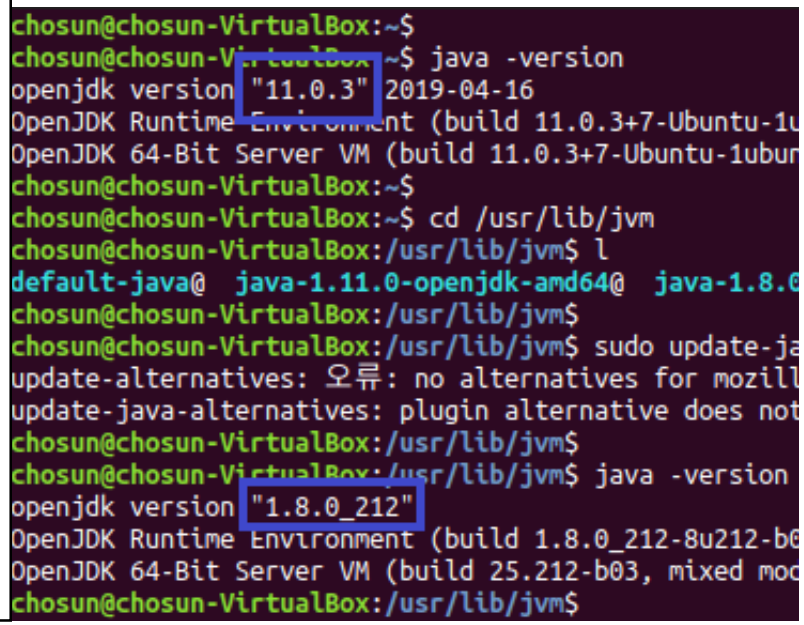
Hive 설치

자바 버전 수정

```
chosun@chosun-VirtualBox:~$
chosun@chosun-VirtualBox:~$ java -version
openjdk version "11.0.3" 2019-04-16
OpenJDK Runtime Environment (build 11.0.3+7-Ubuntu-1ubuntu218.04.1)
OpenJDK 64-Bit Server VM (build 11.0.3+7-Ubuntu-1ubuntu218.04.1, mixed mode, sharing)
chosun@chosun-VirtualBox:~$
chosun@chosun-VirtualBox:~$ cd /usr/lib/jvm
chosun@chosun-VirtualBox:/usr/lib/jvm$ l
default-java@ java-1.11.0-openjdk-amd64@ java-1.8.0-openjdk-amd64@
java-11-openjdk-amd64/ java-8-openjdk-amd64/
chosun@chosun-VirtualBox:/usr/lib/jvm$
chosun@chosun-VirtualBox:/usr/lib/jvm$ sudo update-java-alternatives -s java-1.8.0-openjdk-amd64
update-alternatives: 오류: no alternatives for mozilla-javaplugin.so
update-java-alternatives: plugin alternative does not exist: /usr/lib/jvm/java-8-openjdk-amd64/jre/lib/amd64/IcedTeaPlugin.so
chosun@chosun-VirtualBox:/usr/lib/jvm$
chosun@chosun-VirtualBox:/usr/lib/jvm$ java -version
openjdk version "1.8.0_212"
OpenJDK Runtime Environment (build 1.8.0_212-8u212-b03-0ubuntu1.18.04.1-b03)
OpenJDK 64-Bit Server VM (build 25.212-b03, mixed mode)
chosun@chosun-VirtualBox:/usr/lib/jvm$
chosun@chosun-VirtualBox:/usr/lib/jvm$ cd
chosun@chosun-VirtualBox:~$
```

Hive가 jdk11에서는 작동하지 않으므로 Jdk8을 설치했습니다.

1. Java버전을 확인했습니다. >> 11.0.3
2. Java가 설치된 디렉터리로 이동해서 Java8버전이 있는지 확인했습니다.
3. Java8버전을 설치하고 버전을 확인했습니다. >> 1.8.0_212
4. 홈 디렉터리로 돌아왔습니다.



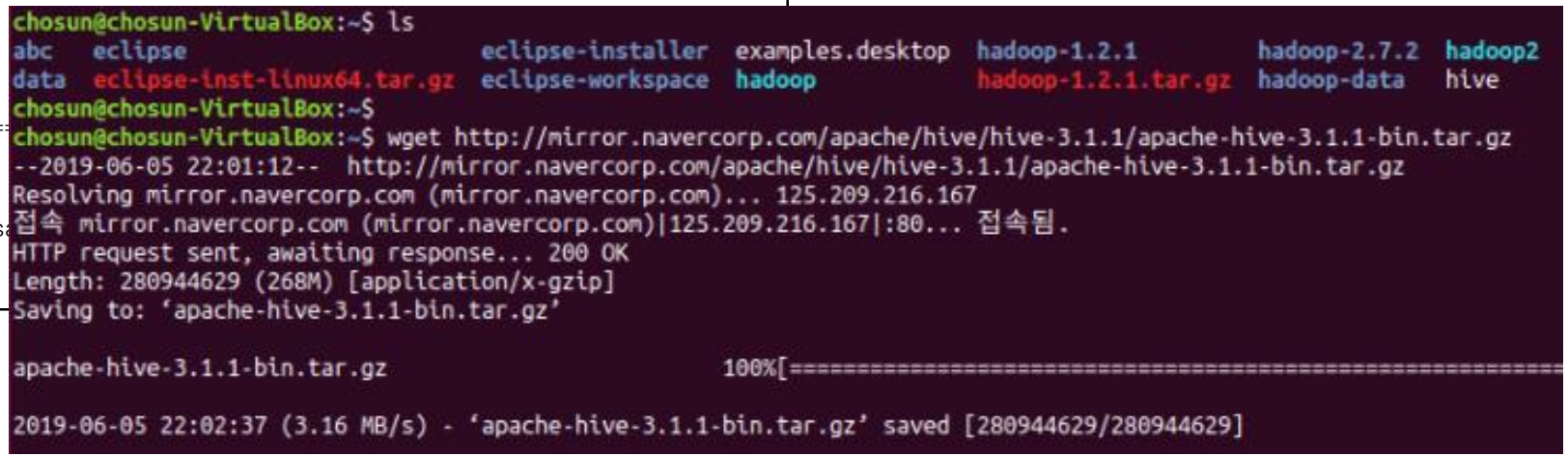
Hive.tar.gz 다운로드

```
chosun@chosun-VirtualBox:~$
chosun@chosun-VirtualBox:~$ ls
Abc eclipse eclipse-installer examples.desktop Hadoop-1.2.1 Hadoop-2.7.2 hadoop2 공개 문서
비디오 음악 data eclipse-inst-linux64.tar.gz eclipse-workspace Hadoop Hadoop-1.2.1.tar.gz
Hadoop-data hive 다운로드 바탕화면 사진 템플릿
chosun@chosun-VirtualBox:~$
chosun@chosun-VirtualBox:~$ wget http://mirror.navercorp.com/apache/hive/hive-3.1.1/apache-hive-3.1.1-
bin.tar.gz
--2019-06-05 22:01:12-- http://mirror.navercorp.com/apache/hive/hive-3.1.1/apache-hive-3.1.1-bin.tar.gz
Resolving mirror.navercorp.com (mirror.navercorp.com)... 125.209.216.167
접속 mirror.navercorp.com (mirror.navercorp.com)|125.209.216.167|:80... 접속됨.
HTTP request sent, awaiting response... 200 OK
Length: 280944629 (268M) [application/x-gzip]
Saving to: 'apache-hive-3.1.1-bin.tar.gz'
apache-hive-3.1.1-bin.tar.gz
100%[==========================================================
==============>] 267.93M 3.25MB/s in 85s
2019-06-05 22:02:37 (3.16 MB/s) - 'apache-hive-3.1.1-bin.tar.gz' sa
chosun@chosun-VirtualBox:~$
```

Hive 파일을 다운로드 합니다.

1. wget 명령어를 사용해서 hive 파일을 다운로드 했습니다.

http://mirror.navercorp.com/apache/hive/hive-3.1.1/apache-hive-3.1.1-bin.tar.gz

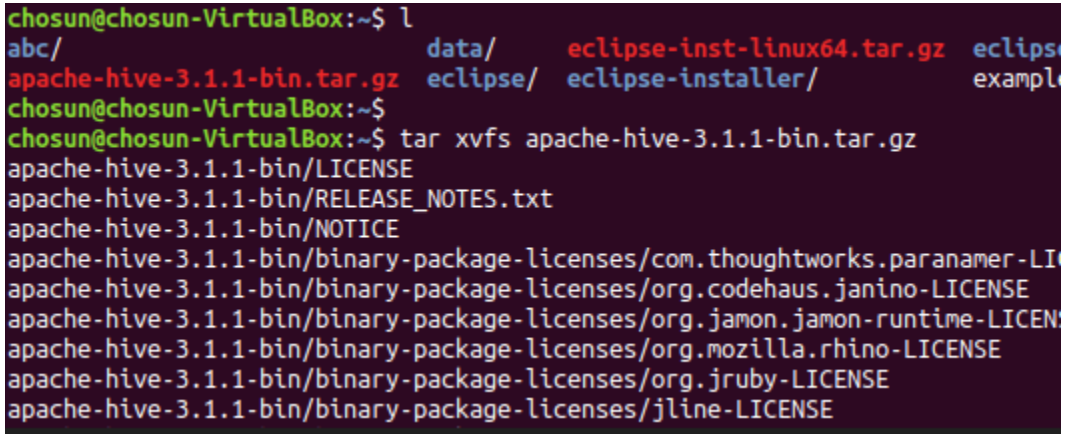


tar Hive.tar.gz

```
chosun@chosun-VirtualBox:~$ l
abc/ data/ eclipse-inst-linux64.tar.gz eclipse-workspace/ hadoop@ hadoop-1.2.1.tar.gz hadoop-data/ hive 다운로드/ 바탕화면/ 사진/
템플릿/ apache-hive-3.1.1-bin.tar.gz eclipse/ eclipse-installer/ examples.desktop hadoop-1.2.1/ hadoop-2.7.2/
hadoop2@ 공개/ 문서/ 비디오/ 음악/
chosun@chosun-VirtualBox:~$
chosun@chosun-VirtualBox:~$ tar xvfs apache-hive-3.1.1-bin.tar.gz
apache-hive-3.1.1-bin/LICENSE
apache-hive-3.1.1-bin/RELEASE_NOTES.txt
…
apache-hive-3.1.1-bin/hcatalog/share/webhcat/java-client/hive-webhcat-java-client-3.1.1.jar
chosun@chosun-VirtualBox:~$
```

다운받은 Hive 파일의 압축을 풉니다.

1. tar 명령어를 사용해서 hive 파일의 압축을 풀었습니다.



ln -s Hive.tar.gz

```
chosun@chosun-VirtualBox:~$ l
abc/ apache-hive-3.1.1-bin.tar.gz eclipse/ eclipse-installer/ examples.desktop hadoop-1.2.1/ hadoop-2.7.2/ hadoop2@
다운로드/ 바탕화면/ 사진/ 템플릿/ apache-hive-3.1.1-bin/ data/ eclipse-inst-linux64.tar.gz eclipse-workspace/ hadoop@ hadoop-
1.2.1.tar.gz hadoop-data/ 공개/ 문서/ 비디오/ 음악/
chosun@chosun-VirtualBox:~$
chosun@chosun-VirtualBox:~$ ln -s apache-hive-3.1.1-bin hive
chosun@chosun-VirtualBox:~$
chosun@chosun-VirtualBox:~$ l
abc/ apache-hive-3.1.1-bin.tar.gz eclipse/ eclipse-installer/ examples.desktop hadoop-1.2.1/ ha
문서/ 비디오/ 음악/ apache-hive-3.1.1-bin/ data/ eclipse-inst-linux64.tar.gz eclipse-workspace/
hadoop-data/ hive@ 다운로드/ 바탕화면/ 사진/ 템플릿/
chosun@chosun-VirtualBox:~$
```

Hive 디렉터리의 심볼릭 링크를 생성합니다.

1. apache-hive-3.1.1-bin/ 디렉터리가 생성됨을 확인했습니다.
2. 생성된 디렉터리의 심볼릭 링크를 hive라는 이름으로 생성했습니다.
3. apache-hive-3.1.1-bin/의 심볼릭 링크가 생성됨을 확인했습니다.

```
chosun@chosun-VirtualBox:~$ l
abc/                  apache-hive-3.1.1-bin.tar.gz  eclipse/                 eclipse-installer/  examples.desktop  hadoop-1.2.1/       hadoop-2.7.2/  hadoop2@  다운로
apache-hive-3.1.1-bin/  data/                       eclipse-inst-linux64.tar.gz  eclipse-workspace/  hadoop@           hadoop-1.2.1.tar.gz  hadoop-data/  공개/     문서/
chosun@chosun-VirtualBox:~$
chosun@chosun-VirtualBox:~$ ln -s apache-hive-3.1.1-bin hive
chosun@chosun-VirtualBox:~$
chosun@chosun-VirtualBox:~$ l
abc/                  apache-hive-3.1.1-bin.tar.gz  eclipse/                 eclipse-installer/  examples.desktop  hadoop-1.2.1/       hadoop-2.7.2/  hadoop2@  공개/
apache-hive-3.1.1-bin/  data/                       eclipse-inst-linux64.tar.gz  eclipse-workspace/  hadoop@           hadoop-1.2.1.tar.gz  hadoop-data/  hive@     다운로
```

Hive 환경변수 설정

Hive 환경변수 파일을 수정합니다.

