

Advanced Databases  
Section A  
Group Y  
Food Inventory Data Mart

Eoin Fitzsimons- x23151374  
Conor Judge- x22165398

# FOOD INVENTORY

## Eoin Fitzsimons (Profit) – Conor Judge (Perish)

Introduction.....	2
Business Requirements.....	3
Food Waste .....	<b>Error! Bookmark not defined.</b>
Profits .....	3
Dimensional Model.....	4
SQL – Fact & Dimension Tables.....	5
SQL – Data into Mart. ....	6
Queries .....	10
Food Waste. ....	10
Profit Per Ingredient. ....	11
Changes made to optimise read access.....	14
References.....	15

## Introduction

When faced with the task of designing a data mart for a business within food inventory we were unsure which of the many options was best to pick; we ended up settling on a restaurant with the onus of the mart being the stockroom of individual ingredients. Two thoughts immediately emerged when deciding upon the business requirements; perishability and profit. We did not find a ready-made database that suited our needs enough, so endeavoured to make our own. We started with what information we would like to obtain from our queries and then worked backwards to flesh out the table and make it appear more realistic akin to how a genuine restaurant would keep track of its produce.

We devised an artisanal culinary experience that stays true to the roots of our vegetables and wants to be carbon-positive. All our food is prepared without electricity, and we only refrigerate and freeze a minuscule amount of ingredients to have the lowest emissions. We have a direct link to the local vegetable farmers and even have our plots within the restaurant to provide that mesmerising organic dining moment™.

We assume this is popular and not a complete waste of time.

# Business Requirements.

## Perishability

A key issue for any business that deals with food will be managing waste. Spoiled items cannot be used or sold and have another further cost in the form of waste disposal. Restaurants must invest in facilities to store the food in appropriate environments to prevent waste.

Being able to identify items that are most and least likely to spoil can be used to change buying and storage patterns.

Storage areas where a significant amount of waste occurs can be identified to adjust storage conditions, implementing better inventory management practices, or reevaluating the allocation of ingredients to different storage areas.

## Profits

All businesses need to make money to run. A restaurant is no exception, our restaurant needs to purchase ingredients and then the meals will make use of ingredients and sell to customers.

