

# UNIT TEST REPORT

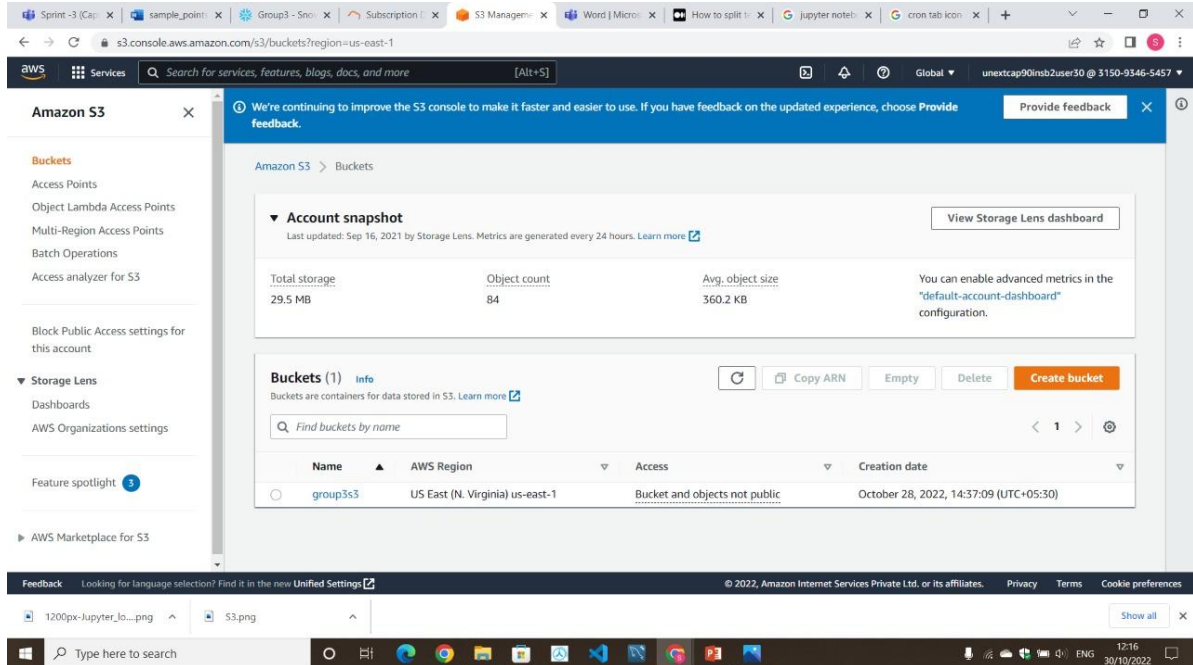
## Age Data set Analysis

### ABSTRACT OF THE PROJECT :

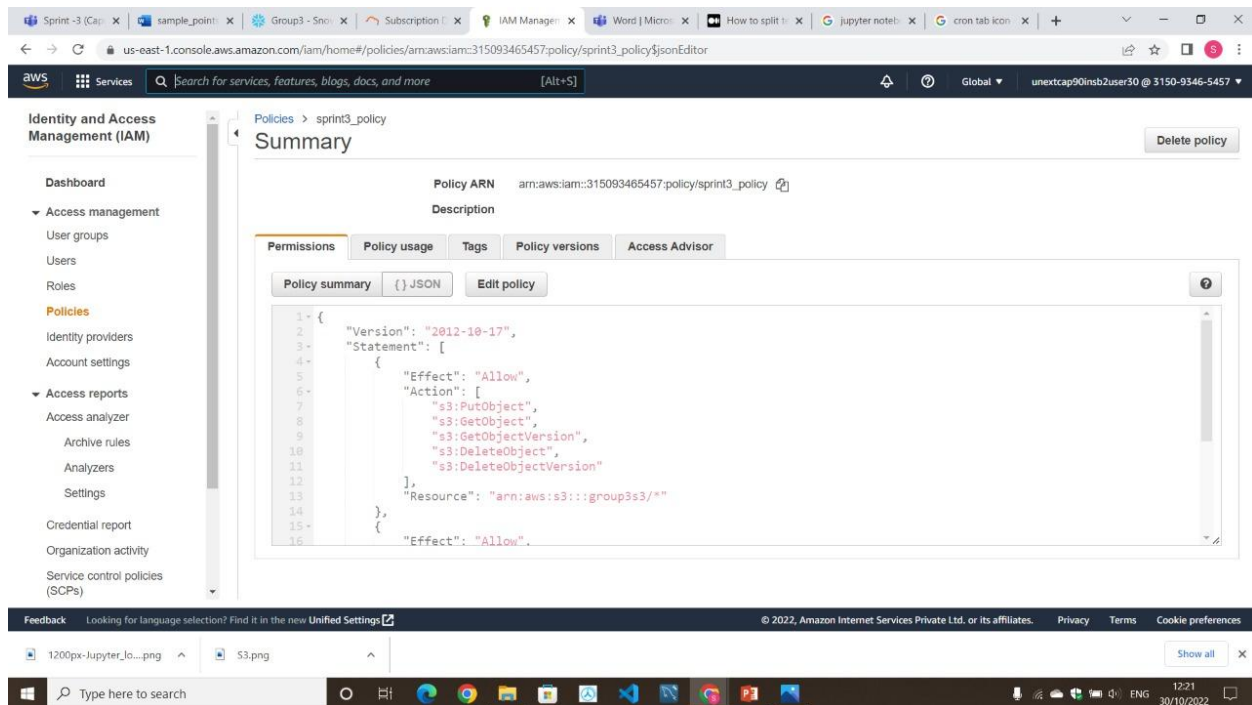
The project is to develop Analysis of Age Data datasets .

