Step 1: Setup the Environment
==========

prerequisite: Create a worksheet semistructured

```
-- Create database and schema
CREATE OR REPLACE DATABASE semi_structured_lab;

USE DATABASE semi_structured_lab;

CREATE OR REPLACE SCHEMA lab_schema;

USE SCHEMA lab_schema;

Create a warehouse
CREATE OR REPLACE WAREHOUSE lab_wh WITH WAREHOUSE_SIZE = 'XSMALL' AUTO_SUSPEND =
60 AUTO_RESUME = TRUE;
USE WAREHOUSE lab_wh;
```

Step 2: Create a Table with VARIANT Column
==========

```
CREATE OR REPLACE TABLE customer_profiles (
    customer_id INT,
    name STRING,
    profile VARIANT
);
```

NOTE:profile will store JSON data with various keys like age, location, preferences, etc.

Step 3: Insert JSON Data into the VARIANT Column
==========

```
INSERT INTO customer_profiles (customer_id, name, profile)
SELECT 1, 'Alice', PARSE_JSON('{
  "age": 30,
  "location": "New York",
  "interests": ["books", "music"]
}')
UNION ALL
SELECT 2, 'Bob', PARSE_JSON('{
  "age": 25,
  "location": "San Francisco",
  "interests": ["gaming", "travel"]
}')
UNION ALL
SELECT 3, 'Charlie', PARSE_JSON('{
  "age": 35,
  "location": "Chicago"
}');
```

Step 4: Query JSON Fields from VARIANT Column
==========

```
SELECT
    customer_id,
    name,
    profile:age AS age,
    profile:location AS location
FROM customer_profiles;
```

NOTE:The profile:age syntax extracts the age field from the JSON object stored

in profile.

```
 Step 5: Filter Rows Based on JSON Field Values
================================================

SELECT
    name,
    profile:location AS location
FROM customer_profiles
WHERE profile:age > 30;

Step 6: Flatten JSON Arrays
===========================
Let’s create another table with an array in the JSON structure.

CREATE OR REPLACE TABLE orders (
    order_id INT,
    customer_name STRING,
    items VARIANT
);

INSERT INTO orders (order_id, customer_name, items)
SELECT 101, 'Alice', PARSE_JSON('[{"item": "Book", "price": 15.5}, {"item":
"Notebook", "price": 5.0}]')
UNION ALL
SELECT 102, 'Bob', PARSE_JSON('[{"item": "Mouse", "price": 25.0}, {"item":
"Keyboard", "price": 45.0}]');

Now flatten the array:
SELECT
    order_id,
    customer_name,
    flattened.value:item AS item,
    flattened.value:price AS price
FROM orders,
LATERAL FLATTEN(input => items) AS flattened;

===================
==>Count customers by location:

SELECT profile:location AS location, COUNT(*) AS count
FROM customer_profiles
GROUP BY profile:location;
```