Delta Architecture - Session 1

==========

we mentioned that your lakehouse architecture can handle different usecases

Delta Architecture

Bronze -> Silver -> Gold

Delta achitecture helps us to improve the quality of data

serve different personas

Medallian architecture

Bronze -

raw data

single source of truth

data can be stored in any format

Silver -

cleaned & filtered data

in delta format

it can serve machine learning & datascience use cases

Gold -

aggregated, business specific data

optimized query layer

BI/reporting

we can even have a platinum layer if business demands that

file1

order_id,order_date,customer_id,order_status

1,2013-07-25 00:00:00.0,11599,CLOSED

2,2013-07-25 00:00:00.0,256,PENDING_PAYMENT

3,2013-07-25 00:00:00.0,12111,COMPLETE

4,2013-07-25 00:00:00.0,8827,CLOSED

5,2013-07-25 00:00:00.0,11318,COMPLETE

6,2013-07-25 00:00:00.0,7130,COMPLETE

7,2013-07-25 00:00:00.0,4530,COMPLETE

8,2013-07-25 00:00:00.0,2911,PROCESSING

9,2013-07-25 00:00:00.0,5657,PENDING_PAYMENT

10,2013-07-25 00:00:00.0,5648,PENDING_PAYMENT

add file1

update orders set order_status = 'CLOSED' where order_id = 3

file2

3,2013-07-25 00:00:00.0,12111,CLOSED

1,2013-07-25 00:00:00.0,11599,CLOSED

2,2013-07-25 00:00:00.0,256,PENDING_PAYMENT

4,2013-07-25 00:00:00.0,8827,CLOSED

5,2013-07-25 00:00:00.0,11318,COMPLETE

6,2013-07-25 00:00:00.0,7130,COMPLETE

7,2013-07-25 00:00:00.0,4530,COMPLETE

8,2013-07-25 00:00:00.0,2911,PROCESSING

9,2013-07-25 00:00:00.0,5657,PENDING_PAYMENT

10,2013-07-25 00:00:00.0,5648,PENDING_PAYMENT

remove file1

add file2

delete from orders where order_id = 4

file3

3,2013-07-25 00:00:00.0,12111,CLOSED

1,2013-07-25 00:00:00.0,11599,CLOSED

2,2013-07-25 00:00:00.0,256,PENDING_PAYMENT

5,2013-07-25 00:00:00.0,11318,COMPLETE

6,2013-07-25 00:00:00.0,7130,COMPLETE

7,2013-07-25 00:00:00.0,4530,COMPLETE

8,2013-07-25 00:00:00.0,2911,PROCESSING

9,2013-07-25 00:00:00.0,5657,PENDING_PAYMENT

10,2013-07-25 00:00:00.0,5648,PENDING_PAYMENT

remove file2

add file3

insert into orders values(11,'2013-07-25 00:00:00.0',918,'PAYMENT_REVIEW')

file4

insert into orders values(11,'2013-07-25 00:00:00.0',918,'PAYMENT_REVIEW')

add file4

Change Data Feed (CDC)

==========

if you want to do audit for example which rows are changed/added/delted in your

table you can get to know.

to merge the incremental changes..

Delta Architecture - Session 2

==========

Change Data Feed

%sql

create table orders(

order_id int,

order_date string,

customer_id int,

order_status string)

using delta

TBLPROPERTIES (delta.enableChangeDataFeed = true)

/user/hive/warehouse/orders

1. while creating a new table we can specify tblproperties

2. alter table tablename set TBLPROPERTIES (delta.enableChangeDataFeed = true)

3. set spark.databricks.delta.properties.defaults.enableChangeDataFeed = true

%sql

describe history orders

%sql

insert into orders values (1,'2013-07-25 00:00:00.0',11599,'CLOSED'),(2,'2013-07-25

00:00:00.0',256,'PENDING_PAYMENT'),

(3,'2013-07-25 00:00:00.0',12111,'COMPLETE'),(4,'2013-07-25

00:00:00.0',8827,'CLOSED'),(5,'2013-07-25 00:00:00.0',11318,'COMPLETE')

%sql

select * from table_changes('orders',1)

%sql

delete from orders where order_id = 3

%sql

describe history orders

%sql

select * from table_changes('orders',2)

you can see one addition folder _change_data (this wont keep the insert information)

%sql

update orders set order_status = 'COMPLETE' where order_id = 4

%sql

describe history orders

%sql

select * from table_changes('orders',3)

changes are logged in _change_data (update,delete,merge)

what is the usecase for this CDC or change data feed?