---

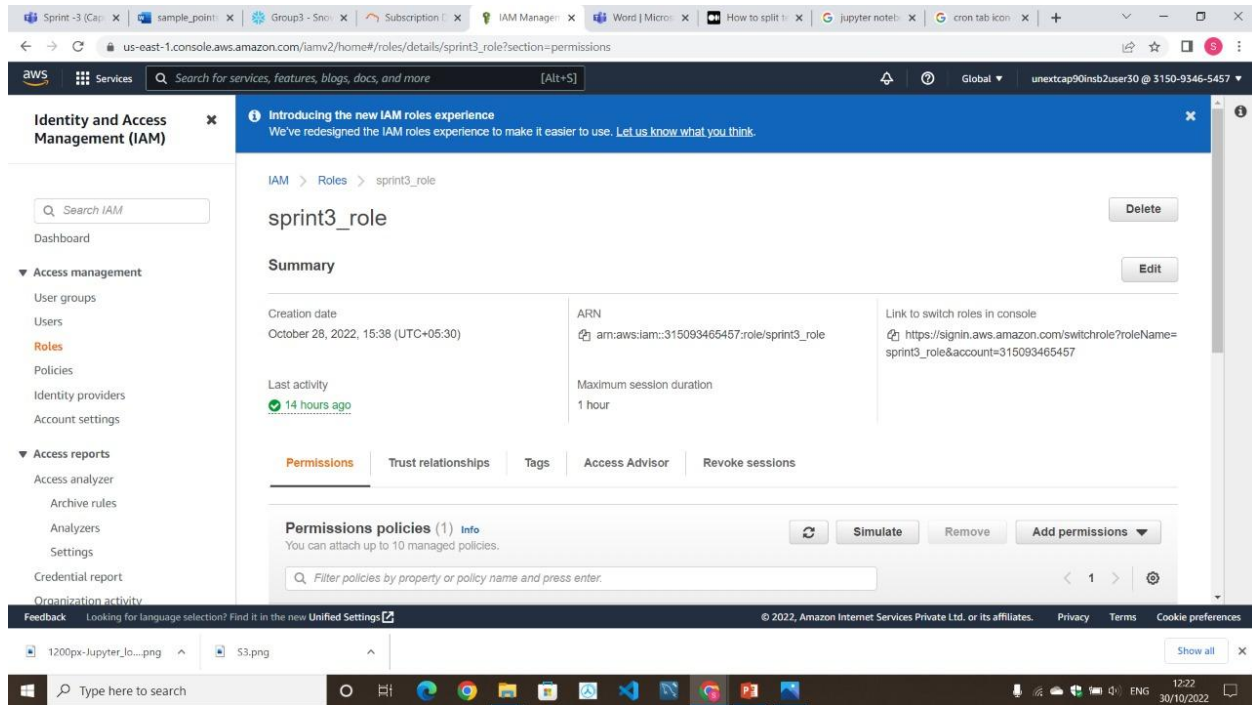
## STEP 1: Create a bucket and upload data into the bucket.



