

# Operators Used in MongoDB:-

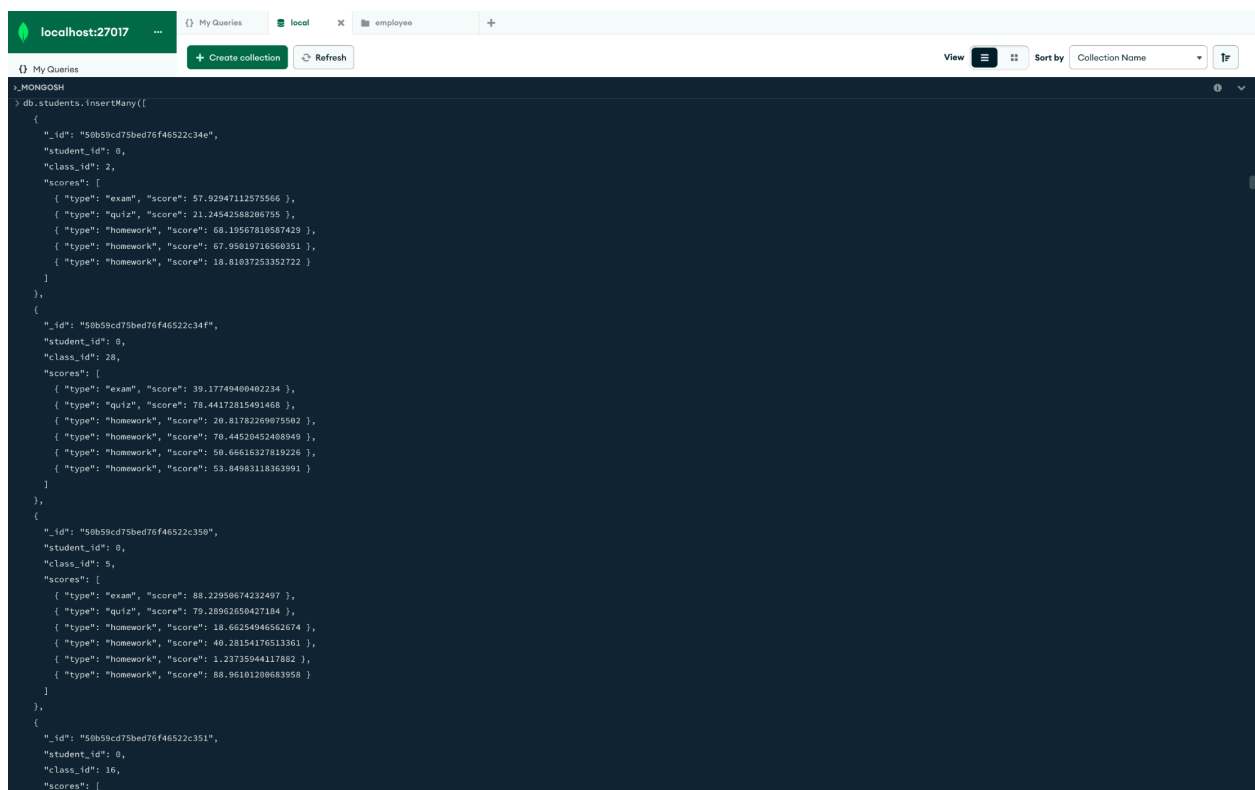
- Comparison Operators:
  - \$eq: Matches values that are equal to a specified value.
  - \$gt: Matches values that are greater than a specified value.
  - \$gte: Matches values that are greater than or equal to a specified value.
  - \$lt: Matches values that are less than a specified value.
  - \$lte: Matches values that are less than or equal to a specified value.
  - \$ne: Matches all values that are not equal to a specified value.
  - \$in: Matches any of the values specified in an array.
  - \$nin: Matches none of the values specified in an array.
- Logical Operators:
  - \$and: Joins query clauses with a logical AND and returns all documents that match the conditions of both clauses.
  - \$or: Joins query clauses with a logical OR and returns all documents that match the conditions of either clause.
  - \$not: Inverts the effect of a query expression and returns documents that do not match the query expression.
  - \$nor: Joins query clauses with a logical NOR and returns all documents that fail to match both clauses.
- Element Operators:
  - \$exists: Matches documents that contain the specified field.
  - \$type: Matches documents where the value of a field is of the specified type.
- Evaluation Operators:
  - \$expr: Allows the use of aggregation expressions within the query language.
  - \$jsonSchema: Validate documents against the given JSON Schema.
  - \$regex: Provides regular expression capabilities for pattern matching strings.
- Array Operators:
  - \$all: Matches arrays that contain all elements specified in the query.
  - \$elemMatch: Matches documents that contain an array field with at least one element that matches all the specified query criteria.
  - \$size: Matches arrays with a specific number of elements.
- Update Operators:
  - \$set: Sets the value of a field in a document.
  - \$unset: Removes the specified field from a document.
  - \$inc: Increments the value of the field by the specified amount.
  - \$push: Adds an item to an array.
  - \$addToSet: Adds elements to an array only if they do not already exist in the set.
  - \$pop: Removes the first or last element of an array.
  - \$pull: Removes all instances of a value from an array.
  - \$pullAll: Removes all instances of the specified values from an array.

## Use Case:-

1.Created a database school with collection students.

```
> use school  
< switched to db school  
> db.createCollection("students");  
< { ok: 1 }
```

2.Insert many to insert multiple from given github database of students.

