

Snowflake Loading and Unloading Assignment

1. I have downloaded a sample country.csv file and Created a table called country and mentioned the field names and datatypes according to the csv file.
using Load wizard, I have loaded the data into the country table

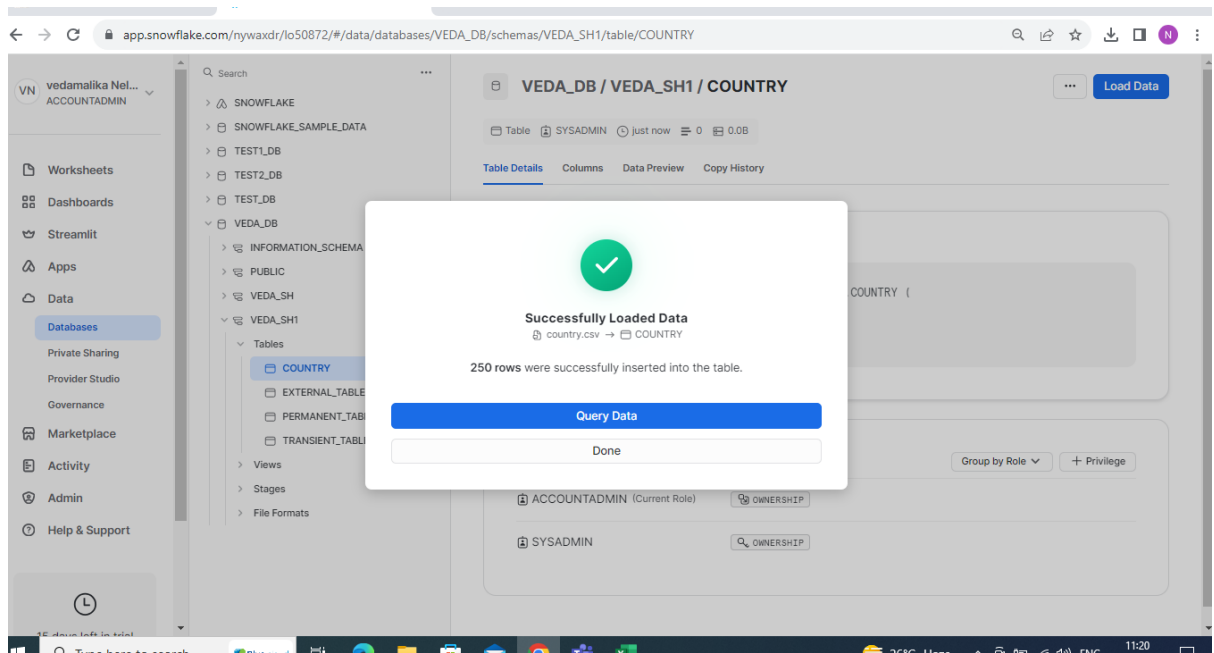
Query :-

```
create table country (  
name varchar(200),  
code varchar(200)  
);
```

