Ejercicios de Hive

SHOW TABLES; para ver todas las tablas que tenemos

1. Modificar la propiedad correspondiente para mostrar por pantalla las cabeceras de las tablas.

Lo primero es abrir el fichero de configuración de hive, y dentro de ahi, poner la propiedad hive.cli.print.header.

```
  <property>
    <name>hive.cli.print.header</name>
    <value>true</value>
    <description>Whether to print the names of the columns in query output.</de$
  </property>
```

Luego reinicio hive para que los cambios surjan efecto

```
[cloudera@quickstart ~]$ sudo service hive-server2 restart
Stopped Hive Server2:                                      [  OK  ]
Started Hive Server2 (hive-server2):                       [  OK  ]
```

2. Crear la base de datos “ejerciciosMasterDB”.

Inicializo la CLI de Hive.

```
[cloudera@quickstart ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.p
roperties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive>
```

Creo la BBDD

```
hive> CREATE DATABASE ejerciciosMasterDB;
OK
Time taken: 1.449 seconds
```

3. Seleccionar nuestra BBDD “ejerciciosMasterDB”.

```
hive> USE ejerciciosmasterdb;
OK
Time taken: 0.072 seconds
```

4. Comprobar el contenido de nuestra BBDD.

```
hive> SHOW tables;
OK
tab_name
Time taken: 0.038 seconds
```

CREACIÓN DE TABLAS Y CARGA DE DATOS

1. Crear una tabla llamada empleados con los siguientes campos:

- **a. Nombre tipo STRING.**
- **b. Apellidos tipo STRING.**
- **c. Edad tipo INT.**
- **d. Género tipo STRING.**

e. Salario tipo INT.

Los datos a cargar se encuentran en un archivo .txt, los campos están separados por tabuladores y cada registro separado por salto de línea.

```
hive> CREATE TABLE empleados (
    >     Nombre STRING,
    >     Apellidos STRING,
    >     Edad INT,
    >     Genero STRING,
    >     Salario INT
    > )
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY '\t'
    > LINES TERMINATED BY '\n';
OK
Time taken: 0.594 seconds
```

2. Comprueba la información de la tabla usando DESCRIBE, DESCRIBE EXTENDED Y DESCRIBE FORMATTED.

DESCRIBE nos da una descripción básica de la estructura de la tabla, mostrando los nombres de las columnas y sus tipos de datos.

```
hive> DESCRIBE empleados;
OK
col_name        data_type       comment
nombre                  string
apellidos               string
edad                    int
genero                  string
salario                 int
Time taken: 0.164 seconds, Fetched: 5 row(s)
```

DESCRIBE EXTEND nos da una descripción extendida de la tabla, incluyendo información adicional como la ubicación de los datos, el formato de las filas y columnas, y las estadísticas básicas.

```
hive> DESCRIBE EXTENDED empleados;
OK
col_name        data_type       comment
nombre                  string
apellidos               string
edad                    int
genero                  string
salario                 int

Detailed Table Information      Table(tableName:empleados, dbName:default, owner
:cloudera, createTime:1720690195, lastAccessTime:0, retention:0, sd:StorageDescr
iptor(cols:[FieldSchema(name:nombre, type:string, comment:null), FieldSchema(nam
e:apellidos, type:string, comment:null), FieldSchema(name:edad, type:int, commen
t:null), FieldSchema(name:genero, type:string, comment:null), FieldSchema(name:s
alario, type:int, comment:null)], location:hdfs://quickstart.cloudera:8020/user/
hive/warehouse/empleados, inputFormat:org.apache.hadoop.mapred.TextInputFormat,
outputFormat:org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat, compres
sed:false, numBuckets:-1, serdeInfo:SerDeInfo(name:null, serializationLib:org.ap
ache.hadoop.hive.serde2.lazy.LazySimpleSerDe, parameters:{serialization.format=,
 line.delim=
, field.delim=  }), bucketCols:[], sortCols:[], parameters:{}, skewedInfo:Skewed
Info(skewedColNames:[], skewedColValues:[], skewedColValueLocationMaps:{}), stor
edAsSubDirectories:false), partitionKeys:[], parameters:{transient_lastDdlTime=1
720690195}, viewOriginalText:null, viewExpandedText:null, tableType:MANAGED_TABL
E)
Time taken: 0.08 seconds, Fetched: 8 row(s)
```

