- The `$sort` stage orders the documents in the pipeline by the `pop` field value, from smallest to largest; i.e. by increasing order. This operation does not alter the documents.
- The next `$group` stage groups the now-sorted documents by the `_id.state` field (i.e. the `state` field inside the `_id` document) and outputs a document for each state.

  The stage also calculates the following four fields for each state. Using the `$last` expression, the `$group` operator creates the `biggestCity` and `biggestPop` fields that store the city with the largest population and that population. Using the `$first` expression, the `$group` operator creates the `smallestCity` and `smallestPop` fields that store the city with the smallest population and that population.

  The documents, at this stage in the pipeline, resemble the following:

  ```
  {
    "_id" : "WA",
    "biggestCity" : "SEATTLE",
    "biggestPop" : 520096,
    "smallestCity" : "BENGE",
    "smallestPop" : 2
  }
  ```

- The final `$project` stage renames the `_id` field to `state` and moves the `biggestCity`, `biggestPop`, `smallestCity`, and `smallestPop` into `biggestCity` and `smallestCity` embedded documents.

The output documents of this aggregation operation resemble the following:

```
{
  "state" : "RI",
  "biggestCity" : {
    "name" : "CRANSTON",
    "pop" : 176404
  },
  "smallestCity" : {
    "name" : "CLAYVILLE",
    "pop" : 45
  }
}
```

7.3.2 Aggregation with User Preference Data

Data Model

Consider a hypothetical sports club with a database that contains a `users` collection that tracks the user's join dates, sport preferences, and stores these data in documents that resemble the following:

```
{
  _id : "jane",
  joined : ISODate("2011-03-02"),
  likes : ["golf", "racquetball"]
}
{
  _id : "joe",
  joined : ISODate("2012-07-02"),
  likes : ["tennis", "golf", "swimming"]
}
```