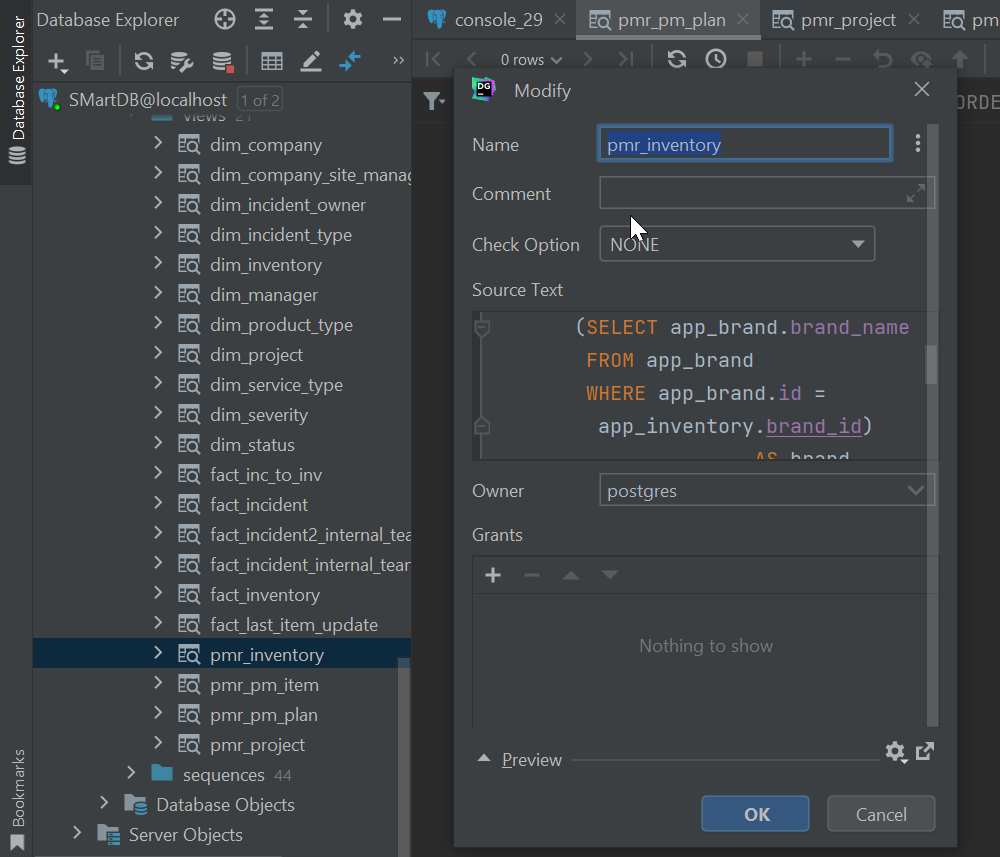
# How to create New Table Merging

## Create view to retrive data from particular table on PQ

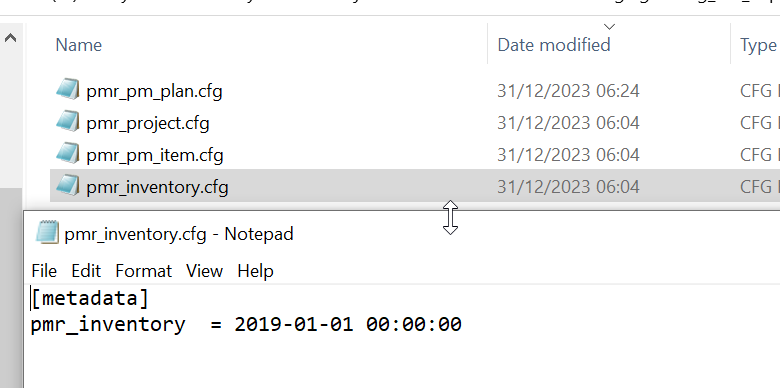
, this step it allow you to perform data transformation



create view pmr\_inventory  
 (inventory\_id, serial\_number, customer\_warranty\_start, customer\_warranty\_end, brand, model, product\_type,  
 project\_id, updated\_at)  
as  
SELECT app\_inventory.id AS inventory\_id,  
 app\_inventory.serial\_number,  
 app\_inventory.customer\_warranty\_start,  
 app\_inventory.customer\_warranty\_end,  
 (SELECT app\_brand.brand\_name  
 FROM app\_brand  
 WHERE app\_brand.id = app\_inventory.brand\_id) AS brand,  
 (SELECT app\_model.model\_name  
 FROM app\_model  
 WHERE app\_model.id = app\_inventory.model\_id) AS model,  
 (SELECT app\_product\_type.productype\_name  
 FROM app\_product\_type  
 WHERE app\_product\_type.id = app\_inventory.product\_type\_id) AS product\_type,  
 app\_inventory.project\_id,  
 app\_inventory.updated\_at  
FROM app\_inventory  
WHERE app\_inventory.is\_dummy = false;  
  
alter table pmr\_inventory  
 owner to postgres;

## CFG file Create last\_import {view name}.cgf

this is in cfg\_last\_import folder , it is time stamp to update recent import (Time is UTC)



## Edit constantval in code file LoadPGToTempBQ.p

there are 4 parts

|  |  |  |
| --- | --- | --- |
|  |  | |
| tableContentID  Find in django\_content\_type table | |
|  | key\_name  look for in view | |
|  | Sp\_merge  Store procedure created on BigQuery to run Merge transaction  Name it like merge\_{table name in main table onBQ}  Such as merge\_inventory | |
|  | changed\_field\_mapping  changed\_field\_mapping=['planned\_date','ended\_pm\_date',  'project\_id','remark','team\_lead\_id']   * All column in view : 'planned\_date' , 'ended\_pm\_date' , ,'remark' * including foreinkey ID : 'project\_id' , ,'team\_lead\_id' * excluding custom column which is column that you create it in view on your own for data transformation : pm\_period (it represent remark) ,team\_lead(it represent team\_lead\_id) * exclude: key id and time stamp column : pm\_id, update\_at       It should be aligned with colums in view  {"remark": {"new": "3/3(Mar-LastPM)", "old": "3/3(LastPM)"}, "updated\_at": {"new": "2023-12-30T14:06:51.615Z", "old": "2023-12-30T14:06:18.159Z"}} | |

