UNIVERSIDAD MAYOR DE SAN SIMÓN
FACULTAD DE CIENCIAS Y
TECNOLOGÍA
DIRECCIÓN DE POSGRADO

DIPLOMADO ESTADÍSTICA APLICADA A LA TOMA DE DECISIONES

SEGUNDA VERSIÓN

LABORATORIO DATABRICKS

NOMBRE : HEBERT JUAN DE DIOS DELGADILLO FERNANDEZ
DOCENTE : DANNY LUIS HUANCA SEVILLA

Cochabamba – Bolivia
2023

Información a entregar

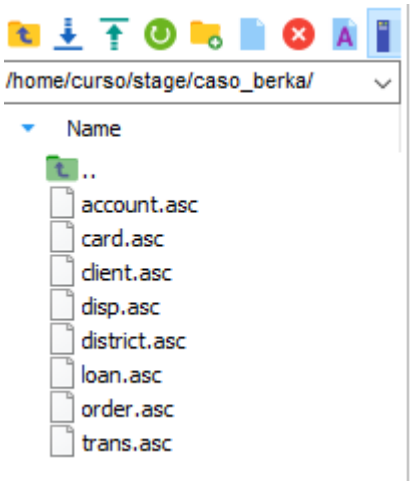
1. ¿Es un problema de aprendizaje supervisado o no supervisado?

Si, se trata de un modelo de Machine Learning supervisado donde la variable Y es el campo STATUS de la tabla de préstamos debido a que esta contiene la información de nuestra variable objetivo asociada a la necesidad del negocio, que en este caso es para clasificar tipos de clientes en base a los datos históricos de otros clientes.

El banco BERKA necesita una forma de poder identificar al tipo de cliente que es un buen candidato para una tarjeta de crédito o un préstamo y que tipo cliente representa un alto riesgo para el banco.

La necesidad del negocio si puede ser resuelta con un algoritmo de Machine Learning, ya que se trata de clasificar a un tipo de cliente en base a sus características y datos propios. El banco cuenta con datos de los préstamos realizados y las transacciones hechas por el cliente, en donde se proporciona un campo en la tabla de préstamos (LOANS) qué es STATUS (estado del pago del préstamo), en el cual se sabe si un cliente pagó o no su deuda o tuvo algún problema, este campo representa una etiqueta para cada registro de la tabla. Por lo cual se puede utilizar un algoritmo de Machine Learning para clasificación y de tipo supervisado.

2. Crear un ETL que permita importar los archivos desde su máquina local al servidor Hive. Cada archivo debe ser una tabla. Puede ingresarlos manualmente o puede usar alguna herramienta ETL.



Lo subimos mediante MobaXterm manualmente.

Creamos una carpeta para cada archivo:

```
(base) curso@cursobigdata:~/stage$ hdfs dfs -mkdir -p /TABLA_EXTERNA/tablaBerka/card
(base) curso@cursobigdata:~/stage$ hdfs dfs -mkdir -p /TABLA_EXTERNA/tablaBerka/client
(base) curso@cursobigdata:~/stage$ hdfs dfs -mkdir -p /TABLA_EXTERNA/tablaBerka/disp
(base) curso@cursobigdata:~/stage$ hdfs dfs -mkdir -p /TABLA_EXTERNA/tablaBerka/district
(base) curso@cursobigdata:~/stage$ hdfs dfs -mkdir -p /TABLA_EXTERNA/tablaBerka/loan
(base) curso@cursobigdata:~/stage$ hdfs dfs -mkdir -p /TABLA_EXTERNA/tablaBerka/order
(base) curso@cursobigdata:~/stage$ hdfs dfs -mkdir -p /TABLA_EXTERNA/tablaBerka/trans
(base) curso@cursobigdata:~/stage$ hdfs dfs -ls /TABLA_EXTERNA/tablaBerka
Found 8 items
drwxr-xr-x   - curso supergroup          0 2023-09-02 02:49 /TABLA_EXTERNA/tablaBerka/account
drwxr-xr-x   - curso supergroup          0 2023-09-02 02:50 /TABLA_EXTERNA/tablaBerka/card
drwxr-xr-x   - curso supergroup          0 2023-09-02 02:50 /TABLA_EXTERNA/tablaBerka/client
drwxr-xr-x   - curso supergroup          0 2023-09-02 02:50 /TABLA_EXTERNA/tablaBerka/disp
drwxr-xr-x   - curso supergroup          0 2023-09-02 02:50 /TABLA_EXTERNA/tablaBerka/district
drwxr-xr-x   - curso supergroup          0 2023-09-02 02:50 /TABLA_EXTERNA/tablaBerka/loan
drwxr-xr-x   - curso supergroup          0 2023-09-02 02:50 /TABLA_EXTERNA/tablaBerka/order
drwxr-xr-x   - curso supergroup          0 2023-09-02 02:50 /TABLA_EXTERNA/tablaBerka/trans
(base) curso@cursobigdata:~/stage$
```

