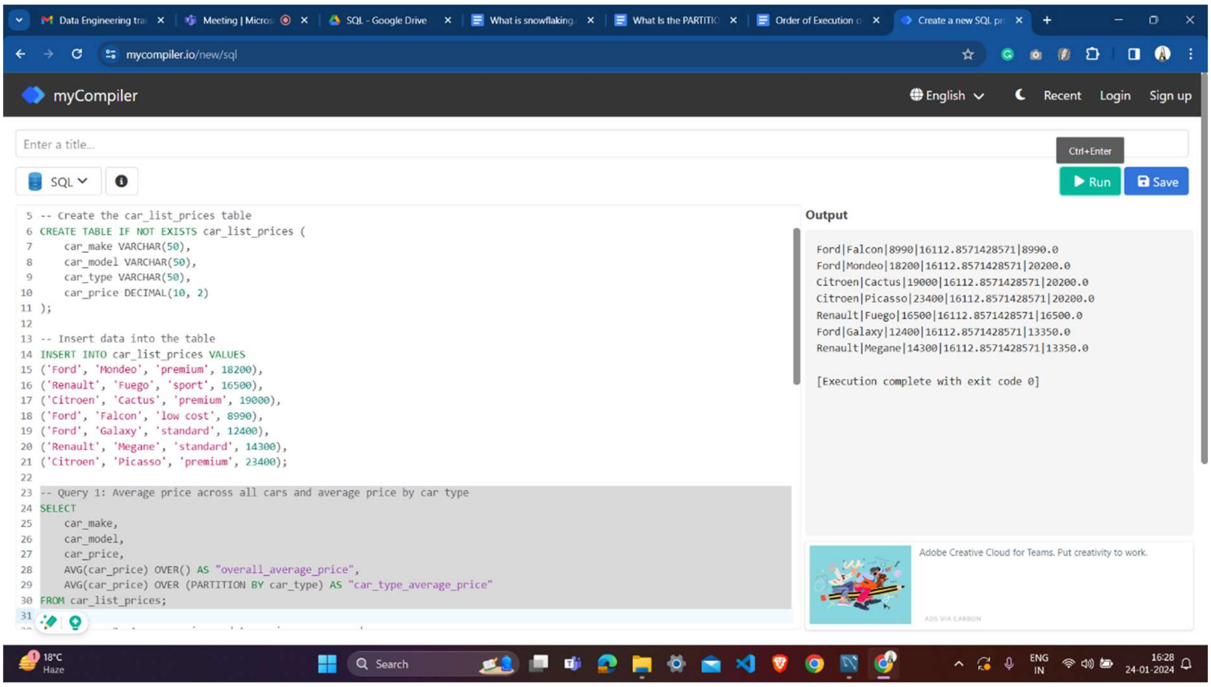
Name: Abhishek Kanoujia

DATA ENGINEERING BATCH 1

DAY 6 ASSIGNMENT

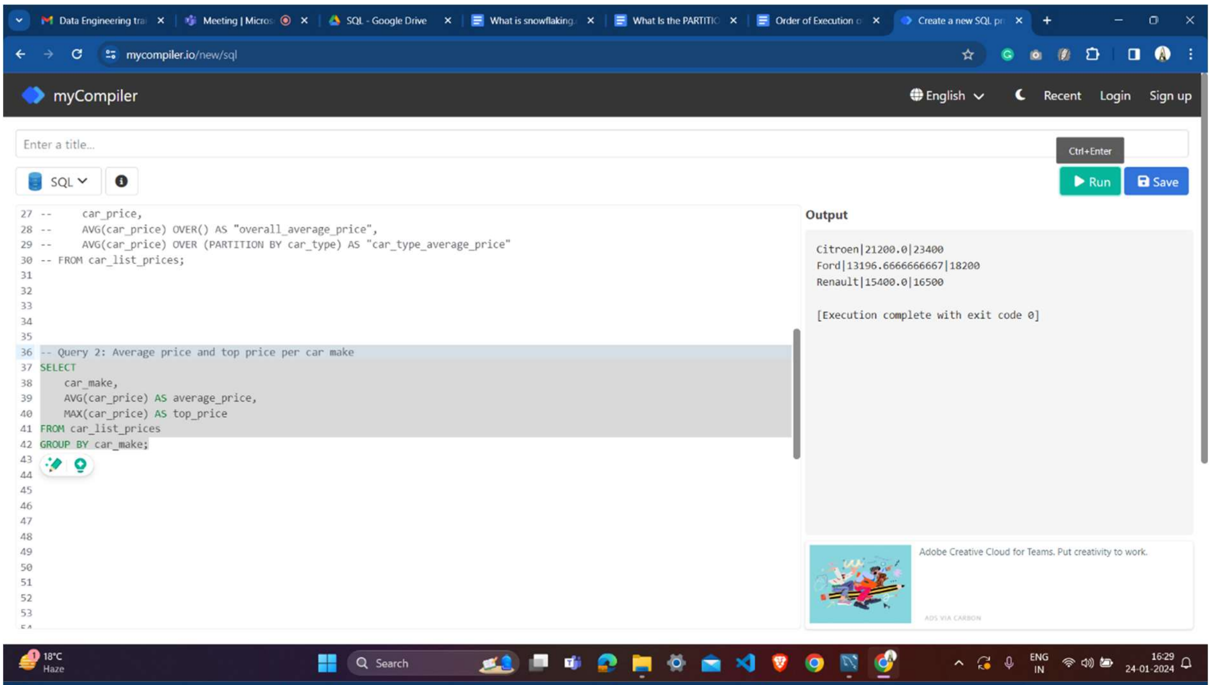
What Is the PARTITION BY Clause in SQL?

The SQL `PARTITION BY` expression is a subclause of the `OVER` clause, which is used in almost all invocations of window functions like `AVG()`, `MAX()`, and `RANK()`. As many readers probably know, window functions operate on window frames which are sets of rows that can be different for each record in the query result. This is where the SQL `PARTITION BY` subclause comes in: it is used to define which records to make part of the window frame associated with each record of the result.