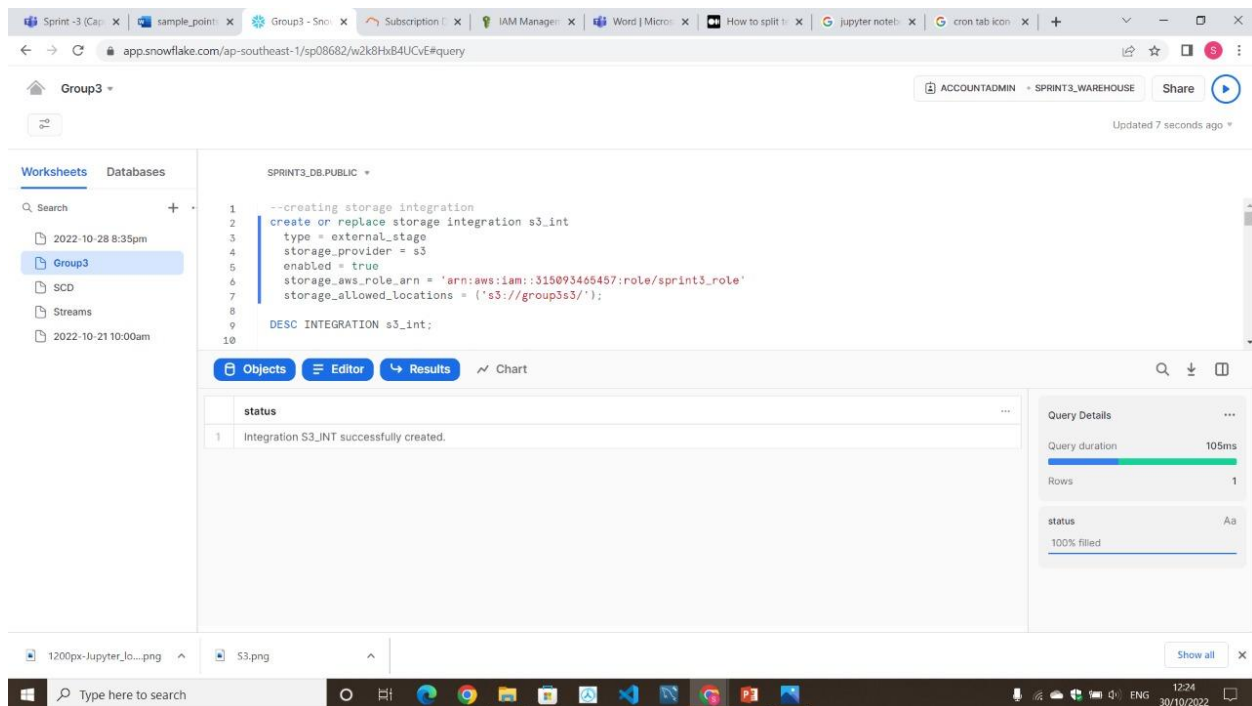
## STEP2: Create a policy.



## STEP 3: Created a role and attached a policy to that role.



## STEP4: Created an integration object.



## STEP5: We described the integration object.

The screenshot shows the Snowflake web interface. The left sidebar displays a list of databases and schemas, including 'Group3'. The main area shows the configuration for an external stage named 'SPRINT3\_DB.PUBLIC.s3\_int'. The configuration includes properties such as 'type', 'storage\_provider', 'storage\_aws\_role\_arn', and 'storage\_allowed\_locations'. Below the configuration, a table displays the properties and their values.

property	property_type	property_value	property_default
ENABLED	Boolean	true	false
STORAGE_PROVIDER	String	S3	
STORAGE_ALLOWED_LOCATIONS	List	s3://group3s3/	
STORAGE_BLOCKED_LOCATIONS	List		
STORAGE_AWS_IAM_USER_ARN	String	arn:aws:iam::790265975554:user/xep10000-s	
STORAGE_AWS_ROLE_ARN	String	arn:aws:iam::315093465457:role/sprint3_role	
STORAGE_AWS_EXTERNAL_ID	String	SP08682_SFCR0le=2_7vuH4j8GHgka4I35cnY811rcuTE=	
COMMENT	String		

## STEP6: After describing the integration object we updated ARN and External Id.

The screenshot shows the AWS IAM console 'Edit trust policy' page for the 'sprint3\_role'. The page displays the current trust policy JSON and a list of actions for STS. The 'AssumeRole' action is selected, and the 'StringEquals' condition is being edited to update the 'sts:ExternalId' value.

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Principal": {
7         "AWS": "arn:aws:iam::790265975554:user/xep10000-s"
8       },
9       "Action": "sts:AssumeRole",
10      "Condition": {
11        "StringEquals": {
12          "sts:ExternalId": "SP08682_SFCR0le=2_7vuH4j8GHgka4I35cnY811rcuTE="
13        }
14      }
15    }
16  ]
17 }
```

## STEP7: Created a file format for the files to be loaded.

The screenshot shows the Snowflake SQL Editor interface. The SQL query in the editor is as follows:

```
9
10
11 DESC INTEGRATION s3_int;
12
13 --creating file format for external stage
14 create or replace file format sprint3_db.public.my_csv_format
15 type = csv field_delimiter = ',' skip_header = 1 null_if = ('NULL', 'null') empty_field_as_null = true;
16
17 desc file format my_csv_format;
18
19 --creation of stage
20 create or replace stage my_s3_stage
```

The Results tab shows a single row with the status: "File format MY\_CSV\_FORMAT successfully created." The Query Details panel on the right indicates a query duration of 91ms and 1 row returned.

## STEP8: File format description.

The screenshot shows the Snowflake SQL Editor interface with the same SQL query as in Step 7. The Results tab now displays the properties of the file format in a table:

	property	property_type	property_value	property_default
1	TYPE	String	csv	CSV
2	RECORD_DELIMITER	String	\n	\n
3	FIELD_DELIMITER	String	,	,
4	FILE_EXTENSION	String		
5	SKIP_HEADER	Integer	1	0
6	DATE_FORMAT	String	AUTO	AUTO
7	TIME_FORMAT	String	AUTO	AUTO
8	TIMESTAMP_FORMAT	String	AUTO	AUTO
9	BINARY_FORMAT	String	UTF8	UTF8

The Query Details panel on the right indicates a query duration of 179ms and 22 rows returned.

## STEP9: Created an external stage.

The screenshot shows the Snowflake web interface. The left sidebar displays the 'Group3' database. The main editor area shows a SQL query in the 'SPRINT3\_DB.PUBLIC' schema. The query creates an external stage named 'my\_s3\_stage' with the following properties: storage integration 's3\_int', url 's3://group3s3/', and file format 'my\_csv\_format'. The query also lists the stage. The 'Results' tab shows a single row with the status: 'Stage area MY\_S3\_STAGE successfully created.' The 'Query Details' panel on the right indicates a query duration of 3.6s and 1 row returned.

