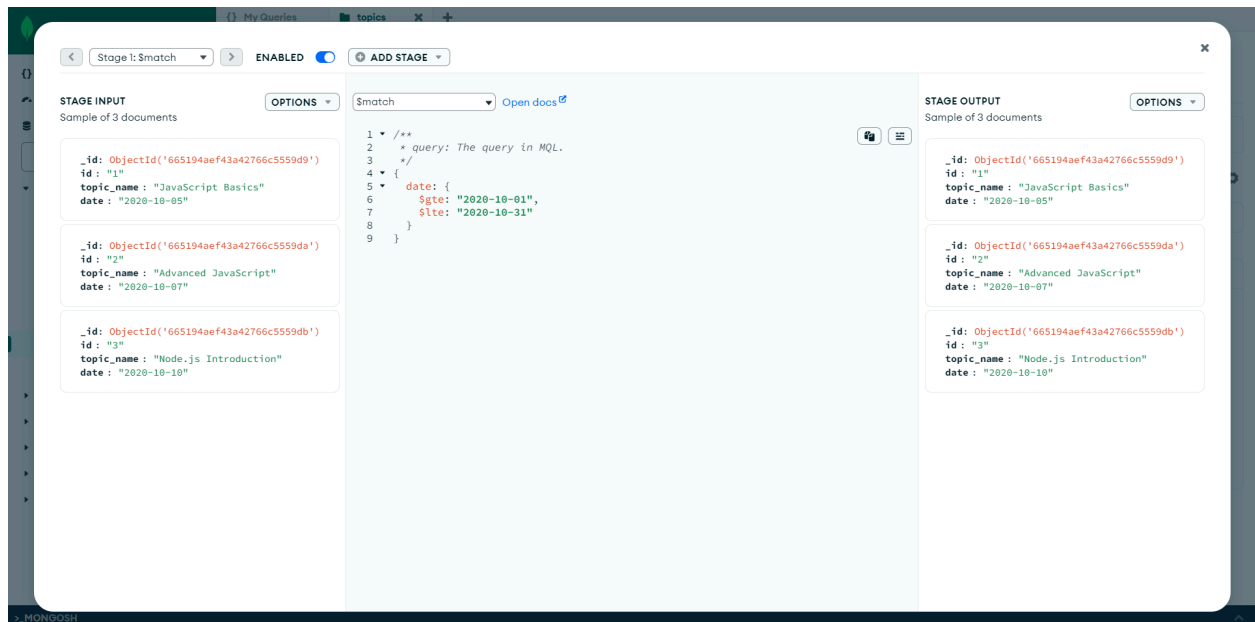


MongoDB Task - 2

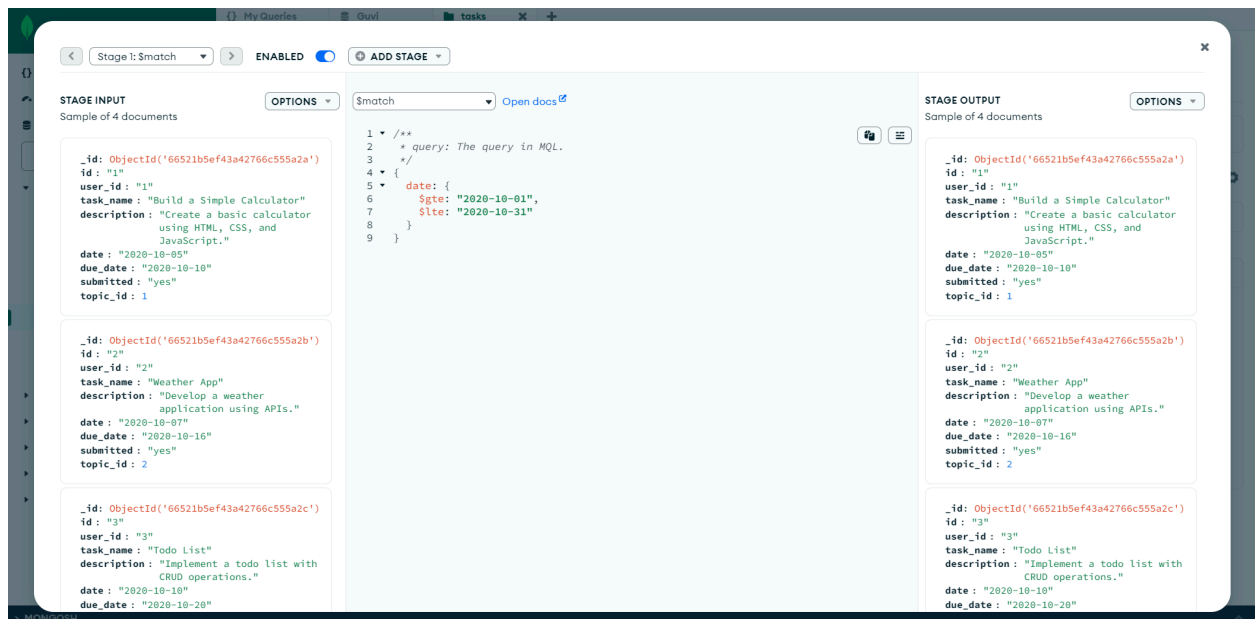
1. Find all the topics and tasks which are taught in the month of October

MongoDB Compass:

Topics were taught in the month of October:



Tasks were given in the month of October



Shell:

Topics were taught in the month of October:

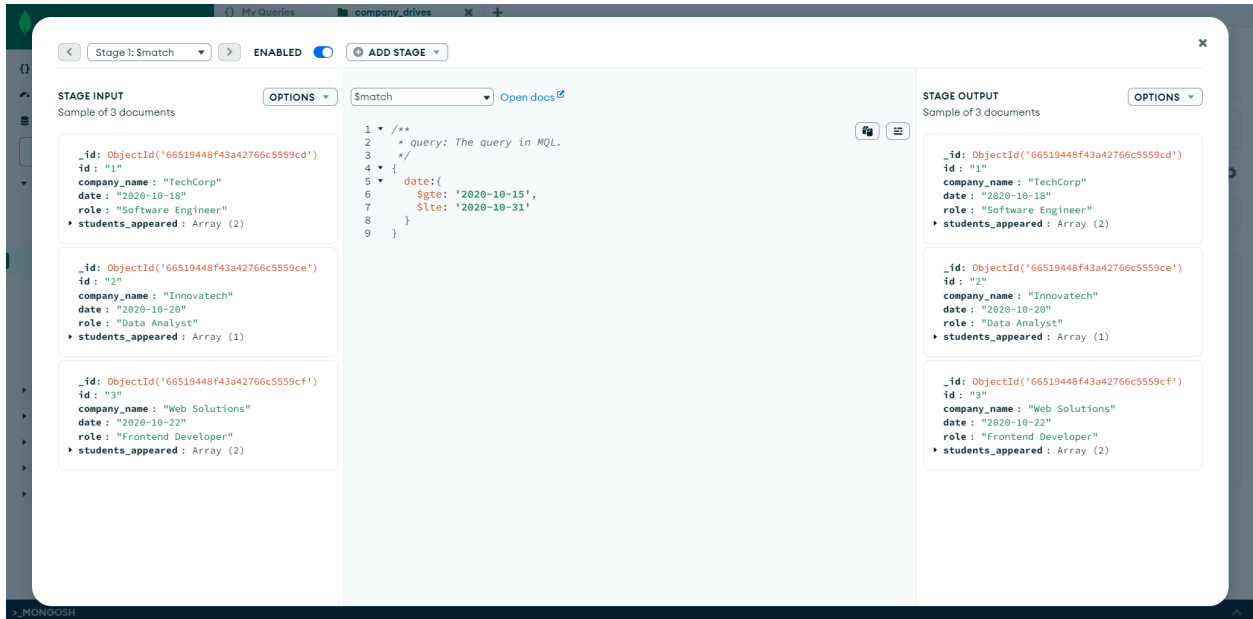
```
Guvi> db.topics.aggregate({$match:{date:{$gte:'2020-10-01', $lte:'2020-10-31'}}})
[
  {
    _id: ObjectId('665194aef43a42766c5559d9'),
    id: '1',
    topic_name: 'JavaScript Basics',
    date: '2020-10-05'
  },
  {
    _id: ObjectId('665194aef43a42766c5559da'),
    id: '2',
    topic_name: 'Advanced JavaScript',
    date: '2020-10-07'
  },
  {
    _id: ObjectId('665194aef43a42766c5559db'),
    id: '3',
    topic_name: 'Node.js Introduction',
    date: '2020-10-10'
  }
]
Guvi> |
```