1. Hive 환경변수 설정파일은 apache-hive-3.1.1-bin/conf 에 있습니다.
   conf 디렉터리로 이동했습니다.
2. 환경변수 파일들을 확인했습니다.

```
chosun@chosun-VirtualBox:~$ cd hive
chosun@chosun-VirtualBox:~/hive$
chosun@chosun-VirtualBox:~/hive$ ls
LICENSE NOTICE RELEASE_NOTES.txt bin binary-package-licenses conf examples hcatalog jdbc lib scripts
chosun@chosun-VirtualBox:~/hive$
chosun@chosun-VirtualBox:~/hive$ cd conf
chosun@chosun-VirtualBox:~/hive/conf$ ls
beeline-log4j2.properties.template hive-env.sh.template hive-log4j2.properties.template
llap-cli-log4j2.properties.template parquet-logging.properties hive-default.xml.template
hive-exec-log4j2.properties.template ivysettings.xml llap-daemon-log4j2.properties.template
chosun@chosun-VirtualBox:~/hive/conf$
```

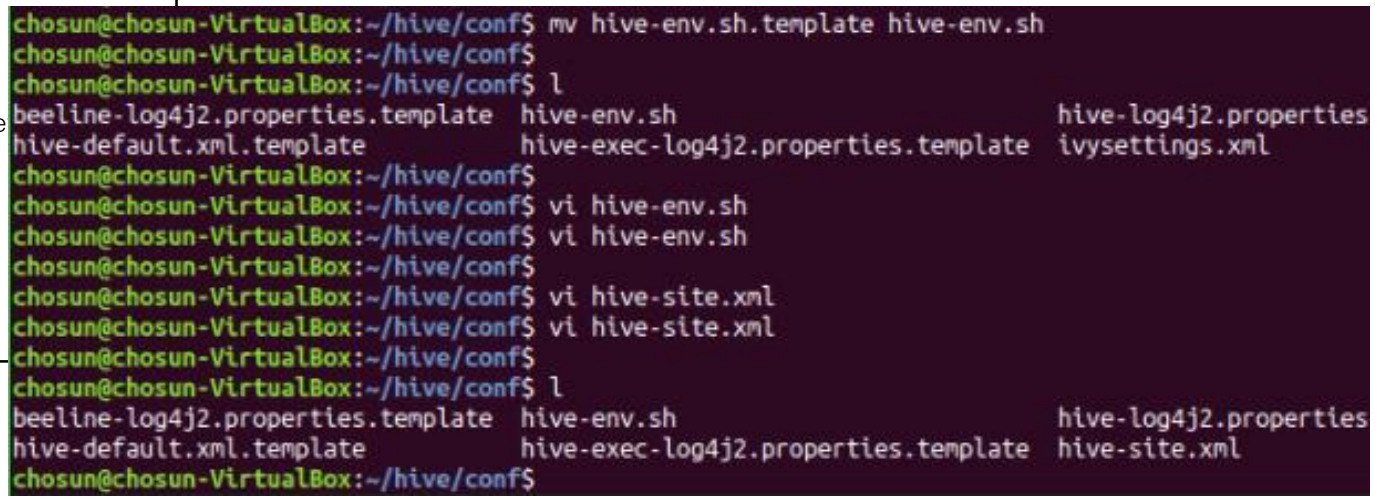
```
chosun@chosun-VirtualBox:~/hive$
chosun@chosun-VirtualBox:~/hive$ cd conf
chosun@chosun-VirtualBox:~/hive/conf$ ls
beeline-log4j2.properties.template  hive-env.sh.template                hive-log4j2.properties.template  llap-cli-log4j2.properties.template     parquet-logging.properties
hive-default.xml.template           hive-exec-log4j2.properties.template  ivysettings.xml                  llap-daemon-log4j2.properties.template
chosun@chosun-VirtualBox:~/hive/conf$
```

Hive 환경변수 설정

```
chosun@chosun-VirtualBox:~/hive/conf$ mv hive-env.sh.template hive-env.sh
chosun@chosun-VirtualBox:~/hive/conf$
chosun@chosun-VirtualBox:~/hive/conf$ l
beeline-log4j2.properties.template  hive-env.sh  hive-log4j2.properties.template
llap-cli-log4j2.properties.template  parquet-logging.properties  hive-default.xml.template
hive-exec-log4j2.properties.template  ivysettings.xml llap-daemon-log4j2.properties.template
chosun@chosun-VirtualBox:~/hive/conf$
chosun@chosun-VirtualBox:~/hive/conf$ vi hive-env.sh
chosun@chosun-VirtualBox:~/hive/conf$ vi hive-env.sh
chosun@chosun-VirtualBox:~/hive/conf$
chosun@chosun-VirtualBox:~/hive/conf$ vi hive-site.xml
chosun@chosun-VirtualBox:~/hive/conf$ vi hive-site.xml
chosun@chosun-VirtualBox:~/hive/conf$
chosun@chosun-VirtualBox:~/hive/conf$ l
beeline-log4j2.properties.template  hive-env.sh  hive-log4j2.properties.template ivyse
llap-daemon-log4j2.properties.template  hive-default.xml.template
hive-exec-log4j2.properties.template  hive-site.xml
llap-cli-log4j2.properties.template  parquet-logging.properties
chosun@chosun-VirtualBox:~/hive/conf$
```

Hive 환경변수 파일을 수정합니다.

1. hive-env.sh.template 파일의 이름을 hive-env.sh으로 수정했습니다.
2. vi 명령어를 사용해서 hive-env.sh파일의 내용을 수정했습니다.
3. vi 명령어를 사용해서 hive-site.xml파일을 생성하고 내용을 수정했습니다.
   명령어를 두번 사용해서 저장한 파일의 내용을 확인했습니다.



Hive 환경변수 설정

```
# Set Hive and Hadoop environment variables here. These variables can be used
# to control the execution of Hive. It should be used by admins to configure
# the Hive installation (so that users do not have to set environment variables
# or set command line parameters to get correct behavior).
#
# The hive service being invoked (CLI etc.) is available via the environment
# variable SERVICE


# Hive Client memory usage can be an issue if a large number of clients
# are running at the same time. The flags below have been useful in
# reducing memory usage:
#
# if [ "$SERVICE" = "cli" ]; then
#   if [ -z "$DEBUG" ]; then
#     export HADOOP_OPTS="$HADOOP_OPTS -XX:NewRatio=12 -Xms10m -XX:MaxHeapFreeRa
#   else
#     export HADOOP_OPTS="$HADOOP_OPTS -XX:NewRatio=12 -Xms10m -XX:MaxHeapFreeRa
#   fi
# fi

# The heap size of the jvm stared by hive shell script can be controlled via:
#
# export HADOOP_HEAPSIZE=1024
#
# Larger heap size may be required when running queries over large number of fil
# By default hive shell scripts use a heap size of 256 (MB).  Larger heap size w
# appropriate for hive server.


# Set HADOOP_HOME to point to a specific hadoop install directory
# HADOOP_HOME=${bin}/../../hadoop
HADOOP_HOME=/home/chosun/hadoop2

# Hive Configuration Directory can be controlled by:
# export HIVE_CONF_DIR=

# Folder containing extra libraries required for hive compilation/execution can
# export HIVE_AUX_JARS_PATH=
```

hive-env.sh

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
  <property>
        <name>hive.metastore.warehouse.dir</name>
        <value>/user/hive/warehouse</value>
  </property>
  <property>
        <name>hive.cli.print.header</name>
        <value>true</value>
  </property>
</configuration>
```

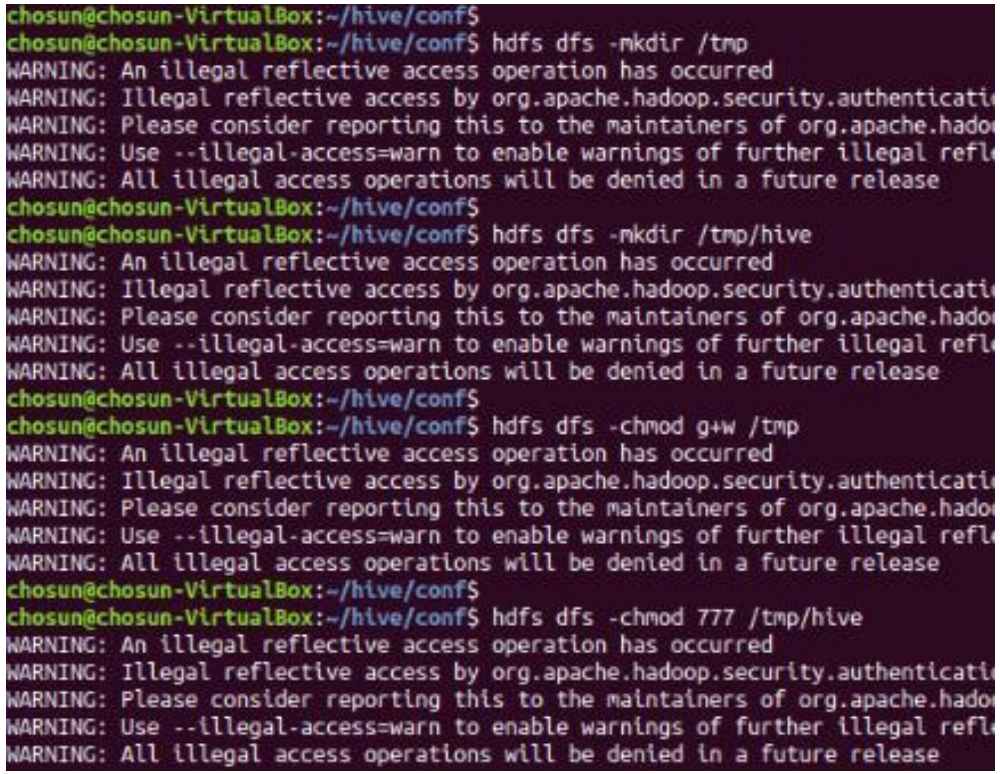
hive-site.xml

Hive 데이터 저장공간 생성

Hive에서 업로드할 데이터의 저장공간을 생성합니다.

1. Hive에서 업로드하는 데이터는 HDFS의 /user/hive/warehouse에 저장됩니다.
   Hive에서 실행하는 잡의 여유 공간으로 HDFS의 /tmp/hive-유저명 디렉터리를 사용합니다.
   이 두 경로에 해당하는 디렉터리를 생성하고, 실행 권한을 설정했습니다.

```
chosun@chosun-VirtualBox:~/hive/conf$ hdfs dfs -mkdir /tmp
WARNING: An illegal reflective access operation has occurred
WARNING: All illegal access operations will be denied in a future release
chosun@chosun-VirtualBox:~/hive/conf$
chosun@chosun-VirtualBox:~/hive/conf$ hdfs dfs -mkdir /tmp/hive
WARNING: An illegal reflective access operation has occurred
WARNING: All illegal access operations will be denied in a future release
chosun@chosun-VirtualBox:~/hive/conf$
chosun@chosun-VirtualBox:~/hive/conf$ hdfs dfs -chmod g+w /tmp
WARNING: An illegal reflective access operation has occurred
WARNING: All illegal access operations will be denied in a future release
chosun@chosun-VirtualBox:~/hive/conf$
chosun@chosun-VirtualBox:~/hive/conf$ hdfs dfs -chmod 777 /tmp/hive
WARNING: An illegal reflective access operation has occurred
WARNING: All illegal access operations will be denied in a future release
chosun@chosun-VirtualBox:~/hive/conf$
```

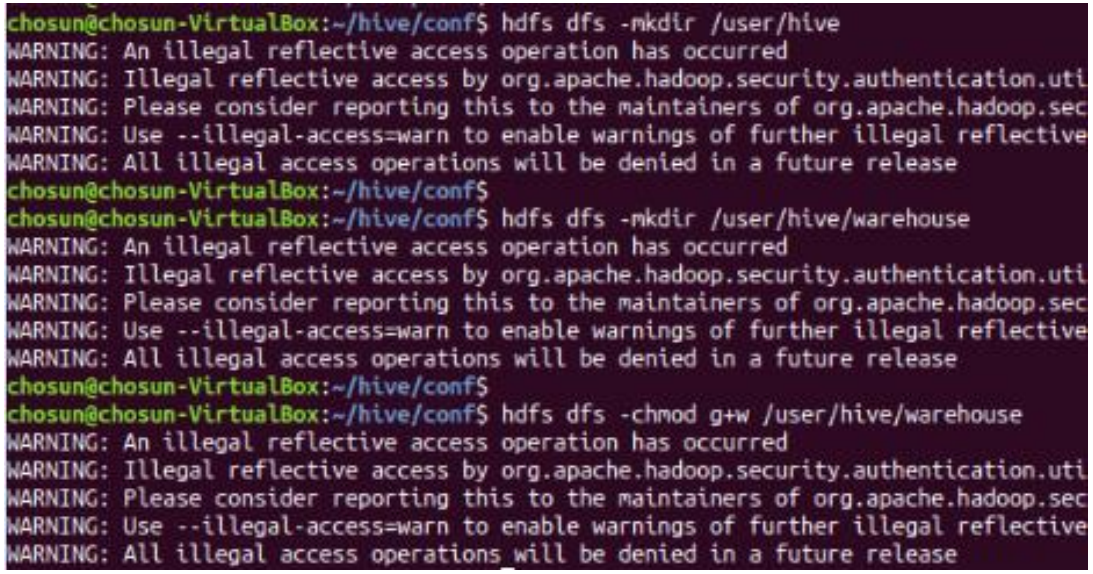


Hive 데이터 저장공간 생성

Hive에서 업로드할 데이터의 저장공간을 생성합니다.

1. Hive에서 업로드하는 데이터는 HDFS의 /user/hive/warehouse에 저장됩니다.
   Hive에서 실행하는 잡의 여유 공간으로 HDFS의 /tmp/hive-유저명 디렉터리를 사용합니다.
   이 두 경로에 해당하는 디렉터리를 생성하고, 실행 권한을 설정했습니다.