```
SPRINT3_DB.PUBLIC >
15 desc file format my_csv_format;
16
17 --creation of stage
18 create or replace stage my_s3_stage
19   storage_integration = s3_int
20   url = 's3://group3s3/'
21   file_format = my_csv_format;
22
23 list @my_s3_stage;
24
25 --creating table
```

status
1 Stage area MY_S3_STAGE successfully created.

Query Details

- Query duration: 3.6s
- Rows: 1
- status: 100% filled

## STEP 10: List of contents in the external stage.

The screenshot shows the Snowflake web interface. The left sidebar displays the 'Group3' database. The main editor area shows a SQL query in the 'SPRINT3\_DB.PUBLIC' schema. The query lists the contents of the 'my\_s3\_stage' stage. The 'Results' tab shows a table with 6 rows, listing the files in the stage. The 'Query Details' panel on the right indicates a query duration of 5.4s and 6 rows returned.

```
SPRINT3_DB.PUBLIC >
21 file_format = my_csv_format;
22
23 list @my_s3_stage;
24
25 --creating table
26 create or replace table agedata(
27   id string, name string,
28   short_description string, gender string,
29   country string, occupation string,
30   birth_year string, death_year string,
```

	name	size	md5	last_modified
1	s3://group3s3/AgeDataset1.csv	8,762,223	8f3d12ee7b837352320fbb84caee3606	Fri, 28 Oct 2022 09:10:42 GMT
2	s3://group3s3/AgeDataset2.csv	8,064,289	e49c949e2758417b12fdf27c186f7a0e	Fri, 28 Oct 2022 09:11:07 GMT
3	s3://group3s3/AgeDataset3.csv	7,894,176	7e8cc21d331303d68217fccc28e92e0e	Fri, 28 Oct 2022 09:11:22 GMT
4	s3://group3s3/AgeDataset4.csv	7,738,448	389a229871826a956c1a14938f9c2a7	Fri, 28 Oct 2022 09:11:35 GMT
5	s3://group3s3/AgeDataset5.csv	8,121,624	a1b4d7946b3078b25fd330d7c55fad16	Fri, 28 Oct 2022 09:11:49 GMT
6	s3://group3s3/AgeDataset6.csv	8,289,276	f50e9f1de40669c4ad60053c67fff93a	Sat, 29 Oct 2022 05:06:05 GMT

Query Details

- Query duration: 5.4s
- Rows: 6
- name: 100% filled
- size: 123



## STEP11: Create a table to load data from s3 bucket.

The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the 'Group3' database and its contents. The main editor area shows the following SQL code:

```
--creating table
create or replace table agedata(
  id string, name string,
  short_description string, gender string,
  country string, occupation string,
  birth_year string, death_year string,
  manner_of_death string,
  age_of_death string
);
--loading data to table from stage
copy into agedata
```

The 'Results' tab shows a single row with the status 'Table AGEDATA successfully created.' The 'Query Details' panel on the right indicates a query duration of 159ms and 1 row returned.

## STEP 12: Copied the data into table.

The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the 'Group3' database and its contents. The main editor area shows the following SQL code:

```
--loading data to table from stage
copy into agedata
from @my_s3_stage
file_format = (type = csv field_optionally_enclosed_by = '')
pattern = '*.csv'
on_error='skip_file';
--creating task to schedule job at 12 AM IST hours on every thursday
CREATE TASK mytask_hour
WAREHOUSE = sprint3_warehouse
SCHEDULE = 'USING CRON 0 0 * * THU Asia/Kolkata'
TIMESTAMP_INPUT_FORMAT = 'YYYY-MM-DD HH24'
```

The 'Results' tab shows a table with the following columns: file, status, rows\_parsed, rows\_loaded, error\_limit, errors\_seen, and first\_error. The table contains 6 rows of data, all with a status of 'LOADED'.

	file	status	rows_parsed	rows_loaded	error_limit	errors_seen	first_error
1	s3://group3s3/AgeDataset4.csv	LOADED	81,535	81,535	1	0	null
2	s3://group3s3/AgeDataset2.csv	LOADED	81,535	81,535	1	0	null
3	s3://group3s3/AgeDataset3.csv	LOADED	81,535	81,535	1	0	null
4	s3://group3s3/AgeDataset5.csv	LOADED	81,535	81,535	1	0	null
5	s3://group3s3/AgeDataset6.csv	LOADED	81,535	81,535	1	0	null
6	s3://group3s3/AgeDataset1.csv	LOADED	81,535	81,535	1	0	null

The 'Query Details' panel on the right indicates a query duration of 13s and 6 rows returned.

## STEP 13: Created a scheduler to update data.

The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays a tree view with 'Group3' selected. The main editor area contains the following SQL code:

```
SPRINT3_DB.PUBLIC >  
--creating task to schedule job at 12 AM IST hours on every thursday  
CREATE TASK mytask_hour  
WAREHOUSE = sprint3_warehouse  
SCHEDULE = 'USING CRON 0 0 * * THU Asia/Kolkata'  
TIMESTAMP_INPUT_FORMAT = 'YYYY-MM-DD HH24'  
AS  
copy into agedata  
from @my_s3_stage  
file_format = (type = csv field_optionally_enclosed_by = '')  
pattern = '.*.csv'  
on_error='skip_file';
```

The 'Results' tab shows a single row with the status: 'Task MYTASK\_HOUR successfully created.' The 'Query Details' panel on the right indicates a query duration of 199ms and 1 row returned. The status is '100% filled'.

## STEP 14: Check the state of the task.

The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays a tree view with 'Group3' selected. The main editor area contains the following SQL code:

```
SPRINT3_DB.PUBLIC >  
-- Alternative method -- Creating Snowpipe  
on_error='skip_file';  
-- Check scheduled task  
show tasks;  
-- Put task in the schedule.  
alter task mytask_hour resume;  
alter task mytask_hour suspend;
```

The 'Results' tab shows a table with the following data:

owner	comment	warehouse	schedule	predecessors	state	definition
ACCOUNTADMIN		SPRINT3_WAREHOUSE	USING CRON 0 0 * * THU Asia/Kolkata	[]	suspended	copy into agedata

The 'Query Details' panel on the right indicates a query duration of 199ms and 1 row returned. The status is '100% filled'.



## STEP 15: Created pipe to auto ingest data into the table.

The screenshot shows the Snowflake web interface. The top navigation bar includes the user 'ACCOUNTADMIN' and the warehouse 'SPRINT3\_WAREHOUSE'. The left sidebar shows a list of databases, with 'Group3' selected. The main panel displays a SQL query in the 'Editor' tab:

```
SPRINT3_DB.PUBLIC >
-- Alternative method -- Creating Snowpipe
61 create or replace pipe sprint3_snowpipe auto_ingest=true as
62   copy into agedata
63   from @my_s3_stage;
64
65 desc pipe sprint3_snowpipe;
66
67 select * from agedata;
68
69 --to check the status of snowpipe
70
```

Below the query editor, the 'Results' tab shows a single row of data:

status
Pipe SPRINT3_SNOWPIPE successfully created.

The right sidebar displays 'Query Details' for the executed query:

- Query duration: 1.1s
- Rows: 1
- status: 100% filled

The bottom of the screen shows a Windows taskbar with various application icons and the system clock indicating 12:41 on 30/10/2022.

## STEP 16: Check the status of the pipe.

The screenshot shows the Snowflake web interface. The top navigation bar includes the user 'ACCOUNTADMIN' and the warehouse 'SPRINT3\_WAREHOUSE'. The left sidebar shows a list of databases, with 'Group3' selected. The main panel displays a SQL query in the 'Editor' tab:

```
SPRINT3_DB.PUBLIC >
-- pipe sprint3_snowpipe
67 select * from agedata;
68
69 --to check the status of snowpipe
70 select SYSTEM$PIPE_STATUS('sprint3_snowpipe');
71
72 --refresh the pipe
73 alter pipe sprint3_snowpipe refresh;
74
75 --creating a stream
76
```

Below the query editor, the 'Results' tab shows a single row of data:

SYSTEM\$PIPE_STATUS('SPRINT3_SNOWPIPE')
["executionState": "RUNNING", "pendingFileCount": 0, "notificationChannelName": "arn:aws:sqs:us-east-1:790265975554:sf-snowpipe-AIDA3P74F2MBOGR"]

The right sidebar displays 'Query Details' for the executed query:

- Query duration: 53ms
- Rows: 1
- SYSTEM\$PIPE\_STATUS('SPRINT3\_SNOWPIPE'): 100% filled

The bottom of the screen shows a Windows taskbar with various application icons and the system clock indicating 12:43 on 30/10/2022.

## STEP 17: Adding new file to bucket.

The screenshot shows the AWS S3 console interface. At the top, there's a navigation bar with the AWS logo and a search bar. Below it, a green banner indicates a successful upload. The main content area shows a summary of the upload operation, including the destination bucket (s3://group3s3) and the status (Succeeded). A table below lists the files and folders, showing one file named 'AgeDataset7.csv' with a size of 7.6 MB and a status of 'Succeeded'.

How would you rate your experience with this service console? ☆☆☆☆

aws Services Search for services, features, blogs, docs, and more [Alt+S]

Global unextcap90lmb2user30 @ 3150-9346-5457

We're continuing to improve the S3 console to make it faster and easier to use. If you have feedback on the updated experience, choose [Provide feedback](#).

**Upload succeeded**  
View details below.

**Summary**

Destination s3://group3s3	Succeeded 1 file, 7.6 MB (100.00%)	Failed 0 files, 0 B (0%)
------------------------------	---------------------------------------	-----------------------------

**Files and folders** Configuration

**Files and folders** (1 Total, 7.6 MB)

Find by name

Name	Folder	Type	Size	Status	Error
AgeDataset7.csv	-	text/csv	7.6 MB	Succeeded	-

Feedback Looking for language selection? Find it in the new Unified Settings

© 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

1200px-Jupyter\_lo...png S3.png Show all

Type here to search

12:47 30/10/2022

## STEP 18: New file added successfully.

The screenshot shows the Snowflake interface. The top navigation bar includes the Snowflake logo and a search bar. Below it, a banner indicates the query execution status. The main content area shows a SQL query in the editor, which has been executed successfully. The results pane displays a table with one row, indicating the file 'AgeDataset7.csv' has been uploaded. The query details pane on the right shows the query duration and the number of rows returned.

Group3

ACCOUNTADMIN SPRINT3\_WAREHOUSE Share

Updated 8 seconds ago

**Worksheets** Databases

Search

- 2022-10-28 8:35pm
- Group3
- SCD
- Streams
- 2022-10-21 10:00am

**SPRINT3\_DB.PUBLIC**