Tasks were given in the month of October

```
mongosh mongodb://127.0.0.1:27000
Guvi> db.tasks.aggregate({$match:{date:{$gte:'2020-10-01', $lte:'2020-10-31'}}})
[
  {
    _id: ObjectId('66521b5ef43a42766c555a2a'),
    id: '1',
    user_id: '1',
    task_name: 'Build a Simple Calculator',
    description: 'Create a basic calculator using HTML, CSS, and JavaScript.',
    date: '2020-10-05',
    due_date: '2020-10-10',
    submitted: 'yes',
    topic_id: 1
  },
  {
    _id: ObjectId('66521b5ef43a42766c555a2b'),
    id: '2',
    user_id: '2',
    task_name: 'Weather App',
    description: 'Develop a weather application using APIs.',
    date: '2020-10-07',
    due_date: '2020-10-16',
    submitted: 'yes',
    topic_id: 2
  },
  {
    _id: ObjectId('66521b5ef43a42766c555a2c'),
    id: '3',
    user_id: '3',
    task_name: 'Todo List',
    description: 'Implement a todo list with CRUD operations.',
    date: '2020-10-10',
    due_date: '2020-10-20',
    submitted: 'no',
    topic_id: 3
  },
  {
    _id: ObjectId('66521b5ef43a42766c555a2d'),
    id: '4',
    user_id: '4',
    task_name: 'Todo List',
    description: 'Implement a todo list with CRUD operations.',
  }
]
```

2. Find all the company drives which appeared between 15 oct-2020 and 31-oct-2020

MongoDB Compass:



Shell:

```
Guvi> db.company_drives.aggregate({$match:{date:{$gte:'2020-10-15', $lte:'2020-10-31'}}})
[
  {
    "_id": ObjectId('66519448f43a42766c5559cd'),
    "id": "1",
    "company_name": "TechCorp",
    "date": "2020-10-18",
    "role": "Software Engineer",
    "students_appeared": [ 1, 2 ]
  },
  {
    "_id": ObjectId('66519448f43a42766c5559ce'),
    "id": "2",
    "company_name": "Innovatech",
    "date": "2020-10-20",
    "role": "Data Analyst",
    "students_appeared": [ 1 ]
  },
  {
    "_id": ObjectId('66519448f43a42766c5559cf'),
    "id": "3",
    "company_name": "Web Solutions",
    "date": "2020-10-22",
    "role": "Frontend Developer",
    "students_appeared": [ 1, 3 ]
  }
]
Guvi>
```

3. Find all the company drives and students who are appeared for the placement

MongoDB Compass:

\$lookup stage:

The screenshot shows the MongoDB Compass interface with a query pipeline. The stage is set to '\$lookup'. The 'STAGE INPUT' section shows three documents from the 'company_drives' collection. The 'STAGE OUTPUT' section shows the result of the \$lookup stage, where the 'students_appeared' field is populated with an array of student objects. The 'STAGE OUTPUT' section also shows the 'students' field as an array of objects.

```
1 /**
2  * from: The target collection.
3  * localField: The local join field.
4  * foreignField: The target join field.
5  * as: The name for the results.
6  * pipeline: Optional pipeline to run on the foreign collection.
7  * let: Optional variables to use in the pipeline field stages.
8  */
9 {
10   from: 'users',
11   localField: 'students_appeared',
12   foreignField: 'id',
13   as: 'students'
14 }
```