DESCRIBED FORMATTED nos da una descripción aún más detallada de la tabla, incluyendo información sobre las propiedades, estadísticas de columnas, ubicación en HDFS, y más.

```
hive> DESCRIBE FORMATTED empleados;
OK
col_name         data_type       comment
# col_name              data_type               comment

nombre                  string
apellidos               string
edad                    int
genero                  string
salario                 int

# Detailed Table Information
Database:               default
Owner:                  cloudera
CreateTime:             Thu Jul 11 11:29:55 CEST 2024
LastAccessTime:         UNKNOWN
Protect Mode:           None
Retention:              0
Location:               hdfs://quickstart.cloudera:8020/user/hive/warehouse/empleados
Table Type:             MANAGED_TABLE
Table Parameters:
        transient_lastDdlTime   1720690195

# Storage Information
SerDe Library:          org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
InputFormat:            org.apache.hadoop.mapred.TextInputFormat
OutputFormat:           org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Compressed:             No
Num Buckets:            -1
Bucket Columns:         []
Sort Columns:           []
Storage Desc Params:
        field.delim             \t
        line.delim              \n
        serialization.format    \t
Time taken: 0.085 seconds, Fetched: 32 row(s)
```

3. Carga los datos del fichero empl_tb.txt

Como no existe el fichero, he decidido crearlo.

```
Santi   Gil     21      H       50000
Miguel  Angel   40      H       50000
Mario   Sainz   20      H       21000
Diego   Ramirez 22      H       10000
Miguel  Orden   21      H       40000
SantiagoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA Gil     22      H       50000
```

Una vez creado, paso a la inserción de datosPara la inserción de datos he tenido problemas debido a los tabs y los enter, pero al final lo he solucionado bien:

```
hive> LOAD DATA LOCAL INPATH '/home/cloudera/Desktop/empl_tb.txt' INTO TABLE empleados;

Loading data to table default.empleados
Table default.empleados stats: [numFiles=2, totalSize=212]
OK
Time taken: 0.164 seconds
```

4. Comprueba con una consulta sencilla que los datos se han cargado correctamente.

```
hive> select * from empleados;
OK
empleados.nombre        empleados.apellidos     empleados.edad  empleados.genero
mpleados.salario
Santiago        Gil     21      H       50000
Miguel  Angel   40      H       50000
Mario   Sainz   20      G       21000
Santi   Gil     21      H       50000
Miguel  Angel   40      H       50000
Mario   Sainz   20      H       21000
Diego   Ramirez 22      H       10000
Miguel  Orden   21      H       40000
Santiago        Gil     22      H       50000
Time taken: 0.042 seconds, Fetched: 9 row(s)
```

a

5. Crea el directorio ejercicios_master en HDFS e importa un el archivo “empl_tb.txt” a ese directorio.

He creado el directorio

```
[cloudera@quickstart ~]$ hadoop fs -mkdir /user/cloudera/ejercicios master
```

Meto el fichero:

```
[cloudera@quickstart ~]$ hdfs dfs -put ~/Desktop/empl_tb2.txt /user/cloudera/ejercicios
_master
```

Se ve aqui:

```
[cloudera@quickstart ~]$ hadoop fs -ls -R /user/cloudera/ejercicios_master
-rw-r--r--   1 cloudera cloudera        131 2024-07-11 13:46 /user/cloudera/ejercicios_
master/empl_tb2.txt
```

6. Carga los datos del archivo empl_tb2.txt a nuestra tabla de empleados;

Cabe destacar que NO HAY QUE PONER LOCAL al cargar desde HDFS.

```
hive> LOAD DATA INPATH '/user/cloudera/ejercicios_master/empl_tb2.txt' INTO TABLE emple
ados;
Loading data to table default.empleados
Table default.empleados stats: [numFiles=3, totalSize=343]
OK
Time taken: 0.715 seconds
```

