**MONGODB CRUD OPERATIONS (SMART KNOWLEDGE HUB)**

**1. Admin Collection**

* **Create:**

db.admin.insertOne({  
 \_id: "admin466",  
 name: "New Admin",  
 email: "newadmin@store.com",  
 role: "manager",  
 permissions: ["manage\_users", "manage\_orders"],  
 last\_login: new Date()  
});

* **Read (Find all managers):**

db.admin.find({ role: "manager" });

* **Update (Update permissions for admin 'admin456'):**

db.admin.updateOne(  
 { \_id: "admin456" },  
 { $set: { permissions: ["manage\_users", "update\_inventory", "audit\_logs"] } }  
);

* **Delete (Remove admin by \_id):**

db.admin.deleteOne({ \_id: "admin465" });

**2. Categories Collection**

* **Create:**

db.categories.insertOne({  
 \_id: "category111",  
 name: "Toys",  
 description: "Children's toys and games.",  
 parent\_category: null,  
 product\_count: 50  
});

* **Read (Find all top-level categories):**

db.categories.find({ parent\_category: null });

* **Update (Change product\_count for 'category101'):**

db.categories.updateOne(  
 { \_id: "category101" },  
 { $inc: { product\_count: 10 } }  
);

* **Delete (Remove a category):**

db.categories.deleteOne({ \_id: "category110" });

**3. Inventory Collection**

* **Create:**

db.inventory.insertOne({  
 \_id: "inventory125",  
 product\_id: "product799",  
 quantity\_available: 100,  
 supplier\_id: "supplier211",  
 last\_updated: new Date(),  
 low\_stock\_threshold: 15  
});

* **Read (Find low stock items):**

db.inventory.find({ $expr: { $lte: ["$quantity\_available", "$low\_stock\_threshold"] } });

* **Update (Restock inventory):**

db.inventory.updateOne(  
 { \_id: "inventory115" },  
 {   
 $inc: { quantity\_available: 25 },  
 $set: { last\_updated: new Date() }  
 }  
);

* **Delete (Remove an inventory record):**

db.inventory.deleteOne({ \_id: "inventory124" });

**4. Orders Collection**

* **Create:**

db.orders.insertOne({  
 \_id: "order122",  
 user\_id: "user133",  
 order\_date: new Date(),  
 status: "pending",  
 total\_amount: 149.99,  
 items: [{ product\_id: "product789", quantity: 1, price: 149.99 }]  
});

* **Read (Find all shipped orders):**

db.orders.find({ status: "shipped" });

* **Update (Change order status):**

db.orders.updateOne(  
 { \_id: "order113" },  
 { $set: { status: "completed" } }  
);

* **Delete (Cancel an order):**

db.orders.deleteOne({ \_id: "order119" });

**5. Payments Collection**

* **Create:**

db.payments.insertOne({  
 \_id: "payment123",  
 order\_id: "order122",  
 amount: 149.99,  
 method: "credit\_card",  
 status: "completed",  
 transaction\_date: new Date()  
});

* **Read (Find all completed payments):**

db.payments.find({ status: "completed" });

* **Update (Update payment status):**

db.payments.updateOne(  
 { \_id: "payment114" },  
 { $set: { status: "completed" } }  
);

* **Delete (Delete a payment record):**

db.payments.deleteOne({ \_id: "payment120" });

**6. Products Collection**

* **Create:**

db.products.insertOne({  
 \_id: "product799",  
 name: "Smart Watch",  
 description: "Waterproof smartwatch with fitness tracking.",  
 price: 199.99,  
 stock\_quantity: 75,  
 category: "electronics",  
 images: ["https://example.com/smartwatch1.jpg"],  
 ratings: 4.6  
});

* **Read (Find products in electronics category):**

db.products.find({ category: "electronics" });

* **Update (Adjust stock quantity):**

db.products.updateOne(  
 { \_id: "product789" },  
 { $inc: { stock\_quantity: -5 } }  
);

* **Delete (Remove a product):**

db.products.deleteOne({ \_id: "product798" });

**7. Review Collection**

* **Create:**

db.review.insertOne({  
 \_id: "review124",  
 product\_id: "product799",  
 user\_id: "user133",  
 rating: 4,  
 comment: "Great features, but a bit pricey.",  
 review\_date: new Date()  
});

* **Read (Find reviews for a product):**

db.review.find({ product\_id: "product789" });

* **Update (Update a review comment and rating):**

db.review.updateOne(  
 { \_id: "review114" },  
 { $set: { rating: 4, comment: "Good sound but could improve bass." } }  
);

* **Delete (Remove a review):**

db.review.deleteOne({ \_id: "review123" });

**8. Shipping Collection**

* **Create:**

db.shipping.insertOne({  
 \_id: "shipping126",  
 order\_id: "order122",  
 carrier: "UPS",  
 tracking\_number: "1Z999AA10123456788",  
 estimated\_delivery: new Date("2025-07-30T00:00:00Z"),  
 actual\_delivery\_date: null,  
 cost: 12.99  
});

* **Read (Find shipping info for an order):**

db.shipping.find({ order\_id: "order112" });

* **Update (Set actual delivery date):**

db.shipping.updateOne(  
 { \_id: "shipping118" },  
 { $set: { actual\_delivery\_date: new Date() } }  
);

* **Delete (Remove shipping record):**

db.shipping.deleteOne({ \_id: "shipping125" });

**9. Suppliers Collection**

* **Create:**

db.suppliers.insertOne({  
 \_id: "supplier212",  
 name: "New Supplier Co.",  
 contact\_email: "contact@newsupplier.com",  
 address: "900 Supply Rd, Dallas, TX 75001",  
 products\_supplied: ["product799", "product800"],  
 rating: 4.5  
});

* **Read (Find suppliers with rating above 4.7):**

db.suppliers.find({ rating: { $gt: 4.7 } });

* **Update (Add a product to supplier's list):**

db.suppliers.updateOne(  
 { \_id: "supplier202" },  
 { $addToSet: { products\_supplied: "product799" } }  
);

* **Delete (Remove a supplier):**

db.suppliers.deleteOne({ \_id: "supplier211" });

**10. Users Collection**

* **Create:**

db.users.insertOne({  
 \_id: "user133",  
 name: "New User",  
 email: "newuser@example.com",  
 password\_hash: "hashedpassword999",  
 address: { street: "100 New St", city: "San Diego", zip: "92101", country: "USA" },  
 registration\_date: new Date(),  
 preferences: ["books", "clothing"]  
});

* **Read (Find users who prefer electronics):**

db.users.find({ preferences: "electronics" });

* **Update (Add a preference for a user):**

db.users.updateOne(  
 { \_id: "user123" },  
 { $addToSet: { preferences: "sports" } }  
);

* **Delete (Remove a user):**

db.users.deleteOne({ \_id: "user132" });