Homework 3

1. (+1) Explain what NPS (Net Promoter Score) is and how it is calculated

Net Promoter Score (NPS) uses a straightforward survey question to gauge customer loyalty:

"On a scale of 0 to 10, how likely are you to recommend us to a friend?"

Promoters (9–10): Faithful and inclined to suggest.

Passives (7-8): Not overly excited, yet content.

Critics (0–6): Disappointed and unlikely to suggest.

The formula for calculating NPS is:

NPS=%Promoters−%Opponents

NPS=%Detractors−%Promoters

This results in a score ranging from -100 to 100.

Interpretation:

Positive NPS (0 to 100): Indicates more promoters than detractors, suggesting strong customer loyalty.

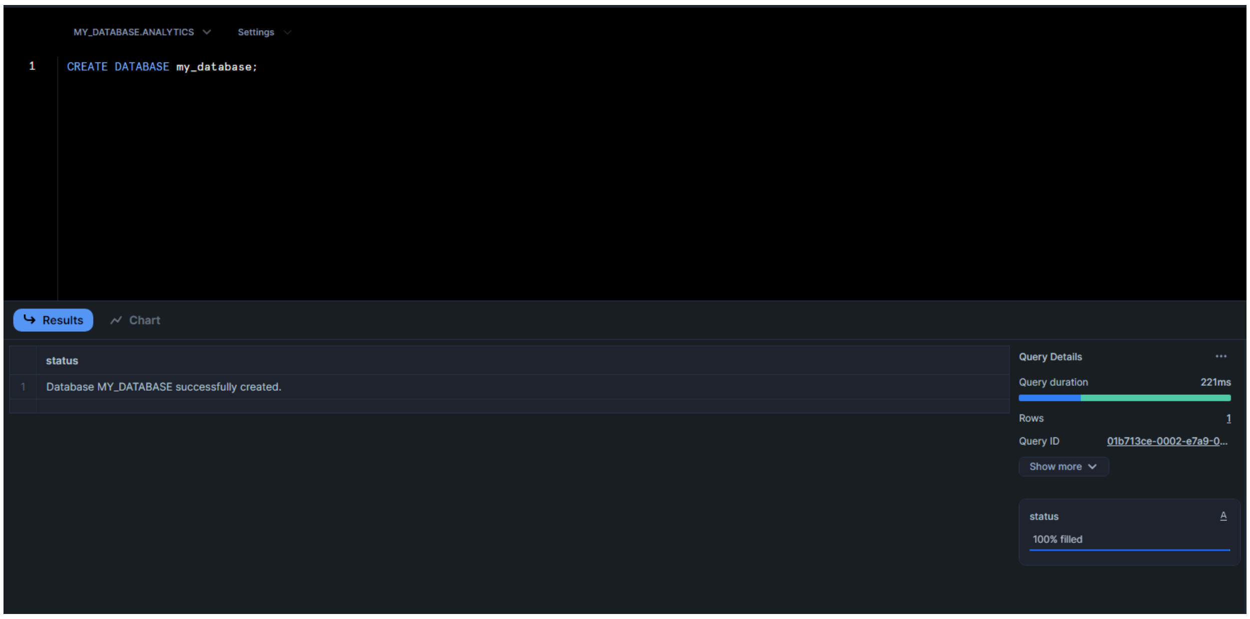
Negative NPS (-100 to 0): Indicates more detractors than promoters, signaling potential dissatisfaction.

