**Ex.No.8 Implement an application that stores big data in Hbase /**

**MangoDB/ Pig using Hadoop / R**

**Aim**

To implement an application that stores big data in MongoDB using R.

#### Pre-Lab Discussion Theory

**MongoDB with R**

Mongodb is a NoSql database platform that works on the concept of collection and documents. Collection: Collections are just like tables in relational databases. They are a group of [Mongodb](https://www.projectpro.io/article/mongodb-projects-ideas/640) documents. These collections contain a set of documents. Document: Documents are like tuples/ rows in a relational database. R provides several libraries for creating a connection between mongodb and R such as: mongolite, Rmongo, rmongodb .

Step 1 - Install 'RMango package' install.packages("RMongo") library(RMongo)

Step 2 - Create a connection

r\_mongo\_con <- mongoDbConnect('db')

Step 3 - Check the connection

print(dbShowCollections(r\_mongo\_con)) # this verifies the established connection , returns errors if any

Step 4 - Run Queries

var\_Query <- dbGetQuery(mongo, 'collection\_name', "{'type': 'required\_data'}")

Step 5 - Install mongolite package install.packages('mongolite') library(mongolite)

Step 6 - Create a connection

mongolite\_conn <- mongo(dataset, url)

The most popular packages to connect MongoDB and R are:

[**mongolite**](https://cran.r-project.org/web/packages/mongolite/)**:** A more recent R MongoDB driver, mongolite can perform various operations like indexing, [aggregation](https://docs.mongodb.com/spark-connector/current/r/aggregation/) pipelines, TLS encryption, and SASL authentication, among others. It’s

based on the [jsonlite package for R](https://cran.r-project.org/web/packages/jsonlite/vignettes/json-aaquickstart.html) and [mongo-c-driver](http://mongoc.org/). We can install mongolite from [CRAN](https://cran.r-project.org/web/packages/mongolite/) or from RStudio ([explained in a later section](https://www.mongodb.com/languages/mongodb-and-r-example#how-to-connect-to-a-mongodb-database-in-r)).

**RMongo:** RMongo was the first R MongoDB driver with a simple R MongoDB interface. It has syntax like the MongoDB shell. RMongo has been deprecated as of now.

rmongodb: rmongodb has functions to create pipelines, handle BSON objects, etc. Its syntax is very complex compared to mongolite. Just like RMongo, rmongodb has been deprecated and is not available or maintained on CRAN.

#### **INSERTING DATA**

Let’s insert the crimes data from [data.gov](https://catalog.data.gov/dataset/crimes-2001-to-present-398a4) to MongoDB. The dataset reflects reported incidents of crime (with the exception of murders where data exists for each victim) that occurred in the City of Chicago since 2001.

### **PROGRAM**:

install.packages('gridExtra') library (ggplot2)

library (dplyr) library (maps) library (ggmap) library (mongolite) library (lubridate) library (gridExtra)

crimes=data.table::fread("crime.csv") names (crimes)

names(crimes) = gsub(" ","",names(crimes)) names(crimes)

my\_collection = mongo(collection = "crimes", db = "Chicago") # create connection, database and collection

my\_collection$insert(crimes) my\_collection$count() my\_collection$iterate()$one() length(my\_collection$distinct("PrimaryType"))

my\_collection$count('{"PrimaryType":"ASSAULT" }')

query1= my\_collection$find('{"PrimaryType" : "THEFT", "Domestic" : false }')

query2= my\_collection$find('{"PrimaryType" : "THEFT", "Domestic" : true }',fields = '{"\_id":0, "PrimaryType":1, "Domestic":1}')

ncol(query1) # with all the columns ncol(query2) # only the selected columns

domestic=my\_collection$find('{"Domestic":true}', fields = '{"\_id":0}')

domestic$Date= mdy\_hms(domestic$Date) domestic$Weekday = weekdays(domestic$Date) domestic$Hour = hour(domestic$Date) domestic$month = month(domestic$Date,label=TRUE)

plot(domestic$Date,domestic$Hour, col=domestic$month) pie(domestic)

barplot(domestic$Hour,domestic$month) plot(domestic$District,domestic$Hour) plot(domestic$District[1:1000], type="l", col="blue")

