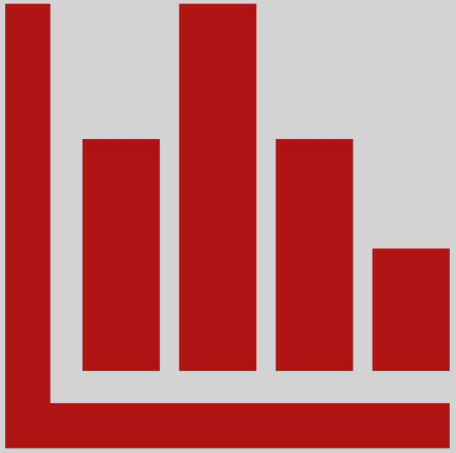

DATA ANALYTICS LEARNCAMP

MINI PROJECT IV

ISAAC IKENNA MAGNUS

STEP ONE: GENERATIING ‘SALES’ TABLE

```
Adventure works Databse
-- view all tables
SELECT * FROM [dbo]. [ Returns]
SELECT * FROM [dbo] .[Sales 2020]
SELECT * FROM [dbo] .[Sales 2021]
SELECT * FROM [dbo] .[Sales 2022]
SELECT * FROM [dbo] .[Calendar]
SELECT * FROM Customers
SELECT * FROM products
SELECT * FROM [dbo] .[Product Categories]
SELECT * FROM [dbo] .[Product Subcategories]
SELECT * FROM Territory
SELECT * FROM Sales

-- Assemble all the sales table using UNION ALL
SELECT * FROM [dbo] .[Sales 2020]
UNION ALL
SELECT * FROM [dbo] .[Sales 2021]
UNION ALL
SELECT * FROM [dbo] .[Sales 2022] AS Sales

-- sales table for 2020, 2021, 2022 all in one using a sub query
 SELECT *  INTO Sales From

(SELECT * FROM [dbo] .[Sales 2020]
UNION ALL
SELECT * FROM [dbo] .[Sales 2021]
UNION ALL
SELECT * FROM [dbo] .[Sales 2022]) AS Sales
```

CONTINUATION OF STEP ONE

```sql
--create New column Order amount' in the Sales table by mutilplying the product price with the order
quantity

SELECT ROUND( sales. OrderQuantity * Products. ProductPrice, 0) AS TotalPrice
FROM Sales
JOIN Products
ON Sales. ProductKey = Products. ProductKey

--B. A new table 'Sales Prep'

SELECT sales.*, ROUND (sales. OrderQuantity * Products. ProductPrice, 0) AS TotalPrice
INTO Sales Prep
FROM Sales
JOIN Products
ON Sales.ProductKey = Products .ProductKey

SELECT * FROM Sales_Prep
SELECT * FROM [dbo]. [Sales]

--create a another table 'Customers_ Prep after adding a new column --FullName
SELECT *, CONCAT (Prefix, ' ', FirstName,' ', LastName ) AS FullName
INTO Customers_Prep
FROM Customers

SELECT * FROM Customers_Prep
```

STEP TWO

QUESTIONS ONE: RETRIEVE THE CUSTOMERS WHO HAVE PLACED ORDERS WITH A TOTAL AMOUNT GREATER THAN $10000

```sql
--Total customers who plced orders with total amount greater than $10000

SELECT
 c. CustomerKey, c. FUllname, SUM(s. TotalPrice) as Total_Amount

FROM Customers_Prep c
JOIN Sales_Prep s
    ON c.CustomerKey = s.CustomerKey
GROUP BY c. CustomerKey, c. fullname
HAVING SUM(s. TotalPrice) > 10000;
```

QUESTIONS TWO: RETRIEVE THE TOTAL REVENUE AND TOTAL ORDERS GENERATED BY EACH PRODUCT CATEGORY

```sql
-- Retrieving the total revenue and total orders generated by each product category

SELECT * FROM Products
SELECT * FROM Sales_Prep
SELECT * FROM [Product Categories]
SELECT * FROM [Product Subcategories]
SELECT
    PC. ProductCategoryKey,
    PC. CategoryName,
    COUNT(s. OrderNumber) AS TotalOrders,
    ROUND (SUM(s. TotalPrice),2) AS TotalRevenue
FROM
    [dbo]. [Product Categories] pc

JOIN
    [dbo]. [Product Subcategories] psc
    ON PC.ProductCategoryKey = psc.ProductCategoryKey
JOIN
    Products p
    ON psc.ProductSubcategoryKey = p. ProductSubCategoryKey
LEFT JOIN
    Sales_Prep S
    ON P. ProductKey = s.Productkey
-- WHERE o.OrderNumber IS NOT NULL
GROUP BY
    PC. ProductCategoryKey, PC.CategoryName
ORDER BY TotalRevenue DESC;
```

