Ahmed Ehab Ahmed
MongoDB Lab3

a- Display employees fullname and department name for all employees

```
inventory> db.employee.aggregate([
...   {
...     $lookup: {
...       from: "department",
...       localField: "department_id",
...       foreignField: "department_id",
...       as: "dept",
...     },
...   },
...   {
...     $unwind: "$dept",
...   },
...   {
...     $project: {
...       _id: 0,
...       full_name: 1,
...       department_name: "$dept.department_description",
...     },
...   },
... ]);
[
  { full_name: 'Sheri Nowmer', department_name: 'HQ General Management' },
  {
    full_name: 'Derrick Whelply',
```

b- Display employees with position “VP Country Manager” (only display employee full name and salary).

```
inventory> db.employee.aggregate([
...   {
...     $match: {
...       position_title: "VP Country Manager"
...     },
...   },
...   {
...     $project: {
...       _id: 0,
...       full_name: 1,
...       salary: 1
...     },
...   },
... ]);
[
  { full_name: 'Derrick Whelply', salary: 40000 },
  { full_name: 'Michael Spence', salary: 40000 },
  { full_name: 'Maya Gutierrez', salary: 35000 },
  { full_name: 'Beverly Baker', salary: 30000 },
  { full_name: 'Pedro Castillo', salary: 35000 },
  { full_name: 'Laurie Borges', salary: 35000 }
]
inventory>
```

c- Display customers full names and their regions.

```
inventory> db.customer.aggregate([
...   {
...     $lookup: {
...       from: "region",
...       localField: "address.customer_region_id",
...       foreignField: "region_id",
...       as: "regions",
...     },
...   },
...   {
...     $unwind: "$regions"
...   },
...   {
...     $project: {
...       _id: 0,
...       fullname: 1,
...       region: "$regions.sales_region"
...     },
...   },
... ]);
[
  { fullname: 'Sheri Nowmer', region: 'Mexico South' },
  { fullname: 'Derrick Whelply', region: 'Canada West' },
  { fullname: 'Jeanne Derry', region: 'North West' },
```

d- In product find all products that was branded by " Washington " (try to createIndex on brand_name and test your query speed)

```
inventory> db.product.aggregate([
...   {
...     $match: {
...       brand_name: "Washington"
...     },
...   },
... ]);
[
  {
    _id: ObjectId('53c7c973ccf26e6de850bd88'),
    product_class_id: Long('30'),
    product_id: Long('1'),
    brand_name: 'Washington',
    product_name: 'Washington Berry Juice',
    SKU: Long('90748583674'),
    SRP: 2.85,
    gross_weight: 8.39,
    net_weight: 6.39,
    recyclable_package: false,
    low_fat: false,
    units_per_case: Long('30'),
    cases_per_pallet: Long('14'),
    shelf_width: 16.9,
    shelf_height: 12.6
```

```
inventory> db.product.explain("executionStats").aggregate([
...   {
...     $match: {
...       brand_name: "Washington"
...     },
...   },
... ]);
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'inventory.product',
    parsedQuery: { brand_name: { '$eq': 'Washington' } },
    indexFilterSet: false,
    queryHash: '44393A5D',
    planCacheShapeHash: '44393A5D',
    planCacheKey: 'ABDBFEB5',
    optimizationTimeMillis: 0,
    optimizedPipeline: true,
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    prunedSimilarIndexes: false,
    winningPlan: {
      isCached: false,
      stage: 'COLLSCAN',
      filter: { brand_name: { '$eq': 'Washington' } },
      direction: 'forward'
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 11,
    executionTimeMillis: 1,
    totalKeysExamined: 0,
    totalDocsExamined: 1560,
    executionStages: {
      isCached: false,
      stage: 'COLLSCAN',
      filter: { brand_name: { '$eq': 'Washington' } },
      nReturned: 11,
```

```
inventory> db.product.createIndex({ brand_name: 1 });
brand_name_1
inventory> db.product.explain("executionStats").aggregate([
...   {
...     $match: {
...       brand_name: "Washington"
...     },
...   },
... ]);
{
  explainVersion: '1',
  queryPlanner: {
    namespace: 'inventory.product',
    parsedQuery: { brand_name: { '$eq': 'Washington' } },
    indexFilterSet: false,
    queryHash: '44393A5D',
    planCacheShapeHash: '44393A5D',
    planCacheKey: 'FFEB9F73',
    optimizationTimeMillis: 0,
    optimizedPipeline: true,
    maxIndexedOrSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExplodeReached: false,
    prunedSimilarIndexes: false,
    winningPlan: {
      isCached: false,
      stage: 'FETCH',
      inputStage: {
        stage: 'IXSCAN',
        keyPattern: { brand_name: 1 },
        indexName: 'brand_name_1',
        isMultiKey: false,
        multiKeyPaths: { brand_name: [] },
        isUnique: false,
        isSparse: false,
        isPartial: false,
        indexVersion: 2,
        direction: 'forward',
        indexBounds: { brand_name: [ '["Washington", "Washington"]' ] }
      }
    },
    rejectedPlans: []
  },
  executionStats: {
    executionSuccess: true,
    nReturned: 11,
    executionTimeMillis: 0,
    totalKeysExamined: 11,
    totalDocsExamined: 11,
    executionStages: {
```

