SQL-03 | Joins and Aggregation

Lecture Queries

Question: Analyze purchases made at the farmer’s market on days when it rained.

Question: Analyze purchases made at the farmer's market on days when it rained.

```
SELECT
    market_date,
    customer_id,
    vendor_id,
    quantity * cost_to_customer_per_qty price
FROM farmers_market.customer_purchases
WHERE
    market_date IN
        (
        SELECT market_date
        FROM farmers_market.market_date_info
        WHERE market_rain_flag = 1
        ) LIMIT 5
```

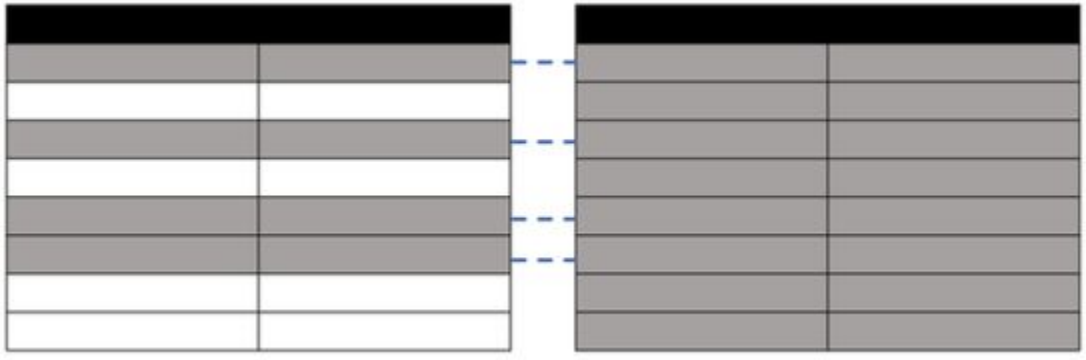
Question: Find out which vendors primarily sell fresh produce and which don’t.

Question: Find out which vendors primarily sell fresh produce and which don’t.

```
SELECT
    vendor_id,
    vendor_name,
    vendor_type,
    CASE
        WHEN LOWER(vendor_type) LIKE '%fresh%'
          THEN 'Fresh Produce'
        ELSE 'Other'
    END AS vendor_type_condensed
 FROM farmers_market.vendor
```

Question: Put the total cost to customer purchases into bins of

- under $5.00,
- $5.00–$9.99,
- $10.00–$19.99, or
- $20.00 and over



Question: Put the total cost to customer purchases into bins of

- under $5.00,
- $5.00–$9.99,
- $10.00–$19.99, or
- $20.00 and over.

```
SELECT
        market_date,
        customer_id,
        vendor_id,
        ROUND(quantity * cost_to_customer_per_qty, 2) AS price,
        CASE
            WHEN quantity * cost_to_customer_per_qty < 5.00
                THEN 'Under $5'
            WHEN quantity * cost_to_customer_per_qty < 10.00
                THEN '$5-$9.99'
            WHEN quantity * cost_to_customer_per_qty < 20.00
                THEN '$10-$19.99'
            WHEN quantity * cost_to_customer_per_qty >= 20.00
                THEN '$20 and Up'
            END AS price_bin
    FROM farmers_market.customer_purchases
    LIMIT 10
```

Question: Let’s say we wanted to list each product name along with its product category name.

Question: Let’s say we wanted to list each product name along with its product category name.

```
SELECT
        p.product_id,
        p.product_name,
        pc.product_category_id,
        pc.product_category_name
    FROM product AS p
        LEFT JOIN product_category AS pc
            ON p.product_category_id = pc.product_category_id
    ORDER BY pc.product_category_name, p.product_name
```

Question: Get all the Customers who have not purchased anything from the market yet.

Question: Get all the Customers who have not purchased nothing from the market yet.

```
SELECT c.*  # select columns from
customer table only
  FROM customer AS c
  LEFT JOIN customer_purchases AS cp
      ON c.customer_id = cp.customer_id
  WHERE cp.customer_id IS NULL
```

Question: List all the customers and their associated purchases?

Question: List all the customers and their associated purchases?

```
SELECT *
  FROM customer AS c
  RIGHT JOIN customer_purchases AS cp
     ON c.customer_id = cp.customer_id
```

Question: Let’s say we want to write a query that returns a list of all customers who did not make a purchase on March 2, 2019.

Question: Let's say we want to write a query that returns a list of all customers who did not make a purchase on March 2, 2019.

```
SELECT c.*, cp.market_date
  FROM customer AS c
  LEFT JOIN customer_purchases AS cp
      ON c.customer_id = cp.customer_id
  WHERE cp.market_date <> '2019-03-02'
```

Two problems with the output:

1. Some rows/ customers are missing because the market_date is NULL.
2. We are getting multiple rows for each customer which is not required.

solution

```
SELECT DISTINCT c.*
  FROM customer AS c
  LEFT JOIN customer_purchases AS cp
      ON c.customer_id = cp.customer_id
  WHERE (cp.market_date <> '2019-03-02' OR cp.market_date IS NULL)
```

Question: Let’s say we want details about all farmer’s market booths, as well as every vendor booth assignment for every market date along with the vendor details.

Question: Let’s say we want details about all farmer’s market booths, as well as every vendor booth assignment for every market date along with the vendor details.

```
SELECT
    b.booth_number,
    b.booth_type,
    vba.market_date,
    v.vendor_id,
    v.vendor_name,
    v.vendor_type
 FROM booth AS b
    LEFT JOIN vendor_booth_assignments AS vba ON b.booth_number = vba.
 booth_number
    LEFT JOIN vendor AS v ON v.vendor_id = vba.vendor_id
 ORDER BY b.booth_number, vba.market_date
```