\$unwind stage:

The screenshot shows the MongoDB Compass interface with a query pipeline. The stage is set to '\$unwind'. The 'STAGE INPUT' section shows three documents from the 'company_drives' collection. The 'STAGE OUTPUT' section shows the result of the \$unwind stage, where the 'students_appeared' field is unwound into individual student objects. The 'STAGE OUTPUT' section also shows the 'students' field as an array of objects.

```
1 /**
2  * path: Path to the array field.
3  * includeArrayIndex: Optional name for index.
4  * preserveNullAndEmptyArrays: Optional
5  *   toggle to unwind null and empty values.
6  */
7 {
8   path: '$students'
9 }
```

\$project stage:

Stage 3: \$project

ENABLED

ADD STAGE

STAGE INPUT

Sample of 5 documents

OPTIONS

\$project

Open docs

```
1 // **
2 * specifications: The fields to
3 * include or exclude.
4 */
5 {
6   _id: 0,
7   company_name: 1,
8   role: 1,
9   'students.name': 1,
10  'students.email': 1
11 }
```

STAGE OUTPUT

Sample of 5 documents

OPTIONS

company_name: "TechCorp"
role: "Software Engineer"
students: Object
name: "Alice"
email: "alice@example.com"

company_name: "TechCorp"
role: "Software Engineer"
students: Object
name: "Bob"
email: "bob@example.com"

company_name: "Innovatech"
role: "Data Analyst"
students: Object
name: "Alice"
email: "alice@example.com"

company_name: "Web Solutions"
role: "Frontend Developer"
students: Object
name: "Alice"
email: "alice@example.com"

company_name: "Web Solutions"
role: "Frontend Developer"
students: Object
name: "Martin"
email: "martin@example.com"

Shell:

```
guvi> db.company_drives.aggregate([{$lookup:{from:"users", localField:"students_appeared", foreignField:"id", as:"students"}}])
[
  {
    _id: ObjectId('66519448f43a42766c5559cd'),
    id: '1',
    company_name: 'TechCorp',
    date: '2020-10-18',
    role: 'Software Engineer',
    students_appeared: [ 1, 2 ],
    students: [
      {
        _id: ObjectId('665194cfff43a42766c5559dd'),
        id: 1,
        name: 'Alice',
        email: 'alice@example.com',
        mentor_id: 1
      },
      {
        _id: ObjectId('665194cfff43a42766c5559de'),
        id: 2,
        name: 'Bob',
        email: 'bob@example.com',
        mentor_id: 1
      }
    ]
  },
  {
    _id: ObjectId('66519448f43a42766c5559ce'),
    id: '2',
    company_name: 'Innovatech',
    date: '2020-10-20',
    role: 'Data Analyst',
    students_appeared: [ 1 ],
    students: [
      {
        _id: ObjectId('665194cfff43a42766c5559dd'),
        id: 1,
        name: 'Alice',
        email: 'alice@example.com',
        mentor_id: 1
      }
    ]
  }
]
```

```

{
  _id: ObjectId('66519448f43a42766c5559ce'),
  id: '2',
  company_name: 'Innovatech',
  date: '2020-10-20',
  role: 'Data Analyst',
  students_appeared: [ 1 ],
  students: [
    {
      _id: ObjectId('665194cff43a42766c5559dd'),
      id: 1,
      name: 'Alice',
      email: 'Alice@example.com',
      mentor_id: 1
    }
  ]
},
{
  _id: ObjectId('66519448f43a42766c5559cf'),
  id: '3',
  company_name: 'Web Solutions',
  date: '2020-10-22',
  role: 'Frontend Developer',
  students_appeared: [ 1, 3 ],
  students: [
    {
      _id: ObjectId('665194cff43a42766c5559dd'),
      id: 1,
      name: 'Alice',
      email: 'Alice@example.com',
      mentor_id: 1
    },
    {
      _id: ObjectId('665194cff43a42766c5559df'),
      id: 3,
      name: 'Martin',
      email: 'Martin@example.com',
      mentor_id: 2
    }
  ]
}
]
}
Guvi>

