# Retrieving data using the SQL SELECT statement:

1. Using column aliases:

SELECT

"A1"."NAME" "HOSPITALNAME",

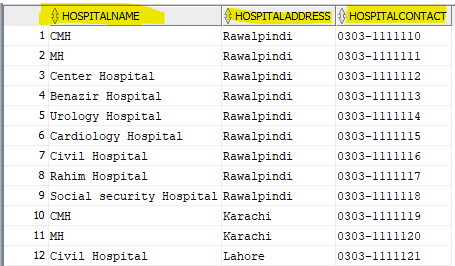
"A1"."ADDRESS" "HOSPITALADDRESS",

"A1"."PHONENO" "HOSPITALCONTACT"

FROM

"GROUPPROJECT"."HOSPITAL" "A1";

**Output:**

****

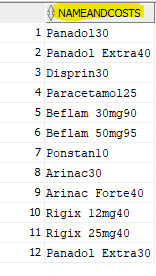
1. Using concatenation operator:

SELECT

"A2".NAME||"A2".COST\_PER\_TAB AS NameAndCosts

FROM "GROUPPROJECT"."MEDICINE" "A2";

Output:



1. Using literal character string:

SELECT

'The phone number of patient '

|| "A1"."NAME"

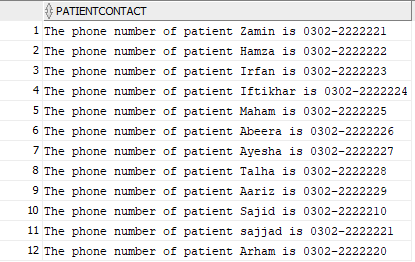
|| ' is '

|| "A1"."PHONENO" "PATIENTCONTACT"

FROM

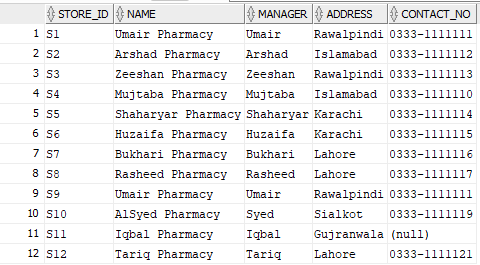
"GROUPPROJECT"."PATIENT" "A1";

**Output:**

****

1. Use of distinct keyword:

Actual Data in store table:



As in this table “Umair Pharmacy” is repeated two times as in S1 and S9, if we only select “name” and “manager” So the distinct keyword remove that duplicate from the table and output the result.

So the query is:

SELECT DISTINCT

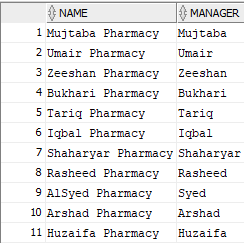
"A1"."NAME" "NAME",

"A1"."MANAGER" "MANAGER"

FROM

"GROUPPROJECT"."STORES" "A1";

**Output:**

****

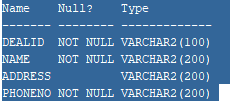
1. Use of DESCRIBE keyword:

To display the structure of dealer table we use the describe keyword.

Query:

DESCRIBE dealer;

**Output:**

****

# Restricting and sorting data:

1. Using the WHERE clause and comparison operator:

Display the medicine whose cost\_per\_tab is less than or equal to 30.

SELECT

"A1"."NAME" "NAME"

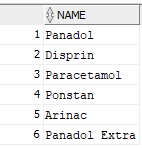
FROM

"GROUPPROJECT"."MEDICINE" "A1"

WHERE

"A1"."COST\_PER\_TAB" <= 30;

**Output:**

****

1. Using the WHERE clause with character String:

* Display the pharmacy name whose manager is “UMAIR”.

Query:

SELECT

"A1"."NAME" "NAME",

"A1"."ADDRESS" "ADDRESS",

"A1"."CONTACT\_NO" "CONTACT\_NO"

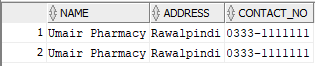
FROM

"GROUPPROJECT"."STORES" "A1"

WHERE

"A1"."MANAGER" = 'Umair';

Output:



1. Range condition using the BETWEEN OPERATOR:

SELECT

"A1"."NAME" "NAME",

"A1"."MFG\_DATE" "MFG\_DATE",

"A1"."EXP\_DATE" "EXP\_DATE",

"A1"."COST\_PER\_TAB" "COST\_PER\_TAB"

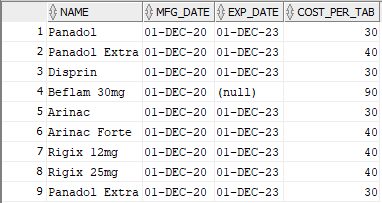
FROM

"GROUPPROJECT"."MEDICINE" "A1"

WHERE

"A1"."COST\_PER\_TAB" BETWEEN 30 AND 90;

**OUTPUT:**

****

1. Combining conditions using the logical operator:

Use of AND operator:

Same query as 3 but with logical AND operator:

Query:

SELECT

"A1"."NAME" "NAME",

"A1"."MFG\_DATE" "MFG\_DATE",

"A1"."EXP\_DATE" "EXP\_DATE",

"A1"."COST\_PER\_TAB" "COST\_PER\_TAB"

FROM

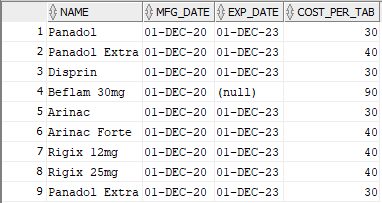
"GROUPPROJECT"."MEDICINE" "A1"

WHERE

"A1"."COST\_PER\_TAB" >= 30

AND "A1"."COST\_PER\_TAB" <= 90;

OUTPUT:



1. Sorting rows using the order by clause:

Sorting “Store” table using the “Contact-no” and then “StoreId”.

Query:

SELECT

"A1"."NAME" "NAME",

"A1"."MANAGER" "MANAGER",

"A1"."ADDRESS" "ADDRESS",

"A1"."CONTACT\_NO" "CONTACT\_NO"

FROM

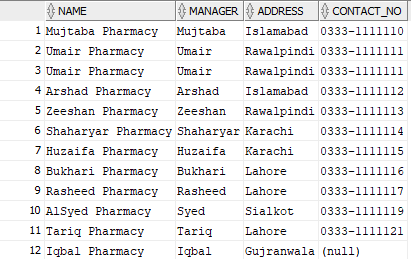
"GROUPPROJECT"."STORES" "A1"

ORDER BY

"A1"."CONTACT\_NO",

"A1"."STORE\_ID";

**OUTPUT:**

****

# Using Single row Function to customize Output:

1. Use of date Functions:

Display the number of weeks we can use the tablet:

Query:

SELECT

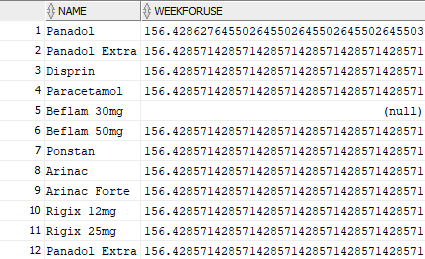
"A1"."NAME" "NAME",

( "A1"."EXP\_DATE" - "A1"."MFG\_DATE" ) / 7 "WEEKFORUSE"

FROM

"GROUPPROJECT"."MEDICINE" "A1";

**OUTPUT:**



# Reported aggregated data using the group function:

1. Using aggregate to find min, max, sum, avg and counts of per tab.

Query:

SELECT

COUNT(\*) "COUNTIDS",

AVG("A1"."COST\_PER\_TAB") "AVERAGECOST",

MIN("A1"."COST\_PER\_TAB") "MINIMUMCOST",

MAX("A1"."COST\_PER\_TAB") "MAXIMUMCOST",

SUM("A1"."COST\_PER\_TAB") "SUMOFCOSTS"

FROM

"GROUPPROJECT"."MEDICINE" "A1"

WHERE

"A1"."MED\_ID" >= 'M4'

AND "A1"."MED\_ID" <= 'M9';

Output:



1. Creating group of data:

Select the name and composition of the “Medicine” table and sum the “cost per tab” of the medicine table whose formula is same and then use the order by clause.

Query:

SELECT

"A1"."COMPOSITION" "COMPOSITION",

SUM("A1"."COST\_PER\_TAB") "SUM(COST\_PER\_TAB)"

FROM

"GROUPPROJECT"."MEDICINE" "A1"

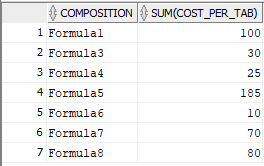
GROUP BY

"A1"."COMPOSITION"

ORDER BY

"A1"."COMPOSITION";

OUTPUT:



# Displaying data from multiple tables using joins:

1. Use of join:

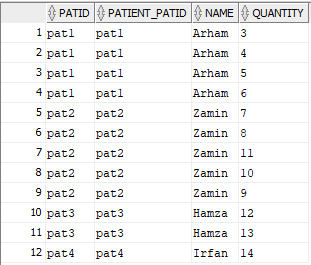
SELECT patient.patid,transactions.patient\_patid,patient.name,transactions.quantity

FROM transactions JOIN patient

on transactions.patient\_patid = patient.patid

ORDER BY patient.patid;

**Output:**

****

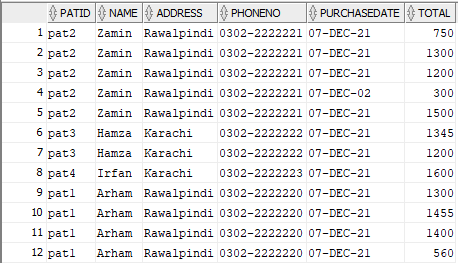
1. Use of natural join:

Query:

SELECT patid,patient.name,patient.address,patient.phoneno,transactions.purchasedate,transactions.total

FROM transactions NATURAL JOIN patient;

