

- **Title:** DB Assignment 5
- **Husam Alanazi**
- **Date:** Nov 22

Query 1: Over how many years was the unemployment data collected?

The screenshot shows a MongoDB query editor interface. At the top, there are tabs for '\$group' and '\$count'. Below the tabs, there are buttons for 'Explain', 'Export', 'Run', and 'Options'. The main area is divided into two panels. The left panel shows a query in JSON format:

```
1 [
2   {
3     "$group": {
4       "_id": "$Year"
5     }
6   },
7   {
8     "$count": "totalYears"
9   }
10 ]
11
```

The right panel shows the 'PIPELINE OUTPUT' section with a 'Sample of 1 document' and the result:

```
totalYears : 3
```

```
db.unemployment.distinct("Year").length;
```

Explanation:

This query retrieves all unique years in the dataset using the `distinct` method on the `Year` field. The `length` property is used to count how many unique years exist, effectively determining over how many years the unemployment data was collected.

Query 2: How many states were reported on in this dataset?

▼

\$group

\$count

ⓘ ExplainExportRunOptions ▶

Untitled – modifiedSAVE + CREATE NEWEXPORT TO LANGUAGEPREVIEWSTAGESTEXTWIZARD⚙

1 ▼ [
2 ▼ {
3 ▼ "\$group": {
4 "\$_id": "\$State"
5 }
6 },
7 ▼ {
8 "\$count": "totalStates"
9 }
10]
11

🔍☰

PIPELINE OUTPUT

Sample of 1 document

totalStates : 47

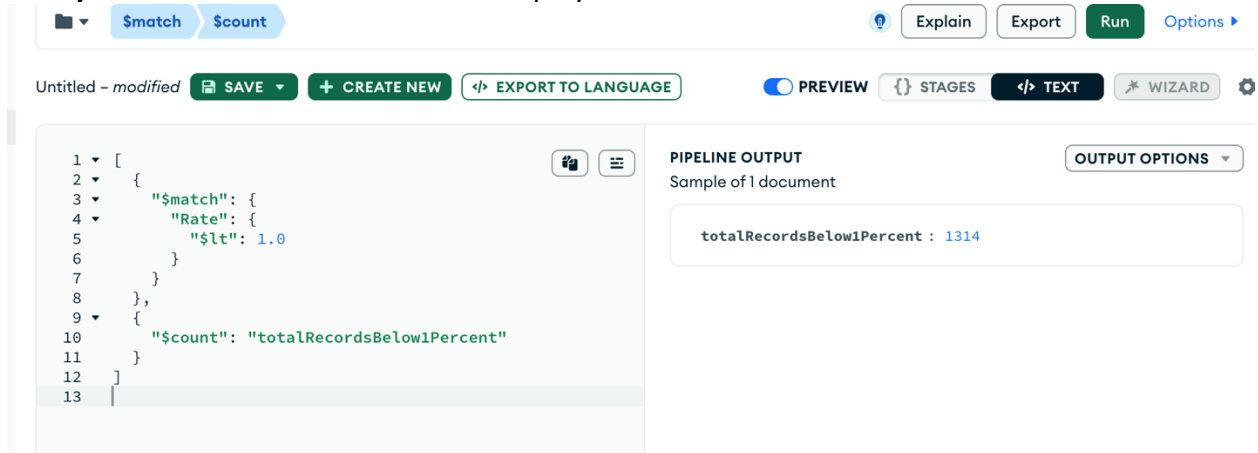
OUTPUT OPTIONS ▼

```
*/  
db.unemployment.distinct("State").length;
```

Explanation:

This query extracts all unique state names from the State field using the distinct method. The length property counts the total number of states that appear in the dataset.

Query 3: Count records where the unemployment rate is less than 1%



The screenshot shows a MongoDB query builder interface. At the top, there are tabs for '\$match' and '\$count'. Below the tabs, there are buttons for 'Explain', 'Export', 'Run', and 'Options'. The main area is divided into two sections: a query editor on the left and a pipeline output section on the right. The query editor shows a pipeline with two stages: a '\$match' stage with a condition 'Rate: { \$lt: 1.0 }' and a '\$count' stage with a field 'totalRecordsBelow1Percent'. The pipeline output section shows a sample of 1 document with the value 'totalRecordsBelow1Percent : 1314'.

```
1 [
2   {
3     "$match": {
4       "Rate": {
5         "$lt": 1.0
6       }
7     },
8   },
9   {
10    "$count": "totalRecordsBelow1Percent"
11  }
12 ]
13
```

PIPELINE OUTPUT
Sample of 1 document

totalRecordsBelow1Percent : 1314

```
db.unemployment.find({ Rate: { $lt: 1.0 } }).count();
```

Explanation:

This query finds and counts all records where the Rate field is less than 1.0. It helps identify how many counties had an unemployment rate below 1%.

Query 4: Find all counties with unemployment rates higher than 10%.