DayHourCounts = as.data.frame(table(domestic$Weekday, domestic$Hour)) DayHourCounts$Hour = as.numeric(as.character(DayHourCounts$Var2)) ggplot(DayHourCounts, aes(x=Hour, y=Freq)) + geom\_line(aes(group=Var1, color=Var1), size=1.4)+ylab("Count")+ ylab("Total Domestic Crimes")+ggtitle("Domestic Crimes in the City of Chicago Since 2001")+theme(axis.title.x=element\_text(size=14),axis.text.y = element\_text(color="blue",size=11,angle=0,hjust=1,vjust=0),axis.text.x

=element\_text(color="blue",size=11,angle=0,hjust=.5,vjust=.5), axis.title.y = element\_text(size=14),legend.title=element\_blank(),plot.title=element\_text(size=16,color=" purple",hjust=0.5))

DayHourCounts$Type = ifelse((DayHourCounts$Var1 == "Sunday") | (DayHourCounts$Var1 == "Saturday"), "Weekend", "Weekday")

ggplot(DayHourCounts, aes(x=Hour, y=Freq)) + geom\_line(aes(group=Var1, color=Type), size=2, alpha=0.5) +ylab("Total Domestic Crimes")+ggtitle("Domestic Crimes in the City of Chicago Since 2001")+theme(axis.title.x=element\_text(size=14),axis.text.y = element\_text(color="blue",size=11,angle=0,hjust=1,vjust=0),axis.text.x = element\_text(color="blue",size=11,angle=0,hjust=.5,vjust=.5), axis.title.y = element\_text(size=14),legend.title=element\_blank(),plot.title=element\_text(size=16,color=" purple",hjust=0.5))

DayHourCounts$Var1 = factor(DayHourCounts$Var1, ordered=TRUE,levels=c("Monday",

"Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday")) ggplot(DayHourCounts, aes(x = Hour, y = Var1)) + geom\_tile(aes(fill = Freq)) + scale\_fill\_gradient(name="Total MV Thefts", low="white", high="red") +ggtitle("Domestic Crimes in the City of Chicago Since 2001")+theme(axis.title.y = element\_blank())+ylab("")+theme(axis.title.x=element\_text(size=14),axis.text.y = element\_text(size=13),axis.text.x = element\_text(size=13), axis.title.y = element\_text(size=14),legend.title=element\_blank(),plot.title=element\_text(size=16,color=" purple"))

domestic=my\_collection$find('{"Domestic":true}', fields ='{"\_id":0, "Domestic":1,"Date":1}')

domestic$Date= mdy\_hms(domestic$Date) domestic$Weekday = weekdays(domestic$Date) domestic$Hour = hour(domestic$Date) domestic$month = month(domestic$Date,label=TRUE)

WeekdayCounts = as.data.frame(table(domestic$Weekday))

WeekdayCounts$Var1 = factor(WeekdayCounts$Var1, ordered=TRUE, levels=c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday","Saturday")) ggplot(WeekdayCounts,aes(x=Var1, y=Freq))+geom\_line(aes(group=1),size=2,color="red")