```
chosun@chosun-VirtualBox:~/hive/conf$ hdfs dfs -mkdir /user/hive
WARNING: An illegal reflective access operation has occurred
WARNING: All illegal access operations will be denied in a future release
chosun@chosun-VirtualBox:~/hive/conf$
chosun@chosun-VirtualBox:~/hive/conf$ hdfs dfs -mkdir /user/hive/warehouse
WARNING: An illegal reflective access operation has occurred
WARNING: All illegal access operations will be denied in a future release
chosun@chosun-VirtualBox:~/hive/conf$
chosun@chosun-VirtualBox:~/hive/conf$ hdfs dfs -chmod g+w /user/hive/warehouse
WARNING: An illegal reflective access operation has occurred
WARNING: All illegal access operations will be denied in a future release
chosun@chosun-VirtualBox:~/hive/conf$
```



Hive 메타데이터 초기화

Hive의 메타데이터를 초기화합니다.

1. initschema 명령어를 사용해서 메타스토어를 초기화했습니다.
   Hive 2.0.0버전부터는 하이브를 실행하기 전에 하이브 메타스토어를 초기화해야 합니다.
   hive-site.xml에 별도의 메타스토어를 설정하지 않았으므로 옵션으로 -dbType derby를 사용했습니다.

```
chosun@chosun-VirtualBox:~$ ./hive/bin/schematool -initSchema -dbType derby
/home/chosun/hive/conf/hive-env.sh: 줄 49: HADOOP: 명령어를 찾을 수 없음
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/chosun/apache-hive-3.1.1-bin/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/chosun/hadoop-2.7.2/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Metastore connection URL:        jdbc:derby:;databaseName=metastore_db;create=true
Metastore Connection Driver :    org.apache.derby.jdbc.EmbeddedDriver
Metastore connection User:       APP
Starting metastore schema initialization to 3.1.0
Initialization script hive-schema-3.1.0.derby.sql
...
Initialization script completed
schemaTool completed
chosun@chosun-VirtualBox:~$
```

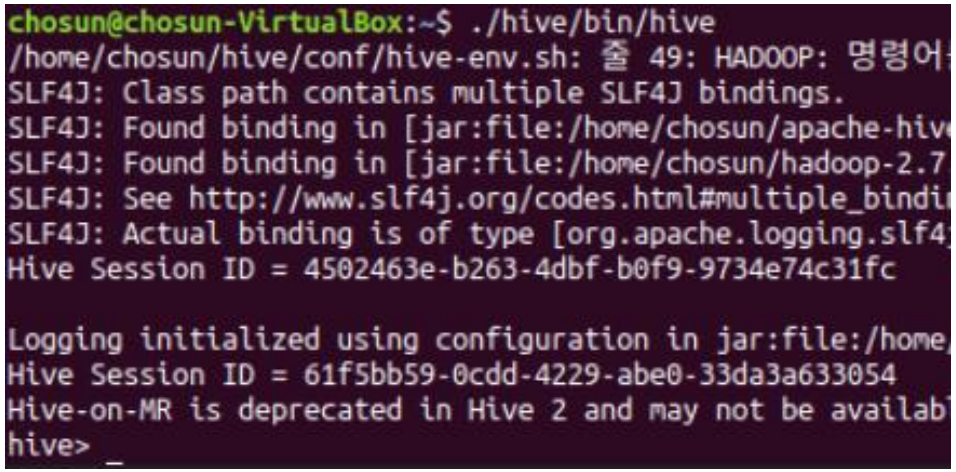
```
chosun@chosun-VirtualBox:~/hive/conf$
chosun@chosun-VirtualBox:~/hive/conf$ cd
chosun@chosun-VirtualBox:~$
chosun@chosun-VirtualBox:~$ ./hive/bin/schematool -initSchema -dbType derby
/home/chosun/hive/conf/hive-env.sh: 줄 49: HADOOP: 명령어를 찾을 수 없음
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/chosun/apache-hive-3.1.1-bin/lib/log4j-s
SLF4J: Found binding in [jar:file:/home/chosun/hadoop-2.7.2/share/hadoop/common/
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Metastore connection URL:        jdbc:derby:;databaseName=metastore_db;create=tr
Metastore Connection Driver :    org.apache.derby.jdbc.EmbeddedDriver
Metastore connection User:       APP
Starting metastore schema initialization to 3.1.0
Initialization script hive-schema-3.1.0.derby.sql

 ...

Initialization script completed
schemaTool completed
chosun@chosun-VirtualBox:~$
```

Hive 실행

Hive가 설치된 디렉토리에서 하이브 명령어를 실행합니다.

1. apache-hive-3.1.1-bin/bin에서 hive명령어를 실행했습니다.
   hive 명령어가 정상적으로 실행되는것을 확인했습니다.

```
chosun@chosun-VirtualBox:~$ ./hive/bin/hive
/home/chosun/hive/conf/hive-env.sh: 줄 49: HADOOP: 명령어를 찾을 수 없음
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/chosun/apache-hive-3.1.1-bin/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/chosun/hadoop-2.7.2/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Hive Session ID = 4502463e-b263-4dbf-b0f9-9734e74c31fc
Logging initialized using configuration in jar:file:/home/chosun/apache-hive-3.1.1-bin/lib/hive-common-3.1.1.jar!/hive-log4j2.properties Async: true
Hive Session ID = 61f5bb59-0cdd-4229-abe0-33da3a633054
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
hive>
```



Hive 실습

Hive 실행 – 데이터 준비

```
chosun@chosun-VirtualBox:~$ cd 다운로드/wk13
chosun@chosun-VirtualBox:~/다운로드/wk13$
chosun@chosun-VirtualBox:~/다운로드/wk13$ l
1987.csv                2019.05.27.Ch13.pdf  2019.06.03.ch17_hiveex.pdf  2019.6.5.Hive_databases.pdf  carriers_after.csv   ca
2019.05.22.ch13.ex.pdf  2019.05.27.ch17.pdf  2019.06.05.ch18_sqoop.pdf   airports_new.csv             carriers_before.csv  ch
chosun@chosun-VirtualBox:~/다운로드/wk13$
chosun@chosun-VirtualBox:~/다운로드/wk13$ mv airports_new.csv carriers_after.csv 1987.csv ../../hive
chosun@chosun-VirtualBox:~/다운로드/wk13$
chosun@chosun-VirtualBox:~/다운로드/wk13$ l
2019.05.22.ch13.ex.pdf  2019.05.27.ch17.pdf         2019.06.05.ch18_sqoop.pdf    carriers_before.csv  ch13_17_18_ex.pdf
2019.05.27.Ch13.pdf     2019.06.03.ch17_hiveex.pdf  2019.6.5.Hive_databases.pdf  carriers_new.csv
chosun@chosun-VirtualBox:~/다운로드/wk13$
chosun@chosun-VirtualBox:~/다운로드/wk13$ cd
chosun@chosun-VirtualBox:~$
chosun@chosun-VirtualBox:~$ cd hive
chosun@chosun-VirtualBox:~/hive$
chosun@chosun-VirtualBox:~/hive$ l
1987.csv  LICENSE  NOTICE  RELEASE_NOTES.txt  airports_new.csv  bin/  binary-package-licenses/  carriers_after.csv  conf/  ex
chosun@chosun-VirtualBox:~/hive$
chosun@chosun-VirtualBox:~/hive$ mv carriers_after.csv homework_carriers.csv
chosun@chosun-VirtualBox:~/hive$ mv airports_new.csv homework_airport.csv
chosun@chosun-VirtualBox:~/hive$
chosun@chosun-VirtualBox:~/hive$ sed -e '1389d' homework_carriers.csv > homework_carriers2.csv
chosun@chosun-VirtualBox:~/hive$
chosun@chosun-VirtualBox:~/hive$ l
1987.csv  NOTICE            bin/                      conf/      hcatalog/            homework_carriers.csv   jdbc/  script
LICENSE   RELEASE_NOTES.txt  binary-package-licenses/  examples/  homework_airport.csv  homework_carriers2.csv  lib/
chosun@chosun-VirtualBox:~/hive$
```

chosun@chosun-VirtualBox:~$ cd 다운로드/wk13

chosun@chosun-VirtualBox:~/다운로드/wk13$ l

1987.csv 2019.05.27.Ch13.pdf 2019.06.03.ch17_hiveex.pdf

carriers_new.csv 2019.05.22.ch13.ex.pdf 2019.05.27.ch17.pdf

carriers_before.csv ch13_17_18_ex.pdf

chosun@chosun-VirtualBox:~/다운로드/wk13$ mv airports_new.csv carriers_after.csv 1987.csv ../../hive

chosun@chosun-VirtualBox:~/다운로드/wk13$ l

1987.csv 2019.05.27.Ch13.pdf 2019.06.03.ch17_hiveex.pdf 2019.6.5.Hive_d

carriers_new.csv 2019.05.22.ch13.ex.pdf 2019.05.27.ch17.pdf 2019.06.05.c

carriers_before.csv ch13_17_18_ex.pdf

chosun@chosun-VirtualBox:~/다운로드/wk13$ cd

chosun@chosun-VirtualBox:~$ cd hive

chosun@chosun-VirtualBox:~/hive$ l

1987.csv LICENSE NOTICE RELEASE_NOTES.txt airports_new.csv bin/ binary-package-licenses/

carriers_after.csv conf/ examples/ hcatalog/ jdbc/ lib/ scripts/

chosun@chosun-VirtualBox:~/hive$ mv carriers_after.csv homework_carriers.csv

chosun@chosun-VirtualBox:~/hive$ mv airports_new.csv homework_airport.csv

chosun@chosun-VirtualBox:~/hive$ sed –e ‘1389d’ homework_carriers.csv > homework_carriers2.csv

chosun@chosun-VirtualBox:~/hive$ l

1987.csv LICENSE NOTICE RELEASE_NOTES.txt homework_airport.csv bin/ binary-package-licenses/

homework_carriers.csv homework_carriers2.csv conf/ examples/ hcatalog/ jdbc/ lib/ scripts/

Hive로 처리할 데이터를 준비합니다.

1. wk13에 있는 airports_new.csv, carriers_after.csv, 1987.csv 파일을 hive 디렉터리로 옮겼습니다.
2. carriers_after.csv, airports_new.csv 파일의 이름을 수정했습니다.
3. 외부 조인을 테스트하기 위해 homework_carriers.csv 파일의 코드 일부를 삭제했습니다.

Hive 실행 – 테이블 생성

테이블을 생성합니다.

1. 데이터를 조회하기 전에 CREATE TABLE과 같은 방식으로 테이블을 생성했습니다.
   HDFS는 저장된 파일에 직접 접근하지만, Hive는 메타스토어에 저장된 테이블을 분석하기 때문입니다.
2. show tables;를 입력해서 메타스토어 데이터베이스에 저장된 테이블 목록을 조회했습니다.

(a) PARTITIONED BY (~~~) 절

Hive는 쿼리문의 수행속도를 향상시키기 위해 파티션을 설정할 수 있습니다.

파티션을 설정하면 해당 테이블의 데이터를 파티션별로 디렉터리를 생성해서 저장하게 됩니다.

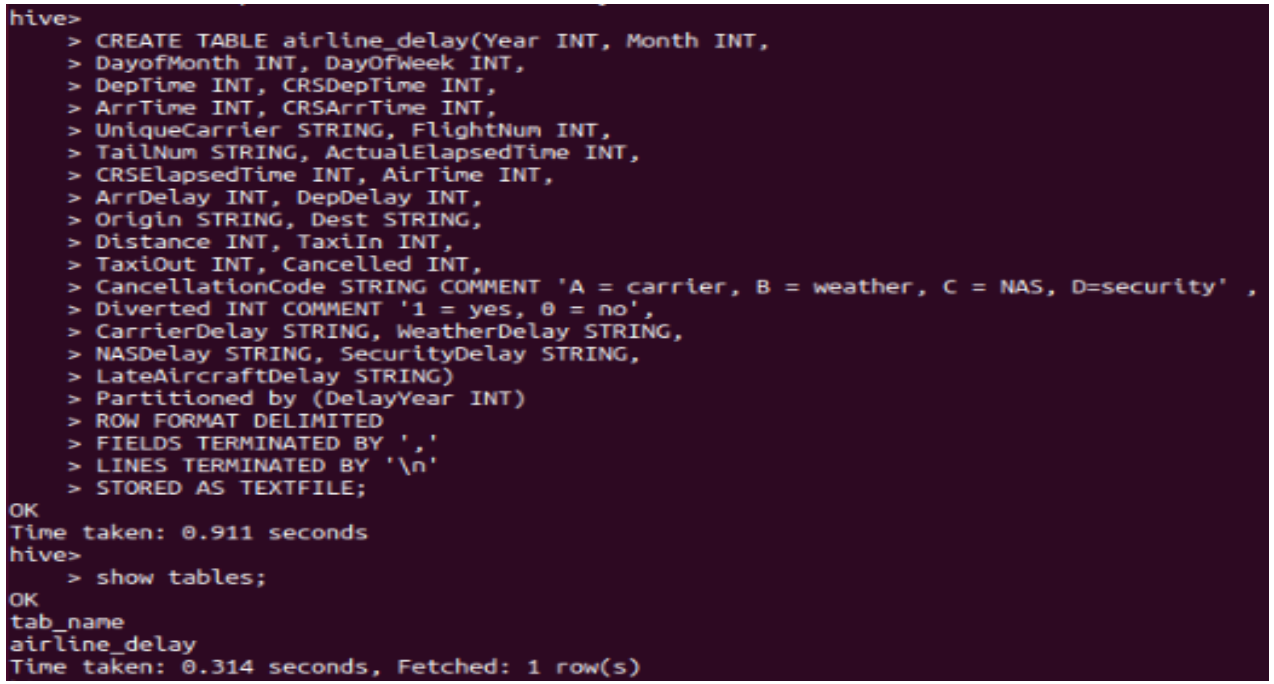
(b) ROW FORMAT 절

필드를 콤마 기준으로 구분하고, 행과 행은 ₩n값으로 구분합니다.

(c) STORED AS 절

Hive는 텍스트 파일을 위한 TEXTFILE과 시컨스 파일을 저장하기 위한 SEQUENCEFILE을 지원합니다.

