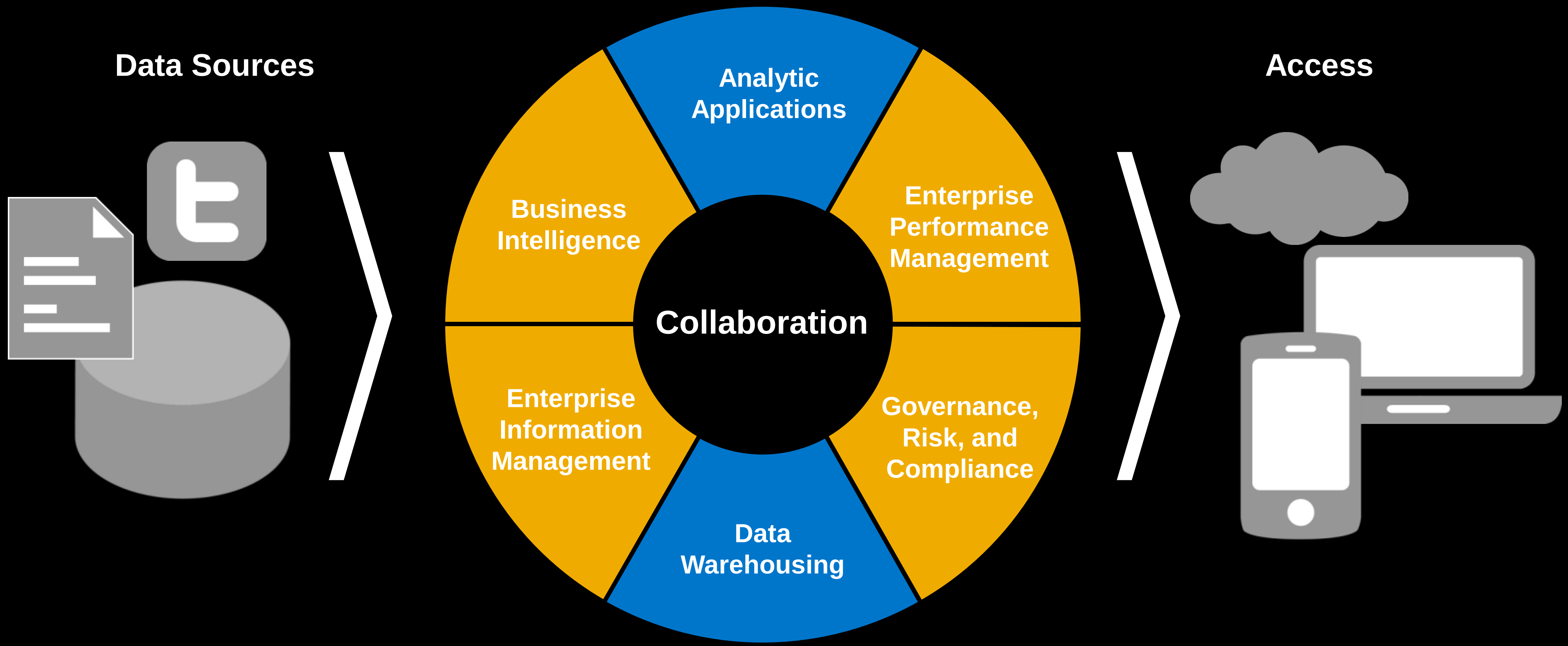
6693

Advanced Analytics with R and SAP HANA

Jitender Aswani and Jens Doerpmund

LETS TALK CODE

Analytics from SAP
Four Topic Areas for DKOM



Data Sources

Analytic Applications

Business Intelligence

Enterprise Performance Management

Collaboration

Enterprise Information Management

Governance, Risk, and Compliance

Data Warehousing

Access

Business Intelligence
Innovate. Accelerate. Simplify.

Reporting and Analysis

Data Discovery

Dashboards and Applications

Deployment Options

BI Platform | **SAP HANA** | **Embedded** | **Cloud**

Code Agenda

- ✓ R and HANA – Why choose this topic for DKOM?
- ✓ Basic R
- ✓ Advanced R
- ✓ R and HANA Integration – A match made in tech wonder-world
- ✓ R, HANA, XS Engine, JSON/HTML5
- ✓ Airlines App Demo

R and HANA:
Why choose this topic for DKOM?

- ✓ **Free, Fun and Fantastic**
- ✓ **Unbelievably simple yet amazingly powerful**
- ✓ **Unparalleled platform for analytical and statistical programming**
- ✓ **Vibrant community – 3,670 packages contributed**
- ✓ **In-memory Technology**
- ✓ **Excellent support in SAP HANA for advanced analytics on big-data**

Basic R:
Setup and Explore Preloaded Data Sets

- ✓ Setup is trivial
  - ✓ Download R (http://www.r-project.org/)

    And / Or
  - ✓ Download RStudio (http://rstudio.org/)
- ✓ Run RStudio
- ✓ Start play with preloaded datasets
  - ✓ library(help="datasets")
- ✓ Type *cars*
  - ✓ dataset with 50 records on two columns (speed and dist)
- ✓ > *?cars*
  - ✓ prints more information on cars dataset
- ✓ > *names(cars)*
  - ✓ Print column names
- ✓ *cars$speed*
  - ✓ Access speed column of cars database
- ✓ > *mycars <- cars*
  - ✓ Copy the cars dataset over to mycars
- ✓ > *mycars$product <- mycars$speed * mycars$dist*
  - ✓ Add a new column to mycars dataset
- ✓ > *names(mycars)* or just type > *mycars*
- ✓ > *Titanic*

Basic R:
Visualization in R

✓ Plots scatterplot (correlation – higher the speed, more stopping distance is required.)

```
> plot (cars)
```