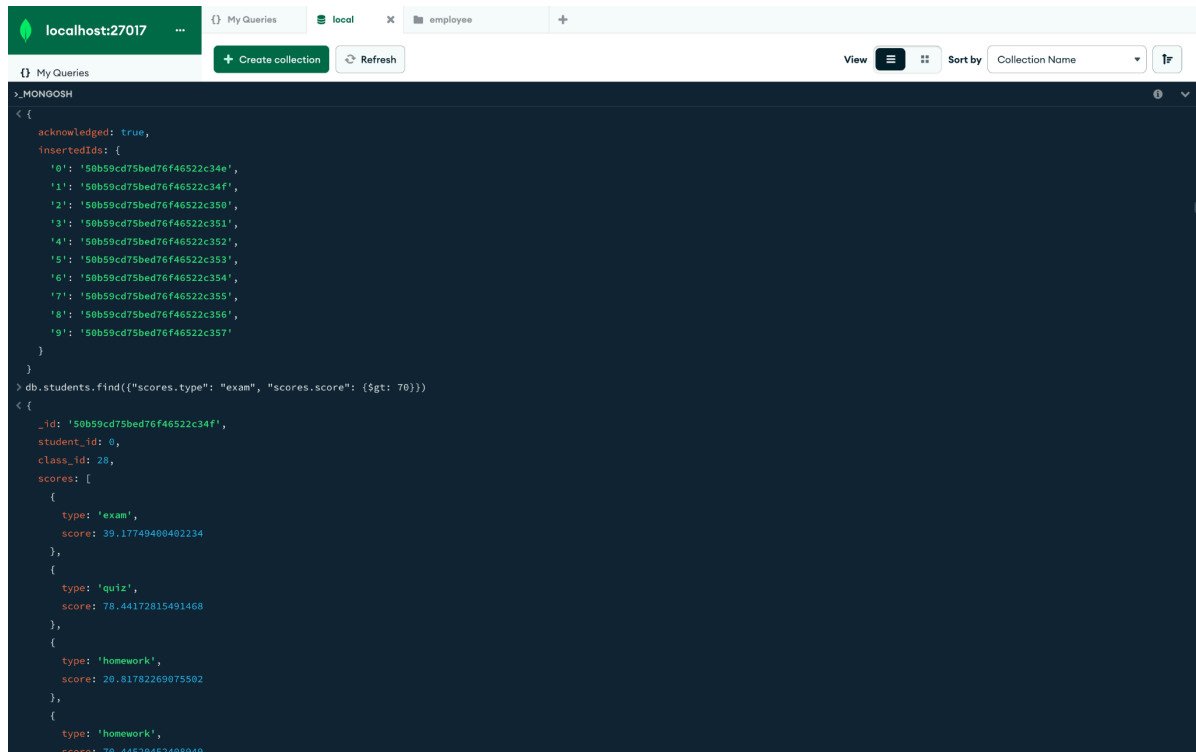


The screenshot shows the MongoDB web interface at localhost:27017. The 'My Queries' tab is active, displaying a successful `insertMany` operation on the `db.students` collection. The interface includes a top navigation bar with 'localhost:27017', 'My Queries', and tabs for 'local' and 'employee'. Below the navigation bar, there are buttons for 'Create collection' and 'Refresh'. The main area shows the command `use school` followed by `db.students.insertMany()` and a large JSON array of student records. The records are grouped into three distinct blocks, each containing a single student object with fields for `_id`, `student_id`, `class_id`, and `scores`. The `scores` field is an array of objects with `type` and `score` properties.

```
use school  
db.students.insertMany([  
  {  
    "_id": "50b59cd75bed76f46522c34e",  
    "student_id": 0,  
    "class_id": 2,  
    "scores": [  
      { "type": "exam", "score": 57.92947112575566 },  
      { "type": "quiz", "score": 21.24542588206756 },  
      { "type": "homework", "score": 68.19567810587429 },  
      { "type": "homework", "score": 67.95819716560351 },  
      { "type": "homework", "score": 18.6193725332732 }  
    ]  
  },  
  {  
    "_id": "50b59cd75bed76f46522c34f",  
    "student_id": 0,  
    "class_id": 28,  
    "scores": [  
      { "type": "exam", "score": 39.17749408402234 },  
      { "type": "quiz", "score": 78.44172815491468 },  
      { "type": "homework", "score": 20.81782269875582 },  
      { "type": "homework", "score": 70.44520452408949 },  
      { "type": "homework", "score": 50.66616327819226 },  
      { "type": "homework", "score": 53.84983118363991 }  
    ]  
  },  
  {  
    "_id": "50b59cd75bed76f46522c350",  
    "student_id": 0,  
    "class_id": 5,  
    "scores": [  
      { "type": "exam", "score": 88.22958674232497 },  
      { "type": "quiz", "score": 79.28962650427184 },  
      { "type": "homework", "score": 18.6625494652674 },  
      { "type": "homework", "score": 40.28154176513361 },  
      { "type": "homework", "score": 1.23735944117882 },  
      { "type": "homework", "score": 88.9610120863958 }  
    ]  
  },  
  {  
    "_id": "50b59cd75bed76f46522c351",  
    "student_id": 0,  
    "class_id": 16,  
    "scores": [  

```

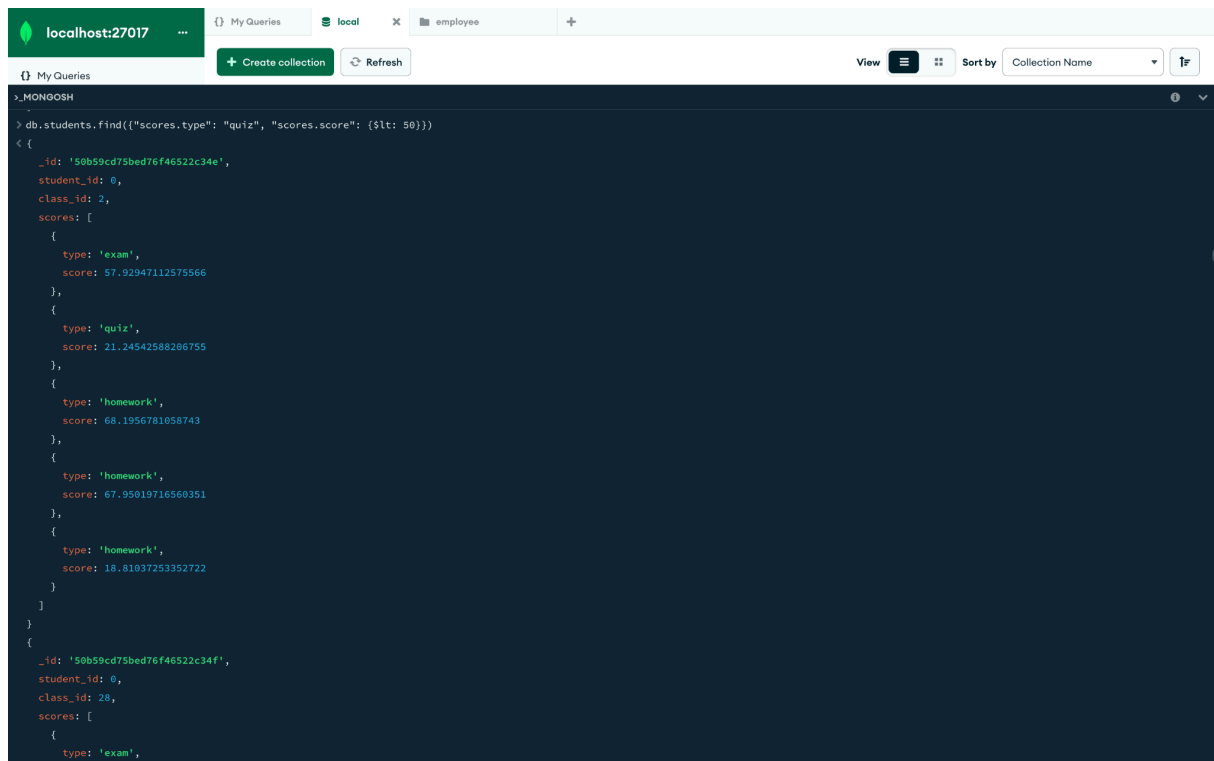
3.\$gt: Matches values that are greater than a specified value.



The screenshot shows the MongoDB Compass interface with a query filter `"scores.score": { $gt: 70 }` applied to the `students` collection. The result set contains one document with an exam score of 39.17749400402234, which is less than 70, indicating that the query did not match any documents.

```
> _MONGODB
< {
  acknowledged: true,
  insertedIds: {
    '0': '50b59cd75bed76f46522c34e',
    '1': '50b59cd75bed76f46522c34f',
    '2': '50b59cd75bed76f46522c350',
    '3': '50b59cd75bed76f46522c351',
    '4': '50b59cd75bed76f46522c352',
    '5': '50b59cd75bed76f46522c353',
    '6': '50b59cd75bed76f46522c354',
    '7': '50b59cd75bed76f46522c355',
    '8': '50b59cd75bed76f46522c356',
    '9': '50b59cd75bed76f46522c357'
  }
}
> db.students.find({"scores.type": "exam", "scores.score": { $gt: 70 }})
< {
  _id: '50b59cd75bed76f46522c34f',
  student_id: 0,
  class_id: 28,
  scores: [
    {
      type: 'exam',
      score: 39.17749400402234
    },
    {
      type: 'quiz',
      score: 78.44172815491468
    },
    {
      type: 'homework',
      score: 20.81782269075502
    },
    {
      type: 'homework',
      score: 76.44578452408949
    }
  ]
}
```

4.\$lt: Matches values that are less than a specified value.