**Output:**

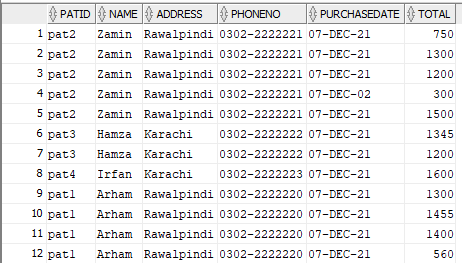


1. Creating joins with the using clause:

SELECT patid,patient.name,patient.address,patient.phoneno,transactions.purchasedate,transactions.total

FROM transactions JOIN patient USING (patid);

**Output:**

****

1. Creating join with ON clause:

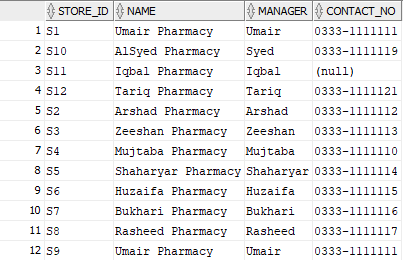
SELECT st.store\_id,st.name,st.manager,st.contact\_no

FROM stores st JOIN supplies su

ON st.store\_id = su.store\_id

ORDER BY st.store\_id;

**OUTPUT:**



# Using subqueries to solve queries:

1. Use of single row query

Query:

SELECT quantity,total

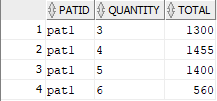
FROM transactions

WHERE transactions.patid = ( SELECT patient.patid

FROM patient

WHERE patient.patid = 'pat1' );

**Output:**



1. Use of subquery with AND operator

Query:

SELECT patient.patid,name,address,phoneno

FROM patient

WHERE patient.patid = ( SELECT patid

FROM transactions

WHERE transactions.stores\_store\_id = 'S2' AND transactions.patid = 'pat1' );

Output:



1. Use of IN operator with subquery:

Query:

SELECT patient.patid,name,address,phoneno

FROM patient

WHERE patient.patid IN

( SELECT transactions.patid

FROM transactions

WHERE transactions.patid = 'pat1' );

**OUTPUT:**

****

# Using the Set Operator:

1. Union Operator:

Query:

( SELECT

"A3"."PATID" "PATID"

FROM

"GROUPPROJECT"."PATIENT" "A3"

)

UNION

( SELECT

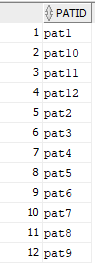
"A2"."PATID" "PATID"

FROM

"GROUPPROJECT"."TRANSACTIONS" "A2"

);

**Output:**

****

1. Using the minus operator with subquery:

Query:

SELECT

"A1"."PATID" "PATID",

"A1"."NAME" "NAME",

"A1"."ADDRESS" "ADDRESS"

FROM

"GROUPPROJECT"."PATIENT" "A1"

WHERE

"A1"."PATID" = ANY (

( SELECT

"A4"."PATID" "PATID"

FROM

"GROUPPROJECT"."PATIENT" "A4"

)

MINUS

( SELECT

"A3"."PATID" "PATID"

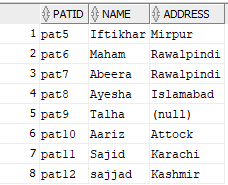
FROM

"GROUPPROJECT"."TRANSACTIONS" "A3"

)

);

**OUTPUT:**

****

# Manipulating data using DML statement:

1. Inserting new rows

Insert into dealer table:

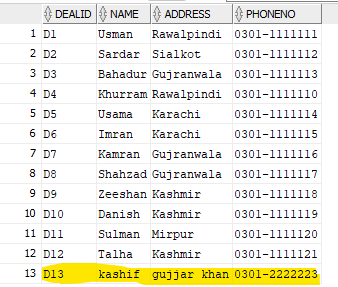
INSERT INTO dealer (dealid,name,address,phoneno)

VALUES ('D13','kashif','gujjar khan','0301-2222223');

**Output:**

****

Table:



1. Update query:

UPDATE dealer

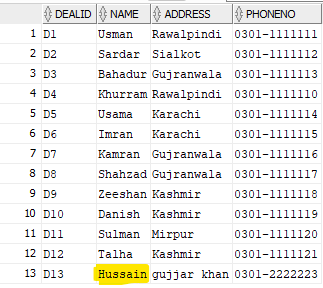
SET name = 'Hussain'

WHERE dealid = 'D13';

**Output:**



Table:



1. Row Delete query

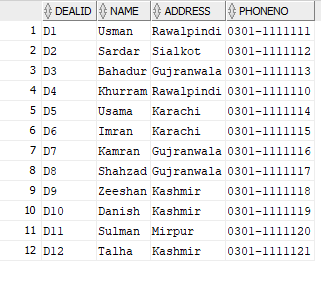
DELETE FROM dealer

WHERE dealid = 'D13';

Output:

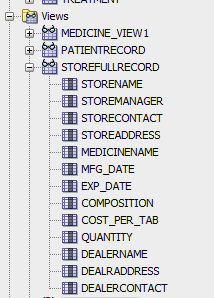


Table now:



# Indexes and views:

Views:



Indexes:

