**Medical inventory optimization**

**Pre-processing Code (SQL) by Pratiksha Saheb**

**Software: MySQL Workbench**

**1. Creating the database.**

CREATE DATABASE medicines\_db;

**2. Set the current database to " medicines\_db ".**

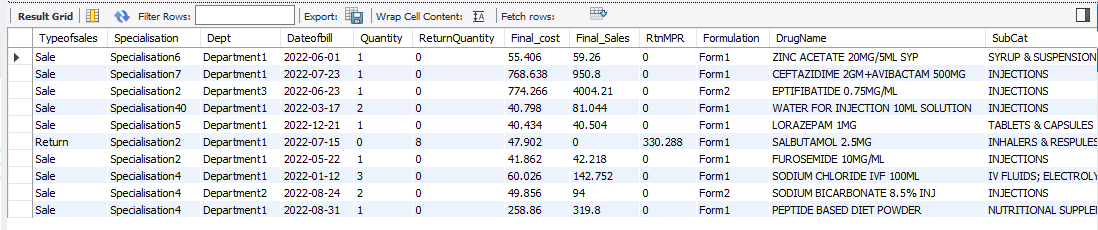
USE med\_inventory;

**3. Importing the table in CSV format into MySQL.**

Right-click on tables -> table data import wizard -> browse to select the table ‘medicine\_detail’.

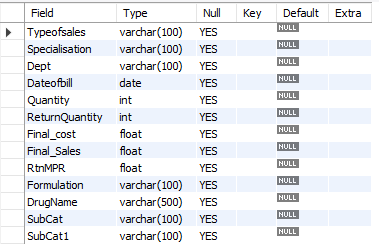
**4. Displaying the table.**

SELECT \* FROM medicine\_detail LIMIT 10



**5. Checking the schema of the dataset to ensure that all columns have the correct format.**

DESCRIBE medicine\_detail;



**6. Counting the missing and non-missing values for each column and the total number of rows in the 'medicine\_detail' table.**

SELECT

COUNT(CASE WHEN TRIM(Typeofsales) = '' OR Typeofsales IS NULL THEN 1 END) AS

typeofsales\_missing,

COUNT(CASE WHEN TRIM(Typeofsales) <> '' AND Typeofsales IS NOT NULL THEN 1 END) AS

typeofsales\_non\_missing,

count(case when trim(Typeofsales) = '' or Typeofsales is null then 1 end) as Typeofsales\_missing,

count(case when trim(Typeofsales) <> '' or Typeofsales is not null then 1 end) as Typeofsales\_non\_missing,

COUNT(CASE WHEN TRIM(Specialisation) = '' OR Specialisation IS NULL THEN 1 END) AS

specialisation\_missing,

COUNT(CASE WHEN TRIM(Specialisation) <> '' AND Specialisation IS NOT NULL THEN 1 END) AS

specialisation\_non\_missing ,

COUNT(CASE WHEN TRIM(Dept) = '' OR Dept IS NULL THEN 1 END) AS dept\_missing,

COUNT(CASE WHEN TRIM(Dept) <> '' AND Dept IS NOT NULL THEN 1 END) AS dept\_non\_missing,

COUNT(CASE WHEN TRIM(Dateofbill) = '' OR Dateofbill IS NULL THEN 1 END) AS dateofbill\_missing,

COUNT(CASE WHEN TRIM(Dateofbill) <> '' AND Dateofbill IS NOT NULL THEN 1 END) AS

dateofbill\_non\_missing,

COUNT(CASE WHEN Quantity IS NULL THEN 1 END) AS quantity\_missing,

COUNT(CASE WHEN Quantity IS NOT NULL THEN 1 END) AS quantity\_non\_missing,

COUNT(CASE WHEN ReturnQuantity IS NULL THEN 1 END) AS returnquantity\_missing,

COUNT(CASE WHEN ReturnQuantity IS NOT NULL THEN 1 END) AS returnquantity\_non\_missing,

COUNT(CASE WHEN Final\_Cost IS NULL THEN 1 END) AS final\_cost\_missing,

COUNT(CASE WHEN Final\_Cost IS NOT NULL THEN 1 END) AS final\_cost\_non\_missing,

COUNT(CASE WHEN Final\_Sales IS NULL THEN 1 END) AS final\_sales\_missing,

COUNT(CASE WHEN Final\_Sales IS NOT NULL THEN 1 END) AS final\_sales\_non\_missing,