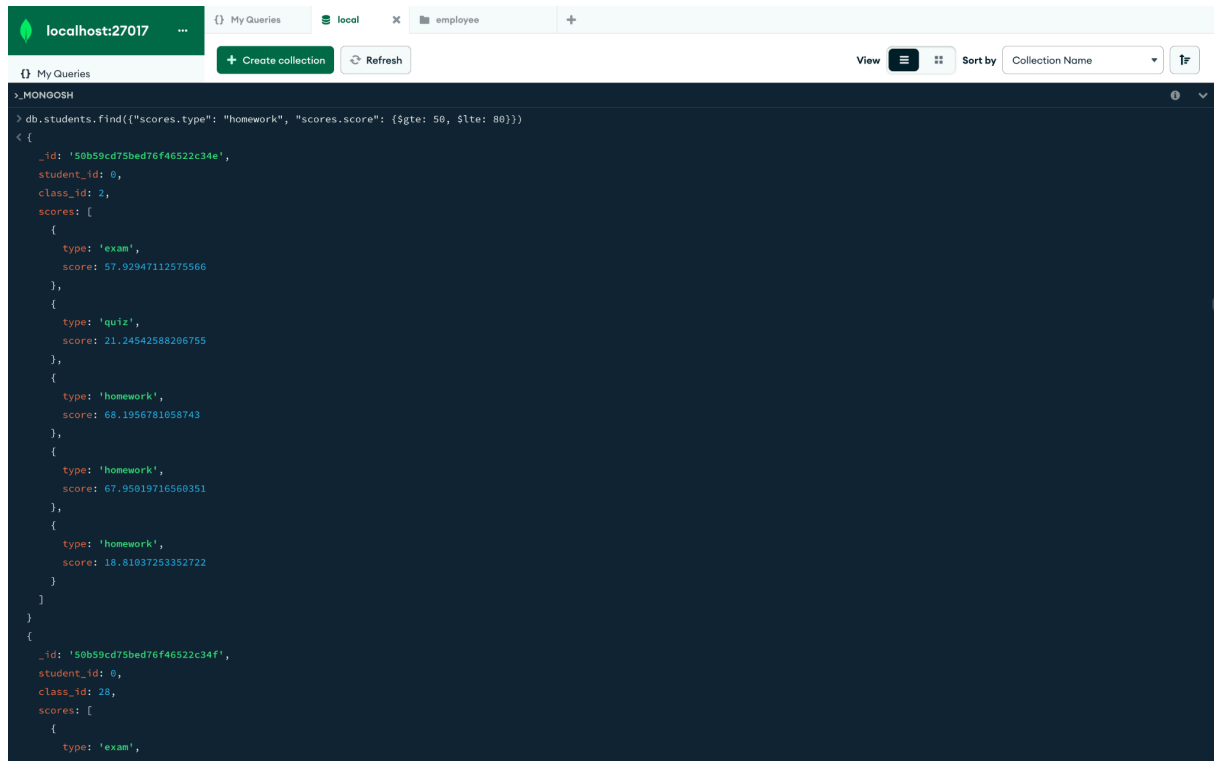


The screenshot shows the MongoDB Compass interface with a query filter `"scores.score": { $lt: 50 }` applied to the `students` collection. The result set contains two documents: one with a quiz score of 21.24542588206755 and another with a homework score of 18.81037253352722, both of which are less than 50.

```
> db.students.find({"scores.type": "quiz", "scores.score": { $lt: 50 }})
< {
  _id: '50b59cd75bed76f46522c34e',
  student_id: 0,
  class_id: 2,
  scores: [
    {
      type: 'exam',
      score: 57.92947112575566
    },
    {
      type: 'quiz',
      score: 21.24542588206755
    },
    {
      type: 'homework',
      score: 68.1956781058743
    },
    {
      type: 'homework',
      score: 67.95019716500351
    },
    {
      type: 'homework',
      score: 18.81037253352722
    }
  ]
}
{
  _id: '50b59cd75bed76f46522c34f',
  student_id: 0,
  class_id: 28,
  scores: [
    {
      type: 'exam',

```

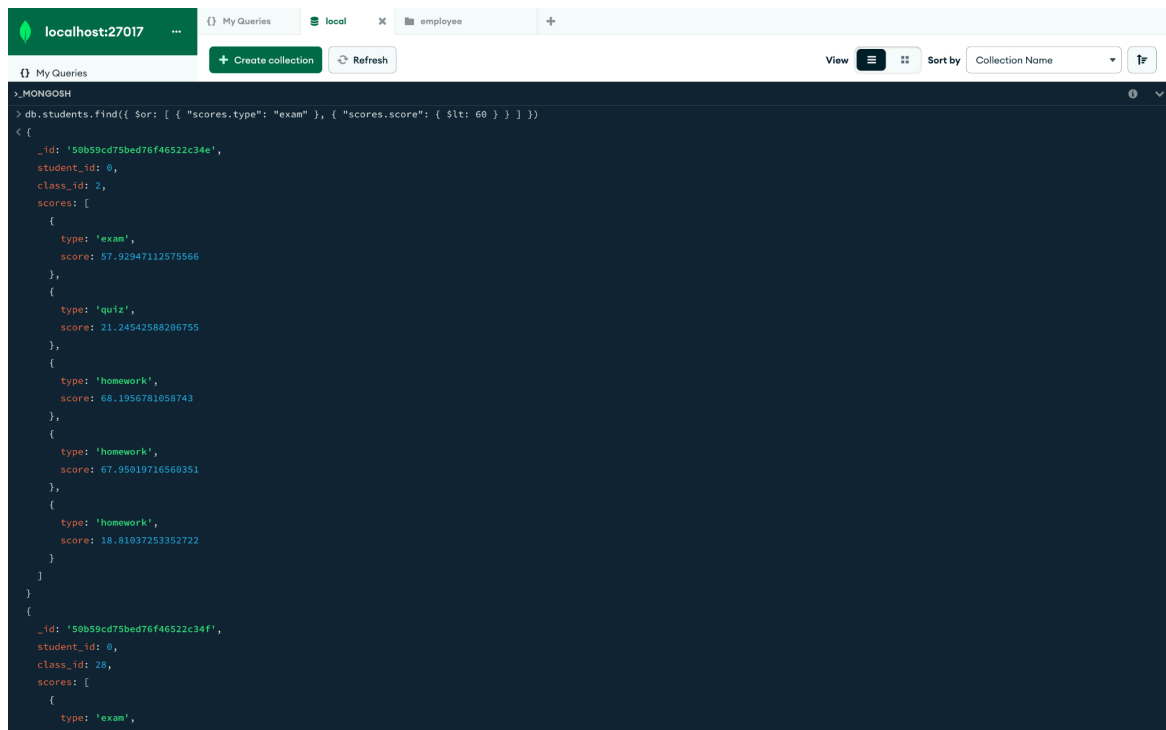
5.\$gte: Matches values that are greater than or equal to a specified value, \$lte: Matches values that are less than or equal to a specified value.