2. (+1) Create raw_data and analytics schemas under a database in your Snowflake

CREATE DATABASE my_database;

CREATE SCHEMA my_database.raw_data;

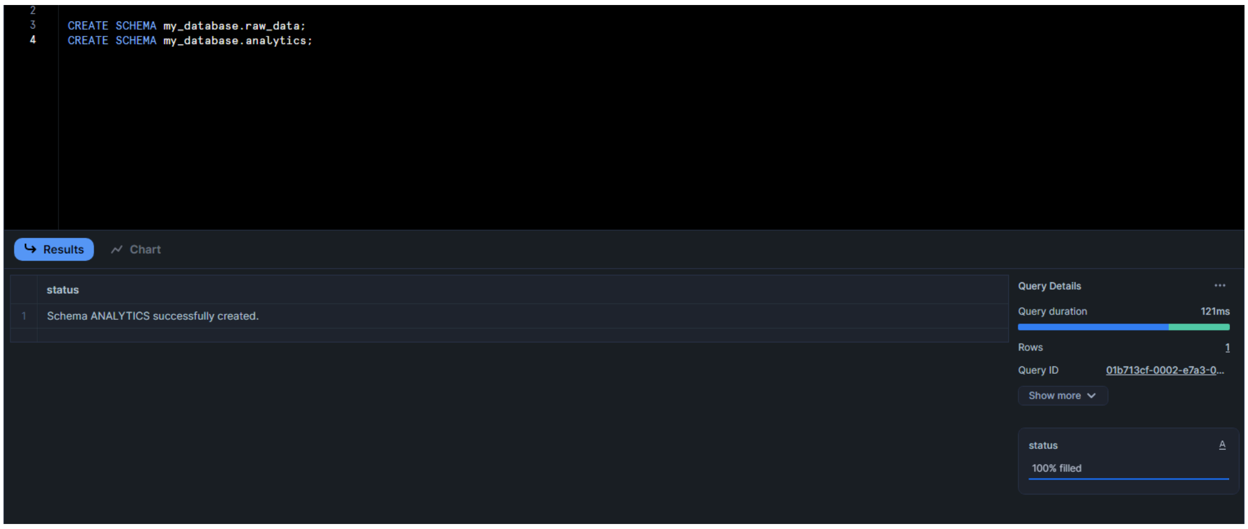
CREATE SCHEMA my_database.analytics;



```
CREATE DATABASE my_database;
```

```
status
1  Database MY_DATABASE successfully created.
```

```
Query duration   221ms
Rows             1
Query ID         01b713ce-0002-e7a9-0...
status
100% filled
```



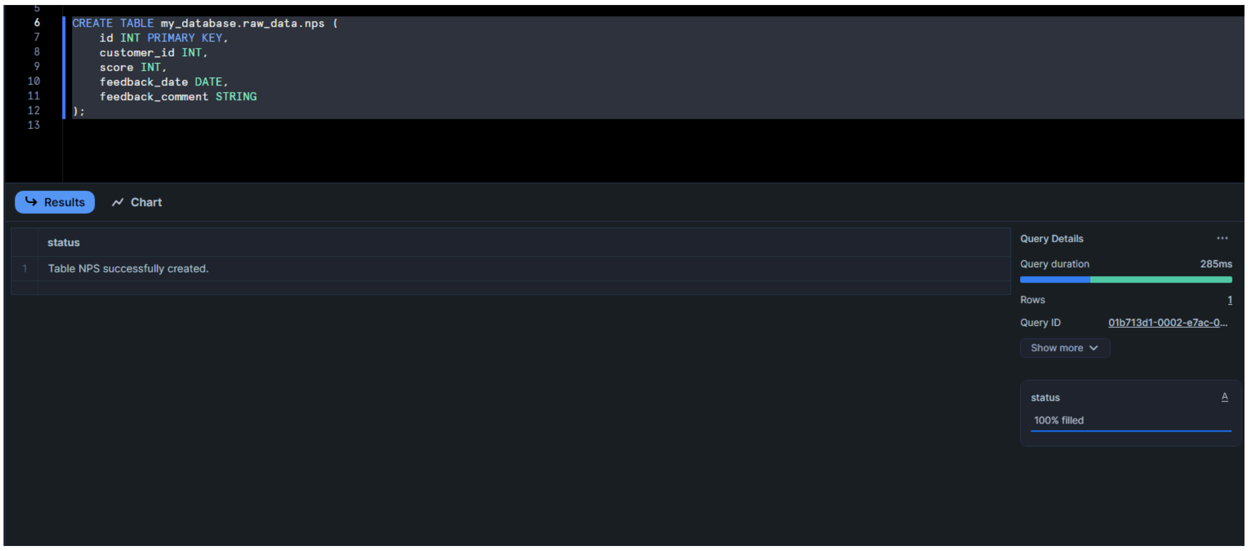
```
CREATE SCHEMA my_database.raw_data;
CREATE SCHEMA my_database.analytics;
```

```
status
Schema ANALYTICS successfully created.
```

```
Query Details
Query duration        121ms
Rows                  1
Query ID              01b713cf-0002-e7a3-0...
Show more
status
100% filled
```

