**SNOWFLAKE ASSIGNMENT**

(Sonam Kumari,TAS-204)

1. Create roles as per the below-mentioned hierarchy. Accountadmin already exists in Snowflake.

CREATE ROLE Admin;

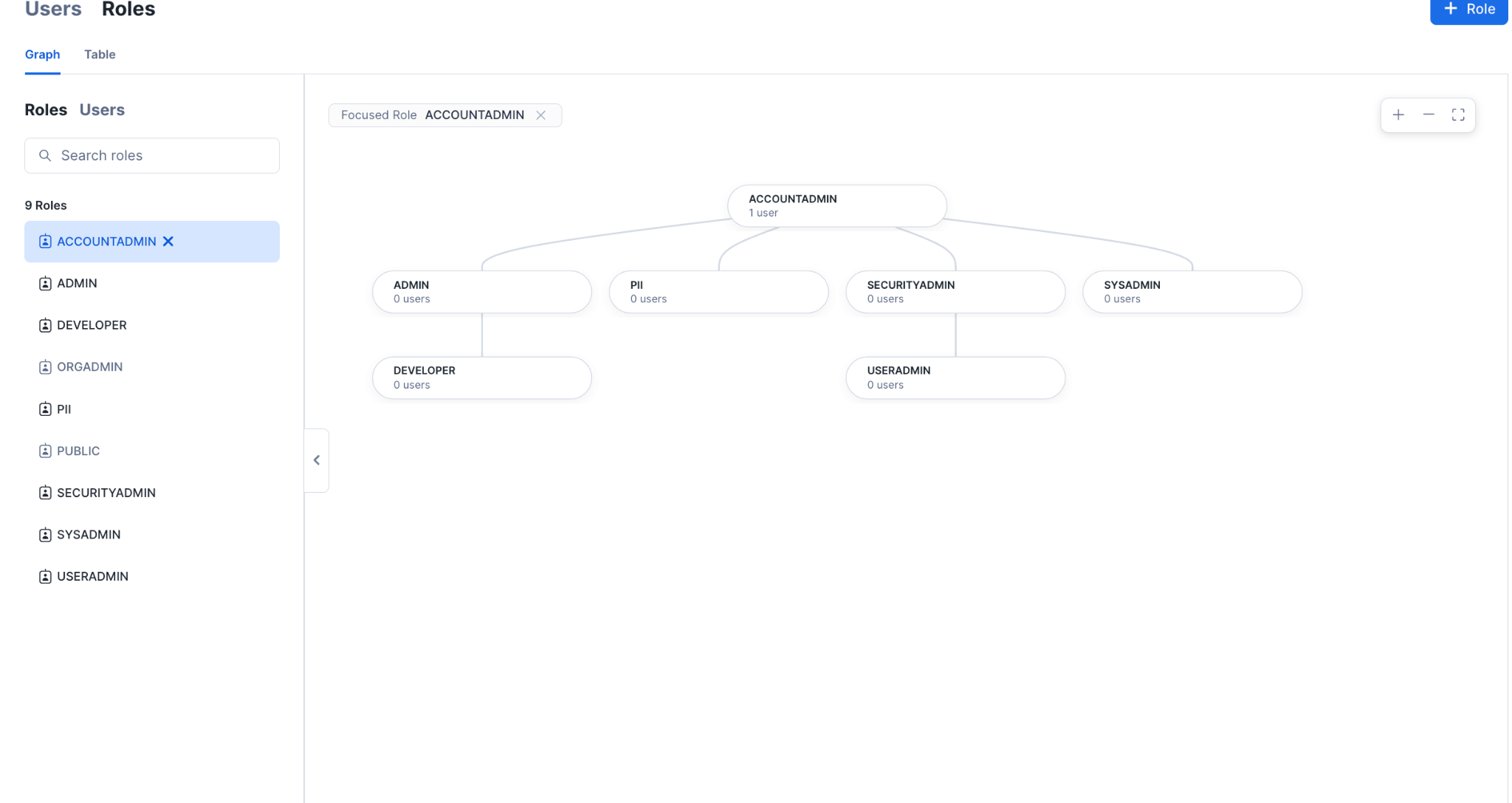
GRANT ROLE Admin TO ROLE Accountadmin;

