Project 2

Core Skills Development

1 – Data Wrangling with SQL

The data provided is a csv containing a small dataset of 8 fields (one being the primary key) and 8 records. When imported into MySQL we get the following.

Order_ID	Customer_Name	Email	Phone	Product_Category	Order_Date	Revenue	Discount (%)
101	John Doe	john@email.com	9876543210	Electronics	12/31/2023	1200	10
102	Alice Smith		9898989898	Clothing	01-05-24	500	
103	Bob Miller	bob@email.com		Electronics	12-01-24	3000	20
104	John Doe	john@email.com	9876543210	Electronics	12/31/2023	1200	10
105	David White	david@email.com	9123456789	Furniture	02-15-2024	2500	15
106	Emma Brown	emma@email.com	9234567890	Clothing	08-03-24	700	5
107	Chris Green		9345678901	Furniture	04-10-24	1800	25
108	Alice Smith	alice@email.com		Clothing	03-08-24	500	

Figure 1 – Pre-transformed sales data imported into MySQL using SELECT all.

From what we can observe in *figure 1*, the data we were provided contains basic customer details, and transaction info. Within the data there are blank values, formatting issues in **Order_date**, and potential duplicates that we would need to address.

1.1 - Duplicates

We remove duplicates first as this can potentially reduce the workload of subsequent steps such as filling in blanks. The easiest way to identify duplicates is to count the number of record occurrences that exceeds 1 when querying by a unique or composite key. As each **Order_ID** is unique, a composite key was used instead.

```
SELECT Customer_Name, Email, Phone, Order_Date, COUNT(*) as Dupes
    FROM sales
    GROUP BY Customer_Name, Email, Phone, Order_Date
    HAVING Dupes > 1;
```

Customer_Name	Email	Phone	Order_Date	Dupes
John Doe	john@email.com	9876543210	12/31/2023	2

Figure 2 – Query & result for identifying duplicates in sales table.

We used a combination of **Customer_Name**, **Email**, and **Phone** to create a unique customer profile, as well as **Order_Date** to uniquely identify the customer by their transaction, as a customer can have multiple transactions. **Revenue** and **Product_Category** can also be combined, but since it's a small dataset, this was already sufficient. In *figure 2*, we can see that there are two John Doe transactions on the same date which would be a duplicate.

```
DELETE s1 FROM sales as s1
    JOIN sales as s2
        ON s1.Order_Date = s2.Order_Date
        AND s1.Customer_Name = s2.Customer_Name
        AND s1.Email = s2.Email
        AND s1.Phone = s2.Phone
        AND s1.Order_ID < s2.Order_ID;
```

Order_ID	Customer_Name	Email	Phone	Product_Category	Order_Date	Revenue	Discount (%)
102	Alice Smith		9898989898	Clothing	01-05-24	500	
103	Bob Miller	bob@email.com		Electronics	12-01-24	3000	20
104	John Doe	john@email.com	9876543210	Electronics	12/31/2023	1200	10
105	David White	david@email.com	9123456789	Furniture	02-15-2024	2500	15
106	Emma Brown	emma@email.com	9234567890	Clothing	08-03-24	700	5
107	Chris Green		9345678901	Furniture	04-10-24	1800	25
108	Alice Smith	alice@email.com		Clothing	03-08-24	500	

Figure 3 – Query & result for removing duplicates in sales table.

Subsequently we run an inner join query to delete one of the two duplicate records, prioritizing keeping the newest record; as recent entries usually contain the most

accurate information, due to being up to date. As shown in *figure 3*, **Order_ID** 101 was removed.

1.2 - Identifying Outliers

Outliers can dilute the purity of data which in turn affects the accuracy during analysis, so they should ideally be removed. The formula below was used to identify values exceeding a 3 standard deviation threshold.

Mean [Value] +/- (3 * Standard Deviation [Value])

Both **Revenue** and **Discounts** were substituted as values into the formula through an SQL query, returning no values – indicating there are no egregious outliers present.

1.3 - Blank Values

There are three fields with blank values in *figure 1*, being: **Email**, **Phone**, and **Discount**. We can either drop a column or fill in the blanks. **Discount** is a field that can be statistically analyzed and therefore should be kept – the blank values can be filled in with a 0 to indicate no discounts were present during purchase, whilst conforming to the data type of integer.