✓ Line chart

```
> lines(cars)
```

✓ Let’s increase the complexity…

✓ Motor Trend Car Road Tests (1974)

```
> mtcars
```

✓ Structure of mtcars

```
str(mtcars)
```

✓
```
> plot(mtcars$wt, mtcars$mpg)
```

✓ Plot bar chart on gears column

```
counts <- table(mtcars$gear)
barplot(counts, main="Car Distribution", xlab="Number
 of Gears")
```

✓ Let’s make a colorful stacked bar chart

```
counts <- table(mtcars$vs, mtcars$gear)
barplot(counts, main="Car Distribution by Gears and
 VS",
  xlab="Number of Gears", col=c("darkblue","red"),
  legend = rownames(counts))
```

✓ Increase the complexity of your visualization

```
coplot(mpg ~ disp | as.factor(cyl), data = mtcars, panel
 = panel.smooth, rows = 1)
```

✓ Glance over your data before you start analyzing

```
pairs(mtcars, main = "mtcars data")
```

Basic R (Visualization)

- ✓ Boxplot of tooth growth in guineapigs using ToothGrowth dataset

  *boxplot(len~supp*dose, data=ToothGrowth, notch=FALSE, col=(c("gold","darkgreen")),main="Tooth Growth", xlab="Suppliment and Dose")*

We have barely scratched the surface on the topic of visualization in R….

Basic R: data.frame and data.table

- ✓ data.frame
  - ✓ Single most important data type - a columnar data-set
- ✓ data.table (Likely replacement for plyr package)
  - ✓ A very powerful pacakage that extends data.frame and simplifies data massaging and aggregation
  - ✓ You just need to learn these two, data.frame and data.table, to get most of your analytics tasks done
  - ✓ *install.packages(“data.table”) to install*
  - ✓ *library(“data.table”)*

- ✓ *> str(mtcars) or class(mtcars)*
- ✓ Create a simple data frame

```
dkom <- data.frame(Age=c(23,24,25,26),
NetWorth=c(100,200,300,400), Names=c("A", "B", "C",
"D"))
```

- ✓ Load a csv file into data.frame/data.table

```
setwd("C:/Users/i827456/Desktop/DKOM-2012/data")

allairports <- data.table(read.csv("airlines/airports.csv",
header=TRUE, sep=",", stringsAsFactors=FALSE))

allairports
```

- ✓ Write the data.frame to a csv file

```
write.csv(allairports , “output/dkomairports.csv",
row.names=FALSE)
```

Advanced R:
SP100 - XML Parsing and Historical Stock Data

```
SAP.Monthly.Data <- to.monthly(getSymbols("SAP", auto.assign=FALSE, from="2007-01-01"))
plot(SAP.Monthly.Data) # Plot monthly stock close since 2007
plot(OpCl(SAP.Monthly.Data)*100) # Plot monthly returns
```

Perform this analysis on all SP100 components

1

```
library(XML) # For XML Parsing
library(plyr) # For data.frame aggregating
library(quantmod) #For downloading financial information
from Yahoo, Google
# get the list of symbols from Wikipedia – View the wikipedia
to get an understanding
sp100.list <-
readHTMLTable('http://en.wikipedia.org/wiki/S%26P_100')[[2]
]
sp100.list <- as.vector(sp100.list$Symbol)
```

2

```
#Get monthly returns
getMonthlyReturns <- function(sym) {
    y <- to.monthly(getSymbols('YHOO', auto.assign=FALSE,
from="2007-01-01"))
    as.vector(OpCl(y)*100)
}
SP100.MR <- ldply(sp100.list, getMonthlyReturns,
.progress="text")
```

3

```
#transpose d to have tickers as columns and date as rows
SP100.MR.T <- t(SP100.MR)
colnames(SP100.MR.T)<- sp100.list
rownames(SP100.MR.T) <- seq(as.Date("2007-01-01"),
today, by="mon")
```

Advanced R:
Geo Code Your Data – Google Maps API

Geo-code an Address – Get Lat/Lng

```
getGeoCode <- function(gcStr)  {
  library("RJSONIO") #Load Library
  gcStr <- gsub(' ','%20',gcStr) #Encode URL Parameters
 #Open Connection
 connectStr <- 
paste('http://maps.google.com/maps/api/geocode/json?sensor=fa
lse&address=',gcStr, sep="") 
  con <- url(connectStr)
  data.json <- fromJSON(paste(readLines(con), collapse=""))
  close(con)
  #Flatten the received JSON
  data.json <- unlist(data.json)
  if(data.json["status"]=="OK")   {
    lat <- data.json["results.geometry.location.lat"]
    lng <- data.json["results.geometry.location.lng"]
    gcodes <- c(lat, lng)
    names(gcodes) <- c("Lat", "Lng")
    return (gcodes)
  }
}
geoCodes <- getGeoCode("Palo Alto,California")
```

Reverse Geo-code – Get Address

```
reverseGeoCode <- function(latlng) {
latlngStr <-  gsub(' ','%20', paste(latlng, collapse=","))#Collapse and 
Encode URL Parameters
  library("RJSONIO") #Load Library
  #Open Connection
  connectStr <- 
paste('http://maps.google.com/maps/api/geocode/json?sensor=false
&latlng=',latlngStr, sep="")
  con <- url(connectStr)
  data.json <- fromJSON(paste(readLines(con), collapse=""))
  close(con)
  #Flatten the received JSON
  data.json <- unlist(data.json)
  if(data.json["status"]=="OK")
    address <- data.json["results.formatted_address"]
  return (address)
}

address <- reverseGeoCode(c(37.4418834, -122.1430195))
```

```
airports <- with(airports, data.frame(Code, Adr,
    c(Lat,Lng)=laply(Adr, function(val){getGeoCode(val)})))
```