The screenshot shows the MongoDB Compass interface. The top bar indicates the connection to 'localhost:27017'. The left sidebar shows the 'My Queries' tab. The main area displays a query: `db.students.find({"scores.type": "homework", "scores.score": {$gte: 50, $lte: 80}})`. The result shows two documents. The first document has a score of 68.1956781058743 for a homework type, which matches the query criteria. The second document has a score of 18.81037253352722 for a homework type, which also matches the query criteria.

```
> db.students.find({"scores.type": "homework", "scores.score": {$gte: 50, $lte: 80}})
< [
  {
    _id: '50b59cd75bed76f46522c34e',
    student_id: 0,
    class_id: 2,
    scores: [
      {
        type: 'exam',
        score: 57.92947112575566
      },
      {
        type: 'quiz',
        score: 21.24542588206755
      },
      {
        type: 'homework',
        score: 68.1956781058743
      },
      {
        type: 'homework',
        score: 67.95019716560351
      },
      {
        type: 'homework',
        score: 18.81037253352722
      }
    ]
  },
  {
    _id: '50b59cd75bed76f46522c34f',
    student_id: 0,
    class_id: 28,
    scores: [
      {
        type: 'exam',
        score: 20.17240400402734
      }
    ]
  }
]
```

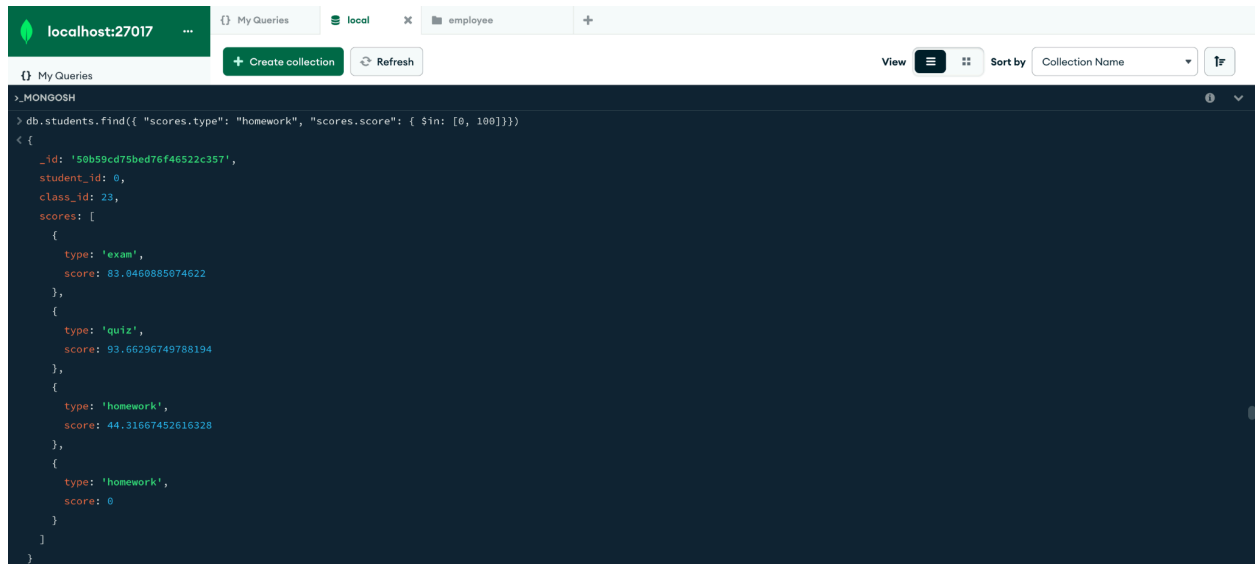
6.\$or: Joins query clauses with a logical OR and returns all documents that match the conditions of either clause.



The screenshot shows the MongoDB Compass interface. The top bar indicates the connection to 'localhost:27017'. The left sidebar shows the 'My Queries' tab. The main area displays a query: `db.students.find({ $or: [ { "scores.type": "exam" }, { "scores.score": { $lt: 60 } } ] })`. The result shows two documents. The first document has a score of 68.1956781058743 for a homework type, which matches the query criteria. The second document has a score of 18.81037253352722 for a homework type, which also matches the query criteria.

```
> db.students.find({ $or: [ { "scores.type": "exam" }, { "scores.score": { $lt: 60 } } ] })
< [
  {
    _id: '50b59cd75bed76f46522c34e',
    student_id: 0,
    class_id: 2,
    scores: [
      {
        type: 'exam',
        score: 57.92947112575566
      },
      {
        type: 'quiz',
        score: 21.24542588206755
      },
      {
        type: 'homework',
        score: 68.1956781058743
      },
      {
        type: 'homework',
        score: 67.95019716560351
      },
      {
        type: 'homework',
        score: 18.81037253352722
      }
    ]
  },
  {
    _id: '50b59cd75bed76f46522c34f',
    student_id: 0,
    class_id: 28,
    scores: [
      {
        type: 'exam',
        score: 20.17240400402734
      }
    ]
  }
]
```

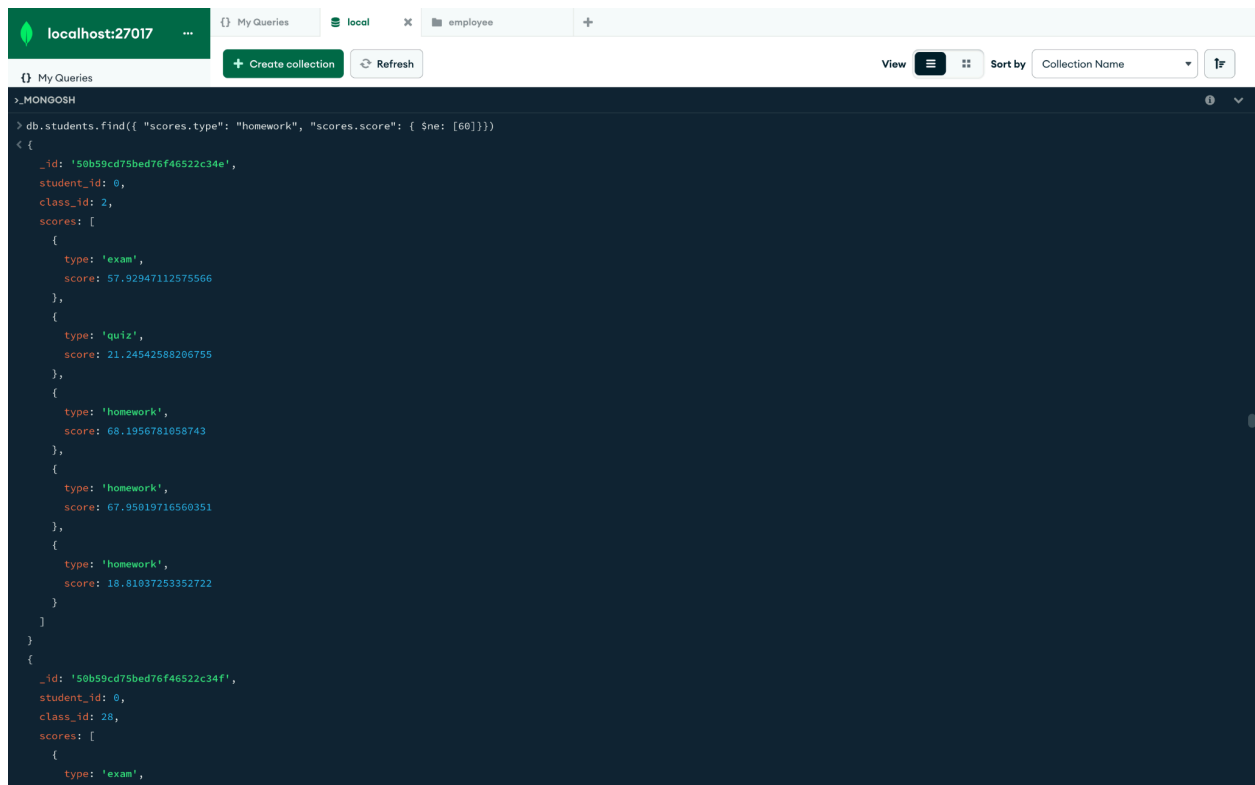
7.\$in: Matches any of the values specified in an array.



