

UPDATES

Update Methods

update()

updateOne()

updateMany()

replaceOne()

Update Operators

\$set

\$unset

\$rename

\$inc

\$mul

\$currentDate

\$pop

\$addToSet

Update Methods Syntax

update() updateOne() updateMany()



db.<collection>.method(<query>, <update>, <options>)

\$set

```
db.shoppingCart.update(  
  { index: 3 },  
  {  
    $set: {  
      cartId: NumberInt(325),  
      customer: {  
        name: "Mike Foster",  
        email: "mfoster@test.com",  
        age: NumberInt(27)  
      },  
      cart: []  
    }  
  }  
);
```

● **Replace or set value of the field**

```
{ $set: { <fieldName1>: <value1>, ... } }
```

\$unset

- Remove certain fields

```
{ $unset: { <fieldName1>: <anyValue>, ... } }
```

```
db.shoppingCart.update(  
  { index: 4 },  
  {  
    $unset: {  
      processed: 1,  
      "customer.preferences": 1  
    }  
  }  
);
```

update()

● Update One Document

db.<collection>.update(<query>, <update>)



```
WriteResult({  
  "nMatched" : 1,  
  "nUpserted" : 0,  
  "nModified" : 1  
})
```

```
WriteResult({  
  "nMatched" : 0,  
  "nUpserted" : 0,  
  "nModified" : 0  
})
```

```
WriteResult({  
  "nMatched" : 1,  
  "nUpserted" : 0,  
  "nModified" : 0  
})
```

update() with {multi: true}

◎ Update Many Documents

db.<collection>.update(<query>, <update>, {multi: true})



```
WriteResult({  
  "nMatched" : 35,  
  "nUpserted" : 0,  
  "nModified" : 35  
})
```

```
WriteResult({  
  "nMatched" : 0,  
  "nUpserted" : 0,  
  "nModified" : 0  
})
```

```
WriteResult({  
  "nMatched" : 70,  
  "nUpserted" : 0,  
  "nModified" : 15  
})
```


updateOne()

● Update One Document

`db.<collection>.updateOne(<query>, <update>, <options>)`



```
{  
  "acknowledged" : true,  
  "matchedCount" : 1,  
  "modifiedCount" : 1,  
}
```

```
{  
  "acknowledged" : true,  
  "matchedCount" : 1,  
  "modifiedCount" : 0,  
}
```

updateMany()

● Update Many Documents

`db.<collection>.updateMany(<query>, <update>, <options>)`



```
{  
  "acknowledged" : true,  
  "matchedCount" : 10,  
  "modifiedCount" : 10,  
}
```

```
{  
  "acknowledged" : true,  
  "matchedCount" : 8,  
  "modifiedCount" : 3,  
}
```

replaceOne()

● Replace one Document

`db.<collection>.replaceOne(<query>, <replacement>, <options>)`

```
db.shoppingCart.replaceOne(
  { index: 1 },
  {
    index: 1,
    processed: false,
    cart: []
  }
)
```

Combine multiple update operators

```
db.shoppingCart.update(  
  { index: 4 },  
  {  
    $set: {  
      cartId: NumberInt(435),  
      "customer.name": "Samanta Larsen",  
      "customer.email": "slarsen@test.com"  
    },  
    $unset: {  
      newOrder: 1  
    }  
  }  
)
```

\$rename

● Rename certain fields

```
{ $rename: { <fieldName1>: <newFieldName1>, ... } }
```

```
db.shoppingCart.update(  
  { cartId: { $exists: true } },  
  {  
    $rename: {  
      cartId: "orderId",  
      anotherField: "anotherName"  
    }  
  },  
  { multi: true }  
);
```

Note

If field is absent,
nothing happens

\$currentDate

- Set value of the field to the current date

```
{ $currentDate: { <fieldName1>: true, ... } }
```

```
db.shoppingCart.update(  
  { cartId: 325 },  
  {  
    $currentDate: {  
      createdAt: true  
    }  
  }  
);
```