```
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, TaxiIn INT,
> TaxiOut INT, Cancelled INT,
> CancellationCode STRING COMMENT 'A = carrier, B = weather, C = NAS, D=security' ,
> Diverted INT COMMENT '1 = yes, 0 = no',
> CarrierDelay STRING, WeatherDelay STRING,
> NASDelay STRING, SecurityDelay STRING,
> LateAircraftDelay STRING) ————————————————> 이후 구문은 부가적인 정보 설정
> Partitioned by (DelayYear INT) ——— > (a) 테이블의 파티션 설정
> ROW FORMAT DELIMITED ————> (b) 해당 테이블 내의 데이터가 어떻게 저장되는지 설정
> FIELDS TERMINATED BY ','
> LINES TERMINATED BY '₩n'
> STORED AS TEXTFILE; ———————— > (c) 데이터 저장 파일 포맷
OK
Time taken: 0.911 seconds
hive>
    > show tables;
OK
tab_name
airline_delay
Time taken: 0.314 seconds, Fetched: 1 row(s)
```



Hive 실행 – DESCRIBE

테이블의 칼럼(메타데이터)을 조회합니다.

1. describe 명령어를 이용해서 CREATE TABLE(..) 테이블 내에 있는 29개 칼럼과 파티션 칼럼인 delayYear가 모두 출력됩니다.

```
hive> hive>                                          //(1)
> describe airline_delay;
OK
col_name        data_type       comment
year            int
month           int
dayofmonth      int
dayofweek       int
deptime         int
crsdeptime      int
arrtime         int
```

```
crsarrtime      int                                  //(2)
uniquecarrier   string
flightnum       int
tailnum         string
actualelapsedtime   int
crselapsedtime  int
airtime         int
arrdelay        int
 depdelay       int
origin          string
dest            string
distance        int
taxiin          int
taxiout         int
```

```
cancelled       int                                  //(3)
cancellationcode  string A = carrier, B = weather, C = NAS, D=security
diverted        int             1 = yes, 0 = no
carrierdelay    string
weatherdelay    string
nasdelay        string
securitydelay   string
lateaircraftdelay               string
delayyear       int

# Partition Information
# col_name      data_type       comment
delayyear       int
Time taken: 0.49 seconds, Fetched: 34 row(s)
```

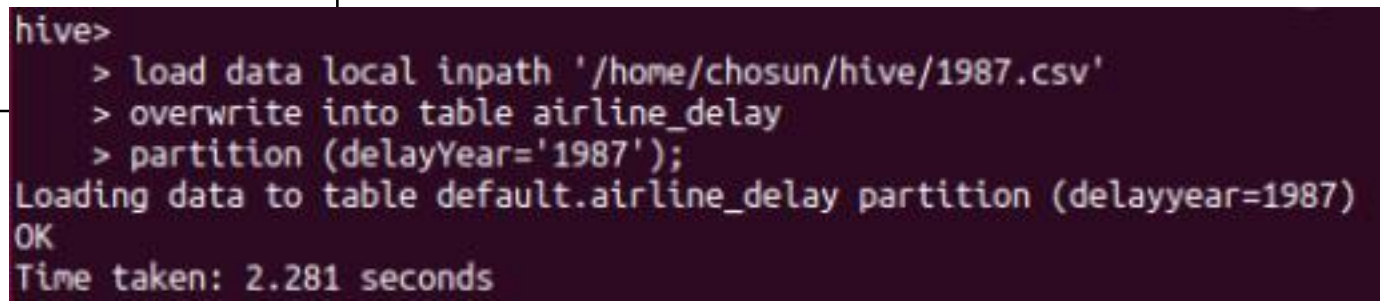
```
hive>
    > describe airline_delay;
OK
col_name        data_type       comment
year                    int
month                   int
dayofmonth              int
dayofweek               int
deptime                 int
crsdeptime              int
arrtime                 int
crsarrtime              int
uniquecarrier           string
flightnum               int
tailnum                 string
actualelapsedtime       int
crselapsedtime          int
airtime                 int
arrdelay                int
depdelay                int
origin                  string
dest                    string
distance                int
taxiin                  int
taxiout                 int
cancelled               int
cancellationcode        string          A = carrier, B = weather, C = NAS, D=security
diverted                int             1 = yes, 0 = no
carrierdelay            string
weatherdelay            string
nasdelay                string
securitydelay           string
lateaircraftdelay       string
delayyear               int

# Partition Information
# col_name              data_type               comment
delayyear               int
Time taken: 0.49 seconds, Fetched: 34 row(s)
```

Hive 실행 – 데이터 업로드

airline_delay 테이블에 데이터를 업로드 합니다.

1. 하이브는 로컬 파일 시스템에 있는 데이터와 HDFS에 저장된 데이터를 모두 저장할 수 있습니다.
(a) 테이블에는 파티션을 설정했는데, 테이블을 등록할 때 PARTITION 절을 선언하지 않으면
LOAD DATA 실행 시 오류가 발생합니다.

```
hive>
> load data local inpath '/home/chosun/hive/1987.csv'
> overwrite into table airline_delay ———> 중복된 데이터가 있어도 무시하고 입력
> partition (delayYear='1987'); ———— > (a) 파티션 키인 delayYear값을 1987로 설정해서 데이터를 입력
> Loading data to table default.airline_delay partition (delayyear=1987)
OK
Time taken: 2.281 seconds
```



Hive 실행 – 데이터 조회

airline_delay 테이블의 데이터를 조회합니다.

1. LOAD DATA가 실행되면 SELECT 쿼리문을 실행해서 데이터가 정상적으로 등록됐는지 확인합니다.
   SELECT 절의 기본 문법은 RDBMS의 SQL 문법과 유사합니다.
2. 원래 Hive는 질의를 실행하면 맵리듀스 잡을 실행했었는데,
   최근 버전의 Hive는 하나의 테이블을 limit 조건으로 조회하면 맵리듀스 잡을 실행하지 않고
   직접 파일을 조회해서 출력합니다.

```
hive>
> select year, month, deptime, arrtime, uniquecarrier, flightnum
> from airline_delay
> where delayYear = '1987'
> limit 10;
OK
year	month	deptime	arrtime	uniquecarrier	flightnum
NULL	NULL	NULL	NULL	UniqueCarrier	NULL
1987	10	741	912	PS	1451
1987	10	729	903	PS	1451
1987	10	741	918	PS	1451
1987	10	729	847	PS	1451
1987	10	749	922	PS	1451
1987	10	728	848	PS	1451
1987	10	728	852	PS	1451
1987	10	731	902	PS	1451
1987	10	744	908	PS	1451
Time taken: 2.141 seconds, Fetched: 10 row(s)
```

Hive 실행 – 도착 지연 건수 조회(집계함수)

연도와 월별로 도착 지역 건수를 조회합니다.

1. airline_delay 테이블의 데이터 중에서 GROUP BY를 사용해서 연도와 월별로 도착 지역 건수를 조회했습니다.
   미국 항공 운항 지연 데이터 중에서 1987년도의 도착 지역건수를 조회합니다.

```
hive>
    >
    > select year, month, count(*) as airline_delay_count
    > from airline_delay
    > where delayYear = '1987' and arrdelay > 0 group by year, month;
Query ID = chosun_20190605230829_c51b5e9d-e729-4b45-8214-12739513044b
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1559739087271_0001, Tracking URL = http://0.0.0.0:8089/proxy/application_1559739087271_0001/
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job  -kill job_1559739087271_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2019-06-05 23:08:40,311 Stage-1 map = 0%,  reduce = 0%
2019-06-05 23:08:50,041 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.42 sec
2019-06-05 23:08:55,213 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 5.43 sec
MapReduce Total cumulative CPU time: 5 seconds 430 msec
Ended Job = job_1559739087271_0001
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 5.43 sec   HDFS Read: 127183633 HDFS Write: 168 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 430 msec
OK
year    month   airline_delay_count
1987    10      265658
1987    11      255127
1987    12      287408
Time taken: 28.249 seconds, Fetched: 3 row(s)
```

Hive 실행 – 도착 지연 건수 조회(집계함수)

```
hive>                                                                    //(1)
> select year, month, count(*) as airline_delay_count
> from airline_delay
> where delayYear = '1987' and arrdelay > 0 group by year, month;
Query ID = chosun_20190605230829_c51b5e9d-e729-4b45-8214-12739513044b
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
set mapreduce.job.reduces=<number>
```

```
Starting Job = job_1559739087271_0001, Tracking URL = http://0.0.0.0:8089/proxy/application_          //(2)
1559739087271_0001/
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job -kill job_1559739087271_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2019-06-05 23:08:40,311 Stage-1 map = 0%, reduce = 0%
2019-06-05 23:08:50,041 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 3.42 sec
2019-06-05 23:08:55,213 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 5.43 sec
MapReduce Total cumulative CPU time: 5 seconds 430 msec
Ended Job = job_1559739087271_0001
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 5.43 sec HDFS Read: 127183633 HDFS Write: 168 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 430 msec
OK
year            month           airline_delay_count
1987            10              265658
1987            11              255127
1987            12              287408
Time taken: 28.249 seconds, Fetched: 3 row(s)
```

Hive 실행 – 평균 지연 시간 산출 (집계함수)

연도와 월별로 평균 지연 시간을 산출합니다.

1. airline_delay 테이블의 데이터 중에서 AVG 함수를 이용해 평균 지연 시간을 산출했습니다.
   1987년도의 평균 지연 시간을 연도와 월별로 계산하는 쿼리문이며, DOUBLE형태로 평균값이 출력됩니다.

```
hive>
    > select year, month, avg(arrdelay) as avg_airline_delay_time, avg(depdelay) as avg_departure_delay_time
    > from airline_delay
    > where delayYear = '1987' and arrdelay > 0 group by year, month;
Query ID = chosun_20190605231105_8eb6e934-436f-4d61-b1cc-6b19fb0253f9
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1559739087271_0002, Tracking URL = http://0.0.0.0:8089/proxy/application_1559739087271_0002/
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job  -kill job_1559739087271_0002
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2019-06-05 23:11:11,959 Stage-1 map = 0%,  reduce = 0%
2019-06-05 23:11:17,518 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.72 sec
2019-06-05 23:11:24,814 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 6.15 sec
MapReduce Total cumulative CPU time: 6 seconds 150 msec
Ended Job = job_1559739087271_0002
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 6.15 sec   HDFS Read: 127185172 HDFS Write: 260 SUCCESS
Total MapReduce CPU Time Spent: 6 seconds 150 msec
OK
year    month   avg_airline_delay_time  avg_departure_delay_time
1987    10      14.154563386007574      8.187481649338624
1987    11      17.786212356982993      11.327370290090817
1987    12      23.840985637143017      17.536394254857207
Time taken: 20.412 seconds, Fetched: 3 row(s)
```

Hive 실행 – 평균 지연 시간 산출 (집계함수)

```
hive>                                                                                    //(1)
> select year, month, avg(arrdelay) as avg_airline_delay_time, avg(depdelay) as avg_departure_delay_time
> from airline_delay
> where delayYear = '1987' and arrdelay > 0 group by year, month;
Query ID = chosun_20190605231105_8eb6e934-436f-4d61-b1cc-6b19fb0253f9
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
set mapreduce.job.reduces=<number>
```

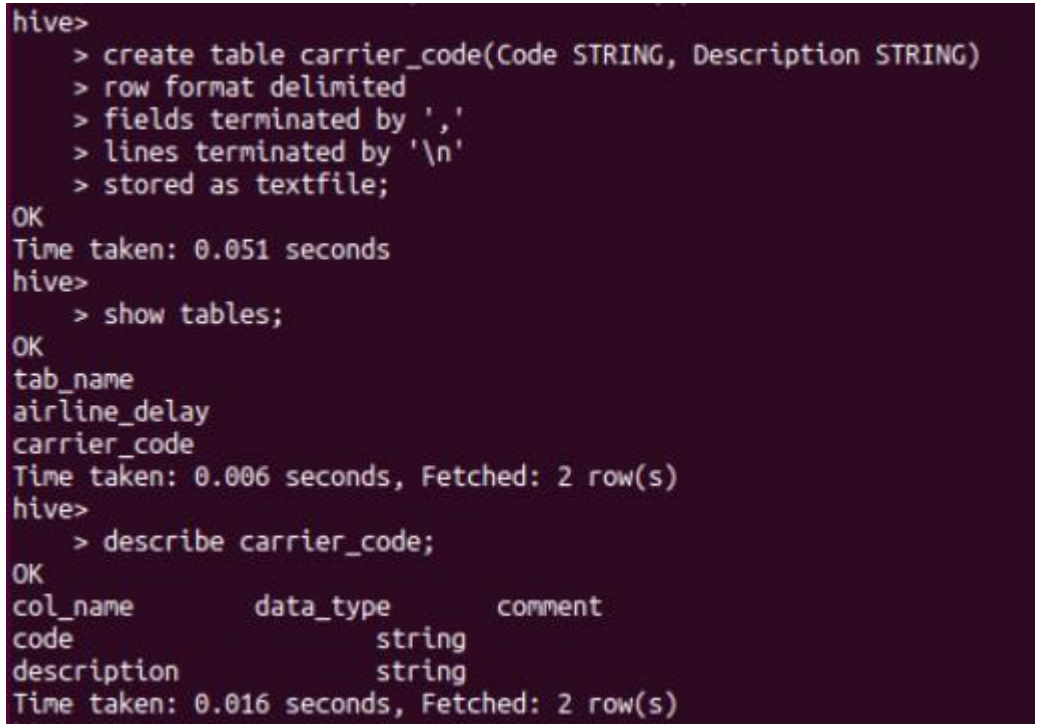
```
Starting Job = job_1559739087271_0002, Tracking URL = http://0.0.0.0:8089/proxy/application_1559739087271_0002/
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job -kill job_1559739087271_0002                    //(2)
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2019-06-05 23:11:11,959 Stage-1 map = 0%, reduce = 0%
2019-06-05 23:11:17,518 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 3.72 sec
2019-06-05 23:11:24,814 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 6.15 sec
MapReduce Total cumulative CPU time: 6 seconds 150 msec
Ended Job = job_1559739087271_0002
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 6.15 sec HDFS Read: 127185172 HDFS Write: 260 SUCCESS
Total MapReduce CPU Time Spent: 6 seconds 150 msec
OK
year        month       avg_airline_delay_time      avg_departure_delay_time
1987        10          14.154563386007574          8.187481649338624
1987        11          17.786212356982993          11.327370290090817
1987        12          23.840985637143017          17.536394254857207
Time taken: 20.412 seconds, Fetched: 3 row(s)
```

Hive 실행 – 테이블 생성 (내부 조인)

테이블을 생성합니다.

1. 항공사 코드 데이터를 저장하기 위한 테이블은 두 개의 문자열로 구성하며, 필드와 라인의 구분은 airline_delay와 동일하게 설정했습니다.
2. 테이블과 메타데이터를 조회합니다.

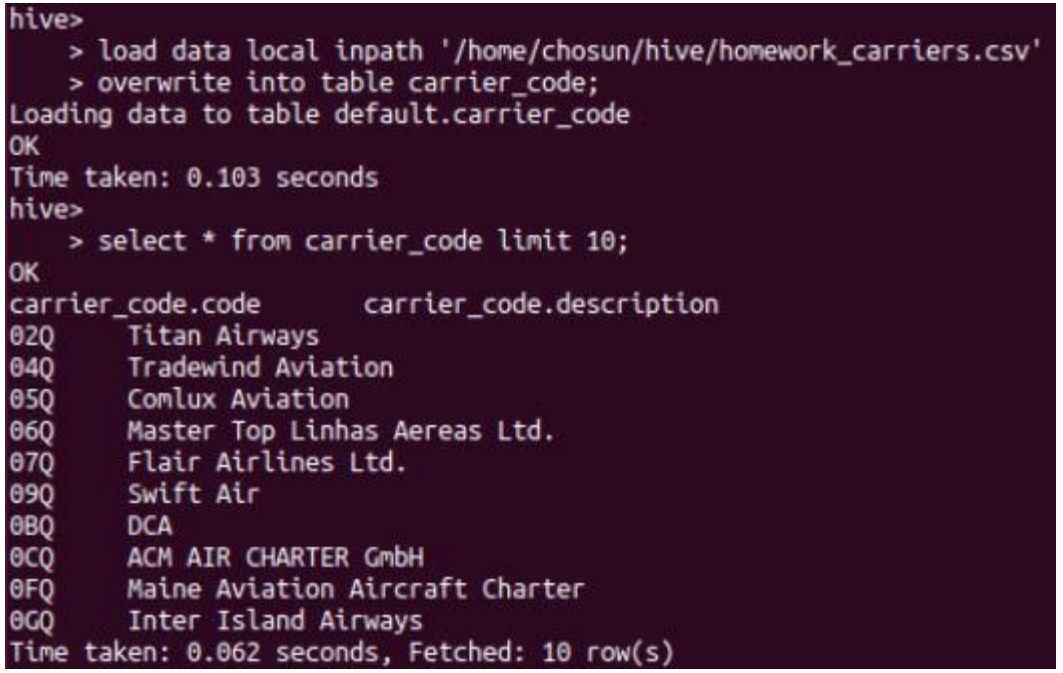
```
hive>
> create table carrier_code(Code STRING, Description STRING)
> row format delimited
> fields terminated by ','
> lines terminated by '₩n'
> stored as textfile;
OK
Time taken: 0.051 seconds
hive>
> show tables;
OK
tab_name
airline_delay
carrier_code
Time taken: 0.006 seconds, Fetched: 2 row(s)
hive>
> describe carrier_code;
OK
col_name        data_type       comment
code            string
description     string
Time taken: 0.016 seconds, Fetched: 2 row(s)
```



Hive 실행 – 데이터 업로드/조회 (내부 조인)

```
hive>
> load data local inpath '/home/chosun/hive/homework_carriers.csv'
> overwrite into table carrier_code;
Loading data to table default.carrier_code
OK
Time taken: 0.103 seconds
hive>
> select * from carrier_code limit 10;
OK
carrier_code.code               carrier_code.description
02Q             Titan Airways
04Q             Tradewind Aviation
05Q             Comlux Aviation
06Q             Master Top Linhas Aereas Ltd.
07Q             Flair Airlines Ltd.
09Q             Swift Air
0BQ             DCA
0CQ             ACM AIR CHARTER GmbH
0FQ             Maine Aviation Aircraft Charter
0GQ             Inter Island Airways
Time taken: 0.062 seconds, Fetched: 10 row(s)
```

1. 따옴표를 제거한 데이터를 업로드 합니다.
2. 샘플로 코드 테이블에서 10건의 데이터를 조회했습니다.



Hive 실행 – 두 테이블의 항공사 코드가 일치하는 데이터만 조회(내부 조인)

1. ON 키워드를 사용해서 조인을 걸었습니다.
2. 두 테이블의 조인키인 항공사 코드로 내부 조인을 처리합니다.
   즉, 두 항공사 코드가 일치하는 데이터만 조회합니다.
3. 쿼리문을 완료하면 SELECT문에 선언한 대로 각 칼럼값과 건수 합계가 출력됩니다.

```
hive>
    > select A.year,A.uniquecarrier,B.description,count(*)
    > from airline_delay A join carrier_code B on (A.uniquecarrier = B.code)
    > where a.arrdelay > 0
    > group by A.year,A.uniquecarrier,B.description;
Query ID = chosun_20190605231415_e3184da4-0f96-481b-9d7f-f969926b0478
Total jobs = 1
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
2019-06-05 23:14:20     Starting to launch local task to process map join;      maximum memory = 477626368
2019-06-05 23:14:21     Uploaded 1 File to: file:/tmp/chosun/4502463e-b263-4dbf-b0f9-9734e74c31fc/hive_2019-06-0
Execution completed successfully
MapredLocal task succeeded
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1559739087271_0003, Tracking URL = http://0.0.0.0:8089/proxy/application_1559739087271_0003/
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job  -kill job_1559739087271_0003
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2019-06-05 23:14:27,417 Stage-2 map = 0%,  reduce = 0%
2019-06-05 23:14:33,928 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 4.44 sec
2019-06-05 23:14:40,167 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 6.27 sec
MapReduce Total cumulative CPU time: 6 seconds 270 msec
Ended Job = job_1559739087271_0003
MapReduce Jobs Launched:
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 6.27 sec   HDFS Read: 127188530 HDFS Write: 911 SUCCESS
Total MapReduce CPU Time Spent: 6 seconds 270 msec
OK
a.year  a.uniquecarrier b.description   _c3
1987    AA      American Airlines Inc.  83421
1987    AS      Alaska Airlines Inc.    15316
1987    CO      Continental Air Lines Inc.      62300
1987    DL      Delta Air Lines Inc.    142189
1987    EA      Eastern Air Lines Inc.  60018
1987    HP      America West Airlines Inc. (Merged with US Airways 9/05. Stopped reporting 10/07.)      32585
1987    NW      Northwest Airlines Inc. 74688
1987    PA (1)  Pan American World Airways (1)  9264
1987    PI      Piedmont Aviation Inc.  80570
1987    PS      Pacific Southwest Airlines      31140
1987    TW      Trans World Airways LLC 42510
1987    UA      United Air Lines Inc.   81104
1987    US      US Airways Inc. (Merged with America West 9/05. Reporting for both starting 10/07.)     63959
1987    WN      Southwest Airlines Co.  29129
Time taken: 25.626 seconds, Fetched: 14 row(s)
```

```
hive>
> select A.year,A.uniquecarrier,B.description,count(*)
> from airline_delay A join carrier_code B on (A.uniquecarrier = B.code) ———> ON 키워드를 사용해서 조인
> where a.arrdelay > 0
> group by A.year,A.uniquecarrier,B.description;
Query ID = chosun_20190605231415_e3184da4-0f96-481b-9d7f-f969926b0478
Total jobs = 1
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
2019-06-05 23:14:20          Starting to launch local task to process map join;          maximum memory = 477626368
2019-06-05 23:14:21          Uploaded 1 File to: file:/tmp/chosun/4502463e-b263-4dbf-b0f9-9734e74c31fc/hive
_2019-06-05_23-14-15_638_3800843140763170630-1/-local-10005/HashTable-Stage-2/MapJoin-mapfile01--.hashtable
(63412 bytes)
Execution completed successfully
MapredLocal task succeeded
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
set mapreduce.job.reduces=<number>
Starting Job = job_1559739087271_0003, Tracking URL = http://0.0.0.0:8089/proxy/application_
1559739087271_0003/
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job -kill job_1559739087271_0003
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2019-06-05 23:14:27,417 Stage-2 map = 0%, reduce = 0%
```

Hive 실행 –
두 테이블의 항공사 코드가
일치하는 데이터만 조회(내부 조인)

Hive 실행 –
두 테이블의 항공사 코드가
일치하는 데이터만 조회(내부 조인)