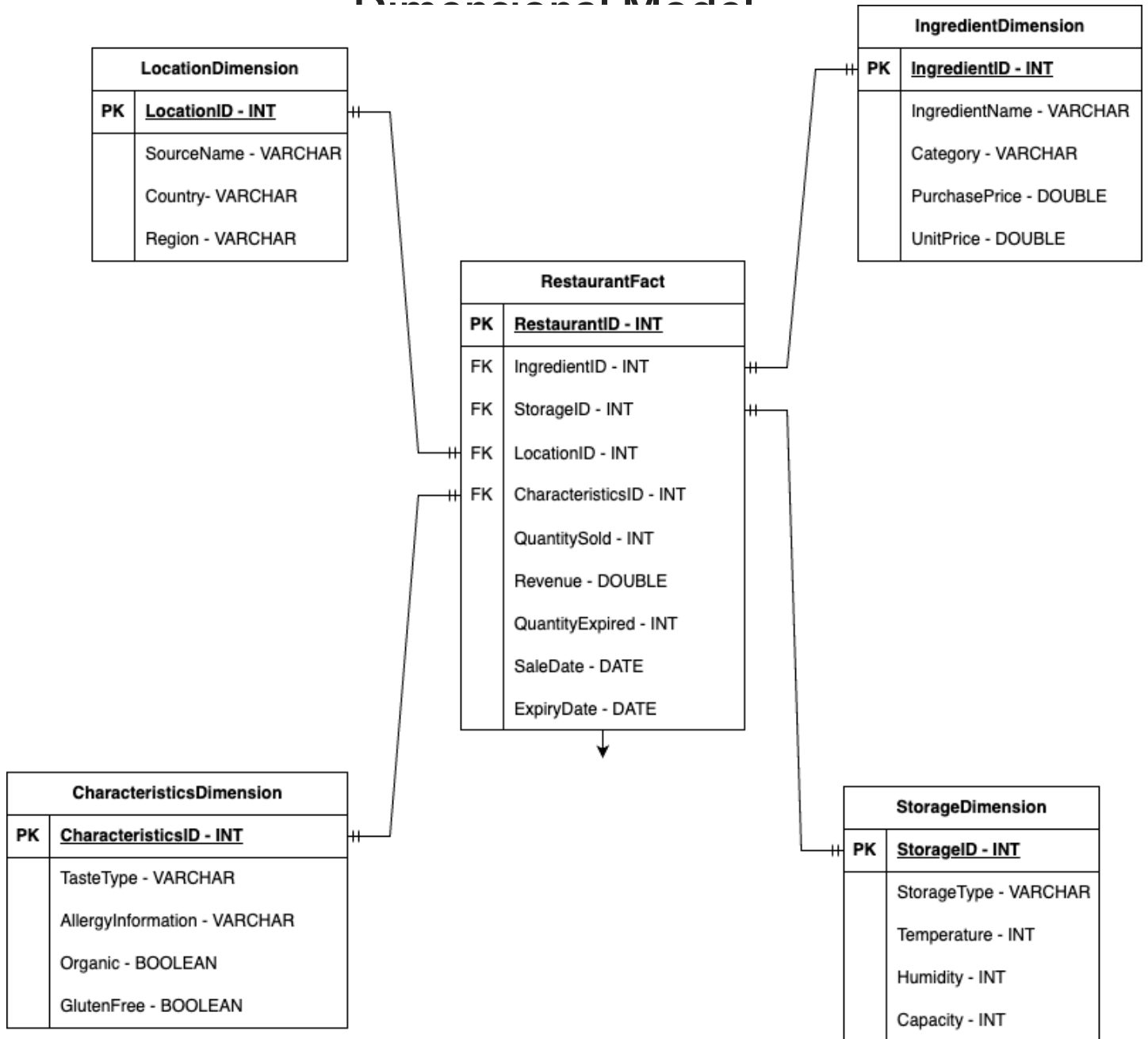
The inventory manager for a restaurant may want to observe multiple dimensions to examine why ingredients are profitable. They should see whether they ought to raise prices or cut costs and obtain the ingredients differently.

To do this they could examine the ingredient dimension to see what produce is bought and sold. Then take note of how much is spent in procurement, and how much an ingredient brings in when used in one of the restaurant's wonderful dishes.

They could also see which country the produce has arrived from to inspect whether the distance the product travels has proportionality with the profit margin. Is the premium on local artisanal goods worth it when compared to mass-produced goods from elsewhere?

Then the inventory manager would examine the characteristics of the ingredients to see which taste has the most profound impact on the profit margin. Is it those sweet ingredients that you only need a minuscule amount of to tie a dessert together? Or should the restaurant focus on those foundational vegetables of society that grow in abundance and are bought at dirt-cheap prices?