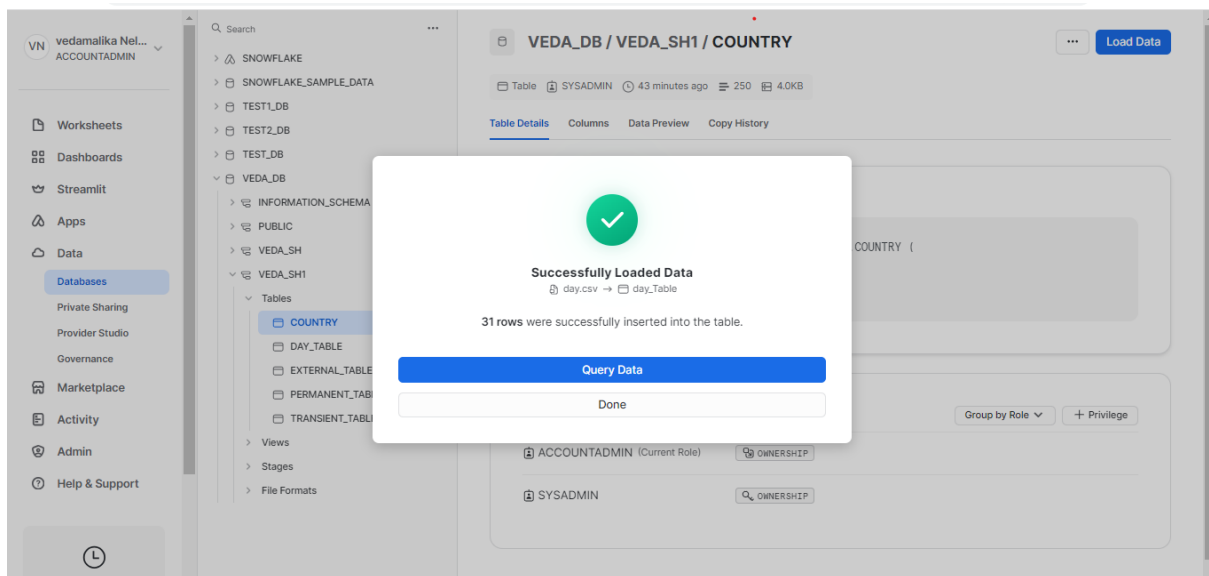
```
select * from country;
```

Data has been loaded into the country table successfully

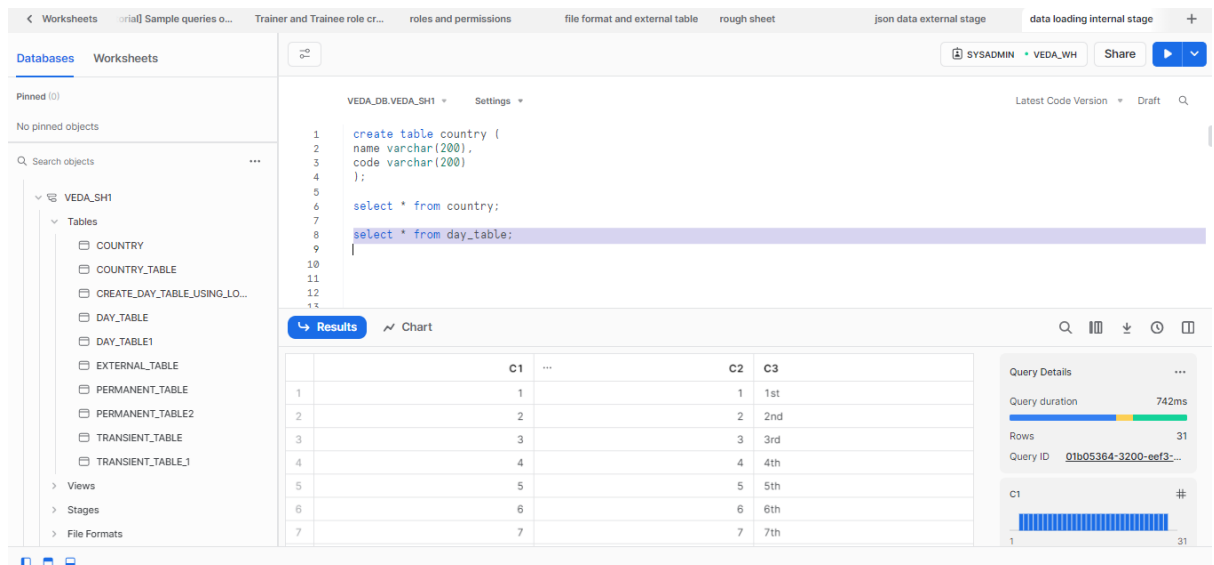
	NAME	CODE
1	Name	Code
2	Afghanistan	AF
3	Åland Islands	AX
4	Albania	AL
5	Algeria	DZ
6	American Samoa	AS
7	Andorra	AD
8	Angola	AO
9	Anguilla	AI
10	Antarctica	AQ
11	Antigua and Barbuda	AG
12	Argentina	AR
13	Armenia	AM
14	Aruba	AW
15	Australia	AU



2. Now I have downloaded a sample day.csv file and using Load wizard, I have selected a create a new table and gave the name of the table as day_table and loaded the data into the table.