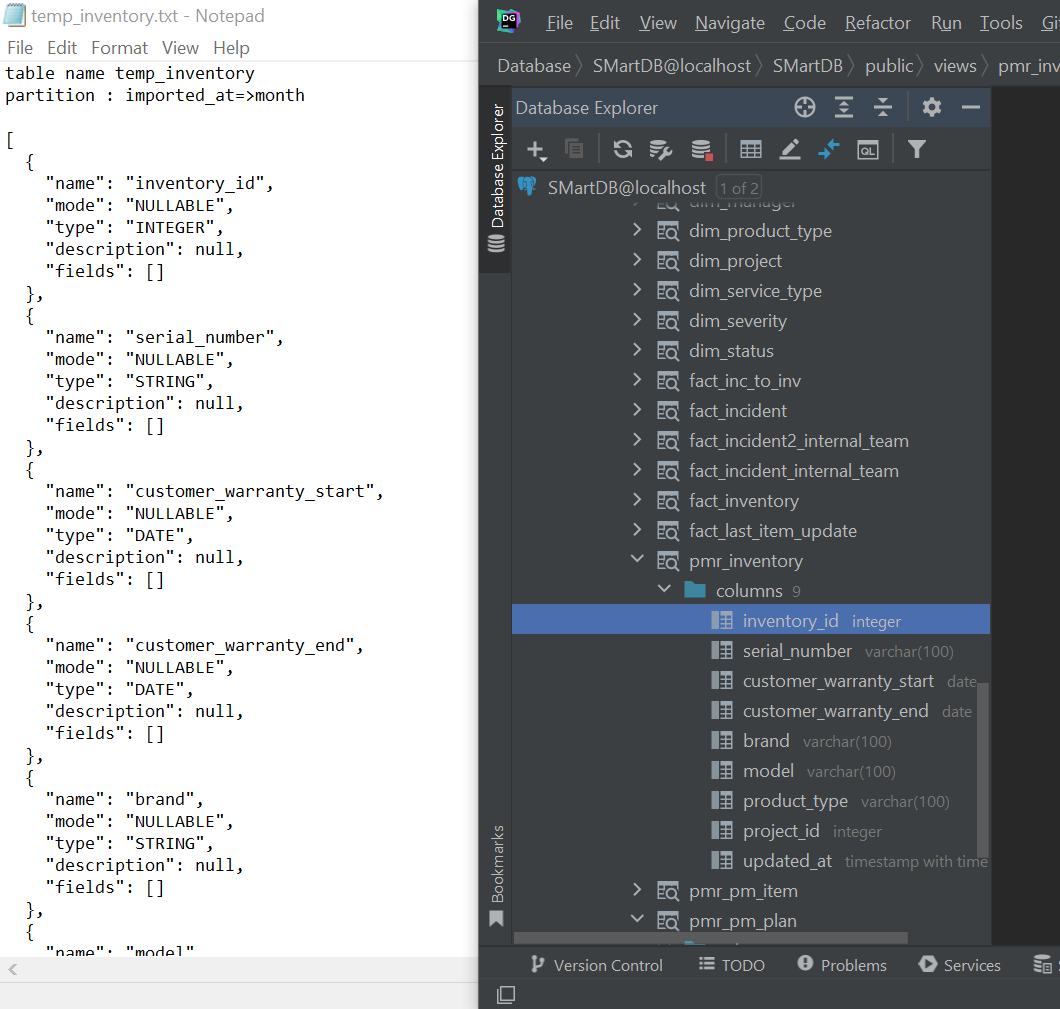
## Create BQ table

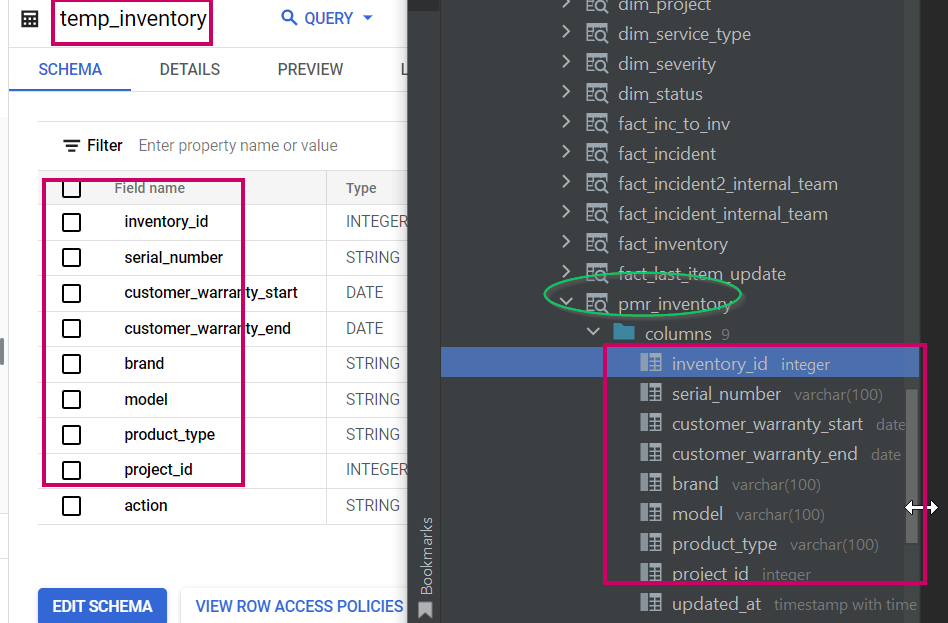
### Temp\_table

All column must be the same view in database

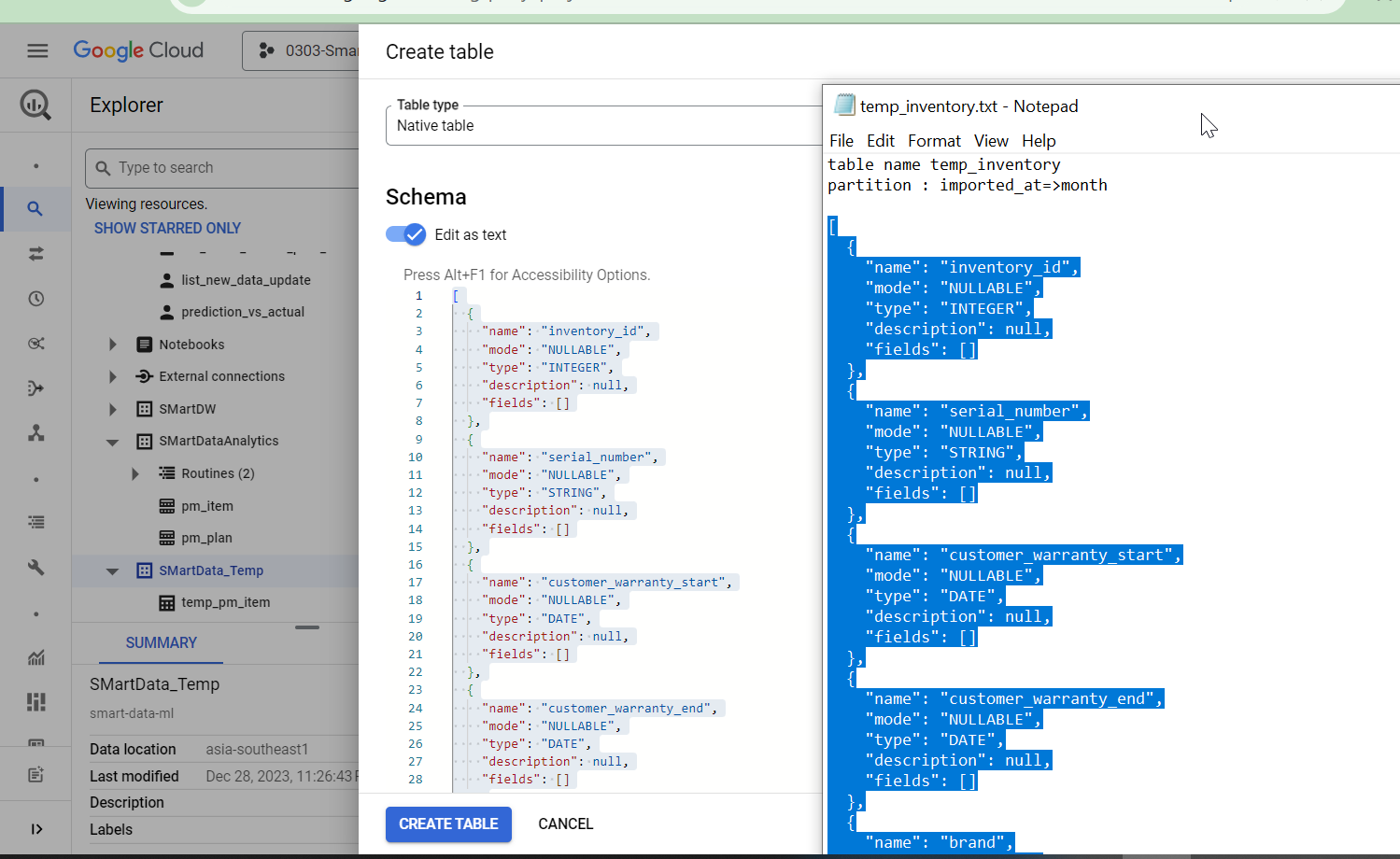
* include action filed
* exclude updated\_At field
* Create table on smart-data-ml.SMartData\_Temp

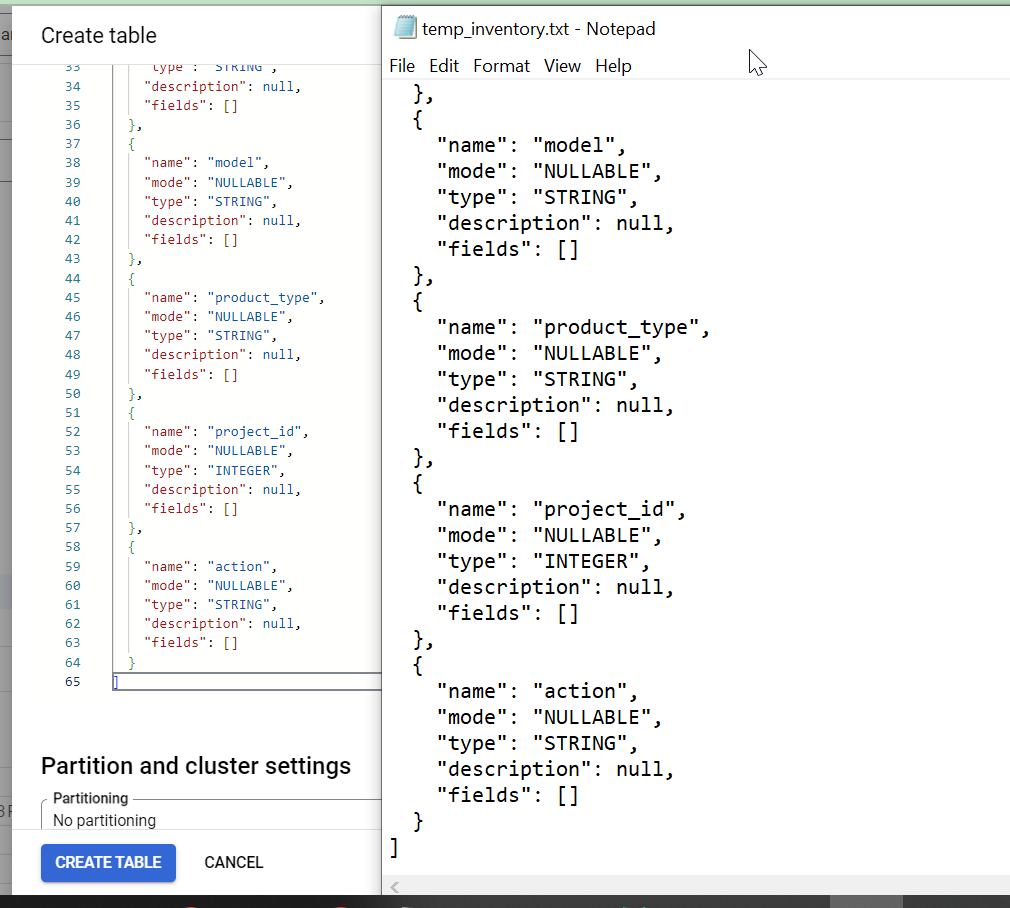
Use temp\_inventory.txt in table\_schema\_script\inventory to create as below figure