# Dimensional Model



# SQL – Fact & Dimension Tables.

```
1  -- Delete Database
2  ● DROP DATABASE IF EXISTS Restaurant;
3  -- Create Database
4  ● CREATE DATABASE IF NOT EXISTS Restaurant;
5
6  -- Switch to the Restaurant Database
7  ● USE Restaurant;
8
9  -- Create Dimension Tables
10 ● CREATE TABLE IngredientDimension (
11     IngredientID INT PRIMARY KEY,
12     IngredientName VARCHAR(50),
13     Category VARCHAR(30),
14     UnitPrice DECIMAL(10, 2),
15     PurchasePrice DECIMAL(10, 2)
16 );
17
18 ● CREATE TABLE StorageDimension (
19     StorageID INT PRIMARY KEY,
20     StorageType VARCHAR(30),
21     Temperature VARCHAR(20),
22     Humidity VARCHAR(20),
23     Capacity DECIMAL(10, 2)
24 );
25
26 ● CREATE TABLE LocationDimension (
27     LocationID INT PRIMARY KEY,
28     SourceName VARCHAR(50),
29     Country VARCHAR(50),
30     Region VARCHAR(50)
31 );
32
33 ● CREATE TABLE CharacteristicsDimension (
34     CharacteristicsID INT PRIMARY KEY,
35     TasteProfile VARCHAR(100),
36     AllergyInformation VARCHAR(100),
37     Organic BOOLEAN,
38     GlutenFree BOOLEAN
39 );
40
41 -- Create Fact Table
42 ● CREATE TABLE RestaurantFact (
43     RestaurantID INT PRIMARY KEY,
44     IngredientID INT,
45     QuantityBought INT,
46     QuantitySold INT,
47     Revenue DECIMAL(10, 2),
48     QuantityExpired INT,
49     StorageID INT,
50     CharacteristicsID INT,
51     LocationID INT,
52     SaleDate DATE,
53     WastageDate DATE,
54     FOREIGN KEY (IngredientID) REFERENCES IngredientDimension(IngredientID),
55     FOREIGN KEY (StorageID) REFERENCES StorageDimension(StorageID),
56     FOREIGN KEY (CharacteristicsID) REFERENCES CharacteristicsDimension(CharacteristicsID),
57     FOREIGN KEY (LocationID) REFERENCES LocationDimension(LocationID)
58 );
59
60 -- Indexes
61 -- Index for IngredientID in RestaurantFact table (for JOIN operations)
62 ● CREATE INDEX idx_IngredientID ON RestaurantFact (IngredientID);
63
64 -- Index for SaleDate in RestaurantFact table (for WHERE clause)
65 ● CREATE INDEX idx_SaleDate ON RestaurantFact (SaleDate);
66
67 -- Index for WastageDate in RestaurantFact table (for WHERE clause)
68 ● CREATE INDEX idx_WastageDate ON RestaurantFact (WastageDate);
69
70 -- Index for QuantityExpired in RestaurantFact table (for Food Waste query)
71 ● CREATE INDEX idx_QuantityExpired ON RestaurantFact (QuantityExpired);
72
```

## **SQL – Data into Mart.**

```

1  -- Insert data into IngredientDimension
2  ● INSERT INTO IngredientDimension (IngredientID, IngredientName, Category, UnitPrice, PurchasePrice)
3  VALUES
4  (1, 'Potatoes', 'Vegetables', 1.00, 0.80),
5  (2, 'Lamb', 'Meat', 8.50, 7.00),
6  (3, 'Carrots', 'Vegetables', 0.80, 0.60),
7  (4, 'Butter', 'Dairy', 3.00, 2.50),
8  (5, 'Salmon', 'Seafood', 10.00, 8.00),
9  (6, 'Mushrooms', 'Vegetables', 2.50, 2.00),
10 (7, 'Oats', 'Grains', 1.20, 1.00),
11 (8, 'Beef', 'Meat', 9.00, 7.50),
12 (9, 'Apples', 'Fruits', 1.50, 1.20),
13 (10, 'Cheese', 'Dairy', 4.00, 3.50),
14 (11, 'Honey', 'Condiments', 5.00, 4.00),
15 (12, 'Eggs', 'Dairy', 0.90, 0.70),
16 (13, 'Spinach', 'Vegetables', 2.20, 1.80),
17 (14, 'Barley', 'Grains', 1.80, 1.50),
18 (15, 'Lettuce', 'Vegetables', 2.00, 1.50),
19 (16, 'Sweet Potatoes', 'Vegetables', 1.20, 1.00),
20 (17, 'Chicken', 'Meat', 7.50, 6.00),
21 (18, 'Broccoli', 'Vegetables', 1.00, 0.80),
22 (19, 'Cream', 'Dairy', 2.50, 2.00),
23 (20, 'Shrimp', 'Seafood', 12.00, 10.00),
24 (21, 'Onions', 'Vegetables', 1.50, 1.20),
25 (22, 'Quinoa', 'Grains', 2.00, 1.80),
26 (23, 'Pork', 'Meat', 8.00, 6.50),
27 (24, 'Oranges', 'Fruits', 1.80, 1.50),
28 (25, 'Cheddar', 'Dairy', 4.50, 3.80),
29 (26, 'Mustard', 'Condiments', 2.50, 2.00),
30 (27, 'Whole Milk', 'Dairy', 1.00, 0.90),
31 (28, 'Kale', 'Vegetables', 2.00, 1.80),
32 (29, 'Rice', 'Grains', 1.50, 1.20),
33 (30, 'Tomatoes', 'Vegetables', 1.80, 1.50);
34
35 -- Insert data into StorageDimension
36 ● INSERT INTO StorageDimension (StorageID, StorageType, Temperature, Humidity, Capacity)
37 VALUES
38 (1, 'Pantry', 'Room Temp', 'Normal', 500.00),
39 (2, 'Freezer', '-18°C', 'Low', 300.00),
40 (3, 'Refrigerator', '4°C', 'Medium', 200.00),
41 (4, 'Dry Storage', 'Room Temp', 'Low', 700.00),
42 (5, 'Cool Room', '12°C', 'High', 400.00),
43 (6, 'Freezer', '-20°C', 'Low', 250.00),
44 (7, 'Refrigerator', '5°C', 'Medium', 180.00),
45 (8, 'Pantry', 'Room Temp', 'Normal', 550.00),
46 (9, 'Dry Storage', 'Room Temp', 'Low', 600.00),
47 (10, 'Cool Room', '10°C', 'High', 350.00),