I have retrieved the data from day_table successfully



The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the database structure for 'VEDA_DB', including tables like 'COUNTRY', 'DAY_TABLE', and 'PERMANENT_TABLE'. The main editor shows the following SQL code:

```
1 create table country (  
2   name varchar(200),  
3   code varchar(200)  
4 );  
5  
6 select * from country;  
7  
8 select * from day_table;  
9  
10  
11  
12  
13
```

The 'Results' tab shows the output of the query, which is a table with 7 rows and 3 columns (C1, C2, C3). The data is as follows:

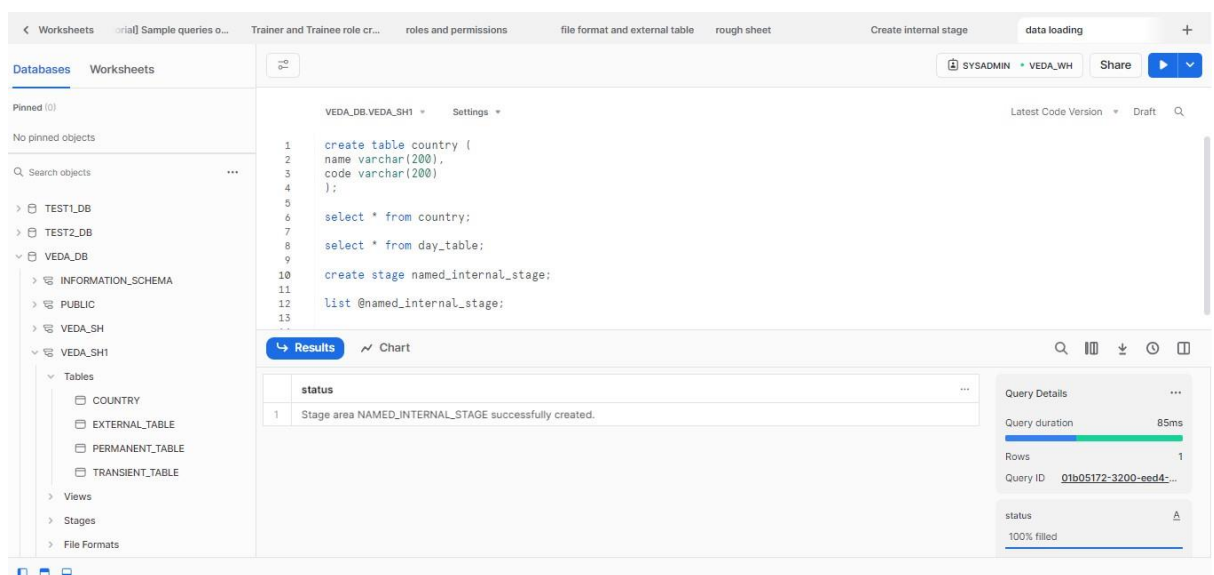
	C1	C2	C3
1	1	1	1st
2	2	2	2nd
3	3	3	3rd
4	4	4	4th
5	5	5	5th
6	6	6	6th
7	7	7	7th

The 'Query Details' panel on the right indicates a query duration of 742ms and 31 rows returned.

3. Named Internal Stage

- Now I am creating named internal stage.
- using put command I am loading the country csv file which is present in my local machine to the named internal stage.
- Then later on I am checking whether the file is loaded into the stage or not.
- Then using copy into command, I am loading the data from stage to tables in schema.

