exclude fields →

```
db.collection.find( { query } , {date: 0, 'address.zip': 0})
```

↓ projection to remove.

Count documents In Mongo

```
db.collection.countDocuments(<query>, <options>)
```

BSON → Optimized for storage, retrieval & transmission across the wire → More secure than text JSON
More datatypes than JSON
↓
represented through Document class.

Append is used to instantiate a document with fields & values.

In Java example

```
⟹ Document test = new Document("_id", new ObjectId()).
        .append("a", ---)
        .app --
InsertOneResult res = collection.insertOne(test);
  BsonValue Id = res.getInsertedID();
```

InsertMany returns a InsertManyResult

ex

```
        List<Document> accounts = Arrays.asList(doc1, doc2);
    InsertManyResult res = Collection.insertMany(accou..);
res.getInsertedIds().forEach((x, y -> sys.o = (y.asObjectID()));
```

Find queries in Java

```
# try(MongoCursor<Document> cursor = collection.find(
    and(gte('balance, 1000), eq("account_type", "checking")))
    .iterator())
  {
    while(cursor.hasNext()){
        System.out.println(cursor.next.toJson());
    }
  }
}.
```

if query given

```
Document doc = collection.find(query).first() → to get the first doc
System.out.print(doc!=null ? doc.toJson() : null);
```

UpdateOne()

```
System.out.println(<collection>.updateOne(<filter>,
                                  <update>));
```

To update in an array → Updates.push();
Updates.pull()
" . popLast();

UpdateMany()

```
<collections>.updateMany(<filter>, <update>);
```

Delete → Bson query = Filters.eq(----);

```
DeleteResult del = collection.deleteOne(query);
System.out.println(---() del.getDeletedCount());
~~delgdeleted~~
```

=) deleteMany () for deleting more than 1

=) delete method with empty query will delete all docs

Multidocument transactions -) A multi-document transaction is an operation that requires atomicity of reads and or/writes to multiple documents.

A transaction is a sequence of db operations that represent a single unit of work

if a transaction is cancelled or doesn't complete all writes are descarded.

Transaction steps:-

① Start a client Session

② Define the transaction options (optional)
③ Sequence of operation to perform
④ Start clientSession withTransaction() method.
⑤ Release the resources used by transaction

Default mongo cancels any transaction which ran for more than 60sec.

=) also always close the resources used.

For example practise the session after this lesson.

MongoDB Indexes

Special data structures, store small portion of the data. ordered and easy to search efficiently, points to the document identity

Indexes improve → query performance, speed up queries, Reduce disk I/O, reduce resources required. It also supports queries like equality matches, range-based operations & return sorted results.

Default Index → _id

When we insert or update, we need to update the index data structure

Too many indexes ↓ the performance.

Most common Index types → single field, compound field multikey Index operate on an array field

Single field Index → db.collection.createIndex({field: 1});

mongo returns the Index name.

Order for ascending

lets say for unique values
↓
db.coll.createIndex({email: 1}, {unique: true}).

db.coll.getIndexes() → to get all Index on collection

to see which Index is used on a query

⇒ db.coll.explain().find({birthdate: {$gt: ISODate("1995-01-01")}})