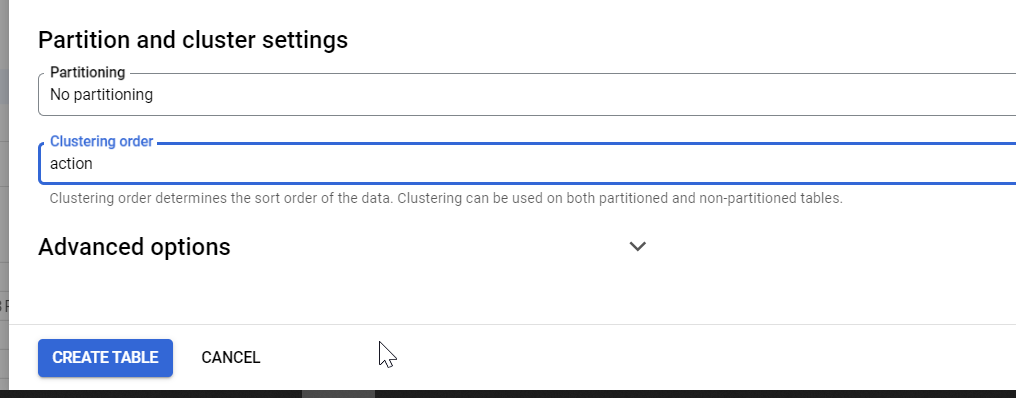


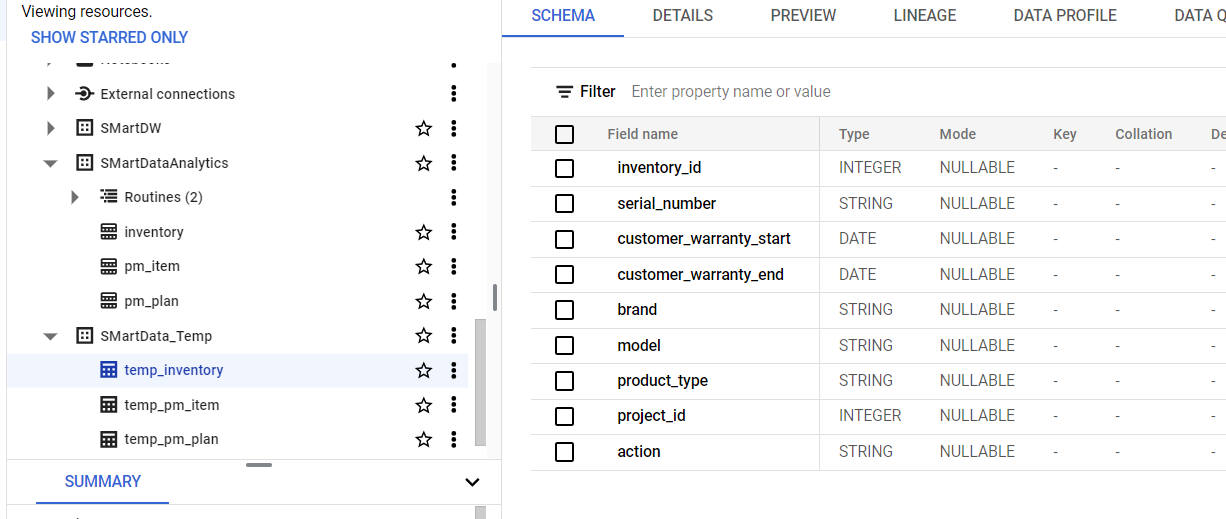










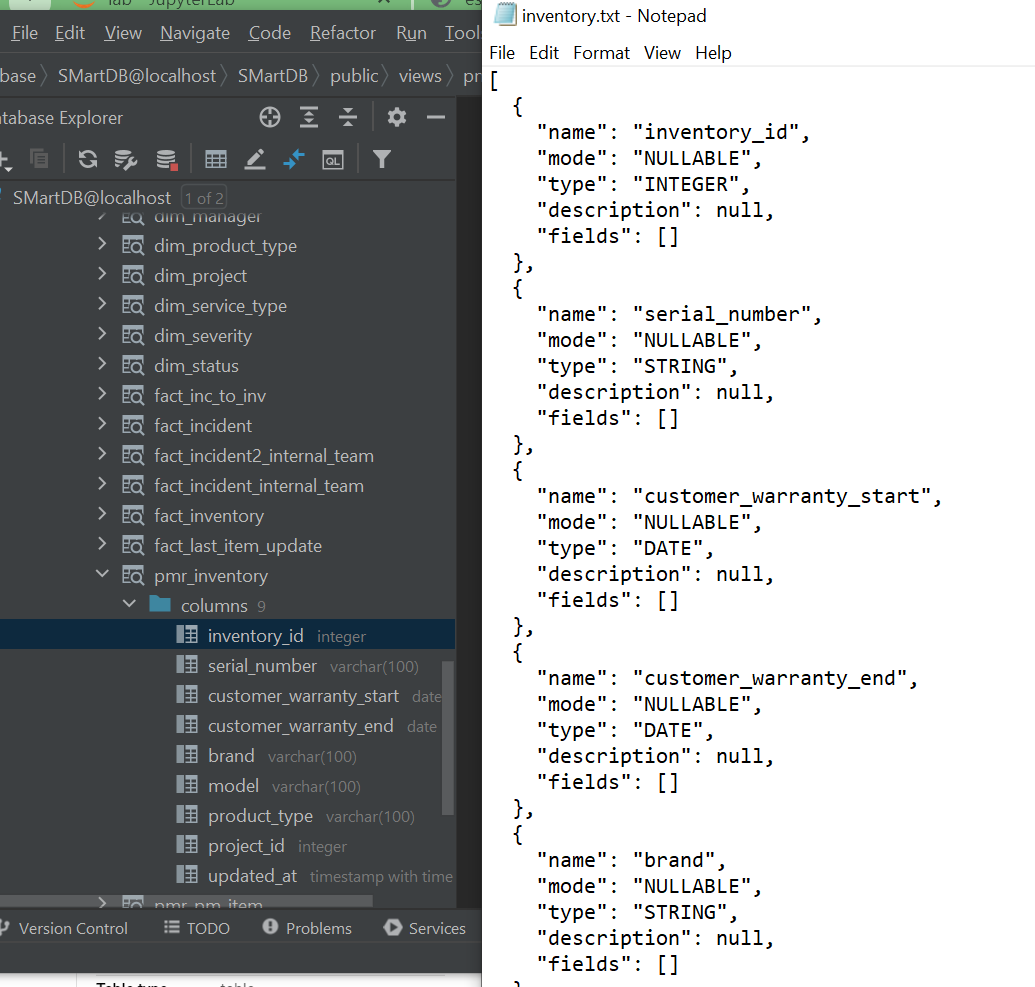


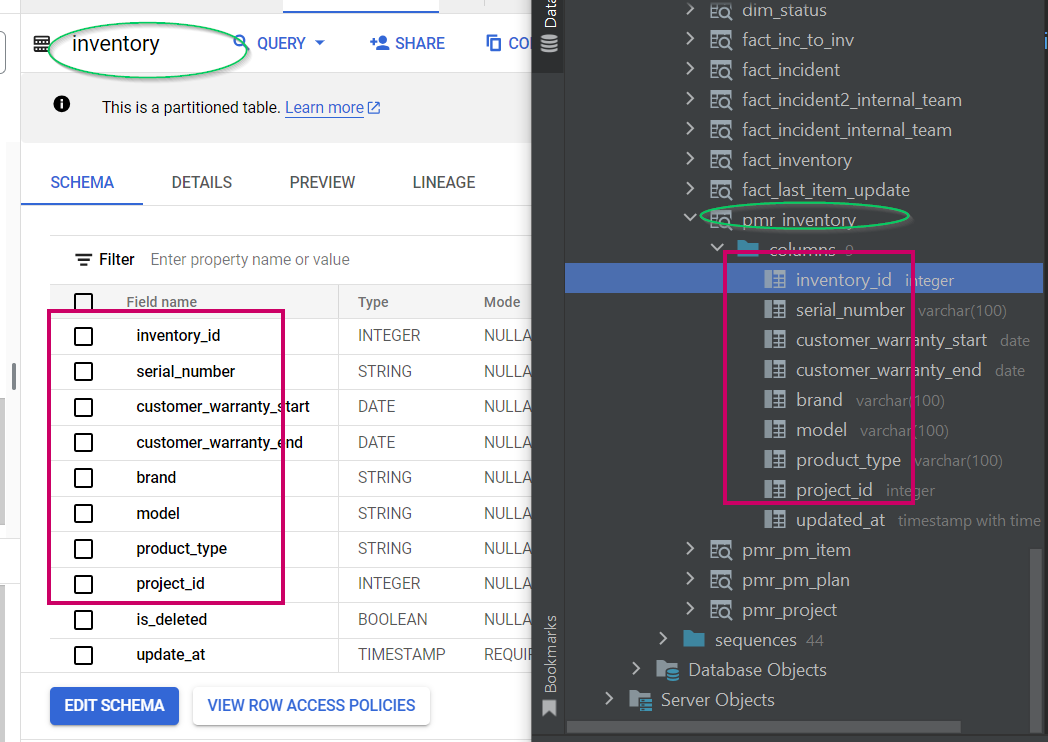
### Main Table

All column must be the same view in database

