EDA and Data Preprocessing Reports

# Using SQL

BY:

Name: Kalluri Vishnu Vardhan Reddy

Email: vishnu15018@gmail.com

Mobile No: 9391586197

To 360digitmg

**Table of Contents:**

1. SQL Queries for EDA
2. Univariate
3. Numerical features
4. Categorical features
5. Multivariate Analysis

2.Data Preprocessing

1. Handling Duplicates
2. Handling Null values
3. Type Casting
4. Outlier Analysis
5. Creating New Column
6. Month wise analysis
7. **EXPLORATORY DATA ANALYSIS REPORT**

**EDA on numerical features:**

**Univariate analysis**

**SQL Queries:**

**NUMERICAL COLUMNS:**

**[‘Quantity’,Return\_Quantity, FInal\_costs, FInal\_sales,RtnMRP]**

**Creating a procedure for EDA on numerical data.**

-- Create the stored procedure for eda

CREATE PROCEDURE EDA\_Numeric

@columnName NVARCHAR(100),

@TableName NVARCHAR(100)

AS

BEGIN

SET NOCOUNT ON;

-- Calculate the mean (first moment)

DECLARE @mean FLOAT;

DECLARE @meanQuery NVARCHAR(MAX) = N'

SELECT @mean = AVG(' + QUOTENAME(@columnName) + N')

FROM ' + QUOTENAME(@TableName) + N';

';

EXEC sp\_executesql @meanQuery, N'@mean FLOAT OUTPUT', @mean OUTPUT;

-- Calculate the variance (second moment)

DECLARE @variance FLOAT;

DECLARE @varianceQuery NVARCHAR(MAX) = N'

SELECT @variance = AVG((' + QUOTENAME(@columnName) + N' - @mean) \* (' + QUOTENAME(@columnName) + N' - @mean))

FROM ' + QUOTENAME(@TableName) + N';

';

EXEC sp\_executesql @varianceQuery, N'@mean FLOAT, @variance FLOAT OUTPUT', @mean, @variance OUTPUT;

-- Calculate the skewness (third moment)

DECLARE @skewness FLOAT;

DECLARE @skewnessQuery NVARCHAR(MAX) = N'

SELECT @skewness = SUM(POWER(' + QUOTENAME(@columnName) + N' - @mean, 3)) / (COUNT(' + QUOTENAME(@columnName) + N') \* POWER(SQRT(@variance), 3))

FROM ' + QUOTENAME(@TableName) + N';

';

EXEC sp\_executesql @skewnessQuery, N'@mean FLOAT, @variance FLOAT, @skewness FLOAT OUTPUT', @mean, @variance, @skewness OUTPUT;

-- Calculate the kurtosis (fourth moment)

DECLARE @kurtosis FLOAT;

DECLARE @kurtosisQuery NVARCHAR(MAX) = N'

SELECT @kurtosis = SUM(POWER(' + QUOTENAME(@columnName) + N' - @mean, 4)) / (COUNT(' + QUOTENAME(@columnName) + N') \* POWER(@variance, 2))

FROM ' + QUOTENAME(@TableName) + N';

';

EXEC sp\_executesql @kurtosisQuery, N'@mean FLOAT, @variance FLOAT, @kurtosis FLOAT OUTPUT', @mean, @variance, @kurtosis OUTPUT;

-- Print the results

print 'column: '+ CAST(@columnName as varchar(30));

PRINT 'Mean: ' + CAST(@mean AS NVARCHAR(20));

PRINT 'Variance: ' + CAST(@variance AS NVARCHAR(20));

print 'Standard deviation: ' +CAST(SQRT(@variance) as nvarchar(20))

PRINT 'Skewness: ' + CAST(@skewness AS NVARCHAR(20));

PRINT 'Kurtosis: ' + CAST(@kurtosis AS NVARCHAR(20));

END;

**Executing the procedure on Numerical Columns:**

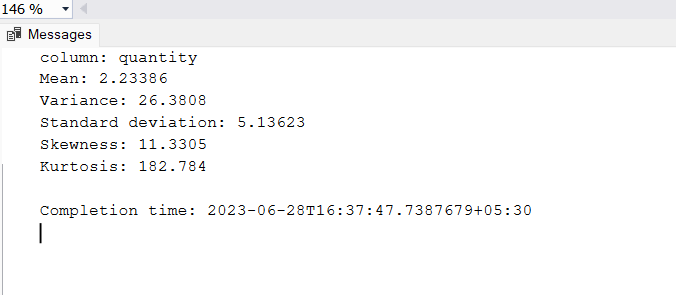
**1.**

**Column Name:** Quantity

**SQL Query:**

exec EDA\_Numeric @columnName = 'Quantity' ,@TableName = 'projectdata';

Results:



--

**Analysis:**

1. On average 2.23 quantity of drugs bought by the patients
2. High Kurtosis representing fat tail and peakness this implies outliers.