+ xlab("Day of the Week") + ylab("Total Domestic Crimes")+ ggtitle("Domestic Crimes in the City of Chicago Since 2001")+ theme(axis.title.x=element\_blank(),axis.text.y = element\_text(color="blue",size=11,angle=0,hjust=1,vjust=0),axis.text.x = element\_text(color="blue",size=11,angle=0,hjust=.5,vjust=.5), axis.title.y = element\_text(size=14), plot.title=element\_text(size=16,color="purple",hjust=0.5)) ASSAULT=my\_collection$count('{"PrimaryType":"ASSAULT", "Domestic" : true }') my\_collection$aggregate('[{"$group":{"\_id":"$LocationDescription","Count":{"$sum":1}}} ')%>%na.omit()%>%arrange(desc(count))%>%head(10)%>% ggplot(aes(x=reorder(`\_id`,count),y=count))+geom\_bar(stat="identity",color='skyblue',fill=' #b35900')+geom\_text(aes(label count), color = "blue") +coord\_flip()+xlab("Location Description")

|  |  |  |
| --- | --- | --- |
| **OUTPUT:** |  | |
| * names (crimes) |
| [1] "ID" | "Case Number" | "Date" |
| [4] "Block" | "IUCR" | "Primary Type" |

[7] "Description" "Location Description" "Arrest"

[10] "Domestic" "Beat" "District"

[13] "Ward" "Community Area" "FBI Code"

[16] "X Coordinate" "Y Coordinate" "Year"

[19] "Updated On" "Latitude" "Longitude"

[22] "Location"

* names(crimes) = gsub(" ","",names(crimes))
* names(crimes)

[1] "ID" "CaseNumber" "Date" "Block"

[5] "IUCR" "PrimaryType" "Description" "LocationDescription"

[9] "Arrest" "Domestic" "Beat" "District"

[13] "Ward" "CommunityArea" "FBICode" "XCoordinate"

[17] "YCoordinate" "Year" "UpdatedOn" "Latitude"

[21] "Longitude" "Location"

* my\_collection = mongo(collection = "crimes", db = "Chicago") # create connection, database and collection
* my\_collection$insert(crimes) List of 5

$ nInserted : num 7750924

$ nMatched : num 0

$ nRemoved : num 0

$ nUpserted : num 0

$ writeErrors: list()

* my\_collection$count()

[1] 7750924

* my\_collection$iterate()$one()

$ID

[1] 10224738

$CaseNumber

[1] "HY411648"

$Date

[1] "09/05/2015 01:30:00 PM"

$Block

[1] "043XX S WOOD ST"

$IUCR

[1] "0486"

$PrimaryType

[1] "BATTERY"

$Description

[1] "DOMESTIC BATTERY SIMPLE"

$LocationDescription

[1] "RESIDENCE"

$Arrest

[1] FALSE

$Domestic

[1] TRUE

$Beat

[1] 924

$District

[1] 9

$Ward

[1] 12

$CommunityArea

[1] 61

$FBICode

[1] "08B"

$XCoordinate

[1] 1165074

$YCoordinate

[1] 1875917

$Year

[1] 2015

$UpdatedOn

[1] "02/10/2018 03:50:01 PM"

$Latitude

[1] 41.81512

$Longitude

[1] -87.67

$Location

[1] "(41.815117282, -87.669999562)"

* length(my\_collection$distinct("PrimaryType"))

[1] 36

* my\_collection$count('{"PrimaryType":"ASSAULT" }')

[1] 504447

* query1= my\_collection$find('{"PrimaryType" : "THEFT", "Domestic" : false }')
* query2= my\_collection$find('{"PrimaryType" : "THEFT", "Domestic" : true }',

+ fields = '{"\_id":0, "PrimaryType":1, "Domestic":1}')

* ncol(query1) # with all the columns

[1] 22

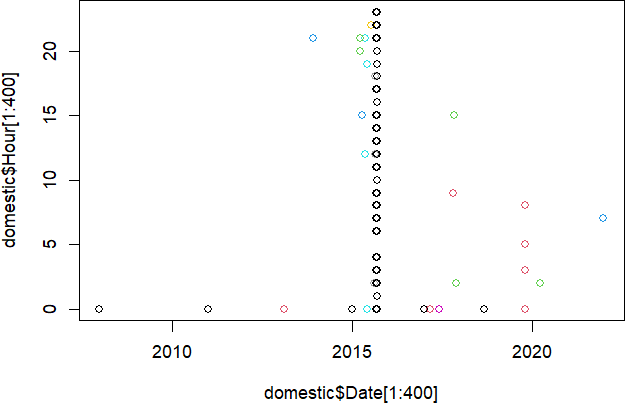
* ncol(query2) # only the selected columns

[1] 2

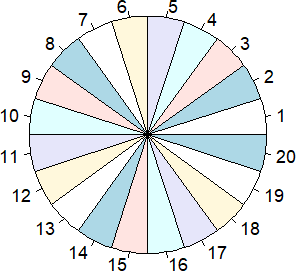
* domestic=my\_collection$find('{"Domestic":true}',

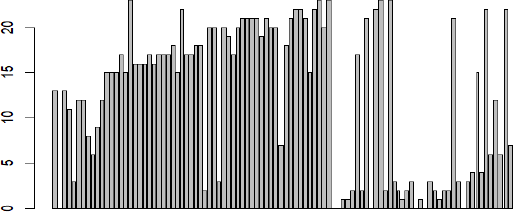
+ fields = '{"\_id":0}')

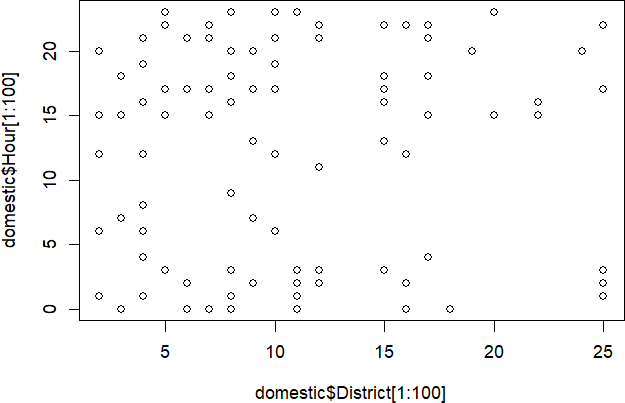
* domestic$Date= mdy\_hms(domestic$Date)
* domestic$Weekday = weekdays(domestic$Date)
* domestic$Hour = hour(domestic$Date)
* domestic$month = month(domestic$Date,label=TRUE)
* domestic$month = month(domestic$Date)
* plot(domestic$Date[1:400],domestic$Hour[1:400], col=domestic$month)

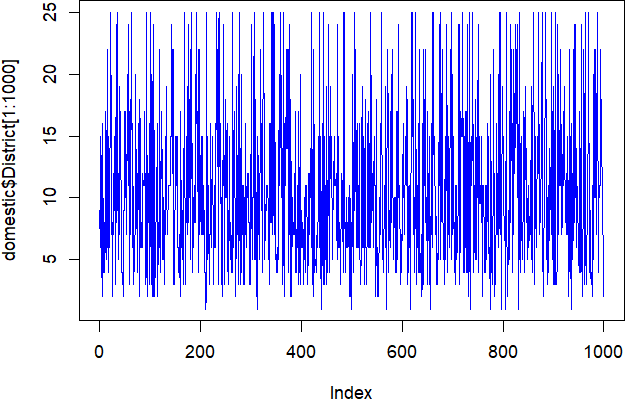


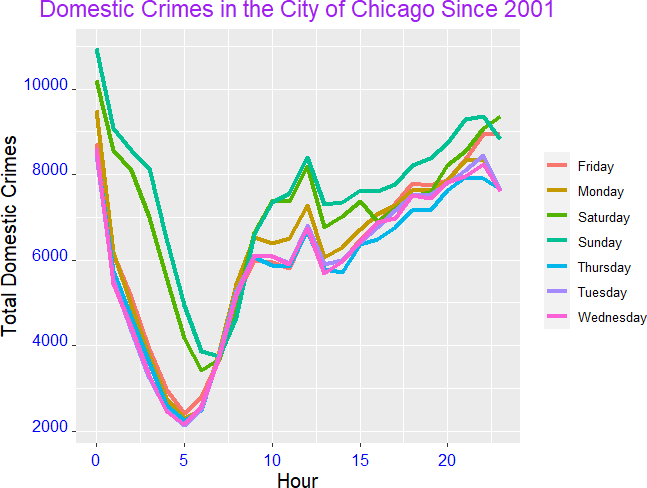
pie(domestic$Year[1:20])

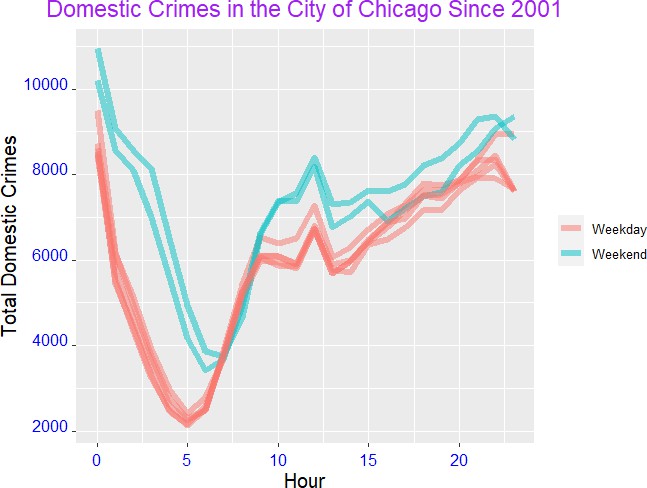


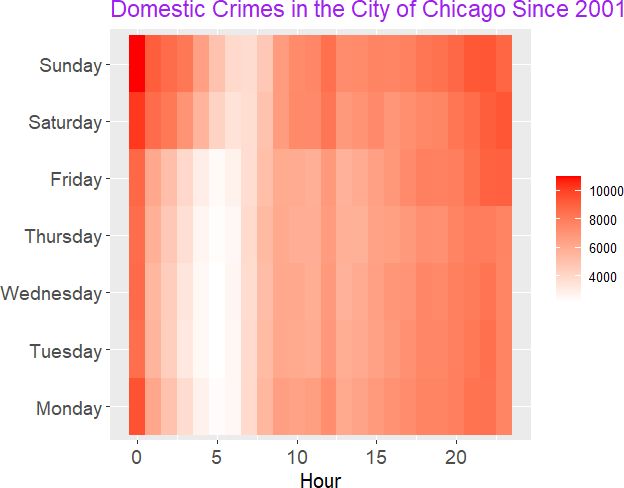


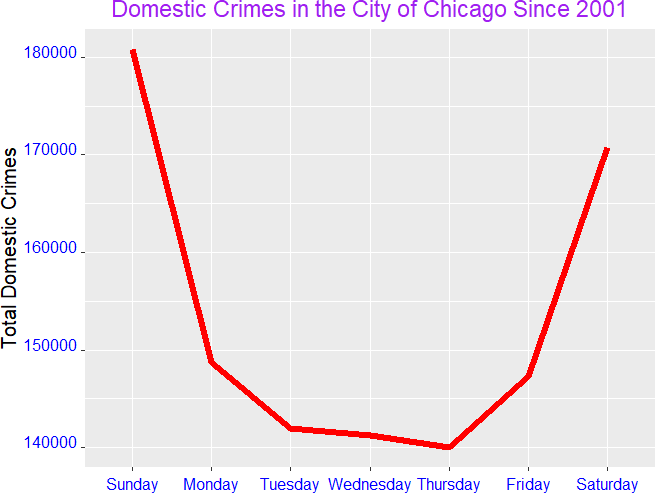












#### **RESULT**

Thus, the application of Crime data set that stores big data in MongoDB using R was executed successfully.