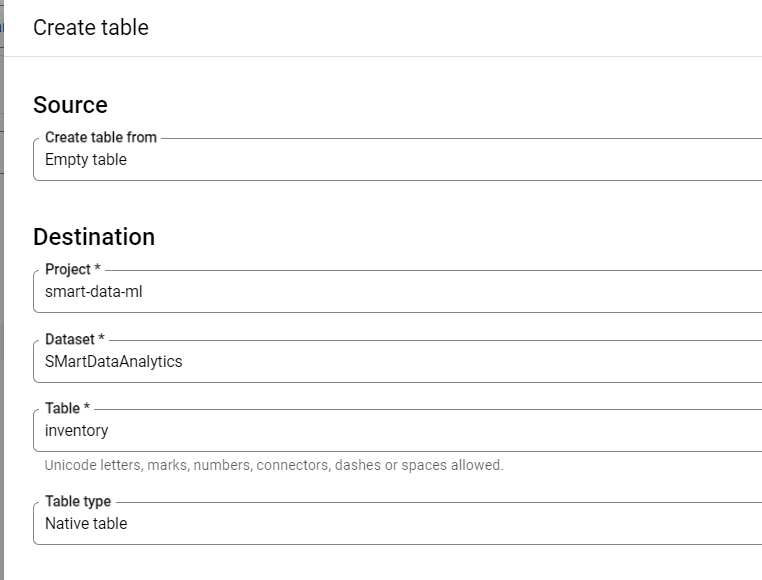
* include is\_deleted and update\_at filed
* exclude updated\_At field

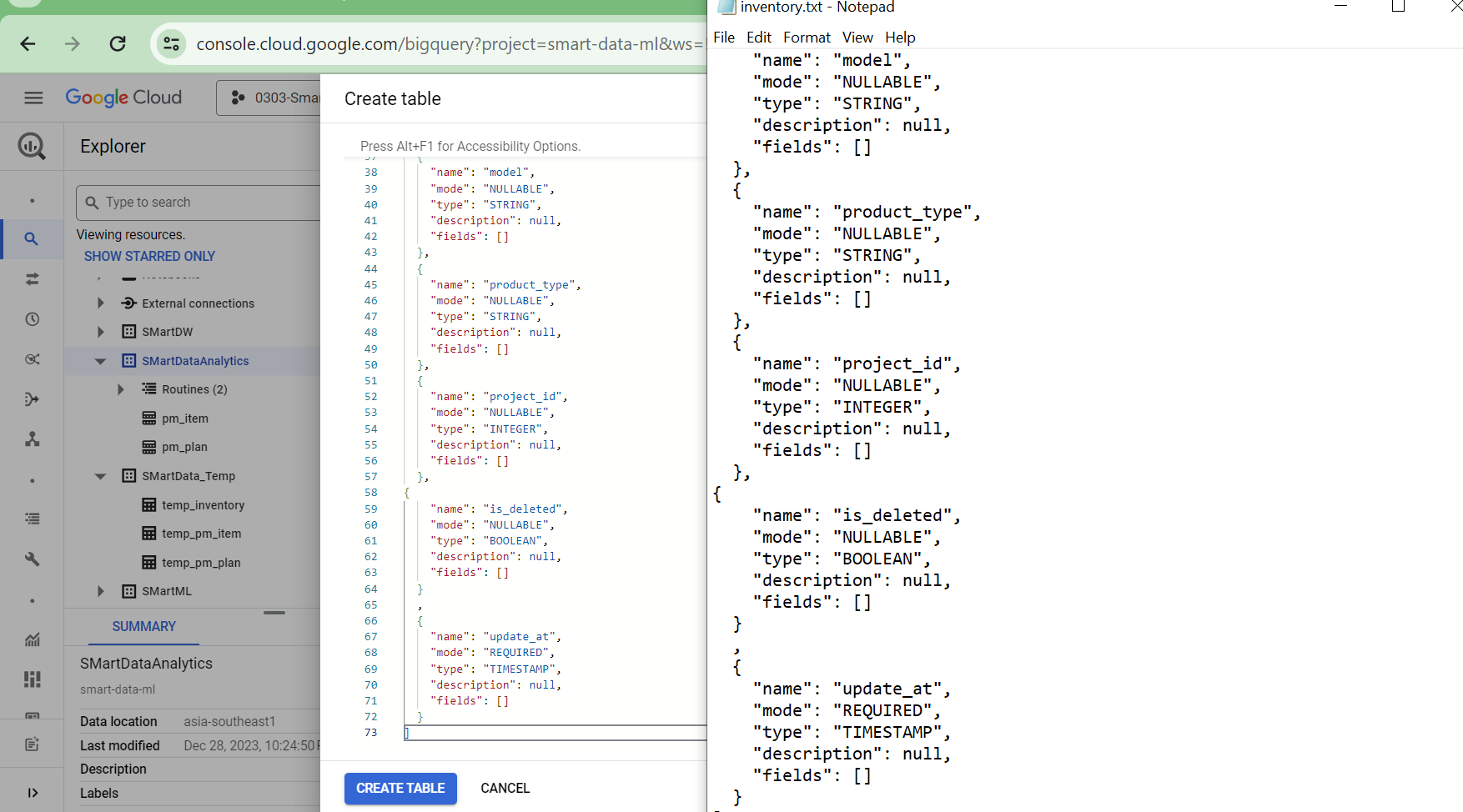
Create table on smart-data-ml.SMartDataAnalytics