a- Group products by brand name, count number

```
inventory> db.product.aggregate([
...   {
...     $group: {
...       _id: "$brand_name",
...       count: { $sum: 1 },
...     },
...   },
...   {
...     $project: {
...       _id: 0,
...       brandName: "$_id",
...       count: 1
...     }
...   }
... ]);
[
  { count: 29, brandName: 'Plato' },
  { count: 5, brandName: 'Dollar' },
```

b- Group products by brand_name and product_name ,only select brand names ("Blue Label","King","Washington") then sort them by brand_name and product_name ascending

```
inventory> db.product.aggregate([
...   {
...     $match: {
...       brand_name: {
...         $in: ["Washington", "Blue Label", "King"],
...       },
...     },
...   },
...   {
...     $group: {
...       _id: {
...         brand: "$brand_name",
...         product: "$product_name",
...       },
...     },
...   },
...   {
...     $sort: {
...       "_id.brand": 1,
...       "_id.product": 1
...     }
...   },
...   {
...     $project: {
...       _id: 0,
...       brand: "$_id.brand",
...       product: "$_id.product",
...     },
...   },
... ]);
[
  { brand: 'Blue Label', product: 'Blue Label Beef Soup' },
  { brand: 'Blue Label', product: 'Blue Label Canned Beets' },
```

BONUS:Display maximum salary for each department (display department name and maximum salary)

```
inventory> db.employee.aggregate([
...   {
...     $group: {
...       _id: "$department_id",
...       maxSalary: { $max: "$salary" },
...     },
...   },
...   {
...     $lookup: {
...       from: "department",
...       localField: "_id",
...       foreignField: "department_id",
...       as: "dept",
...     },
...   },
...   {
...     $unwind: "$dept",
...   },
...   {
...     $project: {
...       _id: 0,
...       departemnt: "$dept.department_description",
...       maxSalary: 1,
...     },
...   },
... ]);
[
  { maxSalary: 9000, departemnt: 'Store Information Systems' },
  { maxSalary: 17000, departemnt: 'Store Management' },
  { maxSalary: 25000, departemnt: 'HQ Information Systems' },
  { maxSalary: 80000, departemnt: 'HQ General Management' },
  { maxSalary: 45000, departemnt: 'HQ Marketing' },
  { maxSalary: 8500, departemnt: 'Store Permanent Butchers' },
  { maxSalary: 50000, departemnt: 'HQ Finance and Accounting' },
  { maxSalary: 6700, departemnt: 'HQ Human Resources' },
  { maxSalary: 6500, departemnt: 'Store Permanent Stockers' },
  { maxSalary: 9000, departemnt: 'Store Permanent Checkers' },
  { maxSalary: 20, departemnt: 'Store Temporary Stockers' },
  { maxSalary: 20, departemnt: 'Store Temporary Checkers' }
]
inventory>
```