The screenshot shows the MongoDB Compass interface. The top bar indicates the connection to 'localhost:27017'. Below the top bar, there are tabs for 'My Queries', 'local', and 'employee'. The 'employee' tab is selected. The main area displays a query: `db.students.find({ "scores.type": "homework", "scores.score": { $in: [0, 100]}})`. The results are shown in a list format, displaying the following document:

```
{
  "_id": "50b59cd75bed76f46522c357",
  "student_id": 0,
  "class_id": 23,
  "scores": [
    {
      "type": "exam",
      "score": 83.0460885074622
    },
    {
      "type": "quiz",
      "score": 93.66296749788194
    },
    {
      "type": "homework",
      "score": 44.31667452616328
    },
    {
      "type": "homework",
      "score": 0
    }
  ]
}
```

8.\$ne: Matches all values that are not equal to a specified value.

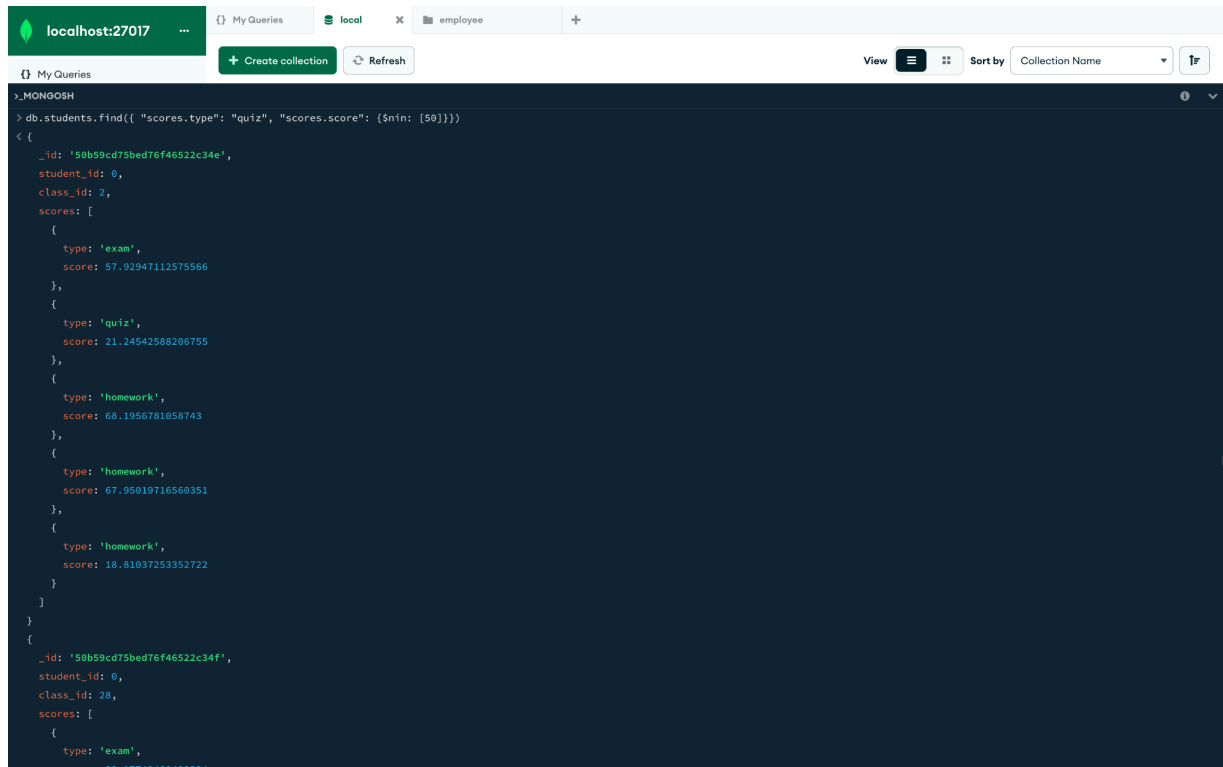


The screenshot shows the MongoDB Compass interface. The top bar indicates the connection to 'localhost:27017'. Below the top bar, there are tabs for 'My Queries', 'local', and 'employee'. The 'employee' tab is selected. The main area displays a query: `db.students.find({ "scores.type": "homework", "scores.score": { $ne: [60]}})`. The results are shown in a list format, displaying the following documents:

```
{
  "_id": "50b59cd75bed76f46522c34e",
  "student_id": 0,
  "class_id": 2,
  "scores": [
    {
      "type": "exam",
      "score": 57.92947112575566
    },
    {
      "type": "quiz",
      "score": 21.24542588206755
    },
    {
      "type": "homework",
      "score": 68.1956781058743
    },
    {
      "type": "homework",
      "score": 67.95019716560351
    },
    {
      "type": "homework",
      "score": 18.8103725352722
    }
  ]
},
{
  "_id": "50b59cd75bed76f46522c34f",
  "student_id": 0,
  "class_id": 28,
  "scores": [
    {
      "type": "exam",

```

9.\$nin: Matches none of the values specified in an array.



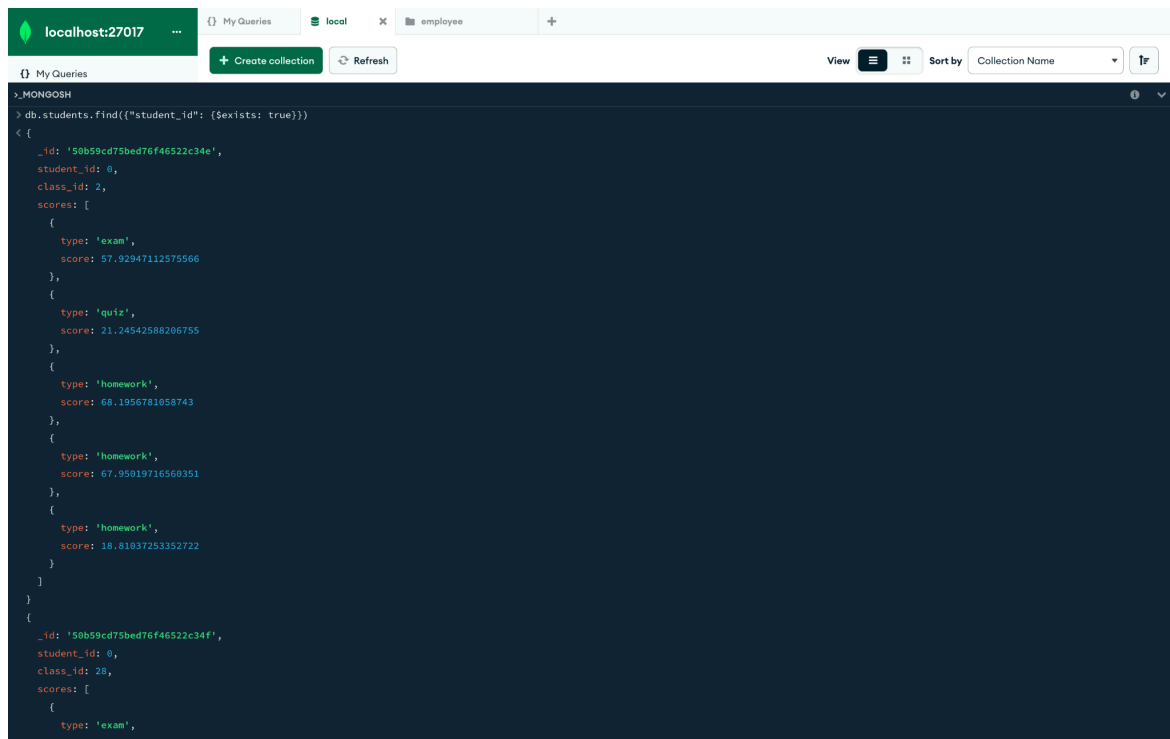
The screenshot shows the MongoDB Compass interface. The top bar indicates the connection to 'localhost:27017' and the 'employee' collection. The query editor shows the following command:

```
> db.students.find({"scores.type": "quiz", "scores.score": {$nin: [50]}})
```

The results pane displays two documents. The first document has a 'quiz' score of 21.24542588206755 and three 'homework' scores. The second document has an 'exam' score of 38.17749400402234 and no other scores.

```
{
  "_id": "50b59cd75bed76f46522c34e",
  "student_id": 0,
  "class_id": 2,
  "scores": [
    {
      "type": "exam",
      "score": 57.92947112575566
    },
    {
      "type": "quiz",
      "score": 21.24542588206755
    },
    {
      "type": "homework",
      "score": 68.1956781058743
    },
    {
      "type": "homework",
      "score": 67.95019716560351
    },
    {
      "type": "homework",
      "score": 18.81037253352722
    }
  ]
},
{
  "_id": "50b59cd75bed76f46522c34f",
  "student_id": 0,
  "class_id": 28,
  "scores": [
    {
      "type": "exam",
      "score": 38.17749400402234
    }
  ]
}
```

10.\$exists: Matches documents that contain the specified field.