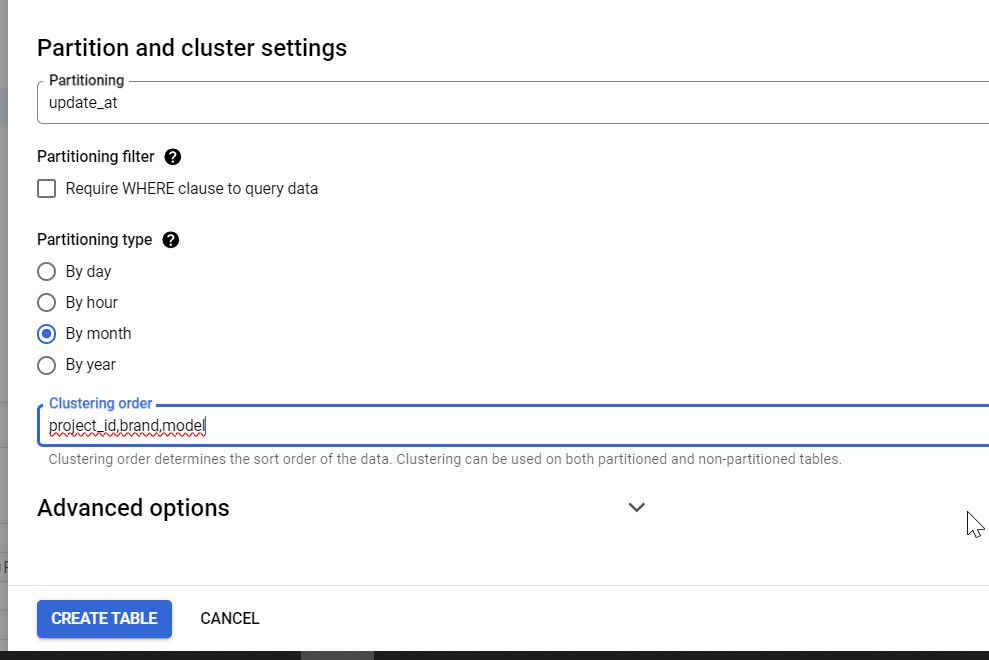


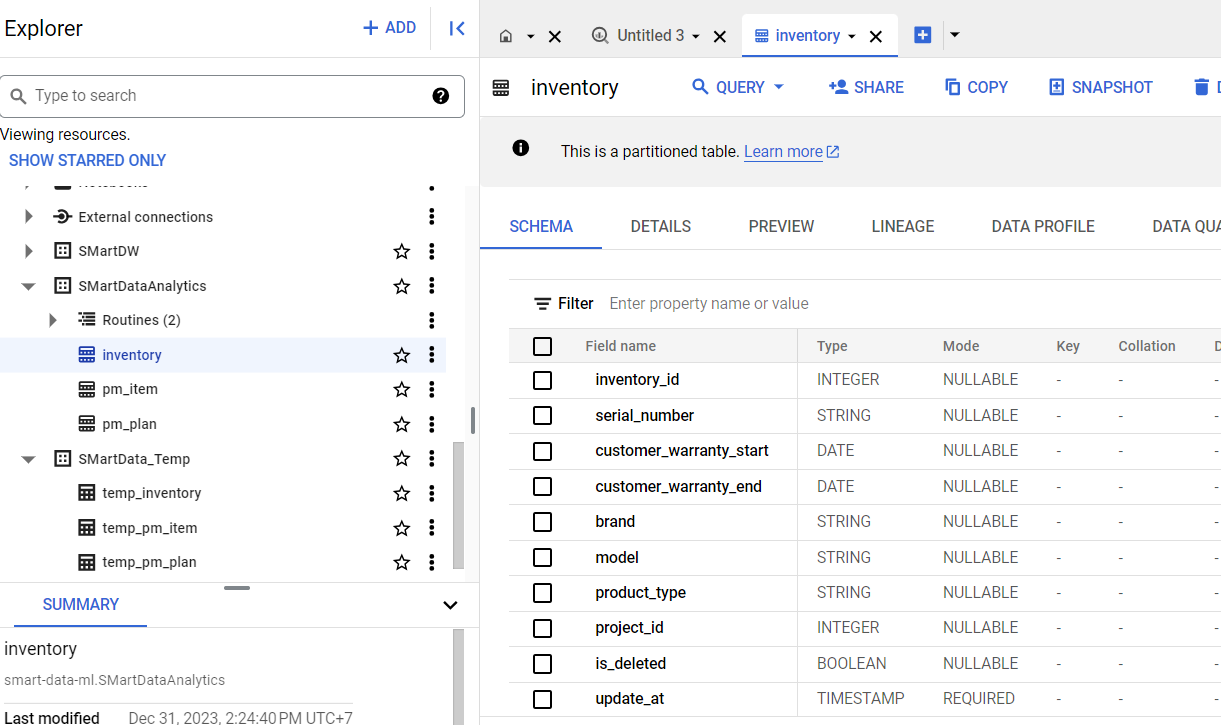


Use inventory.txt in table\_schema\_script\inventory to copy schemat to create as below figure



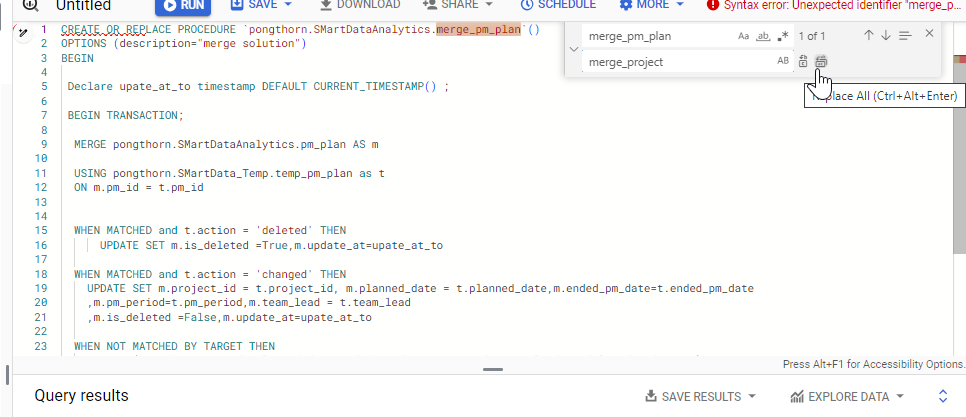


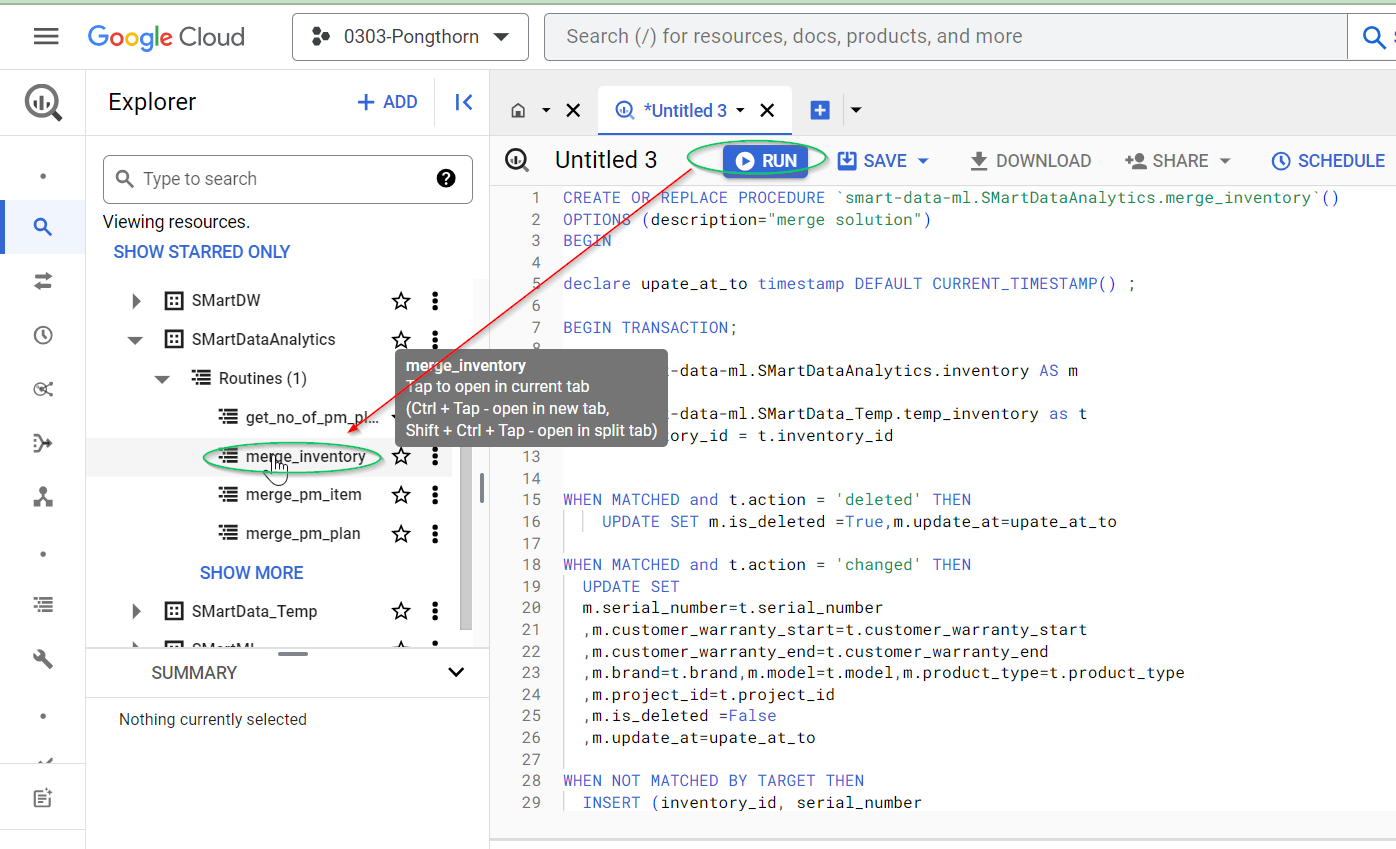




## Create Store Procedure ( Do it carefully and intentionally)

You can copy sp teamp and change field according to columns in table both main and temp table