Subimos cada archivo a las respectivas carpetas:

```
(base) curso@cursobigdata:~/stage$ hdfs dfs -put caso_berka/account.asc /TABLA_EXTERNA/tablaBerka/account
(base) curso@cursobigdata:~/stage$ hdfs dfs -put caso_berka/card.asc /TABLA_EXTERNA/tablaBerka/card
(base) curso@cursobigdata:~/stage$ hdfs dfs -put caso_berka/client.asc /TABLA_EXTERNA/tablaBerka/client
(base) curso@cursobigdata:~/stage$ hdfs dfs -put caso_berka/disp.asc /TABLA_EXTERNA/tablaBerka/disp
(base) curso@cursobigdata:~/stage$ hdfs dfs -put caso_berka/district.asc /TABLA_EXTERNA/tablaBerka/district
(base) curso@cursobigdata:~/stage$ hdfs dfs -put caso_berka/loan.asc /TABLA_EXTERNA/tablaBerka/loan
(base) curso@cursobigdata:~/stage$ hdfs dfs -put caso_berka/order.asc /TABLA_EXTERNA/tablaBerka/orden
(base) curso@cursobigdata:~/stage$ hdfs dfs -put caso_berka/order.asc /TABLA_EXTERNA/tablaBerka/trans
(base) curso@cursobigdata:~/stage$ hdfs dfs -put caso_berka/trans.asc /TABLA_EXTERNA/tablaBerka/trans
```

Tabla card

```
CREATE TABLE CARD_EXT
(card_id int,
disp_id int,
tipo String,
issued date
)
comment 'datos tabla berka'
row format delimited
fields terminated by ';'
stored as textfile
location '/TABLA_EXTERNA/tablaBerka/card'
;

create table card
as
select *
from card_ext

select * from card
```

esultados 1

elect * from card | Enter a SQL expression to filter results (use Ctrl+Space)

	123 card_id	123 disp_id	ABC tipo	issued
1	[NULL]	[NULL]	"type"	[NULL]
2	1.005	9.285	"classic"	[NULL]
3	104	588	"classic"	[NULL]
4	747	4.915	"classic"	[NULL]
5	70	439	"classic"	[NULL]
6	577	3.687	"classic"	[NULL]
7	377	2.429	"classic"	[NULL]
8	721	4.680	"junior"	[NULL]
9	437	2.762	"classic"	[NULL]
10	188	1.146	"classic"	[NULL]
11	13	87	"classic"	[NULL]
12	732	4.763	"classic"	[NULL]

Y su archivo card en el warehouse de Hive:

```
(base) curso@cursobigdata:~/stage$ hdfs dfs -ls /user/hive/warehouse/card
Found 1 items
-rw-r--r--   1 curso supergroup      19079 2023-09-02 03:03 /user/hive/warehouse/card/000000_0
```

