# SQL VS NOSQL(MongoDB)

**MySQL** uses the Structured Query Language (**SQL**).

MySQL represents data in tables and rows.

**MongoDB** uses JavaScript as query language while **MySQL** uses the Structured Query Language (**SQL**).

MongoDB represents data as JSON documents.

# SQL VS NOSQL(MongoDB)

- ✓ INSERT INTO people(user_id, age, status) VALUES ("bcd001", 45, "A")

- ✓ ALTER TABLE people ADD join_date DATETIME

- ✓ ALTER TABLE people DROP COLUMN join_date

- ✓ db.people.insertOne( { user_id: "bcd001", age: 45, status: "A" } )

- ✓ db.people.updateMany( { }, { $set: { join_date: new Date() } } )

- ✓ db.people.updateMany( { }, { $unset: { "join_date": "" } } )

# SQL VS NOSQL(MongoDB)

- ✓ SELECT * FROM people
- ✓ SELECT id, user_id, status FROM people

```
SELECT *
FROM people
WHERE status = "A“

SELECT user_id, status
FROM people
WHERE status = "A"
```

- ✓ db.people.find()
- ✓ db.people.find(
  { },
  { user_id: 1, status: 1 }
  )

```
db.people.find(
   { status: "A" }
)

db.people.find(
   { status: "A" },
   { user_id: 1, status: 1, _id: 0 }
)
```

# SQL VS NOSQL(MongoDB)

```
SELECT *
FROM people
WHERE status != "A“
```

```
db.people.find(
   { status: { $ne: "A" } }
)
```

```
SELECT *
FROM people
WHERE status = "A"
AND age = 50
```

```
db.people.find(
   { status: "A",
     age: 50 }
)
```

```
SELECT *
FROM people
WHERE status = "A"
OR age = 50
```

```
db.people.find(
   { $or: [ { status: "A" } , { age: 50 } ] }
)
```

# SQL VS NOSQL(MongoDB)

```
SELECT *
FROM people
WHERE age > 25
```

```
db.people.find(
    { age: { $gt: 25 } }
)
```

```
SELECT *
FROM people
WHERE age < 25
```

```
db.people.find(
   { age: { $lt: 25 } }
)
```

```
SELECT *
FROM people
WHERE age > 25
AND   age <= 50
```

```
db.people.find(
   { age: { $gt: 25, $lte: 50 } }
)
```

# SQL VS NOSQL(MongoDB)

```
SELECT *
FROM people
WHERE user_id like "%bc%“
```

```
db.people.find( { user_id: /bc/ } )
-or-
db.people.find( { user_id: { $regex: /bc/ } } )
```

```
SELECT *
FROM people
WHERE status = "A"
ORDER BY user_id ASC
```

```
db.people.find( { status: "A" } ).sort( { user_id: 1 } )
```

# SQL VS NOSQL(MongoDB)

```
SELECT *
FROM people
WHERE status = "A"
ORDER BY user_id DESC
```

```
db.people.find( { status: "A" } ).sort( { user_id: -1 } )
```

```
SELECT COUNT(*)
FROM people
```

```
db.people.count()
or
db.people.find().count()
```

```
SELECT COUNT(user_id)
FROM people
```

```
db.people.count( { user_id: { $exists: true } } )
or
db.people.find( { user_id: { $exists: true } } ).count()
```

# SQL VS NOSQL(MongoDB)

```
SELECT COUNT(*)
FROM people
WHERE age > 30
```

```
db.people.count( { age: { $gt: 30 } } )
or
db.people.find( { age: { $gt: 30 } }
).count()
```

```
SELECT DISTINCT(status)
FROM people
```

```
db.people.aggregate( [ { $group : {
_id : "$status" } } ] )

or
db.people.distinct( "status" )
```

```
SELECT *
FROM people
LIMIT 1
```

```
db.people.findOne()
or
db.people.find().limit(1)
```

