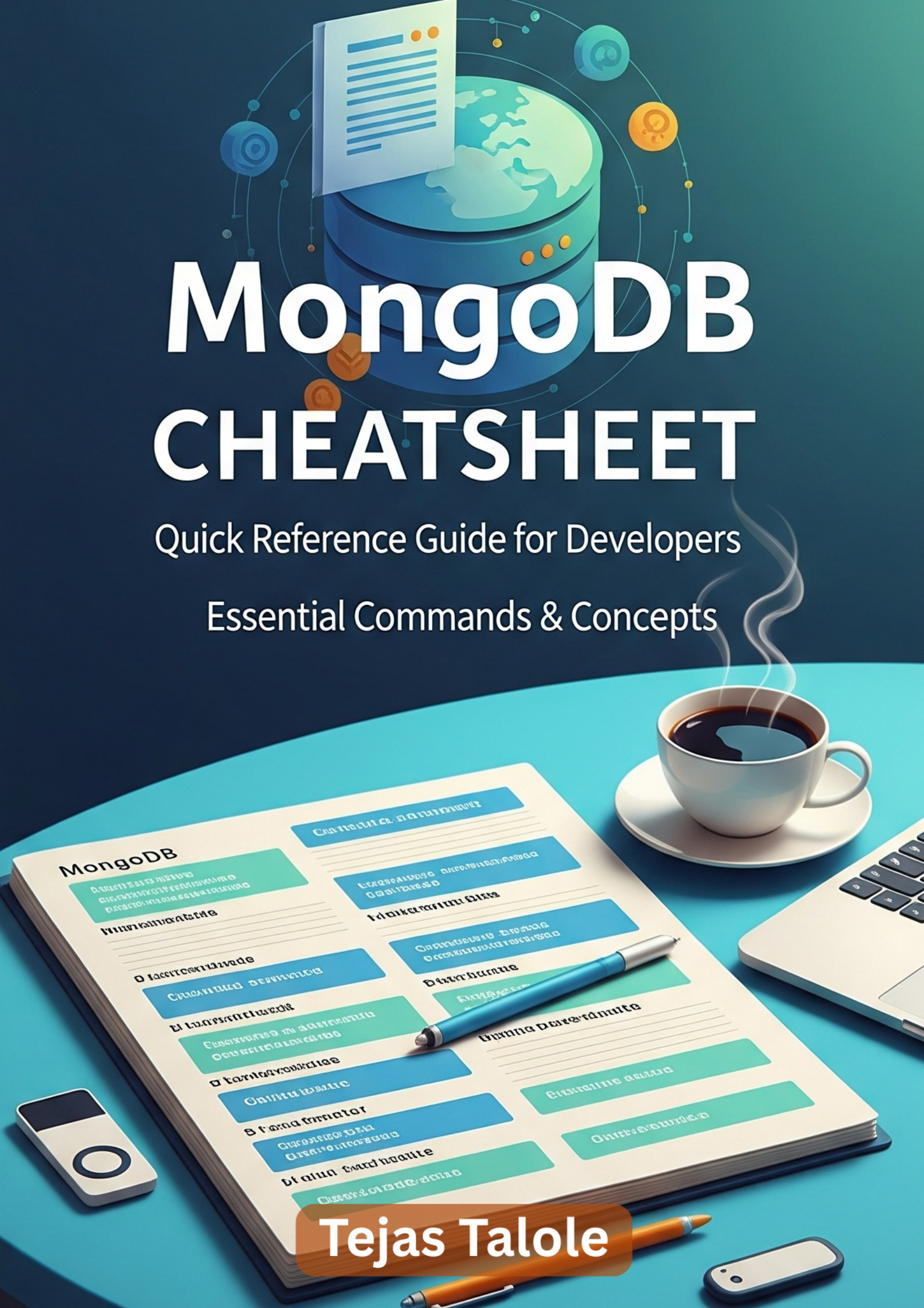
MongoDB
CHEATSHEET
Quick Reference Guide for Developers
Essential Commands & Concepts
MongoDB
Tejas Talole