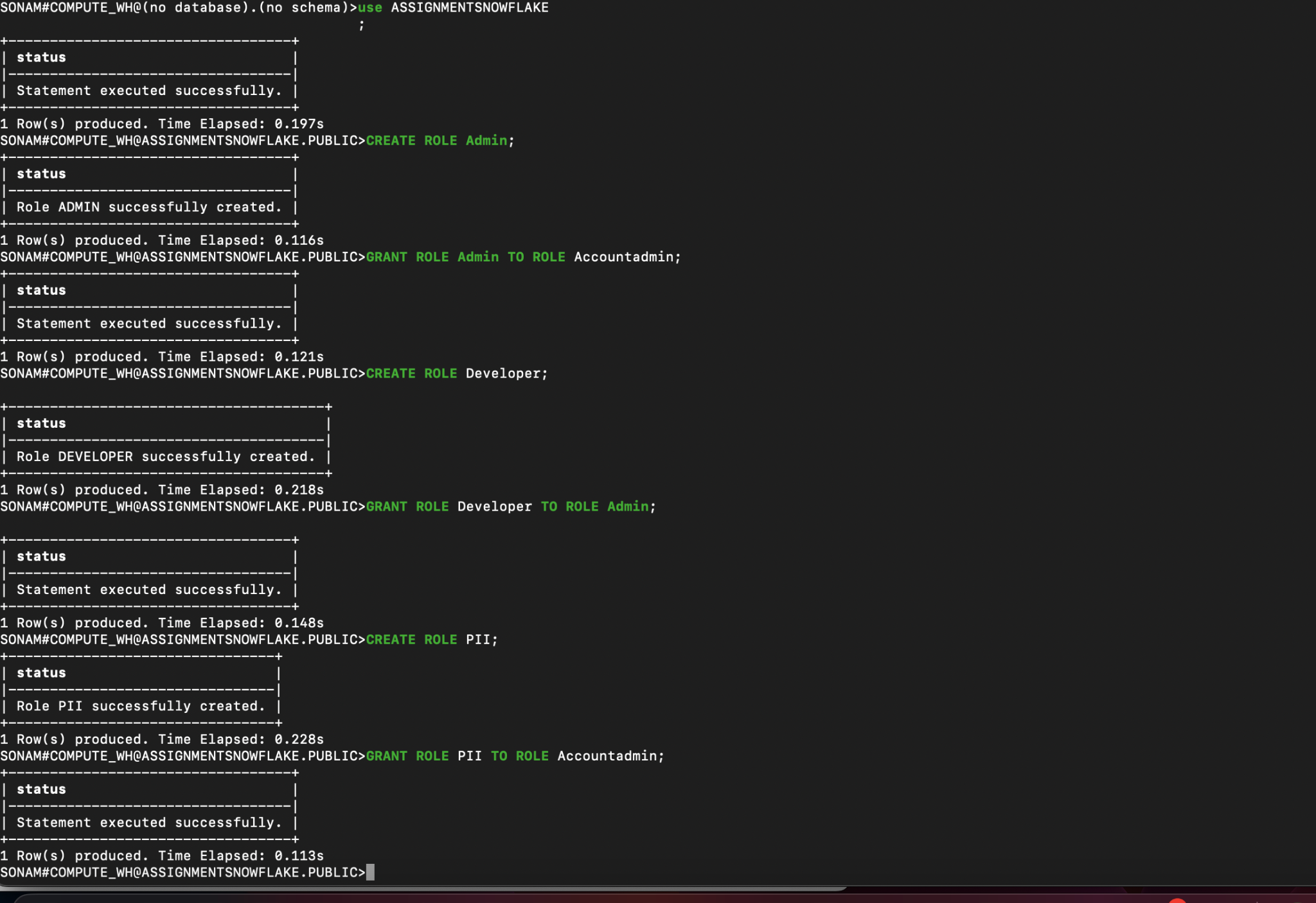
CREATE ROLE Developer;

GRANT ROLE Developer TO ROLE Admin;

CREATE ROLE PII;

GRANT ROLE PII TO ROLE Accountadmin;





1. Create an M-sized warehouse using the accountadmin role, name -> assignment\_wh and use it for all the queries

USE ROLE ACCOUNTADMIN;

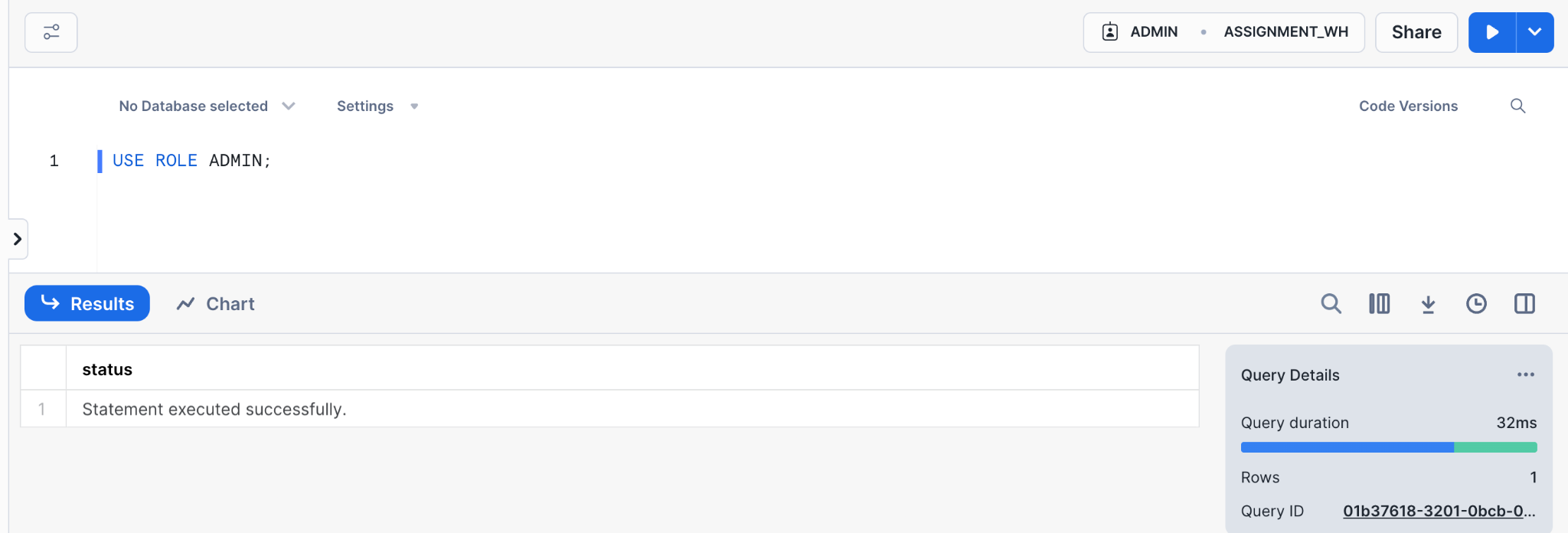
CREATE WAREHOUSE IF NOT EXISTS ASSIGNMENT\_WH