Vemos como queda la tabla ahora…

```
hive> select * from empleados;
OK
empleados.nombre        empleados.apellidos     empleados.edad  empleados.genero
mpleados.salario
Santiago        Gil     21      H       50000
Miguel  Angel   40      H       50000
Mario   Sainz   20      G       21000
Pablo   Loa     21      H       10000
Sergio  Chava   20      H       200
Javi    Julian  25      H       25000
Pedro   Ariz    23      H       130
Isaac   Garrido 19      H       1000
Marta   Doncel  27      M       50
Santi   Gil     21      H       50000
Miguel  Angel   40      H       50000
Mario   Sainz   20      H       21000
Diego   Ramirez 22      H       10000
Miguel  Orden   21      H       40000
Santiago        Gil     22      H       50000
Time taken: 0.383 seconds, Fetched: 15 row(s)
```

CONSULTAS SENCILLAS

7. Selecciona los empleados con genero == “Hombre”.

```
hive> select * from empleados where genero = 'H';
OK
empleados.nombre        empleados.apellidos     empleados.edad  empleados.genero      e
mpleados.salario
Santiago        Gil     21      H       50000
Miguel  Angel   40      H       50000
Pablo   Loa     21      H       10000
Sergio  Chava   20      H       200
Javi    Julian  25      H       25000
Pedro   Ariz    23      H       130
Isaac   Garrido 19      H       1000
Santi   Gil     21      H       50000
Miguel  Angel   40      H       50000
Mario   Sainz   20      H       21000
Diego   Ramirez 22      H       10000
Miguel  Orden   21      H       40000
Santiago        Gil     22      H       50000
Time taken: 0.162 seconds, Fetched: 13 row(s)
```

8. Calcula la media de los empleados mujeres.

Se ha calculado que la media es 50.0

```
hive> select avg(salario) from empleados where genero = 'M';
Query ID = cloudera_20240711141010_5a14e2df-6b65-4a25-ab3b-774646b7a8dc
Total jobs = 1
Launching Job 1 out of 1
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_1720685678318_0001, Tracking URL = http://quickstart.cloudera:8088/p
roxy/application_1720685678318_0001/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1720685678318_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2024-07-11 14:10:14,506 Stage-1 map = 0%,  reduce = 0%
2024-07-11 14:10:22,311 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.41 sec
2024-07-11 14:10:30,032 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 2.87 sec
MapReduce Total cumulative CPU time: 2 seconds 870 msec
Ended Job = job_1720685678318_0001
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 2.87 sec   HDFS Read: 9696 HDFS Writ
e: 5 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 870 msec
OK
_c0
50.0
Time taken: 27.112 seconds, Fetched: 1 row(s)
```

9. Calcula la suma de los empleados hombres.

La media es de 357330:

```
hive> select sum(salario) from empleados where genero = 'H';
Query ID = cloudera_20240711141111_62601832-2b78-4885-a608-e76c1de935a5
Total jobs = 1
Launching Job 1 out of 1
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_1720685678318_0002, Tracking URL = http://quickstart.cloudera:8088/p
roxy/application_1720685678318_0002/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1720685678318_0002
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2024-07-11 14:12:06,417 Stage-1 map = 0%,  reduce = 0%
2024-07-11 14:12:12,941 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.37 sec
2024-07-11 14:12:21,519 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 2.65 sec
MapReduce Total cumulative CPU time: 2 seconds 650 msec
Ended Job = job_1720685678318_0002
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 2.65 sec   HDFS Read: 9297 HDFS Writ
e: 7 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 650 msec
OK
_c0
357330
Time taken: 25.495 seconds, Fetched: 1 row(s)
```

10. El empleado con el sueldo más alto

Cabe destacar que en vez de ‘top 1’, como se podría poner en otras herramientas, en hive se usa ‘limit 1’:

```
hive> select * from empleados order by salario desc limit 1;
Query ID = cloudera_20240711141414_10748ce0-8347-46bd-a666-8cf37351c0b9
Total jobs = 1
Launching Job 1 out of 1
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_1720685678318_0003, Tracking URL = http://quickstart.cloudera:8088/p
roxy/application_1720685678318_0003/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1720685678318_0003
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2024-07-11 14:14:59,018 Stage-1 map = 0%,  reduce = 0%
2024-07-11 14:15:06,551 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.0 sec
2024-07-11 14:15:13,917 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 2.29 sec
MapReduce Total cumulative CPU time: 2 seconds 290 msec
Ended Job = job_1720685678318_0003
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 2.29 sec   HDFS Read: 8539 HDFS Writ
e: 24 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 290 msec
OK
empleados.nombre        empleados.apellidos     empleados.edad  empleados.genero      e
mpleados.salario
Santiago        Gil     21      H       50000
Time taken: 23.124 seconds, Fetched: 1 row(s)
```

