DBMS Assignment 5

Sarah Groark

Nov 20, 2024

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

```
> db.unemployment.distinct("Year")
< [
    1990, 1991, 1992, 1993, 1994,
    1995, 1996, 1997, 1998, 1999,
    2000, 2001, 2002, 2003, 2004,
    2005, 2006, 2007, 2008, 2009,
    2010, 2011, 2012, 2013, 2014,
    2015, 2016
  ]
> db.unemployment.distinct("Year").length
< 27
```

This query takes the distinct aggregate and processes the “Year” field in the schema to output all of the distinct years that are recorded in the data. As a result, the query output represents all of the years, in a list format, that data was recorded. Finally, the second query outputs the length of this list to return the range of years that unemployment data was collected. It should be noted that these two queries do not both have to be executed to get the final answer of 27, but was done so for visualization purposes.

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

```
> db.unemployment.distinct("State")
< [
    'Alabama',        'Arizona',       'Arkansas',
    'California',     'Colorado',      'Connecticut',
    'Delaware',       'Hawaii',        'Idaho',
    'Illinois',       'Indiana',       'Iowa',
    'Kansas',         'Kentucky',      'Louisiana',
    'Maine',          'Maryland',      'Massachusetts',
    'Michigan',       'Minnesota',     'Mississippi',
    'Missouri',       'Montana',       'Nebraska',
    'Nevada',         'New Hampshire', 'New Jersey',
    'New Mexico',     'New York',      'North Carolina',
    'North Dakota',   'Ohio',          'Oklahoma',
    'Oregon',         'Pennsylvania',  'Rhode Island',
    'South Carolina', 'South Dakota',  'Tennessee',
    'Texas',          'Utah',          'Vermont',
    'Virginia',       'Washington',    'West Virginia',
    'Wisconsin',      'Wyoming'
  ]
> db.unemployment.distinct("State").length
< 47
```

This query aggregates the distinct states from the schema and outputs the list of the unique states that data has been recorded under. The next query calculates the length of this list to output the amount of states that were reported on in the dataset. Similar to above, the second query can be calculated independently of the first to retrieve the answer.

Query 3 = What does this query compute?
db.unemployment.find({Rate : {$lt: 1.0}}).count()

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

< 657
```

This query returns the number of instances that a state county had an unemployment rate less than 1% (during any of the recorded months and years).

Query 4 =

```
> db.unemployment.find({Rate : {$gt: 10.0}},{County:1, State:1,_id:0})

< {
    State: 'Mississippi',
    County: 'Kemper County'
  }
  {
    State: 'Mississippi',
    County: 'Jefferson County'
  }
  {
    State: 'Mississippi',
    County: 'Sharkey County'
  }
  {
    State: 'Mississippi',
    County: 'Tunica County'
  }
```

The query outputs the county and associated state where the rate is higher than 10%. It does so by filtering by the Rate field using $gt: 10.0 to retrieve the documents that meet this condition. The second part of the query projects onto the State and County fields to output a neat result.

Query 5 = Calculate the average unemployment rate across all states

```
[
  {
    $group:
      /**
       * _id: null.
       * AverageRate: computed average unemployment rate.
       */
      {
        _id: null,
        AverageRate: {
          $avg: "$Rate"
        }
      }
  },
  {
    $project:
      /**
       * specifications: The fields to
       *   include or exclude.
       */
      {
        _id: 0,
        AverageRate: 1
      }
  }
]
```

OUTPUT:

```
{
  "AverageRate": 5.577907
}
```

This query occurs in two stages: a group and project stage. The group stage calculates the average unemployment rate across all of the documents (thus, all of the states) and stores it as AverageRate. The second project stage excludes the _id field and solely outputs the AverageRate, representing the average unemployment rate across all of the states reported.

Query 6 = Find all **counties** with an unemployment rate between **5% and 8%**

```
[
  {
    $match:
      /**
       * filter Rates between 5 and 8%.
       */
      {
        Rate: {
          $gt: 5.0,
          $lt: 8.0
        }
      }
  },
  {
    $group:
      /**
       * _id: The id of the group - County field.
       */
      {
        _id: "$County"
      }
  },
  {
    $project:
      /**
       * Fields to include: County.
       */
      {
        County: "$_id",
        _id: 0
      }
  }
]
```

PIPELINE OUTPUT OUTPUT OPTIONS

Sample of 10 documents

```
County : "Vanderburgh County"
County : "Allegan County"
County : "Northumberland County"
County : "Washakie County"
County : "Belmont County"
County : "Miller County"
```

This query occurs in three different stages: $match, $group, and $project. The match stage is responsible for filtering the documents to retrieve those that meet the criteria – an unemployment rate between 5 and 8%, this is done through the expression: Rate{ $gt: 5.0, $lt: 8.0}. Then, the group stage asserts the County field as the _id, so that the counties are grouped together. The project stage then includes the County field in the output and excludes the _id field to maintain neatness.

Query 7 = Find the state with the **highest unemployment rate**. **Hint. Use { $limit: 1 }**

