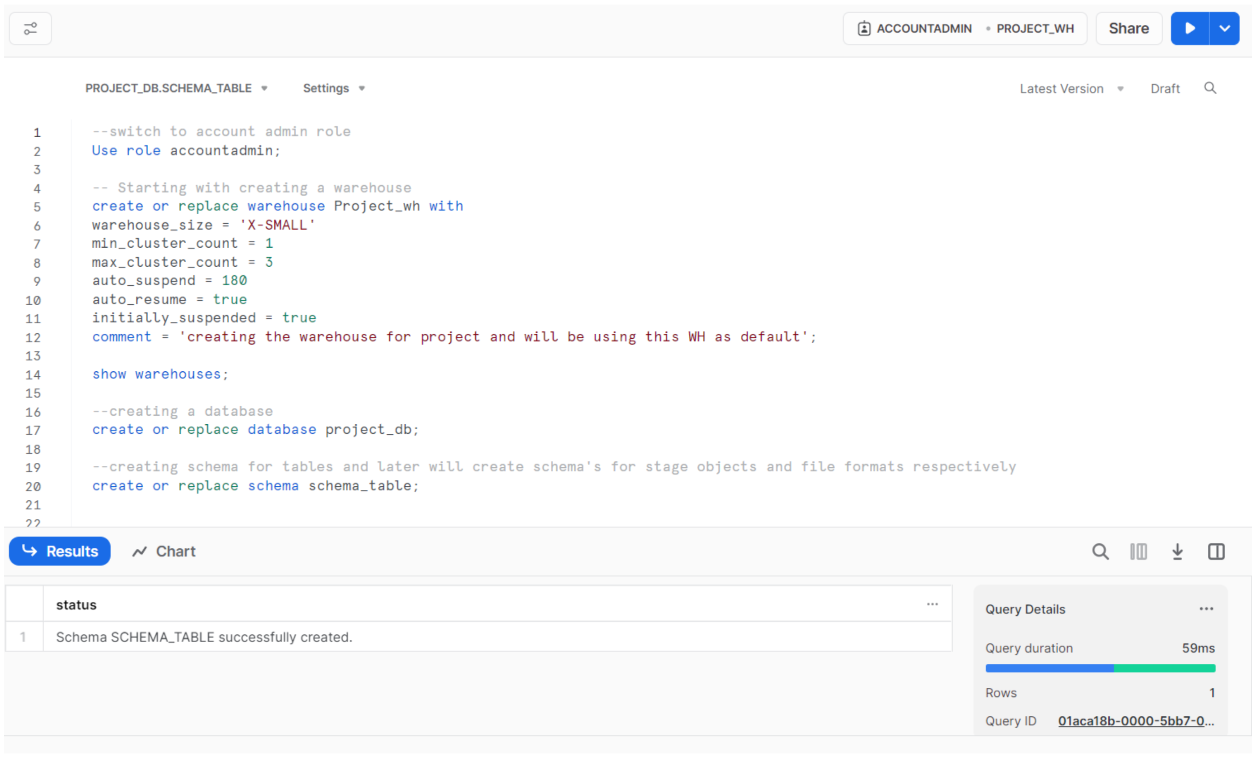
Snowflake Project

Using Snowsight,

Starting with creating warehouse, database and schema.



```
--switch to account admin role
Use role accountadmin;

-- Starting with creating a warehouse
create or replace warehouse Project_wh with
warehouse_size = 'X-SMALL'
min_cluster_count = 1
max_cluster_count = 3
auto_suspend = 180
auto_resume = true
initially_suspended = true
comment = 'creating the warehouse for project and will be using this WH as default';

show warehouses;

--creating a database
create or replace database project_db;

--creating schema for tables and later will create schema's for stage objects and file formats respectively
create or replace schema schema_table;
```

	status
1	Schema SCHEMA_TABLE successfully created.

Query Details
Query duration 59ms
Rows 1
Query ID 01aca18b-0000-5bb7-0...

//Similarly creating schema for stage objects

create or replace schema schema_stage;

//creating schema for file format objects

create or replace schema schema_file_formats;

//creating a resource monitor to notify when certain percent of credit quota is been used

create or replace resource monitor monitor_1

with credit_quota = 60

triggers on 80 percent do notify

on 95 percent do suspend;

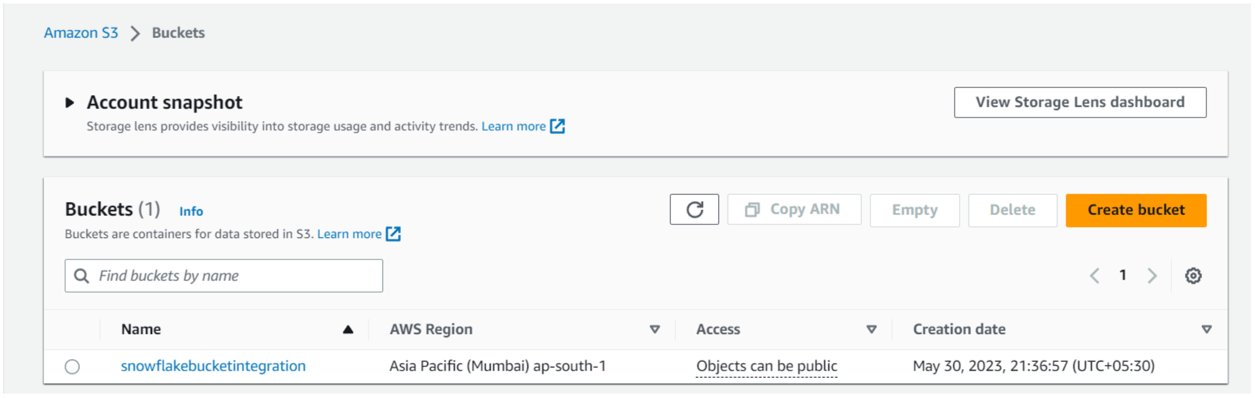
//adding the resource monitor to the warehouse

alter warehouse project_wh set resource_monitor = monitor_1;

//To check the warehouse details

show warehouses;

//Creating S3 bucket to upload CSV, JSON and Parquet data format files in three different file folders



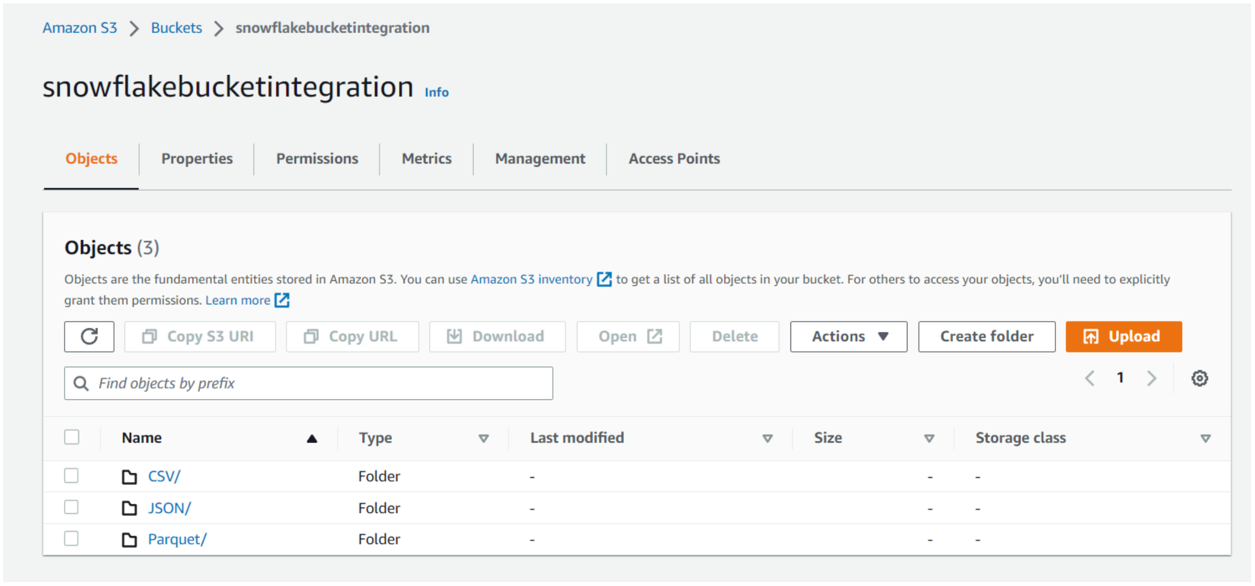
Storage lens provides visibility into storage usage and activity trends. Learn more

Buckets (1)

Buckets are containers for data stored in S3. Learn more

Name	AWS Region	Access	Creation date
snowflakebucketintegration	Asia Pacific (Mumbai) ap-south-1	Objects can be public	May 30, 2023, 21:36:57 (UTC+05:30)

//Created S3 bucket with name snowflakebucketintegration



snowflakebucketintegration

Objects (3)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
CSV/	Folder	-	-	-
JSON/	Folder	-	-	-
Parquet/	Folder	-	-	-

//Created 3 folders for CSV, JSON and Parquet data format files and uploaded the files respectively.

//Creating policy to access the services in AWS with S3 full access to the policy by creating a role with name snowflake_role_policy and assigning s3 full access policy.

//Creating integration object to connect with AWS to create a pipe that inserts data into snowflake database whenever any file uploaded into S3 bucket which we've created

create or replace storage integration AWS_Integration

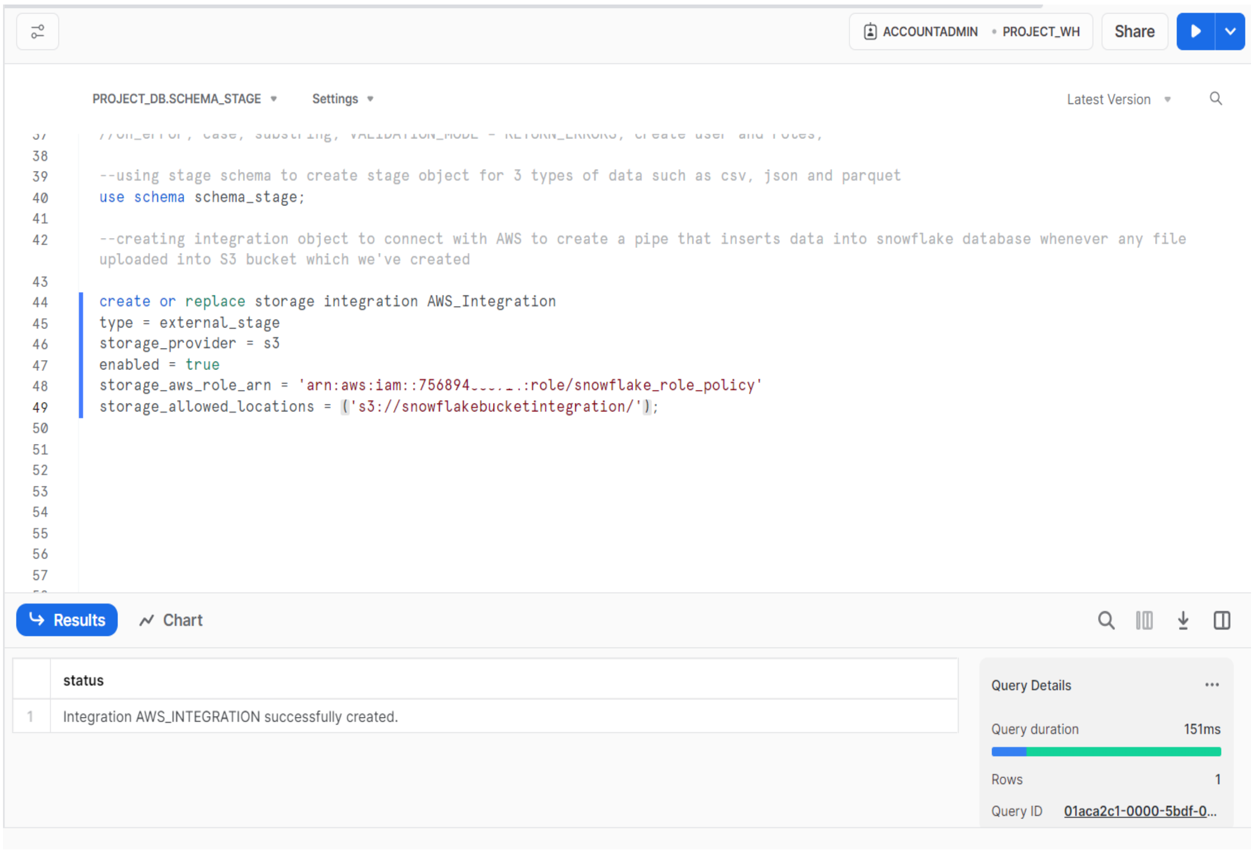
type = external_stage

storage_provider = s3

enabled = true

storage_aws_role_arn = 'arn:aws:iam::756894******:role/snowflake_role_policy'