On the other hand, **Email** and **Phone** are qualitative data that are uniquely tied to the customer. Strictly speaking they have little analysis value and should be dropped unless we are analyzing specific niche trends, such as dialing codes or email domains by spending habits. However, for the sake of the activity, they were filled using default values of; 'no_email_provided@email.com', and '0000000000'. Both of which conform to the validation rules of a string containing '@email.com', and a 10-digit integer.

Order_ID	Customer_Name	Email	Phone	Product_Category	Order_Date	Revenue	Discount (%)
102	Alice Smith	no_email_provided@email.com	9898989898	Clothing	01-05-24	500	0
103	Bob Miller	bob@email.com	0000000000	Electronics	12-01-24	3000	20
104	John Doe	john@email.com	9876543210	Electronics	12/31/2023	1200	10
105	David White	david@email.com	9123456789	Furniture	02-15-2024	2500	15
106	Emma Brown	emma@email.com	9234567890	Clothing	08-03-24	700	5
107	Chris Green	no_email_provided@email.com	9345678901	Furniture	04-10-24	1800	25
108	Alice Smith	alice@email.com	0000000000	Clothing	03-08-24	500	0

Figure 4 – Sales table with all blank values filled in with new default values.

1.4 - Formatting

In *figure 4*, there are two fields that require formatting – **Order_Date** which has a mix of '-', and '/' as delimiters, and **Phone** which has 10-digits instead of the standard 11, leading with a 0.

Order_Date can be resolved by running a query to replace '/' with '-', then converting the string into a date with the first value being the month, day, then year.

Phone requires setting the data type to VARCHAR(11) then using LPAD to append a 0 at the start of all 10-digit phone numbers.

Order_ID	Customer_Name	Email	Phone	Product_Category	Order_Date	Revenue	Discount (%)
102	Alice Smith	no_email_provided@email.com	09898989898	Clothing	2024-01-05	500	0
103	Bob Miller	bob@email.com	00000000000	Electronics	2024-12-01	3000	20
104	John Doe	john@email.com	09876543210	Electronics	2023-12-31	1200	10
105	David White	david@email.com	09123456789	Furniture	2024-02-15	2500	15
106	Emma Brown	emma@email.com	09234567890	Clothing	2024-08-03	700	5
107	Chris Green	no_email_provided@email.com	09345678901	Furniture	2024-04-10	1800	25
108	Alice Smith	alice@email.com	00000000000	Clothing	2024-03-08	500	0

Figure 5 – Sales table with data formatting applied to Order_Date & Phone.

1.5 - Other Considerations

A customer table can be derived from the sales table using **Customer_Name**, **Email**, and Phone, with **Customer_ID** to join the two tables as a foreign key. Due to how small the dataset is, this idea was scrapped as it would have little impact on the end product. **Customer_Name** could also be split into first and last name, however it is currently redundant to do so as both derived columns have low value for analysis.

There are also two Alice Smiths present in the dataset. At present it's difficult to deduce whether they are the same person just based on the name; therefore, default values were used to fill in their respective contact details.

2 – Sales Summary and Analysis Using SQL & Python

2.1 - Correlations between Quantitative Data

As we have numerical data, a correlation matrix can be helpful in identifying trends between quantitative data. The selected fields will be **Revenue**, **Discount**, and **Order_Date**. **Order_ID** and **Phone** will be ignored despite being numerical, as both can also be classified as qualitative data which you would not perform mathematical operations on.

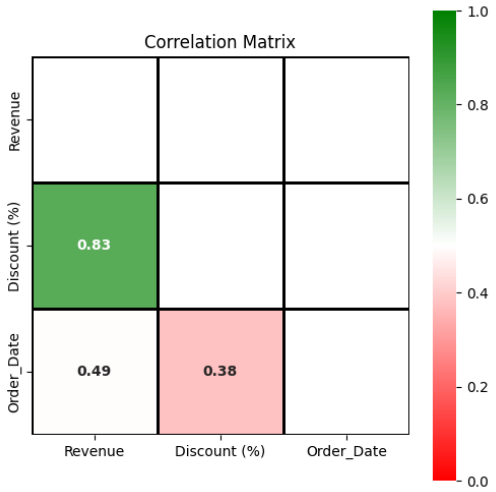

Figure 6 – Seaborn correlation matrix for sales data, not accounting for p-values.

