Medical inventory optimization

Exploratory Data Analysis (SQL) by Pratiksha Saheb

Software: MySQL Workbench

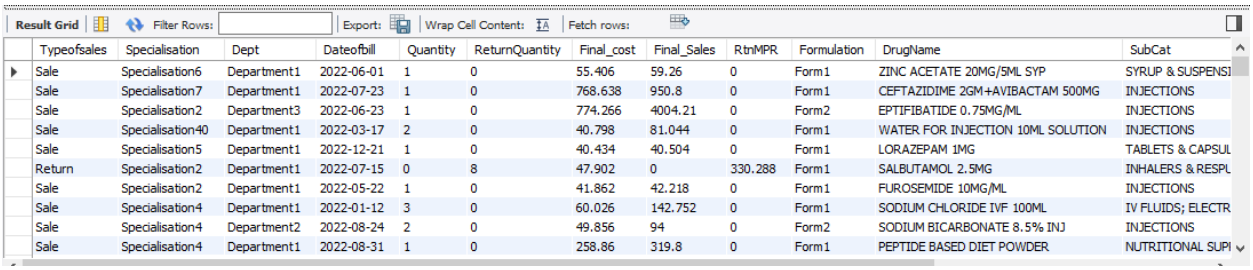
Business decisions based on the unclean 'medicines_db' data

1. Set the current database to "medicines_db".

```
use medicines_db;
```

2. Displaying the Table

```
select * from medicine_detail limit 10;
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Typeofsales	Specialisation	Dept	Dateofbill	Quantity	ReturnQuantity	Final_cost	Final_Sales	RtnMPR	Formulation	DrugName	SubCat
Sale	Specialisation6	Department1	2022-06-01	1	0	55.406	59.26	0	Form1	ZINC ACETATE 20MG/5ML SYP	SYRUP & SUSPENSI
Sale	Specialisation7	Department1	2022-07-23	1	0	768.638	950.8	0	Form1	CEFTAZIDIME 2GM+AVIBACTAM 500MG	INJECTIONS
Sale	Specialisation2	Department3	2022-06-23	1	0	774.266	4004.21	0	Form2	EPTIFIBATIDE 0.75MG/ML	INJECTIONS
Sale	Specialisation40	Department1	2022-03-17	2	0	40.798	81.044	0	Form1	WATER FOR INJECTION 10ML SOLUTION	INJECTIONS
Sale	Specialisation5	Department1	2022-12-21	1	0	40.434	40.504	0	Form1	LORAZEPAM 1MG	TABLETS & CAPSUL
Return	Specialisation2	Department1	2022-07-15	0	8	47.902	0	330.288	Form1	SALBUTAMOL 2.5MG	INHALERS & RESPL
Sale	Specialisation2	Department1	2022-05-22	1	0	41.862	42.218	0	Form1	FUROSEMIDE 10MG/ML	INJECTIONS
Sale	Specialisation4	Department1	2022-01-12	3	0	60.026	142.752	0	Form1	SODIUM CHLORIDE IVF 100ML	IV FLUIDS; ELECTR
Sale	Specialisation4	Department2	2022-08-24	2	0	49.856	94	0	Form2	SODIUM BICARBONATE 8.5% INJ	INJECTIONS
Sale	Specialisation4	Department1	2022-08-31	1	0	258.86	319.8	0	Form1	PEPTIDE BASED DIET POWDER	NUTRITIONAL SUPI

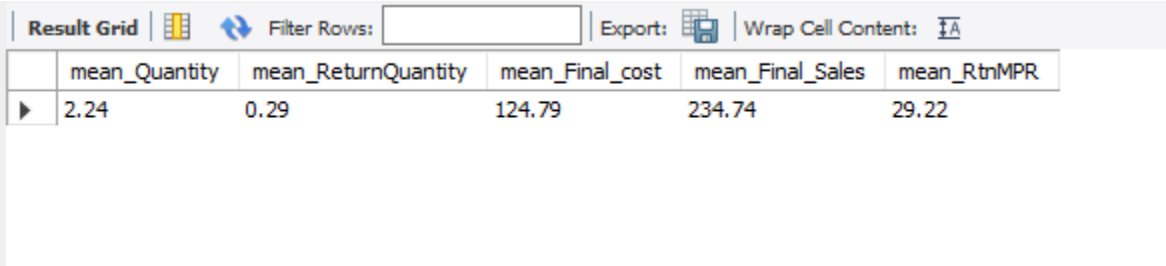
3. Calculating the first moment (measures of central tendency such as mean, median, mode) for the dataset.

Mean:

```
select
round(avg(Quantity),2) as mean_Quantity,
round(avg(ReturnQuantity),2) as mean_ReturnQuantity,
```

