SE-COMPUTER (B-Div)	Roll number : 8942
Experiment no. : 6	Date of Implementation : 26/03/2021

Aim : To implement Join and complex SQL commands

Tool Used : MySql / PostgreSQL

Related Course outcome : At the end of the course, Students will be able to Use SQL : Standard language of relational database

Rubrics for assessment of Experiment:

Indicator	Poor	Average	Good
Timeliness • Maintains assignment deadline (3)	Assignment not done (0)	One or More than One week late (1-2)	Maintains deadline (3)
Completeness and neatness • Complete all parts of assignment(3)	N/A	< 80% complete (1-2)	100% complete (3)
Originality • Extent of plagiarism(2)	Copied it from someone else(0)	At least few questions have been done without copying(1)	Assignment has been solved completely without copying (2)
Knowledge • In depth knowledge of the assignment(2)	Unable to answer 2 questions(0)	Unable to answer 1 question (1)	Able to answer 2 questions (2)

Assessment Marks :

Timeliness	
Completeness and neatness	
Originality	
Knowledge	
Total	

Total : **(Out of 10)**

Teacher's Sign :	
EXPERIMENT 5	Complex SQL commands
Aim	To implement complex SQL queries
Tools	MySql / PostgreSQL

Theory

Joining Tables

The FROM clause allows more than 1 table in its list, however simply listing more than one table will *very* rarely produce the expected results. The rows from one table must be correlated with the rows of the others. This correlation is known as *joining*.

In the subsequent text, the following 3 example tables are used:

p Table (parts)

pno	descr	color
P1	Widget	Blue
P2	Widget	Red
P3	Dongle	Green

s Table (suppliers)

sno	name	city
S1	Pierre	Paris
S2	John	London
S3	Mario	Rome

sp Table (suppliers & parts)

sno	pno	qty
S1	P1	NULL
S2	P1	200
S3	P1	1000
S3	P2	200

An example can best illustrate the rationale behind joins. The following query:

SELECT * FROM sp, p

Produces:

sno	pno	qty	pno	descr	color
S1	P1	NULL	P1	Widget	Blue
S1	P1	NULL	P2	Widget	Red
S1	P1	NULL	P3	Dongle	Green
S2	P1	200	P1	Widget	Blue
S2	P1	200	P2	Widget	Red
S2	P1	200	P3	Dongle	Green
S3	P1	1000	P1	Widget	Blue
S3	P1	1000	P2	Widget	Red
S3	P1	1000	P3	Dongle	Green
S3	P2	200	P1	Widget	Blue
S3	P2	200	P2	Widget	Red
S3	P2	200	P3	Dongle	Green

Each row in *sp* is arbitrarily combined with each row in *p*, giving 12 result rows (4 rows in *sp* X 3 rows in *p*.) This is known as a *cartesian product*.

A more usable query would correlate the rows from *sp* with rows from *p*, for instance matching on the common column -- *pno*:

SELECT *
FROM sp, p
WHERE sp.pno = p.pno

This produces:

sno	pno	qty	pno	descr	color
S1	P1	NULL	P1	Widget	Blue
S2	P1	200	P1	Widget	Blue
S3	P1	1000	P1	Widget	Blue
S3	P2	200	P2	Widget	Red

More information refer this https://www.tutorialspoint.com/sql/sql-using-joins.htm

Procedure

1. Create following table:

We have already created three tables:

client_master, product_master and sales_order in previous experiments.

Table : Client_master

Column Name	Data type	Size	Constraint
client_no	varchar	6	PRIMARY KEY
Name	varchar	20	NOT NULL
Address	varchar	30	
City	varchar	15	
Pincode	Numeric	8	
State	Varchar	15	
Bal_due	Numeric	10,2	

Table : product_master

Column Name	Data type	Size	Constraint
product_no	Varchar	6	PRIMARY KEY
Description	Varchar	15	NOT NULL
Profit_percent	Numeric	4,2	
Unit_measure	Varchar	10	
Qty_on_hand	Numeric	8	
Reorder_level	Numeric	8	
Sell_price	Numeric	8,2	
Cost_price	Numeric	8,2	