11. El empleado con el sueldo más bajo.

```
hive> select * from empleados order by salario asc limit 1;
Query ID = cloudera_20240711141717_ae71341d-21b3-488d-83af-5d5d10b64a10
Total jobs = 1
Launching Job 1 out of 1
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_1720685678318_0004, Tracking URL = http://quickstart.cloudera:8088/p
roxy/application_1720685678318_0004/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1720685678318_0004
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2024-07-11 14:18:00,304 Stage-1 map = 0%,  reduce = 0%
2024-07-11 14:18:06,710 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.9 sec
2024-07-11 14:18:15,249 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 2.06 sec
MapReduce Total cumulative CPU time: 2 seconds 60 msec
Ended Job = job_1720685678318_0004
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 2.06 sec   HDFS Read: 8539 HDFS Writ
e: 21 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 60 msec
OK
empleados.nombre        empleados.apellidos     empleados.edad  empleados.genero      e
mpleados.salario
Marta   Doncel  27      M       50
Time taken: 23.455 seconds, Fetched: 1 row(s)
```

12. Los 5 empleados con menor sueldo.

```
hive> select * from empleados order by salario asc limit 5;
Query ID = cloudera_20240711142020_780d6062-d650-4ce2-abd7-fad6a29539c6
Total jobs = 1
Launching Job 1 out of 1
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_1720685678318_0005, Tracking URL = http://quickstart.cloudera:8088/p
roxy/application_1720685678318_0005/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1720685678318_0005
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2024-07-11 14:20:59,541 Stage-1 map = 0%,  reduce = 0%
2024-07-11 14:21:06,962 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.85 sec
2024-07-11 14:21:14,418 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 1.99 sec
MapReduce Total cumulative CPU time: 1 seconds 990 msec
Ended Job = job_1720685678318_0005
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 1.99 sec   HDFS Read: 8539 HDFS Writ
e: 108 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 990 msec
OK
empleados.nombre        empleados.apellidos     empleados.edad  empleados.genero      e
mpleados.salario
Marta   Doncel  27      M       50
Pedro   Ariz    23      H       130
Sergio  Chava   20      H       200
Isaac   Garrido 19      H       1000
Pablo   Loa     21      H       10000
Time taken: 23.306 seconds, Fetched: 5 row(s)
```

13. Los 5 empleados con mayor sueldo.

```
hive> select * from empleados order by salario desc limit 5;
Query ID = cloudera_20240711142222_df95a9b9-2eaf-4fd3-8635-5d1194900755
Total jobs = 1
Launching Job 1 out of 1
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_1720685678318_0006, Tracking URL = http://quickstart.cloudera:8088/p
roxy/application_1720685678318_0006/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1720685678318_0006
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2024-07-11 14:22:12,977 Stage-1 map = 0%,  reduce = 0%
2024-07-11 14:22:19,432 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.8 sec
2024-07-11 14:22:26,797 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 1.94 sec
MapReduce Total cumulative CPU time: 1 seconds 940 msec
Ended Job = job_1720685678318_0006
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 1.94 sec   HDFS Read: 8539 HDFS Writ
e: 117 SUCCESS
Total MapReduce CPU Time Spent: 1 seconds 940 msec
OK
empleados.nombre        empleados.apellidos     empleados.edad  empleados.genero      e
mpleados.salario
Santiago        Gil     21      H       50000
Miguel  Angel   40      H       50000
Santi   Gil     21      H       50000
Santiago        Gil     22      H       50000
Miguel  Angel   40      H       50000
Time taken: 23.291 seconds, Fetched: 5 row(s)
```

14. Muestra la media por géneros.

```
hive> select genero, avg(salario) from empleados group by genero;
Query ID = cloudera_20240711142424_0390e4da-89c9-4da9-8175-982b7aad260c
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_1720685678318_0007, Tracking URL = http://quickstart.cloudera:8088/p
roxy/application_1720685678318_0007/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1720685678318_0007
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2024-07-11 14:24:39,669 Stage-1 map = 0%,  reduce = 0%
2024-07-11 14:24:46,145 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 0.98 sec
2024-07-11 14:24:53,621 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 2.21 sec
MapReduce Total cumulative CPU time: 2 seconds 210 msec
Ended Job = job_1720685678318_0007
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 2.21 sec   HDFS Read: 9046 HDFS Writ
e: 38 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 210 msec
OK
genero  _c1
G       21000.0
H       27486.923076923078
M       50.0
```

EJERCICIO DATASET POKEMON

1. Crear una tabla llamada pokemonStats y carga los datos del csv pokemon.csv.

a. Ten en cuenta el formato de las columnas que tiene el CSV para establecer los delimitadores de campo.