```

4. Find the number of problems solved by the user in codekata

MongoDB Compass:

The screenshot shows the MongoDB Compass interface with a pipeline stage named 'Stage1: \$group' enabled. The stage input shows 5 documents, and the stage output shows 3 documents. The pipeline script in the center calculates the total problems solved for each user_id.

Stage Input (Sample of 5 documents):

- `{ "_id": "1", "user_id": "1", "module": "Array", "problems_solved": 30 }`
- `{ "_id": "2", "user_id": "2", "module": "Array", "problems_solved": 40 }`
- `{ "_id": "3", "user_id": "3", "module": "Array", "problems_solved": 50 }`
- `{ "_id": "4", "user_id": "1", "module": "Loop", "problems_solved": 30 }`
- `{ "_id": "5", "user_id": "2", "module": "Loop", "problems_solved": 30 }`

Stage Script:

```

1 // **
2 * _id: The id of the group.
3 * fieldN: The first field name.
4 */
5 {
6   _id: '$user_id',
7   total_problem_solved: {
8     $sum: '$problems_solved'
9   }
10 }

```

Stage Output (Sample of 3 documents):

- `{ "_id": "1", "total_problem_solved": 60 }`
- `{ "_id": "3", "total_problem_solved": 50 }`
- `{ "_id": "2", "total_problem_solved": 70 }`

Shell:

```
Guvi> db.codekata.aggregate([{$group:{$_id:$user_id', total_problem_solved:{$sum: "$problems_solved"}}}])
[
  { _id: '2', total_problem_solved: 70 },
  { _id: '3', total_problem_solved: 50 },
  { _id: '1', total_problem_solved: 60 }
]
Guvi> |
```

5.Find all the mentors with who has the mentee's count more than 15

MongoDB Compass:

localhost:27017 > Guvi > mentors

Documents 3 Aggregations Schema Indexes 1 Validation

{ \$expr: { \$gt: [{ \$size: "\$mentees" }, 15] } }

Generate query Explain Reset Find </> Options

ADD DATA EXPORT DATA UPDATE DELETE 1 - 2 of 2

```
{
  "_id": ObjectId('6651c098f43a42766c5559fa'),
  "id": "1",
  "name": "Dr. Smith",
  "expertise": "JavaScript, Node.js",
  "email": "smith@mentors.com",
  "mentees": Array (18)
}
```

