



# **MongoDB Project – Google Store Visitor Data**

BUAN 6320.003

Group Members  
Megha Syam Gutti  
Revanth Chintala  
Srivasanth Balaji  
Sonali Pandey

Group 21:

## Contents

Data Review .....	3
Assumptions/Notes About Data Collections, Attributes and Relationships between Collections .....	3
Physical Database .....	4
Assumptions/Notes About Data Set .....	4
Screen shot of Physical Database objects (Database, Collections and Attributes) .....	4
Data in the Database.....	4
MongoDB Queries/Code .....	5
Query 1.....	5
Question.....	5
Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result) .....	5
Translation .....	5
Screen Shot of MongoDB Query/Code and Results.....	5
Query 2.....	7
Question.....	7
Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result) .....	7
Translation .....	7
Screen Shot of MongoDB Query/Code and Results.....	7
Query 3.....	9
Question.....	9
Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result) .....	9
Translation .....	9
Screen Shot of MongoDB Query/Code and Results.....	9
Query 4.....	11
Question.....	11
Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result) .....	11
Translation .....	11
Screen Shot of MongoDB Query/Code and Results.....	11
Query 5.....	13
Question.....	13
Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result) .....	13
Translation .....	13
Screen Shot of MongoDB Query/Code and Results.....	13

Query 6.....	14
Question.....	14
Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result) .....	14
Translation .....	14
Screen Shot of MongoDB Query/Code and Results.....	14
Query 7.....	15
Question.....	15
Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result) .....	15
Translation .....	15
Screen Shot of MongoDB Query/Code and Results.....	15

## Data Review

### Assumptions/Notes About Data Collections, Attributes and Relationships between Collections

There is only one collection on our project.

Database Name: GroupProject

Collection Name: Group21

```
> use GroupProject
switched to db GroupProject
> db.getCollectionNames()
[ "Group21" ]
```

# Physical Database

## Assumptions/Notes About Data Set

Screen shot of Physical Database objects (Database, Collections and Attributes)

### Data in the Database

Collection Name	Relationships With Other Collections (if any)	# of Rows in Table
Group21	NA	804693

```
> db.Group21.findOne()
{
  "_id" : ObjectId("5bffa3b744ad6b47b0bcc61b"),
  "sessionId" : "6167871330617112363_1508151024",
  "browser" : " Chrome",
  "browserVersion" : " not available in demo dataset",
  "browserSize" : " not available in demo dataset",
  "operatingSystem" : " Macintosh",
  "operatingSystemVersion" : " not available in demo dataset",
  "isMobile" : " false",
  "mobileDeviceBranding" : " not available in demo dataset",
  "mobileDeviceModel" : " not available in demo dataset",
  "mobileInputSelector" : " not available in demo dataset",
  "mobileDeviceInfo" : " not available in demo dataset",
  "mobileDeviceMarketingName" : " not available in demo dataset",
  "flashVersion" : " not available in demo dataset",
  "language" : " not available in demo dataset",
  "screenColors" : " not available in demo dataset",
  "screenResolution" : " not available in demo dataset",
  "deviceCategory" : " desktop",
  "fullVisitorID" : "6.17E+18",
  "visits" : "1",
  "hits" : "4",
  "pageviews" : "4",
  "newVisits" : "",
  "campaign" : " (not set)",
  "source" : " google",
  "medium" : " organic",
  "keywordadwords" : "",
  "continent" : " Asia",
  "subContinent" : " Southeast Asia",
  "country" : " Singapore",
  "region" : " (not set)",
  "metro" : " (not set)",
  "city" : " (not set)",
  "cityId" : " not available in demo dataset",
  "networkDomain" : " myrepublic.com.sg",
  "latitude" : " not available in demo dataset",
  "longitude" : " not available in demo dataset",
  "networkLocation" : " not available in demo dataset",
  "channelGrouping" : "Organic Search",
  "date" : "20171016",
  "socialEngagementType" : "Not Socially Engaged",
  "visitId" : "1508151024",
  "visitNumber" : "2",
  "visitStartTime" : "1508151024"
}
```

## MongoDB Queries/Code

### Query 1

#### Question2

**Is a blackberry user less likely to visit the store than iOS user?**

Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result)

The column in interest in the question above is **OperatingSystem** and its **count**, especially that of **'BlackBerry'** and **'iOS'**. So, rows that have OperatingSystem as BlackBerry and iOS are filtered first. After filtering, the aggregate function **\$group** is applied to find the count of Blackberry and iOS.

#### Translation

Match the OperatingSystem column with the values Blackberry and iOS, group by the column operatingSystem.

Screen Shot of MongoDB Query/Code and Results

```
> db.getCollectionNames()
[ "Group21" ]
> db.Group21.aggregate([
...   {
...     '$match': {
...       'operatingSystem': {
...         '$in': [
...           'iOS', 'BlackBerry'
...         ]
...       }
...     }
...   }, {
...     '$group': {
...       '_id': '$operatingSystem',
...       'OS_Count': {
...         '$sum': 1
...       }
...     }
...   }
... ])
{ "_id" : "BlackBerry", "OS_Count" : 132 }
{ "_id" : "iOS", "OS_Count" : 111669 }
```

Since the count of BlackBerry is **132** and iOS is **111669**, BlackBerry user is less likely to visit store than iOS user.

Query:

```
db.Group21.aggregate([
{
  '$match': {
    'operatingSystem': {
      '$in': [
        ' iOS', ' BlackBerry'
      ]
    }
  }
}, {
  '$group': {
    '_id': '$operatingSystem',
    'OS_Count': {
      '$sum': 1
    }
  }
}
])
```

## Query 2

### Question3

**Which date had the most number of iOS users from Belgium?**

Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result)

The columns in interest in the question above are **OperatingSystem**, **country** and **date**. Rows that have OperatingSystem as 'iOS' and country as 'Belgium' are filtered as per the question. The derived collection is then grouped by date using **\$group** function. The result is then sorted in descending order using **\$sort** aggregate function and the top result is displayed using the aggregate function **\$limit**.

### Translation

Match the Operatingsystem column with the value iOS and country column with Belgium, group collection by the column date, sort the count in descending order and display top result.

### Screen Shot of MongoDB Query/Code and Results

```
> db.Group21.aggregate([
...   {
...     '$match': {
...       'operatingSystem': 'iOS',
...       'country': 'Belgium'
...     }
...   }, {
...     '$group': {
...       '_id': '$date',
...       'count_of_iOS': {
...         '$sum': 1
...       }
...     }
...   }, {
...     '$sort': {
...       'count_of_iOS': -1
...     }
...   }, {
...     '$limit': 1
...   }
... ])
{ "_id" : "20170814", "count_of_iOS" : 19 }
```

**14<sup>th</sup> August 2017** had most iOS users from Belgium.



```
db.Group21.aggregate([
  {
    '$match': {
      'operatingSystem': ' iOS',
      'country': ' Belgium'
    }
  }, {
    '$group': {
      '_id': '$date',
      'count_of_iOS': {
        '$sum': 1
      }
    }
  }, {
    '$sort': {
      'count_of_iOS': -1
    }
  }, {
    '$limit': 1
  }
])
```

### Query 3

#### Question4

**Were more mobile devices (than non-mobile devices) used to visit the store?**

Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result)

The column in interest in the question above is **deviceCategory**. null values are removed using match function and the collection is grouped by the column deviceCategory using the aggregate function **\$group**. The count of each device category is calculated using **\$sum** aggregate function.

#### Translation

Filter the dataset with devicecategory that has no null values, group by deviceCategory and count each category.

Screen Shot of MongoDB Query/Code and Results

```
> db.Group21.aggregate([
...   {
...     '$match': {
...       'deviceCategory': {
...         '$ne': null
...       }
...     }
...   }, {
...     '$group': {
...       '_id': '$deviceCategory',
...       'count': {
...         '$sum': 1
...       }
...     }
...   }
... ])
{ "_id" : " tablet", "count" : 34973 }
{ "_id" : " mobile", "count" : 262611 }
{ "_id" : " desktop", "count" : 507080 }
```

Since, the count of mobile devices '**262611**' is less than the count of desktops '**507080**', No, more mobile devices are not used to visit store.

```
db.Group21.aggregate([
... {
...   '$match': {
...     'deviceCategory': {
...       '$ne': null
...     }
...   }
... }, {
...   '$group': {
...     '_id': '$deviceCategory',
...     'count': {
...       '$sum': 1
...     }
...   }
... }
... ])
```

## Query 4

### Question1

**Which user had the maximum number of visits and when?**

Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result)

The columns in interest in the question above are **fullVisitorID**, **date** and **visitNumber**. Filtered the collection to project only fullVisitorID, date and visitNumber. Sorted the derived collection according to visitNumber in descending order.