DROP TABLE IF EXISTS pokemonStats;

```
hive> CREATE TABLE IF NOT EXISTS pokemonStats (
    >   id STRING,
    >   nombre STRING,
    >   tipoA STRING,
    >   tipoB STRING,
    >   total STRING,
    >   hp STRING,
    >   ataque STRING,
    >   defensa STRING,
    >   ataque_v STRING,
    >   defensa_v STRING,
    >   velocidad STRING,
    >   generacion STRING,
    >   legendario STRING
    > )
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > LINES TERMINATED BY '\n';
OK
Time taken: 0.517 seconds
```

Cargo los datos del csv dentro de la tabla…

```
hive> LOAD DATA LOCAL INPATH '/home/cloudera/Desktop/Pokemon.csv' INTO TABLE pok
emonStats;
Loading data to table default.pokemonstats
Table default.pokemonstats stats: [numFiles=1, totalSize=44414]
OK
Time taken: 0.477 seconds
```

Vemos como queda al final:

```
720     HoopaHoopa Unbound      Psychic Dark    680     80      160     60      1
70      130     80      6       True
721     Volcanion       Fire    Water   600     80      110     120     130     9
0       70      6       True
Time taken: 0.304 seconds, Fetched: 801 row(s)
```

b. Borra la tabla y créala nuevamente para que la carga de los datos use una expresión regular.

Eliminamos la tabla y comprobamos que se ha eliminado:

```
hive> DROP TABLE pokemonStats;
OK
Time taken: 0.198 seconds
hive> select * from pokemonStats;
FAILED: SemanticException [Error 10001]: Line 1:14 Table not found 'pokemonStats'
```

Creamos la tabla de nuevo con la expresión regular:

```
hive> CREATE TABLE IF NOT EXISTS pokemonStats (
    >   id STRING,
    >   nombre STRING,
    >   tipoA STRING,
    >   tipoB STRING,
    >   total STRING,
    >   hp STRING,
    >   ataque STRING,
    >   defensa STRING,
    >   ataque_v STRING,
    >   defensa_v STRING,
    >   velocidad STRING,
    >   generacion STRING,
    >   legendario STRING
    > )
    > ROW FORMAT SERDE 'org.apache.hadoop.hive.contrib.serde2.RegexSerDe'
    > WITH SERDEPROPERTIES (
    >   "input.regex"="(\\d*)\\,(\\w*\\s*\\w*\\s*\\w*)\\,(\\w*\\-*)\\,(\\w*\\-*)
\\,(\\d*)\\,(\\d*)\\,(\\d*)\\,(\\d*)\\,(\\d*)\\,(\\d*)\\,(\\d*)\\,(\\d*)\\,(\\w*
)",
    >   "output.format.string"="%1$s %2$s %3$s %4$s %5$s %6$s %7$s %8$s %9$s %10
$s %11$s %12$s %13$s"
    > )
    > STORED AS TEXTFILE
    > tblproperties("skip.header.line.count"="1");
OK
Time taken: 0.061 seconds
```

ROW FORMAT SERDE es para especifica que se utilizará una SerDe (Serializer/Deserializer) personalizada basada en expresiones regulares para definir cómo se deben interpretar y formatear las filas.
`"input.regex"` define la expresión regular que se usará para analizar las filas de entrada. Cada grupo entre paréntesis captura una parte de la fila correspondiente a una columna de la tabla:

- `(\\d*)` :indican números (* = 0 a infinito, + = 1 a infinito),
- `\\,`:Delimita entre campos,
- `(\\w*\\s*\\w*\\s*\\w*)`:w = cualquier digito, s = espacio en blanco,
- `\\-*`: captura 0 o mas guiones

`"output.format.string"` especifica cómo se deben formatear las filas al escribir los datos.