```

```

47      (10, 'Cool Room', '10°C', 'High', 350.00),
48      (11, 'Freezer', '-15°C', 'Low', 280.00),
49      (12, 'Refrigerator', '3°C', 'Medium', 220.00),
50      (13, 'Dry Storage', 'Room Temp', 'Low', 800.00),
51      (14, 'Pantry', 'Room Temp', 'Normal', 450.00),
52      (15, 'Cool Room', '11°C', 'High', 320.00),
53      (16, 'Pantry', 'Room Temp', 'Normal', 450.00),
54      (17, 'Freezer', '-20°C', 'Low', 320.00),
55      (18, 'Refrigerator', '5°C', 'Medium', 180.00),
56      (19, 'Dry Storage', 'Room Temp', 'Low', 600.00),
57      (20, 'Cool Room', '10°C', 'High', 350.00),
58      (21, 'Freezer', '-15°C', 'Low', 280.00),
59      (22, 'Refrigerator', '3°C', 'Medium', 220.00),
60      (23, 'Pantry', 'Room Temp', 'Normal', 500.00),
61      (24, 'Dry Storage', 'Room Temp', 'Low', 700.00),
62      (25, 'Cool Room', '12°C', 'High', 400.00),
63      (26, 'Freezer', '-18°C', 'Low', 250.00),
64      (27, 'Refrigerator', '4°C', 'Medium', 200.00),
65      (28, 'Dry Storage', 'Room Temp', 'Low', 800.00),
66      (29, 'Pantry', 'Room Temp', 'Normal', 550.00),
67      (30, 'Cool Room', '11°C', 'High', 320.00);
68
69  -- Insert data into LocationDimension
70  ● INSERT INTO LocationDimension (LocationID, SourceName, Country, Region)
71  VALUES
72      (1, 'Local Farmer', 'Ireland', 'Leinster'),
73      (2, 'Irish Seafood Co-op', 'Ireland', 'Munster'),
74      (3, 'Dairy Co-op', 'Ireland', 'Connaught'),
75      (4, 'Green Vegetable Farms', 'Ireland', 'Ulster'),
76      (5, 'British Meat Supplier', 'United Kingdom', 'Various'),
77      (6, 'French Cheese Distributor', 'France', 'Various'),
78      (7, 'Spanish Fruit Co-op', 'Spain', 'Various'),
79      (8, 'Italian Olive Oil Supplier', 'Italy', 'Various'),
80      (9, 'Dutch Dairy Co-op', 'Netherlands', 'Various'),
81      (10, 'Scottish Honey Producer', 'United Kingdom', 'Various'),
82      (11, 'German Egg Farm', 'Germany', 'Various'),
83      (12, 'Belgian Mushroom Co-op', 'Belgium', 'Various'),
84      (13, 'Norwegian Salmon Fisheries', 'Norway', 'Various'),
85      (14, 'Swedish Barley Farm', 'Sweden', 'Various'),
86      (15, 'Danish Lettuce Grower', 'Denmark', 'Various'),
87      (16, 'Scottish Potato Farms', 'United Kingdom', 'Various'),
88      (17, 'New Zealand Lamb Co-op', 'New Zealand', 'Various'),
89      (18, 'Caribbean Roots Farms', 'Jamaica', 'Various'),
90      (19, 'Irish Dairy Collective', 'Ireland', 'Munster'),
91      (20, 'Japanese Seafood Importers', 'Japan', 'Various'),
92      (21, 'Brazilian Mushroom Co-op', 'Brazil', 'Various'),
93      (22, 'Canadian Oats Distributors', 'Canada', 'Various'),