Creating the stage :-



The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the database structure for 'VEDA_DB', including tables like 'COUNTRY', 'DAY_TABLE', and 'PERMANENT_TABLE'. The main editor shows the following SQL code:

```
1 create table country (  
2   name varchar(200),  
3   code varchar(200)  
4 );  
5  
6 select * from country;  
7  
8 select * from day_table;  
9  
10 create stage named_internal_stage;  
11  
12 list @named_internal_stage;  
13
```

The 'Results' tab shows the output of the query, which is a single row with the status 'Stage area NAMED_INTERNAL_STAGE successfully created.'.

status
Stage area NAMED_INTERNAL_STAGE successfully created.

The 'Query Details' panel on the right indicates a query duration of 85ms and 1 row returned.

Put command :-

```
vedamalika#COMPUTE_WH@(<no database>).(<no schema>)>put file:///C:/Users/PC/Downloads/country.csv @veda_db.veda_sh1.named_in
ternal_stage;
-----+-----+-----+-----+-----+-----+-----+-----+
| source | target | source_size | target_size | source_compression | target_compression | status | message |
-----+-----+-----+-----+-----+-----+-----+-----+
| country.csv | country.csv.gz | 3869 | 2224 | NONE | GZIP | UPLOADED |
```

1 Row(s) produced. Time Elapsed: 3.545s

Listing the files in stage

The screenshot shows the Snowflake web interface. On the left, the 'Databases' sidebar is open, showing a tree view of databases including SNOWFLAKE, SNOWFLAKE_SAMPLE_DATA, TEST1_DB, TEST2_DB, INFORMATION_SCHEMA, PUBLIC, and VEDA_DB. The 'Worksheets' tab is active, displaying a SQL query in the editor:

```
9 //creating named internal stage
10 create stage named_internal_stage;
11
12 list @named_internal_stage;
13
14 //creating table now
15 create table country_table (
16   name varchar(250),
17   code varchar(250)
18 );
19
20 copy into country_table
21 from @named_internal_stage
22 file_format = 'csv_format'
23 on_error = 'skip_file';
24
```

Below the query editor, the 'Results' tab is selected, showing a table with the following data:

	name	size	md5	last_modified
1	named_internal_stage/country.csv.gz	2,224	5fb378d6746c8f27da7be607c3edd4d4	Tue, 14 Nov 2023 06:51:36 GV

On the right, the 'Query Details' panel shows the following information:

- Query duration: 63ms
- Rows: 1
- Query ID: 01b0537e-3200-ef1b-0...

Using copy into command

The screenshot shows the Snowflake web interface. On the left, the 'Databases' sidebar is open, showing a tree view of databases including SNOWFLAKE, SNOWFLAKE_SAMPLE_DATA, TEST1_DB, TEST2_DB, INFORMATION_SCHEMA, PUBLIC, and VEDA_DB. The 'Worksheets' tab is active, displaying a SQL query in the editor:

```
16 create table country_table (
17   name varchar(250),
18   code varchar(250)
19 );
20
21 copy into country_table
22 from @named_internal_stage
23 file_format = 'csv_format'
24 on_error = 'skip_file';
25
26 select * from country_table;
27
28
29
30
31
```

Below the query editor, the 'Results' tab is selected, showing a table with the following data:

	file	status	rows_parsed	rows_loaded	error_limit	errors_seen
1	named_internal_stage/country.csv.gz	LOADED	249	249	1	0

On the right, the 'Query Details' panel shows the following information:

- Query duration: 1.6s
- Rows: 1
- Query ID: 01b05180-3200-ef18-0...

I have copied the data to table and Successfully retrieved data from country_table.

The screenshot shows the Snowflake SQL Editor interface. On the left, the 'Databases' sidebar is open, showing the hierarchy: VEDA_DB > VEDA_SH1 > Tables. The main editor contains the following SQL code:

```
10 create stage named_internal_stage;
11
12 list @named_internal_stage;
13
14 //creating table now
15 create table country_table (
16   name varchar(250),
17   code varchar(250)
18 );
19
20 copy into country_table
21 from @named_internal_stage
22 file_format = 'csv_format'
23 on_error = 'skip_file';
24
25 select * from country_table;
```

The 'Results' tab is active, displaying a table with 4 rows and 2 columns: NAME and CODE.