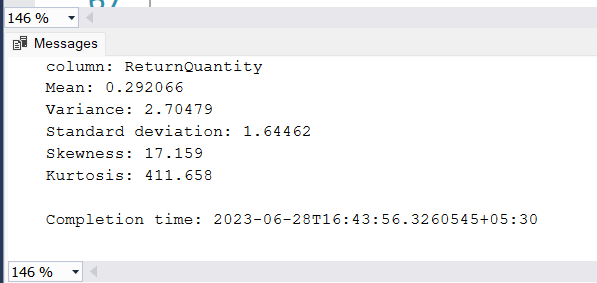
**2.**

**Column Name:** ReturnQuantity

**SQL Query:**

exec EDA\_Numeric @columnName = 'ReturnQuantity' ,@TableName 'projectdata';

Results:

--

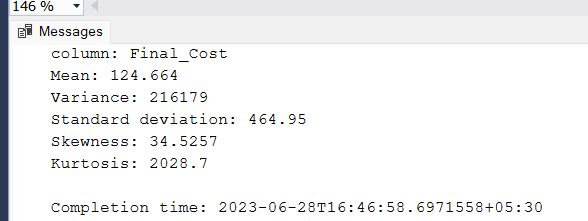
**3.**

**Column Name:** Final\_Cost

**SQL Query:**

exec EDA\_Numeric @columnName = 'Final\_Cost' ,@TableName = 'projectdata';

Results:



**Analysis:**

1. Mean cost of the drugs is 124.66
2. Variance is high indicating the data is widely distributed
3. High Kurtosis representing fat tail and peakness this implies outliers.

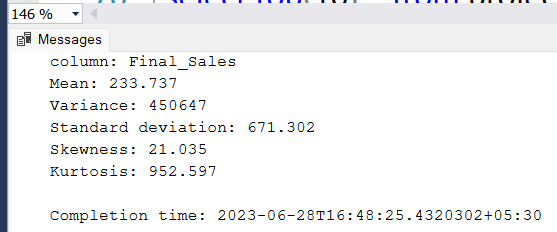
**4.**

**Column Name:** Final\_Sales

**SQL Query:**

exec EDA\_Numeric @columnName = 'Final\_Sales' ,@TableName = 'projectdata';

Results:



**Analysis:**

1. Mean sales value is 233.737
2. Variance is high indicating the data is widely distributed
3. High Kurtosis representing fat tail and peakness this implies outliers.

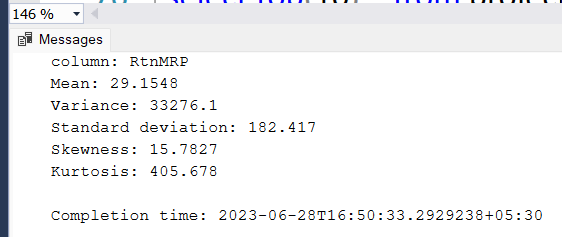
**5.**

**Column Name:** RtnMRP

**SQL Query:**

exec EDA\_Numeric @columnName='RtnMRP' ,@TableName = 'projectdata';

Results:



**NOTE :**

**Adding the ID Column to the dataset to uniquely identify the rows [ Like a primary key]**

**SQL Query:**

ALTER TABLE projectdata

ADD ID INT IDENTITY(1, 1);

**---- Multivariate Analysis ----**

**EDA on Categorical Features:**

**NOTE: Multivariate analysis of categorical column along with numerical columns using the group by keyword with mean aggregation:**

**Stored Procedure for EDA on Categorical Data with mean aggregation:**

create procedure EDA\_CAT

@columnName2 varchar(20),

@TableName2 varchar(20)

AS

BEGIN

SET NOCOUNT ON;

declare @query nvarchar(max);

set @query = N'

select '+QUOTENAME(@columnName2)+ N',count(ID) as count,avg(Quantity) as mean\_quantity,avg(ReturnQuantity)as mean\_Returnquantity,avg(Final\_Cost) as mean\_Final\_Cost, avg(Final\_Sales) as mean\_Final\_Sales

FROM ' + QUOTENAME(@TableName2) + N' as pd group by(pd.Typeofsales) order by count desc;';

exec sp\_executesql @query;