```



```

93 (22, 'Canadian Oats Distributors', 'Canada', 'Various'),
94 (23, 'Australian Beef Producers', 'Australia', 'Various'),
95 (24, 'South African Apple Orchards', 'South Africa', 'Various'),
96 (25, 'Swiss Cheese Importers', 'Switzerland', 'Various'),
97 (26, 'Indian Honey Farms', 'India', 'Various'),
98 (27, 'Greek Egg Producers', 'Greece', 'Various'),
99 (28, 'Mexican Spinach Farms', 'Mexico', 'Various'),
100 (29, 'Argentinian Barley Fields', 'Argentina', 'Various'),
101 (30, 'Chinese Lettuce Growers', 'China', 'Various');
102
103 -- Insert data into CharacteristicsDimension
104 • INSERT INTO CharacteristicsDimension (CharacteristicsID, TasteProfile, AllergyInformation, Organic, GlutenFree)
105 VALUES
106 (1, 'Starchy', 'None', TRUE, TRUE),
107 (2, 'Savory', 'None', TRUE, FALSE),
108 (3, 'Sweet', 'None', TRUE, TRUE),
109 (4, 'Rich', 'Lactose Intolerance', TRUE, FALSE),
110 (5, 'Umami', 'Fish Allergy', TRUE, TRUE),
111 (6, 'Earthy', 'None', TRUE, TRUE),
112 (7, 'Nutty', 'Gluten Intolerance', TRUE, TRUE),
113 (8, 'Savory', 'None', TRUE, FALSE),
114 (9, 'Sweet', 'None', TRUE, TRUE),
115 (10, 'Creamy', 'Lactose Intolerance', TRUE, FALSE),
116 (11, 'Sweet', 'None', TRUE, TRUE),
117 (12, 'Protein-Rich', 'None', TRUE, TRUE),
118 (13, 'Leafy', 'Oxalate Sensitivity', TRUE, TRUE),
119 (14, 'Nutty', 'Gluten Intolerance', TRUE, TRUE),
120 (15, 'Crisp', 'None', TRUE, TRUE),
121 (16, 'Sweet', 'None', TRUE, TRUE),
122 (17, 'Lean', 'None', TRUE, FALSE),
123 (18, 'Fresh', 'None', TRUE, TRUE),
124 (19, 'Creamy', 'Lactose Intolerance', TRUE, FALSE),
125 (20, 'Delicate', 'Shellfish Allergy', TRUE, TRUE),
126 (21, 'Aromatic', 'None', TRUE, TRUE),
127 (22, 'Nutrient-Rich', 'Gluten Intolerance', TRUE, TRUE),
128 (23, 'Juicy', 'None', TRUE, FALSE),
129 (24, 'Citrusy', 'None', TRUE, TRUE),
130 (25, 'Sharp', 'Lactose Intolerance', TRUE, FALSE),
131 (26, 'Spicy', 'Mustard Allergy', TRUE, TRUE),
132 (27, 'Creamy', 'Lactose Intolerance', TRUE, FALSE),
133 (28, 'Leafy', 'Oxalate Sensitivity', TRUE, TRUE),
134 (29, 'Versatile', 'Gluten Intolerance', TRUE, TRUE),
135 (30, 'Crunchy', 'None', TRUE, TRUE);
136
137 -- Insert data into RestaurantFact
138 • INSERT INTO RestaurantFact (RestaurantID, IngredientID, QuantityBought, QuantitySold, Revenue, QuantityExpired, StorageID, CharacteristicsID, LocationID, SaleDate, WastageDate)