MongoDB Full Cheatsheet

1. MongoDB Basics

```
mongosh                      # Start MongoDB shell
show dbs                     # Show all databases
use mydb                     # Switch / create database
db                           # Show current database
show collections             # Show all collections in DB
db.dropDatabase()            # Drop current database
```

2. Collections

```
db.createCollection("users")   // Create collection
db.users.drop()                // Drop collection
```

3. Insert Documents

```
db.users.insertOne({ name: "Alice", age: 25 })

db.users.insertMany([
  { name: "Bob", age: 30 },
  { name: "Charlie", age: 22 }
])
```

4. Find Documents

```
db.users.find()                          // All documents
db.users.findOne()                       // First document
db.users.find({ age: 25 })               // Filter
db.users.find({}, { name: 1, _id: 0 })   // Projection (fields)
```

5. Query Operators

```
db.users.find({ age: { $eq: 25 } })     // Equal
db.users.find({ age: { $ne: 25 } })     // Not equal
db.users.find({ age: { $gt: 20 } })     // Greater than
db.users.find({ age: { $gte: 20 } })    // Greater or equal
db.users.find({ age: { $lt: 30 } })     // Less than
db.users.find({ age: { $lte: 30 } })    // Less or equal
db.users.find({ age: { $in: [20, 25] } })  // In list
db.users.find({ age: { $nin: [20, 25] } }) // Not in list
```

◇ 6. Logical Operators

```
db.users.find({ $and: [ { age: { $gt: 20 } }, { age: { $lt: 30 } } ] })
db.users.find({ $or: [ { age: 25 }, { name: "Alice" } ] })
db.users.find({ age: { $not: { $gt: 25 } } })
db.users.find({ age: { $exists: true } })
```

◇ 7. Update Documents

```
db.users.updateOne(
  { name: "Alice" },
  { $set: { age: 26 } }
)

db.users.updateMany(
  { age: { $lt: 25 } },
  { $set: { status: "young" } }
)

db.users.replaceOne(
  { name: "Bob" },
  { name: "Bobby", age: 31 }
)
```

◇ 8. Delete Documents

```
db.users.deleteOne({ name: "Alice" })
db.users.deleteMany({ age: { $lt: 18 } })
```

◇ 9. Sorting & Limiting

```
db.users.find().sort({ age: 1 })   // Ascending
db.users.find().sort({ age: -1 })  // Descending
db.users.find().limit(5)           // Limit
db.users.find().skip(5)            // Skip
```

◇ 10. Indexes

```
db.users.createIndex({ name: 1 })      // Create index
db.users.getIndexes()                  // Show indexes
db.users.dropIndex("name_1")           // Drop index
```

◇ 11. Aggregation Framework

```
db.users.aggregate([
  { $match: { age: { $gte: 18 } } },             // Filter
  { $group: { _id: "$age", count: { $sum: 1 } } }, // Group by age
  { $sort: { count: -1 } }                       // Sort
])

db.orders.aggregate([
  { $group: { _id: "$customerId", total: { $sum: "$amount" } } }
])
```

◇ 12. Relationships

Embedded Document

```
{
  name: "Tejas",
  address: { city: "Pune", pincode: 411001 }
}
```

Reference (Normalization)

```
{
  name: "Tejas",
  orders: [ ObjectId("64d2..."), ObjectId("64d3...") ]
}
```

◇ 13. Data Types

- String → `"Hello"`
- Number → `123`
- Boolean → `true / false`
- Array → `[1, 2, 3]`
- Object → `{ key: value }`
- Null → `null`
- Date → `ISODate("2025-08-29T12:00:00Z")`
- ObjectId → `_id: ObjectId("507f1f77bcf86cd799439011")`

◇ 14. Backup & Restore

```
mongodump --db=mydb --out=/backup/        # Backup
mongorestore /backup/                     # Restore
```

◇ 15. User & Role Management

```
use admin
db.createUser({
  user: "tejas",
  pwd: "password123",
  roles: [ { role: "readWrite", db: "mydb" } ]
})

db.dropUser("tejas")
```

◇ 16. Useful Shell Commands

```
db.stats()            // Database stats
db.users.stats()      // Collection stats
db.users.countDocuments()    // Count
db.users.distinct("age")     // Unique values
```