```
2019-06-05 23:14:33,928 Stage-2 map = 100%, reduce = 0%, Cumulative CPU 4.44 sec
2019-06-05 23:14:40,167 Stage-2 map = 100%, reduce = 100%, Cumulative CPU 6.27 sec
MapReduce Total cumulative CPU time: 6 seconds 270 msec
Ended Job = job_1559739087271_0003
MapReduce Jobs Launched:
Stage-Stage-2: Map: 1 Reduce: 1 Cumulative CPU: 6.27 sec HDFS Read: 127188530 HDFS Write: 911 SUCCESS
Total MapReduce CPU Time Spent: 6 seconds 270 msec
OK
a.year          a.uniquecarrier b.description    _c3
1987            AA              American Airlines Inc.          83421
1987            AS              Alaska Airlines Inc.            15316
1987            CO              Continental Air Lines Inc.      62300
1987            DL              Delta Air Lines Inc.            142189
1987            EA              Eastern Air Lines Inc.          60018
1987            HP              America West Airlines Inc. (Merged with US Airways 9/05. Stopped reporting 10/07.)      32585
1987            NW              Northwest Airlines Inc.         74688
1987            PA (1)          Pan American World Airways (1)          9264
1987            PI              Piedmont Aviation Inc.          80570
1987            PS              Pacific Southwest Airlines      31140
1987            TW              Trans World Airways LLC         42510
1987            UA              United Air Lines Inc.           81104
1987            US              US Airways Inc. (Merged with America West 9/05. Reporting for both starting 10/07.)      63959
1987            WN              Southwest Airlines Co.          29129
Time taken: 25.626 seconds, Fetched: 14 row(s)
```

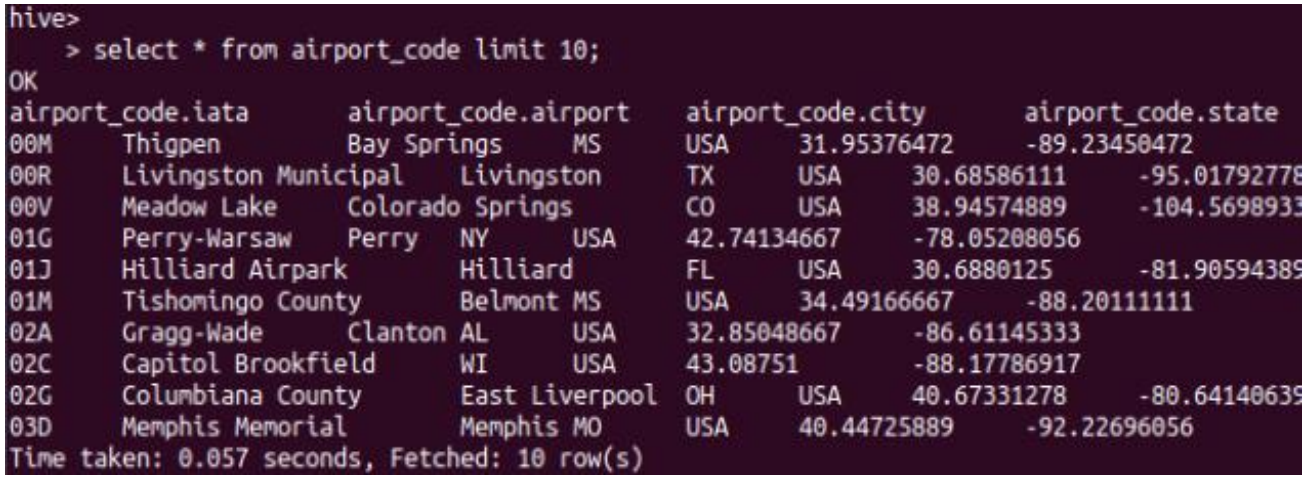
Hive 실행 – 테이블 생성/데이터 업로드 (내부 조인)

```
hive>
> CREATE TABLE airport_code(Iata string, Airport STRING, City STRING, State STRING, Country STRING,
> Lat double, Longitude double)
> ROW FORMAT DELIMITED
> FIELDS TERMINATED BY ','
> LINES TERMINATED BY '₩n'
> STORED AS TEXTFILE;
OK
Time taken: 0.03 seconds
hive>
> LOAD DATA LOCAL INPATH '/home/chosun/hive/homework_airport.csv'
> OVERWRITE INTO TABLE airport_code;
Loading data to table default.airport_code
OK
Time taken: 0.092 seconds
```

1. 미국 내 공항 정보를 저장할 테이블을 생성하고 데이터를 업로드 했습니다.

Hive 실행 – 데이터 조회 (내부 조인)

```
hive>
> select * from airport_code limit 10;
OK
airport_code.iata	airport_code.airport	airport_code.city	airport_code.state	airport_code.country	airport_code.lat	airport_code.longitude
00M	Thigpen	Bay Springs	MS	USA	31.95376472	-89.23450472
00R	Livingston Municipal	Livingston	TX	USA	30.68586111	-95.01792778
00V	Meadow Lake	Colorado Springs	CO	USA	38.94574889	-104.5698933
01G	Perry-Warsaw	Perry	NY	USA	42.74134667	-78.05208056
01J	Hilliard Airpark	Hilliard	FL	USA	30.6880125	-81.90594389
01M	Tishomingo County	Belmont	MS	USA	34.49166667	-88.20111111
02A	Gragg-Wade	Clanton	AL	USA	32.85048667	-86.61145333
02C	Capitol	Brookfield	WI	USA	43.08751	-88.17786917
02G	Columbiana County	East Liverpool	OH	USA	40.67331278	-80.64140639
03D	Memphis Memorial	Memphis	MO	USA	40.44725889	-92.22696056
Time taken: 0.057 seconds, Fetched: 10 row(s)
```



1. airport_code 테이블에서 10건의 데이터를 조회했습니다.

Hive 실행 – 공항별 지연 건수 계산 (내부 조인)

```
hive>
    > SELECT A.Year, A.Origin, B.Airport, A.dest, C.Airport, Count(*)
    > FROM airline_delay A
    > join airport_code B on (A.origin = B.Iata)
    > join airport_code C on (A.dest = C.Iata)
    > where a.arrdelay > 0
    > group by A.year,A.origin,B.airport,A.dest,C.airport;
No Stats for default@airline_delay, Columns: arrdelay, year, origin, dest
No Stats for default@airport_code, Columns: iata, airport
No Stats for default@airport_code, Columns: iata, airport
Query ID = chosun_20190605234454_c0a42667-4377-4b5d-85b5-3e823687acb3
Total jobs = 1
SLF4J: Found binding in [jar:file:/home/chosun/apache-hive-3.1.1-bin/lib/log4j-slf4j-impl-2.10.0.jar!/org/s
SLF4J: Found binding in [jar:file:/home/chosun/hadoop-2.7.2/share/hadoop/common/lib/slf4j-log4j12-1.7.10.ja
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
2019-06-05 23:45:01     Starting to launch local task to process map join;      maximum memory = 477626368
Execution completed successfully
MapredLocal task succeeded
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1559739087271_0004, Tracking URL = http://0.0.0.0:8089/proxy/application_1559739087271_0004/
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job  -kill job_1559739087271_0004
Hadoop job information for Stage-3: number of mappers: 1; number of reducers: 1
2019-06-05 23:45:10,596 Stage-3 map = 0%,  reduce = 0%
2019-06-05 23:45:20,020 Stage-3 map = 100%,  reduce = 0%, Cumulative CPU 6.18 sec
2019-06-05 23:45:26,261 Stage-3 map = 100%,  reduce = 100%, Cumulative CPU 8.52 sec
MapReduce Total cumulative CPU time: 8 seconds 520 msec
Ended Job = job_1559739087271_0004
MapReduce Jobs Launched:
Stage-Stage-3: Map: 1  Reduce: 1   Cumulative CPU: 8.52 sec   HDFS Read: 127193257 HDFS Write: 268838 SUCCESS
Total MapReduce CPU Time Spent: 8 seconds 520 msec
OK
a.year  a.origin        b.airport       a.dest  c.airport       _c5
1987    ABE     Lehigh Valley International     ATL     William B Hartsfield-Atlanta Intl       91
1987    ABE     Lehigh Valley International     AVP     Wilkes-Barre/Scranton Intl      60
1987    ABE     Lehigh Valley International     DTW     Detroit Metropolitan-Wayne County       142
 ...

1987    YKM     Yakima Air Terminal     PSC     Tri-Cities      102
1987    YUM     Yuma MCAS-Yuma International    LAS     McCarran International  6
1987    YUM     Yuma MCAS-Yuma International    PHX     Phoenix Sky Harbor International        234
Time taken: 32.668 seconds, Fetched: 3442 row(s)
```

1. ON 키워드를 사용해서 조인을 걸었습니다.
2. airport_cord 테이블에 공항 데이터 등록이 완료되면,
   두 개의 테이블을 조인하는 쿼리를 실행합니다.
3. 이 쿼리문은 출발 공항 코드와 도착 공항 코드, airport_code
   테이블을 조인해 공항별 지연 건수를 계산합니다.
4. 쿼리문이 실행되면 출발 공항과 도착 공항별 지연 횟수가
   조회됩니다.

Hive 실행 – 공항별 지연 건수 계산 (내부 조인)

```
hive>
> SELECT A.Year, A.Origin, B.Airport, A.dest, C.Airport, Count(*)
> FROM airline_delay A
> join airport_code B on (A.origin = B.Iata) ———> ON 키워드를 사용해서 조인
> join airport_code C on (A.dest = C.Iata) ———> ON 키워드를 사용해서 조인
> where a.arrdelay > 0
> group by A.year,A.origin,B.airport,A.dest,C.airport;
No Stats for default@airline_delay, Columns: arrdelay, year, origin, dest
No Stats for default@airport_code, Columns: iata, airport
No Stats for default@airport_code, Columns: iata, airport
Query ID = chosun_20190605234454_c0a42667-4377-4b5d-85b5-3e823687acb3
Total jobs = 1
SLF4J: Found binding in [jar:file:/home/chosun/apache-hive-3.1.1-bin/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/chosun/hadoop-2.7.2/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
2019-06-05 23:45:01          Starting to launch local task to process map join;          maximum memory = 477626368
Execution completed successfully
MapredLocal task succeeded
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
```

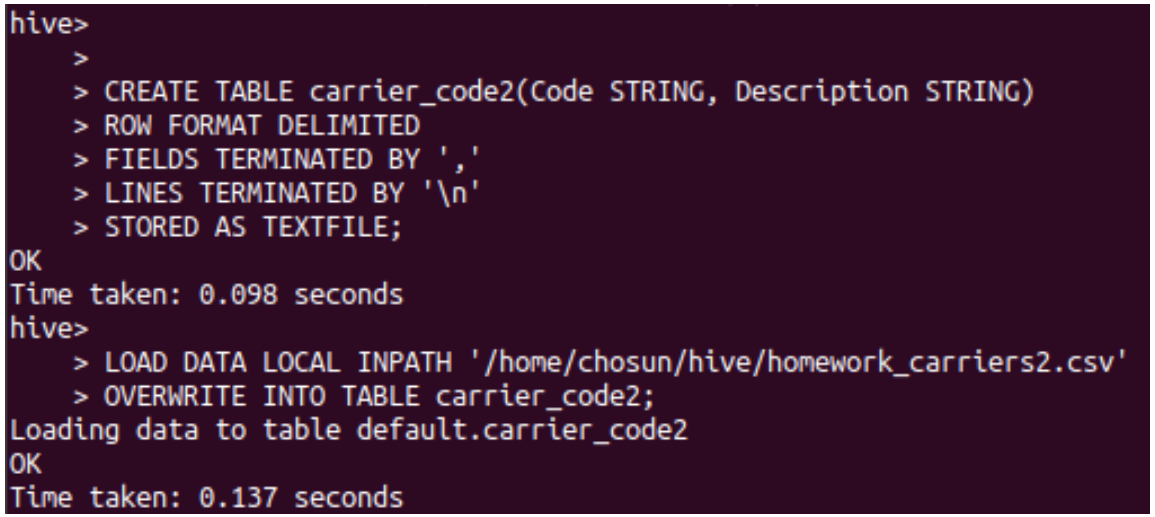
Hive 실행 – 공항별 지연 건수 계산 (내부 조인)

```
set mapreduce.job.reduces=<number>
Starting Job = job_1559739087271_0004, Tracking URL = http://0.0.0.0:8089/proxy/application_1559739087271_0004/
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job -kill job_1559739087271_0004
Hadoop job information for Stage-3: number of mappers: 1; number of reducers: 1
2019-06-05 23:45:10,596 Stage-3 map = 0%, reduce = 0%
2019-06-05 23:45:20,020 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 6.18 sec
2019-06-05 23:45:26,261 Stage-3 map = 100%, reduce = 100%, Cumulative CPU 8.52 sec
MapReduce Total cumulative CPU time: 8 seconds 520 msec
Ended Job = job_1559739087271_0004
MapReduce Jobs Launched:
Stage-Stage-3: Map: 1 Reduce: 1 Cumulative CPU: 8.52 sec HDFS Read: 127193257 HDFS Write: 268838 SUCCESS
Total MapReduce CPU Time Spent: 8 seconds 520 msec
OK
a.year  a.origin  b.airport  a.dest  c.airport  _c5
1987  ABE  Lehigh Valley International  ATL  William B Hartsfield-Atlanta Intl  91
1987  ABE  Lehigh Valley International  AVP  Wilkes-Barre/Scranton Intl  60
1987  ABE  Lehigh Valley International  DTW  Detroit Metropolitan-Wayne County  142

...

