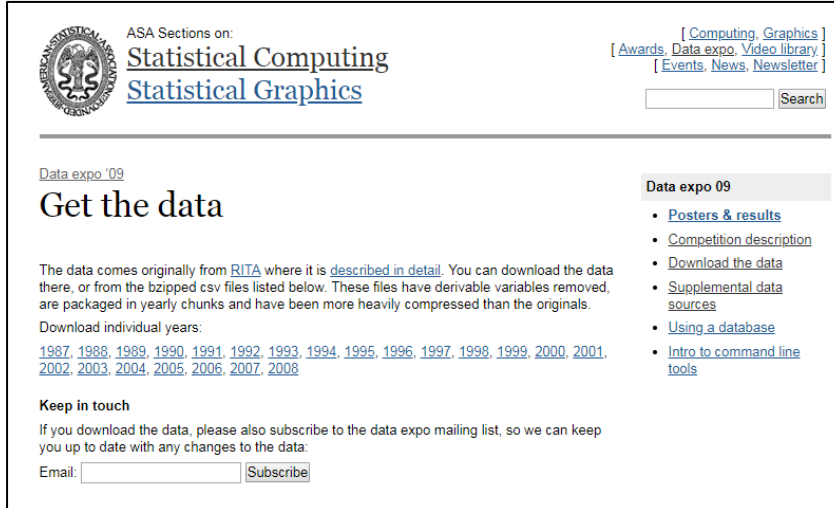


## □ 수행평가 - 빅데이터 분석시스템 구축 과정

### 1. 데이터를 수집 한다.(공공데이터, 서울시 공공데이터)

미국 airline\_delay 2006 ~ 2008 까지 데이터 수집



The screenshot shows the ASA (American Statistical Association) website. The header includes the ASA logo and navigation links for 'Computing, Graphics', 'Awards, Data expo, Video library', and 'Events, News, Newsletter'. A search bar is also present. The main content area is titled 'Data expo '09' and 'Get the data'. It explains that the data comes from RITA and provides a list of years from 1987 to 2008 for download. There are links for 'Posters & results', 'Competition description', 'Download the data', 'Supplemental data sources', 'Using a database', and 'Intro to command line tools'. A 'Keep in touch' section encourages subscribing to a mailing list with an email input field and a 'Subscribe' button.

<http://stat-computing.org/dataexpo/2009/the-data.html>

### 2. 데이터를 빅데이터 시스템(하둡 및 데이터베이스) 저장

- 테이블 생성

```
hive> CREATE TABLE airline_delay(  
Year INT,  
MONTH INT,  
DayofMonth INT,  
DayofWeek INT,  
DepTime INT,  
CRSDepTime INT,  
ArrTime INT,  
CRSArrTime INT,  
UniqueCarrier STRING,  
FlightNum INT,  
TailNum STRING,  
ActualElapsedTime INT,  
CRSElapsedTime INT,  
AirTime INT,  
ArrDelay INT,  
DepDelay INT,  
Origin STRING,  
Dest STRING,  
Distance INT
```

- data load

```

hive> LOAD DATA LOCAL INPATH '/root/airline/2008.csv'
> OVERWRITE INTO TABLE airline_delay
> PARTITION (delayYear='2008');
Loading data to table default.airline_delay partition (delayyear=2008)
Partition default.airline_delay{delayyear=2008} stats: [numFiles=1, numRows=0, totalSize=689413344, rawDataSize=0]
OK
Time taken: 13.296 seconds
hive> LOAD DATA LOCAL INPATH '/root/airline/2007.csv'
> OVERWRITE INTO TABLE airline_delay
> PARTITION (delayYear='2007');
Loading data to table default.airline_delay partition (delayyear=2007)
Partition default.airline_delay{delayyear=2007} stats: [numFiles=1, numRows=0, totalSize=702878193, rawDataSize=0]
OK
Time taken: 25.089 seconds
hive> LOAD DATA LOCAL INPATH '/root/airline/2006.csv'
> OVERWRITE INTO TABLE airline_delay
> PARTITION (delayYear='2006');
Loading data to table default.airline_delay partition (delayyear=2006)

```

- service 돌리기

```
[root@master conf]# hive --service hiveserver2
```

### 3. R을 통해 데이터를 탐색 및 분석 한다

- 필요 라이브러리를 import, HiveDriver와 id/pw/url을 입력

```

library(rJava)
library(RJDBC)
library(DBI)
library(ggplot2)
library(dplyr)

drvName <- 'org.apache.hive.jdbc.HiveDriver';
id <- 'root';
pwd <- '111111';
url <- 'jdbc:hive2://192.168.111.100:10000';

```

- 라이브러리가 담긴 디렉토리를 ClassPath로 지정

```
#라이브러리 디렉토리를 클래스 path로 지정합니다.
hive_lib <- 'C:\\java_hive_lib';
.jinit();
#hive lib에 있는 파일을 클래스파일로 하겠다.
.jaddClassPath(dir(hive_lib, full.names = T));
.jclasspath();
```

- Driver loading을 하고 Connection 해준다.

```
# DB 접근 과정 (Java도 똑같음)
# 1. Driver Loading
drv <- JDBC(driverClass = drvName,
            classPath = 'hive-jdbc-1.0.1.jar')
# 2. Connection
conn <- dbConnect(drv, url, id, pwd)
```

- Statement를 통해 Sql 전송(연도/월 별 연도와 월, 딜레이 된 비중 count)

```
# 3. Statement # sql문 작성 전송
sqlstr <- 'SELECT Year,Month,count(*)
          FROM airline_delay
          where Month IN (1,2,3,4,6,7,8,9,10,11,12)
          AND ArrDelay >0
          GROUP BY Year,Month';
# 4. ResultSet # 값 받기 이 때, data.frame으로 받음
air <- dbGetQuery(conn, sqlstr);
# 5. Close
dbDisconnect(conn)
```

- 연도, 월, 딜레이 Count 칼럼 dataFrame으로 추출

```
air <- rename(air, count = `_c2` )

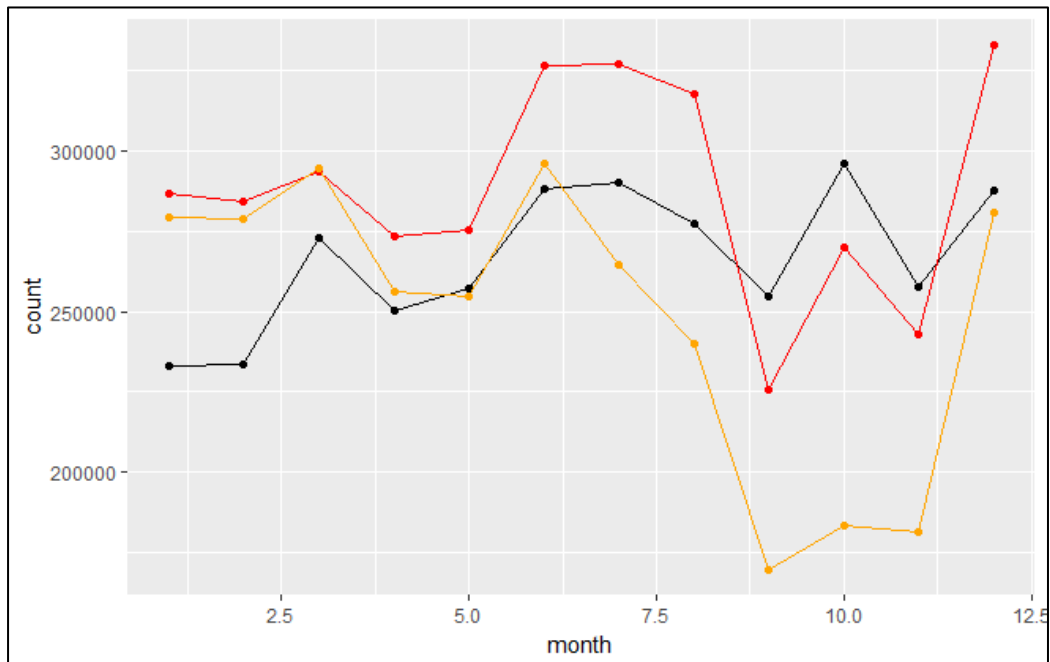
air_06 <- air[air$year == '2006',]
air_07 <- air[air$year == '2007',]
air_08 <- air[air$year == '2008',]
```

#### 4. 데이터 분석 화면을 구현 한다.

- 꺾은선 그래프를 통해 2006, 2007, 2008년도의 월 별 딜레이 횟수를 비교

```
ggplot(data=air_06, aes(x=month, y=count))+
  geom_line()+geom_point()+
  geom_line(data=air_07, aes(x=month, y=count), colour="red")+
  geom_point(data=air_07, aes(x=month, y=count), colour="red")+
  geom_line(data=air_08, aes(x=month, y=count), colour="orange")+
  geom_point(data=air_08, aes(x=month, y=count), colour="orange")
```

- 2006년도 검은색, 2007년도 빨간색, 2008년도 노란색



분석 결과 : 8~9월이 대체로 딜레이 비율이 대폭 하락하며, 6~7월, 11~12월이 딜레이 비율이 대폭 상승한다.

그래프를 분석한 결과, 휴가 시즌에 딜레이 비율이 높은 것으로 나타났다.