WITH WAREHOUSE\_SIZE = 'MEDIUM';



1. Switch to the admin role

**USE** **ROLE** **ADMIN**;



4) Create a database assignment\_db

**CREATE** **DATABASE** ASSIGNMENT\_DB;



5) Create a schema my\_schema

CREATE SCHEMA MY\_SCHEMA;

USE SCHEMA MY\_SCHEMA;

6) Create a table using any sample csv. You can get 1 by googling for sample csv’s. Preferably search for a sample employee dataset so that you have PII related columns else you can consider any column as PII ( 5 ).

CREATE TABLE employeeData1 (

Emp\_ID INT,

Name\_Prefix STRING,

First\_Name STRING,

Middle\_Initial STRING,

Last\_Name STRING,

Gender STRING,

E\_Mail STRING,

Fathers\_Name STRING,

Mothers\_Name STRING,

Mothers\_Maiden\_Name STRING,

Salary FLOAT,

elt\_ts TIMESTAMP DEFAULT CURRENT\_DATE,

elt\_by VARCHAR(50) DEFAULT 'SnowSqL',

file\_name VARCHAR(255)

);

7) Also, create a variant version of this dataset

CREATE OR REPLACE TABLE variant\_employeeData (

Emp\_ID INT,

Name\_Details VARIANT,

Contact\_Details VARIANT,

Family\_Details VARIANT,

Salary FLOAT,

elt\_ts TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

elt\_by VARCHAR(50) DEFAULT 'SnowSQL',

file\_name VARCHAR(255)

);

– To Load data into variant table

– Creating a staging table

CREATE OR REPLACE TEMPORARY TABLE staging\_employeeData (

Emp\_ID INT,

Name\_Prefix STRING,

First\_Name STRING,

Middle\_Initial STRING,

Last\_Name STRING,

Gender STRING,

E\_Mail STRING,

Fathers\_Name STRING,

Mothers\_Name STRING,

Mothers\_Maiden\_Name STRING,

Salary FLOAT

);

– Load data into the staging table

CREATE OR REPLACE STAGE staging\_employeeData

FILE\_FORMAT = my\_csv\_format;

**PUT** **file**:///**Users**/sonamkumari/Desktop/emp\_dataset.csv @staging\_employeeData;

COPY INTO staging\_employeeData

FROM @staging\_employeeData

FILE\_FORMAT = (TYPE = 'CSV', SKIP\_HEADER = 1)

ON\_ERROR = 'CONTINUE';

– Transform and insert data into final table

INSERT INTO variant\_employeeData (Emp\_ID, Name\_Details, Contact\_Details, Family\_Details, Salary, elt\_by, file\_name)