storage_allowed_locations = ('s3://snowflakebucketintegration/');



```
//on_error, case, substring, VALIDATION_MODE = RETURN_ERRORS, create user and roles;

--using stage schema to create stage object for 3 types of data such as csv, json and parquet
use schema schema_stage;

--creating integration object to connect with AWS to create a pipe that inserts data into snowflake database whenever any file
uploaded into S3 bucket which we've created

create or replace storage integration AWS_Integration
type = external_stage
storage_provider = s3
enabled = true
storage_aws_role_arn = 'arn:aws:iam::756894[redacted]:role/snowflake_role_policy'
storage_allowed_locations = ('s3://snowflakebucketintegration/');
```

	status
1	Integration AWS_INTEGRATION successfully created.

Query Details

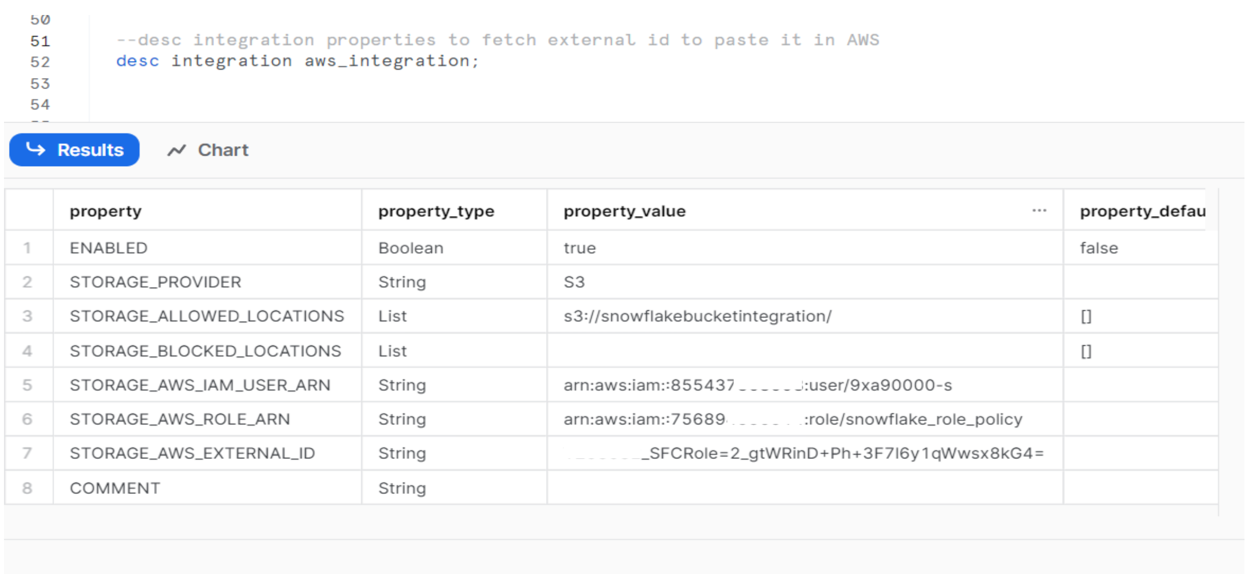
Query duration 151ms

Rows 1

Query ID 01aca2c1-0000-5bdf-0...

//Describing integration properties to fetch external id to paste it in AWS

desc integration aws_integration;



```
--desc integration properties to fetch external id to paste it in AWS
desc integration aws_integration;
```

	property	property_type	property_value	property_defau
1	ENABLED	Boolean	true	false
2	STORAGE_PROVIDER	String	S3	
3	STORAGE_ALLOWED_LOCATIONS	List	s3://snowflakebucketintegration/	[]
4	STORAGE_BLOCKED_LOCATIONS	List		[]
5	STORAGE_AWS_IAM_USER_ARN	String	arn:aws:iam::855437[redacted]:user/9xa90000-s	
6	STORAGE_AWS_ROLE_ARN	String	arn:aws:iam::75689[redacted]:role/snowflake_role_policy	
7	STORAGE_AWS_EXTERNAL_ID	String	[redacted]_SFCRole=2_gtWRinD+Ph+3F7l6y1qWwsx8kG4=	
8	COMMENT	String		

//creating file format objects.

use schema schema_file_formats;

//creating csv format file_format

```
CREATE OR REPLACE file format csv_fileformat
   type = csv

   field_delimiter = ','

   skip_header = 1

   null_if = ('NULL','null')

   empty_field_as_null = TRUE

   FIELD_OPTIONALLY_ENCLOSED_BY = '"';
```

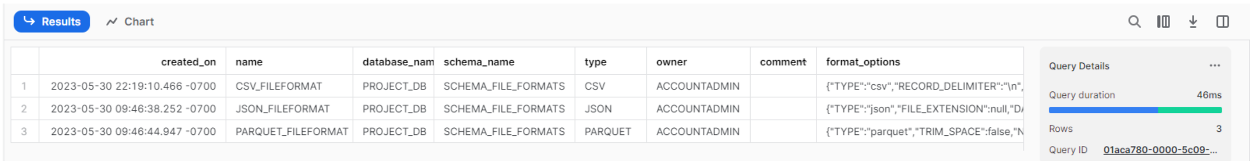
//creating json format file_format

```
CREATE OR REPLACE file format json_fileformat

type = json;
```

//creating parquet format file_format

```
CREATE OR REPLACE file format parquet_fileformat

type = parquet;
```

show file formats;



	created_on	name	database_nam	schema_name	type	owner	comment	format_options
1	2023-05-30 22:19:10.466 -0700	CSV_FILEFORMAT	PROJECT_DB	SCHEMA_FILE_FORMATS	CSV	ACCOUNTADMIN		{"TYPE":"csv","RECORD_DELIMITER":"\n",
2	2023-05-30 09:46:38.252 -0700	JSON_FILEFORMAT	PROJECT_DB	SCHEMA_FILE_FORMATS	JSON	ACCOUNTADMIN		{"TYPE":"json","FILE_EXTENSION":null,"D
3	2023-05-30 09:46:44.947 -0700	PARQUET_FILEFORMAT	PROJECT_DB	SCHEMA_FILE_FORMATS	PARQUET	ACCOUNTADMIN		{"TYPE":"parquet","TRIM_SPACE":false,"N

Query Details

Query duration 46ms

Rows 3

Query ID 01aca780-0000-5c09-...

--creating stage objects

use schema schema_stage;

--creating stage object for csv format

create or replace stage csv_stage_aws

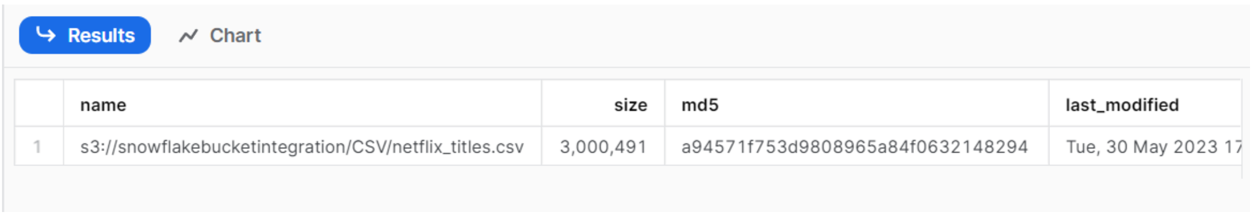
URL = 's3://snowflakebucketintegration/CSV/'

STORAGE_INTEGRATION = aws_integration

FILE_FORMAT = project_db.schema_file_formats.csv_fileformat;

//listing the files in the respective stage

list @csv_stage_aws;



	name	size	md5	last_modified
1	s3://snowflakebucketintegration/CSV/netflix_titles.csv	3,000,491	a94571f753d9808965a84f0632148294	Tue, 30 May 2023 17

//Creating stage object for json format

create or replace stage json_stage_aws

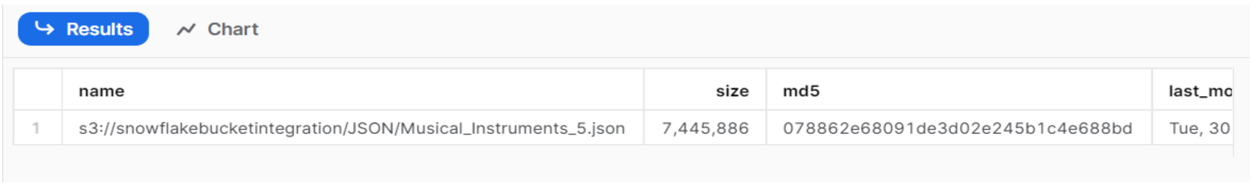
URL = 's3://snowflakebucketintegration/JSON/'

STORAGE_INTEGRATION = aws_integration

FILE_FORMAT = project_db.schema_file_formats.json_fileformat;

//listing the files in the respective stage

list @json_stage_aws;



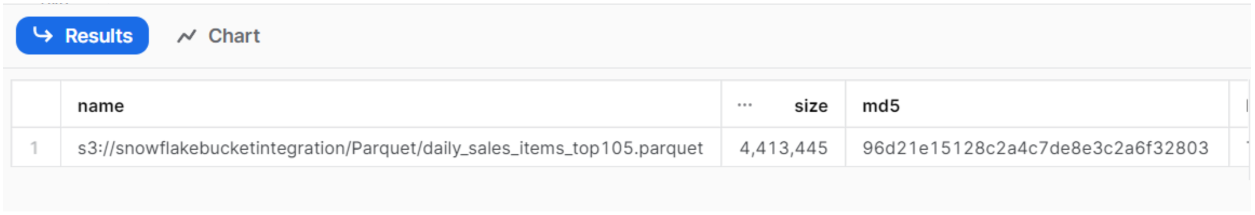
	name	size	md5	last_mo
1	s3://snowflakebucketintegration/JSON/Musical_Instruments_5.json	7,445,886	078862e68091de3d02e245b1c4e688bd	Tue, 30

//Creating stage object for parquet format

```
create or replace stage parquet_stage_aws
    URL = 's3://snowflakebucketintegration/Parquet/'
    STORAGE_INTEGRATION = aws_integration
    FILE_FORMAT = project_db.schema_file_formats.parquet_fileformat;
```

//listing the files in the respective stage

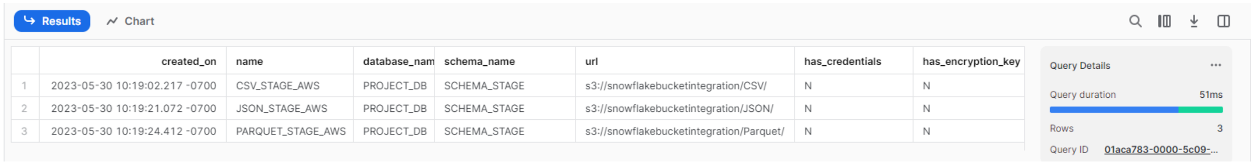
```
list @parquet_stage_aws;
```



	name	size	md5
1	s3://snowflakebucketintegration/Parquet/daily_sales_items_top105.parquet	4,413,445	96d21e15128c2a4c7de8e3c2a6f32803

//Displaying stages

```
show stages;
```



	created_on	name	database_nam	schema_name	url	has_credentials	has_encryption_key
1	2023-05-30 10:19:02.217 -0700	CSV_STAGE_AWS	PROJECT_DB	SCHEMA_STAGE	s3://snowflakebucketintegration/CSV/	N	N
2	2023-05-30 10:19:21.072 -0700	JSON_STAGE_AWS	PROJECT_DB	SCHEMA_STAGE	s3://snowflakebucketintegration/JSON/	N	N
3	2023-05-30 10:19:24.412 -0700	PARQUET_STAGE_AWS	PROJECT_DB	SCHEMA_STAGE	s3://snowflakebucketintegration/Parquet/	N	N

Query Details
Query duration 51ms
Rows 3
Query ID 01aca783-0000-5c09-...