Así para cada tabla con el siguiente código SQL:

```
CREATE TABLE CARD_EXT
(card_id int,
disp_id int,
tipo String,
issued date
)
comment 'datos tabla berka'
row format delimited
fields terminated by ';'
stored as textfile
location '/TABLA_EXTERNA/tablaBerka/card'
;


create table card
as
select *
from card_ext

select * from card

CREATE TABLE trans (
trans_id int,
account_id int,
fecha date,
tipo String,
operaction String,
amount int,
balance int,
k_symbol String,
bank String,
account int
)
comment 'datos tabla trans'
row format delimited fields terminated by ';' stored as textfile
location '/TABLA_EXTERNA/tablaBerka/trans'
;

create table transhive
as
select *
from trans

select * from transhive


CREATE TABLE disp (
disp_id int,
client_id int,
account_id int,
tipo String
)
comment 'datos tabla disp'
row format delimited fields terminated by ';' stored as textfile
location '/TABLA_EXTERNA/tablaBerka/disp'
;

create table disphive
as
select *
```

```
from disp

select * from disphive

CREATE TABLE orden (
order_id int,
account_id int,
bank_to String,
amount float,
k_symbol String
)
comment 'datos tabla order'
row format delimited fields terminated by ';' stored as textfile
location '/TABLA_EXTERNA/tablaBerka/orden'
;

create table orderhive
as
select *
from orden

select * from orderhive

CREATE TABLE loan (
loan_id int,
account_id int,
fecha date,
amount int,
paymentes float,
status String
)
comment 'datos tabla loan'
row format delimited fields terminated by ';' stored as textfile
location '/TABLA_EXTERNA/tablaBerka/loan'
;

create table loanhive
as
select *
from loan

select * from loanhive



CREATE TABLE account (
account_id int,
district_id int,
frequency String,
fecha date
)
comment 'datos tabla account'
row format delimited fields terminated by ';' stored as textfile
location '/TABLA_EXTERNA/tablaBerka/account'
;

create table accounthive
as
select *
from account
```

```
select * from accounthive
```

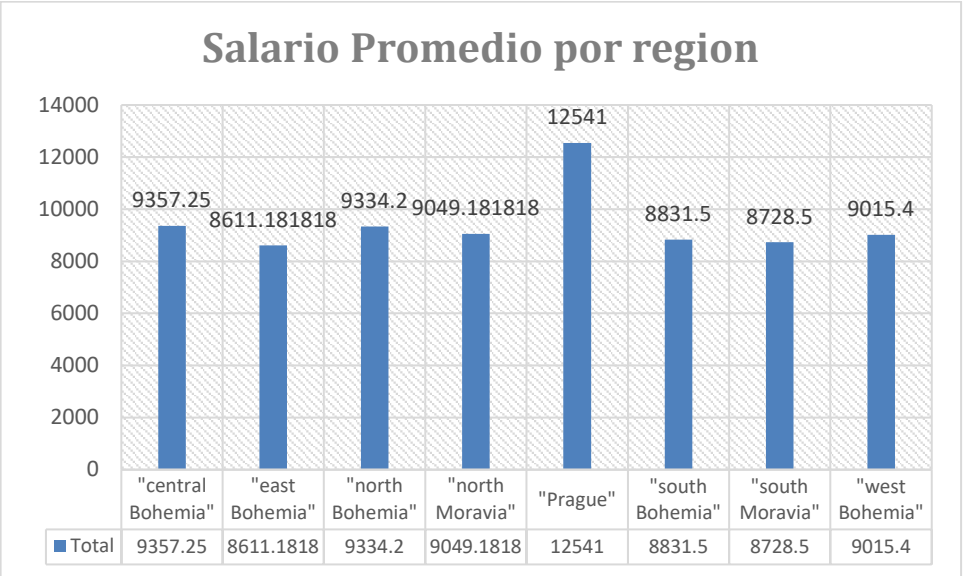
```
CREATE TABLE client (
client_id int,
gender String,
birth_date date,
district_id int
)
comment 'datos tabla client'
row format delimited fields terminated by ';' stored as textfile
location '/TABLA_EXTERNA/tablaBerka/client'
;

create table clienthive
as
select *
from client

select * from clienthive
```

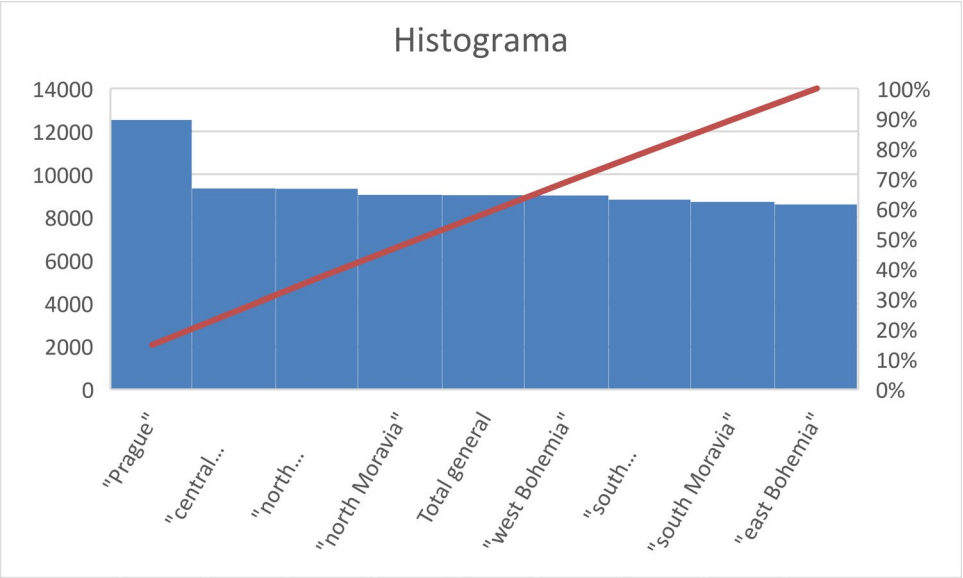
```
CREATE TABLE district (
distrit_id int,
A2 String,
A3 String,
A4 int,
A5 int,
A6 int,
A7 int,
A8 int,
A9 int,
A10 float,
A11 int,
A12 float,
A13 float,
A14 int,
A15 int,
A16 int
)
comment 'datos tabla district'
row format delimited fields terminated by ';' stored as textfile
location '/TABLA_EXTERNA/tablaBerka/district'
;

create table districthive
as
select *
from district

select * from districthive
```

Teniendo también cada tabla almacenada en Hive:

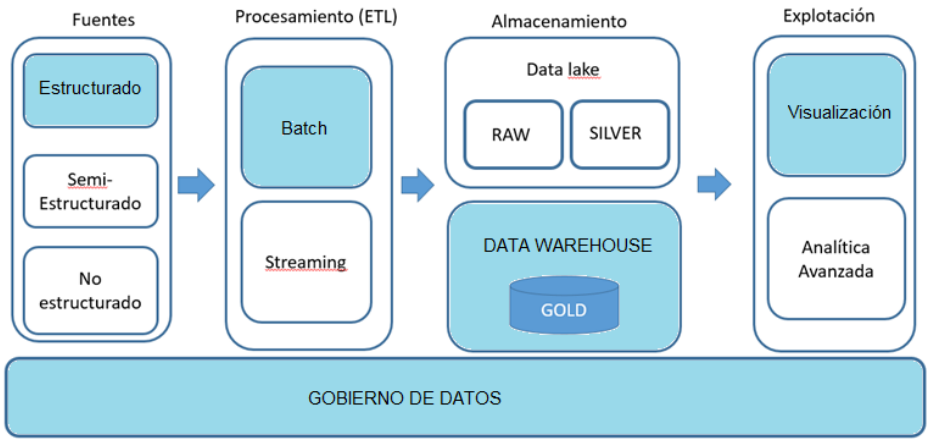
```
(base) curso@cursobigdata:~/stage$ hdfs dfs -ls /user/hive/warehouse/
Found 12 items
drwxr-xr-x   - curso supergroup          0 2023-09-02 03:21 /user/hive/warehouse/accounthive
drwxr-xr-x   - curso supergroup          0 2023-09-02 03:03 /user/hive/warehouse/card
drwxr-xr-x   - curso supergroup          0 2023-09-02 03:22 /user/hive/warehouse/clienthive
drwxr-xr-x   - curso supergroup          0 2023-09-02 03:17 /user/hive/warehouse/disphive
drwxr-xr-x   - curso supergroup          0 2023-09-02 03:23 /user/hive/warehouse/districthive
drwxr-xr-x   - curso supergroup          0 2023-09-02 03:19 /user/hive/warehouse/loanhive
drwxr-xr-x   - curso supergroup          0 2023-09-02 03:18 /user/hive/warehouse/orderhive
drwxr-xr-x   - curso supergroup          0 2023-09-01 14:25 /user/hive/warehouse/prueba.db
drwxrwxr-x   - curso supergroup          0 2020-10-12 02:14 /user/hive/warehouse/src
drwxr-xr-x   - curso supergroup          0 2023-09-01 13:30 /user/hive/warehouse/telco1
drwxr-xr-x   - curso supergroup          0 2023-09-01 14:15 /user/hive/warehouse/telco2
drwxr-xr-x   - curso supergroup          0 2023-09-02 03:15 /user/hive/warehouse/transhive
```

3. Crear una conexión ODBC/JDBC para poder visualizar los datos desde Excel u otra herramienta de visualización. Combine 3 tablas y realice una gráfica de barras y un histograma.