	NAME	CODE
1	Afghanistan	AF
2	Åland Islands	AX
3	Albania	AL
4	Algeria	DZ

Query Details: Query duration 531ms, Rows 249, Query ID 01b0537f-3200-ef1b-0...

4. Table stage (internal)

Now I am loading the csv file from local machine to table stage using put command.

```
syntax error line 1 at position 4 unexpected 'file'
vedamalika@COMPUTE_WH@(<no database>).(<no schema>)>put file:///C:/Users/PC/Downloads/day.csv @veda_db.veda_sh1.%permanent_table2;
```

source	target	source_size	target_size	source_compression	target_compression	status	message
day.csv	day.csv.gz	355	192	NONE	GZIP	UPLOADED	

1 Row(s) produced. Time Elapsed: 5.245s

I am listing the files present in table stage

The screenshot shows the Snowflake SQL Editor interface. On the left, the 'Databases' sidebar is open, showing the hierarchy: VEDA_DB > VEDA_SH1 > Tables. The main editor contains the following SQL code:

```
29 //creating the table stage
30 CREATE TABLE Permanent_Table2 (
31   id INT,
32   id_2 INT,
33   id_suffix VARCHAR(200)
34 );
35
36 list @%permanent_table2;
37
38 copy into permanent_table2
39 from @permanent_table2
40 file_format = 'csv_format'
41 on_error = 'skip_file';
42
43 select * from permanent_table2;
```

The 'Results' tab is active, displaying a table with 1 row and 5 columns: name, size, md5, last_modified, and an ellipsis.

	name	size	md5	last_modified	
1	day.csv.gz	192	32de39074d2ca0be37f6e2e223ee7222	Tue, 14 Nov 2023 07:12:30 GMT	

Query Details: Query duration 61ms, Rows 1, Query ID 01b0538e-3200-ef1b-0...

I am using the copy into command and loading the file from table stage to permanent_table

The screenshot shows the Snowflake SQL Editor interface. On the left, the 'Databases' tab is active, showing a tree of objects including 'PERMANENT_TABLE2'. The main editor area contains the following SQL code:

```
35
36
37 list @permanent_table2;
38
39 copy into permanent_table2
40 from @permanent_table2
41 file_format = 'csv_format'
42 on_error = 'skip_file';
43
44 select * from permanent_table2;
45
46
47 //user stage
48 list @~;
49
50 create table transient_table_1(
51 id int
```

Below the code, the 'Results' tab is selected, displaying a table with the following data:

file	status	rows_parsed	rows_loaded	error_limit	errors_seen	first_error
day.csv.gz	LOADED	31	31	1	0	null

On the right, the 'Query Details' panel shows a progress bar for 'Query duration' at 1.0s, 'Rows' at 1, and 'Query ID' 01b05192-3200-ef18-0...

Retrieving the data from the permanent_table

The screenshot shows the Snowflake SQL Editor interface. On the left, the 'Databases' tab is active, showing a tree of objects including 'PERMANENT_TABLE2'. The main editor area contains the following SQL code:

```
35
36
37 list @permanent_table2;
38
39 copy into permanent_table2
40 from @permanent_table2
41 file_format = 'csv_format'
42 on_error = 'skip_file';
43
44 select * from permanent_table2;
45
46
47 //user stage
48 list @~;
49
50 create table transient_table_1(
51 id int
```

Below the code, the 'Results' tab is selected, displaying a table with the following data:

	ID	ID_2	ID_SUFFIX
1	1	1	1st
2	2	2	2nd
3	3	3	3rd
4	4	4	4th
5	5	5	5th

On the right, the 'Query Details' panel shows a progress bar for 'Query duration' at 130ms, 'Rows' at 93, and 'Query ID' 01b05398-3200-ef16-0...

