DaBA

Keith Goh

October 22, 2019

1 SQL

```
SELECT c1,c2 from t;
```

Query Data in columns c1 and c3 from table t

```
SELECT * from t
```

Query all rows and columns from from table

```
SELECT c1,c2 from t;
Where condition;
```

Query data and filter rows with a condition

```
SELECT c1,c2 from t;
order By c1(DESC/ASC)

SELECT column_name AS alias_name
FROM table_name;
SELECT OrderID, Quantity,
CASE
    WHEN Quantity $>$ 30 THEN "The quantity is greater than 30"
    WHEN Quantity = 30 THEN "The quantity is 30"
    ELSE "The quantity is under 30"
END AS QuantityText
FROM OrderDetails;
```

(creates 3 columns)

```
SELECT IFNULL(NULL, "W3Schools.com");
```

replace null with W3schools

```
SELECT Artist.Name,
InvoiceLine.UnitPrice * InvoiceLine.Quantity AS TrackSales
FROM ((InvoiceLine INNER JOIN Track ON
InvoiceLine.TrackId = Track.TrackId )
INNER JOIN Album ON Track.AlbumId = Album.AlbumId)
INNER JOIN Artist ON Album.ArtistId=Artist.ArtistId
```

```
Create Table Revenue class as select * from t1
```

create new table revenue class from t1 all columns

```
Delete from track where genreid=20
```

```
UPDATE t
SET c1=newvalue
c2=newvalue2
where condition;
updates values in the column c1,c2 that matches the condition.
Select C1,aggregate(c2)
From t
Group By c1
having conditions
```

Group rows using an aggregate function,eg Count(),Sum(),Avg(),Min().Max()
filter groups using having clause
The HAVING clause was added to SQL because the WHERE keyword could not be used with aggregate functions.
EG.

```
SELECT Student, SUM(score) AS total FROM Marks GROUP BY Student
HAVING total > 70
```

LEFT JOIN: Return all records from the left table, and the matched records from the right table
The GROUP BY statement groups rows that have the same values into summary rows, like "find the number of customers in each country".

2 Throughput

Throughput of a multi-stage process (sometimes called "Capacity") is the lowest throughput (rate) among all the stages
For Parallel activities,The overall throughput is the minimum throughput among all the parallel activities.
Throughput for Multiple Paths,denoted with a diamond when split, find the throughput of slowest. if it is determined that there is a fixed split, if not we assign the calculate add the total rate of the work.

3 R

the [1] refers to the index of its element
rm(input)=remove the input
str() is to see the structure of the input
c() concatenate function to add variables together to form a vector or vectors together to form a longer one.
1:4 works in r to create a list form 1 to 4
use arrow function in R

```
here <- function(x,y){
  x+y
}

a  <- 1:10/5
> a
 [1] 0.2 0.4 0.6 0.8 1.0 1.2 1.4 1.6 1.8 2.0

 if ( test_expression1) {
 statement1
 } else if ( test_expression2) {
 statement2
 } else if ( test_expression3) {
 statement3
 } else {
 statement4
 }
```

default values work

```
model <- Taste ~ 0+acetic
result <- lm(model,ccdata)
```

use this to declare no constant term

```
fit <- result$fitted.values
taste <- ccdata$Taste
r1 <- c(0,max(ccdata$Taste$))
lines(r1,r1)
title(’actual vs. fitted values’)
residuals <- taste-fit
plot(taste,residuals)
lines(r1,c(0,0))
lines(lowess(taste,residuals,f=0.8),col=c(’red’))
title(’residuals vs actual values’)
```

AIC the lower the better

```
admitmodel <- admit ~ gre+gpa
admitresult <- glm(admitmodel,family=binomial,data=graddata)
```

$$p = \frac{1}{1 + e^{-(-4.949378 + 0.002691 \times gre + 0.754687 \times gpa)}}$$

for categorical data we need to use

```
  graddata$school_rank <- factor(graddata$school_rank)
```

for counts/proportion

```
  ingotdata$frac_not_ready <- ingotdata$Num_Not_ready/ingotdata$Num_ingots
  ingotmodel<-frac_not_ready ~ soak+heat
  #use a generalized linear model in the binomial family

  ingotresult <- glm(ingotmodel,family=binomial,data=ingotdata,weight=Num_ingots)
  summary(ingotresult)
```