139 VALUES
140 (1, 1, 200, 180, 180.00, 20, 1, 1, 1, '2023-01-15', '2023-01-20'),
141 (2, 2, 30, 25, 212.50, 5, 2, 5, 5, '2023-02-10', '2023-02-15'),
142 (3, 3, 150, 140, 112.00, 10, 3, 7, 8, '2023-03-05', '2023-03-10'),
143 (4, 4, 50, 45, 135.00, 5, 4, 8, 10, '2023-04-20', '2023-04-25'),
144 (5, 5, 80, 75, 150.00, 5, 5, 11, 2, '2023-05-12', '2023-05-17'),
145 (6, 6, 100, 90, 225.00, 10, 6, 13, 12, '2023-06-08', '2023-06-13'),
146 (7, 7, 120, 110, 144.00, 10, 7, 14, 7, '2023-07-25', '2023-07-30'),
147 (8, 8, 60, 55, 270.00, 5, 8, 1, 14, '2023-08-17', '2023-08-22'),
148 (9, 9, 40, 35, 60.00, 5, 9, 10, 3, '2023-09-03', '2023-09-08'),
149 (10, 10, 110, 100, 140.00, 10, 10, 4, 9, '2023-10-19', '2023-10-24'),
150 (11, 11, 65, 60, 200.00, 5, 11, 8, 11, '2023-11-14', '2023-11-19'),
151 (12, 12, 95, 90, 105.00, 5, 12, 2, 7, '2023-12-02', '2023-12-07'),
152 (13, 13, 55, 50, 120.00, 5, 13, 3, 13, '2024-01-08', '2024-01-13'),
153 (14, 14, 70, 65, 165.00, 5, 14, 9, 15, '2024-02-14', '2024-02-19'),
154 (15, 15, 85, 80, 180.00, 5, 15, 12, 9, '2024-03-21', '2024-03-26'),
155 (16, 16, 120, 110, 132.00, 10, 16, 16, 16, '2024-04-02', '2024-04-07'),
156 (17, 17, 40, 35, 262.50, 5, 17, 17, 17, '2024-05-19', '2024-05-24'),
157 (18, 18, 90, 80, 72.00, 10, 18, 18, 18, '2024-06-14', '2024-06-19'),
158 (19, 19, 30, 25, 75.00, 5, 19, 19, 19, '2024-07-02', '2024-07-07'),
159 (20, 20, 70, 65, 165.00, 5, 20, 20, 20, '2024-08-17', '2024-08-22'),
160 (21, 21, 80, 70, 180.00, 10, 21, 21, 21, '2024-09-03', '2024-09-08'),
161 (22, 22, 110, 100, 132.00, 10, 22, 22, 22, '2024-10-19', '2024-10-24'),
162 (23, 23, 65, 60, 180.00, 5, 23, 23, 23, '2024-11-14', '2024-11-19'),
163 (24, 24, 95, 90, 96.00, 5, 24, 24, 24, '2024-12-02', '2024-12-07'),
164 (25, 25, 55, 50, 135.00, 5, 25, 25, 25, '2025-01-08', '2025-01-13'),
165 (26, 26, 70, 65, 180.00, 5, 26, 26, 26, '2025-02-14', '2025-02-19'),
166 (27, 27, 85, 80, 120.00, 5, 27, 27, 27, '2025-03-21', '2025-03-26'),
167 (28, 28, 40, 35, 90.00, 5, 28, 28, 28, '2025-04-08', '2025-04-13'),
168 (29, 29, 60, 55, 150.00, 5, 29, 29, 29, '2025-05-15', '2025-05-20'),
169 (30, 30, 75, 70, 165.00, 5, 30, 30, 30, '2025-06-22', '2025-06-27');