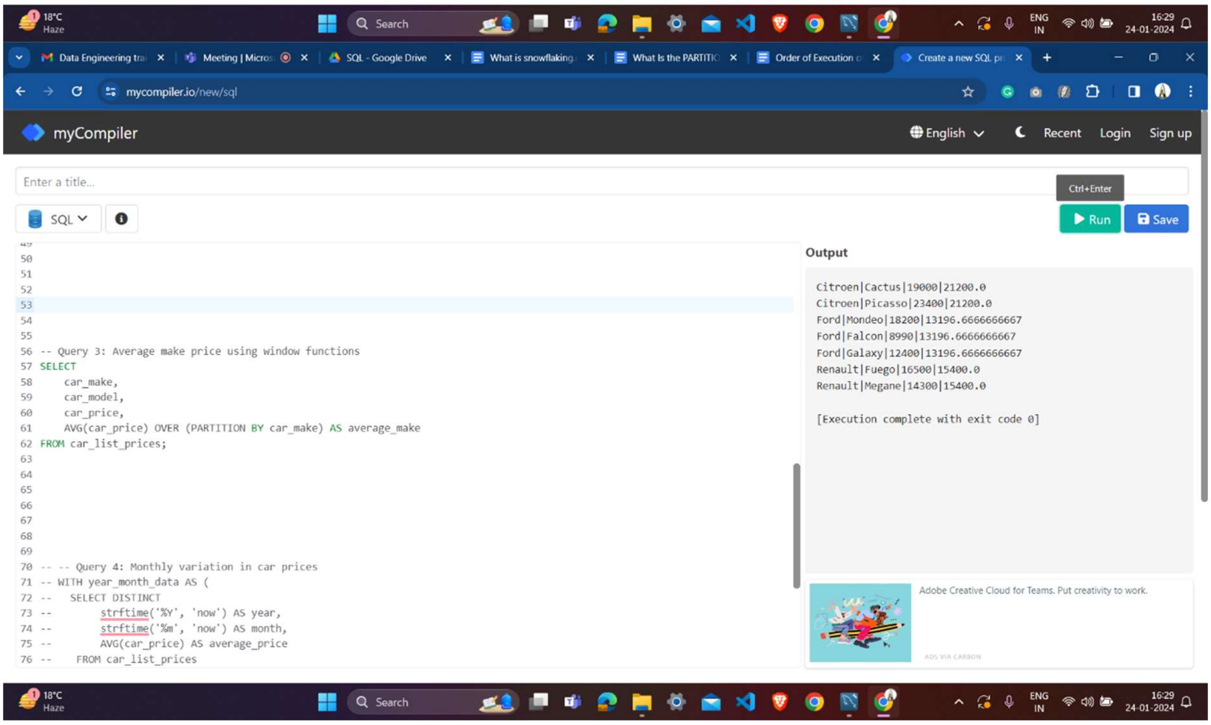
```sql
-- Create the car_list_prices table
CREATE TABLE IF NOT EXISTS car_list_prices (
    car_make VARCHAR(50),
    car_model VARCHAR(50),
    car_type VARCHAR(50),
    car_price DECIMAL(10, 2)
);

-- Insert data into the table
INSERT INTO car_list_prices VALUES
('Ford', 'Mondeo', 'premium', 18200),
('Renault', 'Fuego', 'sport', 16500),
('Citroen', 'Cactus', 'premium', 19000),
('Ford', 'Falcon', 'low cost', 8990),
('Ford', 'Galaxy', 'standard', 12400),
('Renault', 'Megane', 'standard', 14300),
('Citroen', 'Picasso', 'premium', 23400);

-- Query 1: Average price across all cars and average price by car type
SELECT
    car_make,
    car_model,
    car_price,
    AVG(car_price) OVER() AS "overall_average_price",
    AVG(car_price) OVER (PARTITION BY car_type) AS "car_type_average_price"
FROM car_list_prices;
```

```
Ford|Falcon|8990|16112.8571428571|8990.0
Ford|Mondeo|18200|16112.8571428571|20200.0
Citroen|Cactus|19000|16112.8571428571|20200.0
Citroen|Picasso|23400|16112.8571428571|20200.0
Renault|Fuego|16500|16112.8571428571|16500.0
Ford|Galaxy|12400|16112.8571428571|13350.0
Renault|Megane|14300|16112.8571428571|13350.0

[Execution complete with exit code 0]
```