Dependent variable must be a number between 0 and 1
we must specify the ni counts which is the weight
use AIC to pick best model

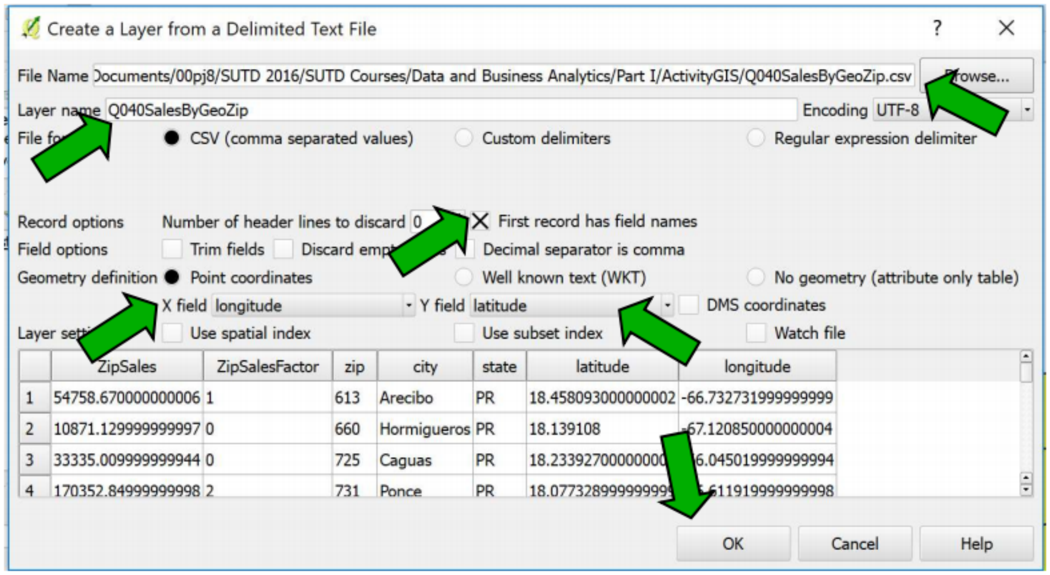
```
setwd ('C:\\Users\\silentfatez\\Downloads')
creditdata <- read.csv("CreditData.csv")
head (creditdata)
plot(creditdata[,c(5,9:14)])
cor(creditdata[,c(5,9:14)])

install.packages("tree")
library(tree)

tree.credit=tree(Status~.,data=creditdata)
summary(tree.credit)
plot(tree.credit)
text(tree.credit,pretty=0)
```

Top part is for corr data, close to -1 and 1 means its correlated,closer to 0 means its not.
Connect R to SQL

```
getextract <- function(query){
dbname <- ’eg.sqlite’
conn <- dbconnect(SQLITE(),dbname)
queryresult <- dbGetquery(conn,query)
dbDisconnect(conn)
queryresult
}
getextract("SELECT FROM NEWEXTRACT Limit 20")
```

4 GIS

GIS rmb longtitude is like length so x then latitude is like high so y



Create a Layer from a Delimited Text File

File Name: Documents/00pj8/SUTD 2016/SUTD Courses/Data and Business Analytics/Part I/ActivityGIS/Q040SalesByGeoZip.csv

Layer name: Q040SalesByGeoZip

Encoding: UTF-8

File fo: ● CSV (comma separated values) ○ Custom delimiters ○ Regular expression delimiter

Record options: Number of header lines to discard 0 ☒ First record has field names

Field options: □ Trim fields □ Discard emp □ Decimal separator is comma

Geometry definition: ● Point coordinates ○ Well known text (WKT) ○ No geometry (attribute only table)

X field: longitude Y field: latitude □ DMS coordinates

Layer sett: □ Use spatial index □ Use subset index □ Watch file

	ZipSales	ZipSalesFactor	zip	city	state	latitude	longitude
1	54758.670000000006	1	613	Arecibo	PR	18.458093000000002	-66.732731999999999
2	10871.129999999997	0	660	Hormigueros	PR	18.139108	67.120850000000004
3	33335.009999999944	0	725	Caguas	PR	18.23392700000000	6.0450199999999994
4	170352.84999999998	2	731	Ponce	PR	18.07732899999999	611919999999998