1987  YAP  Yap International  ROR  Babelthoup/Koror  13
1987  YKM  Yakima Air Terminal  PSC  Tri-Cities  102
1987  YUM  Yuma MCAS-Yuma International  LAS  McCarran International  6
1987  YUM  Yuma MCAS-Yuma International  PHX  Phoenix Sky Harbor International  234
Time taken: 32.668 seconds, Fetched: 3442 row(s)
```

Hive 실행 – 테이블 생성/데이터 업로드(외부 조인)

```
hive>
>
> CREATE TABLE carrier_code2(Code STRING, Description STRING)
> ROW FORMAT DELIMITED
> FIELDS TERMINATED BY ','
> LINES TERMINATED BY '₩n'
> STORED AS TEXTFILE;
OK
Time taken: 0.098 seconds
hive>
> LOAD DATA LOCAL INPATH '/home/chosun/hive/homework_carriers2.csv'
> OVERWRITE INTO TABLE carrier_code2;
Loading data to table default.carrier_code2
OK
Time taken: 0.137 seconds
```

1. 하이브 메타스토어 데이터베이스에 항공사 코드 테이블 carrier_code2을 추가로 생성했습니다.
2. carrier_code2 테이블에 WN코드를 삭제한 homework_carriers2.csv 파일을 업로드 했습니다.



Hive 실행 – 외부 조인 테스트

```
hive>
    > SELECT A.Year, A.UniqueCarrier, B.Code, B.Description
    > FROM airline_delay A
    > LEFT OUTER JOIN carrier_code2 B ON (A.UniqueCarrier = B.Code)
    > WHERE A.UniqueCarrier = 'WN'
    > LIMIT 10;
Warning: Map Join MAPJOIN[14][bigTable=?] in task 'Stage-3:MAPRED' is a cross product
Query ID = chosun_20190605235117_2f87d5a0-c6b3-445f-b0a7-56efa9cfdee0
Total jobs = 1
SLF4J: Found binding in [jar:file:/home/chosun/apache-hive-3.1.1-bin/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/im
SLF4J: Found binding in [jar:file:/home/chosun/hadoop-2.7.2/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
2019-06-05 23:51:23     Uploaded 1 File to: file:/tmp/chosun/4502463e-b263-4dbf-b0f9-9734e74c31fc/hive_2019-06-05_
2019-06-05 23:51:23     End of local task; Time Taken: 1.408 sec.
Execution completed successfully
MapredLocal task succeeded
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1559739087271_0005, Tracking URL = http://0.0.0.0:8089/proxy/application_1559739087271_0005/
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job  -kill job_1559739087271_0005
Hadoop job information for Stage-3: number of mappers: 1; number of reducers: 0
2019-06-05 23:51:30,531 Stage-3 map = 0%,  reduce = 0%
2019-06-05 23:51:35,773 Stage-3 map = 100%,  reduce = 0%, Cumulative CPU 2.48 sec
MapReduce Total cumulative CPU time: 2 seconds 480 msec
Ended Job = job_1559739087271_0005
MapReduce Jobs Launched:
Stage-Stage-3: Map: 1   Cumulative CPU: 2.48 sec   HDFS Read: 9560534 HDFS Write: 347 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 480 msec
OK
a.year  a.uniquecarrier b.code  b.description
1987    WN      NULL    NULL
1987    WN      NULL    NULL
1987    WN      NULL    NULL
1987    WN      NULL    NULL
1987    WN      NULL    NULL
1987    WN      NULL    NULL
1987    WN      NULL    NULL
1987    WN      NULL    NULL
1987    WN      NULL    NULL
1987    WN      NULL    NULL
Time taken: 19.714 seconds, Fetched: 10 row(s)
```

1. ON 키워드를 사용해서 왼쪽 외부 조인을 걸었습니다.
   왼쪽 외부 조인 쿼리를 사용하면 조인할 carrier_code2 테이블에 WN 코드가 없더라도 airline_delay 테이블에서 WN 코드가 등록된 데이터를 모두 출력합니다.
2. airline_delay 테이블에서 항공사 코드가 WN인 데이터를 출력했지만, carrier_code2 테이블에는 WN코드 데이터가 존재하지 않기 때문에 NULL로 출력된 것을 확인했습니다.

Hive 실행 – 외부 조인 테스트

```
hive>                                                                    //(1)
> SELECT A.Year, A.UniqueCarrier, B.Code, B.Description
> FROM airline_delay A
> LEFT OUTER JOIN carrier_code2 B ON (A.UniqueCarrier = B.Code)
> WHERE A.UniqueCarrier = 'WN'
> LIMIT 10;
Warning: Map Join MAPJOIN[14][bigTable=?] in task 'Stage-3:MAPRED' is a cross product
Query ID = chosun_20190605235117_2f87d5a0-c6b3-445f-b0a7-56efa9cfdee0
Total jobs = 1
SLF4J: Found binding in [jar:file:/home/chosun/apache-hive-3.1.1-
bin/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/chosun/hadoop-2.7.2/share/hadoop
/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
2019-06-05 23:51:23 Uploaded 1 File to: file:/tmp/chosun/4502463e-
b263-4dbf-b0f9-9734e74c31fc/hive_
2019-06-05_23-51-17_137_9045315568461493482-1/-local-10004/HashTable-
Stage-3/MapJoin-mapfile31--.hashtable (260 bytes)
2019-06-05 23:51:23          End of local task; Time Taken: 1.408 sec.
Execution completed successfully
MapredLocal task succeeded
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_1559739087271_0005, Tracking URL = http://0.0.0.0:8089/
proxy/application_1559739087271_0005/
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job -kill job_1559739087271_0005
```

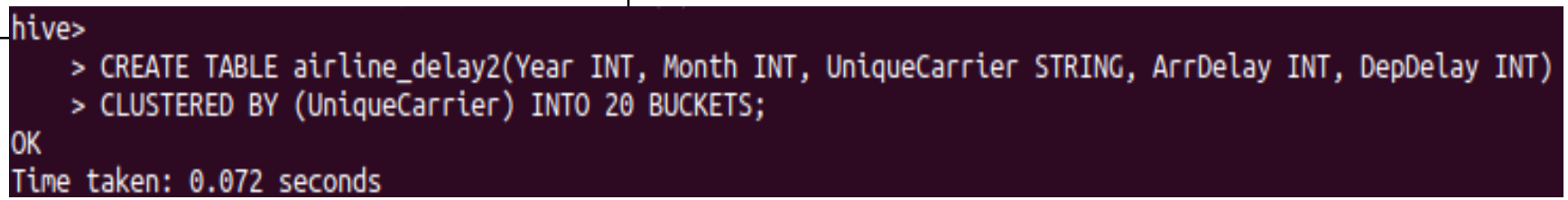
```
Hadoop job information for Stage-3: number of mappers: 1; number of reducers: 0          //(2)
2019-06-05 23:51:30,531 Stage-3 map = 0%, reduce = 0%
2019-06-05 23:51:35,773 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 2.48 sec
MapReduce Total cumulative CPU time: 2 seconds 480 msec
Ended Job = job_1559739087271_0005
MapReduce Jobs Launched:
Stage-Stage-3: Map: 1 Cumulative CPU: 2.48 sec HDFS Read: 9560534 HDFS Write: 347 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 480 msec
OK
a.year          a.uniquecarrier b.code          b.description
1987            WN              NULL            NULL
1987            WN              NULL            NULL
1987            WN              NULL            NULL
1987            WN              NULL            NULL
1987            WN              NULL            NULL
1987            WN              NULL            NULL
1987            WN              NULL            NULL
1987            WN              NULL            NULL
1987            WN              NULL            NULL
1987            WN              NULL            NULL
Time taken: 19.714 seconds, Fetched: 10 row(s)
```

Hive 실행 – 테이블 생성 (버킷 활용)

1. 버킷을 사용해서 airline_delay2 테이블을 생성했습니다.

   버킷을 활용하면 효율적인 쿼리문 수행을 할 수 있습니다.
   버킷은 버킷 칼럼 해시를 기준으로 데이터를 지정된 개수의 파일로 분리해서 저장합니다.
   테이블을 생성할 때 CLUSTERED BY (칼럼) INTO (버킷 개수) BUCKETS; 형태로 선언합니다.

```
hive>
> CREATE TABLE airline_delay2(Year INT, Month INT, UniqueCarrier STRING, ArrDelay INT, DepDelay INT)
> CLUSTERED BY (UniqueCarrier) INTO 20 BUCKETS;
OK
Time taken: 0.072 seconds
```



Hive 실행 – 데이터를 테이블에 등록 (버킷 활용)

1. 1987년도 항공 운항 지연 데이터를 새로운 테이블 airline_delay2에 등록했습니다.

```
hive>
    > INSERT OVERWRITE TABLE airline_delay2
    > SELECT Year, Month, UniqueCarrier, ArrDelay, DepDelay
    > FROM airline_delay
    > WHERE delayYear = 1987;
Query ID = chosun_20190605235205_96054016-6449-420a-92c5-d22f1157d461
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks determined at compile time: 20
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1559739087271_0006, Tracking URL = http://0.0.0.0:8089/proxy/applica
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job  -kill job_1559739087271_0006
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 20
2019-06-05 23:52:11,130 Stage-1 map = 0%,  reduce = 0%
2019-06-05 23:52:18,785 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 5.19 sec
2019-06-05 23:52:30,498 Stage-1 map = 100%,  reduce = 5%, Cumulative CPU 7.1 sec
2019-06-05 23:52:31,556 Stage-1 map = 100%,  reduce = 10%, Cumulative CPU 10.6 sec
2019-06-05 23:52:32,601 Stage-1 map = 100%,  reduce = 15%, Cumulative CPU 12.07 sec
2019-06-05 23:52:41,983 Stage-1 map = 100%,  reduce = 30%, Cumulative CPU 20.92 sec
2019-06-05 23:52:48,891 Stage-1 map = 100%,  reduce = 33%, Cumulative CPU 23.39 sec
2019-06-05 23:54:01,134 Stage-1 map = 100%,  reduce = 33%, Cumulative CPU 23.39 sec
2019-06-05 23:54:02,910 Stage-1 map = 100%,  reduce = 32%, Cumulative CPU 41.74 sec
2019-06-05 23:54:04,149 Stage-1 map = 100%,  reduce = 40%, Cumulative CPU 52.51 sec
2019-06-05 23:54:10,989 Stage-1 map = 100%,  reduce = 45%, Cumulative CPU 54.45 sec
2019-06-05 23:54:12,033 Stage-1 map = 100%,  reduce = 55%, Cumulative CPU 59.82 sec
2019-06-05 23:54:13,101 Stage-1 map = 100%,  reduce = 60%, Cumulative CPU 62.35 sec
2019-06-05 23:54:14,137 Stage-1 map = 100%,  reduce = 65%, Cumulative CPU 63.87 sec
2019-06-05 23:54:15,249 Stage-1 map = 100%,  reduce = 70%, Cumulative CPU 66.48 sec
2019-06-05 23:54:23,517 Stage-1 map = 100%,  reduce = 80%, Cumulative CPU 72.5 sec
2019-06-05 23:54:27,205 Stage-1 map = 100%,  reduce = 92%, Cumulative CPU 78.47 sec
2019-06-05 23:54:29,316 Stage-1 map = 100%,  reduce = 97%, Cumulative CPU 81.47 sec
2019-06-05 23:54:30,359 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 84.4 sec
MapReduce Total cumulative CPU time: 1 minutes 24 seconds 400 msec
Ended Job = job_1559739087271_0006
Loading data to table default.airline_delay2
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1559739087271_0007, Tracking URL = http://0.0.0.0:8089/proxy/application_1559739087271_0007/
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job  -kill job_1559739087271_0007
Hadoop job information for Stage-3: number of mappers: 1; number of reducers: 1
2019-06-05 23:54:41,057 Stage-3 map = 0%,  reduce = 0%
2019-06-05 23:54:45,243 Stage-3 map = 100%,  reduce = 0%, Cumulative CPU 1.54 sec
2019-06-05 23:54:50,415 Stage-3 map = 100%,  reduce = 100%, Cumulative CPU 3.14 sec
MapReduce Total cumulative CPU time: 3 seconds 140 msec
Ended Job = job_1559739087271_0007
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 20   Cumulative CPU: 84.4 sec   HDFS Read: 127381550 HDFS Write: 21304923 SUCCESS
Stage-Stage-3: Map: 1  Reduce: 1   Cumulative CPU: 3.14 sec   HDFS Read: 43417 HDFS Write: 2997 SUCCESS
Total MapReduce CPU Time Spent: 1 minutes 27 seconds 540 msec
OK
year    month   uniquecarrier   arrdelay        depdelay
Time taken: 166.565 seconds
```

Hive 실행 – 데이터를 테이블에 등록 (버킷 활용)

```
hive>                                                                    // (1)
> INSERT OVERWRITE TABLE airline_delay2
> SELECT Year, Month, UniqueCarrier, ArrDelay, DepDelay
> FROM airline_delay
> WHERE delayYear = 1987;
Query ID = chosun_20190605235205_96054016-6449-420a-92c5-d22f1157d461
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks determined at compile time: 20
In order to change the average load for a reducer (in bytes):
set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
set mapreduce.job.reduces=<number>
Starting Job = job_1559739087271_0006, Tracking URL = http://0.0.0.0:8089/proxy/application_1559739087271_0006/
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job -kill job_1559739087271_0006
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 20
2019-06-05 23:52:11,130 Stage-1 map = 0%, reduce = 0%
2019-06-05 23:52:18,785 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 5.19 sec
2019-06-05 23:52:30,498 Stage-1 map = 100%, reduce = 5%, Cumulative CPU 7.1 sec
2019-06-05 23:52:31,556 Stage-1 map = 100%, reduce = 10%, Cumulative CPU 10.6 sec
2019-06-05 23:52:32,601 Stage-1 map = 100%, reduce = 15%, Cumulative CPU 12.07 sec
2019-06-05 23:52:41,983 Stage-1 map = 100%, reduce = 30%, Cumulative CPU 20.92 sec
2019-06-05 23:52:48,891 Stage-1 map = 100%, reduce = 33%, Cumulative CPU 23.39 sec
2019-06-05 23:54:01,134 Stage-1 map = 100%, reduce = 33%, Cumulative CPU 23.39 sec
2019-06-05 23:54:02,910 Stage-1 map = 100%, reduce = 32%, Cumulative CPU 41.74 sec
```

Hive 실행 – 데이터를 테이블에 등록 (버킷 활용)

//(2)