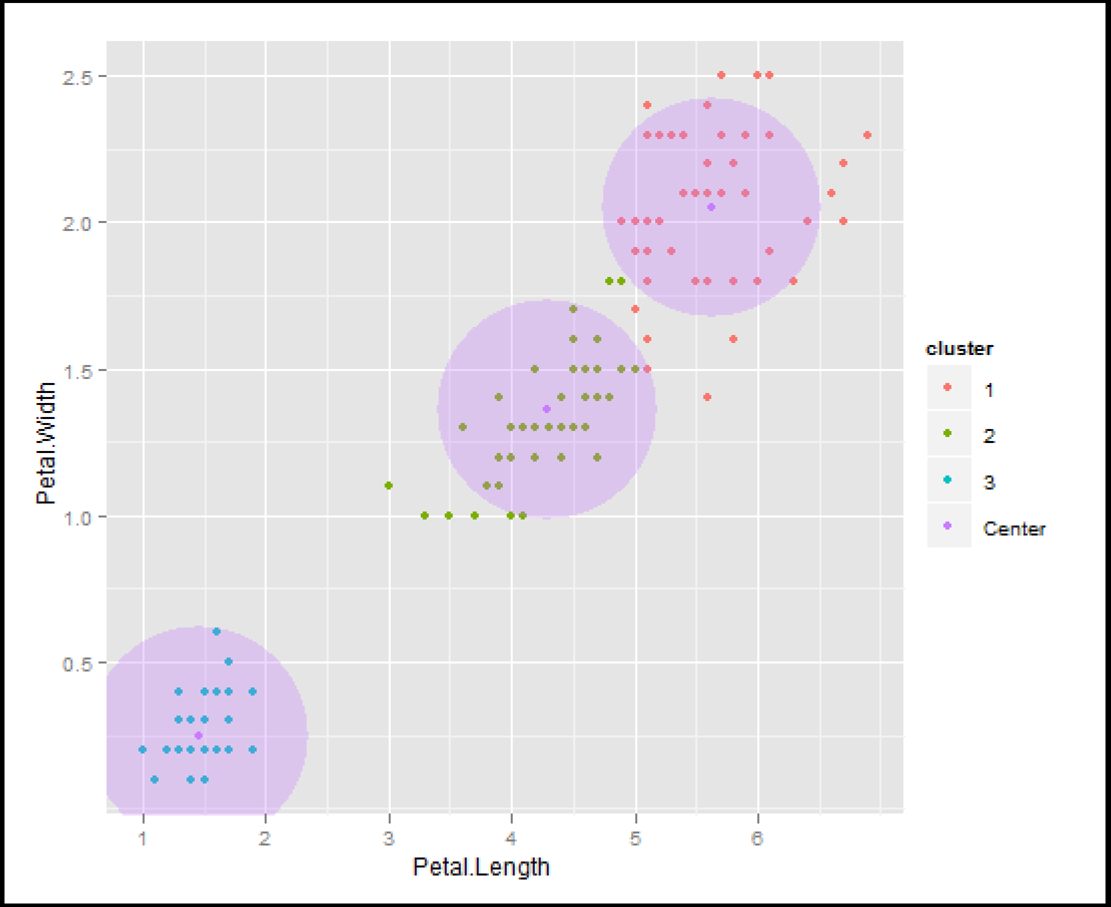
Advanced R:
Cluster Analysis using K-Mean

#Perfrom K-Mean Clustering on IRIS dataset

```
irisdf <- iris
m <- as.matrix(cbind(irisdf$Petal.Length,
                irisdf$Petal.Width),ncol=2)
kcl <- (kmeans(m,3))
kcl$size
kcl$withinss
irisdf$cluster <- factor(kcl$cluster)
centers <- as.data.frame(kcl$centers)
```

#Plot clusters and data in the clusters

```
library(ggplot2)
ggplot(data=irisdf, aes(x=Petal.Length, y=Petal.Width,
color=cluster )) +
 geom_point() +
 geom_point(data=centers, aes(x=V1,y=V2, color='Center'))
+ geom_point(data=centers, aes(x=V1,y=V2,
color='Center'), size=52, alpha=.3, legend=FALSE)
```