Select ‘Graduated’ style

Select ‘ZipSalesFactor’ column

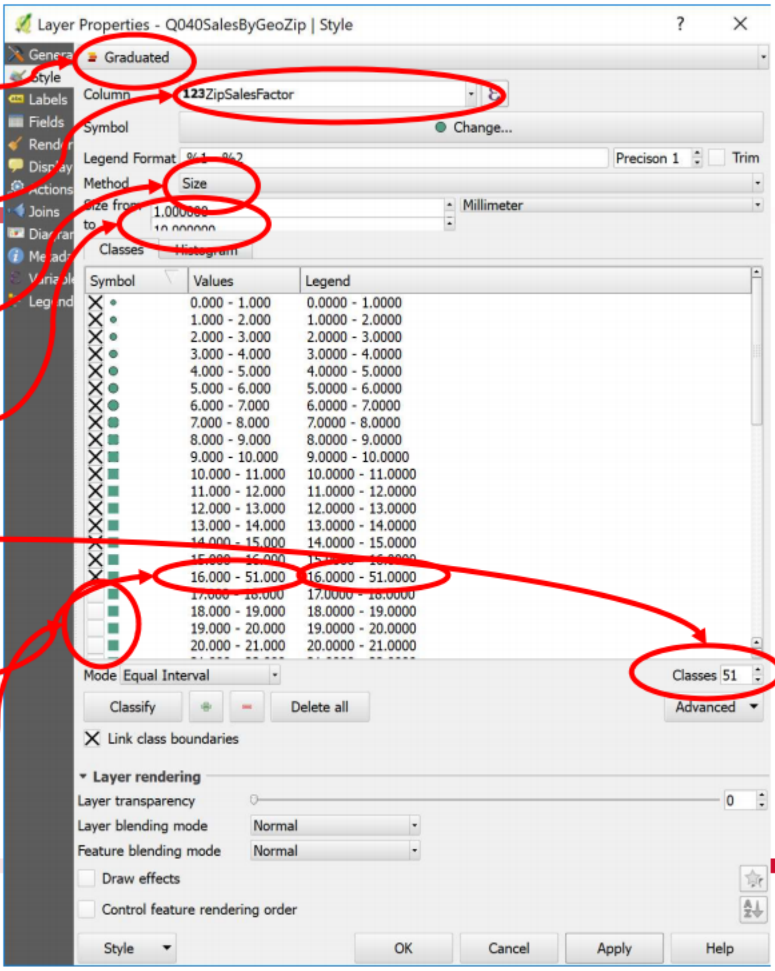
Select ‘Size’ method

Choose range of symbol sizes

Choose number of classes to generate

Adjust range and legend of class 16

Deselect classes above 16



Layer Properties - Q040SalesByGeoZip | Style

Graduated

Column: 123ZipSalesFactor

Symbol: ● Change...

Legend Format: %1 - %2 Precison 1 □ Trim

Method: Size

Size from: 1.000000 Millimeter

to: 10.000000

Symbol	Values	Legend
☒ ●	0.000 - 1.000	0.0000 - 1.0000
☒ ●	1.000 - 2.000	1.0000 - 2.0000
☒ ●	2.000 - 3.000	2.0000 - 3.0000
☒ ●	3.000 - 4.000	3.0000 - 4.0000
☒ ●	4.000 - 5.000	4.0000 - 5.0000
☒ ●	5.000 - 6.000	5.0000 - 6.0000
☒ ●	6.000 - 7.000	6.0000 - 7.0000
☒ ●	7.000 - 8.000	7.0000 - 8.0000
☒ ■	8.000 - 9.000	8.0000 - 9.0000
☒ ■	9.000 - 10.000	9.0000 - 10.0000
☒ ■	10.000 - 11.000	10.0000 - 11.0000
☒ ■	11.000 - 12.000	11.0000 - 12.0000
☒ ■	12.000 - 13.000	12.0000 - 13.0000
☒ ■	13.000 - 14.000	13.0000 - 14.0000
☒ ■	14.000 - 15.000	14.0000 - 15.0000
☒ ■	15.000 - 16.000	15.0000 - 16.0000
☒ ■	16.000 - 51.000	16.0000 - 51.0000
☐ ■	17.000 - 18.000	17.0000 - 18.0000
☐ ■	18.000 - 19.000	18.0000 - 19.0000
☐ ■	19.000 - 20.000	19.0000 - 20.0000
☐ ■	20.000 - 21.000	20.0000 - 21.0000