The screenshot shows a MongoDB query interface. On the left, a query is written in a text editor:

```
1 [
2   {
3     "$match": {
4       "Rate": {
5         "$gt": 10.0
6       }
7     }
8   }
9 ]
10
```

On the right, the "PIPELINE OUTPUT" section displays a sample of 10 documents. The first four documents are visible:

- `_id: ObjectId('673cec8791429a0fb7dbb7e3')`
`Year: 2015`
`Month: "February"`
`State: "Mississippi"`
`County: "Kemper County"`
`Rate: 10.6`
- `_id: ObjectId('673cec8791429a0fb7dbb7e6')`
`Year: 2015`
`Month: "February"`
`State: "Mississippi"`
`County: "Jefferson County"`
`Rate: 14.3`
- `_id: ObjectId('673cec8791429a0fb7dbb7e8')`
`Year: 2015`
`Month: "February"`
`State: "Mississippi"`
`County: "Sharkey County"`
`Rate: 11.1`
- `_id: ObjectId('673cec8791429a0fb7dbb7e9')`
`Year: 2015`
`Month: "February"`
`State: "Mississippi"`

```
db.unemployment.find({ Rate: { $gt: 10.0 } });
```

Explanation:

This query retrieves all documents where the unemployment Rate is greater than 10%. It provides a list of counties experiencing significantly high unemployment rates.

Query 5: Calculate the average unemployment rate across all states

▼ \$group

ⓘ Explain Export Run Options ▶

Untitled - modified

SAVE ▼ + CREATE NEW </> EXPORT TO LANGUAGE

PREVIEW {} STAGES </> TEXT WIZARD ⚙

1 ▼ [
2 ▼ {
3 ▼ "\$group": {
4 " _id": null,
5 "averageRate": {
6 "\$avg": "\$Rate"
7 }
8 }
9 }
10]
11 |

👤 ☰

PIPELINE OUTPUT

Sample of 1 document

OUTPUT OPTIONS ▼

_id: null
averageRate : 5.577907

```
db.unemployment.aggregate([  
  { $group: { _id: null, averageRate: { $avg: "$Rate" } } }  
]);
```

Explanation:

This query uses the \$group stage in an aggregation pipeline to calculate the average unemployment rate across all records. The null value for _id ensures that all documents are grouped together.

Smatch

Untitled - modified

SAVE

CREATE NEW

EXPORT TO LANGUAGE

PREVIEW

STAGES

TEXT

WIZARD

1 [

2 {

3 "\$match": {

4 "Rate": {

5 "\$gte": 5.0,

6 "\$lte": 8.0

7 }

8 }

9 }

10]

11 |

PIPELINE OUTPUT

OUTPUT OPTIONS

Sample of 10 documents

_id: ObjectId('673cec8791429a0fb7dbb7df')

Year: 2015

Month: "February"

State: "Mississippi"

County: "Newton County"

Rate: 6.1

_id: ObjectId('673cec8791429a0fb7dbb7e1')

Year: 2015

Month: "February"

State: "Mississippi"

County: "Monroe County"

Rate: 7.9

_id: ObjectId('673cec8791429a0fb7dbb7e2')

Year: 2015

Month: "February"

State: "Mississippi"

County: "Hinds County"

Rate: 6.1

_id: ObjectId('673cec8791429a0fb7dbb7e4')

Year: 2015

Month: "February"

State: "Mississippi"

```
db.unemployment.find({ Rate: { $gte: 5.0, $lte: 8.0 } });
```

Explanation:

This query finds all records where the unemployment Rate falls between 5% and 8% (inclusive). It helps identify counties with moderate unemployment rates.

Query 7: Find the state with the highest unemployment rate

The screenshot shows the MongoDB Compass interface. At the top, there's a text input field with the placeholder "Tell Compass what aggregation to build (e.g. how many movies were made each year)" and a "Generate" button. Below this, there are tabs for "\$sort" and "\$limit". To the right of these tabs are buttons for "Explain", "Export", "Run", and "Options".

Below the tabs, there's a header bar with "Untitled - modified", a "SAVE" button, a "+ CREATE NEW" button, an "EXPORT TO LANGUAGE" button, a "PREVIEW" toggle, a "STAGES" button, a "TEXT" button, a "WIZARD" button, and a settings gear icon.

The main area is split into two panels. The left panel shows the aggregation query in a code editor:

```
1 [
2   {
3     $sort: {
4       Rate: -1
5     }
6   },
7   {
8     $limit: 1
9   }
10 ]
```

The right panel is titled "PIPELINE OUTPUT" and shows a "Sample of 1 document". It contains the following data:

```
{
  "_id": ObjectId('673cecb391429a0fb7e113de'),
  "Year": 1992,
  "Month": "January",
  "State": "Colorado",
  "County": "San Juan County",
  "Rate": 58.4
}
```

```
db.unemployment.aggregate([
  { $sort: { Rate: -1 } },
  { $limit: 1 }
])
```

Explanation:

This query sorts the dataset in descending order by Rate using the \$sort stage and then retrieves the top document using \$limit: 1. This document represents the state with the highest unemployment rate.