```
round(avg(Final_cost),2) as mean_Final_cost,
round(avg(Final_Sales),2) as mean_Final_Sales,
round(avg(RtnMPR),2) as mean_RtnMPR
from
medicine_detail;
```

Output:



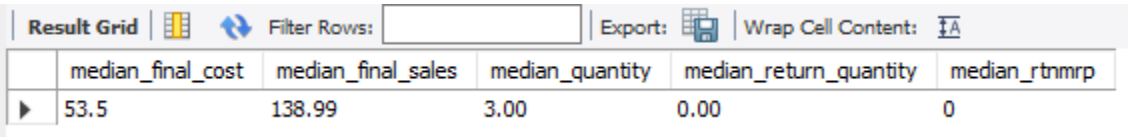
mean_Quantity	mean_ReturnQuantity	mean_Final_cost	mean_Final_Sales	mean_RtnMPR
2.24	0.29	124.79	234.74	29.22

Median:

```
with ranked as
(
select Final_cost,
Final_Sales,
Quantity,
ReturnQuantity,
RtnMPR,
row_number() over (order by Final_cost) as r,
count(*) over () as c
from medicine_detail),
median as
(
select Final_cost,
Final_Sales,
Quantity,
ReturnQuantity,
```

```
RtnMPR
from ranked
where r in (floor((c+1)/2),ceiling((c+1)/2))
)
select round(avg(Final_cost), 2),
round(avg(Final_Sales),2),
round(avg(Quantity),2),
round(avg(ReturnQuantity),2),
round(avg(RtnMPR),2)
from median;
```

Output:



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

median_final_cost	median_final_sales	median_quantity	median_return_quantity	median_rtnmrp
53.5	138.99	3.00	0.00	0

Mode:

```
select
mode_Quantity.mode_value as mode_Quantity,
mode_Quantity.mode_count as mode_Quantity_count,
mode_ReturnQuantity.mode_value as mode_ReturnQuantity,
mode_ReturnQuantity.mode_count as mode_ReturnQuantity_count,
mode_Final_Sales.mode_value as mode_Final_Sales,
mode_Final_Sales.mode_count as mode_Final_Sales_count,
mode_Final_cost.mode_value as mode_Final_cost,
mode_Final_cost.mode_count as mode_Final_cost_count,
mode_RtnMPR.mode_value as mode_RtnMPR,
mode_RtnMPR.mode_count as mode_RtnMPR_count
from(
 SELECT Quantity AS mode_value, COUNT(*) AS mode_count
 FROM medicine_detail
```

```
GROUP BY Quantity
ORDER BY COUNT(*) DESC
LIMIT 1 ) as mode_Quantity,
( SELECT ReturnQuantity AS mode_value, COUNT(*) AS mode_count
FROM medicine_detail
GROUP BY Quantity
ORDER BY COUNT(*) DESC
LIMIT 1
) as mode_ReturnQuantity,
(SELECT final_cost AS mode_value, COUNT(*) AS mode_count
FROM medicine_detail
GROUP BY Quantity
ORDER BY COUNT(*) DESC
LIMIT 1
) as mode_Final_cost,
(SELECT Final_Sales AS mode_value, COUNT(*) AS mode_count
FROM medicine_detail
GROUP BY Quantity
ORDER BY COUNT(*) DESC
LIMIT 1
) as mode_Final_Sales,
( select RtnMPR AS mode_value, COUNT(*) AS mode_count
FROM medicine_detail
GROUP BY Quantity
ORDER BY COUNT(*) DESC
LIMIT 1
) as mode_RtnMPR;
```

Output:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	mode_Quantity	mode_Quantity_count	mode_ReturnQuantity	mode_ReturnQuantity_count	mode_Final_Sales	mode_Final_Sales_count
▶	1	7151	0	7151	59.26	7151

mode_Final_cost	mode_Final_cost_count	mode_RtnMPR	mode_RtnMPR_count
55.406	7151	0	7151

4. Calculating the second moment (measures of dispersion such as variance, standard deviation, range) for the dataset.

Variance:

```
SELECT
 ROUND(VARIANCE(Quantity), 2) AS variance_quantity,
 ROUND(VARIANCE(ReturnQuantity), 2) AS variance_return_quantity,
 ROUND(VARIANCE(Final_Cost), 2) AS variance_final_cost,
 ROUND(VARIANCE(Final_Sales), 2) AS variance_final_sales,
 ROUND(VARIANCE(RtnMPR), 2) AS variance_rtnmrp
FROM medicine_detail;
```

Output:

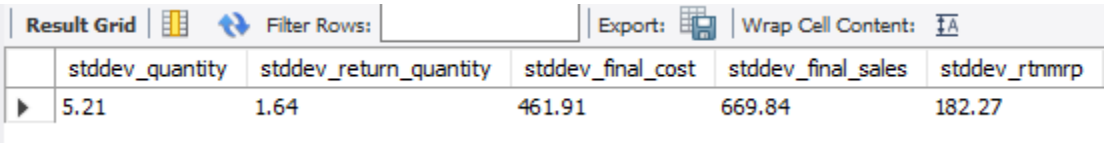
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	variance_quantity	variance_return_quantity	variance_final_cost	variance_final_sales	variance_rtnmrp
▶	27.17	2.68	213363.23	448689.94	33221.34

Standard Deviation:

```
SELECT
 ROUND(STDDEV(Quantity), 2) AS stddev_quantity,
 ROUND(STDDEV(ReturnQuantity), 2) AS stddev_return_quantity,
 ROUND(STDDEV(Final_Cost), 2) AS stddev_final_cost,
 ROUND(STDDEV(Final_Sales), 2) AS stddev_final_sales,
 ROUND(STDDEV(RtnMPR), 2) AS stddev_rtnmrp
FROM medicine_detail;
```

Output:



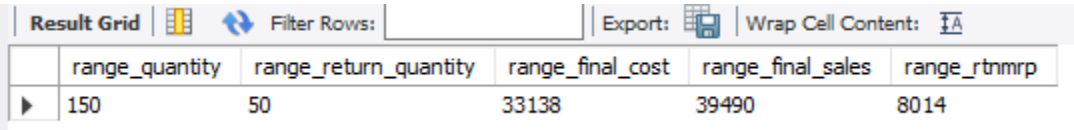
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

stddev_quantity	stddev_return_quantity	stddev_final_cost	stddev_final_sales	stddev_rtnmrp
5.21	1.64	461.91	669.84	182.27

Range:

```
SELECT
 MAX(Quantity) - MIN(Quantity) AS range_quantity,
 MAX(ReturnQuantity) - MIN(ReturnQuantity) AS range_return_quantity,
 MAX(Final_Cost) - MIN(Final_Cost) AS range_final_cost,
 MAX(Final_Sales) - MIN(Final_Sales) AS range_final_sales,
 MAX(RtnMPR) - MIN(RtnMPR) AS range_rtnmrp
FROM medicine_detail;
```

Output:



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

range_quantity	range_return_quantity	range_final_cost	range_final_sales	range_rtnmrp
150	50	33138	39490	8014

5. Calculating the third moment (skewness) for the dataset.

Skewness:

```
select Quantity_skewness1.Quantity_skewness,
ReturnQuantity_skewness1.ReturnQuantity_skewness,
Final_cost_skewness1.Final_cost_skewness,
Final_Sales_skewness1.Final_Sales_skewness,
RtnMPR_skewness1.RtnMPR_skewness
from
(SELECT
ROUND((SUM(POW(Quantity - (SELECT AVG(Quantity) FROM medicine_detail), 3)) / (COUNT(*) *
POW(STDDEV(Quantity), 3))), 2) AS Quantity_skewness
FROM medicine_detail) Quantity_skewness1,
```

```
(select
round((sum(pow(ReturnQuantity - (select avg(ReturnQuantity) from medicine_detail),3))/(count(*) *
pow(stddev(ReturnQuantity),3))),2) as  ReturnQuantity_skewness from medicine_detail)
ReturnQuantity_skewness1,
(select
round((sum(pow(Final_cost - (select avg(Final_cost) from medicine_detail),3))/(count(*) *
pow(stddev(Final_cost),3))),2) as  Final_cost_skewness from medicine_detail)
Final_cost_skewness1,
 (
select
round((sum(pow(Final_Sales - (select avg(Final_Sales) from medicine_detail),3))/(count(*) *
pow(stddev(Final_Sales),3))),2) as  Final_Sales_skewness from medicine_detail)
Final_Sales_skewness1,
(select
round((sum(pow(RtnMPR - (select avg(RtnMPR) from medicine_detail),3))/(count(*) *
pow(stddev(RtnMPR),3))),2) as  RtnMPR_skewness from medicine_detail)
RtnMPR_skewness1;
```

Output:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Quantity_skewness	ReturnQuantity_skewness	Final_cost_skewness	Final_Sales_skewness	RtnMPR_skewness
11.4	17.09	34.49	20.79	15.65

6. Calculating the fourth moment (kurtosis) for the dataset.

Kurtosis:

```
select
kurtosis_quantity1.kurtosis_quantity,
kurtosis_Returnquantity1.kurtosis_Returnquantity,
kurtosis_Final_cost1.kurtosis_Final_cost,
kurtosis_Final_Sales1.kurtosis_Final_Sales,
```

```
kurtosis_RtnMPR1.kurtosis_RtnMPR

from

(SELECT

ROUND((SUM(POWER(Quantity - (SELECT AVG(Quantity) FROM medicine_detail), 4)) /
(COUNT(Quantity) * POWER(STDDEV(Quantity),

4))), 2) AS kurtosis_quantity

from medicine_detail) kurtosis_quantity1,

(SELECT

ROUND((SUM(POWER(ReturnQuantity - (SELECT AVG(ReturnQuantity) FROM medicine_detail), 4)) /
(COUNT(ReturnQuantity) * POWER(STDDEV(ReturnQuantity),

4))), 2) AS kurtosis_Returnquantity

from medicine_detail) kurtosis_Returnquantity1,

(SELECT

ROUND((SUM(POWER(Final_cost - (SELECT AVG(Final_cost) FROM medicine_detail), 4)) /
(COUNT(Final_cost) * POWER(STDDEV(Final_cost),

4))), 2) AS kurtosis_Final_cost

from medicine_detail) kurtosis_Final_cost1,

(SELECT

ROUND((SUM(POWER(Final_Sales - (SELECT AVG(Final_Sales) FROM medicine_detail), 4)) /
(COUNT(Final_Sales) * POWER(STDDEV(Final_Sales),