END

**Executing the procedure on categorical column:**

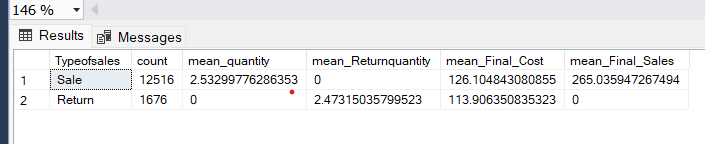
**1.**

**Column:** Type of sales

**SQL Query:**

exec EDA\_CAT @columnName2='Typeofsales',@TableName2='projectdata';

**Results:**

****

**Analysis:**

1. On an average the return quantity by patient is 2.47 and mean cost of the retuned drugs is 113.906
2. On an average each patient bought around 2.53 units (3 units), and the patients spends on average 265 rupees on the medicine’s drugs.

**2.**

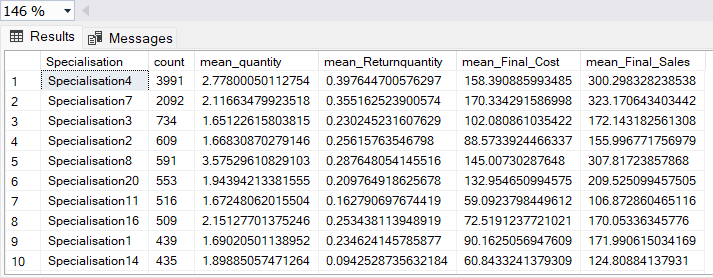
**Column:** Specialisation

**SQL Query:**

exec EDA\_CAT @columnName2='Specialisation',@TableName2='projectdata';

**Analysis:**

1. Below are the top 10 specialisations in demand
2. Overall total 58 specialisations are there in the dataset
3. Mean sales for the top specialisation is 300.29
4. Mean sales for second most specialisation is 323.17 which is greater than top most.

**Results:**

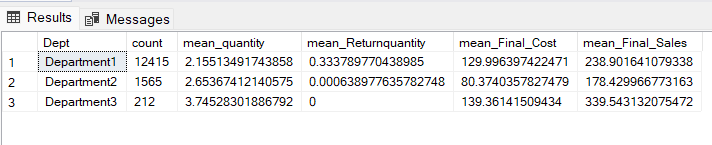
**3.**

**Column:** Dept

**SQL Query:**

exec EDA\_CAT @columnName2='Dept',@TableName2='projectdata';

**Results:**

****

**Analysis:**

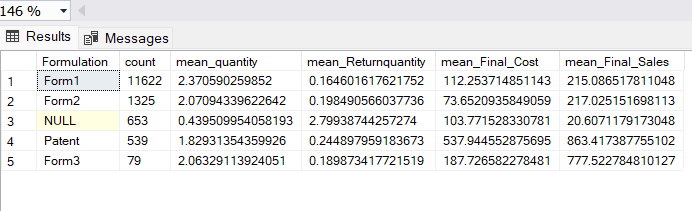
1. Depatment1 dominates in sales with 89.41 percent market.
2. No products related to the department is calculated.
3. Patients never returned the drugs purchased from the department3.

**4.**

**Column:** Formulation

**SQL Query:**

exec EDA\_CAT @columnName2='Formulation',@TableName2='projectdata';

**Results:****Analysis:**

1. Demand for the form1 drugs is more compared to all other formulation drugs.
2. In above null represents formulation is not mentioned the dataset, I filled that with **unknown** in my data preprocessing.
3. **75%** Sales belong to the form1 category.

**NOTE: Multivariate analysis of categorical column along with numerical columns using the group by keyword with sum aggregation:**

**Stored Procedure for EDA on Categorical Data with sum aggregation:**

**SQL Query:**

--------------sum aggregation on numerical data based on grouping the categorical data ------

--- procedure for eda on category for sum aggregation ---------------

create procedure EDA\_CAT\_SUM

@columnName3 varchar(20),

@TableName3 varchar(20)

AS

BEGIN

SET NOCOUNT ON;

declare @query nvarchar(max);

set @query = N'

select '+QUOTENAME(@columnName3)+ N',count(ID) as count,sum(Quantity) as sum\_quantity,sum(ReturnQuantity)as sum\_Returnquantity,sum(Final\_Cost) as sum\_Final\_Cost, sum(Final\_Sales) as sum\_Final\_Sales

FROM ' + QUOTENAME(@TableName3) + N' as pd group by(pd.'+QUOTENAME(@columnName3)+ N') order by count DESC;' ;

exec sp\_executesql @query;