3. (+1) Create a table named nps with primary key attribute under raw_data schema

```
CREATE OR REPLACE TABLE my_database.raw_data.nps (
    id VARCHAR PRIMARY KEY,
    feedback_date TIMESTAMP,
    score INT
);
```

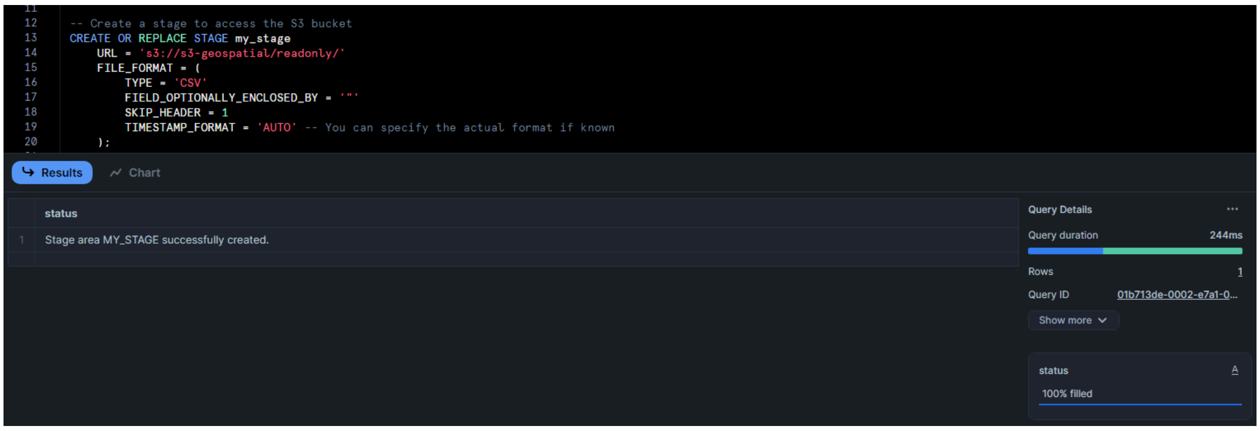


```
CREATE TABLE my_database.raw_data.nps (
    id INT PRIMARY KEY,
    customer_id INT,
    score INT,
    feedback_date DATE,
    feedback_comment STRING
);
```

```
status
Table NPS successfully created.
```

```
Query Details
Query duration        285ms
Rows                  1
Query ID              01b713d1-0002-e7ac-0...
Show more
status
100% filled
```