CREATE OR REPLACE PROCEDURE `smart-data-ml.SMartDataAnalytics.merge\_inventory`()

OPTIONS (description="merge solution")

BEGIN

declare upate\_at\_to timestamp DEFAULT CURRENT\_TIMESTAMP() ;

BEGIN TRANSACTION;

MERGE smart-data-ml.SMartDataAnalytics.inventory AS m

USING smart-data-ml.SMartData\_Temp.temp\_inventory as t

ON m.inventory\_id = t.inventory\_id

WHEN MATCHED and t.action = 'deleted' THEN

UPDATE SET m.is\_deleted =True,m.update\_at=upate\_at\_to

WHEN MATCHED and t.action = 'changed' THEN

UPDATE SET

m.serial\_number=t.serial\_number

,m.customer\_warranty\_start=t.customer\_warranty\_start

,m.customer\_warranty\_end=t.customer\_warranty\_end

,m.brand=t.brand,m.model=t.model,m.product\_type=t.product\_type

,m.project\_id=t.project\_id

,m.is\_deleted =False

,m.update\_at=upate\_at\_to

WHEN NOT MATCHED BY TARGET THEN

INSERT (inventory\_id, serial\_number

,customer\_warranty\_start, customer\_warranty\_end

, brand, model,product\_type

,project\_id

,is\_deleted,update\_at)

VALUES(t.inventory\_id, t.serial\_number

,t.customer\_warranty\_start, t.customer\_warranty\_end

, t.brand, t.model,t.product\_type,

t.project\_id

,False, upate\_at\_to)

;

truncate table smart-data-ml.SMartData\_Temp.temp\_inventory;

COMMIT TRANSACTION;

EXCEPTION WHEN ERROR THEN

select @@error.message,@@error.statement\_text;

ROLLBACK TRANSACTION;

END;

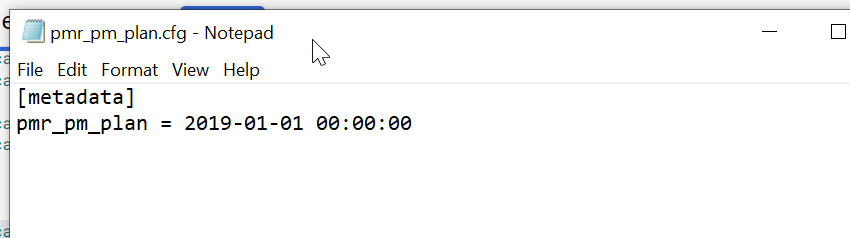
# Frist Load OR ReLoad-All

## Truncate table

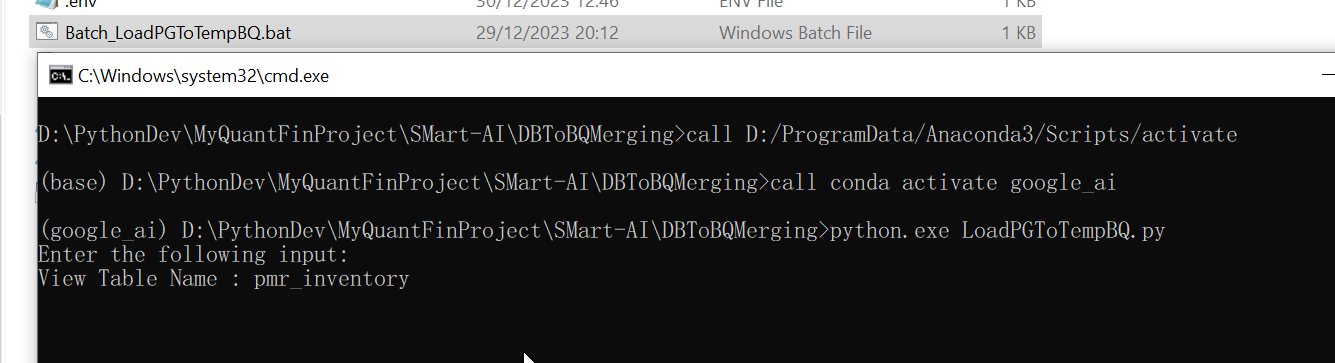
truncate table `pongthorn.SMartData\_Temp.temp\_inventory` ;

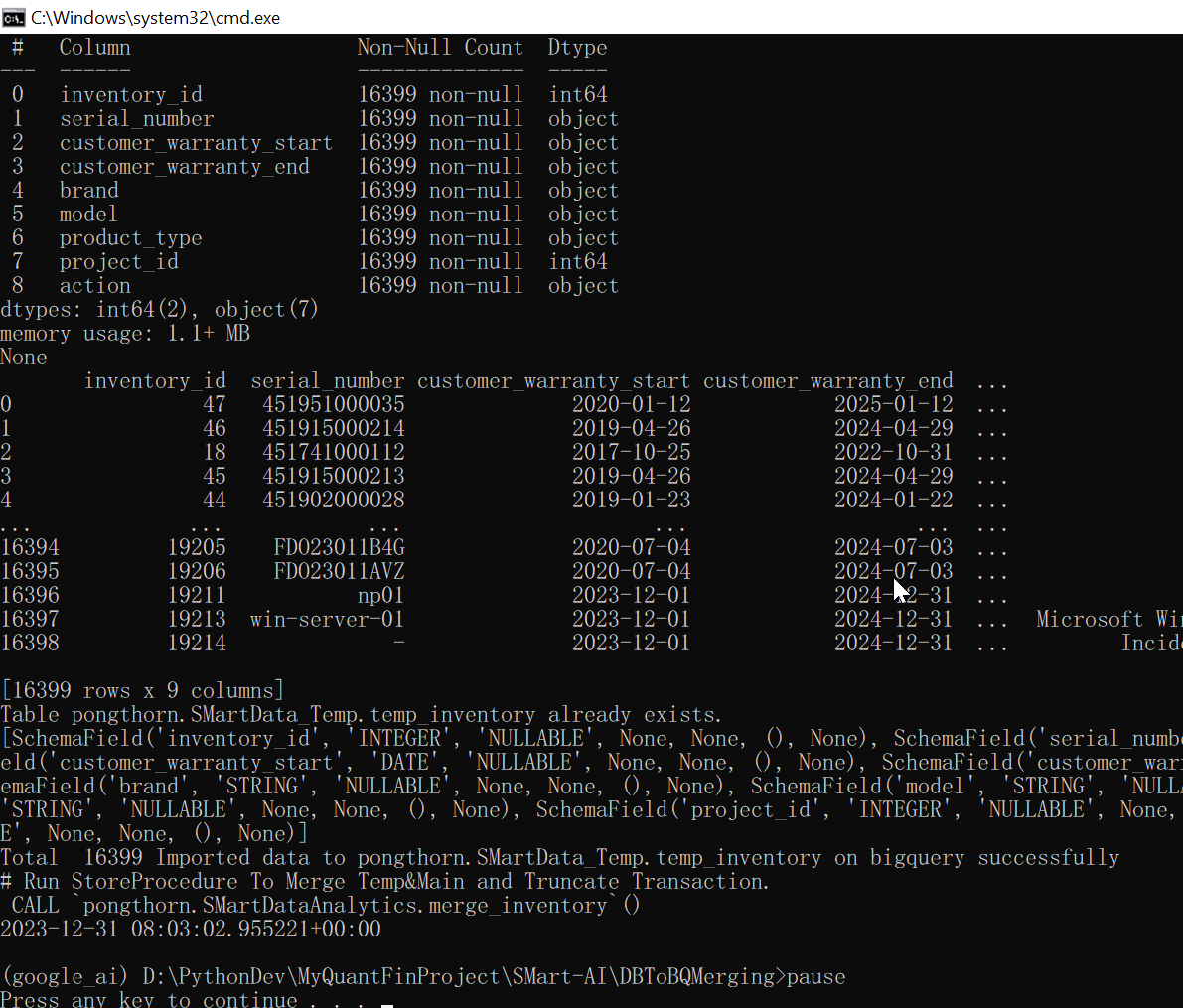
truncate table `pongthorn.SMartDataAnalytics.inventory`;