### Translation

Filter the dataset to include fullVisitorID, Date, and visitnumber ,sort using visitnumber in descending order.

### Screen Shot of MongoDB Query/Code and Results

```
> db.Group21.find( { },
... { fullVisitorID: 1,date:1, visitNumber: 1,_id:0 } ) .sort( {visitNumber: -1}).limit(1)
{ "fullVisitorID" : "7.28E+18", "date" : "20171016", "visitNumber" : "99" }
```

User with Visitor ID: **7200000000000000000** had most number of visits- 99, on **16<sup>th</sup> October 2017**

```
db.Group21.find( { },  
  { fullVisitorId: 1 , date: 1,  
    visitNumber: 1,sessionId: 1, _id:0 } )  
  .sort( {visitNumber: -1}).limit(1)
```

## Query 5

### Question6

**How many users used only Windows devices to visit the store?**

Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result)

Columns in interest in the question above are **fullVisitorID** and **OperatingSystem**. Displayed the distinct userIDs using **distinct** function whose operatingSystem is Windows and used **.length** to find the count.

### Translation

Find distinct fullVisitorIDs that have operatingSystem as Windows and find the length.

Screen Shot of MongoDB Query/Code and Results

```
> db.Group21.distinct("fullVisitorID", {operatingSystem: " Windows"}).length
3320
```

**Answer: 3320** users used only Windows devices to visit the store.

```
> db.Group21.find({operatingSystem: " Windows"}, {fullVisitorID:1}).count()
269648
```

Windows is used **269648** times by users to visit the store.

```
db.Group21.distinct("fullVisitorID",
{operatingSystem: "
Windows"}).length
```

## Query 6

### Question7

**How many visitors had zero pageviews?**

Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result)

Columns in interest in the question above are **fullVisitorID** and **Pageviews**. Displayed the distinct userIDs using **distinct** function who has pageviews as 0 or blank and used **.length** to find the count.

### Translation

Find distinct fullVisitorIDs that have pageviews as "" and find the length.

Screen Shot of MongoDB Query/Code and Results

```
> db.Group21.distinct("fullVisitorID", {pageviews: ""}).length
89
```

**Answer:** 89 visitors had 0 pageviews.

```
> db.Group21.find({pageviews: ""}, {fullVisitorID:1}).count()
102
```

Visitors had 102 times 0 as their pageviews

```
db.Group21.distinct("fullVisitorID",
{pageviews: ""}).length
```

## Query 7

### Question8

**Which city (other than unknown) had the most number of desktop users?**

Notes/Comments About MongoDB Query/Code and Results (Include # of Rows in Result)

The columns in interest in the question above are **city** and **deviceCategory**. Since there are unwanted values in city column, they are removed by **\$nin** operator and rows with deviceCategory value as desktop are selected using the aggregate function **\$match**. The derived collection is then grouped by column city using **\$group** function and the count is found using **\$sum** function. The count is then sorted using **\$sort** function and the top result is displayed using **\$limit** function.

### Translation

Match the rows with the city values not in “not available in demo dataset”, “(not set)” and deviceCategory value as “desktop”, group by column city and find the count. Sort the result and display the top result.

### Screen Shot of MongoDB Query/Code and Results

```
> db.Group21.aggregate([
...   {
...     '$match': {
...       'city': {
...         '$nin': [
...           'not available in demo dataset', ' (not set)'
...         ]
...       },
...       'deviceCategory': ' desktop'
...     }, {
...       '$group': {
...         '_id': '$city',
...         'count_Desktops': {
...           '$sum': 1
...         }
...       }
...     }, {
...       '$sort': {
...         'count_Desktops': -1
...       }
...     }, {
...       '$limit': 1
...     }
...   ])
{ "_id" : " Mountain View", "count_Desktops" : 30712 }
```

**Mountain View** city had most number of Desktop users and the count is **30712**.



```
db.Group21.aggregate([
  {
    '$match': {
      'city': {
        '$nin': [
          ' not available in demo dataset',
          ' (not set)'
        ]
      },
      'deviceCategory': ' desktop'
    }
  }, {
    '$group': {
      '_id': '$city',
      'count_Desktops': {
        '$sum': 1
      }
    }
  }, {
    '$sort': {
      'count_Desktops': -1
    }
  }, {
    '$limit': 1
  }
])
```