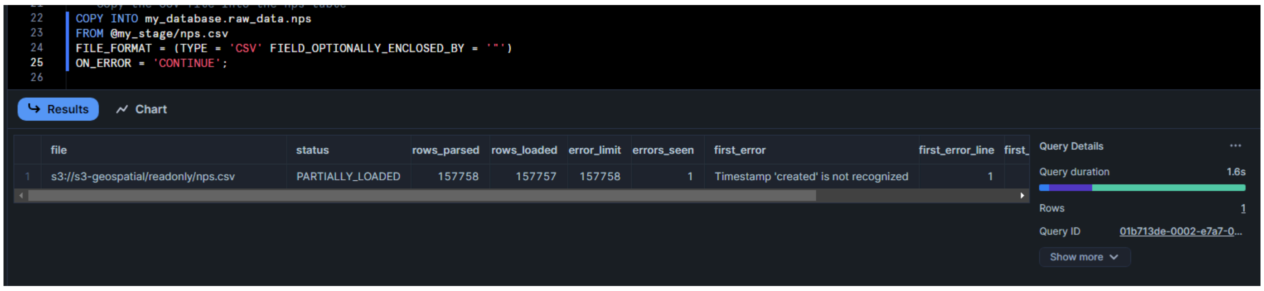
4. (+2) Copy a file (nps.csv) into the nps table using a stage
    1. s3://s3-geospatial/readonly/nps.csv



```
-- Create a stage to access the S3 bucket
CREATE OR REPLACE STAGE my_stage
    URL = 's3://s3-geospatial/readonly/'
    FILE_FORMAT = (
        TYPE = 'CSV'
        FIELD_OPTIONALLY_ENCLOSED_BY = '"'
        SKIP_HEADER = 1
        TIMESTAMP_FORMAT = 'AUTO' -- You can specify the actual format if known
    );
```

```
status
Stage area MY_STAGE successfully created.
```

```
Query Details
Query duration        244ms
Rows                  1
Query ID              01b713de-0002-e7a1-0...
Show more
status
100% filled
```



```
COPY INTO my_database.raw_data.nps
FROM @my_stage/nps.csv
FILE_FORMAT = (TYPE = 'CSV' FIELD_OPTIONALLY_ENCLOSED_BY = '"')
ON_ERROR = 'CONTINUE';
```

	file	status	rows_parsed	rows_loaded	error_limit	errors_seen	first_error	first_error_line	first_
1	s3://s3-geospatial/readonly/nps.csv	PARTIALLY_LOADED	157758	157757	157758	1	Timestamp 'created' is not recognized	1	

Query Details
Query duration 1.6s
Rows 1
Query ID 01b713de-0002-e7a7-0...

CREATE OR REPLACE STAGE my_stage

URL = 's3://s3-geospatial/readonly/'

FILE_FORMAT = (

TYPE = 'CSV'

FIELD_OPTIONALLY_ENCLOSED_BY = '"'

SKIP_HEADER = 1

TIMESTAMP_FORMAT = 'AUTO' -- You can specify the actual format if known

);

COPY INTO my_database.raw_data.nps

FROM @my_stage/nps.csv

FILE_FORMAT = (TYPE = 'CSV' FIELD_OPTIONALLY_ENCLOSED_BY = '"')

ON_ERROR = 'CONTINUE';

5. (+2) What types of data quality validations can we perform against the nps table? Please name at least 3 methods

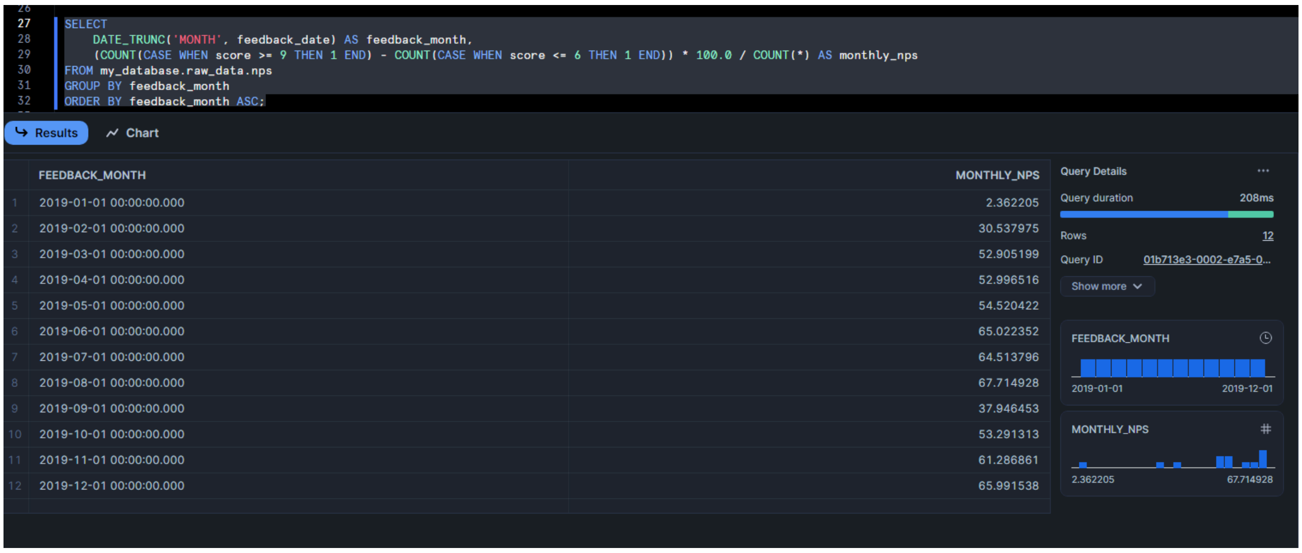
Validations of Data Quality for the nps Table:

Look for any null or missing values: Verify that no null values exist in any significant columns (such as customer_id and score).

Validation of the NPS score range: There should never be a number outside of 0 and 10.

Multiple checks: Verify that no records or entries with the customer_id are duplicates.

6. (+4) Develop a SELECT SQL query to calculate monthly NPS, with results sorted by month in ascending order..



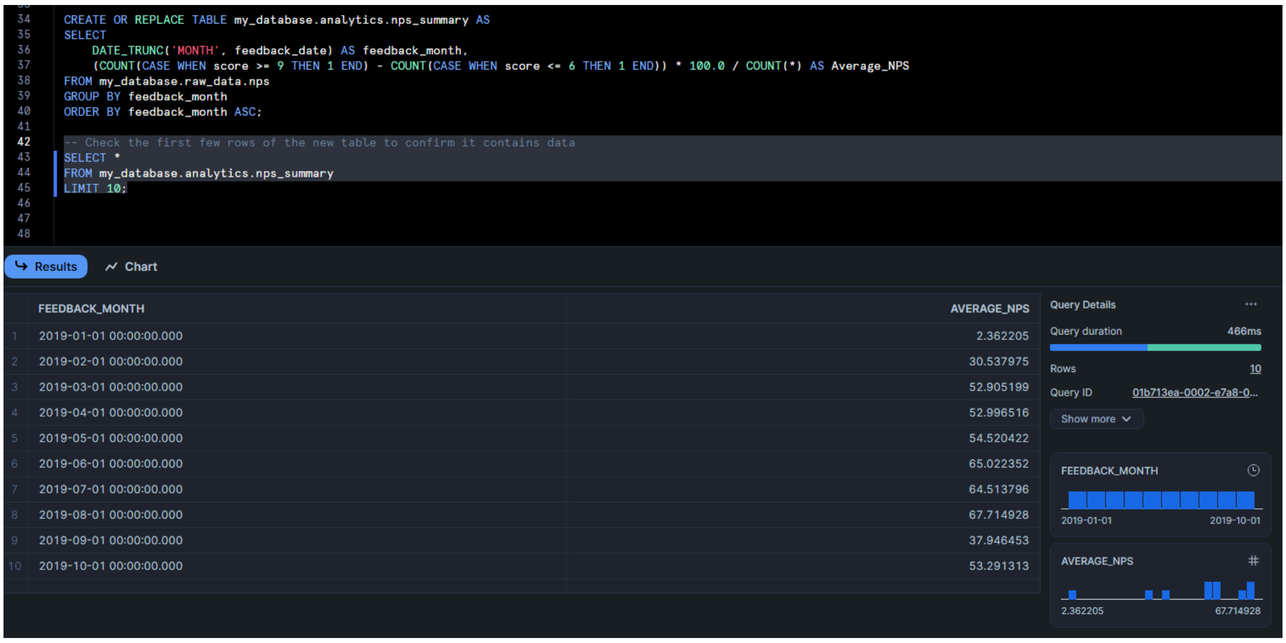
```
SELECT
    DATE_TRUNC('MONTH', feedback_date) AS feedback_month,
    (COUNT(CASE WHEN score >= 9 THEN 1 END) - COUNT(CASE WHEN score <= 6 THEN 1 END)) * 100.0 / COUNT(*) AS monthly_nps
FROM my_database.raw_data.nps
GROUP BY feedback_month
ORDER BY feedback_month ASC;
```