Mode: Equal Interval

Classes: 51

☒ Link class boundaries

Layer rendering

Layer transparency: 0

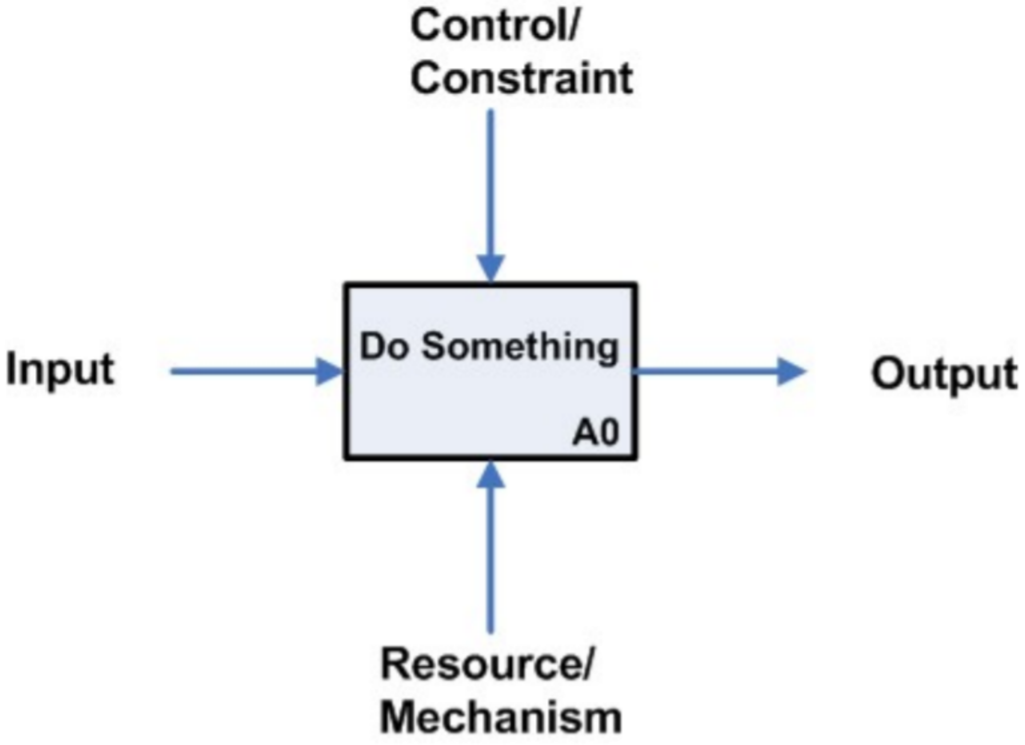
Layer blending mode: Normal

Feature blending mode: Normal

☐ Draw effects

☐ Control feature rendering order

5 functional modelling



6 Timeseries

```
hw <- HoltWinters(AirPassengers) #Holt-Winter
hw <- HoltWinters(AirPassengers,gamma=False) #Holt2
hw <- HoltWinters(AirPassenger,beta=False, gamma=False)#Holt1
```

Holt 2 is to capture changing trend like a down curve
Holt 1 is to consider newer months more
Moving average has no trend, takes evenly from a couple of points.

```
tsactuals <- ts(actuals,frequency=12) we need to convert actuals vector to time series spe
hw <- Holtwinters(tsactuals)
hwfitted <- fitted(hw)[,1]
#get only first column
forecasts <- round(as.numeric(fitted(hw)[,1]))

#calculate WAPE
actualsnew <- tail(actuals,length(forecasts))
errors <- forecast-actualsnew
abserrors <- abs(errors)
totalerror <- sum(abserrors)
if (totalactual=0){
relativeabserror<-totalerror/totalactual
}else{
relativeabserror <- 0
}
Wape <- relativeabsoerror*100
```

7 Credits

http://www.sqltutorial.org/sql-cheat-sheet/
https://www.w3schools.com/sql