4))), 2) AS kurtosis_Final_Sales

from medicine_detail) kurtosis_Final_Sales1,

(SELECT

ROUND((SUM(POWER(RtnMPR - (SELECT AVG(RtnMPR) FROM medicine_detail), 4)) / (COUNT(RtnMPR)
* POWER(STDDEV(RtnMPR),

4))), 2) AS kurtosis_RtnMPR

from medicine_detail) kurtosis_RtnMPR1;
```

Output:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

kurtosis_quantity	kurtosis_Returnquantity	kurtosis_Final_cost	kurtosis_Final_Sales	kurtosis_RtnMPR
182.62	410.21	2037.79	940.42	399.9

Pre-processing Code (SQL)

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
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Typeofsales	Specialisation	Dept	Dateofbill	Quantity	ReturnQuantity	Final_cost	Final_Sales	RtnMPR	Formulation	DrugName	SubCat
Sale	Specialisation6	Department1	2022-06-01	1	0	55.406	59.26	0	Form1	ZINC ACETATE 20MG/5ML SYP	SYRUP & SUSPENSION
Sale	Specialisation7	Department1	2022-07-23	1	0	768.638	950.8	0	Form1	CEFTAZIDIME 2GM+AVIBACTAM 500MG	INJECTIONS
Sale	Specialisation2	Department3	2022-06-23	1	0	774.266	4004.21	0	Form2	EPTIFIBATIDE 0.75MG/ML	INJECTIONS
Sale	Specialisation40	Department1	2022-03-17	2	0	40.798	81.044	0	Form1	WATER FOR INJECTION 10ML SOLUTION	INJECTIONS
Sale	Specialisation5	Department1	2022-12-21	1	0	40.434	40.504	0	Form1	LORAZEPAM 1MG	TABLETS & CAPSULES
Return	Specialisation2	Department1	2022-07-15	0	8	47.902	0	330.288	Form1	SALBUTAMOL 2.5MG	INHALERS & RESPULES
Sale	Specialisation2	Department1	2022-05-22	1	0	41.862	42.218	0	Form1	FUROSEMIDE 10MG/ML	INJECTIONS
Sale	Specialisation4	Department1	2022-01-12	3	0	60.026	142.752	0	Form1	SODIUM CHLORIDE IVF 100ML	IV FLUIDS; ELECTROLY
Sale	Specialisation4	Department2	2022-08-24	2	0	49.856	94	0	Form2	SODIUM BICARBONATE 8.5% INJ	INJECTIONS
Sale	Specialisation4	Department1	2022-08-31	1	0	258.86	319.8	0	Form1	PEPTIDE BASED DIET POWDER	NUTRITIONAL SUPPLE

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

```
DESCRIBE medicine_detail;
```

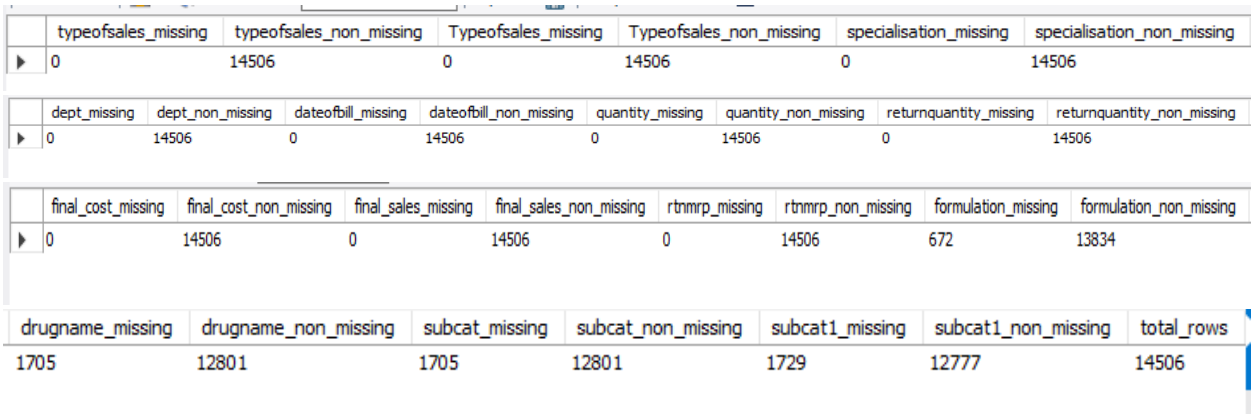
Field	Type	Null	Key	Default	Extra
Typeofsales	varchar(100)	YES		NULL	
Specialisation	varchar(100)	YES		NULL	
Dept	varchar(100)	YES		NULL	
Dateofbill	date	YES		NULL	
Quantity	int	YES		NULL	
ReturnQuantity	int	YES		NULL	
Final_cost	float	YES		NULL	
Final_Sales	float	YES		NULL	
RtnMPR	float	YES		NULL	
Formulation	varchar(100)	YES		NULL	
DrugName	varchar(500)	YES		NULL	
SubCat	varchar(100)	YES		NULL	
SubCat1	varchar(100)	YES		NULL	

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:



typeofsales_missing	typeofsales_non_missing	Typeofsales_missing	Typeofsales_non_missing	specialisation_missing	specialisation_non_missing
0	14506	0	14506	0	14506

dept_missing	dept_non_missing	dateofbill_missing	dateofbill_non_missing	quantity_missing	quantity_non_missing	returnquantity_missing	returnquantity_non_missing
0	14506	0	14506	0	14506	0	14506

final_cost_missing	final_cost_non_missing	final_sales_missing	final_sales_non_missing	rtnmrp_missing	rtnmrp_non_missing	formulation_missing	formulation_non_missing
0	14506	0	14506	0	14506	672	13834

drugname_missing	drugname_non_missing	subcat_missing	subcat_non_missing	subcat1_missing	subcat1_non_missing	total_rows
1705	12801	1705	12801	1729	12777	14506

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;
```