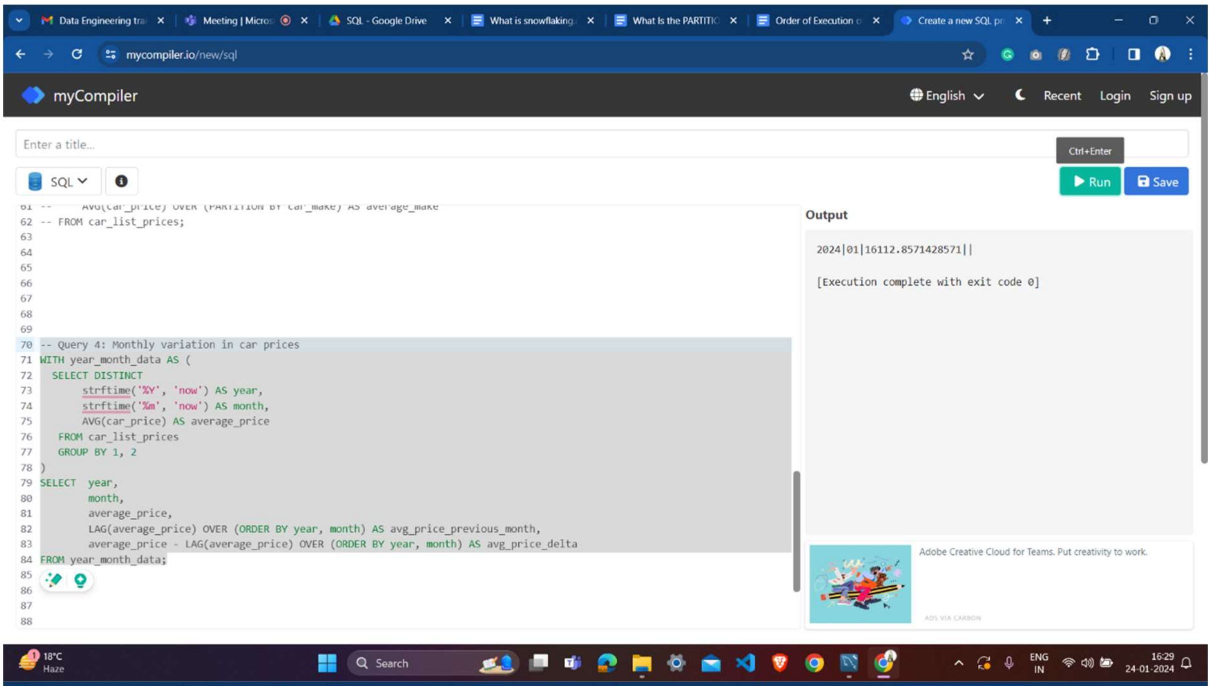
```sql
--     car_price,
--     AVG(car_price) OVER() AS "overall_average_price",
--     AVG(car_price) OVER (PARTITION BY car_type) AS "car_type_average_price"
-- FROM car_list_prices;





-- Query 2: Average price and top price per car make
SELECT
    car_make,
    AVG(car_price) AS average_price,
    MAX(car_price) AS top_price
FROM car_list_prices
GROUP BY car_make;
```

```
Citroen|21200.0|23400
Ford|13196.6666666667|18200
Renault|15400.0|16500

[Execution complete with exit code 0]
```



```sql
-- Query 3: Average make price using window functions
SELECT
    car_make,
    car_model,
    car_price,
    AVG(car_price) OVER (PARTITION BY car_make) AS average_make
FROM car_list_prices;







-- -- Query 4: Monthly variation in car prices
-- WITH year_month_data AS (
--   SELECT DISTINCT
--        strftime('%Y', 'now') AS year,
--        strftime('%m', 'now') AS month,
--        AVG(car_price) AS average_price
--    FROM car_list_prices
```

