1. **Give a brief overview of how you extracted and cleaned the data.**

I launched an EC2 instance (t2.micro) with the EBS volume attached and then mounted:
http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-attaching-volume.html
http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-using-volumes.html

At the time data were available, I dig into the information and realized only files under *"/aviation/airline_ontime"* were required to complete all the question along the different groups. The fields imported were: (*Year, Month, DayofMonth, Weekday, UniqueCarrier, FlightNum, Origin, Dest, CRSDepTime, DepDelay, CRSArrtime, ArrDelay, Cancelled*).

A python script was developed to extract, clean and import data to S3 bucket called "airlineontime", different python modules was used to complete every step:

- The zipfile module was used to manipulate ZIP archive files.
- The os module was used to manipulate path and read all the lines in all the files.
- The csv module was used to manipulate all the files in csv format.
- The Boto module is the AWS SDK for Python, which allows write software that makes use of Amazon services like S3 and EC2.

2. **Give a brief overview of how you integrated each system.**

EC2 volume was loaded into S3 bucket called *"airlineontine"*. Public and secret key was provided to EC2 instance with a proper policy providing rights to load data into S3 bucket. Data from S3 bucket was available for HUE and by means of Hive SQL-like query language could be easily manipulated. S3 bucket was stored in HDFS, using Hive, in logical structures which can be easily queried by means HQL. Data from HDFS can be loaded into Dynamo DB tables by means of Hive queries:

```
CREATE EXTERNAL TABLE airline_ontime (year INT, month INT, day INT, weekday INT,
carrier STRING, flight_num STRING, origin STRING, dest STRING, deptime STRING, depdelay
INT, arrtime STRING, arrdelay INT, cancelled INT)
PARTITIONED BY (date string)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ","
LOCATION 's3n://airline-ontime/';
MSCK REPAIR TABLE airline_ontime;
```

3. **What approaches and algorithms did you use to answer each question?**

Group 1, 2, 3: Hive SQL-like query language on top of Hadoop HDFS file system.
Group 1 ex 2 query example:

```
select carrier, sum(arrdelay)/count(arrdelay) as mean_delay from airline_ontime
where cancelled = 0 group by carrier order by mean_delay asc limit 10;
```

Group 2, 3: Create Dynamo DB tables and insert data from Hive queries result.
Group 2 ex 4 query example:

```
insert overwrite table group2_ex4
select origin, dest as destination, sum(arrdelay)/count(arrdelay) as mean_delay from
airline_ontime where cancelled = 0 group by origin, dest;
```

4. **What are the results of each question?**

Group 1 (Answer any 2):

1. Rank the top 10 most popular airports by numbers of flights to/from the airport.

airport	*total*
ORD	*12051796*
ATL	*11323515*
DFW	*10591818*
LAX	*7586304*
PHX	*6505078*
DEN	*6183518*
DTW	*5504120*
IAH	*5416653*
MSP	*5087036*
SFO	*5062339*

2. Rank the top 10 airlines by on-time arrival performance.

carrier	*mean_delay*
HA	*-1.01180434575*
AQ	*1.15692344248*
PS	*1.45063851278*
ML (1)	*4.74760919573*
PA (1)	*5.32243099993*
F9	*5.46588114882*
NW	*5.55778339267*
WN	*5.56077425988*
OO	*5.73631246366*
9E	*5.8671846617*

3. Rank the days of the week by on-time arrival performance.

weekday	*mean_delay*
6	*4.30166992608*
2	*5.99045884132*
7	*6.61328029244*
1	*6.71610280259*
3	*7.20365639467*
4	*9.09444100834*
5	*9.72103233759*

Group 2 (Answer any 3):

1. For each airport X, rank the top-10 carriers in decreasing order of on-time departure performance from X.

    1. CMI (University of Illinois Willard Airport)

Airport	*carrier*	*mean_delay*
CMI	*OH*	*0*
CMI	*US*	*2*
CMI	*PI*	*4*
CMI	*TW*	*4*

CMI	*DH*	*6*
CMI	*EV*	*6*
CMI	*MQ*	*8*

2. BWI (Baltimore-Washington International Airport)

airport	***carrier***	***mean_delay***
BWI	*F9*	*0*
BWI	*PA (1)*	*4*
BWI	*CO*	*5*
BWI	*NW*	*5*
BWI	*YV*	*5*
BWI	*AA*	*6*
BWI	*9E*	*7*
BWI	*DL*	*7*
BWI	*FL*	*7*
BWI	*UA*	*7*

3. MIA (Miami International Airport)

airport	***carrier***	***mean_delay***
MIA	*9E*	*-3*
MIA	*EV*	*1*
MIA	*TZ*	*1*
MIA	*XE*	*1*
MIA	*NW*	*4*
MIA	*PA (1)*	*4*
MIA	*UA*	*6*
MIA	*US*	*6*
MIA	*ML (1)*	*7*
MIA	*FL*	*8*

4. LAX (Los Angeles International Airport)

airport	***carrier***	***mean_delay***
LAX	*MQ*	*2*
LAX	*FL*	*4*
LAX	*OO*	*4*
LAX	*PS*	*4*
LAX	*TZ*	*4*
LAX	*F9*	*5*
LAX	*HA*	*5*
LAX	*NW*	*5*
LAX	*US*	*6*
LAX	*YV*	*6*