5. User stage

Now I am loading the csv file from local machine to user stage using put command.

```
vedamalika@COMPUTE_WH:(no database).(no schema)>put file:///C:/Users/PC/Downloads/day.csv @~;
+-----+-----+-----+-----+-----+-----+-----+-----+
| source | target | source_size | target_size | source_compression | target_compression | status | message |
+-----+-----+-----+-----+-----+-----+-----+-----+
| day.csv | day.csv.gz | 355 | 192 | NONE | GZIP | UPLOADED | |
+-----+-----+-----+-----+-----+-----+-----+-----+
1 Row(s) produced. Time Elapsed: 4.307s
```

I am listing the files present in user stage

The screenshot shows the Databricks interface. On the left, the 'Databases' sidebar lists various tables, with 'PERMANENT_TABLE2' selected. The main area displays a SQL query in the editor:

```
//USER stage
list @~;

create table transient_table_1(
  id_1 int,
  id_2 int,
  id_suffix_1 varchar(250)
);

copy into transient_table_1
from @~
file_format='CSV_FORMAT'
on_error='skip_file';

select * from transient_table_1;
```

Below the editor, the 'Results' tab shows a table with the following data:

name	size	md5	last_modified
day.csv.gz	192	8f12fa15fdbce5d44b64c241030d3a44	Tue, 14 Nov 2023 07:16:41 GMT

Query Details on the right indicate a duration of 85ms and 1 row.

I am using the copy into command and loading the file from user stage to permanent_table

The screenshot shows the Databricks interface. The SQL query in the editor is:

```
copy into permanent_table_1
from @~
file_format='CSV_FORMAT'
on_error='skip_file';

select * from permanent_table_1;
```

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

file	status	rows_parsed	rows_loaded	error_limit	errors_seen	first_error
day.csv.gz	LOADED	31	31	1	0	null

Query Details on the right show a duration of 786ms and 1 row.

Retrieving the data from the transient_table_1

The screenshot shows the Databricks interface. The SQL query in the editor is:

```
select * from transient_table_1;
```

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

ID_1	ID_2	ID_SUFFIX_1
1	1	1st
2	2	2nd
3	3	3rd
4	4	4th
5	5	5th
6	6	6th
7	7	7th

Query Details on the right show a duration of 424ms and 31 rows.

6. Loading the data using external stage

Now I have downloaded a sample json file and loaded the file into AWS external stage and now I am listing the files in s3 bucket

The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the database schema with categories like Pinned, Views, Stages, and File Formats. The main editor contains the following SQL code:

```
1 CREATE OR REPLACE TABLE details4_json
2 {
3   "RAW_AUTHOR" VARIANT
4 };
5
6 CREATE OR REPLACE FILE FORMAT veda_db.veda_sh1.json_format
7 TYPE = 'JSON'
8 COMPRESSION = 'AUTO'
9 ENABLE_OCTAL = FALSE
10 ALLOW_DUPLICATE = FALSE
11 STRIP_OUTER_ARRAY = TRUE
12 STRIP_NULL_VALUES = FALSE
13 IGNORE_UTF8_ERRORS = FALSE;
14
15 list @AWS_EXTERNAL_STAGE;
```

Below the editor, the 'Results' tab shows a table with 4 rows and 3 columns: name, size, and md5.

	name	size	md5
1	s3://vedabucket-101123/a.json	317	3056e44b42904781f857e4e4
2	s3://vedabucket-101123/annual-enterprise-survey-2021-financial-year-provisions	1,492,633	c5c7761c40fbec8c990cef00c
3	s3://vedabucket-101123/country.csv	3,869	a86aa14ff61ef53622c2809dbx
4	s3://vedabucket-101123/d.json	217	3056e44b42904781f857e4e4

Query Details on the right show a duration of 715ms and 5 rows.