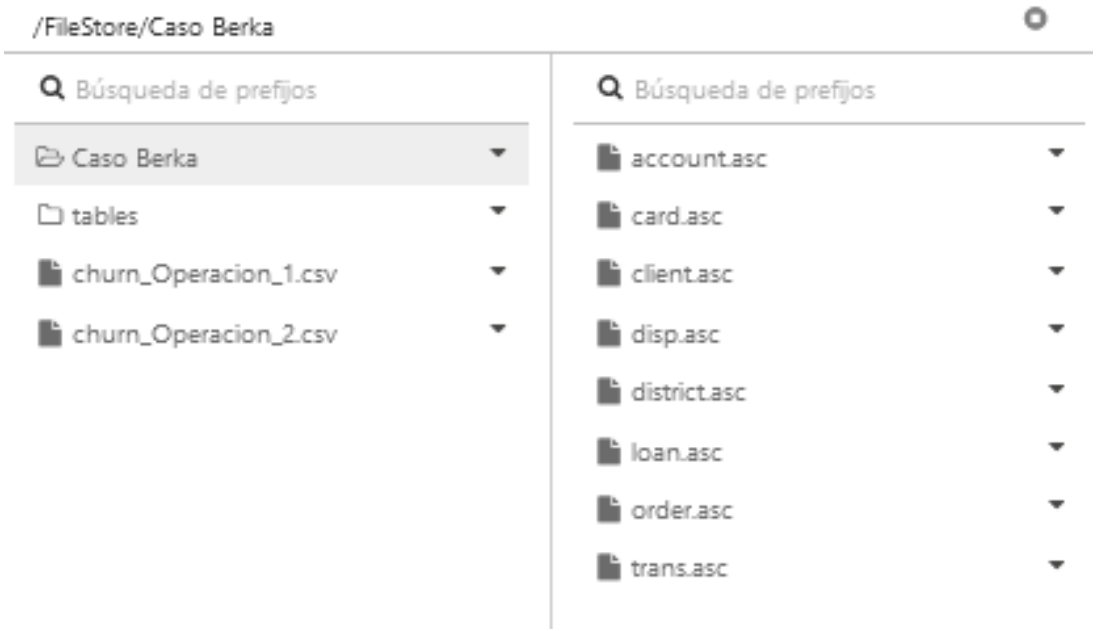


¿Qué componentes del framework de arquitectura de datos utilizó? Pinte los componentes usados.



4. Realizar la misma operación en Databricks, subir los archivos y crear tablas para cada uno.

   Subimos los archivos al DBFS:



   Creamos tablas para cada archivo:

```
DROP TABLE IF EXISTS account;
CREATE TABLE account
USING csv
OPTIONS (path "/FileStore/Caso Berka/account.asc",delimiter ";", header "true");

DROP TABLE IF EXISTS card;
CREATE TABLE card
USING csv
OPTIONS (path "/FileStore/Caso Berka/card.asc",delimiter ";", header "true");

DROP TABLE IF EXISTS client;
CREATE TABLE client
USING csv
OPTIONS (path "/FileStore/Caso Berka/client.asc",delimiter ";", header "true");

DROP TABLE IF EXISTS disp;
CREATE TABLE disp
USING csv
OPTIONS (path "/FileStore/Caso Berka/disp.asc",delimiter ";", header "true");

DROP TABLE IF EXISTS district;
CREATE TABLE district
USING csv
OPTIONS (path "/FileStore/Caso Berka/district.asc",delimiter ";", header "true");

DROP TABLE IF EXISTS loan;
CREATE TABLE loan
USING csv
OPTIONS (path "/FileStore/Caso Berka/loan.asc",delimiter ";", header "true");

DROP TABLE IF EXISTS orden;
CREATE TABLE orden
USING csv
OPTIONS (path "/FileStore/Caso Berka/order.asc",delimiter ";", header "true");

DROP TABLE IF EXISTS trans;
CREATE TABLE trans
USING csv
OPTIONS (path "/FileStore/Caso Berka/trans.asc",delimiter ";", header "true");
```

Verificamos que todas las tablas fueron creadas:

```
show tables;
```

Tabla +

	database	tableName	isTemporary
3	default	card	false
4	default	client	false
5	default	disp	false
6	default	district	false
7	default	loan	false
8	default	orden	false
9	default	telco_operacion1	false

Verificamos si los datos fueron cargados correctamente en cada tabla:

```
select * from district
```