El output, los numeros especifican a que expresion regular del input se refieren, y luego si es $d espera un numero, y $s una cadena

`STORED AS TEXTFILE` indica que los datos se almacenarán como archivos de texto.
`tblproperties("skip.header.line.count"="1")` especifica que se debe omitir la primera línea del archivo de texto (generalmente, esta línea contiene los encabezados de las columnas).

Realizamos la inserción de datos de nuevo:

```
hive> LOAD DATA LOCAL INPATH '/home/cloudera/Desktop/Pokemon.csv' INTO TABLE pok
emonStats;
Loading data to table default.pokemonstats
Table default.pokemonstats stats: [numFiles=1, totalSize=44414]
OK
Time taken: 0.233 seconds
```

Y comprobamos que se han cargado bien los datos:

```
720     HoopaHoopa Unbound      Psychic Dark    680     80      160     60      1
70      130     80      6       True
721     Volcanion       Fire    Water   600     80      110     120     130     9
0       70      6       True
Time taken: 0.048 seconds, Fetched: 800 row(s)
```

Se puede ver que la inserción ha sido correcta.

2. Crea una nueva tabla a partir de pokemonstats teniendo en cuenta que la mayor parte de las consultas se realizan filtrando por generación de Pokémon.
Usaré “Partitioned by”, que divide los datos en subdirectorios basados en los valores de una o más columnas de partición. Esto mejora el rendimiento de las consultas que filtran por esas columnas, ya que solo se escanean las particiones relevantes en lugar de toda la tabla.
Por lo tanto, el primer paso es crear la tabla particionada “pokemonGeneration”.

```
hive> CREATE TABLE IF NOT EXISTS pokemonGeneration (
    >     id INT,
    >     nombre STRING,
    >     tipoA STRING,
    >     tipoB STRING,
    >     total INT
    > )
    > PARTITIONED BY (generacion INT)
    > STORED AS TEXTFILE;
OK
Time taken: 0.054 seconds
```

El “STORED AS TEXTFILE”, se pone porque define que los datos se almacenarán como archivos de texto.
El siguiente paso es cargar los datos en la nueva tabla desde la tabla pokemonStats:
Me daba fallo de primeras, pq tenia que cambiar a mone=nonstrict para que no hubiese problemas en la particion de columna:

```
FAILED: SemanticException [Error 10096]: Dynamic partition strict mode requires
at least one static partition column. To turn this off set hive.exec.dynamic.par
tition.mode=nonstrict
hive> -- Desactivar el modo estricto de partición dinámica
    > SET hive.exec.dynamic.partition = true;
hive> SET hive.exec.dynamic.partition.mode = nonstrict;
```

Ahora si, la inserción:

```
hive> INSERT OVERWRITE TABLE pokemonGeneration
    > PARTITION (generacion)
    > SELECT CAST(id AS INT), nombre, tipoA, tipoB, CAST(total AS INT), CAST(gen
eracion AS INT)
    > FROM pokemonStats;
Query ID = cloudera_20240712090404_42d9be3f-5869-4a8e-acca-17f75a0ae8ae
Total jobs = 3
```

`INSERT OVERWRITE TABLE pokemonGeneration PARTITION (generacion)` sobreescribe cualquier dato existente de la tabla que se especifica y los datos los organiza en particiones basadas en la columna `generacion`.

Comprobamos que ha ido bien:

```
717     Yveltal Dark    Flying  680     6
719     Diancie Rock    Fairy   600     6
719     DiancieMega Diancie     Rock    Fairy   700     6
720     HoopaHoopa Confined     Psychic Ghost   600     6
720     HoopaHoopa Unbound      Psychic Dark    680     6
721     Volcanion       Fire    Water   600     6
NULL    NULL    NULL    NULL    NULL    NULL
NULL    NULL    NULL    NULL    NULL    NULL
NULL    NULL    NULL    NULL    NULL    NULL
NULL    NULL    NULL    NULL    NULL    NULL
NULL    NULL    NULL    NULL    NULL    NULL
NULL    NULL    NULL    NULL    NULL    NULL
NULL    NULL    NULL    NULL    NULL    NULL
NULL    NULL    NULL    NULL    NULL    NULL
NULL    NULL    NULL    NULL    NULL    NULL
Time taken: 0.083 seconds, Fetched: 800 row(s)
```

3. Crea una nueva tabla a partir de pokemonstats teniendo en cuenta que la mayor parte de las consultas se realizan filtrando por generación de Pokémon, tipoA y tipoB.