COUNT(CASE WHEN RtnMPR IS NULL THEN 1 END) AS rtnmrp\_missing,

COUNT(CASE WHEN RtnMPR IS NOT NULL THEN 1 END) AS rtnmrp\_non\_missing,

COUNT(CASE WHEN TRIM(Formulation) = '' OR Formulation IS NULL THEN 1 END) AS

formulation\_missing,

COUNT(CASE WHEN TRIM(Formulation) <> '' AND Formulation IS NOT NULL THEN 1 END) AS

formulation\_non\_missing,

COUNT(CASE WHEN TRIM(DrugName) = '' OR DrugName IS NULL THEN 1 END) AS

drugname\_missing,

COUNT(CASE WHEN TRIM(DrugName) <> '' AND DrugName IS NOT NULL THEN 1 END) AS

drugname\_non\_missing,

COUNT(CASE WHEN TRIM(SubCat) = '' OR SubCat IS NULL THEN 1 END) AS subcat\_missing,

COUNT(CASE WHEN TRIM(SubCat) <> '' AND SubCat IS NOT NULL THEN 1 END) AS

subcat\_non\_missing,

COUNT(CASE WHEN TRIM(SubCat1) = '' OR SubCat1 IS NULL THEN 1 END) AS subcat1\_missing,

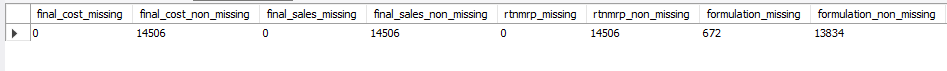
COUNT(CASE WHEN TRIM(SubCat1) <> '' AND SubCat1 IS NOT NULL THEN 1 END) AS

subcat1\_non\_missing,

COUNT(\*) AS total\_rows

FROM medicine\_detail;

**Output:**

**Observation: Missing values have been identified in the following columns: Formulation(672), DrugName(1705), SubCat(1705), and SubCat1(1729).**

**8. Replacing the missing values with ‘unknown’ in the columns Formulation, DrugName, SubCat and SubCat1.**

UPDATE medicine\_detail

SET Formulation = CASE WHEN Formulation = '' THEN 'unknown' ELSE Formulation END;

UPDATE medicine\_detail

SET DrugName = CASE WHEN DrugName = '' THEN 'unknown' ELSE DrugName END;

UPDATE medicine\_detail

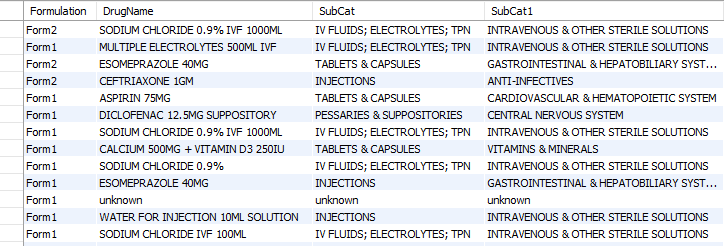
SET SubCat = CASE WHEN SubCat = '' THEN 'unknown' ELSE SubCat END;

UPDATE medicine\_detail

SET SubCat1 = CASE WHEN SubCat1 = '' THEN 'unknown' ELSE SubCat1 END;

**Showing the columns after replacing the missing values with ‘unknown’:**

select Formulation, DrugName,SubCat,SubCat1 from medicine\_detail;



**9. Creating a new table called `missing\_values` by selecting rows from `medicine\_detail` where any of the columns (`Formulation`, `DrugName`, `SubCat`, or `SubCat1`) has the value 'unknown'.**

CREATE TABLE missing\_values AS

SELECT \*

FROM medicine\_detail

WHERE Formulation = 'unknown'

OR DrugName = 'unknown'

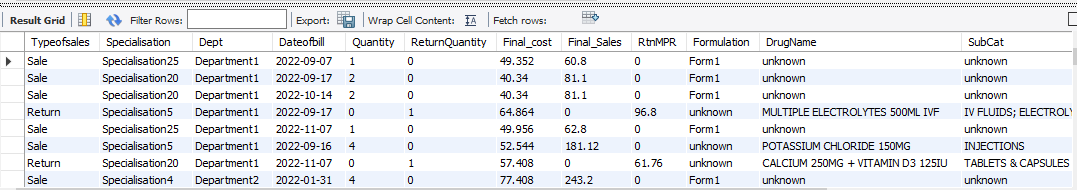
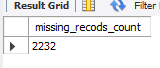
OR SubCat = 'unknown'

OR SubCat1 = 'unknown';

**Showing missing\_values table and count of records with at least one or more missing values:**

select \* from missing\_values;

select count(\*) as missing\_recods\_count from missing\_values;

