1. **Give per share the average closing price, the minimum closing price, the maximum closing price and the average number of shares traded per day.**

db.bel20.aggregate([

{

$group: {

\_id: { name: "$name", date: "$date" },

average\_closing\_price: { $avg: "$price.end" },

minimum\_closing\_price: { $min: "$price.end" },

maximum\_closing\_price: { $max: "$price.end" },

average\_shares\_traded\_per\_day: { $avg: "$number" }

}

},

{

$sort: { "\_id.date": 1 } // Sort by date in ascending order

}

])

2. **Give per share the minimum closing price and the week in which this minimum closing price occurred. E.g.**

**//KBC: minprice = 39.1 ; week = 44**

**//Elia: minprice = 33.38 ; week = 46**

db.bel20.aggregate([

{ $match: { "price.end": { $exists: true } } },

{ $project: {

name: 1,

week: { $isoWeek: { $toDate: "$date" } },

closingPrice: "$price.end"

}

},

{ $group: {

\_id: { name: "$name", week: "$week" },

minClosingPrice: { $min: "$closingPrice" }

}

},

{ $sort: { "\_id.name": 1, "\_id.week": 1 } }

])

3. **Specify the number of cheeses per cheesetype. Put the cheesetype in capital letters.**

db.cheeses.aggregate([ { $group:{\_id:{$toUpper:"$properties.cheesetype"},count:{$count:{}}} } ])

4. **Give the number of cheeses per cheese factory with at least 2 variants.**

db.cheeses.aggregate([

{ $unwind: "$variants" },

{

$group: {

\_id: { factory: "$factory", cheese\_id: "$\_id" },

variants: { $push: "$variants" }

}

},

{

$group: {

\_id: "$\_id.factory",

cheeses: { $addToSet: "$\_id.cheese\_id" },

variantCount: { $sum: { $size: "$variants" } }

}

},

{ $match: { variantCount: { $gte: 2 } } },

{

$project: {

factory: "$\_id",

numberOfCheeses: { $size: "$cheeses" }

}

}

])

5. **normalize usernames: in upper case and in alphabetical order**

db.userdetails.aggregate([

{

$project: {

normalizedUserId: { $toUpper: "$user\_id" }

}

},

{

$sort: { normalizedUserId: 1 }

}

])

6. **Return the Five Most Common “Likes”**

db.users.aggregate([

{ $unwind: "$likes" },

{

$group: {

\_id: "$likes",

count: { $sum: 1 }

}

},

{ $sort: { count: -1 } },{$limit:5}])

7. **Return States with Populations above 10 Million**

db.zipcodes.aggregate([

{

$group: {

\_id: "$state",

total\_population: { $sum: "$pop" }

}

},

{

$match: {

total\_population: { $gte: 10000000 }

}

}

])

8. **Return Average City Population by State**

db.zipcodes.aggregate([ { $group: { \_id: { state: "$state", city: "$city" }, pop: { $sum: "$pop" } } }, { $group: { \_id: "$\_id.state", avgCityPop: { $avg: "$pop" } } }] )

9. **Return Largest and Smallest Cities by State (name and population)**

db.zipcodes.aggregate([

{

$group: {

\_id: "$state",

largestCity: { $max: { name: "$city", population: "$pop" } },

smallestCity: { $min: { name: "$city", population: "$pop" } }

}

},

{

$project: {

\_id: 0,

state: "$\_id",

largestCity: {

name: "$largestCity.name",

population: "$largestCity.population"

},

smallestCity: {

name: "$smallestCity.name",

population: "$smallestCity.population"

}

}

},

{

$sort: { state: 1 }

}

])

10. **Return the top 5 states with the most number of cities**

db.zipcodes.aggregate([

{

$group: {

\_id: "$state",

cityCount: { $sum: 1 }

}

},

{

$sort: { cityCount: -1 }

},

{

$limit: 5

}

])

**11.fetch the result by sorting on "education" column in ascending order**

db.userdetails.find().sort({"education":1})

**12.fetch the result by sorting on "education" column in descending order**

db.userdetails.find().sort({"education":-1})

**13.fetch the result by sorting on "education" column in ascending order and "password" column in descending order**

db.userdetails.find().sort({"education":1},{"password":-1})

**14.fetch first two documents from the collection "userdetails"**

db.userdetails.find().limit(2)

**15.fetch two documents after the first two documents from the collection 'userdetails'**

db.userdetails.find().limit(2).skip(2)

**16.fetch the two documents after the first document from the collection 'userdetails'**

db.userdetails.find().limit(2).skip(1)