From *figure 6*, we can identify that all relationships have a positive correlation, however only **Discount** and **Revenue** have a strong positive correlation, suggesting that the higher the discount, the higher the potential revenue.

P-Values:

	Revenue	Discount (%)	Order_Date
Revenue	1.0000	0.3900	0.3279
Discount (%)	0.3900	1.0000	0.0622
Order_Date	0.3279	0.0622	1.0000

Figure 7 – P-values between sales data.

It is worth mentioning that **none of the p-values** between the column and value pairs are **less than 0.05.** Suggesting that correlations in the quantitative data are all due to random chance. However, it's also worth noting that the sales dataset is small so there is room for error.

2.2 - Statistical Analysis

	Revenue	Discount (%)
count	7.000000	7.000000
mean	1457.142857	10.714286
std	1004.750621	9.759001
min	500.000000	0.000000
25%	600.000000	2.500000
50%	1200.000000	10.000000
75%	2150.000000	17.500000
max	3000.000000	25.000000

Figure 8 – Compiled statistics by Revenue & Discount.

We can observe in *figure 8*, that an **average** customer spends around **£1450** with a **10%** discount.

A **standard deviation** of **£1000** suggests a wide spread of revenue between orders. Whereas a **10% standard deviation** in discounts suggests a lower spread, with a **max** of **25%** and **min** of **0**.

The **maximum** amount a customer has spent is **£3000**, which is **£1500** greater than the average. With the **minimum** being **£500** (**£1000** less than average). When comparing the difference, we can determine that the distribution is skewed to the left due to the smaller range from average to minimum. This suggests a higher frequency of low revenue purchases, but greater spread of high.

2.3 - Data Aggregation

Product_Category	Total_Orders	Total_Revenue	Average_Revenue	STD_Revenue	Average_Discount
Clothing	3	1700	566.6667	94.28090415820634	1.6666666666666667
Electronics	2	4200	2100.0000	900	15
Furniture	2	4300	2150.0000	350	20

Figure 9 – Aggregated sales data grouped by category.

Furniture generates the **most revenue** on average, followed shortly by Electronics, both of which having an **average of 15-20% discounts**. However, **Clothing has more orders**.

MONTH(Order_Date)	Total_Orders	Total_Revenue	Average_Revenue	Average_Discount
1	1	500	500.0000	0
2	1	2500	2500.0000	15
3	1	500	500.0000	0
4	1	1800	1800.0000	25
8	1	700	700.0000	5
12	2	4200	2100.0000	15

Figure 10 – Aggregated sales data grouped by month.

December generated the most revenue and orders, but **February** made the most revenue on **average**. The least popular months with orders are **January, March, and August**. With no orders between **May to June**, and **September to November**.

Based on the aggregated data, we can identify that there is to some degree of positive correlation between revenue and discounts, as the segments with the most revenue also have a high average percentage of discount.

Order_Size	Sum of Revenue	Average Revenue	Count of Order	Average Discount
Large	£5,500.00	£2,750.00	2	18
Medium	£3,000.00	£1,500.00	2	18
Small	£1,700.00	£566.67	3	2
Grand Total	**£10,200.00**	**£1,457.14**	**7**	**11**

Figure 11 – Aggregated sales data of order sizes in Excel pivot table.

In *figure 11*, customers are more inclined towards small orders, however the bulk of the total revenue comes from large orders. Conversely, the average discount increase if the order is not small.

3 – Visualizations Using Excel & Power BI

Sum of Revenue	Order_Month						
Product_Category	January	February	March	April	August	December	Grand Total
Clothing	£500.00	£0.00	£500.00	£0.00	£700.00	£0.00	£1,700.00
Electronics	£0.00	£0.00	£0.00	£0.00	£0.00	£4,200.00	£4,200.00
Furniture	£0.00	£2,500.00	£0.00	£1,800.00	£0.00	£0.00	£4,300.00
Grand Total	£500.00	£2,500.00	£500.00	£1,800.00	£700.00	£4,200.00	£10,200.00

Figure 12 – Heatmap of Revenue by Category & Month using Excel pivot tables.

With red being low performing, and green being high. We can infer from *figure 11*, that **Furniture** generates the most revenue by category, with **December** being the month. When accounting for both, **Electronics sold the best during December**.