QUESTIONS THREE: RETRIEVE THE NAMES OF PRODUCTS, SUB-CATEGORIES AND THEIR CATEGORIES FOR PRODUCTS WITH A SELLING PRICE GREATER THAN THE AVERAGE SELLING PRICE OF ALL PRODUCTS – SQ

```sql
-- Retrieving the names of products, sub-categories and their categories for products with a selling
price greater than the average selling price of all products. - SQ

SELECT * FROM Products
SELECT * FROM [Product Categories]
SELECT * FROM [Product Subcategories]
SELECT p.Productname, psc. subcategoryName, pc.Categoryname
FROM Products p
JOIN [Product Subcategories] Psc ON p. ProductSubcategoryKey = psc.ProductSubcategoryKey
JOIN [dbo].[Product Categories] pc ON pc.ProductCategoryKey = psc.ProductCategoryKey
WHERE p.ProductPrice > (SELECT AVG(Productprice) FROM Products)
```

QUESTIONS FOUR: HOW MANY CUSTOMERS HAVE PLACED ORDERS IN “ CANADA” ? LIST THE CUSTOMER NAMES – SQ

```sql
-- List of customers who placed orders in "Canada"? and their names - SQ
SELECT * FROM Customers _Prep
SELECT * FROM Territory
SELECT * FROM [dbo]. [Sales_Prep]
SELECT
    c. FullName
FROM Customers_Prep c
WHERE c. CustomerKey IN
    (SELECT s.CustomerKey
    FROM Sales_Prep s
    JOIN Territory t ON s. TerritoryKey = t. SalesTerritoryKey
    WHERE t.Country = 'Canada')
```

QUESTIONS FIVE: FIND 10 PRODUCTS THAT HAVE BEEN RETURNED THE MOST. HOW MUCH MONEY WAS GENERATED BY THESE PRODUCTS ? – SQ

```sql
-- 10 products that have been returned the most. How much money was generated by these products? - SQ

SELECT * FROM [ Returns ]
SELECT * FROM Sales_Prep
SELECT * FROM Products

SELECT p.productname, return_count, revenue_generated
FROM products p
JOIN (SELECT TOP 10 s.ProductKey, COUNT(r. TerritoryKey) AS return_count, SUM(s. TotalPrice) AS
revenue_generated

FROM Sales_Prep s
    LEFT JOIN [dbo].[ Returns ] r ON s. TerritoryKey= r.territorykey
    GROUP BY s. ProductKey
    ORDER BY return count DESC)
    Returned _most ON p.ProductKey = Returned_most.ProductKey;
```

QUESTIONS SIX: GET THE LIST OF CUSTOMERS WHO HAVE PLACED ORDERS IN MORE THAN ONE TERRITORY.

```sql
-- list of customers who have placed orders in more than one territory

SELECT * FROM Customers_Prep
SELECT * FROM Territory
SELECT * FROM Sales_Prep

SELECT C. fullname, s.orderdate, t.SalesTerritoryKey
FROM Sales Prep S
JOIN Customers_Prep c ON s. CustomerKey = c.CustomerKey
JOIN Territory t ON t.SalesTerritoryKey = S.TerritoryKey
GROUP BY C.fullname, s.orderdate, t.SalesTerritoryKey
HAVING t. SalesTerritoryKey > 1
```

QUESTIONS SEVEN: RETRIEVE THE PRODUCT NAMES AND CORRESPONDING SUB-CATEGORIES FOR PRODUCTS THAT HAVE BEEN ORDERED AT LEAST 10 TIMES.

```sql
-- Retrieving the product names and their corresponding sub-categories for products that have been
ordered at least 10 times.

SELECT * FROM Products
SELECT * FROM [Product Subcategories]
SELECT * FROM Sales_Prep

SELECT p.ProductName, psc.subcategoryName
FROM products p
JOIN Sales_Prep s ON p.ProductKey = s.ProductKey
JOIN [dbo].[Product Subcategories] psc ON p.ProductSubcategoryKey = psc. ProductSubcategoryKey
GROUP BY p.productname, psc.subcategoryname
HAVING COUNT(s.OrderNumber) > = 10;
```

QUESTIONS EIGHT: LIST THE CUSTOMER NAMES WHO HAVE PLACED ORDERS AFTER 2021. WHAT IS THE DISTRIBUTION OF THEIR OCCUPATION ?

```sql
-- List the customer names who have placed orders after 2021. What is the distribution of their
occupation?

SELECT * FROM Customer_prep
SELECT FROM Sales_Prep
SELECT c.Fullname, c.occupation, COUNT(*) AS OrderCount
FROM Customers_Prep c
JOIN Sales_Prep S
ON C.CustomerKey = s.CustomerKey
WHERE s.OrderDate >'2020-12-31'
GROUP BY c.Fullname, c.occupation
ORDER BY OrderCount DESC;
```

