</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt; show collections</td>
<td></td>
</tr>
<tr>
<td>zips</td>
<td></td>
</tr>
<tr>
<td>&gt; show tables // same</td>
<td></td>
</tr>
<tr>
<td>zips</td>
<td></td>
</tr>
</tbody>
</table>
## Where is my InnoDB?

<table>
<thead>
<tr>
<th>MySQL</th>
<th>MongoDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>MyISAM</td>
<td>MMAPv1 memory mapped stored engine,</td>
</tr>
<tr>
<td>InnoDB</td>
<td>WiredTiger transactional, with</td>
</tr>
<tr>
<td></td>
<td>compression, btree</td>
</tr>
<tr>
<td>TokuDB</td>
<td>Percona Memory Engine</td>
</tr>
<tr>
<td>MyRocks (RocksDB)*</td>
<td>MongoRocks (RocksDB)</td>
</tr>
</tbody>
</table>
Where is my Processlist?

mysql> show processlist
*************************** 1. row
***************************
    Id: 137259
   User: root
   Host: localhost
db: geonames
Command: Query
   Time: 0
     State: init
    Info: show processlist
Rows_sent: 0
Rows_examined: 0
1 row in set (0.00 sec)

> db.currentOp()
{
    "inprog": [
        {
            "desc": "conn28",
            "threadId": "0x19b85260",
            "connectionId": 28,
            "opid": 27394208,
            "active": true,
            "secs_running": 3,
            "microsecs_running": NumberLong(3210539),
            "op": "query",
            "ns": "osm.points3",
            "query": {
                "name": "Durham"
            },
            "planSummary": "COLLSCAN",
            "client": "127.0.0.1:58835",
            "numYields": 24905,
            "locks": {
                "Global": "r",
                "Database": "r",
                "Collection": "r"
            },
            "waitingForLock": false,
        }
    ]
}
### Where are my **Grants**?

<table>
<thead>
<tr>
<th>MySQL Command</th>
<th>MongoDB Command</th>
</tr>
</thead>
</table>
| `grant all on *.* to user@localhost identified by 'pass';` | `use products`<br>`db.createUser({
  user: "accountUser",
  pwd: "password",
  roles: [{ "readWrite", "dbAdmin" }]
})` |
### MySQL

```sql
mysql> show keys from zips;
*************************** 1. row
***************************
  Table: zips
  Non_unique: 0
  Key_name: PRIMARY
  Seq_in_index: 1
  Column_name: id
  Collation: A
  Cardinality: 0
  Sub_part: NULL
  Packed: NULL
  Null: ON
  Index_type: BTREE
  Comment: 

*************************** 2. row
***************************
  Table: zips
  Non_unique: 1
  Key_name: postal_code
  Seq_in_index: 1
```

### MongoDB

```javascript
> db.zips.getIndexes()
[ {   "v" : 1,
    "key" : {   "_id" : 1
    },
    "name" : "_id",
    "ns" : "zips.zips"
  }
]```
mysql> alter table zips add key (postal_code);
Query OK, 0 rows affected (0.10 sec)
Records: 0  Duplicates: 0
Warnings: 0

> db.zips.createIndex({ state : 1 } )
{ "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 1,
  "numIndexesAfter" : 2,
  "ok" : 1
}

// Index can be sorted:
> db.zips.createIndex({ state : -1 } )
{ "createdCollectionAutomatically" : false,
  "numIndexesBefore" : 2,
  "numIndexesAfter" : 3,
  "ok" : 1
}
### MySQL

```sql
mysql> set global long_query_time = 0.1;
Query OK, 0 rows affected (0.02 sec)

mysql> set global slow_query_log = 1;
Query OK, 0 rows affected (0.02 sec)

mysql> show global variables like 'slow_query_log_file';
+---------------------+---------------------------------------------+
| Variable_name       | Value                        |
+---------------------+---------------------------------------------+
| slow_query_log_file | /var/lib/mysql/thor-slow.log |
+---------------------+---------------------------------------------+
1 row in set (0.00 sec)
```

### MongoDB

```javascript
db.setProfilingLevel(level, slowms)
Level: 0 for no profiling, 1 for only slow operations, or 2 for all operations.
Slowms = long_query_time but in milliseconds
> db.setProfilingLevel(2, 100);
{ "was" : 0, "slowms" : 100, "ok" : 1 }

> db.system.profile.find( { millis : { $gt : 100 } } ).pretty()
{
    "op" : "query",
    "ns" : "zips.zips",
    "query" : {
        "city" : "DURHAM"
    },
    "ntoreturn" : 0,
...
Export from MySQL 5.7:

```sql
mysql> SELECT JSON_OBJECT('name', replace(name, '"', ''), 'other_tags', replace(other_tags, '"', ''), 'geometry', st_asgeojson(shape)) as j
       FROM `points` INTO OUTFILE '/var/lib/mysql-files/points.json';
Query OK, 13660667 rows affected (4 min 1.35 sec)
```
Load to MongoDB (parallel):

mongoimport --db osm --collection points -j 24 --file /var/lib/mysql-files/points.json

2016-04-11T22:38:10.029+0000 connected to: localhost
2016-04-11T22:38:13.026+0000 [........................] osm.points 31.8 MB/2.2 GB (1.4%)
2016-04-11T22:38:16.026+0000 [........................] osm.points 31.8 MB/2.2 GB (1.4%)
2016-04-11T22:38:19.026+0000 [........................] osm.points 31.8 MB/2.2 GB (1.4%)
...
2016-04-11T23:12:13.447+0000 [################################################] osm.points 2.2 GB/2.2 GB (100.0%)
2016-04-11T23:12:15.614+0000 imported 13660667 documents
Thinking about using MongoDB?
Consider trying out Percona Server for MongoDB
Percona Server for MongoDB 3.4

100% Compatible with MongoDB 3.4 Community Edition

Open Source with Alternatives to many MongoDB Enterprise Features

MongoRocks (RocksDB) and Percona Memory Engine

New: Sensitive Data Masking

New: Query Sampling

New: Hot Backup for WiredTiger and MongoRocks
Percona Memory Engine for MongoDB Benchmarks
WiredTiger vs MongoRocks – write intensive
Database Performance Matters