Usaré una tabla particionada y además usare bucketing para las columnas tipoA y tipoB.

La tabla se particiona por generacion. (PARTITIONED BY …)
`CLUSTERED BY (tipoA, tipoB) INTO 30 BUCKETS`: Clustered by se utiliza para organizar físicamente los datos de una tabla en función de una o más columnas específicas.

Los datos se agruparán (bucket) por las columnas `tipoA` y `tipoB` en 30 segmentos o cubos. Bucketing mejora el rendimiento de las consultas cuando se filtra por las columnas `tipoA` y `tipoB`.

```
hive> CREATE TABLE IF NOT EXISTS pokemonGenerationType (
    >     id INT,
    >     nombre STRING,
    >     total INT,
    >     hp INT,
    >     attack INT,
    >     defense INT,
    >     sp_atk INT,
    >     sp_def INT,
    >     speed INT,
    >     tipoA STRING,
    >     tipoB STRING
    > )
    > PARTITIONED BY (generacion STRING)
    > CLUSTERED BY (tipoA, tipoB) INTO 30 BUCKETS
    > STORED AS TEXTFILE;
OK
Time taken: 0.058 seconds
```

Para la inserción de datos he tenido que hacer de nuevo esto:

```
FAILED: SemanticException [Error 10096]: Dynamic partition strict mode requires
at least one static partition column. To turn this off set hive.exec.dynamic.par
tition.mode=nonstrict
hive> SET hive.exec.dynamic.partition = true;
hive> SET hive.exec.dynamic.partition.mode = nonstrict;
```

y una vez hecho, pasar a la inserción:

```
hive> INSERT OVERWRITE TABLE pokemonGenerationType
    >     PARTITION (generacion)
    >     SELECT
    >         CAST(id AS INT),
    >         nombre,
    >         CAST(total AS INT),
    >         CAST(hp AS INT),
    >         CAST(ataque AS INT),
    >         CAST(defensa_v AS INT),
    >         CAST(ataque_v AS INT),
    >         CAST(defensa_v AS INT),
    >         CAST(velocidad AS INT),
    >         tipoA,
    >         tipoB,
    >         CAST(generacion AS STRING)
    >     FROM pokemonStats;
Query ID = cloudera_20240712094949_f4f42d1b-836f-4ef3-a885-c547af2ac9de
Total jobs = 3
Launching Job 1 out of 3
```

Comprobamos que ha ido bien:

```
720     HoopaHoopa Unbound      680     80      160     130     170     130     8
0       Psychic Dark    6
721     Volcanion       600     80      110     90      130     90      70      F
ire     Water   6
NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    N
ULL     __HIVE_DEFAULT_PARTITION__
NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    N
ULL     __HIVE_DEFAULT_PARTITION__
NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    N
ULL     __HIVE_DEFAULT_PARTITION__
NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    N
ULL     __HIVE_DEFAULT_PARTITION__
NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    N
ULL     __HIVE_DEFAULT_PARTITION__
NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    N
ULL     __HIVE_DEFAULT_PARTITION__
NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    N
ULL     __HIVE_DEFAULT_PARTITION__
NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    NULL    N
ULL     __HIVE_DEFAULT_PARTITION__
```

4. ¿Qué generación es más poderosa?

```
hive> SELECT generacion, SUM(total) AS suma_total
    > FROM pokemonGeneration
    > GROUP BY generacion
    >
    > ORDER BY suma_total DESC
    > LIMIT 1;
```

Se puede ver que la mas poderosa es la 5:

```
generacion      suma_total
5       71773
```

5. ¿Qué generación tiene el mayor número de Pokémon de tipo fuego? ¿Cuántos tiene?

Realizamos la consulta:

```
hive> select generacion, count(tipoa) as num_pokemon from pokemongenerationtype
where tipoa = 'Fire' group by generacion order by num_pokemon desc limit 1;
Query ID = cloudera_20240712104040_9022a88c-093b-487f-bd5a-9c19b98cb37f
Total jobs = 2
Launching Job 1 out of 2
```

Y vemos lo que sale:

```
generacion      num_pokemon
1       14
Time taken: 48.79 seconds, Fetched: 1 row(s)
```

Por lo tanto, la generación 1, con 14 pokemon fuego.