# SQL VS NOSQL(MongoDB)

```
SELECT * FROM people LIMIT 5 SKIP 2
```

```
db.people.find().limit(5).skip(2)
```

# SQL VS NOSQL(MongoDB)

```
UPDATE people SET status = "C"
WHERE age > 25
```

```
db.people.updateMany(
   { age: { $gt: 25 } },
   { $set: { status: "C" } }
)
```

```
UPDATE people
SET age = age + 3
WHERE status = "A“
```

```
db.people.updateMany(
   { status: "A" } ,
   { $inc: { age: 3 } }
)
```

```
DELETE FROM people
WHERE status = "D“
```

```
db.people.deleteMany( { status: "D" }
)
```

```
DELETE FROM people
```

```
db.people.deleteMany({})
```

# SQL VS NOSQL(MongoDB)

DROP TABLE people

db.people.drop()

# SQL VS NOSQL(MongoDB)

```
update mydb
set VALORE_01 = 5.5
where NUM = -1;
```

```
db.mydb.update(
    {"NUM" : -1},
    { $set : { "VALORE_01" : 5.5}
});
```

# MongoDB CRUD Operations

## Enter MySQL Query:

```sql
SELECT Type FROM Places
WHERE Type IN('Type1','Type 2')
ORDER BY Type;
```

## MongoDB Syntax:

```mongodb
db.Places.find({
    "Type": {
        "$in": ["Type1", "Type 2"]
    }
}, {
    "Type": 1
}).sort({
    "Type": 1
});
```

# MongoDB CRUD Operations

## SQL VS NoSQL Queries

NoSQL Query:

```
db.users.find(                          ← collection
    { age: { $gt: 18 } },               ← query criteria
    { name: 1, address: 1 }             ← projection
).limit(5)                              ← cursor modifier
```

SQL Query:

```
SELECT _id, name, address   ← projection
FROM   users                ← table
WHERE  age > 18             ← select criteria
LIMIT  5                    ← cursor modifier
```

# SQL VS NOSQL(MongoDB)

**MongoDB CRUD Operations**

```
db.users.insertOne(          ← collection
  {
    name: "sue",             ← field: value
    age: 26,                 ← field: value   } document
    status: "pending"        ← field: value
  }
)
```

```
db.users.updateMany(                  ← collection
  { age: { $lt: 18 } },               ← update filter
  { $set: { status: "reject" } }      ← update action
)
```