SELECT

Emp\_ID,

OBJECT\_CONSTRUCT(

'Name Prefix', Name\_Prefix,

'First Name', First\_Name,

'Middle Initial', Middle\_Initial,

'Last Name', Last\_Name,

'Gender', Gender

) AS Name\_Details,

OBJECT\_CONSTRUCT(

'E Mail', E\_Mail

) AS Contact\_Details,

OBJECT\_CONSTRUCT(

'Father''s Name', Fathers\_Name,

'Mother''s Name', Mothers\_Name,

'Mother''s Maiden Name', Mothers\_Maiden\_Name

) AS Family\_Details,

Salary,

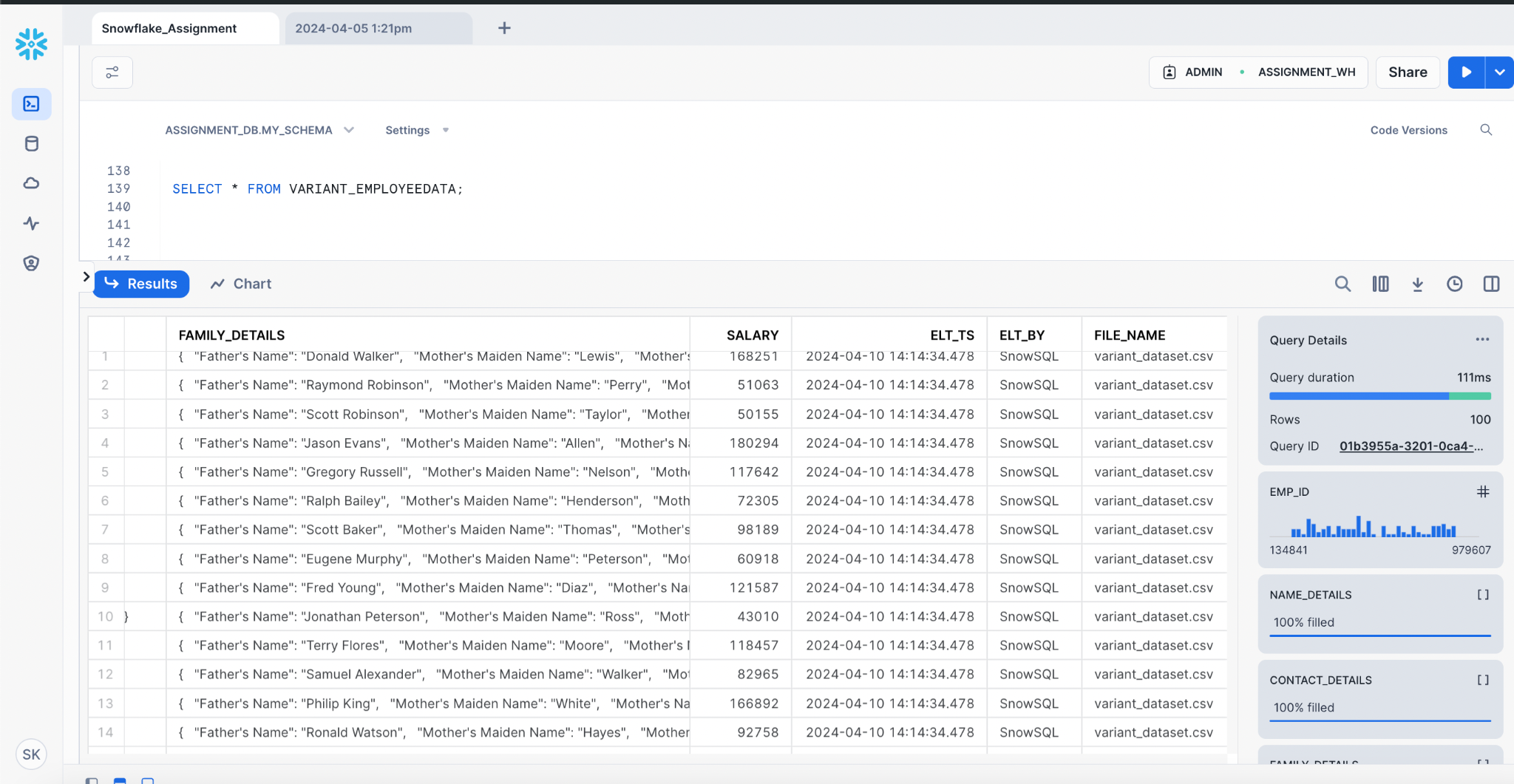
'SnowSQL', -- Assuming the elt\_by is hardcoded for this example

'variant\_dataset.csv' -- Assuming a single source file for simplicity

FROM

staging\_employeeData;





8) Load the file into an external and internal stage

AND

9) Load data into the tables using copy into statements. In one table load from the internal stage and in another from the external

*Internal Stage*

**CREATE** **OR** **REPLACE** **FILE** **FORMAT** my\_csv\_format

**TYPE** = 'CSV'

FIELD\_DELIMITER = ','

SKIP\_HEADER= 1;

**CREATE** **OR** **REPLACE** **STAGE** my\_stage

FILE\_FORMAT = my\_csv\_format;

**PUT** **file**:///**Users**/sonamkumari/Desktop/emp\_dataset.csv @my\_stage;



**COPY** **INTO** employeeData1 (

Emp\_ID,

Name\_Prefix,

First\_Name,

Middle\_Initial,

Last\_Name,

Gender,

E\_Mail,

Fathers\_Name,

Mothers\_Name,

Mothers\_Maiden\_Name,

Salary,

elt\_ts,

elt\_by,

file\_name

)

**FROM** (

**SELECT**

$1, $2, $3, $4, $5, $6, $7, $8, $9, $10, $11,

**CURRENT\_TIMESTAMP**,

'SnowSqL',

METADATA$FILENAME

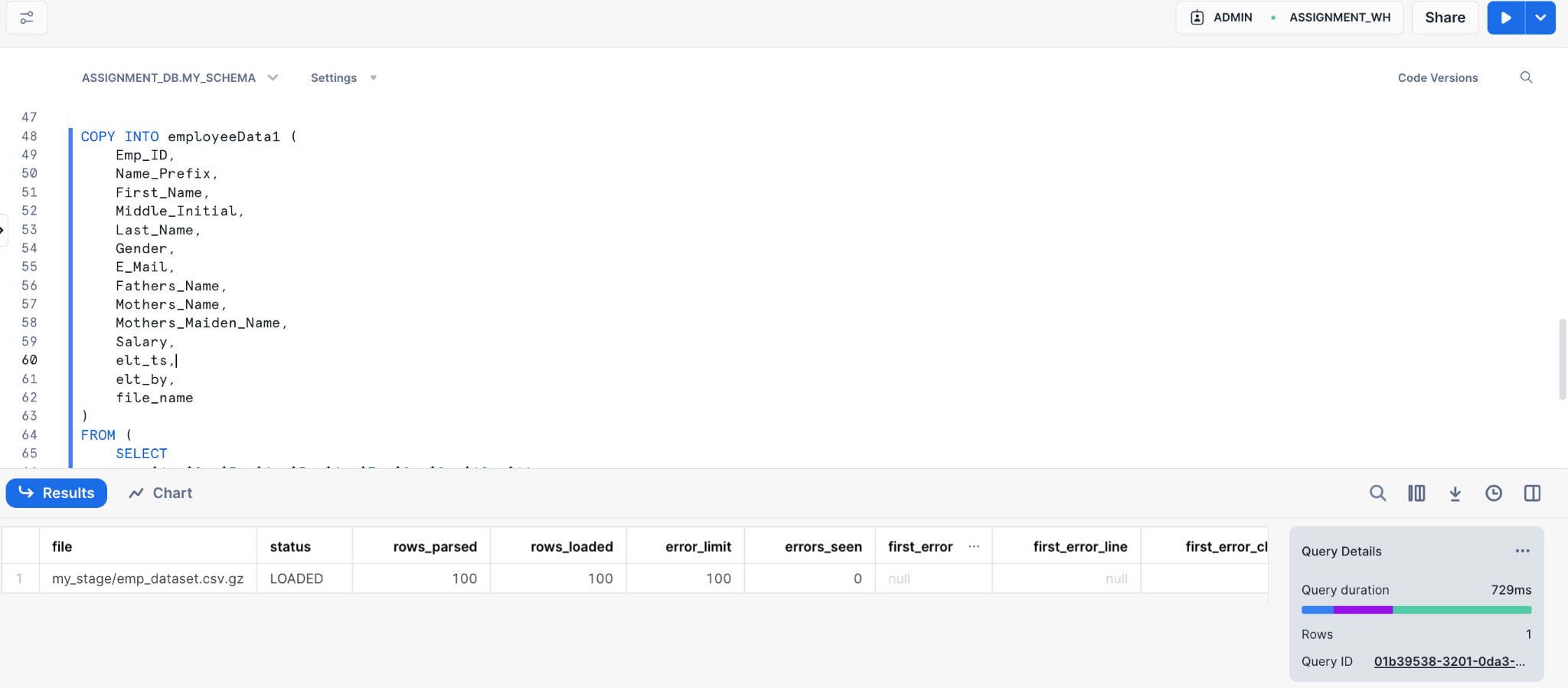
**FROM**

@my\_stage/emp\_dataset.csv.gz

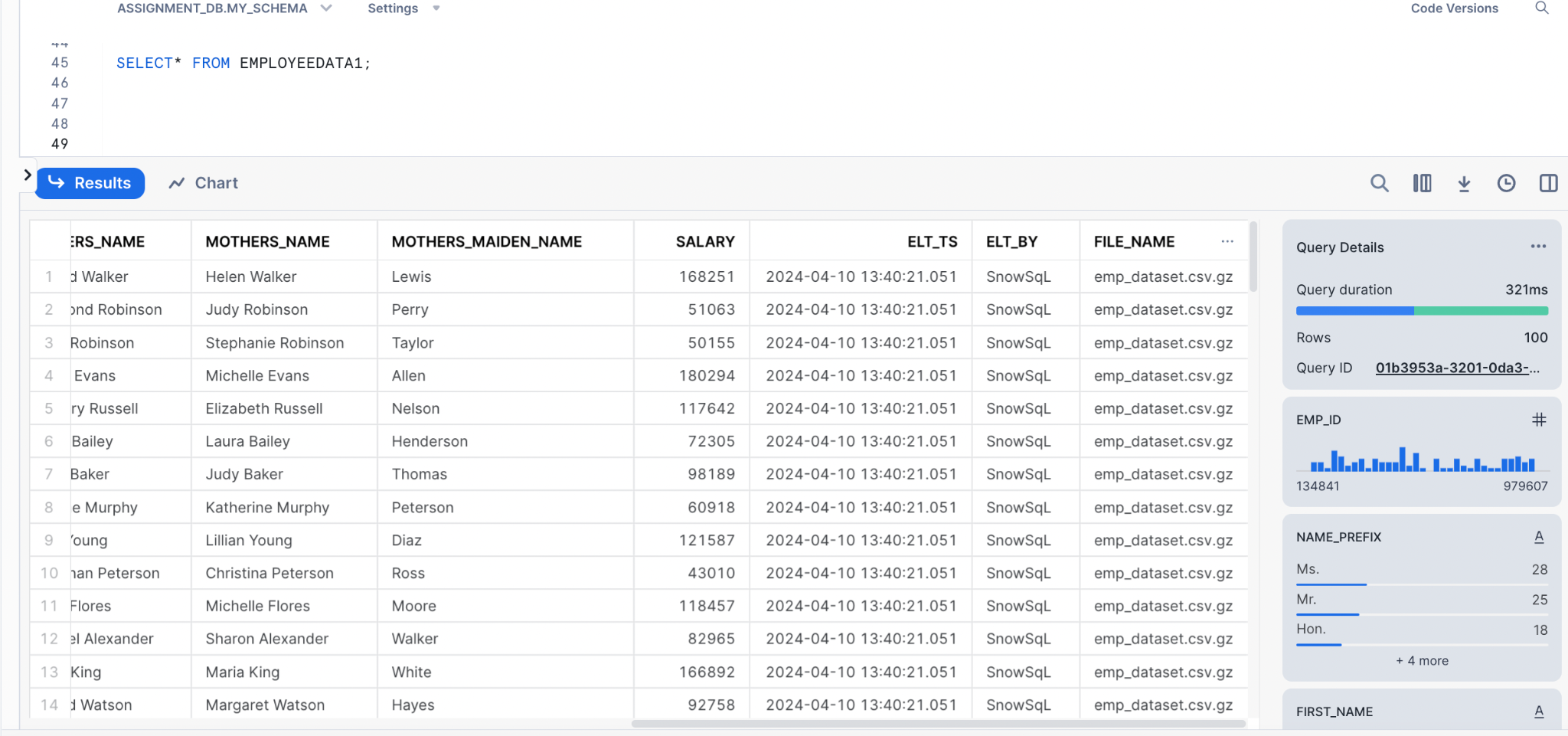
(file\_format => my\_csv\_format) **t**

)