//using table schema for creating tables

```
use schema schema_table;
```

//Creating table for csv format file

```
create or replace table Netflix_csv(
    show_id         varchar,
    type    varchar,
    title   varchar,
    director varchar,
    casts   varchar,
    country         varchar,
    date_added      varchar,
    release_year integer,
    rating  varchar,
    duration varchar,
    listed_in varchar,
    description varchar
);
```

//Creating table for json format file

```
create or replace table instruments_json(
    asin varchar,
    helpful varchar,
    overall float,
    reviewText varchar,
    reviewTime date,
    reviewerID varchar,
    reviewerName varchar,
    summary varchar,
    unixReviewTime date
);
```

//Creating table for parquet format

```
CREATE OR REPLACE TABLE sales_parquet (
    ROW_NUMBER int,
    index_level int,
    cat_id VARCHAR,
    date date,
    dept_id VARCHAR,
    id VARCHAR,
    item_id VARCHAR,
    state_id VARCHAR,
    store_id VARCHAR,
    value int,
    Load_date timestamp default TO_TIMESTAMP_NTZ(current_timestamp));
```

//Loading into respective tables

--validating if any errors while copying into table

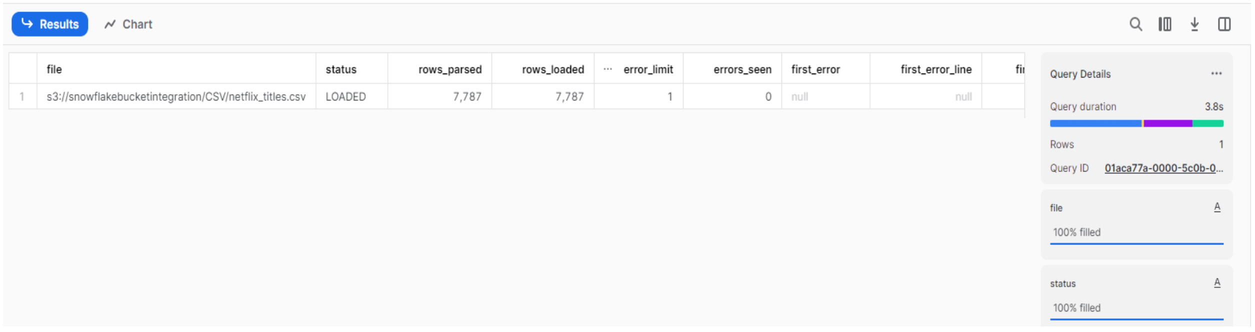
```
COPY INTO Netflix_csv
   FROM @project_db.schema_stage.csv_stage_aws
   VALIDATION_MODE = RETURN_ERRORS;
```

--validating the copy command to return 5 rows if no error occurs

```
COPY INTO Netflix_csv
   FROM @project_db.schema_stage.csv_stage_aws
   VALIDATION_MODE = RETURN_5_rows;
```

--Loading the csv format data

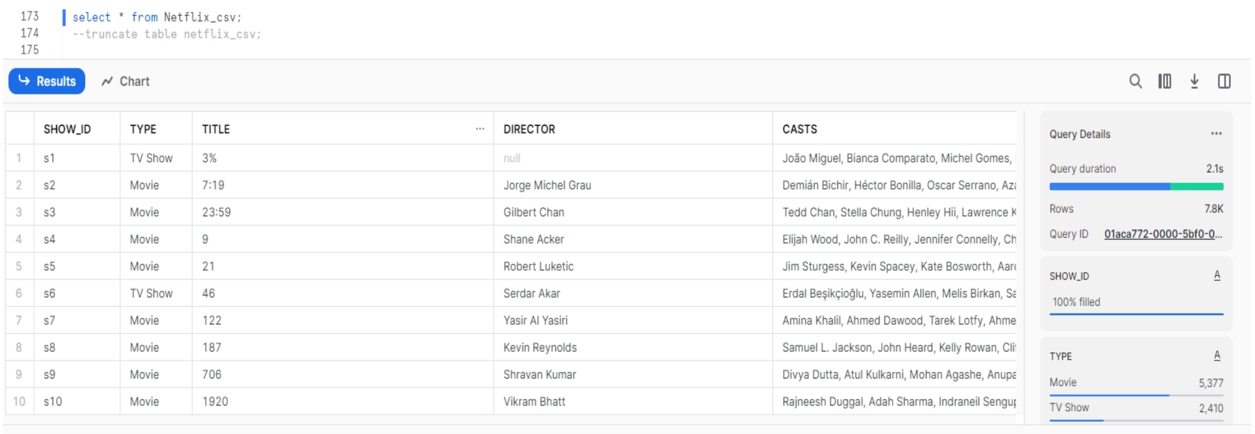
```
copy into Netflix_csv
from
@project_db.schema_stage.csv_stage_aws;
```



	file	status	rows_parsed	rows_loaded	error_limit	errors_seen	first_error	first_error_line	fi
1	s3://snowflakebucketintegration/CSV/netflix_titles.csv	LOADED	7,787	7,787	1	0	null	null	

Query Details
Query duration 3.8s
Rows 1
Query ID 01aca77a-0000-5c0b-0...
file 100% filled
status 100% filled

```
select * from Netflix_csv;
```



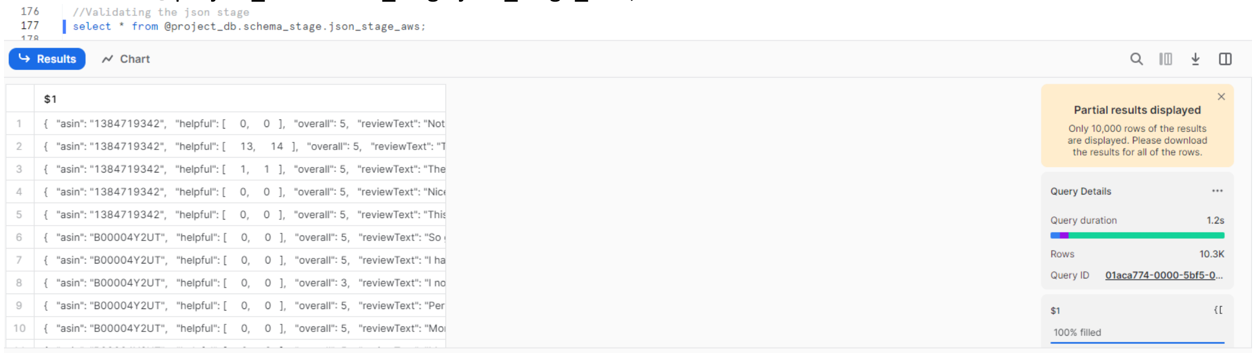
```
select * from Netflix_csv;
--truncate table netflix_csv;
```

	SHOW_ID	TYPE	TITLE	DIRECTOR	CASTS
1	s1	TV Show	3%	null	João Miguel, Bianca Comparato, Michel Gomes,
2	s2	Movie	7:19	Jorge Michel Grau	Demián Bichir, Héctor Bonilla, Oscar Serrano, Az
3	s3	Movie	23:59	Gilbert Chan	Tedd Chan, Stella Chung, Henley Hii, Lawrence K
4	s4	Movie	9	Shane Acker	Elijah Wood, John C. Reilly, Jennifer Connelly, Ch
5	s5	Movie	21	Robert Luketic	Jim Sturgess, Kevin Spacey, Kate Bosworth, Aar
6	s6	TV Show	46	Serdar Akar	Erdal Beşikçioğlu, Yasemin Allen, Melis Birkan, Sa
7	s7	Movie	122	Yasir Al Yasiri	Amina Khalil, Ahmed Dawood, Tarek Lotfy, Ahme
8	s8	Movie	187	Kevin Reynolds	Samuel L. Jackson, John Heard, Kelly Rowan, Cli
9	s9	Movie	706	Shravan Kumar	Divya Dutta, Atul Kulkarni, Mohan Agashe, Anupa
10	s10	Movie	1920	Vikram Bhatt	Rajneesh Duggal, Adah Sharma, Indraneil Sengu

Query Details
Query duration 2.1s
Rows 7.8K
Query ID 01aca772-0000-5bf0-0...
SHOW_ID 100% filled
TYPE
Movie 5,377
TV Show 2,410

//Validating the json stage

select * from @project_db.schema_stage.json_stage_aws;



```
//Validating the json stage
select * from @project_db.schema_stage.json_stage_aws;
```

```
   $1
1  {  "asin": "1384719342",  "helpful": [   0,   0  ],  "overall": 5,  "reviewText": "Not
2  {  "asin": "1384719342",  "helpful": [   13,   14  ],  "overall": 5,  "reviewText": "T
3  {  "asin": "1384719342",  "helpful": [   1,   1  ],  "overall": 5,  "reviewText": "The
4  {  "asin": "1384719342",  "helpful": [   0,   0  ],  "overall": 5,  "reviewText": "Nic
5  {  "asin": "1384719342",  "helpful": [   0,   0  ],  "overall": 5,  "reviewText": "This
6  {  "asin": "B00004Y2UT",  "helpful": [   0,   0  ],  "overall": 5,  "reviewText": "So
7  {  "asin": "B00004Y2UT",  "helpful": [   0,   0  ],  "overall": 5,  "reviewText": "I ha
8  {  "asin": "B00004Y2UT",  "helpful": [   0,   0  ],  "overall": 3,  "reviewText": "I no
9  {  "asin": "B00004Y2UT",  "helpful": [   0,   0  ],  "overall": 5,  "reviewText": "Per
10 {  "asin": "B00004Y2UT",  "helpful": [   0,   0  ],  "overall": 5,  "reviewText": "Mo
```

Partial results displayed

Only 10,000 rows of the results are displayed. Please download the results for all of the rows.