Formulation	DrugName	SubCat	SubCat1
Form2	SODIUM CHLORIDE 0.9% IVF 1000ML	IV FLUIDS; ELECTROLYTES; TPN	INTRAVENOUS & OTHER STERILE SOLUTIONS
Form1	MULTIPLE ELECTROLYTES 500ML IVF	IV FLUIDS; ELECTROLYTES; TPN	INTRAVENOUS & OTHER STERILE SOLUTIONS
Form2	ESOMEPRAZOLE 40MG	TABLETS & CAPSULES	GASTROINTESTINAL & HEPATOBILIARY SYST...
Form2	CEFTRIAXONE 1GM	INJECTIONS	ANTI-INFECTIVES
Form1	ASPIRIN 75MG	TABLETS & CAPSULES	CARDIOVASCULAR & HEMATOPOIETIC SYSTEM
Form1	DICLOFENAC 12.5MG SUPPOSITORY	PESSARIES & SUPPOSITORIES	CENTRAL NERVOUS SYSTEM
Form1	SODIUM CHLORIDE 0.9% IVF 1000ML	IV FLUIDS; ELECTROLYTES; TPN	INTRAVENOUS & OTHER STERILE SOLUTIONS
Form1	CALCIUM 500MG + VITAMIN D3 250IU	TABLETS & CAPSULES	VITAMINS & MINERALS
Form1	SODIUM CHLORIDE 0.9%	IV FLUIDS; ELECTROLYTES; TPN	INTRAVENOUS & OTHER STERILE SOLUTIONS
Form1	ESOMEPRAZOLE 40MG	INJECTIONS	GASTROINTESTINAL & HEPATOBILIARY SYST...
Form1	unknown	unknown	unknown
Form1	WATER FOR INJECTION 10ML SOLUTION	INJECTIONS	INTRAVENOUS & OTHER STERILE SOLUTIONS
Form1	SODIUM CHLORIDE IVF 100ML	IV FLUIDS; ELECTROLYTES; TPN	INTRAVENOUS & OTHER STERILE SOLUTIONS

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;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Typeofsales	Specialisation	Dept	Dateofbill	Quantity	ReturnQuantity	Final_cost	Final_Sales	RtnMPR	Formulation	DrugName	SubCat
Sale	Specialisation25	Department1	2022-09-07	1	0	49.352	60.8	0	Form1	unknown	unknown
Sale	Specialisation20	Department1	2022-09-17	2	0	40.34	81.1	0	Form1	unknown	unknown
Sale	Specialisation20	Department1	2022-10-14	2	0	40.34	81.1	0	Form1	unknown	unknown
Return	Specialisation5	Department1	2022-09-17	0	1	64.864	0	96.8	unknown	MULTIPLE ELECTROLYTES 500ML IVF	IV FLUIDS; ELECTROL
Sale	Specialisation25	Department1	2022-11-07	1	0	49.956	62.8	0	Form1	unknown	unknown
Sale	Specialisation5	Department1	2022-09-16	4	0	52.544	181.12	0	unknown	POTASSIUM CHLORIDE 150MG	INJECTIONS
Return	Specialisation20	Department1	2022-11-07	0	1	57.408	0	61.76	unknown	CALCIUM 250MG + VITAMIN D3 125IU	TABLETS & CAPSULES
Sale	Specialisation4	Department2	2022-01-31	4	0	77.408	243.2	0	Form1	unknown	unknown