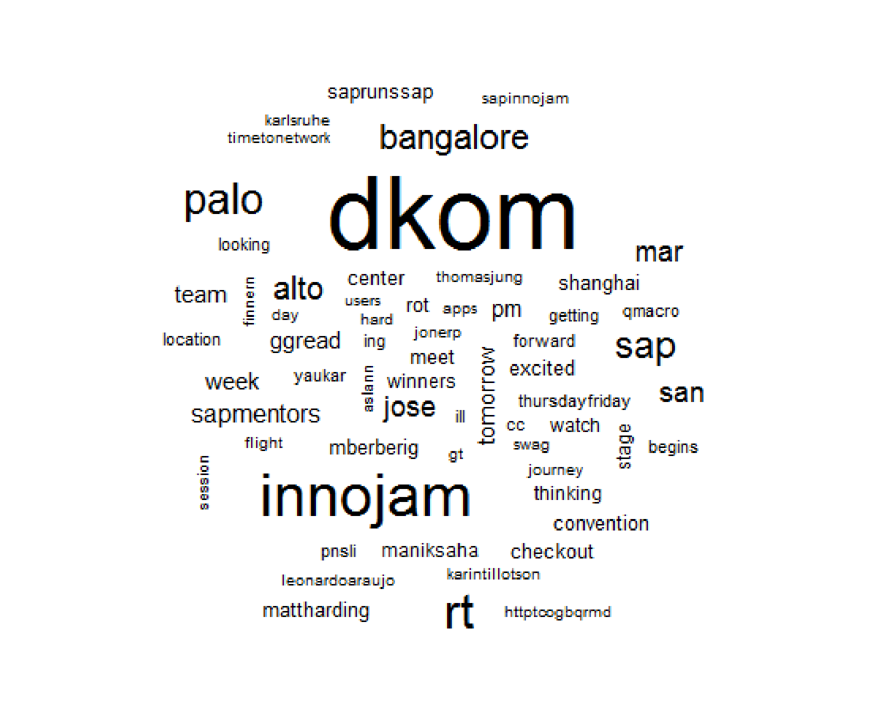


Source: R-Chart

Advanced R:
Sentiment Analysis on #DKOM and a WordCloud

TwitterTag	TotalTweetsFetched	PositiveTweets	NegativeTweets	AverageScore	TotalTweets	Sentiment
#DKOM	64	19	9	0	28	0.68

```
#Populate the list of sentiment words from Hu and Liu (http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html)
huliu.pwords <- scan('opinion-lexicon/positive-words.txt', what='character', comment.char=';')
huliu.nwords <- scan('opinion-lexicon/negative-words.txt', what='character', comment.char=';')

# Add some words
huliu.nwords <- c(huliu.nwords,'wtf','wait','waiting','epicfail', 'crash', 'bug', 'bugy', 'bugs', 'slow', 'lie')
#Remove some words
huliu.nwords <- huliu.nwords[!huliu.nwords=='sap']
huliu.nwords <- huliu.nwords[!huliu.nwords=='cloud']
#which('sap' %in% huliu.nwords)

twitterTag <- “#DKOM"
# Get 1500 tweets - an individual is only allowed to get 1500 tweets
 tweets <- searchTwitter(tag, n=1500)
  tweets.text <- laply(tweets,function(t)t$getText())
  sentimentScoreDF <- getSentimentScore(tweets.text)
  sentimentScoreDF$TwitterTag <- twitterTag

# Get rid of tweets that have zero score and seperate +ve from -ve tweets
sentimentScoreDF$posTweets <- as.numeric(sentimentScoreDF$SentimentScore >=1)
sentimentScoreDF$negTweets <- as.numeric(sentimentScoreDF$SentimentScore <=-1)

#Summarize finidings
summaryDF <- ddply(sentimentScoreDF,"TwitterTag", summarise,
                TotalTweetsFetched=length(SentimentScore),
                PositiveTweets=sum(posTweets), NegativeTweets=sum(negTweets),
                AverageScore=round(mean(SentimentScore),3))

summaryDF$TotalTweets <- summaryDF$PositiveTweets + summaryDF$NegativeTweets

#Get Sentiment Score
summaryDF$Sentiment  <- round(summaryDF$PositiveTweets/summaryDF$TotalTweets, 2)
```



Blog: Big Four and the Battle of Sentiments - Oracle, IBM, Microsoft and SAP

Source: R-Bloggers

Advanced R:
Visualization Using Google Visualizations Libraries

```
library("googleVis")
data(Fruits)
motionChart <- gvisMotionChart(Fruits, idvar="Fruit", timevar="Year")
plot(motionChart)
```

Again, we have barely scratched the surface on the advanced capabilities of R…

R – Let’s recap

- ✓ Basic data munging and visualization
- ✓ Advanced data munging, mash-ups, aggregation, visualization, sentiment analysis and more…
- ✓ There is a package for everything you can imagine in R community, if not, go and create one
- ✓ We are just getting warmed up!!!
- ✓ Please learn more about R at www.r-project.org

wordcloud XML

Data.table GoogleVis

quantmod

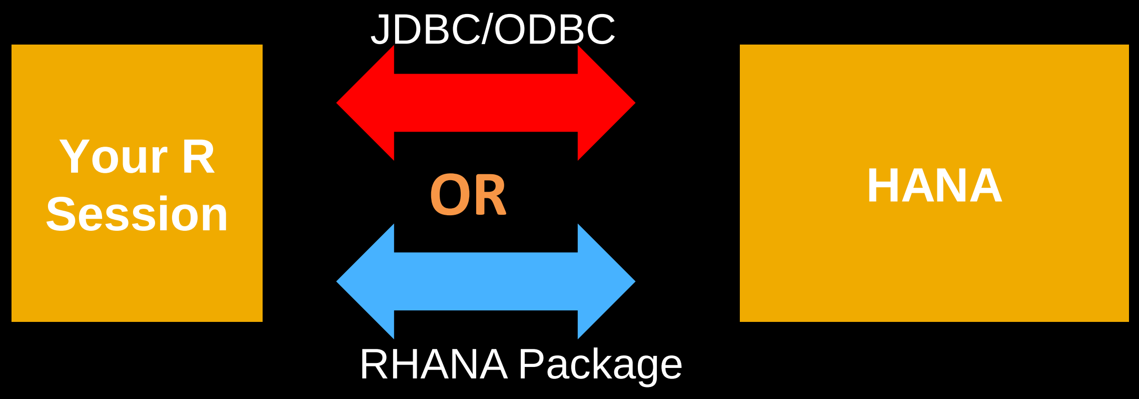
twitteR RJSONIO plyr

R and HANA Integration – Big Data Setup

- ✓ Airlines sector in the travel industry
- ✓ 22 years (1987-2008) of airlines on time performance data on US airlines
- ✓ 123 million records
- ✓ Extract Transform Load – ETL work to combine this data with data on airports, data on carriers with this data to setup for Big Data analysis in R and HANA
- ✓ D20 with 96GB of RAM and 24 Cores
- ✓ Massive amount of data crunching using R and HANA
- ✓ Let’s dive in…

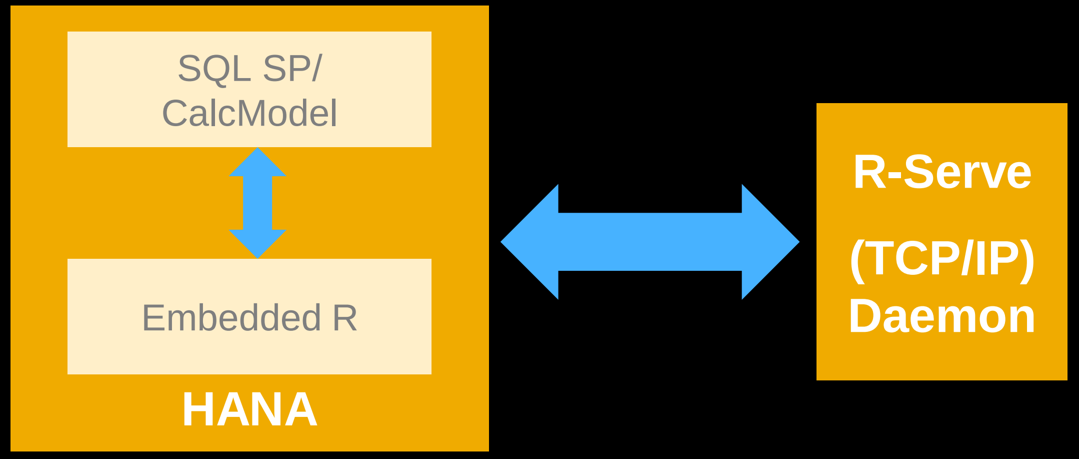
R and HANA Integration – Two Alternative Approaches

#1 Outside-In



- ✓ Use HANA as “just another database” and connect using JDBC/ODBC **(not recommended)**
  - ✓ Row-Vector-Column
  - ✓ Column-Vector-Row
- ✓ Use SAP RHANA package to transfer large amounts of columnar datasets **(recommended)**

#2 Inside Out



- ✓ From inside HANA, transfer huge amounts of data to R, leverage the advanced analytical and statistical capabilities of R and get aggregated results sets back.

R and HANA Integration – Outside In (ODBC Approach)

```
library("RODBC")
# open a new connection
conn <- odbcConnect("LocalHANA", uid="system", pwd="manager")
# create a new table in HANA
sqlSave(conn, iris, "iris", rownames = FALSE)
# Read iris back from HANA into a data.frame
newIris <- sqlFetch(conn, "iris")

# Try saiving again with append true.
sqlSave(conn, newiris, "iris", rownames = FALSE, append = TRUE)
# now we have twice as many records in "SYSTEM.DFTEST"

# let's add a new column to the data frame
newIris$Useless = newIris$Sepal.Length * newdf$Sepal.Width

#Drop table
sqlDrop(conn, "SYSTEM.newIris")
sqlSave(conn, newdf, "SYSTEM.newIris", rownames = FALSE)
#Cleanup
close(conn)
```

Creates a row table, no, no for HANA!!!

R and HANA Integration – Outside In (RHANA Approach)

Use RHANA to Read and Write Dataframes

```
library("RHANA")
library("RJDBC")
 jdbcDriver <- JDBC("com.sap.db.jdbc.Driver", "C:/Program
Files/sap/hdbclient/ngdbc.jar", identifier.quote="`")
#setup connection to RDKOM server
conn_server <- dbConnect(jdbcDriver,
"jdbc:sap:rdkom12.dhcp.pal.sap.corp:30015", "system",
"manager")
#write this airlines data to HANA
system.time(air08 <- read.csv("airlines/2008-short.csv",
header=TRUE, sep=",", na.strings = "NA",
stringsAsFactors=FALSE))
writeDataFrame(conn_server, "air08",  "SYSTEM",
"Airlines_08_1M", force=TRUE)
#Read the data
sql <- 'select * from "Airlines_08_1M" '
system.time(getDataFrame(conn_server, sql,
"airlines081m"))
```

Data transfer between two HANA machines

```
#Read 5y historical data for bay area airports from
# the HANA server to this loca HANA server
sql <- 'select * from airlines_hist_5y_ba'
system.time(getDataFrame(conn_server, sql, "airlines5yba"),
gcFirst=FALSE)
str(airlines5yba)

#setup connection to local host
conn_local <- dbConnect(jdbcDriver,
"jdbc:sap:localhost:30015", "system", "manager")

writeDataFrame(conn_local, "airlines5yba",  "SYSTEM",
"airlines_hist_5y_ba", force=TRUE)
```

R and HANA Integration – Using RHANA to ETL Big Data in HANA Eight Simple Steps

#1 Acquire Airlines Data (12GB)

#2 Load 123M records into a R data.frame

#3 Upload this data.frame into HANA using RHANA

#4 Read the Airport Information (All the airports in US including major airports) in R

#5 Geo-code the airports data using Google GeoCoding APIs (code you saw earlier)

#6 Merge the two datasets to extract major airports information (Not interested in Palo Alto or Hayward airports)

#7 Upload All Airports, Major Airports in HANA

#8 Ready Set Go on your real-time big-data analytics mission!!!

R and HANA Integration – Using RHANA to ETL Big Data in HANA

```
#####################################################################################
#Read all the airlines data, 120M records and upload this into HANA
#####################################################################################
setwd("C:/Users/i827456/Desktop/DKOM-2012/data")
#load libraries
library("RHANA")
library("RJDBC")
library("data.table")

#Read all CSV files from 1987 - 2008, 120M records, create a data frame to hold 120M records
filenames <- list.files(path="../data/airlines/test", pattern="*.csv")
airlines <- data.frame()
for(i in filenames) {
    file <- file.path("../data/airlines/test",i)
    airlines <- rbind(airlines, read.csv(file, header=TRUE, sep=","))
}

#Setup connection to HANA Box - 96Gig, 24Core

jdbcDriver <- JDBC("com.sap.db.jdbc.Driver", "C:/Program Files/sap/hdbclient/ngdbc.jar",
identifier.quote="`")
conn_server <- dbConnect(jdbcDriver, "jdbc:sap:rdkom12.dhcp.pal.sap.corp:30015", "system",
"manager")

#####################################################################################
#Load this airlines data using RHANA into HANA
#####################################################################################
writeDataFrame(conn, "airlines",  "SYSTEM", "Airlines_Hist_Perf", force=TRUE)

# That is it, 120M records uploaded into HANA. Let's take a look at HANA Studio
```

```
#####################################################################################
#Read this airlines data using RHANA from HANA
#####################################################################################
sql <- 'select * from "Airlines_Hist_Perf"'
getDataFrame(conn, sql, "airlinesDB")
#Read this datafram into data.table
airlinesDB <- data.table(airlinesDB)
setkey(airlineDB, Origin, UniqueCarrier, Year, Month)


#####################################################################################
#ETL - Read the AIRPORT Information, get major aiport informatoin extracted and upload this
#transfromed dataset into HANA
#####################################################################################
allairports <- data.table(read.csv("airlines/airports.csv",  header=TRUE, sep=",",
stringsAsFactors=FALSE))
setkey(allairports, iata)

# Extract major airportcodes from 120M records
majorairportscode <- airlinesDB[, union(Origin, Dest)]

#Build Major airports dataset
majorairports <- allairports[majorairportscode]
majorairports$iata <- as.character(majorairports$iata)

#write both the "All Airports" and "Major Airports" dataframes to Hana
writeDataFrame(conn, "majorairports", "SYSTEM", "Major_Airports", force=TRUE)
writeDataFrame(conn, "allairports", "SYSTEM", "All_Airports", force=TRUE)

# Export to local disk
write.csv(majorairports, "output/MajorAirports.csv", row.names=FALSE)
# Completed - extract, transform load
```

In few hours, you can get your big-data story moving…

R and HANA Integration – Using RHANA to ETL Big Data in HANA

```
########################################################################################
#Analysis Time - Summarize data - Airlines performance by month for all of 22 years
#######################################################################################
system.time(averageDelayOverYears <- airlinesDB[,list(
                    CarriersCount=length(unique(UniqueCarrier)),
                    FlightCount=prettyNum(length(FlightNum), big.mark=","),
                                   AirportCount_Origin=length(unique(Origin)),
                    DistanceFlew_miles=prettyNum(sum(as.numeric(Distance), na.rm=TRUE), big.mark=","),
                                   AvgArrDelay_mins=round(mean(ArrDelay, na.rm=TRUE), digits=2),
                    AvgDepDelay_mins=round(mean(DepDelay, na.rm=TRUE), digits=2)
                                   ),by=list(Year, Month)][order(Year, Month)]
)
###  user  system elapsed
### 17.842   1.596  19.487
write.csv(averageDelayOverYears, "output/AggrHViewofAirlines_YM.csv", row.names=FALSE)
```

In just under 20 seconds, 123 MILLION records were analyzed, aggregated and sorted!!!

R and HANA Integration – Using RHANA to ETL Big Data in HANA

```
#########################################################################################################
#Get monthly data for Southwest (WN)
#########################################################################################################
system.time(averageDelayOverMonths <- airlinesDB[UniqueCarrier=="WN",list(
                  FlightCount=prettyNum(length(FlightNum), big.mark=","),
                                AirportCount_Origin=length(unique(Origin)),
                  DistanceFlew_miles=prettyNum(sum(as.numeric(Distance), na.rm=TRUE), big.mark=","),
                                AvgArrDelay_mins=round(mean(ArrDelay, na.rm=TRUE), digits=2),
                  AvgDepDelay_mins=round(mean(DepDelay, na.rm=TRUE), digits=2)
                                ),by=list(Year, Month)][order(Year, Month)]
)
#  user  system elapsed
# 43.214   6.804  50.143
```

In 50 seconds, you can address a business question for Southwest on its growth over the last 20 years!!!

R and HANA Integration – Inside Out
Embedding R in HANA Stored Procedure

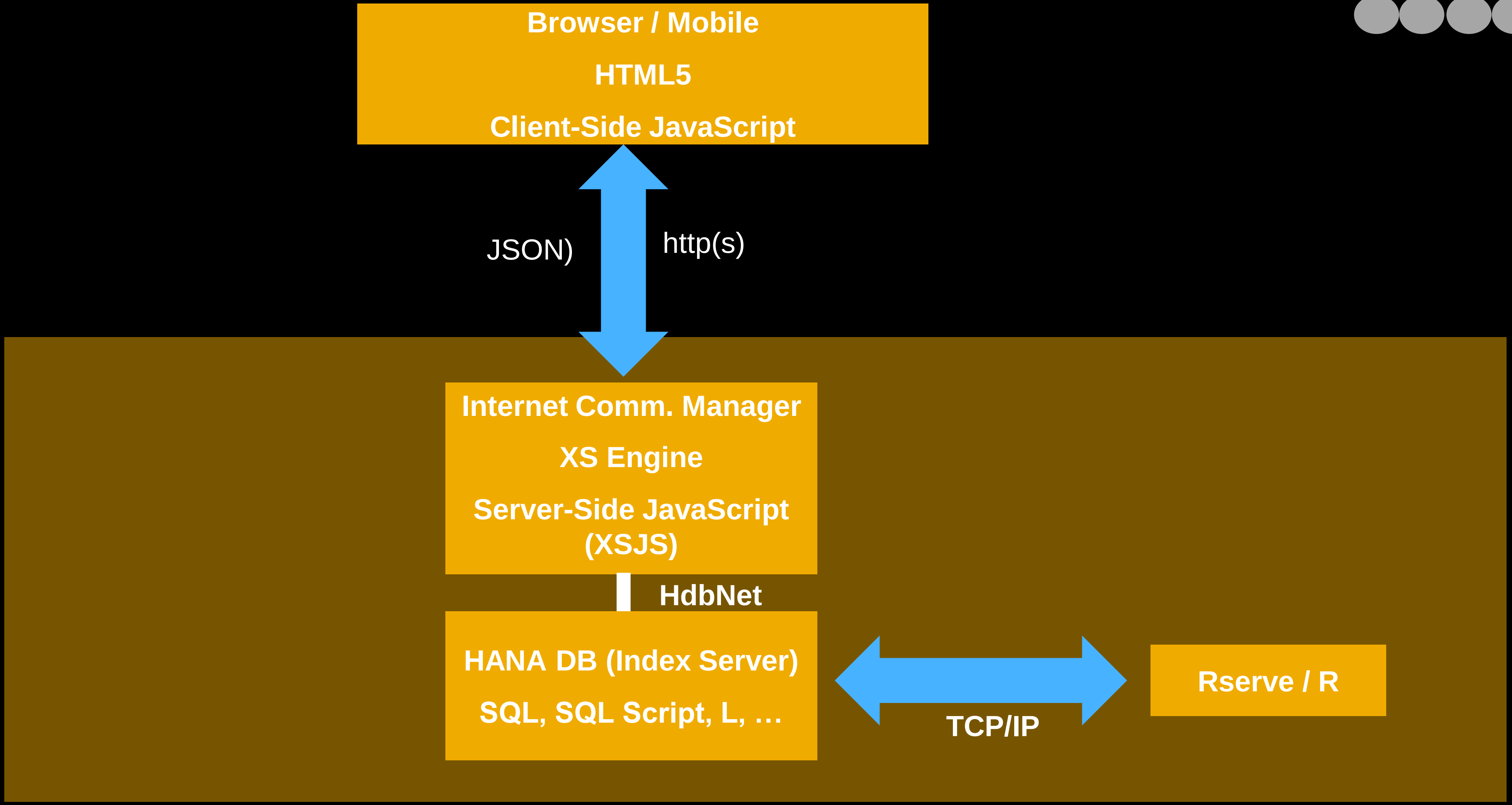
```
drop table json_out; create table json_out (res CLOB);
drop table json_in;  create table json_in (req VARCHAR(250));
delete from json_in; insert into json_in values('{"lat": 40.730885,
"lng":-73.997383}');

DROP PROCEDURE getNearByAirports;
create procedure getNearByAirports(IN airports "Major_Airports",
IN request json_in, OUT response json_out)
LANGUAGE RLANG AS
BEGIN
library(RJSONIO)
library(fossil)
library(data.table)
  req <- fromJSON(as.character(request[[1]]))
clat <- as.numeric(req["lat"])
  clng <- as.numeric(req["lng"])
  #cat(paste(clat, clng, sep=","), file="outputlog1.txt", sep=",")
  #lat <- quote()
  #lng <- quote()
  #Load airports table
  airportsDT <- data.table(airports)
```

```
#Get nearby airports
  nba <- airportsDT[, list(AirportCode=iata,
  Distance=round(deg.dist(clat, clng, lat,
long),digits=0),
     Address=paste(airport, city, state, sep=", "),
     Lat=lat, Lng=long)][order(Distance)][1:3]
  #setkey(nba, AirportCode)
  #cat(toJSON(nba), file="outputlog2.txt", sep=",")
objs <- apply(nba, 1, toJSON)
  nbajson <- paste('[', paste(objs, collapse=', '), ']')
  response <- head(data.frame(RES=nbajson))
END;
call getNearByAirports("Major_Airports", json_in, json_out) with
overview
select * from json_out;
```