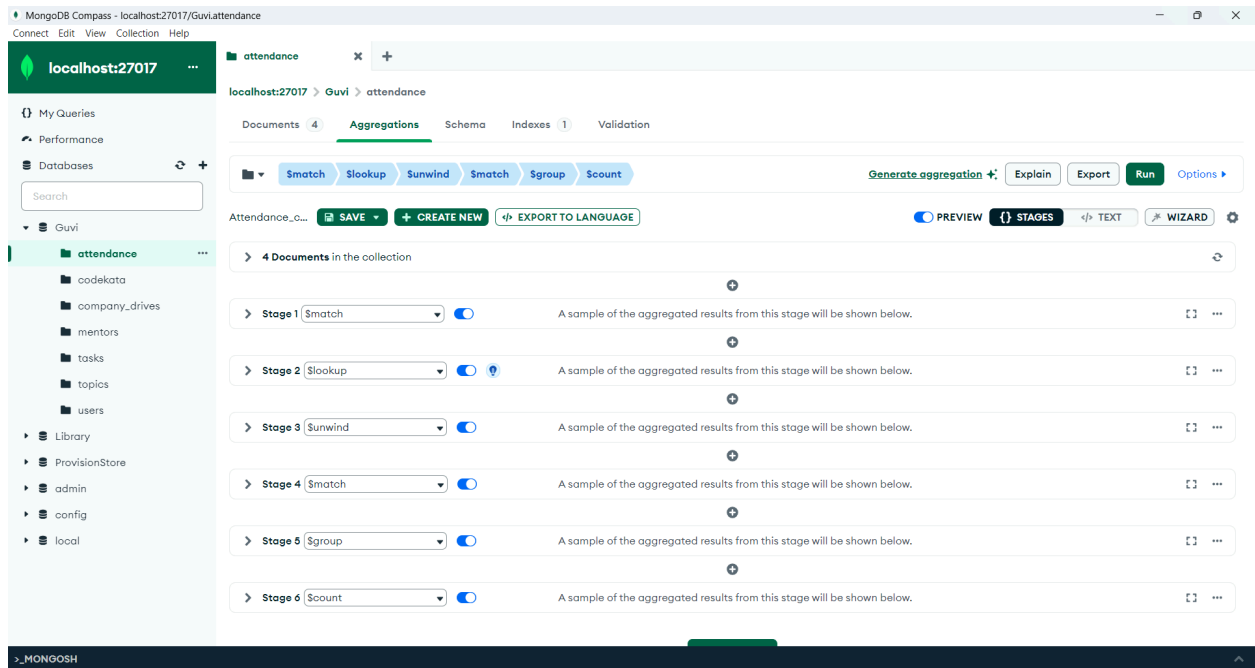
```
{
  "_id": ObjectId('6651c098f43a42766c5559fc'),
  "id": "3",
  "name": "Mr. Lee",
  "expertise": "DevOps, AWS",
  "email": "lee@mentors.com",
  "mentees": Array (20)
}
```

Shell:

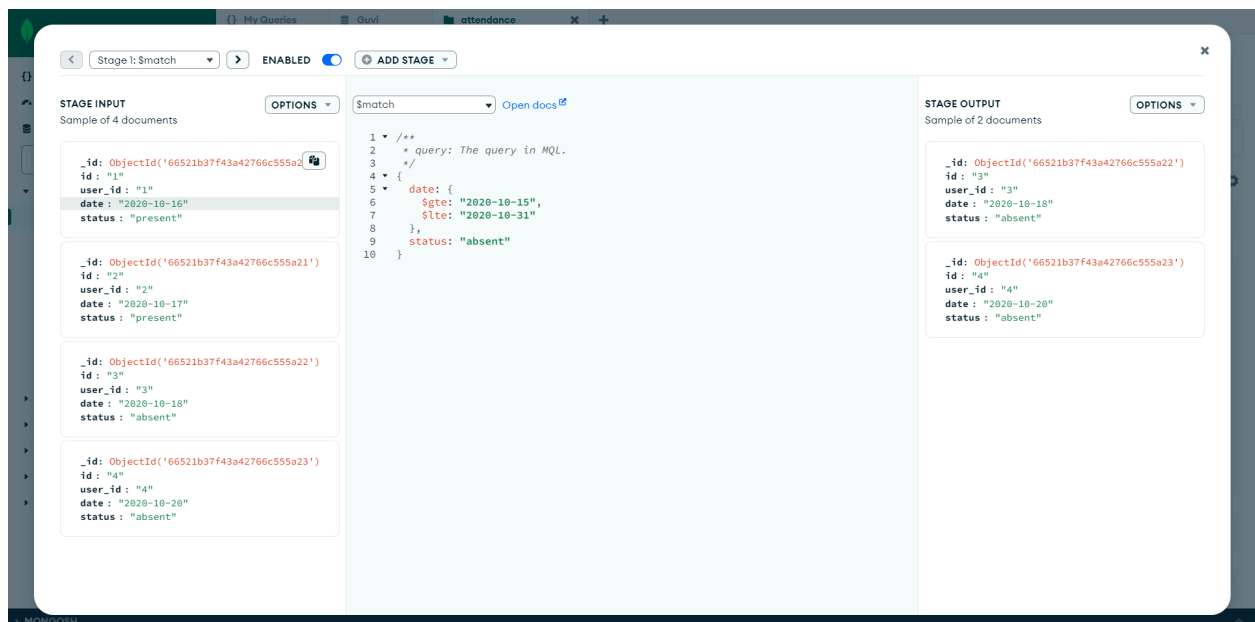
```
Guvi> db.mentors.find({$expr:{$gt:[$size:"$mentees"],15}})
[
  {
    _id: ObjectId('6651c098f43a42766c5559fa'),
    id: '1',
    name: 'Dr. Smith',
    expertise: 'JavaScript, Node.js',
    email: 'smith@mentors.com',
    mentees: [
      1, 2, 3, 4, 5, 6, 7,
      8, 9, 10, 11, 12, 13, 14,
      15, 16, 17, 18
    ]
  },
  {
    _id: ObjectId('6651c098f43a42766c5559fc'),
    id: '3',
    name: 'Mr. Lee',
    expertise: 'DevOps, AWS',
    email: 'lee@mentors.com',
    mentees: [
      1, 2, 3, 4, 5, 6, 7,
      8, 9, 10, 11, 12, 13, 14,
      15, 16, 17, 18, 19, 20
    ]
  }
]
Guvi> |
```

