NoSQL - Mongo

Agenda

- Mongo CRUD
- NoSQL concepts
- Normalization
- Codd's rules

Mongo

Insert

```
db.dept.insert({
    _id: 50,
    dname: "Training",
    loc: "Pune"
});
// insert record in dept collection
// if dept collection doesn't exists, it will be auto-created
```

```
// can also create collection  explicitly
db.createCollection("students");
```

- If _id field is not given, id is auto-generated by the "client".
- "unique" auto-generated id is of 12 bytes.
    - client process id (2 bytes)
    - client machine id (3 bytes) (each client get unique id from server upon connection)
    - timestamp (4 bytes) -- time at which record is inserted
    - counter (3 bytes) -- counter managed by client

Query

- Like SELECT.
- db.col.find({criteria}, {projection});
    - criteria -- WHERE clause
    - selected fields
- Projection
    - 1: display field, 0: hide field
    - cannot combine display & hide.
        - { _id: 1, ename: 1, sal: 1 }
        - { job: 0, sal: 0, hire: 0, comm: 0 }
        - { ename: 1, sal: 0, hire: 1, comm: 0 } --> error
    - By default id is visible.

- Criteria
  - Relational: $lt, $gt, $lte, $gte, $eq, $ne
  - Logical: $and, $or, $nor
  - Other Operators: $in, $nin, $regex, $type, ...

```
use classwork;

show collections;
```

```
db.emp.find();

db.emp.find({}, { _id: 1, ename: 1, sal: 1 });

db.emp.find({}, { job: 0, sal: 0, hire: 0, comm: 0 });

db.emp.find({}, { ename: 1, sal: 0, hire: 1, comm: 0 }); // error

db.emp.find({}, { ename: 1, job: 1, sal: 1 });
// id is default included

db.emp.find({}, { _id: 0, ename: 1, job: 1, sal: 1 });
// hide id -- exception -- hiding id is allowed in inclusion projection.
```

```
// empty criteria -- all records
db.emp.find({});

// find emp with sal > 2500
db.emp.find({ sal: { $gt: 2500 } });

// find clerk
db.emp.find({ job: { $eq: 'CLERK' } });
db.emp.find({ job: 'CLERK' });

// find all emps working in dept 10 and 20.
db.emp.find({ deptno: { $in: [10, 20] } });

// find all CLERK working in dept 20.
db.emp.find({ $and: [
    { job: 'CLERK' },
    { deptno: 20 }
] });

// find emps with sal <= 1500 or job ANALYST.
db.emp.find({ $or: [
    { sal: { $lte: 5000 } },
    { job: 'ANALYST' }
]});
```

```
// find all emps whose name start with S.
db.emp.find({
    ename: { $regex: /^S/ }
});

db.emp.find({
    ename: { $regex: /^s/i }
});
// --> /regex/mode
// regex -- using wild card chars
// mode -- "i": case insensitive
```

Update/Upsert

- db.col.update({criteria}, {new_object or changes using $set});
  - by default upsert = false.
- updateOne(), updateMany()
- UPSERT operation --> If exists then UPDATE, otherwise INSERT.
  - db.col.update({criteria}, {new_object or changes using $set}, true);
  - arg 3: upsert = true
  - If record doesn's exists, a new record is created with given criteria and then it is updated with given values.

```
db.dept.insert({_id: 50, dname: 'TRAINING', loc: 'PUNE'});

db.dept.find();

db.dept.update({_id: 50}, { loc: 'BANGLORE' });
// {_id: 50, dname: 'TRAINING', loc: 'PUNE'} replaced by {_id: 50, loc: 'BANGLORE'
}

db.dept.find();

db.emp.update({ ename: 'KING' }, { sal: 5500 });
// whole record is replaced by { _id + sal }

db.emp.find();

db.emp.update({ ename: 'JAMES' }, {
    $set: {
        sal: 1000
    }
});

db.emp.find();

db.emp.update({ ename: 'JAMES' }, {
    $set: {
        sal: 1100,
        mgr: 7836
```

```
    }
});

// change sal=1200, comm=200, job=SALESMAN for emp with name = 'JOHN'
db.emp.update( {ename: 'JOHN'}, {
    $set: {
        sal: 1200,
        comm: 200,
        job: 'SALESMAN'
    }
});
// no records found, and hence no records modified

db.emp.update( {ename: 'JOHN'}, {
    $set: {
        sal: 1200,
        comm: 200,
        job: 'SALESMAN'
    }
},
true );

db.emp.find();

db.dept.update({_id: 60}, { $set: { dname: "SECURITY" } }, true);

db.dept.find().count();
```

Delete

- db.col.remove({criteria}); --> delete one or more records as per criteria
- db.col.deleteOne({criteria}); --> delete first as per criteria
- db.col.deleteMany({criteria}); --> delete one or more records as per criteria

```
db.dept.find();

db.dept.deleteMany({ _id: { $gte : 50 } });

db.dept.find();

db.dept.deleteMany({});
// delete all records (but not collection)

show collections;

db.dept.drop();

show collections;
```

Aggregation Pipeline

- Not in syllabus
- Group By, Projection, Joins, ...
- Pipeline -- set of stages

```
db.collection_name.aggregate([
{
    $group: { ... }
},
{
    $match: { ... }
},
{
    $sort: { ... }
},
{
    $limit: { ... }
}
]);
```

Indexes

- faster searching
- Types of Indexes
    - Simple index
    - Unique index
        - By default _id is unique index.
    - Composite index
    - TTL index (Time To Live)
        - Auto expire/delete object after certain time.
    - GeoSpatial index
        - Location based (long + lat) analysis

```
db.emp.createIndex({
    'job': 1
});

db.emp.createIndex({
    'ename': 1,
}, {
    unique: true
});

db.emp.createIndex({
    'deptno': 1,
    'job': 1
});

db.emp.getIndexes();
```

Mongo Import

- Import form JS script.
    - cmd> mongo -d classwork empdept.js
- MongoImport
    - cmd> mongoimport -type csv -headerline -d classwork -c emp emp.csv