Gets the distance on a earth's spherical surface

In milliseconds, you can get nearby airports based on lat/lng and advanced functions in R!!!

R, HANA, XS Engine, JSON/HTML5



R, HANA, XS Engine, JSON/HTML5
Get Nearby Airports Usecase – R, HANA, XS, JSON/HTML5/Jquery

```
function handleGet() {
var body = '';
response.setContentType('application/json');
var deleteJsonIn = "delete from json_in";
var insertJsonIn = "insert into json_in values ('{\"lat\": 40.730885, \"lng\":-73.997383}')";
var callNBAR = "call getNearByAirports(\"Major_Airports\", json_in, null)";
var deleteJsonOut = "delete from json_out";
var readJsonOut = "select top 1 * from json_out";
var conn = getConnection();
var pstmt1 = conn.prepareStatement(deleteJsonIn);
var pstmt2 = conn.prepareStatement(insertJsonIn);
var cstmt =  conn.prepareCall(callNBAR);
var pstmt3 = conn.prepareStatement(deleteJsonOut);
var pstmt4 = conn.prepareStatement(readJsonOut);
var r1 = pstmt1.execute();
var r2 = pstmt2.execute();
var r3 = pstmt3.execute();
conn.commit();
var r4 = cstmt.execute();
var rs = pstmt4.executeQuery();
rs.next();
var jsonout = rs.getClob(1);
trace.error(jsonout);  rs.close();
pstmt1.close();  pstmt2.close();  cstmt.close();  pstmt3.close();  pstmt4.close();
conn.close();  body = JSON.stringify( {"jsonout"} ); response.addBody(body);
response.setReturnCode(200); }
```

JSON Response

```
[
    {
        "AirportCode": "LGA",
        "Distance": "14",
        "Address": "LaGuardia, New York, NY",
        "Lat": "40.77724",
        "Lng": "-73.87261"
    },
    {
        "AirportCode": "EWR",
        "Distance": "19",
        "Address": "Newark Intl, Newark, NJ",
        "Lat": "40.69250",
        "Lng": "-74.16866"
    },
    {
        "AirportCode": "JFK",
        "Distance": "24",
        "Address": "John F Kennedy Intl, New York, NY",
        "Lat": "40.63975",
        "Lng": "-73.77893"
    }
]
```

R and HANA Integration – Nothing is impossible!