Now I am creating a json table

The screenshot shows the Snowflake SQL Editor interface. The left sidebar displays the database schema. The main editor contains the following SQL code:

```
1 CREATE OR REPLACE TABLE details4_json
2 {
3   "RAW_AUTHOR" VARIANT
4 };
5
6 CREATE OR REPLACE FILE FORMAT veda_db.veda_sh1.json_format
7 TYPE = 'JSON'
8 COMPRESSION = 'AUTO'
9 ENABLE_OCTAL = FALSE
10 ALLOW_DUPLICATE = FALSE
11 STRIP_OUTER_ARRAY = TRUE
12 STRIP_NULL_VALUES = FALSE
13 IGNORE_UTF8_ERRORS = FALSE;
14
15 list @AWS_EXTERNAL_STAGE;
```

Below the editor, the 'Results' tab shows a single row with the status: 'Table DETAILS4_JSON successfully created.'

status
1 Table DETAILS4_JSON successfully created.

Query Details on the right show a duration of 132ms and 1 row.

Now I am loading the data from external stage to json table using copy into command

VEDA.DB.VEDA_SH1 - Settings - Latest Code Version

```

38 COPY INTO DETAILS4_JSON
39 FROM @aws_external_stage
40 FILE_FORMAT = 'json_format'
41 PATTERN = 'a.json'
42 on_error = 'continue';
43
44 select * from details4_json;
45
46
47 SELECT
48   raw_author:color::STRING AS color,
49   raw_author:value::STRING AS value
50 FROM
51   details4_json;

```

Results Chart

file	status	rows_parsed	rows_loaded	error_limit	errors_seen
s3://vedabucket-101123/a.json	LOADED	7	7	7	0

Query Details

- Query duration: 1.8s
- Rows: 1
- Query ID: 01b053ac-3200-ef1b-0...
- file: 100% filled

DETAILS4_JSON

- RAW_AUTHOR

Now I am retrieving the data from json_table

VEDA.DB.VEDA_SH1 - Settings - Latest Code Version

```

38 COPY INTO DETAILS4_JSON
39 FROM @aws_external_stage
40 FILE_FORMAT = 'json_format'
41 PATTERN = 'a.json'
42 on_error = 'continue';
43
44 select * from details4_json;
45
46
47 SELECT
48   raw_author:color::STRING AS color,
49   raw_author:value::STRING AS value
50 FROM
51   details4_json;

```

Results Chart

RAW_AUTHOR
{ "color": "red", "value": "#f00" }
{ "color": "green", "value": "#0f0" }
{ "color": "blue", "value": "#00f" }
{ "color": "cyan", "value": "#0ff" }
{ "color": "magenta", "value": "#f0f" }
{ "color": "yellow", "value": "#ff0" }

Query Details

- Query duration: 1.3s
- Rows: 7
- Query ID: 01b053ad-3200-ef38-...
- RAW_AUTHOR: 100% filled

DETAILS4_JSON

- RAW_AUTHOR

Now in tabular format We need to keep STRIP_OUTER_ARRAY = TRUE

Then result is :-

VEDA.DB.VEDA_SH1 - Settings - Latest Code Version

```

42 on_error = 'continue';
43
44 select * from details4_json;
45
46
47 SELECT
48   raw_author:color::STRING AS color,
49   raw_author:value::STRING AS value
50 FROM
51   details4_json;

```

Results Chart

COLOR	VALUE
1 red	#f00
2 green	#0f0
3 blue	#00f
4 cyan	#0ff
5 magenta	#f0f
6 yellow	#ff0

Query Details

- Query duration: 296ms
- Rows: 7
- Query ID: 01b053ae-3200-ef3-0...
- COLOR: 100% filled

