

Referência Rápida de MongoDB

CRUD, consultas, agregação, índices, design de schema

Conexão

String de Conexão

```
mongosh "mongodb://localhost:27017"
mongosh "mongodb://user:pass@host:27017/mydb"
mongosh "mongodb+srv://user:pass@cluster.mongodb.net/mydb"
```

Conexão com Driver (Node.js)

```
const { MongoClient } = require('mongodb');
const client = new MongoClient(uri);
await client.connect();
const db = client.db('mydb');
```

Bancos de Dados e Coleções

Operações com Banco de Dados

```
show dbs
use mydb
db.dropDatabase()
```

Operações com Coleção

```
db.createCollection("users")
show collections
db.users.drop()
```

Coleção Limitada

```
db.createCollection("logs", {
  capped: true, size: 10485760, max: 5000
})
```

Operações CRUD

Inserir

```
db.users.insertOne({ name: "Alice", age: 30 })
db.users.insertMany([
  { name: "Bob", age: 25 },
  { name: "Carol", age: 28 }
])
```

Buscar

```
db.users.findOne({ name: "Alice" })
db.users.find({ age: { $gte: 25 } })
db.users.find({}, { name: 1, _id: 0 })
db.users.find().sort({ age: -1 }).limit(10)
```

Atualizar

```
db.users.updateOne(
  { name: "Alice" },
  { $set: { age: 31, city: "Boston" } }
)
db.users.updateMany(
  { active: false },
  { $set: { archived: true } }
)
```

Excluir

```
db.users.deleteOne({ name: "Alice" })
db.users.deleteMany({ active: false })
```

Substituir e Upsert

```
db.users.replaceOne(
  { name: "Alice" },
  { name: "Alice", age: 32, city: "NYC" }
)
db.users.updateOne(
  { email: "a@ex.com" },
  { $set: { name: "Alice" } },
  { upsert: true }
)
```

Operadores de Consulta

Comparação

| | |
|---------------------|----------------------------|
| \$eq / \$ne | Igual / diferente |
| \$gt / \$gte | Maior que / maior ou igual |
| \$lt / \$lte | Menor que / menor ou igual |
| \$in / \$nin | No array / fora do array |

Lógicos

| | |
|-----------------|---------------------------------------|
| \$and | Todas as condições devem corresponder |
| \$or | Pelo menos uma condição corresponde |
| \$not | Nega uma condição |
| \$exists | Campo existe (true/false) |
| \$regex | Correspondência de expressão regular |

Operadores de Atualização

| | |
|------------------------|---------------------------------------|
| \$set | Definir valor do campo |
| \$unset | Remover campo |
| \$inc | Incrementar valor numérico |
| \$push / \$pull | Adicionar / remover elemento do array |
| \$addToSet | Adicionar ao array se não presente |
| \$rename | Renomear um campo |

Agregação

Estágios do Pipeline

| | |
|-------------------------|------------------------------------|
| \$match | Filtrar documentos (como WHERE) |
| \$group | Agrupar e agregar |
| \$project | Remodelar documentos (como SELECT) |
| \$sort | Ordenar resultados |
| \$limit / \$skip | Paginação |
| \$lookup | Left outer join com outra coleção |
| \$unwind | Desconstruir array em documentos |

Exemplo de Agregação

```
db.orders.aggregate([
  { $match: { status: "completed" } },
  { $group: {
    _id: "$customer_id",
    total: { $sum: "$amount" },
    count: { $sum: 1 }
  }},
  { $sort: { total: -1 } },
  { $limit: 10 }
])
```

Índices

Criar e Remover

```
db.users.createIndex({ email: 1 }, { unique: true })
db.users.createIndex({ name: 1, age: -1 })
db.users.createIndex({ location: "2dsphere" })
db.users.dropIndex("email_1")
```

Tipos de Índice

| | |
|---------------------|---|
| Single field | Índice em um campo ({ name: 1 }) |
| Compound | Múltiplos campos ({ a: 1, b: -1 }) |
| Text | Busca de texto completo ({ field: 'text' }) |
| 2dsphere | Consultas geoespaciais |
| TTL | Expirar documentos automaticamente após tempo |

Informações de Índice

```
db.users.getIndexes()
db.users.find({ name: "Alice" }).explain()
```

Design de Schema

Embutir vs Referenciar

| | |
|------------------|---|
| Embed | 1:1 ou 1:poucos, dados lidos juntos |
| Reference | 1:muitos, dados acessados independentemente |
| Embed | Sub-documento raramente excede 16 MB |
| Reference | Relacionamentos muitos-para-muitos |

Validação de Schema

```
db.createCollection("users", {
  validator: { $jsonSchema: {
    bsonType: "object",
    required: ["name", "email"],
    properties: {
      name: { bsonType: "string" },
      email: { bsonType: "string" }
    }
  }
})
```

Replicação

Conceitos de Replica Set

| | |
|------------------|---|
| Primary | Recebe todas as gravações |
| Secondary | Replica do primário, pode servir leituras |
| Arbiter | Vota em eleições, não armazena dados |

Comandos de Replica Set

```
rs.initiate()
rs.add("mongo2:27017")
rs.addArb("mongo3:27017")
rs.status()
rs.conf()
```

Padrões Comuns

Transações

```
const session = client.startSession();
session.startTransaction();
await db.collection("accounts").updateOne(
  { _id: 1 }, { $inc: { bal: -100 } }, { session });
await db.collection("accounts").updateOne(
  { _id: 2 }, { $inc: { bal: 100 } }, { session });
await session.commitTransaction();
```

Gravação em Lote

```
db.users.bulkWrite([
  { insertOne: { document: { name: "Dan" } } },
  { updateOne: {
    filter: { name: "Alice" },
    update: { $set: { age: 31 } }
  }},
  { deleteOne: { filter: { name: "old" } } }
])
```

Referência Rápida de MongoDB

Change Streams

```
const stream = db.collection("orders")
  .watch([{$match: { "fullDocument.status": "new" }}]);
stream.on("change", (change) => {
  console.log(change.fullDocument);
});
```

Comandos mongosh

Helpers do Shell

| | |
|---|--------------------------------|
| show dbs | Listar bancos de dados |
| show collections | Listar coleções no banco atual |
| db.stats() | Estatísticas do banco de dados |
| db.collection.stats() | Estatísticas da coleção |
| db.collection.countDocuments({}) | Contar documentos |
| db.collection.distinct('field') | Valores distintos de um campo |

Exportar e Importar

```
mongoexport --db=mydb --collection=users \
  --out=users.json
mongoimport --db=mydb --collection=users \
  --file=users.json
mongodump --db=mydb --out=/backup/
mongorestore --db=mydb /backup/mydb/
```