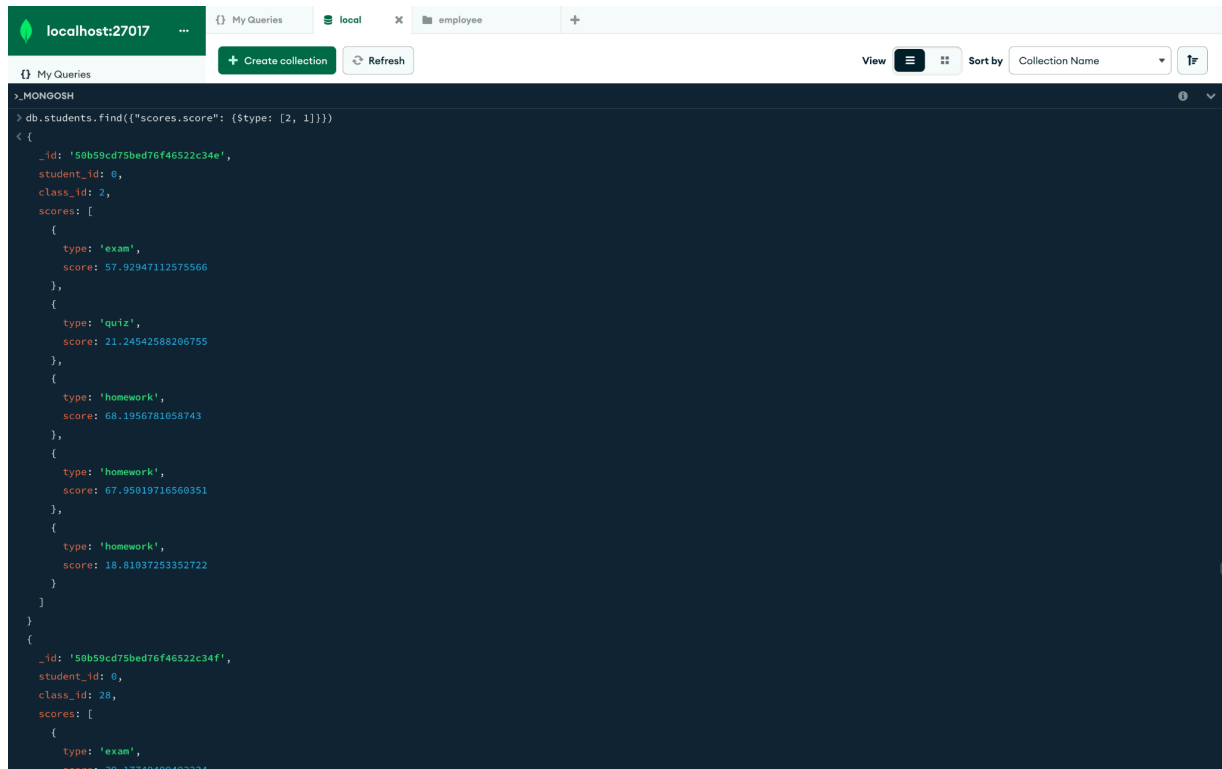
The screenshot shows the MongoDB Compass interface. The top bar indicates the connection to 'localhost:27017' and the 'employee' collection. The query editor shows the following command:

```
> db.students.find({"student_id": {$exists: true}})
```

The results pane displays two documents. The first document has a 'student\_id' of 0 and three 'homework' scores. The second document has a 'student\_id' of 0 and one 'exam' score.

```
{
  "_id": "50b59cd75bed76f46522c34e",
  "student_id": 0,
  "class_id": 2,
  "scores": [
    {
      "type": "exam",
      "score": 57.92947112575566
    },
    {
      "type": "quiz",
      "score": 21.24542588206755
    },
    {
      "type": "homework",
      "score": 68.1956781058743
    },
    {
      "type": "homework",
      "score": 67.95019716560351
    },
    {
      "type": "homework",
      "score": 18.81037253352722
    }
  ]
},
{
  "_id": "50b59cd75bed76f46522c34f",
  "student_id": 0,
  "class_id": 28,
  "scores": [
    {
      "type": "exam",
      "score": 38.17749400402234
    }
  ]
}
```

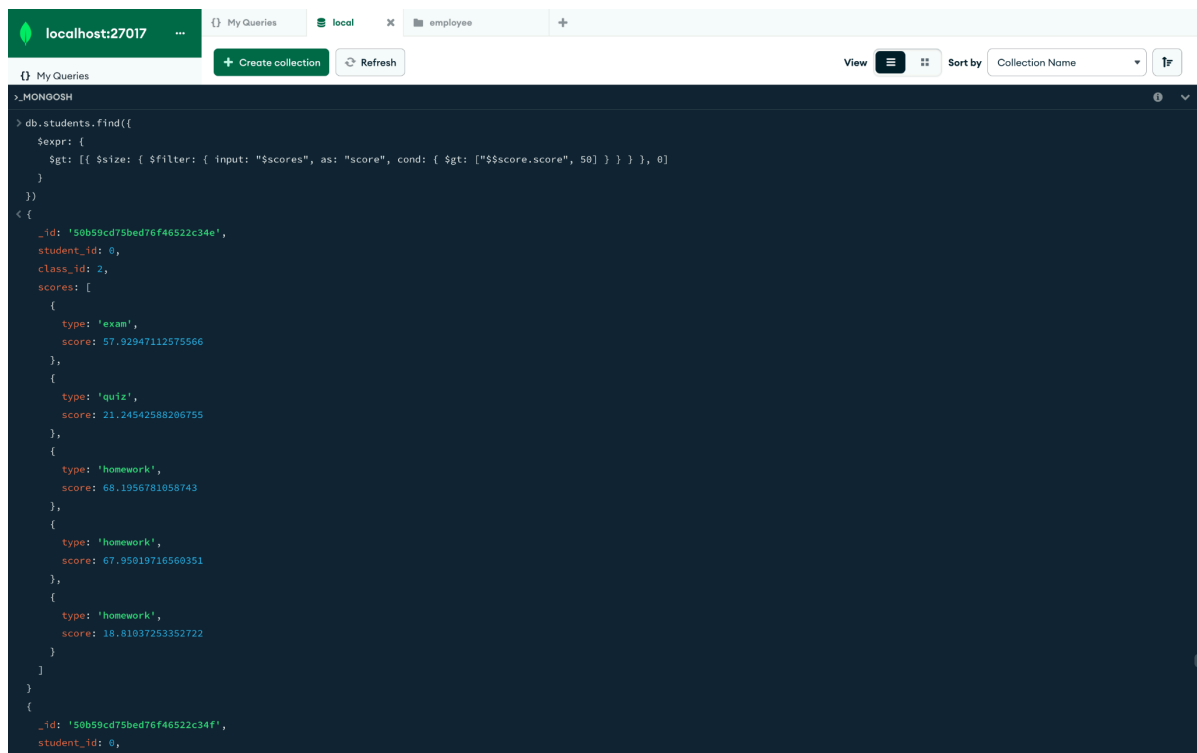
11.\$type: Matches documents where the value of a field is of the specified type.