```

# Queries

## Food Waste.

-- Food Waste

SELECT

IngredientName,

SUM(QuantityExpired) AS TotalWaste

FROM RestaurantFact

JOIN IngredientDimension ON RestaurantFact.IngredientID = IngredientDimension.IngredientID

WHERE WastageDate BETWEEN '2000-01-01' AND '3024-01-11'

GROUP BY IngredientName;

## Profit Per Ingredient.

```
-- Profit Per Ingredient
SELECT
    IngredientName,
    Category,
    SUM(UnitPrice - PurchasePrice) AS Profit
FROM RestaurantFact
JOIN IngredientDimension ON RestaurantFact.IngredientID = IngredientDimension.IngredientID
WHERE SaleDate BETWEEN '2000-01-01' AND '3000-01-01'
GROUP BY IngredientName;
```

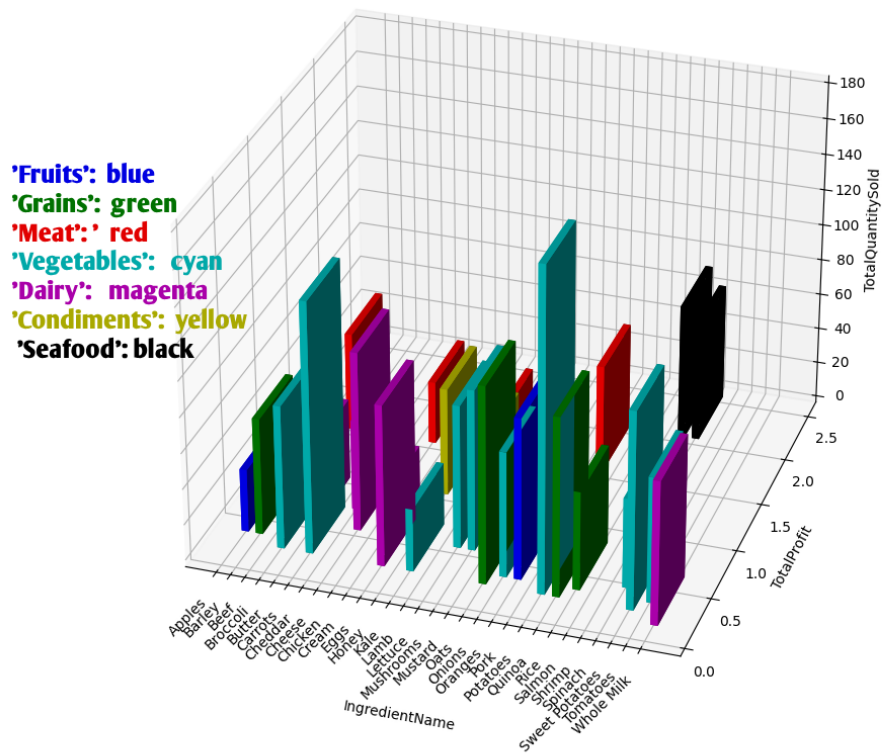
```
Help  ← → Search

untitled.py X Settings
C:\Users\> eoin0 > untitled.py > ...
1 import matplotlib.pyplot as plt
2 from mpl_toolkits.mplot3d import Axes3D
3 import numpy as np
4 import mysql.connector
5
6 # connect to database
7 connection = mysql.connector.connect(
8     host="localhost",
9     user="root",
10    database="Restaurant"
11 )
12
13 # Create a cursor to execute SQL queries
14 cursor = connection.cursor()
15
16 # Execute your SQL query - the years are stupid to catch all data
17 query = """
18 SELECT
19     IngredientName,
20     Category,
21     SUM(UnitPrice - PurchasePrice) AS TotalProfit,
22     (QuantitySold) AS TotalSold
23 FROM RestaurantFact
24 JOIN IngredientDimension ON RestaurantFact.IngredientID = IngredientDimension.IngredientID
25 WHERE SaleDate BETWEEN '2000-01-01' AND '2024-01-11'
26 GROUP BY IngredientName;
27 """
28
29 cursor.execute(query)
30
31 # Fetch the result set
32 result = cursor.fetchall()
33
34 # Close the database connection
35 cursor.close()
36 connection.close()
37
38 # Check if the result is not empty
39 if not result:
40     print("No data found.")
41 else:
42     # Extract data for plotting
43     ingredient_names, categories, total_profit, total_quantity_sold = zip(*result)
44
45     # Convert Decimal values to float - needed for plotting the data due to library limitations
46     total_profit = [float(value) for value in total_profit]
47     total_quantity_sold = [float(value) for value in total_quantity_sold]
48
49     # Define ingredient_indices - sets an order for the names
50     ingredient_indices = np.arange(len(ingredient_names))
51
```

```
Help  ← → Search

untitled.py X Settings
C:\Users\> eoin0 > untitled.py > ...
51 # ingredient_indices = np.arange(len(ingredient_names))
52
53 # Map categories to colours
54 category_color_mapping = {
55     'Fruits': 'b', # blue
56     'Grains': 'g', # green
57     'Meat': 'r', # red
58     'Vegetables': 'c', # cyan
59     'Dairy': 'm', # magenta
60     'Condiments': 'y', # yellow
61     'Seafood': 'k', # black
62 }
63
64 # Get the colours for each category - have to use use spelling ew
65 colors = [category_color_mapping[category] for category in categories]
66
67 # Plot the 3D bar chart with thicker bars - it was difficult to see prior
68 fig = plt.figure(figsize=(5, 5))
69 ax = fig.add_subplot(111, projection='3d')
70
71 bar_width = 0.5
72 ax.bar3d(ingredient_indices, total_profit, np.zeros_like(total_profit),
73         bar_width, bar_width, total_quantity_sold, color=colors)
74
75 ax.set_xticks(ingredient_indices)
76 ax.set_xticklabels(ingredient_names, rotation=45, ha='right') # Rotate it to see it better
77 ax.set_xlabel('IngredientName', labelpad=30)
78
79 ax.set_ylabel('TotalProfit')
80 ax.set_zlabel('TotalQuantitySold')
81
82 plt.show()
```