Query Details

Query duration 1.2s

Rows 10.3K

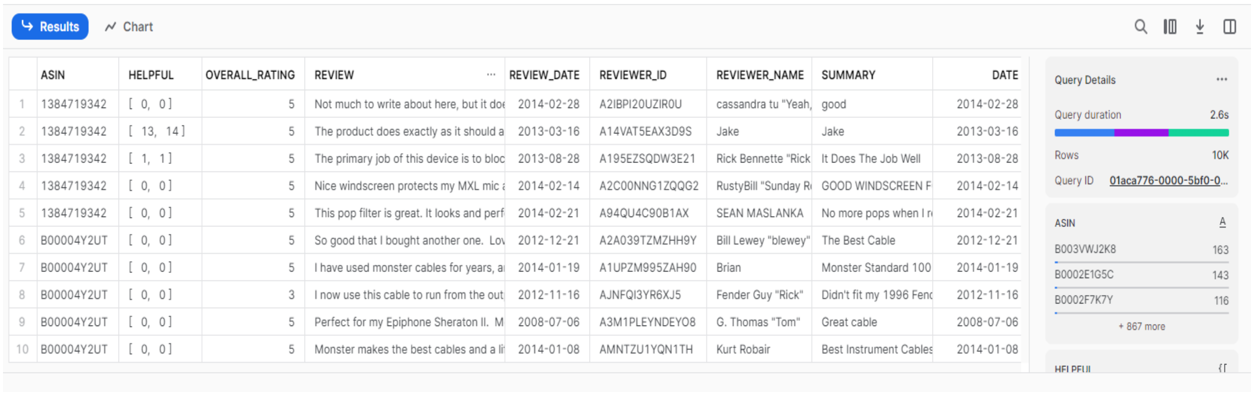
Query ID 01aca774-0000-5bf5-0...

$1

100% filled

//fetching them into individual columns

```
select $1:asin::varchar as Asin,
    $1:helpful as Helpful,
    $1:overall::integer as Overall_Rating,
    $1:reviewText::varchar as Review,
    date_from_parts(right($1:reviewTime::varchar,4), left($1:reviewTime::varchar,2),
    case when substr($1:reviewTime::varchar,5,1) = ','
        then substr($1:reviewTime::varchar,4,1)
        else substr($1:reviewTime::varchar,4,2) end) as Review_date,
    $1:reviewerID::varchar as Reviewer_ID,
    $1:reviewerName::varchar as Reviewer_Name,
    $1:summary::varchar as Summary,
    date($1:unixReviewTime::int) as date
    from @project_db.schema_stage.json_stage_aws;
```



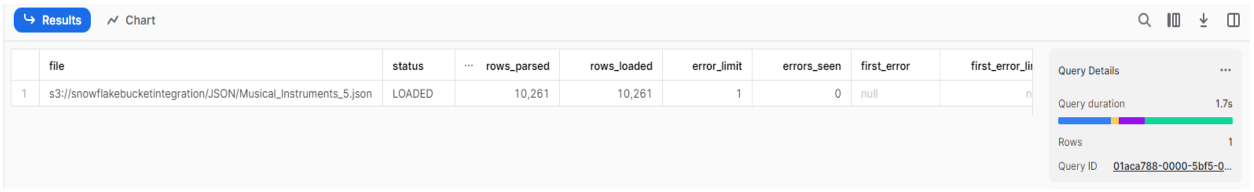
	ASIN	HELPFUL	OVERALL_RATING	REVIEW	REVIEW_DATE	REVIEWER_ID	REVIEWER_NAME	SUMMARY	DATE
1	1384719342	[0, 0]	5	Not much to write about here, but it doe	2014-02-28	A2IBPI20UZIR0U	cassandra tu "Yeah,	good	2014-02-28
2	1384719342	[13, 14]	5	The product does exactly as it should a	2013-03-16	A14VAT5EAX3D9S	Jake	Jake	2013-03-16
3	1384719342	[1, 1]	5	The primary job of this device is to bloc	2013-08-28	A195EZSQDW3E21	Rick Bennette "Rick	It Does The Job Well	2013-08-28
4	1384719342	[0, 0]	5	Nice windscreen protects my MXL mic a	2014-02-14	A2C00NNG1ZQQG2	RustyBill "Sunday Ro	GOOD WINDSCREEN F	2014-02-14
5	1384719342	[0, 0]	5	This pop filter is great. It looks and perf	2014-02-21	A94QU4C90B1AX	SEAN MASLANKA	No more pops when I r	2014-02-21
6	B00004Y2UT	[0, 0]	5	So good that I bought another one. Lov	2012-12-21	A2A039TZMZHH9Y	Bill Lewey "blewey"	The Best Cable	2012-12-21
7	B00004Y2UT	[0, 0]	5	I have used monster cables for years, a	2014-01-19	A1UPZM995ZAH90	Brian	Monster Standard 100	2014-01-19
8	B00004Y2UT	[0, 0]	3	I now use this cable to run from the out	2012-11-16	AJNFQI3YR6XJ5	Fender Guy "Rick"	Didn't fit my 1996 Fend	2012-11-16
9	B00004Y2UT	[0, 0]	5	Perfect for my Epiphone Sheraton II. M	2008-07-06	A3M1PLEYNDEYO8	G. Thomas "Tom"	Great cable	2008-07-06
10	B00004Y2UT	[0, 0]	5	Monster makes the best cables and a li	2014-01-08	AMNTZU1YQN1TH	Kurt Robair	Best Instrument Cables	2014-01-08

Query Details
Query duration 2.6s
Rows 10K
Query ID 01aca776-0000-5bf0-0...

ASIN	
B003VWJ2K8	163
B0002E1G5C	143
B0002F7K7Y	116
+ 867 more	

// Loading the json format table

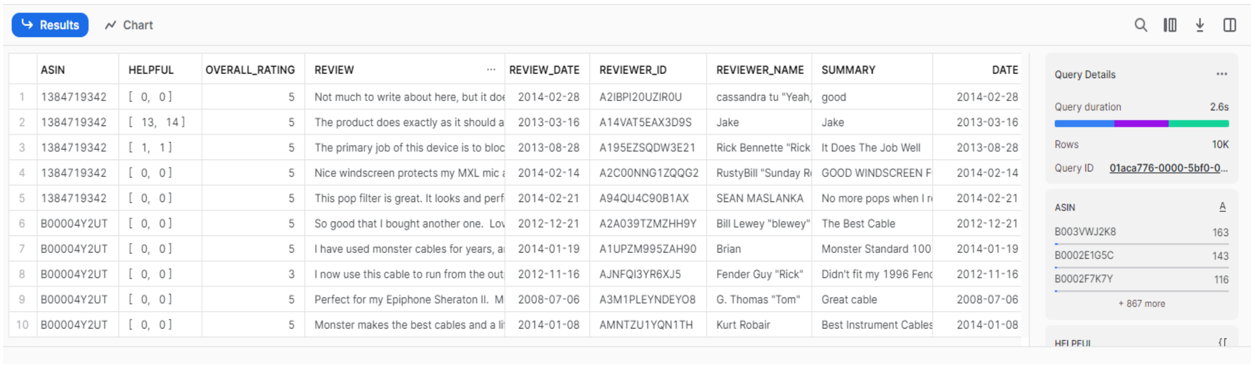
```
copy into instruments_json
from (select $1:asin::varchar as Asin,
     $1:helpful as Helpful,
     $1:overall::integer as Overall_Rating,
     $1:reviewText::varchar as Review,
     date_from_parts(right($1:reviewTime::varchar,4), left($1:reviewTime::varchar,2),
     case when substr($1:reviewTime::varchar,5,1) = ','
         then substr($1:reviewTime::varchar,4,1)
         else substr($1:reviewTime::varchar,4,2) end) as Review_date,
     $1:reviewerID::varchar as Reviewer_ID,
     $1:reviewerName::varchar as Reviewer_Name,
     $1:summary::varchar as Summary,
     date($1:unixReviewTime::int) as date
     from @project_db.schema_stage.json_stage_aws);
```



	file	status	rows_parsed	rows_loaded	error_limit	errors_seen	first_error	first_error_li
1	s3://snowflakebucketintegration/JSON/Musical_Instruments_5.json	LOADED	10,261	10,261	1	0	null	n

Query Details
Query duration 1.7s
Rows 1
Query ID 01aca788-0000-5bf5-0...

```
select * from instruments_json;
```



	ASIN	HELPFUL	OVERALL_RATING	REVIEW	REVIEW_DATE	REVIEWER_ID	REVIEWER_NAME	SUMMARY	DATE
1	1384719342	[0, 0]	5	Not much to write about here, but it doe	2014-02-28	A2IBPI20UZIR0U	cassandra tu "Yeah,	good	2014-02-28
2	1384719342	[13, 14]	5	The product does exactly as it should a	2013-03-16	A14VAT5EAX3D9S	Jake	Jake	2013-03-16
3	1384719342	[1, 1]	5	The primary job of this device is to bloc	2013-08-28	A195EZSQDW3E21	Rick Bennette "Rick	It Does The Job Well	2013-08-28
4	1384719342	[0, 0]	5	Nice windscreen protects my MXL mic a	2014-02-14	A2C00NNG1ZQQG2	RustyBill "Sunday R	GOOD WINDSCREEN F	2014-02-14
5	1384719342	[0, 0]	5	This pop filter is great. It looks and perf	2014-02-21	A94QU4C90B1AX	SEAN MASLANKA	No more pops when I r	2014-02-21
6	B00004Y2UT	[0, 0]	5	So good that I bought another one. Lov	2012-12-21	A2A039TZMZHH9Y	Bill Lewey "blewey"	The Best Cable	2012-12-21
7	B00004Y2UT	[0, 0]	5	I have used monster cables for years, a	2014-01-19	A1UPZM995ZAH90	Brian	Monster Standard 100	2014-01-19
8	B00004Y2UT	[0, 0]	3	I now use this cable to run from the out	2012-11-16	AJNFQI3YR6XJ5	Fender Guy "Rick"	Didn't fit my 1996 Fenc	2012-11-16
9	B00004Y2UT	[0, 0]	5	Perfect for my Epiphone Sheraton II. M	2008-07-06	A3M1PLEYNDEYO8	G. Thomas "Tom"	Great cable	2008-07-06
10	B00004Y2UT	[0, 0]	5	Monster makes the best cables and a li	2014-01-08	AMNTZU1YQN1TH	Kurt Robair	Best Instrument Cables	2014-01-08

Query Details

Query duration	2.6s
Rows	10K
Query ID	01aca776-0000-5bf0-0...

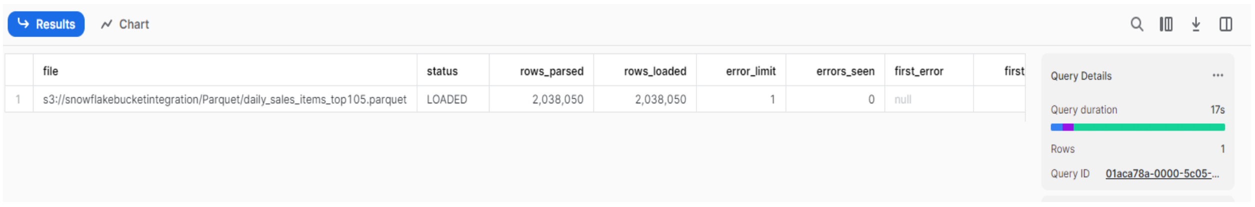
ASIN	
B003VWJ2K8	163
B0002E1G5C	143
B0002F7K7Y	116

+ 867 more

HELPFUL

// Loading the parquet format table

```
COPY INTO sales_parquet
  FROM (SELECT
        METADATA$FILE_ROW_NUMBER,
        $1:__index_level_0__::int,
        $1:cat_id::VARCHAR,
        DATE($1:date::int ),
        $1:"dept_id"::VARCHAR,
        $1:"id"::VARCHAR,
        $1:"item_id"::VARCHAR,
        $1:"state_id"::VARCHAR,
        $1:"store_id"::VARCHAR,
        $1:"value"::int,
        TO_TIMESTAMP_NTZ(current_timestamp)
    FROM @project_db.schema_stage.parquet_stage_aws);
```

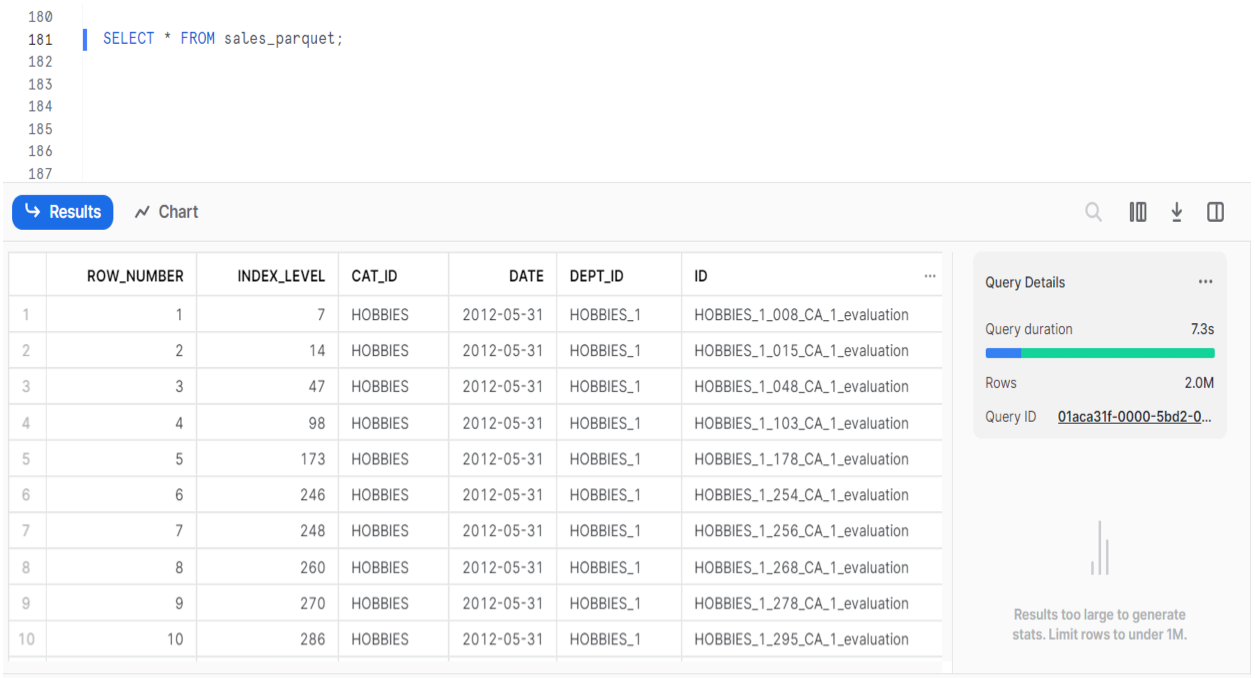


	file	status	rows_parsed	rows_loaded	error_limit	errors_seen	first_error	first
1	s3://snowflakebucketintegration/Parquet/daily_sales_items_top105.parquet	LOADED	2,038,050	2,038,050	1	0	null	

Query Details

Query duration	17s
Rows	1
Query ID	01aca78a-0000-5c05-...

SELECT * FROM sales_parquet;



```
SELECT * FROM sales_parquet;
```

Results   Chart

	ROW_NUMBER	INDEX_LEVEL	CAT_ID	DATE	DEPT_ID	ID
1	1	7	HOBBIES	2012-05-31	HOBBIES_1	HOBBIES_1_008_CA_1_evaluation
2	2	14	HOBBIES	2012-05-31	HOBBIES_1	HOBBIES_1_015_CA_1_evaluation
3	3	47	HOBBIES	2012-05-31	HOBBIES_1	HOBBIES_1_048_CA_1_evaluation
4	4	98	HOBBIES	2012-05-31	HOBBIES_1	HOBBIES_1_103_CA_1_evaluation
5	5	173	HOBBIES	2012-05-31	HOBBIES_1	HOBBIES_1_178_CA_1_evaluation
6	6	246	HOBBIES	2012-05-31	HOBBIES_1	HOBBIES_1_254_CA_1_evaluation
7	7	248	HOBBIES	2012-05-31	HOBBIES_1	HOBBIES_1_256_CA_1_evaluation
8	8	260	HOBBIES	2012-05-31	HOBBIES_1	HOBBIES_1_268_CA_1_evaluation
9	9	270	HOBBIES	2012-05-31	HOBBIES_1	HOBBIES_1_278_CA_1_evaluation
10	10	286	HOBBIES	2012-05-31	HOBBIES_1	HOBBIES_1_295_CA_1_evaluation

Query Details

Query duration 7.3s

Rows 2.0M

Query ID 01aca31f-0000-5bd2-0...

Results too large to generate stats. Limit rows to under 1M.

//Creating Snowpipe to auto insert the data into snowflake database whenever a file gets uploaded in S3 bucket.

//Creating table in snowflake to ingest the data

CREATE OR REPLACE TABLE employees_pipe (

id INT,

first_name STRING,

last_name STRING,

email STRING,

location STRING,

department STRING

);

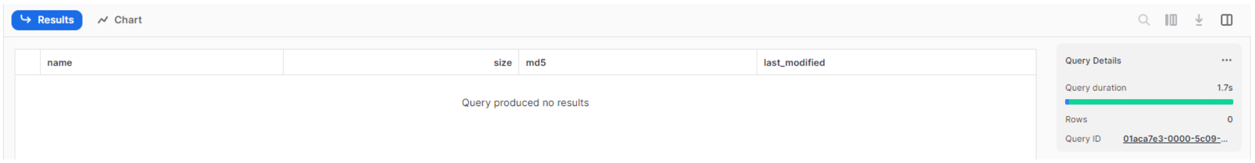
// Create file format object

use schema schema_file_formats;

```
CREATE OR REPLACE file format csv_fileformat_pipe
    type = csv
    field_delimiter = ','
    skip_header = 1
    null_if = ('NULL','null')
    empty_field_as_null = TRUE;
```

// Create stage object with integration object & file format object

```
use schema schema_stage;

CREATE OR REPLACE stage stage_pipe
    URL = 's3://snowflakebucketintegration/CSV/Snowpipe/'
    STORAGE_INTEGRATION = aws_integration
    FILE_FORMAT = project_db.SCHEMA_FILE_FORMATS.csv_fileformat_pipe;

list@stage_pipe;
```



name	size	md5	last_modified

Query produced no results

Query Details
Query duration 1.7s
Rows 0
Query ID 01aca7e3-0000-5c09-...

//the result contains zero rows because we didn’t upload any file in S3 bucket yet

//Creating a schema for pipes

```
create or replace schema schema_pipe;
```

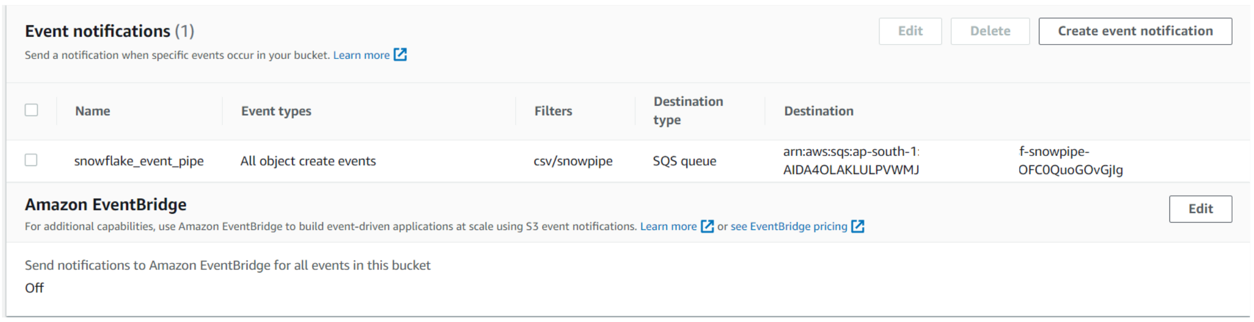
//Creating pipe to auto ingest the data into snowflake database whenever a file gets uploaded in S3 bucket

```
Create or replace pipe Snowflake_pipe
auto_ingest = true
as
copy into project_db.schema_table.employees_pipe
from @project_db.schema_stage.stage_pipe;
```

//describing the pipe to fetch notification channel

```
desc pipe Snowflake_pipe;
```

//We then copy the notification_channel in the SQS Queue ARN by configuring the event notification in the S3 bucket properties.



Event notifications (1)

Send a notification when specific events occur in your bucket. Learn more

	Name	Event types	Filters	Destination type	Destination
	snowflake_event_pipe	All object create events	csv/snowpipe	SQS queue	arn:aws:sqs:ap-south-1: AIDA4OLAKLULPVWMJ f-snowpipe-OFC0QuoGOvGjlg

Amazon EventBridge

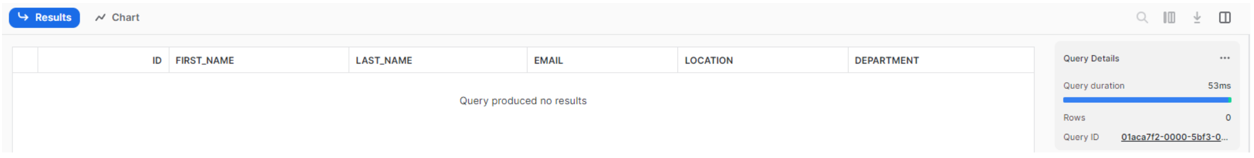
For additional capabilities, use Amazon EventBridge to build event-driven applications at scale using S3 event notifications. Learn more or see EventBridge pricing

Send notifications to Amazon EventBridge for all events in this bucket

Off

//validating the table before insertion of files in S3 bucket

select * from employees_pipe;



	ID	FIRST_NAME	LAST_NAME	EMAIL	LOCATION	DEPARTMENT

Query produced no results

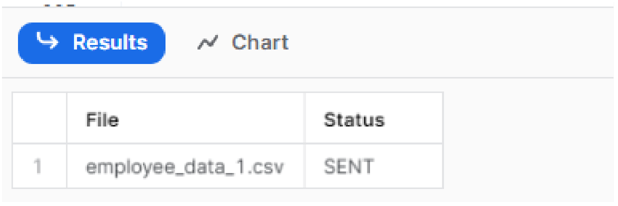
Query Details

Query duration	53ms
Rows	0
Query ID	01aca7f2-0000-5bf3-0...