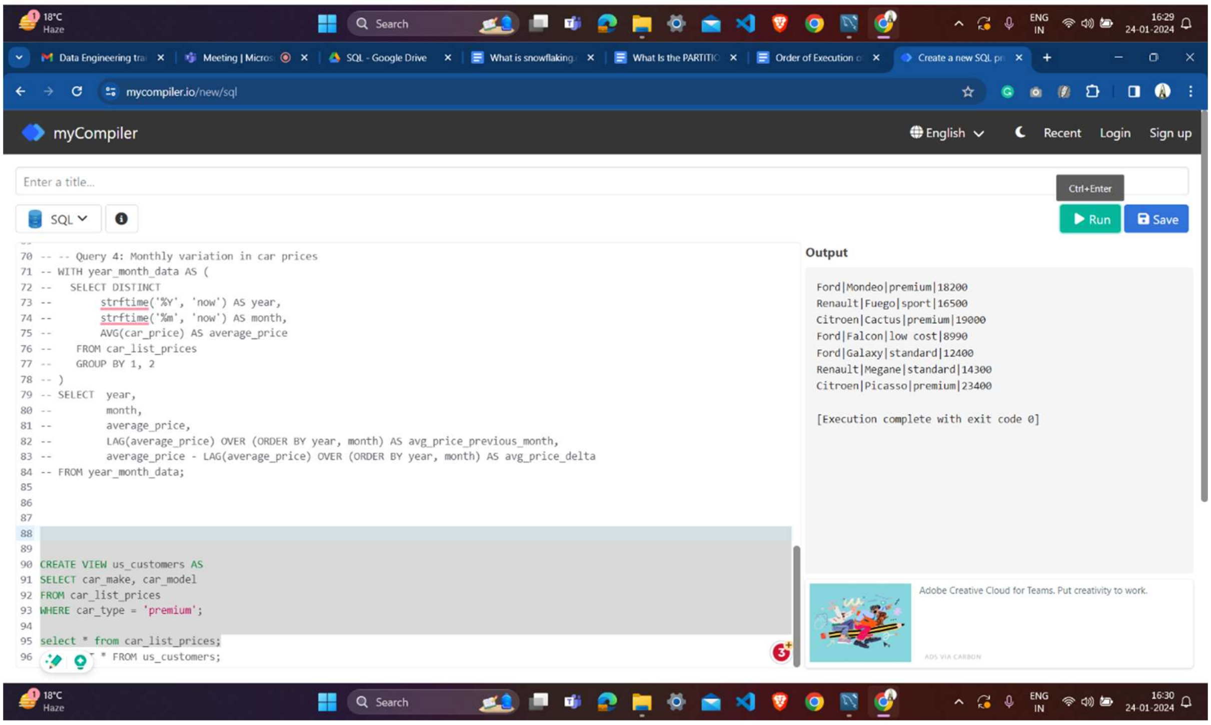
```
Citroen|Cactus|19000|21200.0
Citroen|Picasso|23400|21200.0
Ford|Mondeo|18200|13196.6666666667
Ford|Falcon|8990|13196.6666666667
Ford|Galaxy|12400|13196.6666666667
Renault|Fuego|16500|15400.0
Renault|Megane|14300|15400.0

[Execution complete with exit code 0]
```



```sql
--     AVG(car_price) OVER (PARTITION BY car_make) AS average_make
-- FROM car_list_prices;







-- Query 4: Monthly variation in car prices
WITH year_month_data AS (
  SELECT DISTINCT
       strftime('%Y', 'now') AS year,
       strftime('%m', 'now') AS month,
       AVG(car_price) AS average_price
   FROM car_list_prices
   GROUP BY 1, 2
)
SELECT  year,
        month,
        average_price,
        LAG(average_price) OVER (ORDER BY year, month) AS avg_price_previous_month,
        average_price - LAG(average_price) OVER (ORDER BY year, month) AS avg_price_delta
FROM year_month_data;
```

Output

```
2024|01|16112.8571428571||

[Execution complete with exit code 0]
```