END

**Executing the procedure on columns:**

exec EDA\_CAT\_SUM @columnName3='Typeofsales',@TableName3='projectdata';

exec EDA\_CAT\_SUM @columnName3='Specialisation',@TableName3='projectdata';

exec EDA\_CAT\_SUM @columnName3='Dept',@TableName3='projectdata';

exec EDA\_CAT\_SUM @columnName3='Formulation',@TableName3='projectdata';

exec EDA\_CAT\_SUM @columnName3='SubCat' ,@TableName3='projectdata';

exec EDA\_CAT\_SUM @columnName3='SubCat1' ,@TableName3='projectdata';

exec EDA\_CAT\_SUM @columnName3='DrugName' ,@TableName3='projectdata';

Analysis:

1.Anti-infectives and Intravenous and other sterile solution are the top most performed drugs with respect to sales, but the demand with respect to quantity is highest in Intravenous and other sterile solutions

2.There may be viral, bacterial infections, abdominal related infections, cardio related conditions, fever related disease or any specific medical condition more prevalent.

3.Almost 90.8 sales belong to Injections, fluids electrolytes, tablets and capsules.

4.for details about the infections and diseases we need to go through the patient disease data. [Those data not in our scope]

5. Given the high sales of "INJECTIONS," "IV FLUIDS, ELECTROLYTES, TPN," and "TABLETS & CAPSULES," it may be wise to ensure sufficient inventory levels for these subcategories to meet customer demand and avoid stockouts.

6. Top5 specialisation are specialisation4, 7, 8,3 ,20. there may be a chance that among this specialization any one must belong to treating the corona related cases. [ as the data is taken from 2022]

7 Almost 87% percent of drugs bought patients are from deparment1

8 Dept 1 has highest demand and sales compared to all other departments (almost 89% sales are from department1), and also among all three departments money was spent on dept1.

**2.DATA PREPROCESSING**

**NOTE:** We start with making a copy of the existing table

**SQL Query:**

SELECT \*INTO copyprojectdata FROM projectdata;

**Results:**

1. Copy of the dataset is stored in the copyprojectdata table.

**Steps for preprocessing the data:**

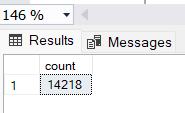
**Step1: Removing Duplicate Rows**

Total No of records in the dataset

**SQL Query:**

select count(\*) as count from projectdata;

Results:

****

Below query is to remove the rows using the CTE expressions using the row number function which provide a unique value to each row.

**SQL Query:**

BEGIN TRANSACTION;

WITH DuplicateRows AS (

SELECT \*,

ROW\_NUMBER() OVER (PARTITION BY Typeofsales, Patient\_ID, Specialisation, Dept, Dateofbill, Quantity, ReturnQuantity, Final\_Cost, Final\_Sales, RtnMRP, Formulation, DrugName, SubCat, SubCat1 ORDER BY ID) AS RowNumber

FROM projectdata

)

DELETE FROM DuplicateRows

WHERE RowNumber > 1;

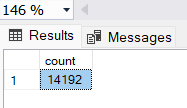
COMMIT TRANSACTION;

No of rows after the deletion

**SQL Query:**

select count(\*) as count from projectdata;

**Results:**



**Analysis:**

1. 26 duplicate rows found and they are deleted.
2. Duplicate rows will reflect redundancy

**Step2: Handling Null Values**

**SQL Query:**

**Below procedure will find the null value in a column**

create procedure findnull

@columnname varchar(30),

@tablename varchar(30)

as

begin

declare @query Nvarchar(max);

set @query = N'

select '''+@columnname+N''' as column\_name ,count(\*) as count\_null

from '+QUOTENAME(@tablename)+N'

where '+quotename(@columnname)+N' IS NULL;

';

exec sp\_executesql @query;

end

Executing the procedure on all columns:

exec findnull @columnname='Typeofsales', @tablename='projectdata';

exec findnull @columnname='Patient\_ID', @tablename='projectdata';

exec findnull @columnname='DEPT', @tablename='projectdata';

exec findnull @columnname='Dateofbill', @tablename='projectdata';

exec findnull @columnname='Quantity', @tablename='projectdata';

exec findnull @columnname='ReturnQuantity', @tablename='projectdata';

exec findnull @columnname='Final\_Cost', @tablename='projectdata';

exec findnull @columnname='Final\_Sales', @tablename='projectdata';

exec findnull @columnname='RtnMRP', @tablename='projectdata';

exec findnull @columnname='Formulation', @tablename='projectdata';

exec findnull @columnname='DrugName', @tablename='projectdata';

exec findnull @columnname='SubCat', @tablename='projectdata';

exec findnull @columnname='SubCat1', @tablename='projectdata';