//Now inserting one csv file with the name employee_data_1

//checking the status of pipe

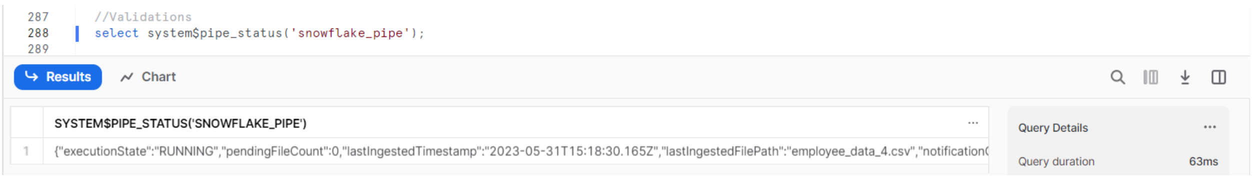
alter pipe project_db.schema_pipe.snowflake_pipe refresh;



	File	Status
1	employee_data_1.csv	SENT

//Validations

select system$pipe_status('snowflake_pipe');



```
//Validations
select system$pipe_status('snowflake_pipe');
```

	SYSTEM$PIPE_STATUS('SNOWFLAKE_PIPE')
1	{"executionState":"RUNNING","pendingFileCount":0,"lastIngestedTimestamp":"2023-05-31T15:18:30.165Z","lastIngestedFilePath":"employee_data_4.csv","notification(

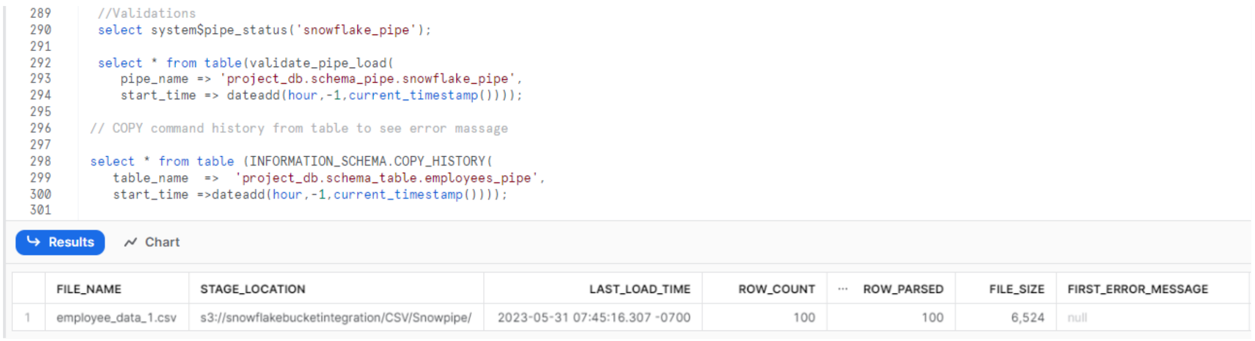
Query Details

Query duration	63ms

```
  select * from table(validate_pipe_load(
  pipe_name => 'project_db.schema_pipe.snowflake_pipe',
  start_time => dateadd(hour,-1,current_timestamp())));
```

// COPY command history from table to see error massage

```
select * from table (INFORMATION_SCHEMA.COPY_HISTORY(
  table_name  =>  'project_db.schema_table.employees_pipe',
  start_time =>dateadd(hour,-1,current_timestamp())));
```

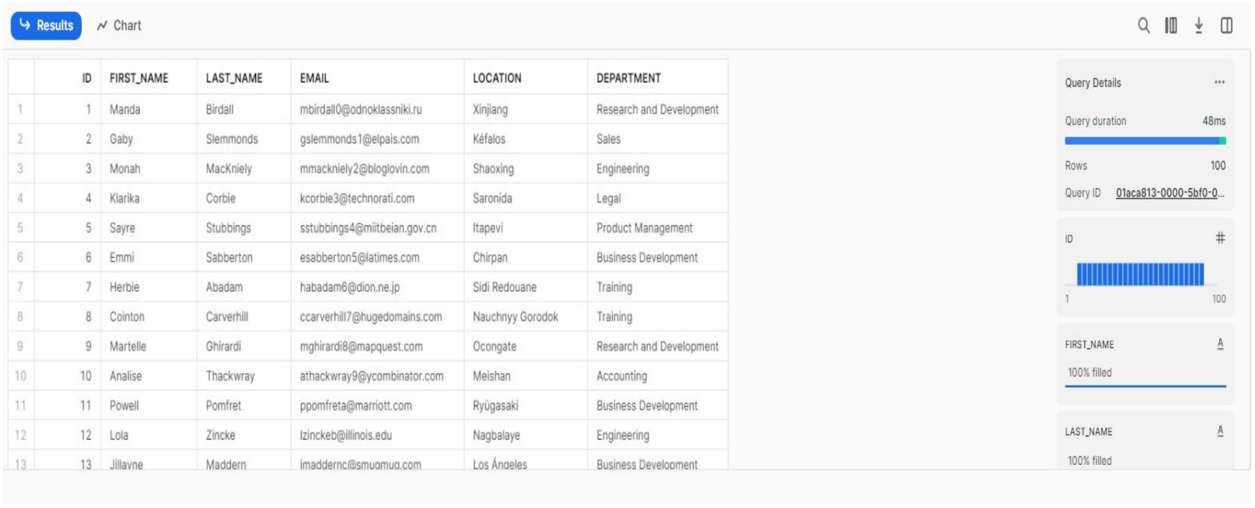


```
 //Validations
 select system$pipe_status('snowflake_pipe');

 select * from table(validate_pipe_load(
    pipe_name => 'project_db.schema_pipe.snowflake_pipe',
    start_time => dateadd(hour,-1,current_timestamp())));

// COPY command history from table to see error massage

select * from table (INFORMATION_SCHEMA.COPY_HISTORY(
   table_name  =>  'project_db.schema_table.employees_pipe',
   start_time =>dateadd(hour,-1,current_timestamp())));
```

	FILE_NAME	STAGE_LOCATION	LAST_LOAD_TIME	ROW_COUNT	···	ROW_PARSED	FILE_SIZE	FIRST_ERROR_MESSAGE
1	employee_data_1.csv	s3://snowflakebucketintegration/CSV/Snowpipe/	2023-05-31 07:45:16.307 -0700	100		100	6,524	null

//after insertion of csv file

```
select * from employees_pipe;
```

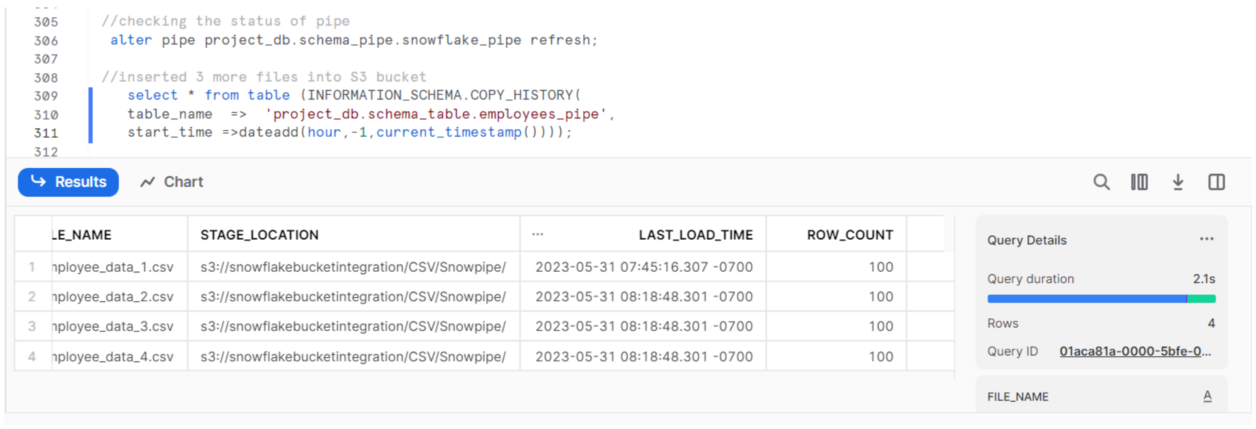


	ID	FIRST_NAME	LAST_NAME	EMAIL	LOCATION	DEPARTMENT
1	1	Manda	Birdall	mbirdall0@odnoklassniki.ru	Xinjiang	Research and Development
2	2	Gaby	Slemmonds	gslemmonds1@elpais.com	Kéfalos	Sales
3	3	Monah	MacKniely	mmackniely2@bloglovin.com	Shaoxing	Engineering
4	4	Klarika	Corbie	kcorbie3@technorati.com	Saronída	Legal
5	5	Sayre	Stubbings	sstubbings4@miitbeian.gov.cn	Itapevi	Product Management
6	6	Emmi	Sabberton	esabberton5@latimes.com	Chirpan	Business Development
7	7	Herbie	Abadam	habadam6@dion.ne.jp	Sidi Redouane	Training
8	8	Cointon	Carverhill	ccarverhill7@hugedomains.com	Nauchnyy Gorodok	Training
9	9	Martelle	Ghirardi	mghirardi8@mapquest.com	Ocongate	Research and Development
10	10	Analise	Thackwray	athackwray9@ycombinator.com	Meishan	Accounting
11	11	Powell	Pomfret	ppomfreta@marriott.com	Ryūgasaki	Business Development
12	12	Lola	Zincke	lzinckeb@illinois.edu	Nagbalaye	Engineering
13	13	Jillayne	Maddern	jmadderno@smugmug.com	Los Ángeles	Business Development

Query Details
Query duration 48ms
Rows 100
Query ID 01aca813-0000-5bf0-0...

//Successfully loaded 100 rows

//After insertion of 3 more files into S3 bucket



```
//checking the status of pipe
 alter pipe project_db.schema_pipe.snowflake_pipe refresh;

//inserted 3 more files into S3 bucket
   select * from table (INFORMATION_SCHEMA.COPY_HISTORY(
   table_name  =>  'project_db.schema_table.employees_pipe',
   start_time =>dateadd(hour,-1,current_timestamp())));
```

Results Chart

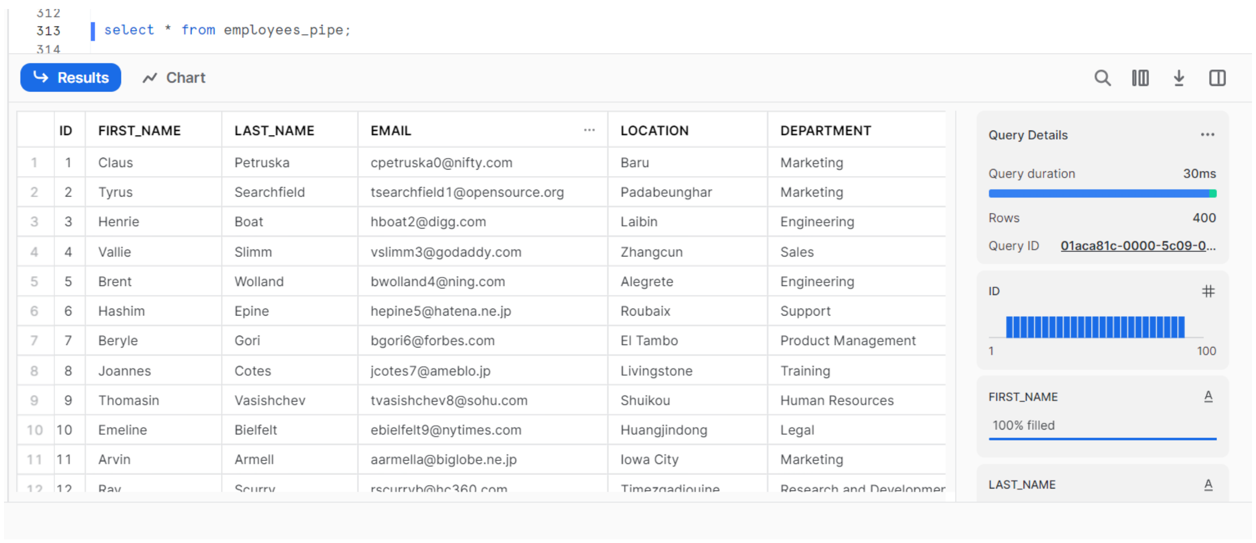
	LE_NAME	STAGE_LOCATION	LAST_LOAD_TIME	ROW_COUNT
1	nployee_data_1.csv	s3://snowflakebucketintegration/CSV/Snowpipe/	2023-05-31 07:45:16.307 -0700	100
2	nployee_data_2.csv	s3://snowflakebucketintegration/CSV/Snowpipe/	2023-05-31 08:18:48.301 -0700	100
3	nployee_data_3.csv	s3://snowflakebucketintegration/CSV/Snowpipe/	2023-05-31 08:18:48.301 -0700	100
4	nployee_data_4.csv	s3://snowflakebucketintegration/CSV/Snowpipe/	2023-05-31 08:18:48.301 -0700	100

Query Details
Query duration 2.1s
Rows 4
Query ID 01aca81a-0000-5bfe-0...
FILE_NAME