Query 8: Count counties with unemployment rates above 5%.

▼

\$match

\$count

🔔

Explain

Export

Run

Options ▶

Untitled – modified

SAVE

+ CREATE NEW

EXPORT TO LANGUAGE

PREVIEW

STAGES

TEXT

WIZARD

⚙️

1 ▼ [

2 ▼ {

3 ▼ {

4 "Rate": { "\$gt": 5.0 } }

5 } }

6 },

7 {

8 "\$count": "totalCount"

9 }

10]

11

PIPELINE OUTPUT

Sample of 1 document

totalCount : 1020346

```
db.unemployment.aggregate([
  { $match: { Rate: { $gt: 5.0 } } },
  { $count: "totalCount" }
]);
```

Explanation:

This query filters documents where the Rate is greater than 5% using the \$match stage and counts the resulting records using \$count. The count represents the number of counties with an unemployment rate above 5%

Query 9: Calculate the average unemployment rate per state by year. The query groups documents by State and Year, then calculates the average unemployment rate (Rate) for each group.

▼

\$group

?

 Explain

Export

Run

Options ▶

Jntitled - modified

SAVE

+ CREATE NEW

EXPORT TO LANGUAGE

PREVIEW

STAGES

TEXT

WIZARD

⚙

1 ▼

[

2 ▼

{

3 ▼

\$group: {

4 ▼

_id: {

5

State: "\$State",

6

Year: "\$Year"

7

},

8 ▼

averageRate: {

9

\$avg: "\$Rate"

10

}

11

}

12

}

13

]

PIPELINE OUTPUT

Sample of 10 documents

▶ _id: Object

averageRate : 3.656439393939394

▶ _id: Object

averageRate : 4.747916666666667

▶ _id: Object

averageRate : 4.746536796536796

▶ _id: Object

averageRate : 4.022936507936508

▶ _id: Object

averageRate : 6.554130434782609

▶ _id: Object

averageRate : 4.383

▶ _id: Object

averageRate : 5.423260869565218

```
db.unemployment.aggregate([
  { $group: { _id: "$State", totalRate: { $sum: "$Rate" } } }
]);
```

Explanation:

This query groups the dataset by both State and Year using the \$group stage. It then calculates the average unemployment rate for each state in each year.

Query 10: For each state, calculate the total unemployment rate across all counties sum of all county rates.

The screenshot shows a MongoDB query editor interface. At the top, there's a toolbar with buttons for 'Sgroup', 'Explain', 'Export', 'Run', and 'Options'. Below this is a status bar with 'Untitled - modified', 'SAVE', 'CREATE NEW', 'EXPORT TO LANGUAGE', 'PREVIEW', 'STAGES', 'TEXT', 'WIZARD', and a settings icon. The main editor area is split into two panes. The left pane shows a JSON document structure with line numbers 1 through 9. The right pane, titled 'PIPELINE OUTPUT', shows a 'Sample of 10 documents' with a table of results. Below the editor panes, a dark-themed code block contains the MongoDB aggregation query.

```
1 [
2   {
3     "$group": {
4       "_id": "$State",
5       "totalRate": { "$sum": "$Rate" }
6     }
7   }
8 ]
9
```

_id	totalRate
"Kentucky"	28833.8
"Arizona"	4577.9
"Utah"	4928.8
"Louisiana"	12575.2
"South Dakota"	8408.8
"Delaware"	553.4
"Arkansas"	15876.6

```
db.unemployment.aggregate([
  { $group: { _id: "$State", totalRate: { $sum: "$Rate" } } }
]);
```

Explanation:

This query groups the dataset by State and calculates the total unemployment rate for each state by summing up all the Rate values for counties in that state.

Query 11: Calculate the Total Unemployment Rate Across All Counties for Each State Starting from 2015 Onward

The screenshot shows the MongoDB Atlas query editor interface. At the top, there are tabs for '\$match' and '\$group'. Below the tabs, there are buttons for 'Explain', 'Export', 'Run', and 'Options'. The main editor area shows a JSON query pipeline:

```
1 [
2   {
3     "$match": {
4       "Year": { "$gte": 2015 }
5     }
6   },
7   {
8     "$group": {
9       "_id": "$State",
10      "totalRate": { "$sum": "$Rate" }
11    }
12  }
13 ]
14
```

On the right side, the 'PIPELINE OUTPUT' section shows a sample of 10 documents. The output is a list of states and their total unemployment rates:

_id	totalRate
"Arizona"	4556.9
"Washington"	9376.9
"Oklahoma"	13672.5
"Minnesota"	13857.8
"Michigan"	7516.3
"Vermont"	1988.7
"Colorado"	8583.8

```
db.unemployment.aggregate([
  { $match: { Year: { $gte: 2015 } } },
  { $group: { _id: "$State", totalRate: { $sum: "$Rate" } } }
]);
```

Explanation:

This query first filters the dataset to include only documents from the year 2015 onward using the \$match stage. It then groups the filtered data by State and calculates the total unemployment rate for each state.