**Banking Customer Analysis – MongoDB**

1. Give marketing success rate (No. of people subscribed / total no. of entries)

db.customers.aggregate([

{

$facet: {

totalCount: [{ $count: "total" }],

subscribedCount: [

{ $match: { y: "yes" } },

{ $count: "subscribed" }

]

}

},

{

$project: {

totalCount: { $arrayElemAt: ["$totalCount.total", 0] },

subscribedCount: { $arrayElemAt: ["$subscribedCount.subscribed", 0] },

successRate: {

$divide: [

{ $arrayElemAt: ["$subscribedCount.subscribed", 0] } ,

{ $arrayElemAt: ["$totalCount.total", 0] }

]

}

}

}

])

**or using Python**

list(db.customers.aggregate([

{

"$facet": {

"totalCount": [{ "$count": "total" }],

"subscribedCount": [

{ "$match": { "y": "yes" } },

{ "$count": "subscribed" }

]

}

},

{

"$project": {

"totalCount": { "$arrayElemAt": ["$totalCount.total", 0] },

"subscribedCount": { "$arrayElemAt": ["$subscribedCount.subscribed", 0] },

"successRate": {

"$divide": [

{ "$arrayElemAt": ["$subscribedCount.subscribed", 0] } ,

{ "$arrayElemAt": ["$totalCount.total", 0] }

]

}

}

}

]))

1. Give marketing failure rate

list(db.customers.aggregate([

{

"$facet": {

"totalCount": [{ "$count": "total" }],

"nosuscribedCount": [

{ "$match": { "y": "no" } },

{ "$count": "notSubscribed" }

]

}

},

{

"$project": {

"totalCount": { "$arrayElemAt": ["$totalCount.total", 0] },

"notsubscribedCount": { "$arrayElemAt": ["$nosuscribedCount.notSubscribed", 0] },

"failureRate": {

"$divide": [

{ "$arrayElemAt": ["$nosuscribedCount.notSubscribed", 0] } ,

{ "$arrayElemAt": ["$totalCount.total", 0] }

]

}

}

}

]))

1. Give the maximum, mean, median, minimum age of the average targeted customer.

db.customers.aggregate([

{$group:{\_id:null, age:{$push:"$age"}}},

{$project:{\_id:0,

avgage:{$avg:"$age"},

minage:{$min:"$age"},

maxage:{$max:"$age"},

medianage: {$median: {input:"$age",method:"approximate"}}}}

])

1. Check if age matters in marketing subscription for deposit

db.customers.aggregate([

{$match: {y:"yes"}},

{$group:{\_id:"$age",count:{$sum:1}}},

{$sort:{count:-1}}

])

1. Check if marital status mattered for a subscription to deposit

db.customers.aggregate([

{$match: {y:"yes"}},

{$group:{\_id:"$marital",count:{$sum:1}}},

{$sort:{count:-1}}

])

1. Check if age and marital status together mattered for a subscription to deposit scheme

db.customers.aggregate([

{$match: {y:"yes"}},

{$group:{\_id:{age:"$age",marital:"$marital"},count:{$sum:1}}},

{$sort:{count:-1}}

])

1. Find All Records Where Housing Loan is Approved
2. Find Records with High Account Balance
3. Count Records by Job Type

db.customers.aggregate([

{$group:{\_id:"$job",count:{$sum:1}}},

{$sort:{count:-1}}

])

1. Count the Total Married Individuals with Secondary Education who have been approached.

db.customers.aggregate([

{$match: {education:"secondary",marital:"married"}},

{$group:{\_id:{marital:"$marital",education:"$education"},count:{$sum:1}}},

{$sort:{count:-1}}

])

1. Calculate the Average Balance for Each Job Type

db.customers.aggregate([

{$group:{\_id:"$job",averageBalance:{$avg:"$balance"}}},

{$sort:{averageBalance:-1}}

])

1. Find the Most Common Education Level Among Those with Loans

db.customers.aggregate([

{$match: {loan:"yes"}},

{$group:{\_id:"$education",count:{$sum:1}}},

{$sort:{count:-1}},

{$limit:1}

])

1. Find Individuals with Multiple Campaign Contacts

db.customers.find(

{

campaign:{$gt:1}

}

).count()

1. Identify Anomalies in Account Balances

db.customers.aggregate([

{$group:{\_id:null,avgBalance:{$avg:"$balance"},

stddevBalance: {$stdDevSamp:"$balance"}}},

{$lookup:{

from:"customers",

pipeline:[{

$project:{

isAno

}

}]

}

}])

1. Analyze Seasonal Patterns

db.customers.aggregate(

{

$group:{\_id:"$month",avgDuration:{$avg:"$duration"},totalCall:{$sum:1}

}

},

{

$sort:{avgDuration:-1,totalCall:-1}

})

1. Determine Correlation Between Loan Status and Balance

db.customers.aggregate([

{

$group:{\_id:"$loan",avgbalance:{$avg:"$balance"},count:{$sum:1}

}

}])

1. Identify the Most Common Day for Successful Campaigns

db.customers.aggregate([

{$match:{y:"yes"}},

{

$group:{\_id:"$day",count:{$sum:1}

}

},

{$sort:{count:-1}},

{$limit:5}])

1. Find the Youngest Person with the Highest Account Balance

db.customers.find({},{age:1,balance:1}).sort({age:1,balance:-1}).limit(1)

1. Identify Customers with Consistently Low Balances
2. Analyze the Relationship Between Campaign Duration and Outcome

db.customers.createIndex({duration:1,y:1})

db.customers.aggregate(

{$group:{\_id:"$y",

avgduration: {$avg:"$duration"},

maxduration: {$max:"$duration"},

minduration: {$min:"$duration"}

}

}

)

1. Detect Loan Trends Among Different Job Categories

db.customers.createIndex({job:1,loan:1})  
db.customers.aggregate([

{$match:{loan:"yes"}},

{$group:{\_id:"$job",count:{$sum:1}}},

{$sort:{count:-1}}

])

– People taking more loan with less salary and low profile jobs.

1. Identify the Most Common Outcomes for Married Individuals
2. Find the Distribution of Account Balances Across Different Education Levels
3. Find ranking of Customers based on Account Balance.