	FEEDBACK_MONTH	MONTHLY_NPS
1	2019-01-01 00:00:00.000	2.362205
2	2019-02-01 00:00:00.000	30.537975
3	2019-03-01 00:00:00.000	52.905199
4	2019-04-01 00:00:00.000	52.996516
5	2019-05-01 00:00:00.000	54.520422
6	2019-06-01 00:00:00.000	65.022352
7	2019-07-01 00:00:00.000	64.513796
8	2019-08-01 00:00:00.000	67.714928
9	2019-09-01 00:00:00.000	37.946453
10	2019-10-01 00:00:00.000	53.291313
11	2019-11-01 00:00:00.000	61.286861
12	2019-12-01 00:00:00.000	65.991538

Query Details
Query duration 208ms
Rows 12
Query ID 01b713e3-0002-e7a5-0...

```
SELECT
    DATE_TRUNC('MONTH', feedback_date) AS feedback_month,
    (COUNT(CASE WHEN score >= 9 THEN 1 END) - COUNT(CASE WHEN score <= 6 THEN 1 END)) * 100.0 / COUNT(*) AS monthly_nps
FROM my_database.raw_data.nps
GROUP BY feedback_month
ORDER BY feedback_month ASC;
```

7. (+2) Use CTAS (CREATE TABLE AS SELECT) to generate the nps_summary table in the analytics schema, populating it with results from step 6.



```
CREATE OR REPLACE TABLE my_database.analytics.nps_summary AS
SELECT
    DATE_TRUNC('MONTH', feedback_date) AS feedback_month,
    (COUNT(CASE WHEN score >= 9 THEN 1 END) - COUNT(CASE WHEN score <= 6 THEN 1 END)) * 100.0 / COUNT(*) AS Average_NPS
FROM my_database.raw_data.nps
GROUP BY feedback_month
ORDER BY feedback_month ASC;

-- Check the first few rows of the new table to confirm it contains data
SELECT *
FROM my_database.analytics.nps_summary
LIMIT 10;
```

	FEEDBACK_MONTH	AVERAGE_NPS
1	2019-01-01 00:00:00.000	2.362205
2	2019-02-01 00:00:00.000	30.537975
3	2019-03-01 00:00:00.000	52.905199
4	2019-04-01 00:00:00.000	52.996516
5	2019-05-01 00:00:00.000	54.520422
6	2019-06-01 00:00:00.000	65.022352
7	2019-07-01 00:00:00.000	64.513796
8	2019-08-01 00:00:00.000	67.714928
9	2019-09-01 00:00:00.000	37.946453
10	2019-10-01 00:00:00.000	53.291313

Query Details
Query duration 466ms
Rows 10
Query ID 01b713ea-0002-e7a8-0...

```
CREATE OR REPLACE TABLE my_database.analytics.nps_summary AS
SELECT
    DATE_TRUNC('MONTH', feedback_date) AS feedback_month,
    (COUNT(CASE WHEN score >= 9 THEN 1 END) - COUNT(CASE WHEN score <= 6 THEN 1 END)) * 100.0 / COUNT(*) AS Average_NPS
FROM my_database.raw_data.nps
GROUP BY feedback_month
ORDER BY feedback_month ASC;

-- Check the first few rows of the new table to confirm it contains data
SELECT *
FROM my_database.analytics.nps_summary
LIMIT 10;
```