```
[
  {
    $project:
      /**
       * specifications: The fields to
       *   include or exclude.
       */
      {
        State: 1,
        Rate: 1,
        _id: 0
      }
  },
  {
    $sort:
      /**
       * Provide any number of field/order pairs.
       */
      {
        Rate: -1
      }
  },
  {
    $limit:
      /**
       * Provide the number of documents to limit.
       */
      1
  }
]
```

OUTPUT:

{
  "State": "Colorado",
  "Rate": 58.4
}

This query projects the State and Rate fields of the documents, then sorts them by the Rate in ascending order (highest to lowest). The final stage, $limit, takes the top document in the pipeline output, which represents the highest unemployment rate.

Query 8 = Count how many **counties** have an unemployment rate above **5%**.

```
[
  {
    $match:
      /**
    * query: matches with instances of
    unemployment rates over 5.0%.
    */
      {
        Rate: {
          $gt: 5.0
        }
      }
  },
  {
    $group:
      /**
       * _id: The id of the group (Count)
       */
      {
        _id: "$County"
      }
  },
  {
    $count:
      /**
       *Count number of counties.
       */
      "count"
  }
]
```

OUTPUT:

{
  "Count": 1697
}

This query's pipeline first matches the documents that meet the criteria of having an unemployment rate above 5%. The group stage groups the current pipeline output by the County field to access these counties that are above 5%. Then, the $count stage counts the amount of these instances.

9 = Calculate the average unemployment rate per state by year.

```
[
  {
    $group:
      /**
       * query: The query in MQL.
       */
      {
        _id: {
          state: "$State",
          year: "$Year"
        },
        avg_unemployment_rate: {
          $avg: "$Rate"
        }
      }
  },
  {
    $project:
      /**
       * specifications: The fields to
       *   include or exclude.
       */
      {
        _id: 0,
        state: "$_id.state",
        year: "$_id.year",
        avg_unemployment_rate:
          "$avg_unemployment_rate"
      }
  },
```

```
state : "Alabama"
year : 2014
avg_unemployment_rate : 7.917164179104478
```

```
state : "Arizona"
year : 2014
avg_unemployment_rate : 9.470555555555556
```

```
state : "Arkansas"
year : 2014
avg_unemployment_rate : 6.968
```

```
state : "California"
year : 2014
avg_unemployment_rate : 8.820258620689655
```

```
state : "Colorado"
year : 2014
avg_unemployment_rate : 5.198958333333334
```

This query groups the _id field by state and year as well as calculates the average unemployment rate as specified by the _id field. Next, the project stage includes the state, year, and avg_unemployment_rate fields in the output.

10 = (Extra Credit) For each state, calculate the total unemployment rate across all counties (sum of all county rates).

```
[
  {
    $group:
      /**
       * _id: The id of the group (State field).
       * fieldN: sum of unemployment rates.
       */
      {
        _id: "$State",
        total_unemployment_rate: {
          $sum: "$Rate"
        }
      }
  },
  {
    $project:
      /**
     * include total_unemployment_rate
     and state fields.
     */
      {
        total_unemployment_rate: 1,
        _id: 0,
        state: "$_id"
      }
  }
]
```

```
total_unemployment_rate : 553.4
state : "Delaware"

total_unemployment_rate : 3533
state : "Wyoming"

total_unemployment_rate : 8776.9
state : "Montana"

total_unemployment_rate : 13668.3
state : "Iowa"
```

This query takes the group stage to group by the State field and also calculates the sum of these rates by state. The project field includes the state and total_unemployment_rate fields in the output, but excludes the _id field.

11 = (Extra Credit) The same as Query 10 but for states with data from 2015 onward

```
[
  {
    $match:
      /**
       * Criteria: years from 2015 and forward
       */
      {
        Year: {
          $gte: 2015
        }
      }
  },
  {
    $group:
      /**
       * _id: The id of the group (State).
       * total_unemployment_rate: summation of the rates by state.
       */
      {
        _id: "$State",
        total_unemployment_rate: {
          $sum: "$Rate"
        }
      }
  },
  {
    $project:
      /**
       * excludes _id, includes state and total_unemployment_rate.
       */
      {
        _id: 0,
        state: "$_id",
        total_unemployment_rate: 1
      }
  }
]
```

```
total_unemployment_rate : 3078.1
state : "Maryland"

total_unemployment_rate : 3377.5
state : "Utah"

total_unemployment_rate : 9605.4
state : "Arkansas"

total_unemployment_rate : 10408.2
state : "Indiana"

total_unemployment_rate : 4298.2
state : "North Dakota"
```

This query does the same as #10 but includes a match stage at the beginning of the pipeline to filter documents with data from the year 2015 and onward.