QUESTIONS NINE: GET A LIST OF PRODUCTS AND THEIR CORRESPONDING ORDER QUANTITIES FOR PRODUCTS THAT HAVE BEEN ORDERED AT LEAST 5 TIMES.

```sql
-- Get a list of products and their corresponding order quantitles for products that have been ordered
at least 5 times

SELECT * FROM Products
SELECT * FROM Sales_Prep

SELECT p.ProductName, SUM(OrderQuantity) as total_quantity
FROM products p
JOIN Sales_Prep s ON p.ProductKey = s. ProductKey
GROUP BY p.productname
HAVING COUNT(OrderQuantity) >= 5;
```

QUESTIONS TEN: RETRIEVE THE PRODUCTS NAMES THAT START WITH THE LETTER “C” OR “H” AND ARE FROM THE CLOTHING CATEGORY.

```sql
-- Retrieving the product names that start with the letter "C" or "H" and are from the "Clothing"
category

SELECT * FROM [Product Categories]
SELECT * FROM [Product Subcategories]
SELECT * FROM Products

SELECT psc.* , p.productName, pc.categoryName
FROM [Product Subcategories] psc
JOIN Products p ON psc.productsubcategorykey = p.ProductSubcategoryKey
JOIN [Product Categories] pc ON pc.ProductCategoryKey = psc.ProductCategoryKey
WHERE ProductName LIKE '[CH]%' AND CategoryName = 'clothing'
```

QUESTIONS ELEVEN: RETRIEVE THE PRODUCT NAMES THAT HAVE BEEN ORDERED IN THE “UNITED STATES” OR “AUSTRALIA”

```sql
-- Retrieving the product names that have been ordered in the 'United States or "Australia"

SELECT * FROM Sales_Prep
SELECT * FROM Products
SELECT * FROM Territory

SELECT S.*, p.productName, t. country
FROM Sales_Prep s
JOIN Products p ON S.ProductKey = p.ProductKey
JOIN Territory t ON t.SalesTerritoryKey = s.TerritoryKey
WHERE Country IN ( 'United States', 'Australia')
```

QUESTIONS TWELVE: FIND THE CUSTOMER NAMES WHO PLACED ORDERS WITH A TOTAL AMOUNT GREATER THAN THE AVERAGE TOTAL AMOUNT OF ORDERS – SQ

```sql
--Find the customer names who have placed orders with a total amount greater than the average total
amount of orders - SQ

SELECT * FROM Customers_Prep
SELECT * FROM Sales_Prep

SELECT c.FullName, TotalPrice
FROM Sales_Prep s
JOIN Customers_Prep c ON c.CustomerKey = S.CustomerKey
GROUP BY c.FullName, TotalPrice
HAVING TotalPrice > (SELECT AVG(TotalPrice) FROM Sales_Prep)
```

QUESTIONS THIRTEEN: RETRIEVE THE TOP 5 CUSTOMERS WHO PLACED THE HIGHEST NUMBER OF ORDERS, ALONGSIDE THEIR ORDER COUNTS.

```sql
-- Retrieving the top 5 customers who have placed the highest number of orders, along with their order counts

SELECT * FROM Customers_ Prep
SELECT * FROM Sales_Prep

SELECT TOP 5 COUNT( orderQuantity) AS 'Order Count', c.FullName
FROM Customers_Prep C
JOIN Sales_Prep s ON c.CustomerKey = s.CustomerKey
GROUP BY c.FullName
ORDER BY COUNT(OrderQuantity) DESC
```

QUESTIONS FOURTEEN: LIST THE CUSTOMERS WHO PLACED ORDERS WITHIN THE LAST 6 MONTHS (WITH RESPECT TO THE LAST DATE IN THE GIVEN DATA).

```sql
-- List the customers who have placed orders within the last 6 months (with respect to the last date in the given data)

SELECT * FROM Customer_prep
SELECT * FROM Sales_Prep

SELECT DISTINCT c.FirstName, c.LastName
FROM Customers Prep c
JOIN sales_Prep s ON C.CustomerKey = s.CustomerKey
WHERE s.OrderDate >= DATEADD( MONTH, -6, (SELECT MAX(OrderDate) FROM Sales _Prep))
```

QUESTIONS FIFTEEN: WE WANT TO REACH OUT TO OUR BEST CUSTOMERS IN 2022. CAN YOU GET EMAILS OF TOP 50 CUSTOMERS BASED ON REVENUE ?

```sql
-- Best customers in 2022 and the emails of Top 50 customers based on revenue?

SELECT * FROM Customer_prep
SELECT * FROM Sales_Prep
SELECT c.Fullname, c.EmailAddress
FROM Customers_Prep c
JOIN (
    SELECT TOP 50 customerKey, SUM(TotalPrice) as total_revenue
    FROM Sales_ Prep s
    WHERE OrderDate >= '2022-01-01' AND OrderDate <= '2022-12-31'
    GROUP BY CustomerKey
    ORDER BY total_revenue DESC)
 top_customers ON c.CustomerKey = top_customers.CustomerKey
```

THE END