5. IAH (George Bush Intercontinental Airport)

airport	***carrier***	***mean_delay***
IAH	*NW*	*3*
IAH	*PA (1)*	*3*
IAH	*PI*	*3*

IAH	*AA*	*5*
IAH	*F9*	*5*
IAH	*US*	*5*
IAH	*HP*	*6*
IAH	*MQ*	*6*
IAH	*OO*	*6*
IAH	*TW*	*6*

6. SFO (San Francisco International Airport)

airport	***carrier***	***mean_delay***
SFO	*TZ*	*3*
SFO	*MQ*	*4*
SFO	*F9*	*5*
SFO	*NW*	*5*
SFO	*PA (1)*	*5*
SFO	*DL*	*6*
SFO	*PS*	*6*
SFO	*AA*	*7*
SFO	*CO*	*7*
SFO	*TW*	*7*

2. For each airport X, rank the top-10 airports in decreasing order of on-time departure performance from X.

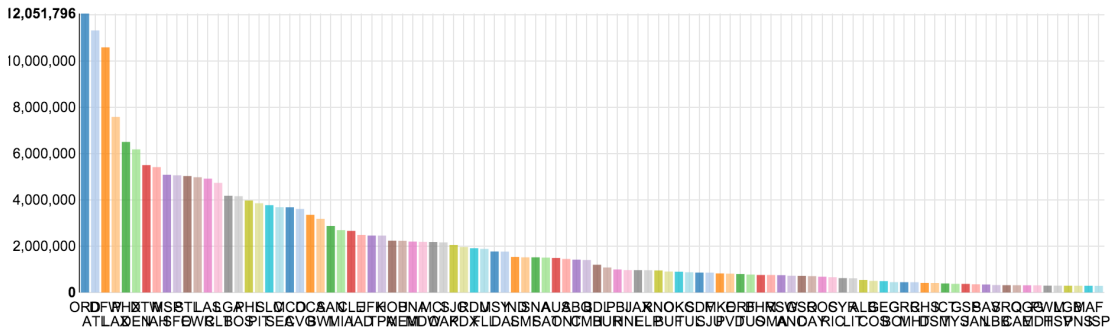
airport	***List of top-10 airports***
CMI	*ABI, PIT, CVG, DAY, STL, PIA, DFW, ATL, ORD*
BWI	*SAV, SRQ, DAB, IAD, MLB, UCA, CHO, DCA, IAH, OAJ*
MIA	*SHV, BUF, SAN, SLC, HOU, ISP, MEM, PSE, GNV, TLH*
LAX	*GRR, AZO, MSP, DTW, DAY, PIT, CVG, CLE, IAD, ATL*
IAH	*MSN, MLI, AGS, EFD, JAC, HOU, MTJ, VCT, RNO, BPT*
SFO	*SDF, MSO, PIH, LGA, PIE, FAR, OAK, BNA, MEM, SCK*

3. For each source-destination pair X-Y, rank the top-10 carriers in decreasing order of on-time arrival performance at Y from X.
4. For each source-destination pair X-Y, determine the mean arrival delay (in minutes) for a flight from X to Y.

origin	***destination***	***mean_delay***
CMI	*ORD*	*10*
IND	*CMH*	*2*
DFW	*IAH*	*7*
LAX	*SFO*	*9*
JFK	*LAX*	*6*
ATL	*PHX*	*9*

Group 3 (Answer both using only Hadoop and Spark):

1. Does the popularity distribution of airports follow a Zipf distribution? If not, what distribution does it follow?



Do not follow a Zipf distribution, on the other hand it looks like a logarithmic distribution.

2. Find, for each X-Y-Z and day/month combination in the year 2008, the two flights (X-Y and Y-Z) that satisfy constraints, if such flights exist.

route	***depdate***	***flight_xy***	***flight_yz***
CMI-ORD-LAX	*2008-03-04*	*MQ4278*	*AA1345*
CMI-ORD-LAX	*2008-03-04*	*MQ4401*	*AA1345*
JAX-DFW-CRP	*2008-09-09*	*AA845*	*MQ3627*
SLC-BFL-LAX	*2008-04-01*	*OO3755*	*OO5429*
LAX-SFO-PHX	*2008-07-12*	*WN3534*	*US412*
DFW-ORD-DFW	*2008-06-10*	*UA1104*	*OO6119*
LAX-ORD-JFK	*2008-01-01*	*UA944*	*B6918*

5. **What system- or application-level optimizations (if any) did you employ?**
   DynamoDB sort keys were properly selected in order to obtain results faster. S3 buckets provide persistent storage which reduce wastes populating HDFS every time the Hadoop cluster is stopped.

6. **Your opinion about whether the results make sense and are useful in any way.**
   Conclusion about size/popularity of different airport was quite predictable. The information obtained regarding different types of mean delay were not predictable at all and are quite useful from different points of views: carriers could take actions to reduce delays just in case these are not acceptable and passengers could adapt their preferences not only about carriers but also regarding flights' scheduling to avoid delays.

   VIDEO DEMONSTRATION:
   https://sendvid.com/nqczdubk