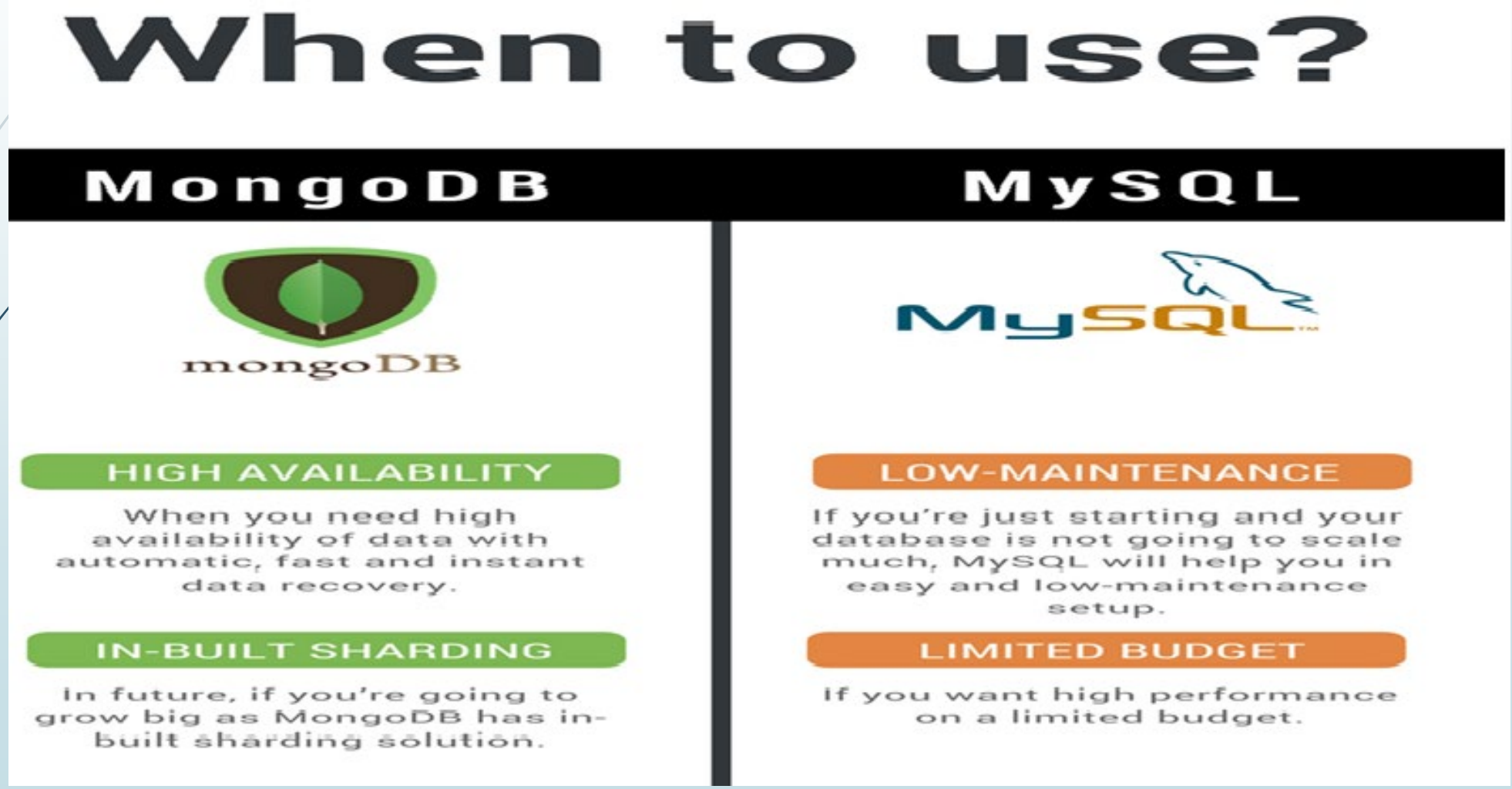
```
db.users.find(                        ← collection
  { age: { $gt: 18 } },               ← query criteria
  { name: 1, address: 1 }             ← projection
).limit(5)                            ← cursor modifier
```

```
db.users.deleteMany(                  ← collection
  { status: "reject" }                ← delete filter
)
```

# MongoDB CRUD Operations

| MySQL | MongoDB |
|---|---|
| **INSERT** | |
| INSERT INTO account (<br>`A/c number`, `first name`, `last name`<br>)<br>VALUES (<br>'12345746352',<br>'Mark',<br>'Jacobs'<br>); | db.account.insert({<br>A/c number: "12345746352",<br>first name: "Mark",<br>last name: "Jacobs"<br>}); |
| **UPDATE** | |
| UPDATE account<br>SET contact number = 9426227364<br>WHERE A/c number = '12345746352' | db.account.update(<br>{ A/c number: '12345746352' },<br>{ $set: {contact number: 9426227364<br>} }<br>); |
| **DELETE** | |
| DELETE FROM account<br>WHERE e-mail address =<br>'jv1994@gmail.com'; | db.account.remove({<br>"E-mail address": "jv1994@gmail.com"<br>}); |

# SQL VS NOSQL(MongoDB)



# SQL VS NOSQL(MongoDB)

**UNSTABLE SCHEMA**

If you have an unstable schema and you want to reduce your schema migration cost.

**NO DBA**

If you don't have a Database Administrator (but you'll have to hire one if you're going to go BIG).

**CLOUD COMPUTING**

If most of your services are cloud-based, MongoDB is best suitable for you

**FIXED SCHEMA**

If you've fixed schema and data structure isn't going to change over the time like WikiPedia

**HIGH TRANSACTION**

If high transaction rate is going to be your requirement (like BBC around 30,000 inserts/minute, 4000 selects/hour)

**DATA SECURITY**

If data security is your top priority, MySQL is the most secure DBMS.