6. Find the number of users who are absent and task is not submitted between 15 oct-2020 and 31-oct-2020

MongoDB Compass:



1. \$match Stage



2.\$lookup Stage:

The screenshot shows the MongoDB Compass interface for a query stage named "Stage 2: \$lookup". The stage is enabled. The input is a sample of 2 documents. The output is also a sample of 2 documents. The configuration for the \$lookup stage is as follows:

```
1 /**
2  * from: The target collection.
3  * localField: The local join field.
4  * foreignField: The target join field.
5  * as: The name for the results.
6  * pipeline: Optional pipeline to run on the foreign collection.
7  * let: Optional variables to use in the pipeline field stages.
8  */
9 {
10   from: "tasks",
11   localField: "user_id",
12   foreignField: "user_id",
13   as: "tasks"
14 }
```

The stage output shows two documents. The first document has a `tasks` array containing one object with `task_name` "Todo List", `description` "Implement a todo list with CRUD operations.", `date` "2020-10-10", `due_date` "2020-10-20", `submitted` "no", and `topic_id` 3. The second document has a `tasks` array containing one object with `task_name` "Todo List", `description` "Implement a todo list with CRUD operations.", `date` "2020-10-18", `due_date` "2020-10-20", `submitted` "no", and `topic_id` 3.

3.\$unwind stage

The screenshot shows the MongoDB Compass interface for a query stage named "Stage 3: \$unwind". The stage is enabled. The input is a sample of 2 documents. The output is also a sample of 2 documents. The configuration for the \$unwind stage is as follows:

```
1 /**
2  * path: Path to the array field.
3  * includeArrayIndex: Optional name for index.
4  * preserveNullAndEmptyArrays: Optional
5  * toggle to unwind null and empty values.
6  */
7 {
8   path: "$tasks"
9 }
```

The stage output shows two documents. The first document has a `tasks` object with `task_name` "Todo List", `description` "Implement a todo list with CRUD operations.", `date` "2020-10-10", `due_date` "2020-10-20", `submitted` "no", and `topic_id` 3. The second document has a `tasks` object with `task_name` "Todo List", `description` "Implement a todo list with CRUD operations.", `date` "2020-10-18", `due_date` "2020-10-25", `submitted` "no", and `topic_id` 3.

4.\$match stage:

The screenshot shows the MongoDB Compass interface for a query named "attendance". The "Stage 4: \$match" stage is selected and enabled. The stage input shows two sample documents. The stage output shows the same two documents, indicating that both documents match the query criteria.