**Analysis:**

1. Formulation column has 650 null values
2. Drug Name has 1668 null values
3. Subcat has 1668 null values
4. Subcat1 has 1692 null values

**Handling the NULL values:**

**Analysis:**

1. As all the null values are present in the categorical column, so filling it with imputed strategies like mode, backfill won’t give good results. Because when we impute the null values with mode or backfill it will provide the wrong sales results for the category that is mode.
2. So, I am going to fill it with the ‘Unknown’ value, so that the sales related to the null category will be analysed separately.

**SQL Query:**

update projectdata

set Formulation='Unknown'

where Formulation IS NULL;

update projectdata

set DrugName='Unknown'

where DrugName IS NULL;

update projectdata

set SubCat='Unknown'

where SubCat IS NULL;

update projectdata

set SubCat1='Unknown'

where SubCat1 IS NULL;

**Step4:** Type Casting

* Type casting the Date column to date.
* Initially the date column is in string format in the raw data [mm-dd-yyyy,mm/dd/yyyy]

**SQL Query:**

begin transaction

begin try

update projectdata

set Dateofbill=convert(date,replace(Dateofbill,'-', '/'),101);

alter table projectdata

alter column Dateofbill DATE;

commit;

END TRY

BEGIN CATCH

ROLLBACK;

END CATCH;

**Step5:** Outlier Analysis

**To find the No of outliers in the each numerical column using the z Score method with threshold 3 standard deviations.**

**1.Reason for the outliers**

1. outliers in this dataset are natural outliers.
2. 3.These outliers occurs because sometimes some patient needs antibiotics and other costly drug to cure a specific disease and also when a patient bought more quantity of a drug it leads to high transaction value, these high values cause outliers in sales
3. we need to keep these outliers as well to perform sales analysis
4. As these outliers are natural, we need to keep those outliers.
5. . We need to analyse this outliers separately, by creating a new column cost\_per\_unit and price \_category

**reason for creation of the cost\_per\_unit and price\_category columns**

1. To analyse the sales better, to find the market segments by separately analysing the outliers drugs that belongs to high and medium range.
2. To gain more view into the sales segments of each department.

**SQL Query:**

**1.-- adding the cost\_per\_unit column**

ALTER TABLE projectdata

ADD cost\_per\_unit DECIMAL(10, 2); -- Adjust the data type and precision as needed

UPDATE projectdata

SET cost\_per\_unit =

CASE

WHEN Quantity > 0 THEN Final\_Cost / Quantity

WHEN ReturnQuantity > 0 THEN Final\_Cost / ReturnQuantity

ELSE 0 -- or any default value you prefer

END;

**2. --- creating the column price \_category**

ALTER TABLE projectdata

ADD price\_category VARCHAR(20); -- Adjust the data type and length as needed

DECLARE @mean FLOAT, @std FLOAT;

DECLARE @zscore\_threshold FLOAT = 3;

select @mean=AVG(cost\_per\_unit) , @std=STDEV(cost\_per\_unit) from projectdata;

UPDATE projectdata

SET price\_category =

CASE

WHEN cost\_per\_unit >= 0 AND cost\_per\_unit <= 200 THEN 'Low'

WHEN cost\_per\_unit > 200 AND cost\_per\_unit <= @mean + (3 \* @std) THEN 'Medium'

ELSE 'High'

END;

**3.Creating the column Profit**

----- Adding column profit for profit analysis ---------------------------------

ALTER TABLE projectdata

ADD Profit DECIMAL(10,2); -- Adjust the data type and precision as needed

UPDATE projectdata

SET Profit = Final\_Sales - Final\_Cost;

**Analysing the data using the month**

SELECT

YEAR(Dateofbill) AS Year,

DATENAME(MONTH, Dateofbill) AS Month,

SUM(Quantity) AS Total\_Quantity,

SUM(ReturnQuantity) AS Total\_Return\_Quantity,

SUM(Final\_Cost) AS Total\_Final\_Cost,

SUM(Final\_Sales) AS Total\_Final\_Sales

FROM projectdata

GROUP BY YEAR(Dateofbill), DATENAME(MONTH, Dateofbill)

ORDER BY YEAR(Dateofbill), DATENAME(MONTH,Dateofbill);

**ANALYSIS**

---December has the highest sales and highest demand with respect to quantity.

---But in august highest return quantity is recorded