```
66 --refresh the pipe
67
68 select * from agedata;
69
70 --to check the status of snowpipe
71 select SYSTEM$PIPE_STATUS('sprint3_snowpipe');
72
73 --refresh the pipe
74 alter pipe sprint3_snowpipe refresh;
75
76 --creating a stream
```

**Objects** Editor Results Chart

File	Status
AgeDataset7.csv	SENT

**Query Details**

Query duration 1.3s

Rows 1

File 100% filled

Status 100% filled

1200px-Jupyter\_lo...png S3.png Show all

Type here to search

12:48 30/10/2022

## STEP 19: Creating stream.

The screenshot shows the Snowflake SQL Editor interface. The top navigation bar includes tabs for various applications and the current session. The main editor area displays a SQL query in the 'Editor' tab, which creates a stream named 'AGEDATA\_CHECK' on the 'AGEDATA' table. The query is as follows:

```
--creating a stream
create or replace stream agedata_check on table agedata;

select * from agedata_check;

select * from agedata;

--creating target table
create or replace table sprint3_target_t (id string, name string, short_description string, gender string,
country string, occupation string, birth_year string, death_year string,
```

The 'Results' tab shows a single row with the status 'Stream AGEDATA\_CHECK successfully created.' The 'Query Details' panel on the right indicates a query duration of 275ms and 1 row returned. The bottom status bar shows the current time as 12:50 on 30/10/2022.

## STEP 20: Created a table to store the dml changes captured by stream.

The screenshot shows the Snowflake SQL Editor interface. The main editor area displays a SQL query in the 'Editor' tab, which creates a target table named 'sprint3\_target\_t' and updates the 'AGEDATA' table. The query is as follows:

```
--creating target table
create or replace table sprint3_target_t (id string, name string, short_description string, gender string,
country string, occupation string, birth_year string, death_year string,
manner_of_death string, age_of_death string, stream_type string default null,
rec_version number default 0, REC_DATE TIMESTAMP_LTZ);

--make the changes in the record
update agedata set name = 'Tom downey' where id='023';
```

The 'Results' tab shows a single row with the status 'Table SPRINT3\_TARGET\_T successfully created.' The 'Query Details' panel on the right indicates a query duration of 174ms and 1 row returned. The bottom status bar shows the current time as 12:51 on 30/10/2022.

## STEP21: Updation of record.

The screenshot shows the Snowflake web interface. The query editor displays the following SQL code:

```
SPRINT3_DB.PUBLIC >  
85     country string, occupation string, birth_year string, death_year string,  
86     manner_of_death string, age_of_death string, stream_type string default null,  
87     rec_version number default 0, REC_DATE TIMESTAMP_LTZ);  
88  
89 --updating the record  
90 update agedata set name = 'Tom downey' where id='023';  
91  
92 --check  
93 select * from agedata_check;
```

The Results tab shows the following table:

	number of rows updated	number of multi-joined rows updated
1	1	0

The Query Details panel on the right shows:

- Query duration: 850ms
- Rows: 1
- number of rows updated: 123 (100% filled)
- number of multi-joined rows updated: 123 (100% filled)

## STEP 22: Shows that the rows have been updated.

The screenshot shows the Snowflake web interface. The query editor displays the following SQL code:

```
SPRINT3_DB.PUBLIC >  
94 --resume task  
95 alter task mytask_hour resume;  
96  
97 merge into sprint3_target_t t  
98 using agedata_check a  
99 on t.id=a.id and (metadata$action='DELETE')  
100 when matched and metadata$update='FALSE' then update set rec_version=9999, stream_type='DELETE';  
101 when matched and metadata$update='TRUE' then update set rec_version=rec_version-1, stream_type='UPDATE';  
102 when not matched then insert (id, name, short_description, gender, country, occupation, birth_year, death_year, manner_of_death,  
103 age_of_death, stream_type, rec_version, REC_DATE) values(a.id, a.name, a.short_description, a.gender, a.country, a.occupation, a.birth_year,  
104 a.death_year,  
105 a.manner_of_death, a.age_of_death, metadata$action, 0, CURRENT_TIMESTAMP());  
106  
107 --to see the history of records  
108 select * from sprint3_target_t;  
109  
110 --suspend task  
111 alter task mytask_hour suspend;
```

The Results tab shows the following table:

	number of rows inserted	number of rows updated
1	2	0

The Query Details panel on the right shows:

- Query duration: 1.1s
- Rows: 1

## STEP 23: History of Records

The screenshot shows the Snowflake SQL Editor interface. The query editor contains the following SQL code:

```
SPRINT3_DB.PUBLIC >  
a.death_year,  
a.manner_of_death, a.age_of_death, metadata$action,0,CURRENT_TIMESTAMP());  
--to see the history of records  
select id, name, stream_type, rec_version, rec_date from sprint3_target_t;  
--suspend task  
alter task mytask_hour suspend;
```

The query results are displayed in a table with the following columns: ID, NAME, STREAM\_TYPE, REC\_VERSION, and REC\_DATE. The results show two records:

ID	NAME	STREAM_TYPE	REC_VERSION	REC_DATE	
1	Q23	Tom downey	INSERT	0	2022-10-30 00:24:57.767 -0700
2	Q23	George Washington	DELETE	0	2022-10-30 00:24:57.767 -0700

The right sidebar shows the Query Details panel with the following information:

- Query duration: 45ms
- Rows: 2
- ID: 100% filled
- NAME: 100% filled
- STREAM\_TYPE: 100% filled

## STEP 24: Implementing row-level security

The screenshot shows the Snowflake SQL Editor interface. The query editor contains the following SQL code:

```
--Row Level security  
--creating role  
create or replace role United_States_of_America;  
create or replace role Kingdom_of_France;  
--table->agedata_roles  
create or replace table agedata_roles( agedata_role_name varchar,  
agedata_role_alias varchar);  
insert into agedata_roles values('United_States_of_America','United States of America'),  
( 'Kingdom_of_France','Kingdom of France'),('Kingdom_of_France','France');  
select * from agedata_roles;
```

The query results are displayed in a table with the following columns: AGEDATA\_ROLE\_NAME and AGEDATA\_ROLE\_ALIAS. The results show three records:

AGEDATA_ROLE_NAME	AGEDATA_ROLE_ALIAS
United_States_of_America	United States of America
Kingdom_of_France	Kingdom of France
Kingdom_of_France	France

The right sidebar shows the Query Details panel with the following information:

- Query duration: 259ms
- Rows: 3
- AGEDATA\_ROLE\_NAME: 100% filled

## STEP 25: Creating user and assigning role to that particular user

The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the 'Group3' workspace with a search bar and a list of objects: '2022-10-28 8:35pm', 'Group3', 'SCD', and 'Streams'. The main editor area shows a SQL script for creating two users and assigning roles. The script is as follows:

```
--creating users
126 create or replace user "George Washington" password = 'temp123' default_role = 'United_States_of_America';
127 grant role United_States_of_America to user "George Washington";
128
129
130 create or replace user "François Villon" password = 'temp123' default_role = 'Kingdom_of_France';
131 grant role Kingdom_of_France to user "François Villon";
132
133 select current_user();
134
135 --assigning role to particular user
136 grant role United_States_of_America to user ADITISHINDE;
137 grant role Kingdom_of_France to user ADITISHINDE;
138
```

The 'Results' tab shows a single row with the status 'Statement executed successfully.' The 'Query Details' panel on the right indicates a query duration of 56ms and 1 row returned. The status bar at the bottom shows the system time as 13:05 on 30/10/2022.

## STEP 26: Creating a secure view.

The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the 'Group3' workspace with a search bar and a list of objects: '2022-10-28 8:35pm', 'Group3', 'SCD', and 'Streams'. The main editor area shows a SQL script for creating a secure view and granting privileges. The script is as follows:

```
--creating secure view
139 create or replace secure view vw_age as
140 select a.*
141 from agedata a
142 where upper(a.country) in (select upper(agedata_role_alias) from agedata_roles
143 where upper(agedata_role_name) = upper(current_role()));
144
145
146 grant select on view vw_age to role United_States_of_America;
147 grant select on view vw_age to role Kingdom_of_France;
148
149 grant usage on warehouse sprint3_warehouse to role United_States_of_America;
150 grant usage on warehouse sprint3_warehouse to role Kingdom_of_France;
151
152 grant usage on database sprint3_db to role United_States_of_America;
153 grant usage on database sprint3_db to role Kingdom_of_France;
154
155 grant usage on schema public to role United_States_of_America;
156 grant usage on schema public to role Kingdom_of_France;
```

The 'Results' tab shows a single row with the status 'View VW\_AGE successfully created.' The 'Query Details' panel on the right indicates a query duration of 240ms and 1 row returned. The status bar at the bottom shows the system time as 13:06 on 30/10/2022.



## STEP 27: Granting permissions to the user's

The screenshot shows the Snowflake web interface. The left sidebar displays a search bar and a list of objects: 2022-10-28 8:35pm, Group3, SCD, Streams, and 2022-10-21 10:00am. The main panel shows the 'SPRINT3\_DB.PUBLIC' schema with the following SQL commands:

```
grant select on view vw_age to role United_States_of_America;
grant select on view vw_age to role Kingdom_of_France;

grant usage on warehouse sprint3_warehouse to role United_States_of_America;
grant usage on warehouse sprint3_warehouse to role Kingdom_of_France;

grant usage on database sprint3_db to role United_States_of_America;
grant usage on database sprint3_db to role Kingdom_of_France;

grant usage on schema public to role United_States_of_America;
grant usage on schema public to role Kingdom_of_France;
```

The 'Results' tab shows a single row with the status 'Statement executed successfully.' The 'Query Details' panel on the right indicates a query duration of 51ms and 1 row returned.

## STEP 28: View for role United\_states\_of\_America.

The screenshot shows the Snowflake web interface. The left sidebar is the same as in Step 27. The main panel shows the 'SPRINT3\_DB.PUBLIC' schema with the following SQL commands:

```
-- Verify the rows for United_States_of_America role
use role United_States_of_America;
use database sprint3_db;
use schema public;
select id, name, country from vw_age;
```

The 'Results' tab displays a table with 10 rows of data:

ID	NAME	COUNTRY
1	Q2275441	Charles Albert Wiley
2	Q2275751	Oliver Payne Pearson
3	Q2275886	Seymour Chatman
4	Q2275891	Seymour Felix
5	Q2275917	Seymour Magoon
6	Q2275923	Seymour Parker Gilbert
7	Q2275925	Bob Armstrong
8	Q2276420	Victor Ernest Shelford
9	Q2276480	Shailer Mathews
10	Q2276591	Shakir Stewart

The 'Query Details' panel on the right indicates a query duration of 543ms and 10 rows returned. A warning message states: 'Result limit exceeded. The query result exceeded our maximum size of 10k rows. This will limit the number of rows shown in the table.'

## STEP 29: View for the role of kindom\_of\_france.

The screenshot shows the Snowflake SQL Editor interface. The query being executed is:

```
-- Verify the rows for Kingdom_of_France
use role KINGDOM_OF_FRANCE;
use database sprint3_db;
use schema public;
select id, name, country from vw_age;
```

The query results are displayed in a table with the following columns: ID, NAME, and COUNTRY. The results show 15 rows of data, including names like Pierre Pruvost, Paul de Froment, and Estienne de La Roche.

A warning message on the right indicates: "Result limit exceeded. The query result exceeded our maximum size of 10k rows. This will limit the number of rows shown in the table." The query duration is 668ms.

## STEP 30: Implementing Column level security and creating masking policy.

The screenshot shows the Snowflake SQL Editor interface. The query being executed is:

```
--column level security
create or replace table mod_1 as select * from agedata;

--creating masking policy
CREATE MASKING POLICY SPRINT3_DB.PUBLIC.agedata_mask AS (VAL STRING) RETURNS STRING =>
CASE
  WHEN CURRENT_ROLE() IN ('mod_agedata') THEN VAL
  ELSE '*****'
END;

--creating role
create role mod_agedata;

create or replace table mod_2(manner_of_death1 string masking policy agedata_mask, manner_of_death2 string);

insert into mod_2(manner_of_death1, manner_of_death2)
select manner_of_death, manner_of_death from agedata;

--grant permission to the role
```

The query results show a single row with the value "number of rows inserted" and the value "489,210".

The query duration is 693ms, and the number of rows is 1.

## STEP 31: Column level security

The screenshot shows the Snowflake web interface. The top navigation bar includes the user 'ACCOUNTADMIN' and the warehouse 'SPRINT3\_WAREHOUSE'. The left sidebar shows a tree view with 'Group3' selected. The main area displays a SQL query in the 'Editor' tab:

```
SPRINT3_DB.PUBLIC >
188 --grant permission to the role
189 GRANT SELECT ON sprint3_db.PUBLIC.mod_2 TO ROLE mod_agedata;
190 grant usage on warehouse sprint3_warehouse to role mod_agedata;
191
192 grant usage on database sprint3_db to role mod_agedata;
193
194 grant usage on schema public to role mod_agedata;
195
196 grant role mod_agedata to user ADITISHINDE;
197
198 --fetch the result
199 select * from mod_2;
200
201
```

Below the query, the 'Results' tab shows a table with two columns: 'MANNER\_OF\_DEATH1' and 'MANNER\_OF\_DEATH2'. The table contains 6 rows of data:

MANNER_OF_DEATH1	MANNER_OF_DEATH2
37 *****	null
38 *****	natural causes
39 *****	capital punishment
40 *****	natural causes
41 *****	null
42 *****	null

A message on the right side of the results table states: 'Result limit exceeded. The query result exceeded our maximum size of 10k rows. This will limit the number of rows shown in the table.' A 'Show all' button is located at the bottom right of the results area.