ON\_ERROR = 'CONTINUE';



– Viewing the loaded data



*External Staging:*

**USE** **SCHEMA** EXTERNAL\_MY\_SCHEMA;

**CREATE** **OR** **REPLACE** **FILE** **FORMAT** my\_csv\_format

**TYPE** = 'CSV'

FIELD\_DELIMITER = ','

SKIP\_HEADER= 1;

**create** **or** **replace** **stage** external\_stage

url='azure://snowflakesonamassignment.blob.core.window

s.net/snowflakeassignmentcontainer/emp\_dataset.csv'

**CREDENTIALS**=(

azure\_sas\_token='sv=2022-11-02&ss=bfqt&srt=sco&sp=rwdl

acupiytfx&se=2024-04-05T20:01:09Z&st=2024-04-05T12:01:

09Z&spr=https&sig=6ZGlk5Do8eioascWWAy8W%2FPRMFqj4Np58N

Syr%2FobhXY%3D'

)

File\_format=my\_csv\_format;

**COPY** **INTO** employeeDataExternal (

Emp\_ID,

Name\_Prefix,

First\_Name,

Middle\_Initial,

Last\_Name,

Gender,

E\_Mail,

Fathers\_Name,

Mothers\_Name,

Mothers\_Maiden\_Name,

Salary,

elt\_ts,

elt\_by,

file\_name

)

**FROM** (

**SELECT**

$1, $2, $3, $4, $5, $6, $7, $8, $9, $10, $11,

**CURRENT\_TIMESTAMP**,

stamp

'SnowSqL',

METADATA$FILENAME

**FROM**

@external\_stage/emp\_dataset.csv.gz

(file\_format => my\_csv\_format) **t**

)

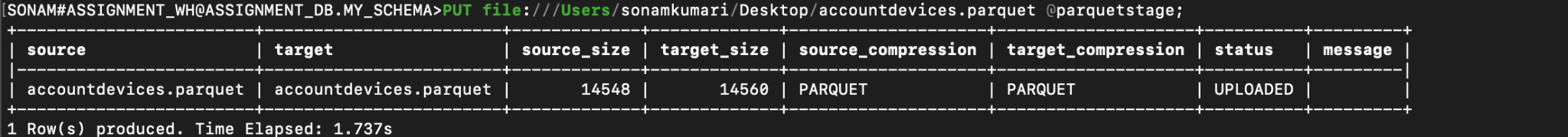
ON\_ERROR = 'CONTINUE';



10) Upload any parquet file to the stage location and infer the schema of the file

**CREATE** **OR** **REPLACE** **FILE** **FORMAT** parquet\_format **TYPE**=parquet;

**PUT** **file**:///**Users**/sonamkumari/Desktop/accountdevices.parquet @parquetstage;



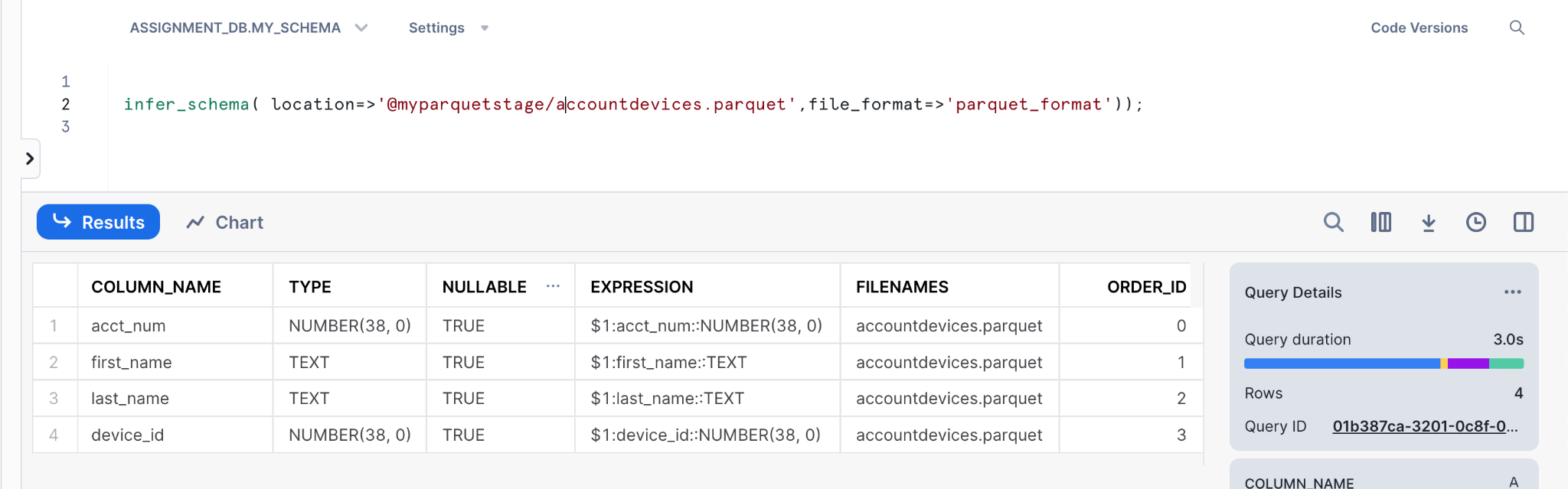
11) Run a select query on the staged parquet file without loading it to a snowflake table