STAGE INPUT
Sample of 2 documents

```
{ "_id": ObjectId("66521b37f43a42766c555a22"), "id": "3", "user_id": "3", "date": "2020-10-18", "status": "absent", "tasks": { "_id": ObjectId("66521b5ef43a42766c555a22..."), "id": "3", "user_id": "3", "task_name": "Todo List", "description": "Implement a todo list with CRUD operations.", "date": "2020-10-10", "due_date": "2020-10-20", "submitted": "no", "topic_id": 3 } }
```

```
{ "_id": ObjectId("66521b37f43a42766c555a23"), "id": "4", "user_id": "4", "date": "2020-10-20", "status": "absent", "tasks": { "_id": ObjectId("66521b5ef43a42766c555a22..."), "id": "4", "user_id": "4", "task_name": "Todo List", "description": "Implement a todo list with CRUD operations.", "date": "2020-10-18", "due_date": "2020-10-25", "submitted": "no" } }
```

STAGE OUTPUT
Sample of 2 documents

```
{ "_id": ObjectId("66521b37f43a42766c555a22"), "id": "3", "user_id": "3", "date": "2020-10-18", "status": "absent", "tasks": { "_id": ObjectId("66521b5ef43a42766c555a22..."), "id": "3", "user_id": "3", "task_name": "Todo List", "description": "Implement a todo list with CRUD operations.", "date": "2020-10-10", "due_date": "2020-10-20", "submitted": "no", "topic_id": 3 } }
```

```
{ "_id": ObjectId("66521b37f43a42766c555a23"), "id": "4", "user_id": "4", "date": "2020-10-20", "status": "absent", "tasks": { "_id": ObjectId("66521b5ef43a42766c555a22..."), "id": "4", "user_id": "4", "task_name": "Todo List", "description": "Implement a todo list with CRUD operations.", "date": "2020-10-18", "due_date": "2020-10-25", "submitted": "no" } }
```

5.\$group stage:

The screenshot shows the MongoDB Compass interface for a query named "attendance". The "Stage 5: \$group" stage is selected and enabled. The stage input shows two sample documents. The stage output shows two documents, each representing a group of documents with the same user_id. The first document has user_id "4" and the second has user_id "3".

STAGE INPUT
Sample of 2 documents

```
{ "_id": ObjectId("66521b37f43a42766c555a22"), "id": "3", "user_id": "3", "date": "2020-10-18", "status": "absent", "tasks": { "_id": ObjectId("66521b5ef43a42766c555a22..."), "id": "3", "user_id": "3", "task_name": "Todo List", "description": "Implement a todo list with CRUD operations.", "date": "2020-10-10", "due_date": "2020-10-20", "submitted": "no", "topic_id": 3 } }
```

```
{ "_id": ObjectId("66521b37f43a42766c555a23"), "id": "4", "user_id": "4", "date": "2020-10-20", "status": "absent", "tasks": { "_id": ObjectId("66521b5ef43a42766c555a22..."), "id": "4", "user_id": "4", "task_name": "Todo List", "description": "Implement a todo list with CRUD operations.", "date": "2020-10-18", "due_date": "2020-10-25", "submitted": "no" } }
```

STAGE OUTPUT
Sample of 2 documents

```
{ "_id": "4" }
```

```
{ "_id": "3" }
```

6.\$count stage:

The screenshot displays the MongoDB Atlas query editor interface for a query named "attendance". The "Stage 6: \$count" stage is selected and enabled. The stage configuration is as follows:

- Stage Input:** Sample of 2 documents. The input fields are `_id: "4"` and `_id: "3"`.
- Stage Output:** Sample of 1 document. The output field is `absent_and_task_not_submitted: 2`.
- Options:** The "Options" dropdown is set to "\$count". A link "Open docs" is available.
- Code Editor:** The stage is configured using a JavaScript function. The code is as follows:

```
1 // **
2 * Provide the field name for the count.
3 */
4 'absent_and_task_not_submitted'
```

The interface also shows a sidebar with a list of stages and a bottom status bar indicating the MongoDB version.