Array Update Operators

\$

\$push

\$addToSet

\$pull

\$pullAll

\$pop

\$push

- Appends element to the array

```
{ $push: { <arrayFieldName>: <element> } }
```

```
db.shoppingCart.update(  
  { cartId: 561 },  
  {  
    $push: {  
      cart: "item1"  
    }  
  }  
)
```

```
db.shoppingCart.update(  
  { cartId: 561 },  
  {  
    $push: {  
      cart: { $each:  
        ["item2", "item3"]  
      }  
    }  
  }  
)
```

\$each - modifier
operator



Note

Array with one element will be created automatically if field doesn't exist

\$addToSet

- Appends element to the array if this element doesn't exist already

```
{ $addToSet: { <arrayFieldName>: <element> } }
```

```
db.shoppingCart.update(  
  { cartId: 561 },  
  {  
    $addToSet: {  
      cart: "item1"  
    }  
  }  
)
```

```
db.shoppingCart.update(  
  { cartId: 561 },  
  {  
    $addToSet: {  
      cart: { $each:  
        ["item2", "item3"]  
      }  
    }  
  }  
)
```

Note

Array with one element will be created automatically if field doesn't exist

\$pop

- Removes first or last element in the Array

```
{ $pop: { <arrayFieldName>: < -1 | 1 > } }
```

```
db.shoppingCart.update(  
  { cartId: 561 },  
  {  
    $pop: {  
      cart: 1  
    }  
  }  
);
```

Note

<field>: -1 removes first element
<field>: 1 removes last element

\$pull

- Removes all elements in the Array matching specific element or condition

```
{ $pull: { <arrayFieldName>: <element | condition> } }
```

```
db.shoppingCart.update(  
  { cartId: 561 },  
  {  
    $pull: {  
      cart: "item1",  
      spentAmounts: { $gt: 400 }  
    }  
  }  
)
```

\$pullAll

- Removes all elements in the Array that match specified values

```
{ $pullAll: { <arrayFieldName>: [ <value1>, <value2>, ... ] } }
```

```
db.shoppingCart.update(  
  { cartId: 561 },  
  {  
    $pullAll: {  
      cart: ["item1", "item2"]  
    }  
  }  
);
```

Equivalent to

```
$pull: {  
  cart: {$in: ["item1", "item2"]}  
}
```

Positional Operator \$

- Points to certain element in the Array that matched query

```
{ $set: { <arrayField.$>: <value> } }
```

```
{ $set: { <arrayField.$.field>: <value> } }
```

```
db.shoppingCart.updateOne(  
  { cartId: 325, cart: "item2" },  
  {  
    $set: {  
      "cart.$": "updatedItem2"  
    }  
  }  
)
```

Positional Operator \$ in Nested Docs

```
{cart: [  
  {  
    title: "TV",  
    price: NumberInt(340),  
    quantity: NumberInt(2)  
  },  
  {  
    title: "Phone",  
    price: NumberInt(150),  
    quantity: NumberInt(1)  
  }  
]}
```

```
db.shoppingCart.updateOne(  
  {  
    "cart.title": "Phone"  
  },  
  {  
    $set: {  
      "cart.$.quantity": NumberInt(2)  
    }  
  }  
)
```

Update specific nested
document in the Array



Positional Operator \$ with \$elemMatch

```
{cart: [  
  {  
    title: "TV",  
    price: NumberInt(340),  
    quantity: NumberInt(2)  
  },  
  {  
    title: "Phone",  
    price: NumberInt(150),  
    quantity: NumberInt(1)  
  }  
]}
```

```
db.shoppingCart.updateOne(  
  {  
    cart: {$elemMatch: { title: "Phone", price: 150 }}  
  },  
  {  
    $set: {  
      "cart.$.quantity": NumberInt(2)  
    }  
  }  
)
```



Update specific nested document in the Array

\$inc

- Increment value by specified amount

```
{ $inc: { <fieldName1>: <amount>, ... } }
```

```
db.shoppingCart.update(  
  { cartId: 325 },  
  {  
    $inc: {  
      totalCartItems: NumberInt(1)  
    }  
  }  
);
```

Note

If number NumberInt(<number>) will be used without type, value will be converted to Double Type

SUMMARY

- **Update Methods**

- update()

- updateOne()

- updateMany()

- replaceOne()

- **Update Operators**

- \$set

- \$unset

- \$rename

- \$currentDate

- \$elemMatch and \$

- \$inc