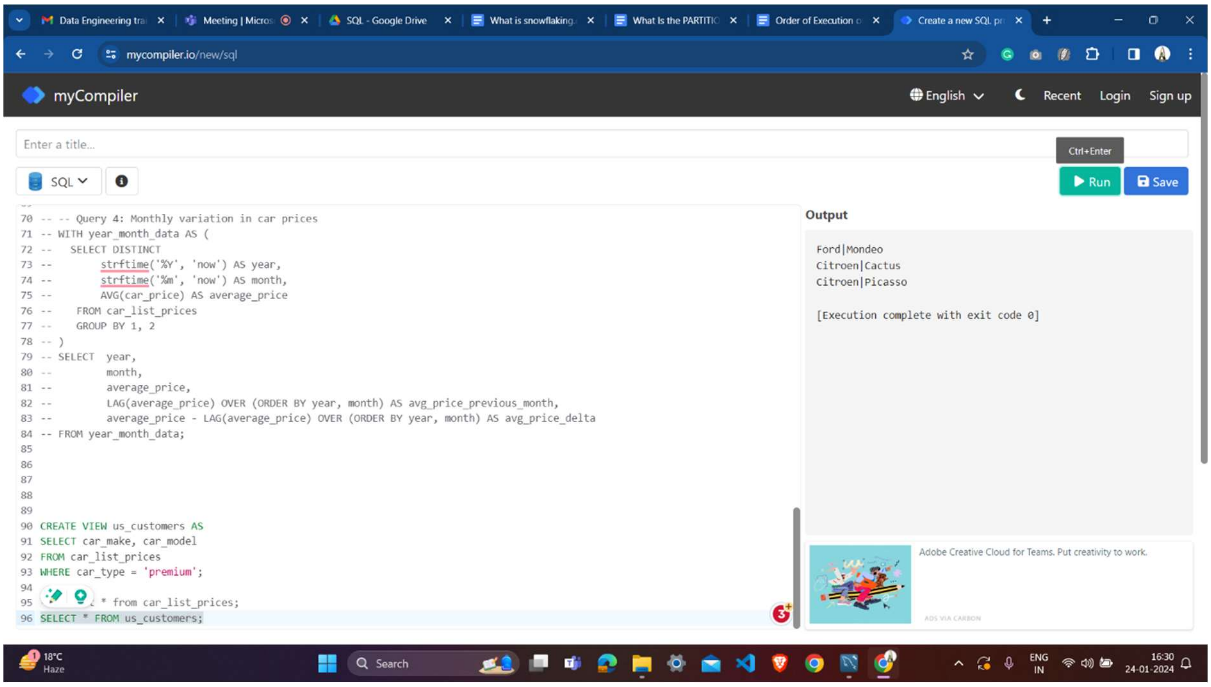


```sql
-- -- Query 4: Monthly variation in car prices
-- WITH year_month_data AS (
--   SELECT DISTINCT
--        strftime('%Y', 'now') AS year,
--        strftime('%m', 'now') AS month,
--        AVG(car_price) AS average_price
--    FROM car_list_prices
--    GROUP BY 1, 2
-- )
-- SELECT  year,
--         month,
--         average_price,
--         LAG(average_price) OVER (ORDER BY year, month) AS avg_price_previous_month,
--         average_price - LAG(average_price) OVER (ORDER BY year, month) AS avg_price_delta
-- FROM year_month_data;





CREATE VIEW us_customers AS
SELECT car_make, car_model
FROM car_list_prices
WHERE car_type = 'premium';

select * from car_list_prices;
      * FROM us_customers;
```

Output

```
Ford|Mondeo|premium|18200
Renault|Fuego|sport|16500
Citroen|Cactus|premium|19000
Ford|Falcon|low cost|8990
Ford|Galaxy|standard|12400
Renault|Megane|standard|14300
Citroen|Picasso|premium|23400

[Execution complete with exit code 0]
```

Material View in SQL syntax and example:



```sql
-- -- Query 4: Monthly variation in car prices
-- WITH year_month_data AS (
--   SELECT DISTINCT
--        strftime('%Y', 'now') AS year,
--        strftime('%m', 'now') AS month,
--        AVG(car_price) AS average_price
--    FROM car_list_prices
--    GROUP BY 1, 2
-- )
-- SELECT  year,
--         month,
--         average_price,
--         LAG(average_price) OVER (ORDER BY year, month) AS avg_price_previous_month,
--         average_price - LAG(average_price) OVER (ORDER BY year, month) AS avg_price_delta
-- FROM year_month_data;





CREATE VIEW us_customers AS
SELECT car_make, car_model
FROM car_list_prices
WHERE car_type = 'premium';

 * from car_list_prices;
SELECT * FROM us_customers;
```

Output

```
Ford|Mondeo
Citroen|Cactus
Citroen|Picasso

[Execution complete with exit code 0]
```