**The Profitability of ingredients, compared with the total amount sold categorised by which food group it is part of.**



Potatoes have the highest quantity sold but a rather meagre profit, this tells us that the price we sell it at can be increased.

Kale has the lowest profit and the second lowest amount sold; we need to reduce the price we buy it.

All types of seafood have a good profit margin and a middling amount of sales; this would imply we are selling it correctly.

## Changes made to optimise read access.

A star schema was chosen for our relationships rather than a snowflake as all our customers are stars of the world. The star schema allows us to achieve faster querying speeds due to its relative lack of complexity in comparison to the snowflake. This will allow us to gain insight into the ever-changing situation of stockroom inventory. The perishable nature of many of our products is what encourages us to use a system of less normalisation with a focus on greater write and read speed, which will allow those who need to read queries to get the answers they seek faster. This allows us to better reach our business requirements of minimising waste and maximising wealth.

We indexed our table through the field of ingredient name, this was done by

[illegible]

```

61 CREATE INDEX idx_instrument ON testinstrument (instrumentid)
62
63 -- Index for stations in testinstrument table (for webid column)
64 CREATE INDEX idx_station ON testinstrument (stationid)
65
66 -- Index for wavelengths in testinstrument table (for waveid column)
67 CREATE INDEX idx_wavelength ON testinstrument (wavelengthid)
68
69 -- Index for quantities in testinstrument table (for most waveid data)
70 CREATE INDEX idx_quantity ON testinstrument (quantityid)
71
72 -- Index for ingested on ingestedinstrument table (for ISO operation)
73 CREATE INDEX idx_ingested ON ingestedinstrument (ingested)
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
```

# References

We used ChatGPT to generate the data, we found it was the most reliable for pairing foodstuff to its real characteristics. It was also used for assistance in formatting the graph of profit.