## Reset last uplate in cfg file



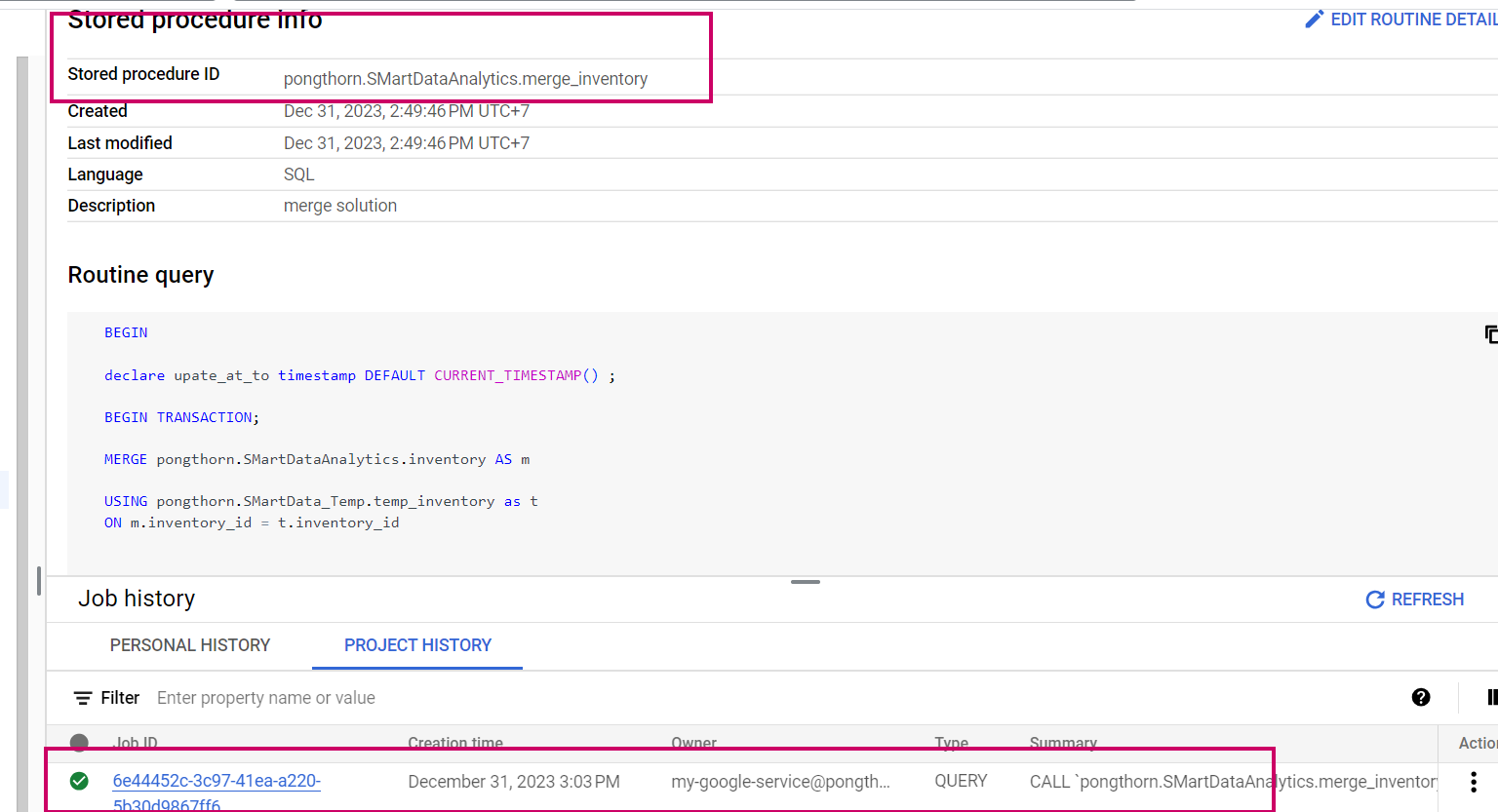
## Run script by passing view\_name as argument



