

```
(base) student@student-HP-ProDesk-2-Tower-G1a-Desktop-AI-PC:~$ mongosh
Current Mongosh Log ID: 68e76540a2fc5318c4fa334f
Connecting to:
mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.7
Using MongoDB:      5.0.31
Using Mongosh:      2.5.7
mongosh 2.5.8 is available for download:
https://www.mongodb.com/try/download/shell
```

For mongosh info see: <https://www.mongodb.com/docs/mongodb-shell/>

```
-----
The server generated these startup warnings when booting
2025-10-09T09:45:43.873+05:30: Using the XFS filesystem is strongly
recommended with the WiredTiger storage engine. See
http://dochub.mongodb.org/core/prodnotes-filesystem
2025-10-09T09:45:44.089+05:30: Access control is not enabled for the
database. Read and write access to data and configuration is unrestricted
-----
```

Warning: Found ~/.mongorc.js, but not ~/.mongoshrc.js. ~/.mongorc.js will not be loaded.

You may want to copy or rename ~/.mongorc.js to ~/.mongoshrc.js.

```
test> db.workers.insertMany([
...   {
...     id: 201,
...     name: "Anna",
...     salary: 55000,
...     department: "HR",
...     age: 29,
...     city: "Denver"
...   },
...   {
...     id: 202,
...     name: "Brian",
...     salary: 62000,
...     department: "Engineering",
...     age: 34,
...     city: "Austin"
...   },
...   {
...     id: 203,
...     name: "Clara",
...     salary: 48000,
...     department: "Sales",
...     age: null,
...     city: "Miami"
...   },
...   {
...     id: 204,
...     name: "David",
...     salary: 70000,
...     department: "Marketing",
```

```

...     age: 31,
...     city: "Seattle"
...   },
...   {
...     id: 205,
...     name: "Eva",
...     salary: 53000,
...     department: "Sales",
...     age: null,
...     city: "Boston"
...   }
... ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('68e765b8a2fc5318c4fa3350'),
    '1': ObjectId('68e765b8a2fc5318c4fa3351'),
    '2': ObjectId('68e765b8a2fc5318c4fa3352'),
    '3': ObjectId('68e765b8a2fc5318c4fa3353'),
    '4': ObjectId('68e765b8a2fc5318c4fa3354')
  }
}
test> db.workers.aggregate([
...   {
...     $group: {
...       _id: "$department",
...       totalSalary: { $sum: "$salary" }
...     }
...   },
...   {
...     $sort: { totalSalary: -1 }
...   }
... ])
[
  { _id: 'Sales', totalSalary: 101000 },
  { _id: 'Marketing', totalSalary: 70000 },
  { _id: 'Engineering', totalSalary: 62000 },
  { _id: 'HR', totalSalary: 55000 }
]
test> db.workers.aggregate([
...   {
...     $group: {
...       _id: "$department",
...       averageSalary: { $avg: "$salary" }
...     }
...   }
... ])
[
  { _id: 'Marketing', averageSalary: 70000 },
  { _id: 'HR', averageSalary: 55000 },
  { _id: 'Engineering', averageSalary: 62000 },
  { _id: 'Sales', averageSalary: 50500 }
]
test>

```

```

... db.workers.aggregate([
...   {
...     $group: {
...       _id: "$department",
...       maxSalary: { $max: "$salary" }
...     }
...   }
... ])
[
  { _id: 'Marketing', maxSalary: 70000 },
  { _id: 'HR', maxSalary: 55000 },
  { _id: 'Engineering', maxSalary: 62000 },
  { _id: 'Sales', maxSalary: 53000 }
]
test> db.workers.aggregate([
...   {
...     $group: {
...       _id: null,
...       minSalary: { $min: "$salary" }
...     }
...   }
... ])
[ { _id: null, minSalary: 48000 } ]
test> db.workers.aggregate([
...   { $count: "totalWorkers" }
... ])
[ { totalWorkers: 5 } ]
test> db.workers.aggregate([
...   {
...     $group: {
...       _id: "$department",
...       averageSalary: { $avg: "$salary" }
...     }
...   }
... ])
[
  { _id: 'Marketing', averageSalary: 70000 },
  { _id: 'HR', averageSalary: 55000 },
  { _id: 'Engineering', averageSalary: 62000 },
  { _id: 'Sales', averageSalary: 50500 }
]
test> db.workers.find().sort({ age: 1 })
[
  {
    _id: ObjectId('68e765b8a2fc5318c4fa3352'),
    id: 203,
    name: 'Clara',
    salary: 48000,
    department: 'Sales',
    age: null,
    city: 'Miami'
  },
  {
    _id: ObjectId('68e765b8a2fc5318c4fa3354'),
    id: 205,

```

```

    name: 'Eva',
    salary: 53000,
    department: 'Sales',
    age: null,
    city: 'Boston'
  },
  {
    _id: ObjectId('68e765b8a2fc5318c4fa3350'),
    id: 201,
    name: 'Anna',
    salary: 55000,
    department: 'HR',
    age: 29,
    city: 'Denver'
  },
  {
    _id: ObjectId('68e765b8a2fc5318c4fa3353'),
    id: 204,
    name: 'David',
    salary: 70000,
    department: 'Marketing',
    age: 31,
    city: 'Seattle'
  },
  {
    _id: ObjectId('68e765b8a2fc5318c4fa3351'),
    id: 202,
    name: 'Brian',
    salary: 62000,
    department: 'Engineering',
    age: 34,
    city: 'Austin'
  }
]
test> db.products.createIndex({ category: 1 })
category_1
test> db.products.getIndexes();
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { category: 1 }, name: 'category_1' }
]
test> db.products.createIndex({ category: 1, price: -1 })
category_1_price_-1
test> db.products.getIndexes();
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { category: 1 }, name: 'category_1' },
  {
    v: 2,
    key: { category: 1, price: -1 },
    name: 'category_1_price_-1'
  }
]
test> db.products.dropIndexes("category_1")

```

```
{ nIndexesWas: 3, ok: 1 }
test> db.products.dropIndexes()
{
  nIndexesWas: 2,
  msg: 'non-_id indexes dropped for collection',
  ok: 1
}
test>
```