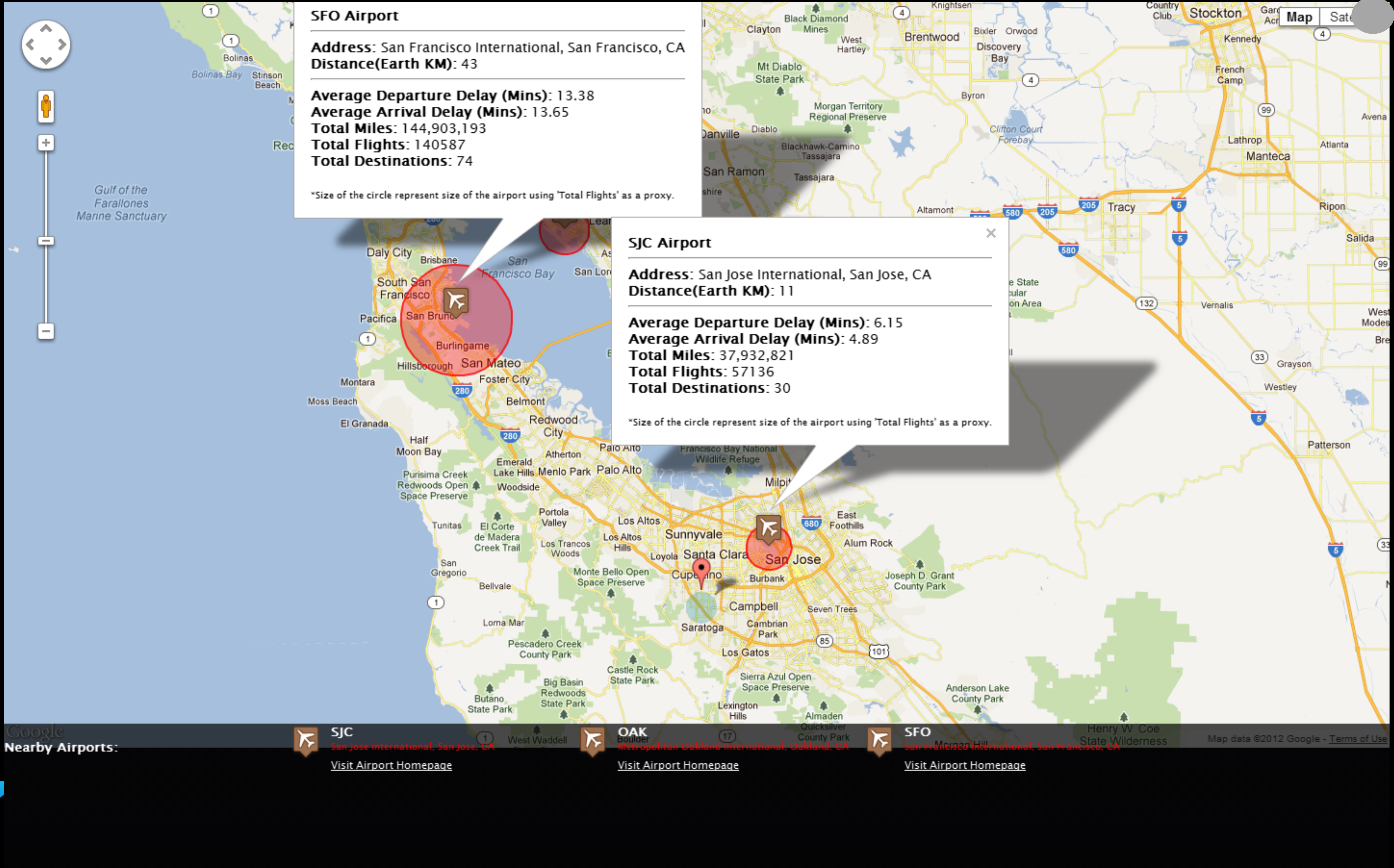
A match made in heaven – the two highly sophisticated in-memory technologies for advanced data analytics and data visualization….

Nothing in the world of analytics and visualization is impossible with R and SAP HANA!

Airlines App Usecases

- ✓ A travel app which will advise you:
  - ✓ Which airport you should fly out from or fly into
  - ✓ Which day and which time of the day are the best to fly
  - ✓ Which airlines you should avoid
  - ✓ What is the most efficient way to get to your desitnation – hassle free, trouble free!
- ✓ Airlines performance data, airports data, weather data, twitter data, financial data and more
- ✓ Geocode and get nearby airports
- ✓ Show ontime performance of these airports
- ✓ Show which day are the best to fly out from a given airport
- ✓ And on and on and on…

iPad Demo of HTML5 App (with R and HANA)



SFO Airport

Address: San Francisco International, San Francisco, CA
Distance(Earth KM): 43

Average Departure Delay (Mins): 13.38
Average Arrival Delay (Mins): 13.65
Total Miles: 144,903,193
Total Flights: 140587
Total Destinations: 74

*Size of the circle represent size of the airport using 'Total Flights' as a proxy.

SJC Airport

Address: San Jose International, San Jose, CA
Distance(Earth KM): 11

Average Departure Delay (Mins): 6.15
Average Arrival Delay (Mins): 4.89
Total Miles: 37,932,821
Total Flights: 57136
Total Destinations: 30

*Size of the circle represent size of the airport using 'Total Flights' as a proxy.

Nearby Airports:

SJC
San Jose International, San Jose, CA
Visit Airport Homepage

OAK
Metropolitan Oakland International, Oakland, CA
Visit Airport Homepage

SFO
San Francisco International, San Francisco, CA
Visit Airport Homepage

Credits

- ✓ All the R Bloggers
- ✓ SAP R Project team
- ✓ SAP RHANA team
- ✓ SAP tRex team
- ✓ SAP XSE team
- ✓ SAP IT
- ✓ DKOM Organization Committee
- ✓ BI and HANA Solution and Product Management Teams

Advanced Analytics with R and SAP HANA

Scan Image to Join the StreamWork Dedicated to this Topic

Thank You!

Contact information:

- ✓ Jitender Aswani (@jaswani)
  My Blog: AllThingsAnalytics
- ✓ Jens Doerpmund

https://streamwork.com/activities/I1WoxlHRjBKOlMPUn4cZ3n?seasip=sap

Legal Disclaimer

The information in this presentation is confidential and proprietary to SAP and may not be disclosed without the permission of SAP. This presentation is not subject to your license agreement or any other service or subscription agreement with SAP. SAP has no obligation to pursue any course of business outlined in this document or any related presentation, or to develop or release any functionality mentioned therein. This document, or any related presentation and SAP's strategy and possible future developments, products and or platforms directions and functionality are all subject to change and may be changed by SAP at any time for any reason without notice. The information on this document is not a commitment, promise or legal obligation to deliver any material, code or functionality. This document is provided without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement. This document is for informational purposes and may not be incorporated into a contract. SAP assumes no responsibility for errors or omissions in this document, except if such damages were caused by SAP intentionally or grossly negligent.

All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, which speak only as of their dates, and they should not be relied upon in making purchasing decisions.

© 2011 SAP AG. All rights reserved.