1. auditing

2. incremental merges among table

Delta Architecture - Session 3

==========

Bronze -> Silver -> Gold

it helps to improve the quality of data

serves different usecases

medallian architecture

CDC (change data feed)

%sql

create database retaildb

%sql

create table retaildb.orders_bronze

(

order_id int,

order_date string,

```
customer_id int,
order_status string,
filename string,
createdon timestamp
)
using delta
location "dbfs:/FileStore/data/orders_bronze.delta"
partitioned by (order_status)
TBLPROPERTIES (delta.enableChangeDataFeed = true)
%sql
create table retaildb.orders_silver
(
order_id int,
order_date date,
customer_id int,
order_status string,
order_year int GENERATED ALWAYS AS (YEAR(order_date)),
order_month int GENERATED ALWAYS AS (MONTH(order_date)),
createdon timestamp,
modifiedon timestamp
)
using delta
location "dbfs:/FileStore/data/orders_silver.delta"
partitioned by (order_status)
TBLPROPERTIES (delta.enableChangeDataFeed = true)
customer_id year order_status
```

100,2013,completed - 5

100,2013,closed - 3

100,2014,completed - 2

```
%sql
create table retaildb.orders_gold
(
customer_id int,
order_status string,
order_year int,
num_orders int
)
using delta
location "dbfs:/FileStore/data/orders_gold.delta"
TBLPROPERTIES (delta.enableChangeDataFeed = true)
```

Delta Architecture - Session 4

==========

dbfs:/FileStore/raw

```
%sql
create table retaildb.orders_bronze
(
order_id int,
order_date string,
customer_id int,
order_status string,
filename string,
createdon timestamp
```

)

using delta

location "dbfs:/FileStore/data/orders_bronze.delta"

partitioned by (order_status)

TBLPROPERTIES (delta.enableChangeDataFeed = true)

%sql

copy into retaildb.orders_bronze from (

select order_id::int,

order_date::string,

customer_id::int,

order_status::string,

INPUT_FILE_NAME() as filename,

CURRENT_TIMESTAMP() as createdon

FROM 'dbfs:/FileStore/raw'

)

fileformat = CSV

format_options('header' = 'true')

68883

if you try to run the above command again it wont impact..

%sql

select * from retaildb.orders_bronze

%sql

describe history retaildb.orders_bronze

%sql

select * from table_changes ('retaildb.orders_bronze',1)

Next step is to take the changes in bronze table and merge it to silver table.

```
%sql

create or replace temporary view orders_bronze_changes

as

select * from table_changes('retaildb.orders_bronze',2) where order_id > 0
and

customer_id > 0and order_status in

('PAYMENT_REVIEW','PROCESSING','CLOSED','SUSPECTED_FRAUD','C
OMPLETE','P

ENDING','CANCELLED','PENDING_PAYMENT')

%sql

SELECT COUNT(*) from orders_bronze_changes

63657

create table retaildb.orders_silver

(

order_id int,

order_date date,

customer_id int,

order_status string,

order_year int GENERATED ALWAYS AS (YEAR(order_date)),

order_month int GENERATED ALWAYS AS (MONTH(order_date)),

createdon timestamp,

modifiedon timestamp

)

%sql

merge into retaildb.orders_silver tgt

using orders_bronze_changes src on tgt.order_id = src.order_id
```

when matched

then

update set tgt.order_status = src.order_status, tgt.customer_id = src.customer_id,

tgt.modifiedon = CURRENT_TIMESTAMP()

when not matched

then

insert(order_id, order_date, customer_id, order_status, createdon, modifiedon)

values (order_id, order_date, customer_id, order_status, CURRENT_TIMESTAMP(),

CURRENT_TIMESTAMP())

%sql

select * from retaildb.orders_silver

create table retaildb.orders_gold

(

customer_id int,

order_status string,

order_year int,

num_orders int

)

%sql

insert overwrite table retaildb.orders_gold

select customer_id, order_status, order_year, count(order_id) as num_orders from

retaildb.orders_silver group by customer_id, order_status, order_year

68883 - bronze

63657 - silver

48510 - gold

%sql

select * from retaildb.orders_gold