//Other 3 files have been captured by the pipe

// Querying the data to check

select * from employees_pipe;



```
select * from employees_pipe;
```

Results Chart

	ID	FIRST_NAME	LAST_NAME	EMAIL	LOCATION	DEPARTMENT
1	1	Claus	Petruska	cpetruska0@nifty.com	Baru	Marketing
2	2	Tyrus	Searchfield	tsearchfield1@opensource.org	Padabeunghar	Marketing
3	3	Henrie	Boat	hboat2@digg.com	Laibin	Engineering
4	4	Vallie	Slimm	vslimm3@godaddy.com	Zhangcun	Sales
5	5	Brent	Wolland	bwolland4@ning.com	Alegrete	Engineering
6	6	Hashim	Epine	hepine5@hatena.ne.jp	Roubaix	Support
7	7	Beryle	Gori	bgori6@forbes.com	El Tambo	Product Management
8	8	Joannes	Cotes	jcotes7@ameblo.jp	Livingstone	Training
9	9	Thomasin	Vasishchev	tvasishchev8@sohu.com	Shuikou	Human Resources
10	10	Emeline	Bielfelt	ebielfelt9@nytimes.com	Huangjindong	Legal
11	11	Arvin	Armell	aarmella@biglobe.ne.jp	Iowa City	Marketing
12	12	Ray	Scurry	rscurryb@hc360.com	Timezgadiouine	Research and Developmer

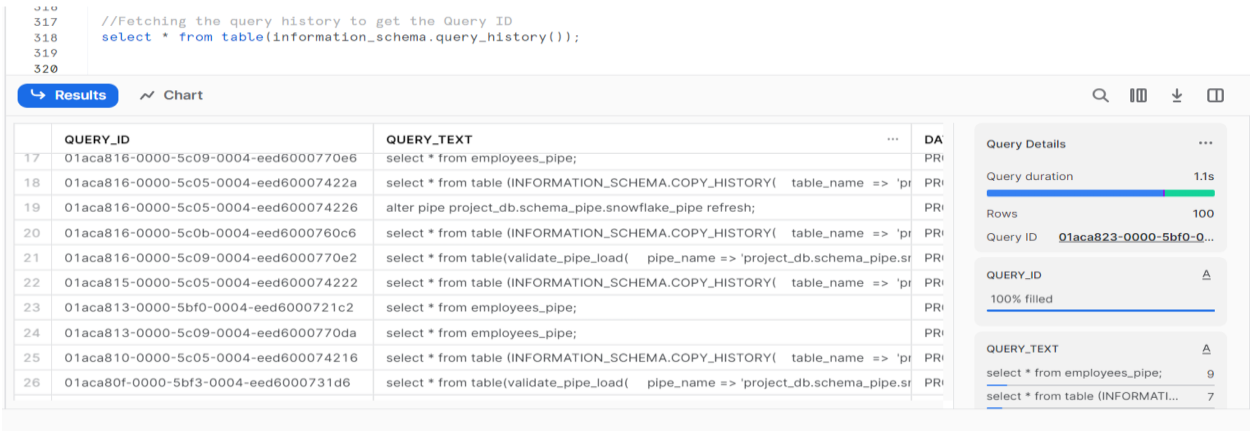
Query Details
Query duration 30ms
Rows 400
Query ID 01aca81c-0000-5c09-0...
ID
1 100
FIRST_NAME
100% filled
LAST_NAME

//Fetched 400 rows. Successfully loaded all 4 files into snowflake database automatically by using Snowpipe

//To work with Time Travel, checking the above table employee_pipe before insertion of 3 files

//Fetching the query history to get the Query ID

select * from table(information_schema.query_history());

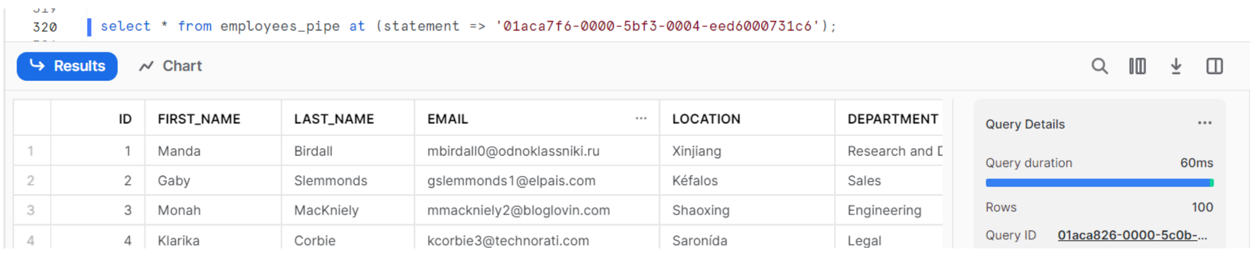


```
//Fetching the query history to get the Query ID
select * from table(information_schema.query_history());
```

	QUERY_ID	QUERY_TEXT	DA
17	01aca816-0000-5c09-0004-eed6000770e6	select * from employees_pipe;	PR
18	01aca816-0000-5c05-0004-eed60007422a	select * from table (INFORMATION_SCHEMA.COPY_HISTORY(table_name => 'pr	PR
19	01aca816-0000-5c05-0004-eed600074226	alter pipe project_db.schema_pipe.snowflake_pipe refresh;	PR
20	01aca816-0000-5c0b-0004-eed6000760c6	select * from table (INFORMATION_SCHEMA.COPY_HISTORY(table_name => 'pr	PR
21	01aca816-0000-5c09-0004-eed6000770e2	select * from table(validate_pipe_load(pipe_name => 'project_db.schema_pipe.sr	PR
22	01aca815-0000-5c05-0004-eed600074222	select * from table (INFORMATION_SCHEMA.COPY_HISTORY(table_name => 'pr	PR
23	01aca813-0000-5bf0-0004-eed6000721c2	select * from employees_pipe;	PR
24	01aca813-0000-5c09-0004-eed6000770da	select * from employees_pipe;	PR
25	01aca810-0000-5c05-0004-eed600074216	select * from table (INFORMATION_SCHEMA.COPY_HISTORY(table_name => 'pr	PR
26	01aca80f-0000-5bf3-0004-eed6000731d6	select * from table(validate_pipe_load(pipe_name => 'project_db.schema_pipe.sr	PR

Query Details
Query duration 1.1s
Rows 100
Query ID 01aca823-0000-5bf0-0...
QUERY_ID
100% filled
QUERY_TEXT
select * from employees_pipe; 9
select * from table (INFORMATI... 7

select * from employees_pipe at (statement => '01aca7f6-0000-5bf3-0004-eed6000731c6');



```
select * from employees_pipe at (statement => '01aca7f6-0000-5bf3-0004-eed6000731c6');
```

	ID	FIRST_NAME	LAST_NAME	EMAIL	LOCATION	DEPARTMENT
1	1	Manda	Birdall	mbirdall0@odnoklassniki.ru	Xinjiang	Research and D
2	2	Gaby	Slemmonds	gslemmonds1@elpais.com	Kéfalos	Sales
3	3	Monah	MacKniely	mmackniely2@bloglovin.com	Shaoxing	Engineering
4	4	Klarika	Corbie	kcorbie3@technorati.com	Saronída	Legal

Query Details
Query duration 60ms
Rows 100
Query ID 01aca826-0000-5c0b-...

//As we see from above image, using time travel, we can fetch the table on how it looks few queries before or some time before any operations performed on the table. From above image the table consists of 100 rows before 3 other files with 100 rows each inserted.

//Working with streams to capture change which is also called as Change Data Capture (CDC)

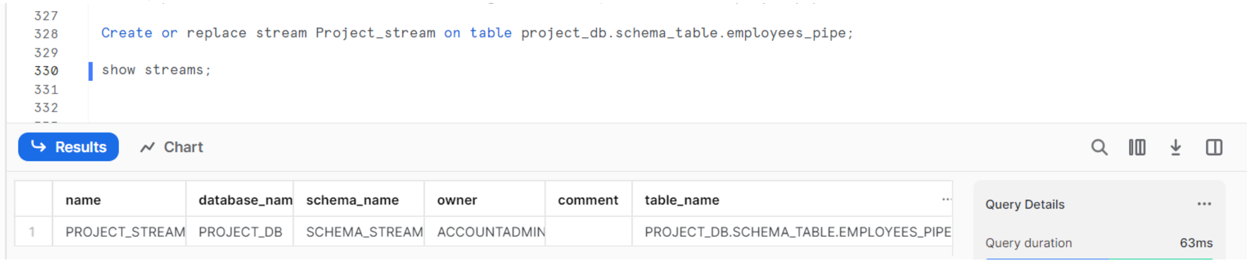
//Creating schema for stream object

create or replace schema schema_stream;

//TO capture the inserted data, we are creating a stream object on table employee_pipe

Create or replace stream Project_stream on table project_db.schema_table.employees_pipe;

show streams;



```
Create or replace stream Project_stream on table project_db.schema_table.employees_pipe;

show streams;
```

	name	database_nam	schema_name	owner	comment	table_name
1	PROJECT_STREAM	PROJECT_DB	SCHEMA_STREAM	ACCOUNTADMIN		PROJECT_DB.SCHEMA_TABLE.EMPLOYEES_PIPE

Query Details
Query duration 63ms

//Validating the stream by inserting 2 rows

insert into project_db.schema_table.employees_pipe

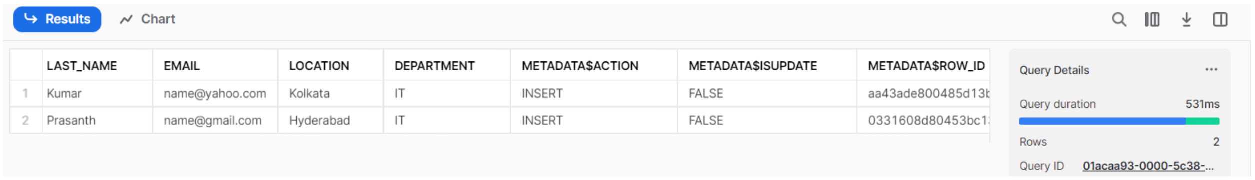
values

(401,'Bharani','Prasanth','name@gmail.com','Hyderabad', 'IT'),

(402,'SHiva','Kumar','name@yahoo.com','Kolkata','IT');

//Inserted 2 rows

Select * from project_stream;



	LAST_NAME	EMAIL	LOCATION	DEPARTMENT	METADATA$ACTION	METADATA$ISUPDATE	METADATA$ROW_ID
1	Kumar	name@yahoo.com	Kolkata	IT	INSERT	FALSE	aa43ade800485d13b
2	Prasanth	name@gmail.com	Hyderabad	IT	INSERT	FALSE	0331608d80453bc1

Query Details
Query duration 531ms
Rows 2
Query ID 01acaa93-0000-5c38-...

//We can see the Metadata$action as insert and Metadata$update as false because the data has been inserted.

//If we consume the data the stream object will become empty. Consuming the data meant to be, for example, when a stream object is used in join operation with other table, the data will be consumed by the final table.

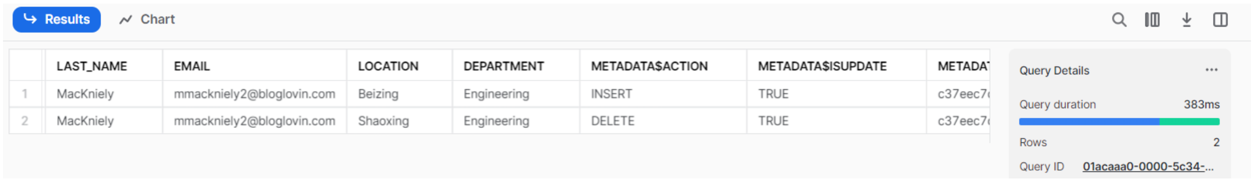
//Validating stream object by updating 1 row

update project_db.schema_table.employees_pipe

set location = 'Beizing'

where first_name = 'Monah' and last_name = 'MacKniely';

Select * from project_stream;



	LAST_NAME	EMAIL	LOCATION	DEPARTMENT	METADATA$ACTION	METADATA$ISUPDATE	METADAT
1	MacKniely	mmackniely2@bloglovin.com	Beizing	Engineering	INSERT	TRUE	c37eec7
2	MacKniely	mmackniely2@bloglovin.com	Shaoxing	Engineering	DELETE	TRUE	c37eec7

Query Details
Query duration 383ms
Rows 2
Query ID 01acaaa0-0000-5c34-...