**SELECT** \* **FROM** **TABLE**(

infer\_schema( **location**=>'@myparquetstage/accountdevices.parquet',

file\_format=>'parquet\_format')

);



12) Add masking policy to the PII columns such that fields like email, phone number, etc. show as \*\*masked\*\* to a user with the developer role. If the role is PII the value of these columns should be visible

**CREATE** **OR** **REPLACE** **MASKING** **POLICY** pii\_masking\_policy **AS** (val STRING) **RETURNS** STRING ->

**CASE**

**WHEN** **CURRENT\_ROLE**() **IN** ('PII') **THEN** val

**ELSE** '\*\*masked\*\*'

**END**;



**ALTER** **TABLE** EMPLOYEEDATA

**MODIFY** **COLUMN** E\_Mail **SET** **MASKING** **POLICY** pii\_masking\_policy;

**CREATE** **OR** **REPLACE** **MASKING** **POLICY** pii\_masking\_policy\_for\_salary **AS** (val FLOAT)

**RETURNS** FLOAT ->

**CASE**

**WHEN** **CURRENT\_ROLE**() **IN** ('PII') **THEN** val

**ELSE** -1.0

**END**;

**ALTER** **TABLE** EMPLOYEEDATA **MODIFY** **COLUMN** SALARY **SET** **MASKING** **POLICY** pii\_masking\_policy\_for\_salary;

USE ROLE ACCOUNTADMIN;

GRANT USAGE ON DATABASE ASSIGNMENT\_DB TO ROLE PII;

GRANT USAGE ON DATABASE ASSIGNMENT\_DB TO ROLE DEVELOPER;

GRANT USAGE ON SCHEMA ASSIGNMENT\_DB.MY\_SCHEMA TO ROLE PII;

GRANT USAGE ON SCHEMA ASSIGNMENT\_DB.MY\_SCHEMA TO ROLE DEVELOPER;

GRANT SELECT ON ALL TABLES IN SCHEMA ASSIGNMENT\_DB.MY\_SCHEMA TO ROLE PII;

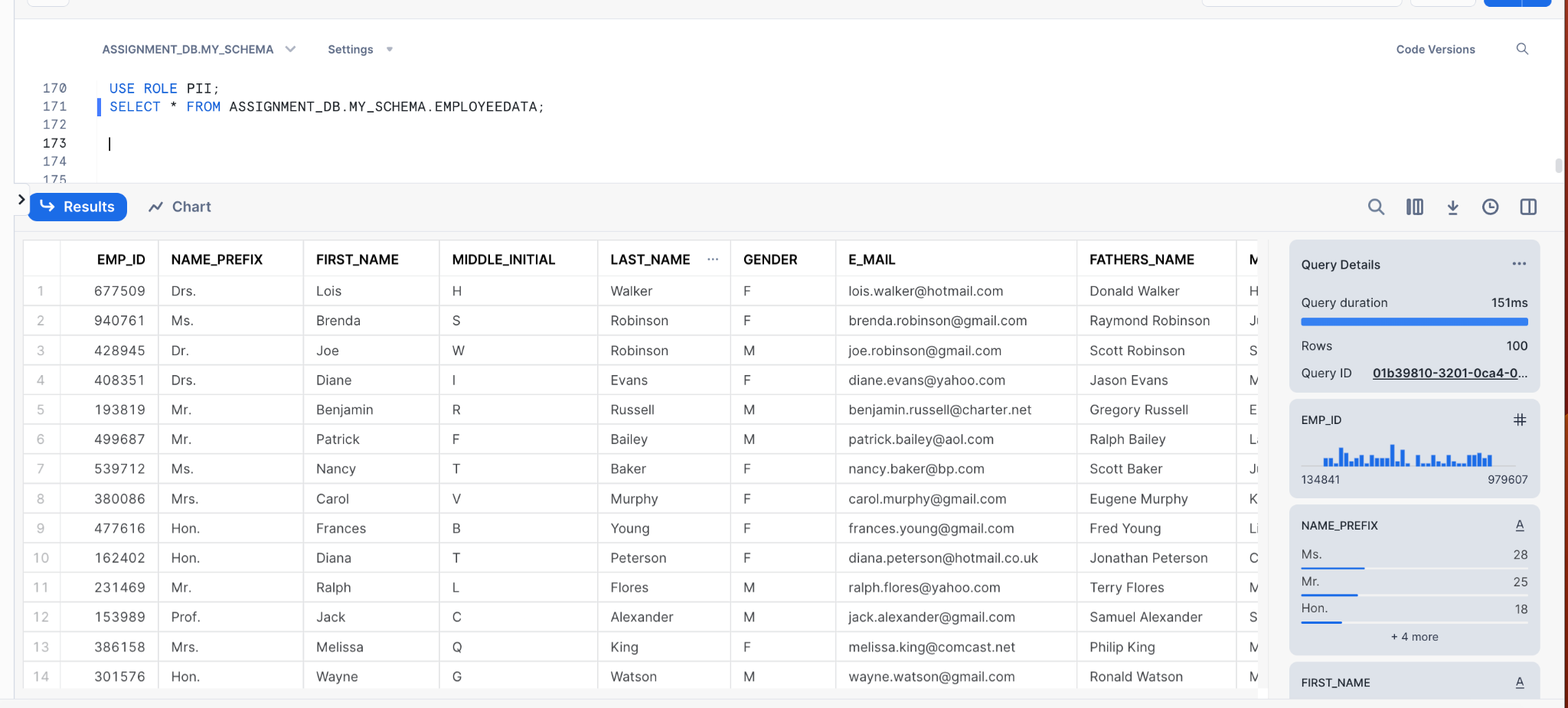
GRANT SELECT ON ALL TABLES IN SCHEMA ASSIGNMENT\_DB.MY\_SCHEMA TO ROLE DEVELOPER;