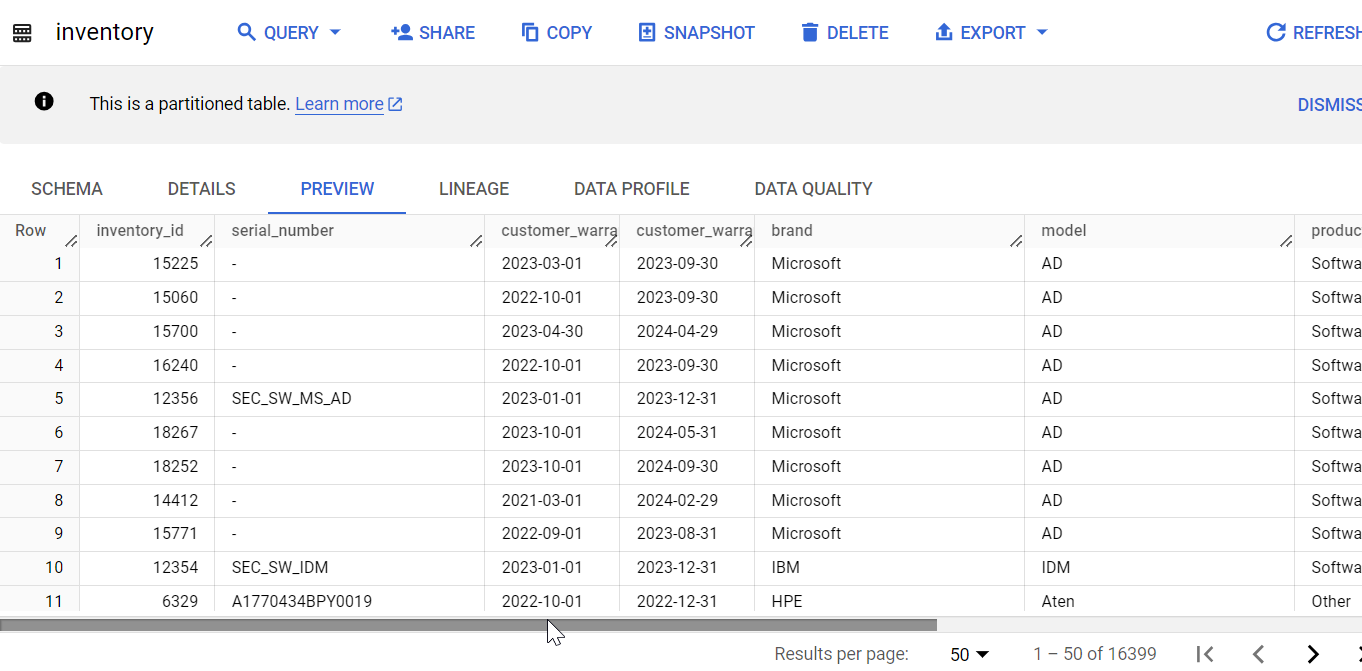


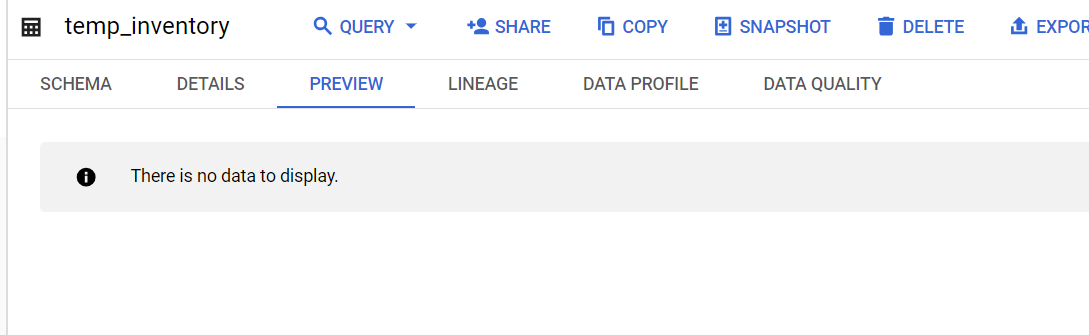
## Check result as the following items

### SP

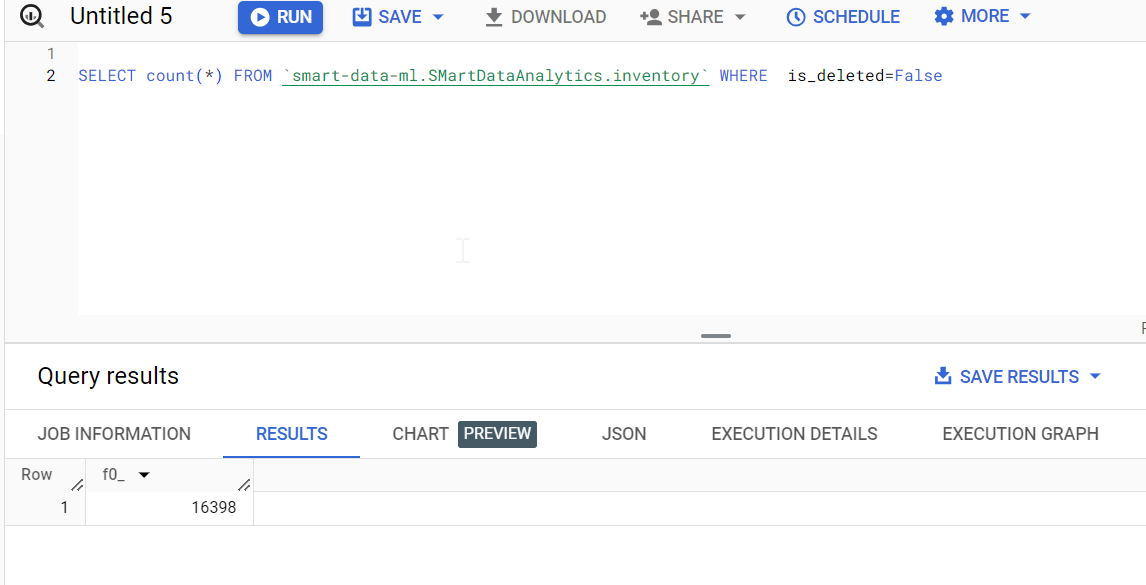


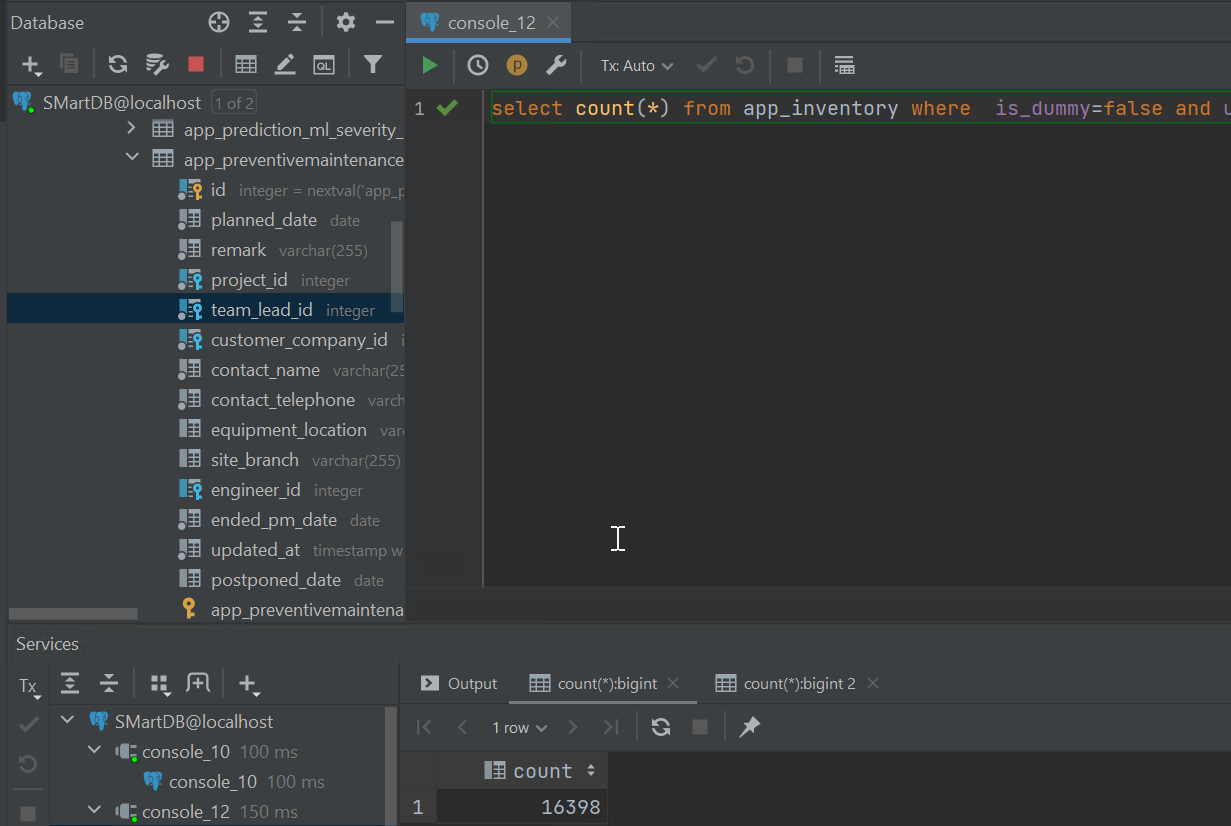
### Table result





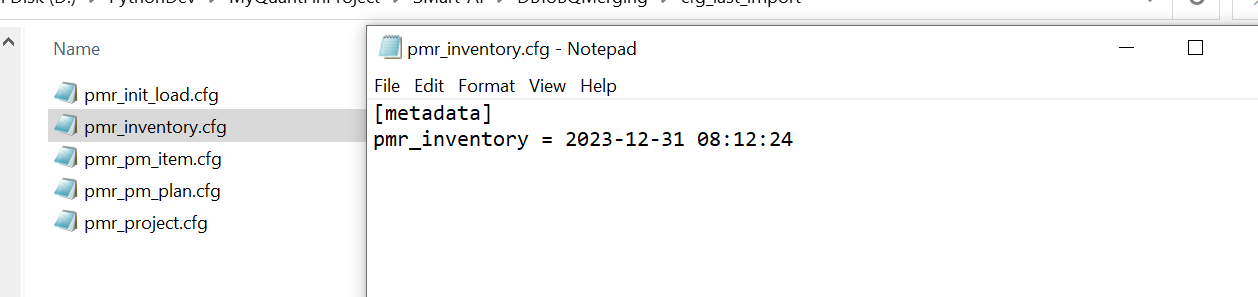
### Count total app\_inventory in database = table inventoru in BQ



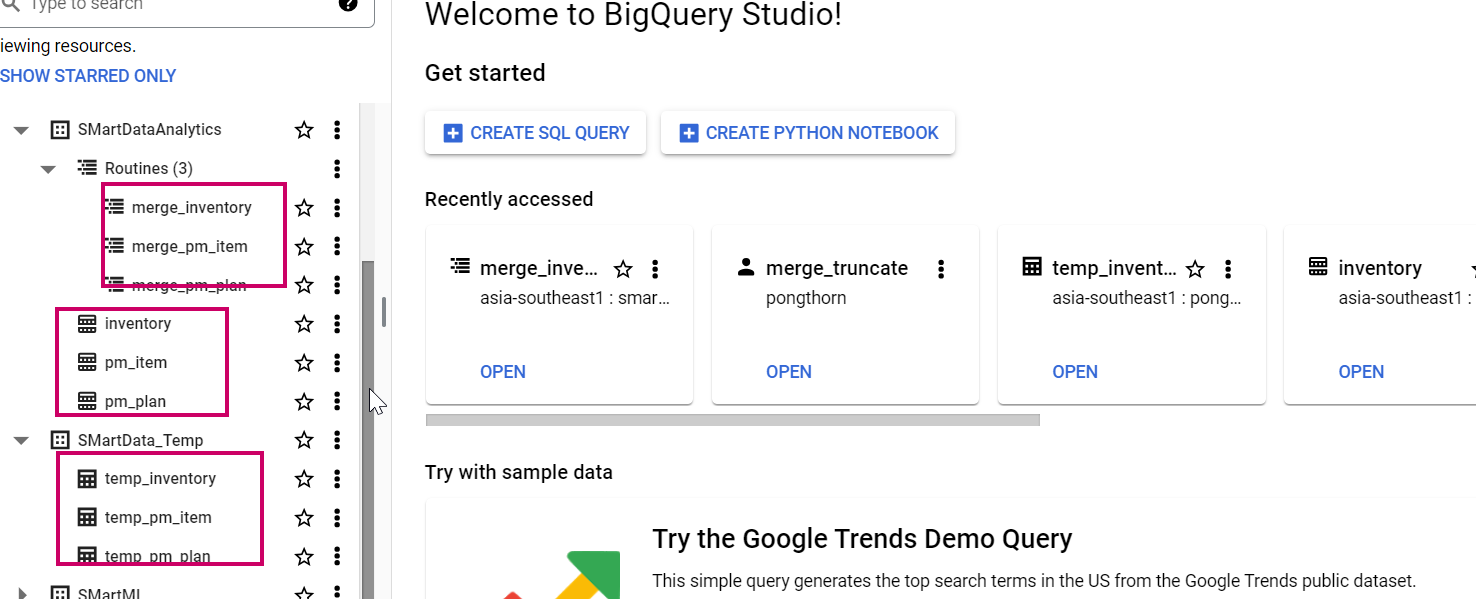


### Check timestpam update

Change from 2019-01-01 00:00:00 to 2023-12-31 08:12:24



# All the things are ready to run



# Set Job to run