The screenshot shows the MongoDB Compass interface with a query editor on the left and a results pane on the right. The query is `db.students.find({"scores.score": {$type: [2, 1]}})`. The results pane displays two documents from the 'students' collection. The first document has a score of 57.92947112575566 for an 'exam' type and 21.24542588206755 for a 'quiz' type. The second document has a score of 68.1956781058743 for a 'homework' type and 67.95019716500351 for a 'homework' type.

```
>_MONGODB
> db.students.find({"scores.score": {$type: [2, 1]}})
{
  "_id": "50b59cd75bed76f46522c34e",
  "student_id": 0,
  "class_id": 2,
  "scores": [
    {
      "type": "exam",
      "score": 57.92947112575566
    },
    {
      "type": "quiz",
      "score": 21.24542588206755
    },
    {
      "type": "homework",
      "score": 68.1956781058743
    },
    {
      "type": "homework",
      "score": 67.95019716500351
    },
    {
      "type": "homework",
      "score": 18.81037253352722
    }
  ]
}
{
  "_id": "50b59cd75bed76f46522c34f",
  "student_id": 0,
  "class_id": 28,
  "scores": [
    {
      "type": "exam",
      "score": 30.127200000000000
    }
  ]
}
```

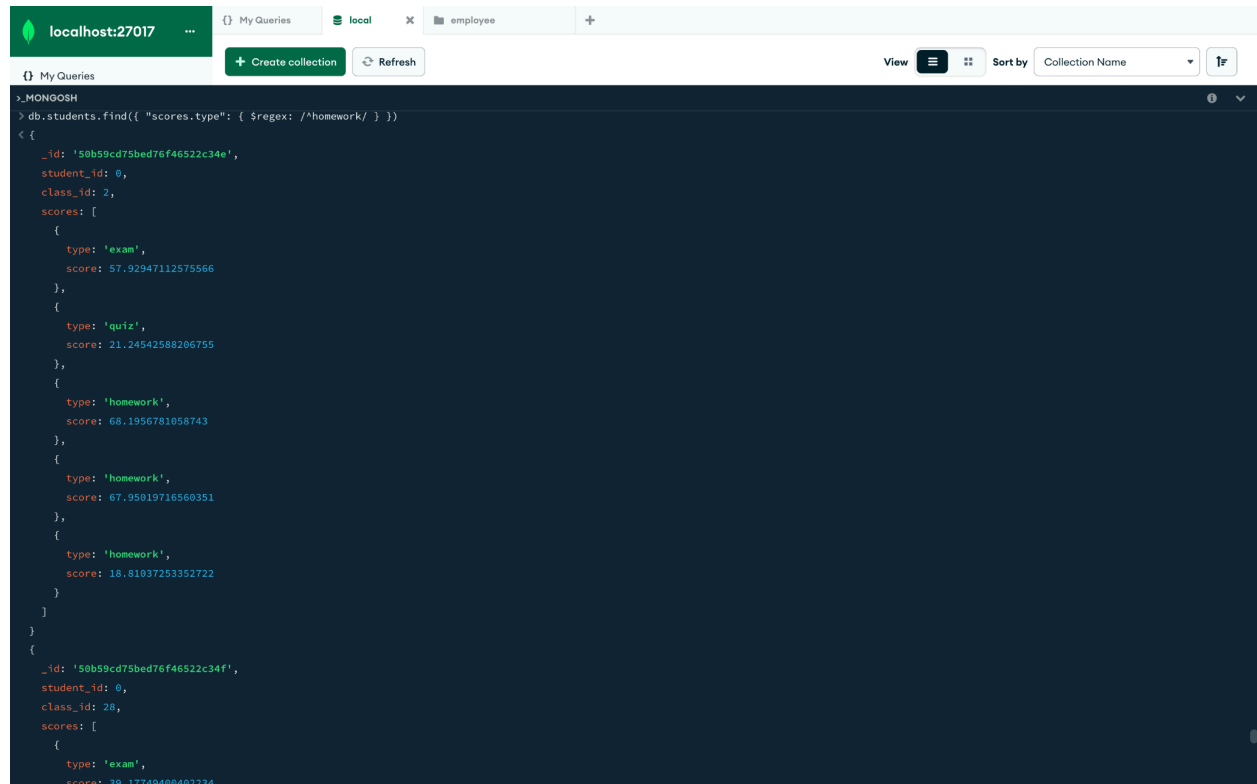
12.\$expr: Allows the use of aggregation expressions within the query language, \$size: Matches arrays with a specific number of elements, \$filter to filter out particular parameters.



The screenshot shows the MongoDB Compass interface with a query editor on the left and a results pane on the right. The query is `db.students.find({$expr: {$gt: [{ $size: { $filter: { input: "$scores", as: "score", cond: { $gt: ["$$score.score", 50] } } } ], 0}}})`. The results pane displays two documents from the 'students' collection. The first document has a score of 57.92947112575566 for an 'exam' type and 21.24542588206755 for a 'quiz' type. The second document has a score of 68.1956781058743 for a 'homework' type and 67.95019716500351 for a 'homework' type.

```
>_MONGODB
> db.students.find({$expr: {$gt: [{ $size: { $filter: { input: "$scores", as: "score", cond: { $gt: ["$$score.score", 50] } } } ], 0}}})
{
  "_id": "50b59cd75bed76f46522c34e",
  "student_id": 0,
  "class_id": 2,
  "scores": [
    {
      "type": "exam",
      "score": 57.92947112575566
    },
    {
      "type": "quiz",
      "score": 21.24542588206755
    },
    {
      "type": "homework",
      "score": 68.1956781058743
    },
    {
      "type": "homework",
      "score": 67.95019716500351
    },
    {
      "type": "homework",
      "score": 18.81037253352722
    }
  ]
}
{
  "_id": "50b59cd75bed76f46522c34f",
  "student_id": 0,
  "class_id": 28,
  "scores": [
    {
      "type": "exam",
      "score": 30.127200000000000
    }
  ]
}
```

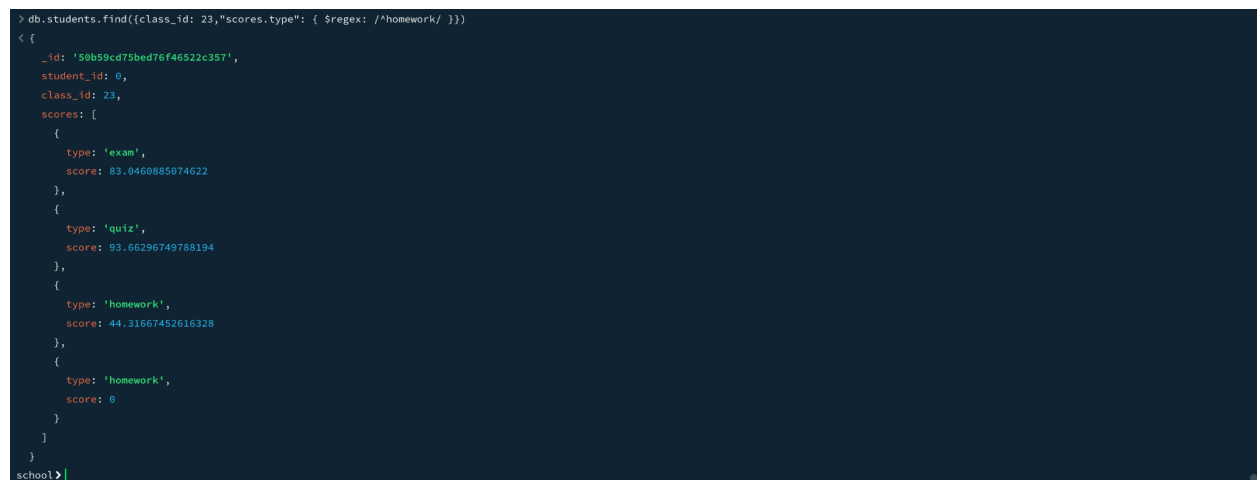
13.\$regex: Provides regular expression capabilities for pattern matching strings.



