Practical Exam: Grocery Store Sales

FoodYum is a grocery store chain that is based in the United States.

Food Yum sells items such as produce, meat, dairy, baked goods, snacks, and other household food staples.

As food costs rise, FoodYum wants to make sure it keeps stocking products in all categories that cover a range of prices to ensure they have stock for a broad range of customers.

Data

The data is available in the table `products`.

The dataset contains records of customers for their last full year of the loyalty program.

Column Name	Criteria
product_id	Nominal. The unique identifier of the product. Missing values are not possible due to the database structure.
product_type	Nominal. The product category type of the product, one of 5 values (Produce, Meat, Dairy, Bakery, Snacks). Missing values should be replaced with “Unknown”.
brand	Nominal. The brand of the product. One of 7 possible values. Missing values should be replaced with “Unknown”.
weight	Continuous. The weight of the product in grams. This can be any positive value, rounded to 2 decimal places. Missing values should be replaced with the overall median weight.
price	Continuous. The price the product is sold at, in US dollars. This can be any positive value, rounded to 2 decimal places. Missing values should be replaced with the overall median price.
average_units_sold	Discrete. The average number of units sold each month. This can be any positive integer value. Missing values should be replaced with 0.
year_added	Nominal. The year the product was first added to FoodYum stock. Missing values should be replaced with 2022.
stock_location	Nominal. The location that stock originates. This can be one of four warehouse locations, A, B, C or D Missing values should be replaced with “Unknown”.

Task 1

Last year (2022) there was a bug in the product system. For some products that were added in that year, the `year_added` value was not set in the data. As the year the product was added may have an impact on the price of the product, this is important information to have.

Write a query to determine how many products have the `year_added` value missing. Your output should be a single column, `missing_year`, with a single row giving the number of missing values.

Unknown integration DataFrame as `missing_year`

```
-- Write your query for task 1 in this cell
SELECT COUNT(*) AS missing_year
FROM products
WHERE year_added IS NULL;
```

	missing_year
0	

1 rows

Task 2

Given what you know about the year added data, you need to make sure all of the data is clean before you start your analysis. The table below shows what the data should look like.

Write a query to ensure the product data matches the description provided. Do not update the original table.

Column Name	Criteria
product_id	Nominal. The unique identifier of the product. Missing values are not possible due to the database structure.
product_type	Nominal. The product category type of the product, one of 5 values (Produce, Meat, Dairy, Bakery, Snacks). Missing values should be replaced with “Unknown”.
brand	Nominal. The brand of the product. One of 7 possible values. Missing values should be replaced with “Unknown”.
weight	Continuous. The weight of the product in grams. This can be any positive value, rounded to 2 decimal places. Missing values should be replaced with the overall median weight.
price	Continuous. The price the product is sold at, in US dollars. This can be any positive value, rounded to 2 decimal places. Missing values should be replaced with the overall median price.
average_units_sold	Discrete. The average number of units sold each month. This can be any positive integer value. Missing values should be replaced with 0.
year_added	Nominal. The year the product was first added to FoodYum stock. Missing values should be replaced with last year (2022).

Column Name	Criteria
stock_location	Nominal. The location that stock originates. This can be one of four warehouse locations, A, B, C or D Missing values should be replaced with “Unknown”.

Unknown integration DataFrame as clean_data

```sql
-- Write your query for task 2 in this cell
SELECT
    product_id,
    COALESCE(product_type, 'Unknown') AS product_type,
    COALESCE(NULLIF(REPLACE(brand, '-', ''), ''), 'Unknown') AS brand,
    COALESCE(ROUND(CAST(REGEXP_REPLACE(weight, '[^\d.]', '', 'g') AS DECIMAL(10, 2)), 2),
ROUND((SELECT PERCENTILE_DISC(0.5) WITHIN GROUP (ORDER BY CAST(REGEXP_REPLACE(weight,
'[^\d.]', '', 'g') AS DECIMAL(10, 2))) FROM products), 2)) AS weight,

COALESCE(
    TO_CHAR(CAST(price AS DECIMAL(10, 2)), '9999999999.99'),
    TO_CHAR(CAST((SELECT PERCENTILE_DISC(0.5) WITHIN GROUP (ORDER BY price) FROM products)
AS DECIMAL(10, 2)), '9999999999.99')
) AS price,

    COALESCE(average_units_sold, 0) AS average_units_sold,
    COALESCE(year_added, 2022) AS year_added,
    COALESCE(UPPER(stock_location), 'Unknown') AS stock_location
FROM products;
```

	product_id	product_type	brand	weight
0	1	Bakery	TopBrand	
1	2	Produce	SilverLake	
2	3	Produce	TastyTreat	
3	4	Bakery	StandardYums	
4	5	Produce	GoldTree	
5	6	Meat	TopBrand	
6	7	Produce	GoldTree	
7	8	Meat	SilverLake	
8	9	Meat	StandardYums	
9	10	Meat	TastyTreat	
10	11	Dairy	StandardYums	
11	12	Bakery	StandardYums	
12	13	Snacks	SmoothTaste	
13	14	Meat	StandardYums	
14	15	Produce	SilverLake	

1,700 rows

Task 3

To find out how the range varies for each product type, your manager has asked you to determine the minimum and maximum values for each product type.

Write a query to return the `product_type`, `min_price` and `max_price` columns.

Unknown integration DataFrame as `m`

```sql
-- Write your query for task 3 in this cell
SELECT product_type,
       MIN(price) AS min_price,
       MAX(price) AS max_price
FROM products
GROUP BY product_type;
```

	product_type	min_

Task 4

The team want to look in more detail at meat and dairy products where the average units sold was greater than ten.

Write a query to return the `product_id`, `price` and `average_units_sold` of the rows of interest to the team.

5 rows

Unknown integration DataFrame as a

```
-- Write your query for task 4 in this cell
SELECT product_id, price, average_units_sold
FROM products
WHERE product_type IN ('Meat', 'Dairy')
      AND average_units_sold > 10;
```