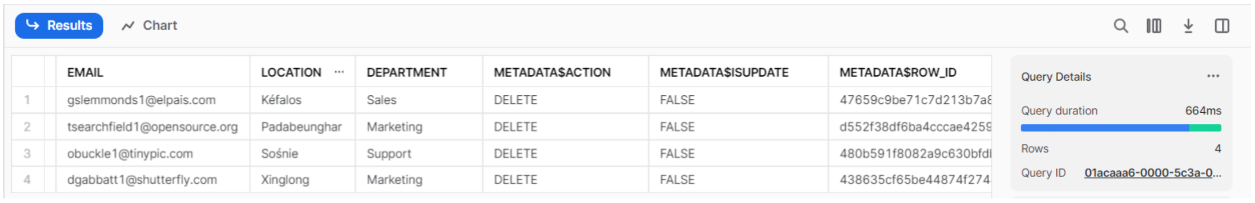
//As we can see here two rows were being reflected although we had just updated 1 row. This is because whenever a row gets updated it undergoes two operations, beginning with deletion and ending with insertion into the table. So, in this stream object we can see Metadata$action as delete and insert and with Metadata$update as true.

//Validating stream object by deleting rows

delete from project_db.schema_table.employees_pipe

where id = 2;

select * from project_stream;



	EMAIL	LOCATION	DEPARTMENT	METADATA$ACTION	METADATA$ISUPDATE	METADATA$ROW_ID
1	gslemmonds1@elpais.com	Kéfalos	Sales	DELETE	FALSE	47659c9be71c7d213b7a8
2	tsearchfield1@opensource.org	Padabeunghar	Marketing	DELETE	FALSE	d552f38df6ba4cccae4259
3	obuckle1@tinypic.com	Sośnie	Support	DELETE	FALSE	480b591f8082a9c630bfdl
4	dgabbatt1@shutterfly.com	Xinglong	Marketing	DELETE	FALSE	438635cf65be44874f274

Query Details
Query duration 664ms
Rows 4
Query ID 01acaaa6-0000-5c3a-0...

//Here 4 rows have been deleted where id = 2 and we can see the Metadata$action as delete with Metadata$update as false. We can also see that for every insert, update and delete of data from table, the metadata$row_id as different

//Creating a reader account for non-Snowflake user

//To create a share to non_snowflake user we need to create shared account for them and for that we need account admin role

use role accountadmin;

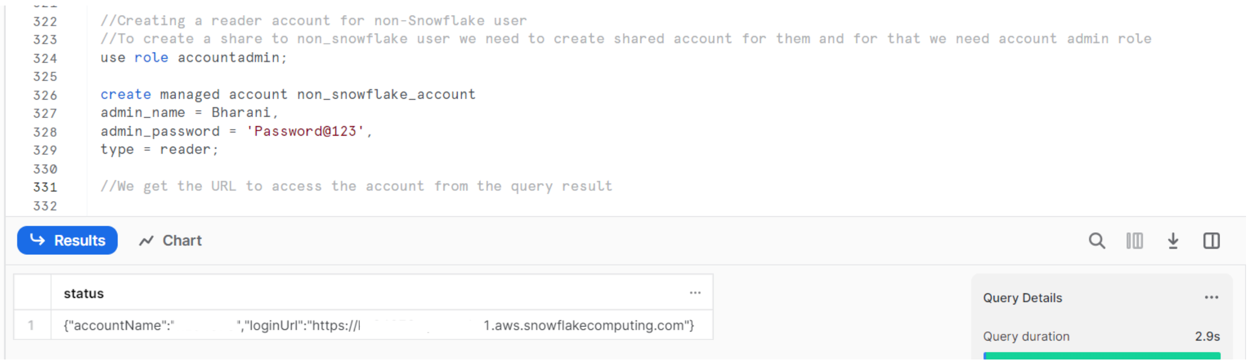
create managed account non_snowflake_account

admin_name = Bharani,

admin_password = 'Password@123',

type = reader;

//We get the URL to access the account from the query result



```
//Creating a reader account for non-Snowflake user
//To create a share to non_snowflake user we need to create shared account for them and for that we need account admin role
use role accountadmin;

create managed account non_snowflake_account
admin_name = Bharani,
admin_password = 'Password@123',
type = reader;

//We get the URL to access the account from the query result

```

	status
1	{"accountName":" ","loginUrl":"https://l 1.aws.snowflakecomputing.com"}

Query Details
Query duration 2.9s

//we can also get the URL from below query

show managed accounts;

//Creating share object to add the managed account we just created

create or replace share Project_share;

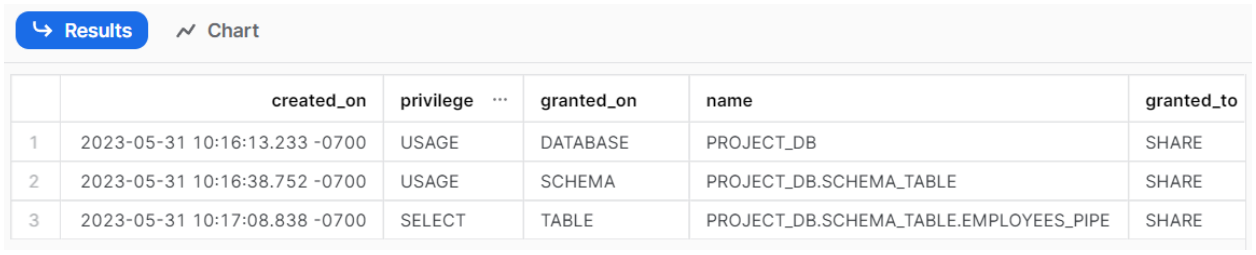
//Granting usage and select on database, schema and table respectively to share object

Grant usage on database project_db to share project_share;

Grant usage on schema schema_table to share project_share;

grant select on table employees_pipe to share project_share;

show grants to share project_share;



	created_on	privilege	granted_on	name	granted_to
1	2023-05-31 10:16:13.233 -0700	USAGE	DATABASE	PROJECT_DB	SHARE
2	2023-05-31 10:16:38.752 -0700	USAGE	SCHEMA	PROJECT_DB.SCHEMA_TABLE	SHARE
3	2023-05-31 10:17:08.838 -0700	SELECT	TABLE	PROJECT_DB.SCHEMA_TABLE.EMPLOYEES_PIPE	SHARE

//Now login to that reader account from the URL generated when created the managed account.

//We need to use account admin role to view the shares. To view shares from reader account, run the following query and to get the appropriate share name

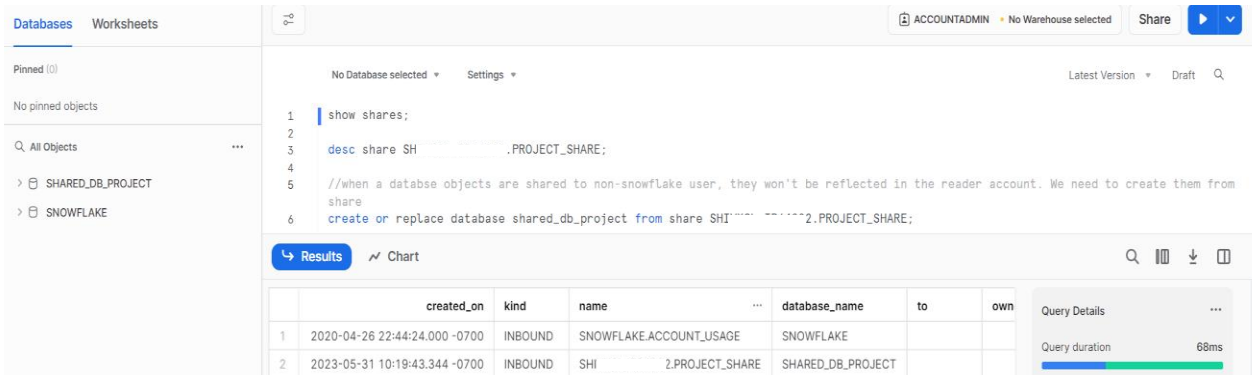
Show shares;

//Describing share to see what grants are provided on this share

desc share SH***CL.IR6****.PROJECT_SHARE;

//when a databse objects are shared to non-snowflake user, they won't be reflected in the reader account. We need to create them from share

create or replace database shared_db_project from share SH***CL.IR6****.PROJECT_SHARE;



```
show shares;

desc share SH[redacted].PROJECT_SHARE;

//when a databse objects are shared to non-snowflake user, they won't be reflected in the reader account. We need to create them from share
create or replace database shared_db_project from share SHI[redacted]2.PROJECT_SHARE;
```

	created_on	kind	name	database_name	to	own
1	2020-04-26 22:44:24.000 -0700	INBOUND	SNOWFLAKE.ACCOUNT_USAGE	SNOWFLAKE		
2	2023-05-31 10:19:43.344 -0700	INBOUND	SHI[redacted]2.PROJECT_SHARE	SHARED_DB_PROJECT		

//Need to create warehouse to query the shared table

create or replace warehouse shared_wh with

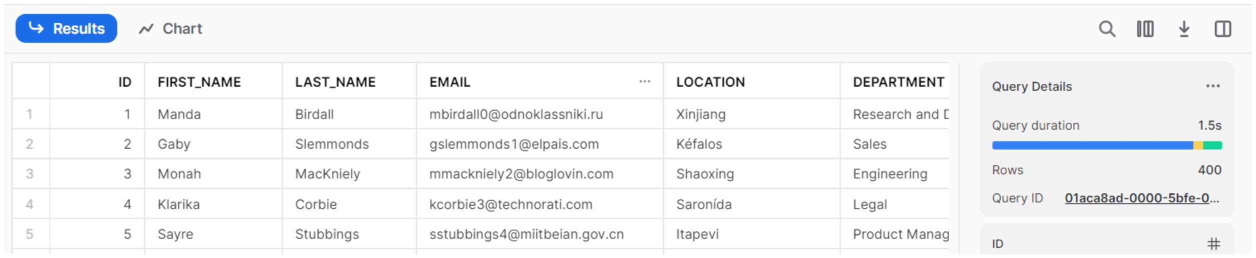
warehouse_size = 'X-SMALL'

auto_suspend = 180

auto_resume = true

initially_suspended = true;

select * from employees_pipe;



	ID	FIRST_NAME	LAST_NAME	EMAIL	LOCATION	DEPARTMENT
1	1	Manda	Birdall	mbirdall0@odnoklassniki.ru	Xinjiang	Research and D
2	2	Gaby	Slemmonds	gslemmonds1@elpais.com	Kéfalos	Sales
3	3	Monah	MacKniely	mmackniely2@bloglovin.com	Shaoxing	Engineering
4	4	Klarika	Corbie	kcorbie3@technorati.com	Saronída	Legal
5	5	Sayre	Stubbings	sstubbings4@miitbeian.gov.cn	Itapevi	Product Manag

Query Details

Query duration 1.5s

Rows 400

Query ID 01aca8ad-0000-5bfe-0...

ID

/*Concepts/Utilities used in this Project

Loading/Copying structured and unstructured data from AWS S3 bucket,

Snowpipe - Automating ingestion of data,

Time Travel,

Streams - Change Data Capture (CDC),

Data Sharing

*/