(1) trabajos de Spark

Tabla +

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16
1	1	Hl.m. Praha	Prague	1204953	0	0	0	1	1	100.0	12541	0.29	0.43	167	85677	9910
2	2	Benesov	central Bohemia	88884	80	26	6	2	5	46.7	8507	1.67	1.85	132	2159	2674
3	3	Beroun	central Bohemia	75232	55	26	4	1	5	41.7	8980	1.95	2.21	111	2824	2813
4	4	Kladno	central Bohemia	149893	63	29	6	2	6	67.4	9753	4.64	5.05	109	5244	5892
5	5	Kolin	central Bohemia	95616	65	30	4	1	6	51.4	9307	3.85	4.43	118	2616	3040
6	6	Kutna Hora	central Bohemia	77963	60	23	4	2	4	51.5	8546	2.95	4.02	126	2640	3120
7	7	Melnik	central Bohemia	94725	38	28	1	3	6	63.4	9920	2.26	2.87	130	4288	4941

77 filas | 0,25 segundos tiempo de ejecución — Actualizado hace hace 3 horas

5. Utilizando SQL cree una tabla minable que permita resolver un problema de ciencia de datos en el entorno de Databricks, proponga uno en base al caso de estudio (esto solo en Databricks).

Creamos la tabla minable mediante comando SQL, donde la variable confiabilidad es la variable Y para un posible modelo de Machine Learning supervisado.

```
alter view transcompleta as select t.account_id as Cuenta, count(*) as Nro_movimientos,sum(t.amount) as Total_Dinero_movido,
CASE
    WHEN l.status  = "A" THEN "Excelente candidato"
    WHEN l.status  = "B" THEN "Inconfiable"
    WHEN l.status  = "C" THEN "Confiable"
    WHEN l.status  = "D" THEN "Dudoso"
    else "No se presto"
end Confiabilidad,
CASE
    WHEN a.frequency  = "POPLATEK MESICNE" THEN "Uso mensual"
    WHEN a.frequency  = "POPLATEK TYDNE" THEN "Uso semanal"
    when a.frequency  = "POPLATEK PO OBRATU" THEN "Frecuente"
end Frecuencia, d.a3 as Region,d.A11 as Salario_Promedio,d.a14 as Empresarios_en_miles
from trans t
inner join account a
on t.account_id =a.account_id
left join loan l
on l.account_id = a.account_id
left join district d
on d.a1 = a.district_id
group by t.account_id, Confiabilidad, Frecuencia, d.A3,d.a11,d.a14;

select * from transcompleta
```

(5) trabajos de Spark

Tabla +

	Cuenta	Nro_movimientos	Total_Dinero_movido	Confiabilidad	Frecuencia	Region	Salario_Promedio	Empresarios_en_miles
1	477	543	2506673.1999999997	No se presto	Uso mensual	Prague	12541	167
2	1786	389	2628673.6999999993	No se presto	Uso mensual	Prague	12541	167
3	6273	398	3593880.599999999	Excelente candidato	Uso mensual	north Bohemia	8965	104
4	601	326	427848.7000000002	No se presto	Uso mensual	west Bohemia	8843	113
5	1197	380	588929.1000000002	No se presto	Uso mensual	north Moravia	9893	96
6	2987	359	3176167.000000001	No se presto	Uso mensual	south Bohemia	8968	131
7	1811	470	3371901.1000000006	Confiable	Uso mensual	north Bohemia	9317	97

4500 filas | 2.90 segundos tiempo de ejecución

Conclusiones y recomendaciones

Pudimos comparar al servidor Have con Databricks, siendo este ultimo muy superior a Hive, por la versatilidad, por la interfaz, por el potencial de cómputo, por estar en la nube además que Hive tiene sus errores que nos hicieron la vida difícil al intentar crear las tablas en Hive, ya que si había algún error al momento de crear las tablas, lo cual era muy fácil de cometer, los archivos asc del caso Berka se iban borrando, desde solo desaparecer sin explicación alguna, hasta quedarse sin formato o borrándose la carpeta entera, sin mencionar que necesitamos crear una carpeta para cada tabla que vaya a ser leída por Hive, dicho problema nunca sucedió en Databricks, su entorno es mas amigable, los archivos son más fáciles de tratar y también puede conectarse directamente a GitHub para tenerlo en el repositorio, sin dudar Databricks es una mejor opción que Hive.