```
2019-06-05 23:54:04,149 Stage-1 map = 100%, reduce = 40%, Cumulative CPU 52.51 sec
2019-06-05 23:54:10,989 Stage-1 map = 100%, reduce = 45%, Cumulative CPU 54.45 sec
2019-06-05 23:54:12,033 Stage-1 map = 100%, reduce = 55%, Cumulative CPU 59.82 sec
2019-06-05 23:54:13,101 Stage-1 map = 100%, reduce = 60%, Cumulative CPU 62.35 sec
2019-06-05 23:54:14,137 Stage-1 map = 100%, reduce = 65%, Cumulative CPU 63.87 sec
2019-06-05 23:54:15,249 Stage-1 map = 100%, reduce = 70%, Cumulative CPU 66.48 sec
2019-06-05 23:54:23,517 Stage-1 map = 100%, reduce = 80%, Cumulative CPU 72.5 sec
2019-06-05 23:54:27,205 Stage-1 map = 100%, reduce = 92%, Cumulative CPU 78.47 sec
2019-06-05 23:54:29,316 Stage-1 map = 100%, reduce = 97%, Cumulative CPU 81.47 sec
2019-06-05 23:54:30,359 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 84.4 sec
MapReduce Total cumulative CPU time: 1 minutes 24 seconds 400 msec
Ended Job = job_1559739087271_0006
Loading data to table default.airline_delay2
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
set mapreduce.job.reduces=<number>
Starting Job = job_1559739087271_0007, Tracking URL =
http://0.0.0.0:8089/proxy/application_1559739087271_0007/
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job -kill job_1559739087271_0007
```

//(3)

```
Hadoop job information for Stage-3: number of mappers: 1; number of reducers: 1
2019-06-05 23:54:41,057 Stage-3 map = 0%, reduce = 0%
2019-06-05 23:54:45,243 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 1.54 sec
2019-06-05 23:54:50,415 Stage-3 map = 100%, reduce = 100%, Cumulative CPU 3.14 sec
MapReduce Total cumulative CPU time: 3 seconds 140 msec
2019-06-05 23:54:45,243 Stage-3 map = 100%, reduce = 0%, Cumulative CPU 1.54 sec
2019-06-05 23:54:50,415 Stage-3 map = 100%, reduce = 100%, Cumulative CPU 3.14 sec
MapReduce Total cumulative CPU time: 3 seconds 140 msec
Ended Job = job_1559739087271_0007
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 20 Cumulative CPU: 84.4 sec
HDFS Read: 127381550 HDFS Write: 21304923 SUCCESS
Stage-Stage-3: Map: 1 Reduce: 1 Cumulative CPU: 3.14 sec
HDFS Read: 43417 HDFS Write: 2997 SUCCESS
Total MapReduce CPU Time Spent: 1 minutes 27 seconds 540 msec
OK
year            month           uniquecarrier   arrdelay        depdelay
Time taken: 166.565 seconds
```

Hive 실행 – 하이브 웨어하우스 디렉터리 조회 (버킷 활용)

1. HDFS의 하이브 웨어하우스 디렉터리의 airline_delay2 디렉터리에서 20개의 파일이 생성된 것을 확인했습니다.
   이 때 파일 000000_0 ~ 000019_0 은 모두 버킷을 나타냅니다.

```
hive>
    > dfs -ls /user/hive/warehouse/airline_delay2;
Found 20 items
-rw-r--r--   1 chosun supergroup    5212308 2019-06-05 23:52 /user/hive/warehouse/airline_delay2/000000_0
-rw-r--r--   1 chosun supergroup          0 2019-06-05 23:52 /user/hive/warehouse/airline_delay2/000001_0
-rw-r--r--   1 chosun supergroup          0 2019-06-05 23:52 /user/hive/warehouse/airline_delay2/000002_0
-rw-r--r--   1 chosun supergroup     677199 2019-06-05 23:52 /user/hive/warehouse/airline_delay2/000003_0
-rw-r--r--   1 chosun supergroup    4699869 2019-06-05 23:52 /user/hive/warehouse/airline_delay2/000004_0
-rw-r--r--   1 chosun supergroup    1735187 2019-06-05 23:52 /user/hive/warehouse/airline_delay2/000005_0
-rw-r--r--   1 chosun supergroup          0 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000006_0
-rw-r--r--   1 chosun supergroup          0 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000007_0
-rw-r--r--   1 chosun supergroup          0 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000008_0
-rw-r--r--   1 chosun supergroup     336860 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000009_0
-rw-r--r--   1 chosun supergroup          0 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000010_0
-rw-r--r--   1 chosun supergroup    1128072 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000011_0
-rw-r--r--   1 chosun supergroup     719243 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000012_0
-rw-r--r--   1 chosun supergroup    1881748 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000013_0
-rw-r--r--   1 chosun supergroup          0 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000014_0
-rw-r--r--   1 chosun supergroup     994797 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000015_0
-rw-r--r--   1 chosun supergroup    2018623 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000016_0
-rw-r--r--   1 chosun supergroup          0 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000017_0
-rw-r--r--   1 chosun supergroup    1874288 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000018_0
-rw-r--r--   1 chosun supergroup         26 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000019_0
```

Hive 실행 – 하이브 웨어하우스 디렉터리 조회 (버킷 활용)

```
hive>
> dfs -ls /user/hive/warehouse/airline_delay2;
Found 20 items
-rw-r--r-- 1 chosun supergroup 5212308 2019-06-05 23:52 /user/hive/warehouse/airline_delay2/000000_0
-rw-r--r-- 1 chosun supergroup 0 2019-06-05 23:52 /user/hive/warehouse/airline_delay2/000001_0
-rw-r--r-- 1 chosun supergroup 0 2019-06-05 23:52 /user/hive/warehouse/airline_delay2/000002_0
-rw-r--r-- 1 chosun supergroup 677199 2019-06-05 23:52 /user/hive/warehouse/airline_delay2/000003_0
-rw-r--r-- 1 chosun supergroup 4699869 2019-06-05 23:52 /user/hive/warehouse/airline_delay2/000004_0
-rw-r--r-- 1 chosun supergroup 1735187 2019-06-05 23:52 /user/hive/warehouse/airline_delay2/000005_0
-rw-r--r-- 1 chosun supergroup 0 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000006_0
-rw-r--r-- 1 chosun supergroup 0 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000007_0
-rw-r--r-- 1 chosun supergroup 0 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000008_0
-rw-r--r-- 1 chosun supergroup 336860 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000009_0
-rw-r--r-- 1 chosun supergroup 0 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000010_0
-rw-r--r-- 1 chosun supergroup 1128072 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000011_0
-rw-r--r-- 1 chosun supergroup 719243 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000012_0
-rw-r--r-- 1 chosun supergroup 1881748 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000013_0
-rw-r--r-- 1 chosun supergroup 0 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000014_0
-rw-r--r-- 1 chosun supergroup 994797 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000015_0
-rw-r--r-- 1 chosun supergroup 2018623 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000016_0
-rw-r--r-- 1 chosun supergroup 0 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000017_0
-rw-r--r-- 1 chosun supergroup 1874288 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000018_0
-rw-r--r-- 1 chosun supergroup 26 2019-06-05 23:54 /user/hive/warehouse/airline_delay2/000019_0
```

Hive 실행 – 버킷 조회 (버킷 활용)

1. airline_delay를 조회했을때와 항공사별 합계 건수가 다르게 조회되는데, 이는 파일 전체를 조회하지 않고 첫 번째 버킷만 조회했기 때문입니다.
2. 버킷을 활용하면 샘플용 데이터를 조회할 때 크게 도움이 됩니다.
   대신 하둡에서 너무 작은 파일을 많이 처리하게 되면 부하가 발생하므로, 버킷을 사용할 때는 버킷이 너무 작은 크기로 만들어지지 않게 주의해야 합니다.
3. 버킷은 사용자가 생각하는 샘플 데이터와 크기가 같거나 작아야합니다.

```
hive>
    > SELECT UniqueCarrier, COUNT(*)
    > FROM airline_delay2
    > TABLESAMPLE(BUCKET 1 OUT OF 20)
    > GROUP BY UniqueCarrier;
Query ID = chosun_20190605235547_a7eafa64-d398-4f1a-9172-5d36f3a59443
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1559739087271_0008, Tracking URL = http://0.0.0.0:8089/proxy/application_1559739087271
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job  -kill job_1559739087271_0008
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2019-06-05 23:55:51,384 Stage-1 map = 0%,  reduce = 0%
2019-06-05 23:55:57,596 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.64 sec
2019-06-05 23:56:01,738 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 5.42 sec
MapReduce Total cumulative CPU time: 5 seconds 420 msec
Ended Job = job_1559739087271_0008
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 5.42 sec   HDFS Read: 21294965 HDFS Write: 87 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 420 msec
OK
uniquecarrier   _c1
Time taken: 15.191 seconds
```

Hive 실행 – 버킷 조회 (버킷 활용)

```
hive>                                                                    //(1)
> SELECT UniqueCarrier, COUNT(*)
> FROM airline_delay2
> TABLESAMPLE(BUCKET 1 OUT OF 20)
> GROUP BY UniqueCarrier;
Query ID = chosun_20190605235547_a7eafa64-d398-4f1a-9172-5d36f3a59443
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
set hive.exec.reducers.max=<number>
```

```
In order to set a constant number of reducers:                           //(2)
set mapreduce.job.reduces=<number>
Starting Job = job_1559739087271_0008, Tracking URL = http://0.0.0.0:8089/proxy/application_1559739087271_0008/
Kill Command = /home/chosun/hadoop-2.7.2/bin/mapred job -kill job_1559739087271_0008
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2019-06-05 23:55:51,384 Stage-1 map = 0%, reduce = 0%
2019-06-05 23:55:57,596 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 3.64 sec
2019-06-05 23:56:01,738 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 5.42 sec
MapReduce Total cumulative CPU time: 5 seconds 420 msec
Ended Job = job_1559739087271_0008
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 5.42 sec HDFS Read: 21294965 HDFS Write: 87 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 420 msec
OK
uniquecarrier    _c1
Time taken: 15.191 seconds
```

Hive 설치 & 실습 소감

실습 소감은 음… 제가 2년전에 정보처리기능사 책을 볼 기회가 있었는데 그 때 살짝 공부했었던 SQL문이 생각나서 하둡을 사용했던 것보다 재미..있었습니다.
그리고 이번 과제가 (아마?) 마지막일텐데, 한 학기동안 빅데이터 과목 들으면서 느꼈던 점을 써보겠습니다!

처음 아무것도 모른채로 하둡을 배웠을 때 교수님이 교재 여기저기 넘어다니면서 수업하시는게
솔직히 말해서 도대체 가만히 듣고 있어도 무슨 소리 하는건지 모르겠고,,,,,, 모른채로 그냥 무작정 종이에 써서 집에가서 봐도 모르겠고,,,,,,
그런 와중에 저만 빼고 전부 수업 따라가고 있는 것 같아서 집에서 열심히 명령어 외우고 실습 해보고! 해도 모르겠고,,,,,,
머리가 나빠서 그런가 싶어서 무슨 뜻인지도 모르고 열심히 외우기만 했던 날들,,,,,, 아무튼 이런 이유때문에 빅데이터 과목 든 날은 무조건 밤샜던 기억만 납니다!
(전날 과제 못끝내고 수업들으면 진짜로 못따라가니까요!!! 그리고 사실 이번 과제도 현재 진행형입니다. 과제 그만..!! 밤새는거 제발그만..!!)

근데 좀 놀란게 중간고사 보고나서 음 나만 못따라가는게 아니었군!! 다같이 모르는거였어~! 하는 마음이 들어서 안심(?) 하고
다음 수업을 들었는데 여전히 뭐 배우고 있는지 모르겠지만 대충 뭘 해야겠는지는 알아먹겠는거에요!
오오~~스스로 감탄하면서 명령어도 많이 외워져서 거침없이 (여전히 밤을 새가며,,,,,,) 과제를 해나가는 중에
이전 과제에서 자바 소스를 짜라고 하셨을땐 아아,,, 내가 스스로 할 줄 알았던게 아니라 따라하는 법을 알았구나! 라고 다시 한번 알았습니다!
방학때 열심히 자바 공부,,,하겠습니다!

마지막으로, 원래 지금까지는 과제 제출을 hwp에 작성해서 제출했었는데, 이번 과제는 로그가 너무 길어서 ppt를 사용해서 작성했습니다.
그런데 작성하고 나서 보니까 ppt가 왠지 hwp보다 깔끔하고 만들기 편한 듯 했습니다,,, 진작 이렇게 할걸 싶었습니다.
음 아직 수업 한 번 남았지만,,, 교수님 한 학기 수업 고생하셨습니다!! (_ _)