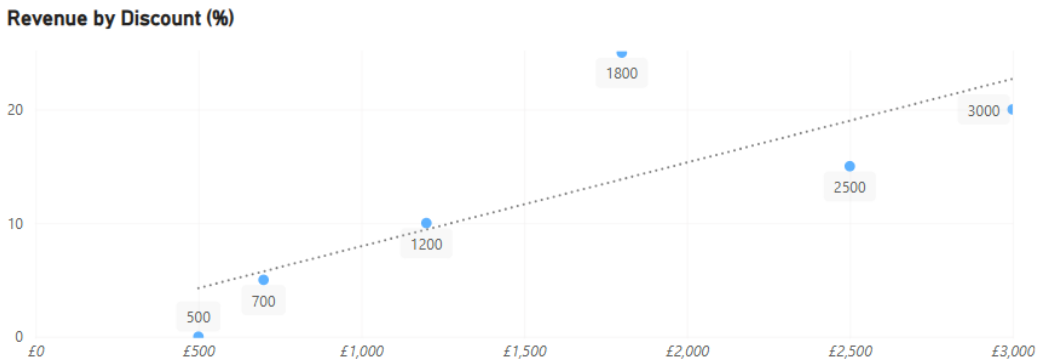

Figure 13 – Scatter graph of sum Revenue by Discount w/ trend line.

Based on the available data, there is a **positive correlation** between discounts and revenue as indicated by the trend line. The higher the discount the greater the revenue.

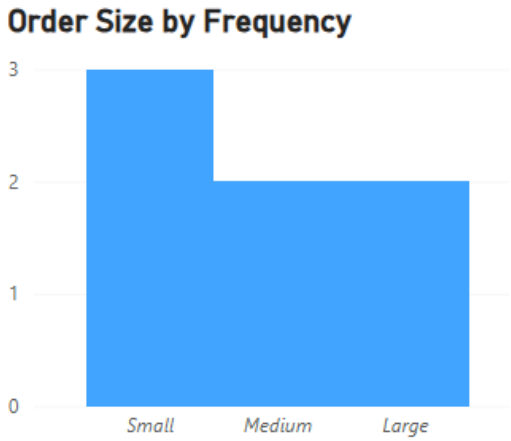

Figure 14 – Histogram of Order Size by frequency.

Order sizes are categorized by a revenue range per order:

- Small <= £1000.
- Medium <= £2000.
- Large > £2000.

The distribution is **skewed to the left**, indicating customers are more inclined to purchase smaller orders, with medium and large being similar in order frequency.

4 – Summary Report

Conclusions that can be drawn from the findings:

- Clothing generates the least revenue of the product categories.
- December generates the most revenue by month.
- Higher discounts can increase revenue.
- Electronics & Furniture are only ordered when there is a 15% or higher discount.
- Most of the total revenue comes from medium and large orders.

4.1 - Recommended Actions

Electronics and furniture should be the primary focus of promotion, both of which account for over 40% of the total revenue each. A single order from either category makes over three times as much as 3 clothing orders – the downside is that both categories have individually less orders, however promotions can mitigate this issue.

Discounts can be leveraged to incentivize purchases, especially higher costed orders such as electronics and furniture; which categorically fall into both medium and large orders. Customers are willing to spend more, the greater the discount is; however, a business will want to reduce the discount as much as possible to maximize profit.

According the large and medium orders, **18%** is a sweet spot for discounts – using it as the midpoint, a range of 15-20% for orders over £2000 (maximum of £3000) is a good balance between discount and revenue. Whilst the same range could also be applied for medium orders (£1000 to £2000) it's worth noting that the mean spending of a customer and medium orders are roughly the same at **£1500**. The average discounts can be rounded to around 10%, meaning that the minimum range of discounts for medium orders can also be dropped to 10%. A safe range for medium orders can be set to 10-20%, or a riskier 10-15% for increased profits at the potential expense of less orders.

Regarding clothing, a suggestion is to sell them as a bundle with a discount attached. We've identified that small orders do not generate much revenue, so naturally the best course of action is to simply convert it into a medium or large order when possible. The average spending of a customer is £1500, so medium orders are the best choice; whilst retaining as many customers as possible.

December is the on-peak month; therefore, marketing campaigns should be targeted for this period especially for electronics. Increased discounts should also be available to incentivize sales during May to June, and September to November, as there were no recorded sales during these months.