GRANT USAGE ON WAREHOUSE ASSIGNMENT\_WH TO ROLE PII;

GRANT USAGE ON WAREHOUSE ASSIGNMENT\_WH TO ROLE DEVELOPER;

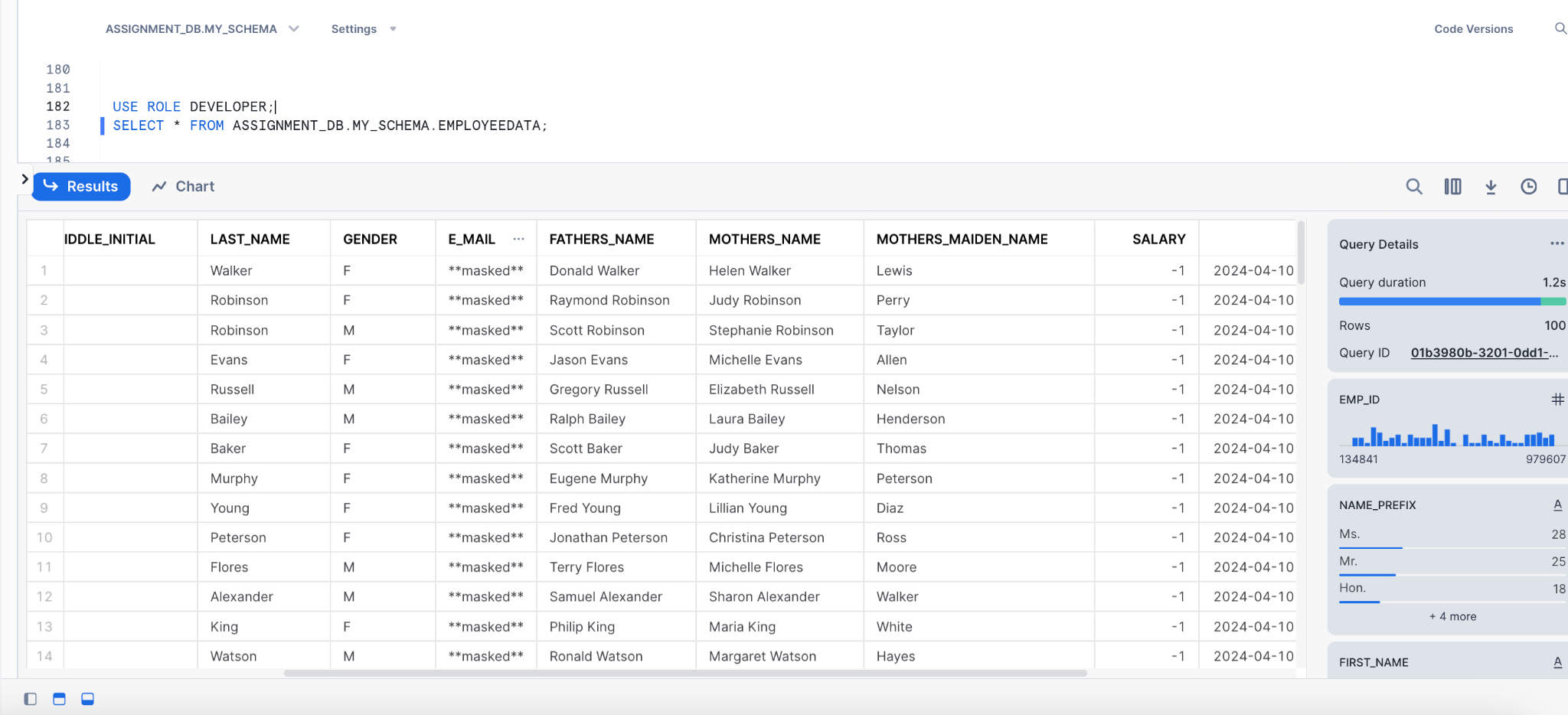
USE ROLE PII;

SELECT \* FROM ASSIGNMENT\_DB.MY\_SCHEMA.EMPLOYEEDATA;



USE ROLE DEVELOPER;

SELECT \* FROM ASSIGNMENT\_DB.MY\_SCHEMA.EMPLOYEEDATA;



—-x—x—