The screenshot shows the MongoDB Compass interface. The top bar indicates the connection is to 'localhost:27017'. The main area displays a query: `db.students.find({ "scores.type": { $regex: /^homework/ } })`. The results are shown in a JSON format, listing two documents. Each document contains an ID, student ID, class ID, and an array of scores with types and scores.

```
> db.students.find({ "scores.type": { $regex: /^homework/ } })
< [
  {
    _id: '50b59cd75bed76f46522c34e',
    student_id: 0,
    class_id: 2,
    scores: [
      {
        type: 'exam',
        score: 57.92947112575566
      },
      {
        type: 'quiz',
        score: 21.24542588206755
      },
      {
        type: 'homework',
        score: 68.1956781058743
      },
      {
        type: 'homework',
        score: 67.95019716560351
      },
      {
        type: 'homework',
        score: 18.8103725352722
      }
    ]
  },
  {
    _id: '50b59cd75bed76f46522c34f',
    student_id: 0,
    class_id: 28,
    scores: [
      {
        type: 'exam',
        score: 30.17749408402234
      }
    ]
  }
]
```

14.\$and: Joins query clauses with a logical AND and returns all documents that match the conditions of both clauses.



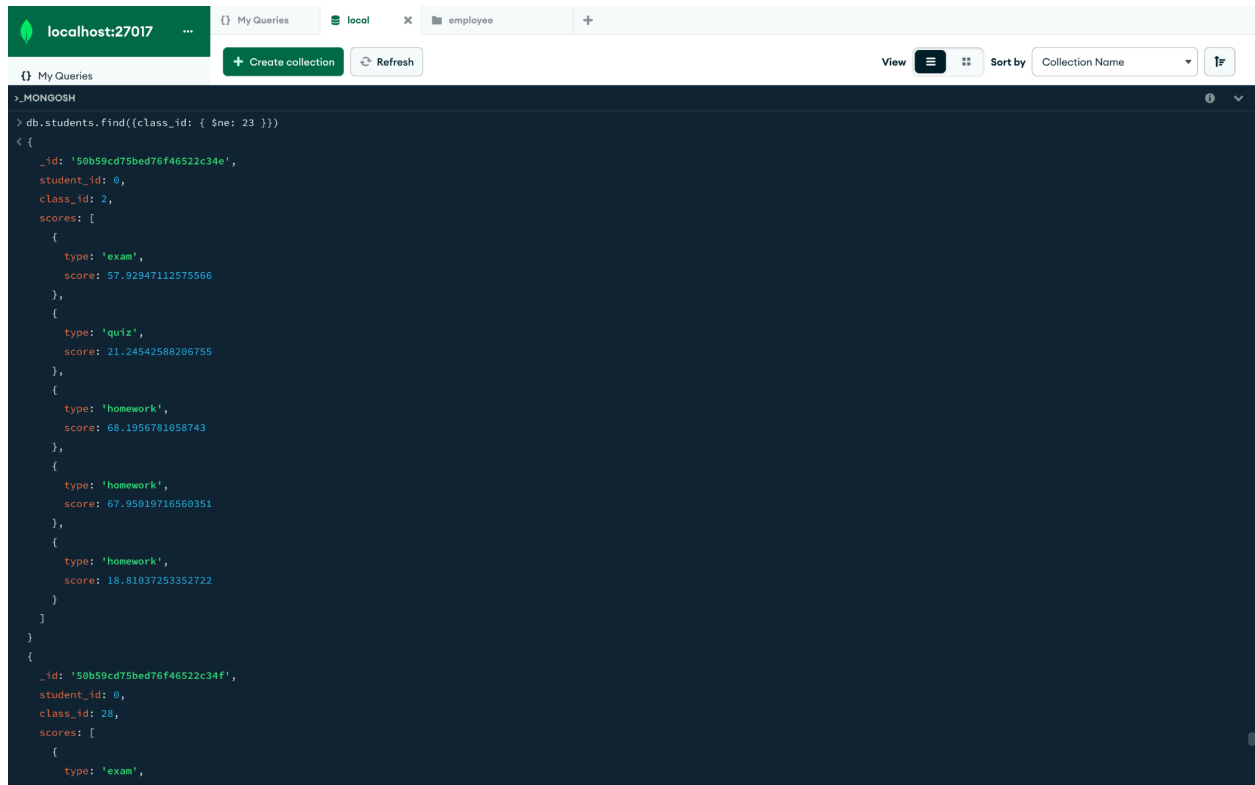
The screenshot shows the MongoDB Compass interface. The main area displays a query: `db.students.find({class_id: 23,"scores.type": { $regex: /^homework/ } })`. The results are shown in a JSON format, listing one document. The document contains an ID, student ID, class ID, and an array of scores with types and scores.

```
> db.students.find({class_id: 23,"scores.type": { $regex: /^homework/ } })
< [
  {
    _id: '50b59cd75bed76f46522c357',
    student_id: 0,
    class_id: 23,
    scores: [
      {
        type: 'exam',
        score: 83.0460885074622
      },
      {
        type: 'quiz',
        score: 93.66296749788194
      },
      {
        type: 'homework',
        score: 44.31667452616328
      },
      {
        type: 'homework',
        score: 0
      }
    ]
  }
]
```

schoal>



15.\$not: Inverts the effect of a query expression and returns documents that do not match the query expression.



The screenshot shows the MongoDB Compass web interface. At the top, there's a header with 'localhost:27017' and a 'My Queries' tab. Below the header, there are buttons for 'Create collection' and 'Refresh'. The main area displays a query result for the 'students' collection. The query is 'db.students.find({class\_id: { \$ne: 23 }})'. The result shows two documents. The first document has a student\_id of 0, a class\_id of 2, and scores for 'exam', 'quiz', and 'homework'. The second document has a student\_id of 0, a class\_id of 28, and scores for 'exam' and 'homework'.

```
> db.students.find({class_id: { $ne: 23 }})
{
  "_id": "50b59cd75bed76f46522c34e",
  "student_id": 0,
  "class_id": 2,
  "scores": [
    {
      "type": "exam",
      "score": 57.92947112575566
    },
    {
      "type": "quiz",
      "score": 21.24542588206755
    },
    {
      "type": "homework",
      "score": 68.1956781058743
    },
    {
      "type": "homework",
      "score": 67.95019716560351
    },
    {
      "type": "homework",
      "score": 18.8103725352722
    }
  ]
}
{
  "_id": "50b59cd75bed76f46522c34f",
  "student_id": 0,
  "class_id": 28,
  "scores": [
    {
      "type": "exam",
      "score": 20.12710100400001
    },
    {
      "type": "homework",
      "score": 18.8103725352722
    }
  ]
}
```