Result Grid | Filter

missing_recods_count
2232

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;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Typeofsales	Specialisation	Dept	Dateofbill	Quantity	ReturnQuantity	Final_cost	Final_Sales	RtnMPR	Formulation	DrugName	SubCat
Sale	Specialisation6	Department1	2022-06-01	1	0	55.406	59.26	0	Form1	ZINC ACETATE 20MG/5ML SYP	SYRUP & SUSPENS
Sale	Specialisation7	Department1	2022-07-23	1	0	768.638	950.8	0	Form1	CEFTAZIDIME 2GM+AVIBACTAM 500MG	INJECTIONS
Sale	Specialisation2	Department3	2022-06-23	1	0	774.266	4004.21	0	Form2	EPTIFIBATIDE 0.75MG/ML	INJECTIONS
Sale	Specialisation40	Department1	2022-03-17	2	0	40.798	81.044	0	Form1	WATER FOR INJECTION 10ML SOLUTION	INJECTIONS
Sale	Specialisation5	Department1	2022-12-21	1	0	40.434	40.504	0	Form1	LORAZEPAM 1MG	TABLETS & CAPSU
Return	Specialisation2	Department1	2022-07-15	0	8	47.902	0	330.288	Form1	SALBUTAMOL 2.5MG	INHALERS & RESPI
Sale	Specialisation2	Department1	2022-05-22	1	0	41.862	42.218	0	Form1	FUROSEMIDE 10MG/ML	INJECTIONS
Sale	Specialisation4	Department1	2022-01-12	3	0	60.026	142.752	0	Form1	SODIUM CHLORIDE IVF 100ML	IV FLUIDS; ELECTF
Sale	Specialisation4	Department2	2022-08-24	2	0	49.856	94	0	Form2	SODIUM BICARBONATE 8.5% INJ	INJECTIONS

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;
```

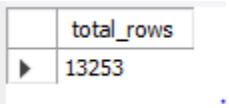
	total_rows
▶	13799

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;
```



	total_rows
▶	13253

Exploratory Data Analysis (SQL)

Software: MySQL Workbench

Business decisions based on the clean ‘medicines_db’ data

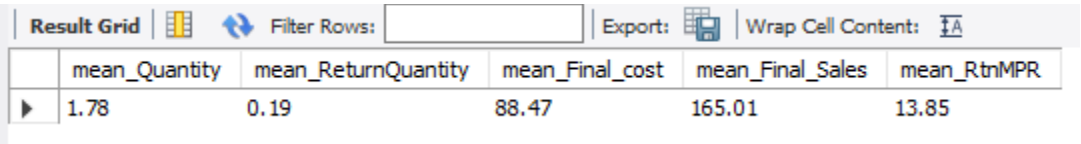
1. Calculating the first moment (measures of central tendency such as mean, median, mode) for the dataset.

Mean :

```
select
round(avg(Quantity),2) as mean_Quantity,
round(avg(ReturnQuantity),2) as mean_ReturnQuantity,
round(avg(Final_cost),2) as mean_Final_cost,
round(avg(Final_Sales),2) as mean_Final_Sales,
round(avg(RtnMPR),2) as mean_RtnMPR
```

from

new_table;



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

mean_Quantity	mean_ReturnQuantity	mean_Final_cost	mean_Final_Sales	mean_RtnMPR
1.78	0.19	88.47	165.01	13.85

Median:

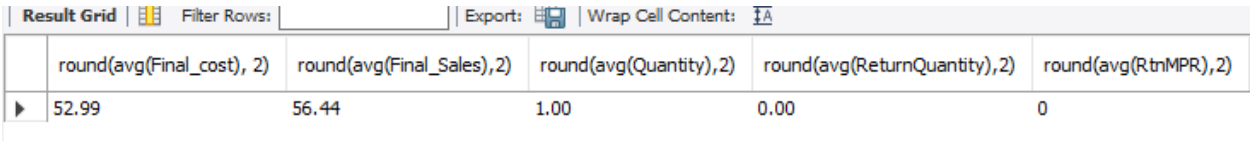
```
with ranked as
(
select Final_cost,
Final_Sales,
Quantity,
ReturnQuantity,
RtnMPR,
row_number() over (order by Final_cost) as r,
count(*) over () as c
from new_table),
median as
(
select Final_cost,
Final_Sales,
Quantity,
ReturnQuantity,
RtnMPR
from ranked
where r in (floor((c+1)/2),ceiling((c+1)/2))
)
select round(avg(Final_cost), 2),
round(avg(Final_Sales),2),
```

round(avg(Quantity),2),

round(avg(ReturnQuantity),2),

round(avg(RtnMPR),2)

from median;



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

round(avg(Final_cost), 2)	round(avg(Final_Sales),2)	round(avg(Quantity),2)	round(avg(ReturnQuantity),2)	round(avg(RtnMPR),2)
52.99	56.44	1.00	0.00	0

Mode:

select

mode_Quantity.mode_value as mode_Quantity,

mode_Quantity.mode_count as mode_Quantity_count,

mode_ReturnQuantity.mode_value as mode_ReturnQuantity,

mode_ReturnQuantity.mode_count as mode_ReturnQuantity_count,

mode_Final_Sales.mode_value as mode_Final_Sales,

mode_Final_Sales.mode_count as mode_Final_Sales_count,

mode_Final_cost.mode_value as mode_Final_cost,

mode_Final_cost.mode_count as mode_Final_cost_count,

mode_RtnMPR.mode_value as mode_RtnMPR,

mode_RtnMPR.mode_count as mode_RtnMPR_count

from(

SELECT Quantity AS mode_value, COUNT(*) AS mode_count

FROM new_table

GROUP BY Quantity

ORDER BY COUNT(*) DESC

LIMIT 1) as mode_Quantity,

(SELECT ReturnQuantity AS mode_value, COUNT(*) AS mode_count

FROM new_table

```
GROUP BY Quantity
ORDER BY COUNT(*) DESC
LIMIT 1
) as mode_ReturnQuantity,
(SELECT final_cost AS mode_value, COUNT(*) AS mode_count
FROM new_table
GROUP BY Quantity
ORDER BY COUNT(*) DESC
LIMIT 1
) as mode_Final_cost,
(SELECT Final_Sales AS mode_value, COUNT(*) AS mode_count
FROM new_table
GROUP BY Quantity
ORDER BY COUNT(*) DESC
LIMIT 1
) as mode_Final_Sales,
( select RtnMPR AS mode_value, COUNT(*) AS mode_count
FROM new_table
GROUP BY Quantity
ORDER BY COUNT(*) DESC
LIMIT 1
) as mode_RtnMPR;
```

mode_Quantity	mode_Quantity_count	mode_ReturnQuantity	mode_ReturnQuantity_count	mode_Final_Sales	mode_Final_Sales_count
1	6744	0	6744	60.8	6744

mode_Final_cost	mode_Final_cost_count	mode_RtnMPR	mode_RtnMPR_count
49.352	6744	0	6744

2. Calculating the second moment (measures of dispersion such as variance, standard deviation, range) for the dataset.

Variance:

```
SELECT
 ROUND(VARIANCE(Quantity), 2) AS variance_quantity,
 ROUND(VARIANCE(ReturnQuantity), 2) AS variance_return_quantity,
 ROUND(VARIANCE(Final_Cost), 2) AS variance_final_cost,
 ROUND(VARIANCE(Final_Sales), 2) AS variance_final_sales,
 ROUND(VARIANCE(RtnMPR), 2) AS variance_rtnmrp
FROM new_table;
```

variance_quantity	variance_return_quantity	variance_final_cost	variance_final_sales	variance_rtnmrp
3.32	0.41	17736.69	71134.19	2496.22

```
SELECT
 ROUND(STDDEV(Quantity), 2) AS stddev_quantity,
 ROUND(STDDEV(ReturnQuantity), 2) AS stddev_return_quantity,
 ROUND(STDDEV(Final_Cost), 2) AS stddev_final_cost,
 ROUND(STDDEV(Final_Sales), 2) AS stddev_final_sales,
 ROUND(STDDEV(RtnMPR), 2) AS stddev_rtnmrp
FROM new_table;
```

stddev_quantity	stddev_return_quantity	stddev_final_cost	stddev_final_sales	stddev_rtnmrp
1.82	0.64	133.18	266.71	49.96

Range:

```
SELECT
 MAX(Quantity) - MIN(Quantity) AS range_quantity,
 MAX(ReturnQuantity) - MIN(ReturnQuantity) AS range_return_quantity,
 MAX(Final_Cost) - MIN(Final_Cost) AS range_final_cost,
```

```
 MAX(Final_Sales) - MIN(Final_Sales) AS range_final_sales,
 MAX(RtnMPR) - MIN(RtnMPR) AS range_rtnmrp
FROM new_table;
```

range_quantity	range_return_quantity	range_final_cost	range_final_sales	range_rtnmrp
17	5	1363.1099853515625	2143.639892578125	568.176025390625

3. Calculating the third moment (skewness) for the dataset.

Skewness:

```
select Quantity_skewness1.Quantity_skewness,
ReturnQuantity_skewness1.ReturnQuantity_skewness,
Final_cost_skewness1.Final_cost_skewness,
Final_Sales_skewness1.Final_Sales_skewness,
RtnMPR_skewness1.RtnMPR_skewness
from
(SELECT
ROUND((SUM(POW(Quantity - (SELECT AVG(Quantity) FROM new_table), 3)) / (COUNT(*) *
POW(STDDEV(Quantity), 3))), 2) AS Quantity_skewness
FROM new_table) Quantity_skewness1,
(select
round((sum(pow(ReturnQuantity - (select avg(ReturnQuantity) from new_table),3))/(count(*) *
pow(stddev(ReturnQuantity),3))),2) as  ReturnQuantity_skewness from new_table)
ReturnQuantity_skewness1,
(select
round((sum(pow(Final_cost - (select avg(Final_cost) from new_table),3))/(count(*) *
pow(stddev(Final_cost),3))),2) as  Final_cost_skewness from new_table)
Final_cost_skewness1,
 (
select
round((sum(pow(Final_Sales - (select avg(Final_Sales) from new_table),3))/(count(*) *
```

```
pow(stddev(Final_Sales),3))),2) as  Final_Sales_skewness from new_table)
Final_Sales_skewness1,
(select
round((sum(pow(RtnMPR - (select avg(RtnMPR) from new_table),3))/(count(*) *
pow(stddev(RtnMPR),3))),2) as  RtnMPR_skewness from new_table)
RtnMPR_skewness1;
```

Quantity_skewness	ReturnQuantity_skewness	Final_cost_skewness	Final_Sales_skewness	RtnMPR_skewness
2.91	4.23	5.86	4.3	5.11