DETAILS4_JSON

- RAW_AUTHOR

7. Data Unloading

Now I am creating unload_stage

The screenshot shows a database interface with a top navigation bar containing tabs like 'Worksheets', 'Trainer and Trainee role cr...', 'roles and permissions', 'file format and external table', 'rough sheet', 'json data external stage', and 'data loading internal stage'. The 'data loading internal stage' tab is active. On the left, a sidebar shows a tree view of objects: 'Pinned (0)', 'No pinned objects', 'Search objects', 'EXTERNAL_TABLE', 'PERMANENT_TABLE', 'PERMANENT_TABLE2', 'TRANSIENT_TABLE', 'TRANSIENT_TABLE_1' (selected), 'Views', 'Stages' (expanded), 'AWS_EXTERNAL_STAGE', 'MY_UNLOAD_STAGE', 'NAMED_INTERNAL_STAGE', and 'File Formats'. Below this, a table for 'TRANSIENT_TABLE_1' is shown with 31 rows and columns: ID_1 (NUMBER(38,0)), ID_2 (NUMBER(38,0)), and ID_SUFFIX_1 (VARCHAR(250)). The main editor area shows a SQL script for VEDA_DB.VEDA_SH1 with the following code:

```
--  
78  
79 //unloading  
80 create stage my_unload_stage;  
81  
82 list @my_unload_stage;  
83  
84  
85 copy into @my_unload_stage/unload/country_table  
86 from veda_db.veda_sh1.country_table;  
87  
88  
89
```

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

status
1 Stage area MY_UNLOAD_STAGE successfully created.

On the right, a 'Query Details' panel shows: 'Query duration: 72ms', 'Rows: 1', 'Query ID: 01b051ef-3200-ef18-9...', and a progress bar for 'status' at 100% filled.

Then I am using copy into command and unloading the data into my_unload stage from table called country_table

The screenshot shows the same database interface as before, but the SQL script in the main editor is now:

```
--  
78  
79 //unloading  
80 create stage my_unload_stage;  
81  
82 list @my_unload_stage;  
83  
84  
85 copy into @my_unload_stage/unload/country_table  
86 from veda_db.veda_sh1.country_table;  
87  
88  
89
```

The 'Results' tab now shows a table with columns: 'rows_unloaded', 'input_bytes', and 'output_bytes'.

rows_unloaded	input_bytes	output_bytes
249	3,844	2,219

The 'Query Details' panel on the right shows: 'Query duration: 2.6s', 'Rows: 1', 'Query ID: 01b051ef-3200-ee44-0...', and a progress bar for 'rows_unloaded' at 100% filled.

Next I am listing the table in stage

The screenshot shows the Snowflake SQL Editor interface. On the left, the 'Databases' sidebar is open, showing a tree view of databases and stages. The 'VEDA_DB.VEDA_SH1' database is selected, and the 'MY_UNLOAD_STAGE' stage is highlighted. The main editor area shows a SQL query:

```
--  
78  
79 //unloading  
80 create stage my_unload_stage;  
81  
82 list @my_unload_stage;  
83  
84  
85 copy into @my_unload_stage/unload/country_table  
86 from veda_db.veda_sh1.country_table;  
87  
88  
89
```

The query results are displayed in a table with the following columns: name, size, md5, and last_modified. The results show a single row for the file 'my_unload_stage/unload/country_table_0_0_0.csv.gz'.

	name	size	md5	last_modified
1	my_unload_stage/unload/country_table_0_0_0.csv.gz	2,224	8b467890da5f5c42ea57e57a32409280	Tue, 14 Nov 2023 10:00:00

On the right, the 'Query Details' panel shows the query duration as 71ms, 1 row, and the query ID as 01b053ba-3200-ef18-0...

Then I am unloading the file from my_unload_stage to local machine documents folder using “GET” command

The screenshot shows a terminal window with the following commands and output:

```
vedamalika@COMPUTE_WH@ (no database). (no schema) > get @veda_db.veda_sh1.my_unload_stage/unload/country_table_0_0_0.csv.gz file://c:\users\PC\Documents;  
+-----+-----+-----+-----+  
| file | size | status | message |  
+-----+-----+-----+-----+  
| country_table_0_0_0.csv.gz | 2219 | DOWNLOADED |  
+-----+-----+-----+-----+  
1 Row(s) produced. Time Elapsed: 3.794s  
vedamalika@COMPUTE_WH@ (no database). (no schema) >  
vedamalika@COMPUTE_WH@ (no database). (no schema) >
```