**10. Identifying duplicate rows based on Typeofsales, Specialisation, Dept, Dateofbill, Quantity, ReturnQuantity, Final\_cost, Final\_Sales, RtnMPR, Formulation, DrugName, SubCat, SubCat1, Dateofbill, where the DrugName is 'unknown',** **SubCat is 'unknown',** **subcat1 is 'unknown' and Formulation is 'unknown'.**

select Typeofsales, Specialisation, Dept, Dateofbill, Quantity, ReturnQuantity,

Final\_cost, Final\_Sales, RtnMPR, Formulation, DrugName, SubCat, SubCat1, count(\*)

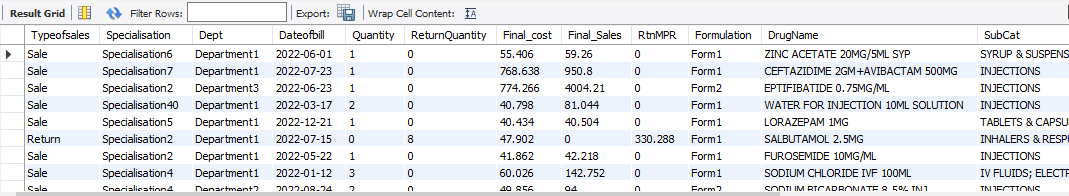
from medicine\_detail

WHERE Formulation<>'unknown'and DrugName <> 'unknown'and SubCat <> 'unknown' and subcat1 <>'unknown'

GROUP BY Typeofsales, Specialisation, Dept, Dateofbill, Quantity, ReturnQuantity,

Final\_cost, Final\_Sales, RtnMPR, Formulation, DrugName, SubCat, SubCat1

HAVING COUNT(\*) > 1;



**11. Removing the duplicate rows from medicine\_detail table and counting the remaining rows.**

DELETE FROM medicine\_detail

WHERE (Typeofsales, Specialisation, Dept, Dateofbill, Quantity, ReturnQuantity,

Final\_cost, Final\_Sales, RtnMPR, Formulation, DrugName, SubCat, SubCat1) IN (

SELECT t.Typeofsales, t.Specialisation, t.Dept,t.Dateofbill,t.Quantity,

t.ReturnQuantity,t.Final\_cost,t.Final\_Sales,t.RtnMPR,t.Formulation,t.DrugName,t.SubCat,t.SubCat1

FROM (

SELECT Typeofsales, Specialisation, Dept, Dateofbill, Quantity, ReturnQuantity,

Final\_cost, Final\_Sales, RtnMPR, Formulation, DrugName, SubCat, SubCat1

FROM medicine\_detail

WHERE Formulation<>'unknown'and DrugName <> 'unknown'and SubCat <> 'unknown' and subcat1 <>'unknown'

GROUP BY Typeofsales, Specialisation, Dept, Dateofbill, Quantity, ReturnQuantity,

Final\_cost, Final\_Sales, RtnMPR, Formulation, DrugName, SubCat, SubCat1

HAVING COUNT(\*) > 1

) AS t

);

SELECT COUNT(\*) AS total\_rows FROM medicine\_detail;



**12. In a normal distribution, approximately 68%, 95%, and 99.7% of the data falls within one, two, and three standard deviations of the mean respectively. By using three standard deviations as the threshold for removing outliers, we are effectively removing data points that are more than three standard deviations away from the mean.**

CREATE TABLE new\_table AS

SELECT \*

FROM medicine\_detail

WHERE Quantity BETWEEN

(SELECT AVG(Quantity) - 3 \* STDDEV(Quantity) FROM medicine\_detail)

AND

(SELECT AVG(Quantity) + 3 \* STDDEV(Quantity) FROM medicine\_detail)

AND ReturnQuantity BETWEEN

(SELECT AVG(ReturnQuantity) - 3 \* STDDEV(ReturnQuantity) FROM medicine\_detail)

AND

(SELECT AVG(ReturnQuantity) + 3 \* STDDEV(ReturnQuantity) FROM medicine\_detail)

AND Final\_Cost BETWEEN

(SELECT AVG(Final\_Cost) - 3 \* STDDEV(Final\_Cost) FROM medicine\_detail)

AND

(SELECT AVG(Final\_Cost) + 3 \* STDDEV(Final\_Cost) FROM medicine\_detail)

AND Final\_Sales BETWEEN

(SELECT AVG(Final\_Sales) - 3 \* STDDEV(Final\_Sales) FROM medicine\_detail)

AND

(SELECT AVG(Final\_Sales) + 3 \* STDDEV(Final\_Sales) FROM medicine\_detail)

AND RtnMPR BETWEEN

(SELECT AVG(RtnMPR) - 3 \* STDDEV(RtnMPR) FROM medicine\_detail)

AND

(SELECT AVG(RtnMPR) + 3 \* STDDEV(RtnMPR) FROM medicine\_detail);

select count(\*) as total\_rows from new\_table;