4. Calculating the fourth moment (kurtosis) for the dataset.

Kurtosis:

```
select
kurtosis_quantity1.kurtosis_quantity,
kurtosis_Returnquantity1.kurtosis_Returnquantity,
kurtosis_Final_cost1.kurtosis_Final_cost,
kurtosis_Final_Sales1.kurtosis_Final_Sales,
kurtosis_RtnMPR1.kurtosis_RtnMPR
from
(SELECT
ROUND((SUM(POWER(Quantity - (SELECT AVG(Quantity) FROM new_table), 4)) / (COUNT(Quantity) *
POWER(STDDEV(Quantity),
4))), 2) AS kurtosis_quantity
from new_table) kurtosis_quantity1,
(SELECT
ROUND((SUM(POWER(ReturnQuantity - (SELECT AVG(ReturnQuantity) FROM new_table), 4)) /
(COUNT(ReturnQuantity) * POWER(STDDEV(ReturnQuantity),
4))), 2) AS kurtosis_Returnquantity
from new_table) kurtosis_Returnquantity1,
(SELECT
```

```
ROUND((SUM(POWER(Final_cost - (SELECT AVG(Final_cost) FROM new_table), 4)) / (COUNT(Final_cost)
* POWER(STDDEV(Final_cost),

4))), 2) AS kurtosis_Final_cost

from new_table) kurtosis_Final_cost1,

(SELECT

ROUND((SUM(POWER(Final_Sales - (SELECT AVG(Final_Sales) FROM new_table), 4)) /
(COUNT(Final_Sales) * POWER(STDDEV(Final_Sales),

4))), 2) AS kurtosis_Final_Sales

from new_table) kurtosis_Final_Sales1,

(SELECT

ROUND((SUM(POWER(RtnMPR - (SELECT AVG(RtnMPR) FROM new_table), 4)) / (COUNT(RtnMPR) *
POWER(STDDEV(RtnMPR),

4))), 2) AS kurtosis_RtnMPR

from new_table) kurtosis_RtnMPR1;
```

kurtosis_quantity	kurtosis_Returnquantity	kurtosis_Final_cost	kurtosis_Final_Sales	kurtosis_RtnMPR
14.7	23.43	41.6	24.76	35.42

Comparison table showing the business decisions results for the unclean and clean data:

Parameter	Uncleaned Data	Cleaned_Data
Mean	**Quantity -2.25**	**Quantity-1.78**
	ReturnQuantity -0.30	**ReturnQuantity -0.19**
	Final_cost-124.43	**Final_cost-88.47**
	Final_Sales-232.79	**Final_Sales-165.01**
	RtnMPR-29.95	**RtnMPR-13.85**
Median	**Quantity-3.00**	**Quantity-1.00**
	ReturnQuantity -0.00	**ReturnQuantity -0.00**
	Final_cost-141	**Final_cost-52.99**
	Final_Sales-53.57	**Final_Sales-56.44**
	RtnMPR-0-00	**RtnMPR-0.00**
Mode	**Quantity-1**	**Quantity-1**
	ReturnQuantity -0	**ReturnQuantity -0**
	Final_cost-49.352	**Final_cost-49.352**
	Final_Sales-60.8	**Final_Sales-60.8**
	RtnMPR-0	**RtnMPR-0**
Variance	**Quantity-27.14**	**Quantity-3.32**

	ReturnQuantity 2.17	**ReturnQuantity -0.41**
	Final_cost-221685.41	**Final_cost-17736.69**
	Final_Sales-458223.23	**Final_Sales-71134.19**
	RtnMPR-34178.39	**RtnMPR-2596.22**
StandardDeviation	**Quantity-5.21**	**Quantity-1.82**
	ReturnQuantity -1.67	**ReturnQuantity -0.64**
	Final_cost-470.83	**Final_cost-133.18**
	Final_Sales-676.93	**Final_Sales-266.71**
	RtnMPR-184.87	**RtnMPR-49.96**
Range	**Quantity-150**	**Quantity-17**
	ReturnQuantity -50	**ReturnQuantity -5**
	Final_cost-33138	**Final_cost-1363.11**
	Final_Sales-39490	**Final_Sales-2143.64**
	RtnMPR-8014	**RtnMPR-568.18**
Skewness	**Quantity-11.27**	**Quantity-2.91**
	ReturnQuantity -16.88	**ReturnQuantity -4.23**
	Final_cost-34.22	**Final_cost-5.86**
	Final_Sales-21.09	**Final_Sales-4.3**
	RtnMPR-15.58	**RtnMPR5.11**
Kurtosis	**Quantity-180.5**	**Quantity-14.7**
	ReturnQuantity -398.03	**ReturnQuantity -23.43**
	Final_cost-1984.44	**Final_cost-41.6**
	Final_Sales-947.71	**Final_Sales-24.76**
	RtnMPR-395.28	**RtnMPR-35.42**

Observation: Overall, the results indicate that the unclean data exhibits higher mean, variance, standard deviation, range, skewness, and kurtosis values compared to the clean data. This suggests greater inconsistencies, variability, and potential outliers in the unclean data. Cleaning the data has resulted in more stable and normalized distributions with reduced variability and potential biases, making it more reliable for business decision-making.