Table :Sales_order

Column Name	Data type	Size	Constraint
order_no	varchar	6	Primary Key
Order_date	Date		NOT NULL
Client_no	varchar	6	NOT NULL
Dely_addr	varchar	25	
Salesman_no	varchar	6	
Dely_type	char	1	
Billed_yn	char	1	
Dely_date	Date		
Order_status	varchar	10	

1. Now, Create one more table **sales_order_details:**

Column Name	Data type	Size	Attributes
order_no	varchar	6	Primary key / Foreign key references order_no of sales_order
Product_no	varchar	6	Primary key / foreign key references product_no of product_master
Qty_ordered	numeric	8	
Qty_disp	numeric	8	
Product_rate	numeric	10,2	

2. Find the products which has been sold to client 'Ivan Bayross'
3. Find out product and their quantities that is to be delivered.
4. Find out the product number and description of Moving products.
5. Find out the names of clients who have purchased 'CD Drive'.
6. List the product_no and order_no of customers having quantity ordered less than 5 form sales_order_details table for the product 'floppies'.
7. Find the products and their quantities for the orders placed by client_no 'C0001' and 'C0002'.
8. Find the description and total quantity sold for each products.
9. Find the value of each product sold.
10. Find out the name of customers who have given the order of more than 10 qty.

Post Lab Questions:

1. What is the difference between inner Join and outer Join.
2. Give one example for equi_join and non equi_join.
3. complete online exercise and add screen shots https://www.w3schools.com/sql/exercise.asp?filename=exercise_join1

****

1. Creating a database as exp_6

Query:

create database exp_6;

Create following table:

We have already created three tables:

client_master, product_master and sales_order in previous experiments.

A.] Client_master

1.] Creating table

Query:

```
create table client_master(
Client_no varchar(6), PRIMARY KEY(Client_no),
Name varchar(20) not null,
Address varchar(30),
City varchar(15),
Pincode numeric(8),
State varchar(15),
Bal_due numeric(10,2));
```

Screenshot:

2.] Inserting values

Query:

Insert into client_master value('C0001', 'Praisy', 'Mira Road', 'Mumbai', 401107, 'Maharashtra', 400);

Insert into client_master value('C0002', 'Ashley', 'Ashok Nagar', 'Chennai', 401107, 'Tamil Nadu', 400);

Insert into client_master value('C0003', 'Prejith', 'Adyar', 'Chennai', 401107, 'Tamil Nadu', 400);

Insert into client_master value('C0004', 'Elwin', 'Bandra', 'Mumbai', 401107, 'Maharashtra', 400);
Insert into client_master value('C0005', 'Elaine', 'Pallom', 'Kottayam', 401107, 'Kerala', 400);
Insert into client_master value('C0006', 'Jerry', 'Borivali', 'Mumbai', 401107, 'Maharashtra', 400);
Insert into client_master value('C0007', 'Aarya', 'Adyar', 'Chennai', 401107, 'Tamil Nadu', 400);
Insert into client_master value('C0008', 'Aamina', 'Bandra', 'Mumbai', 401107, 'Maharashtra', 400);
Insert into client_master value('C0009', 'Ram', 'Adyar', 'Madras', 401107, 'Tamil Nadu', 400);
Insert into client_master value('C0010', 'Vandana', 'Bandra', 'Mumbai', 401107, 'Maharashtra', 400);

Screenshot:

Result Grid | Filter Rows: | Edit: | Export/Import:

Client_no	Name	Address	City	Pincode	State	Bal_due
C0001	Praisy	Mira Road	Mumbai	401107	Maharashtra	400.00
C0002	Ashley	Ashok Nagar	Chennai	401107	Tamil Nadu	400.00
C0003	Prejith	Adyar	Chennai	401107	Tamil Nadu	400.00
C0004	Elwin	Bandra	Mumbai	401107	Maharashtra	400.00
C0005	Elaine	Pallom	Kottayam	401107	Kerala	400.00
C0006	Jerry	Borivali	Mumbai	401107	Maharashtra	400.00
C0007	Aarya	Adyar	Chennai	401107	Tamil Nadu	400.00
C0008	Aamina	Bandra	Mumbai	401107	Maharashtra	400.00
C0009	Ram	Adyar	Madras	401107	Tamil Nadu	400.00
C0010	Ivan Bayross	Bandra	Mumbai	401107	Maharashtra	400.00

B.]Product_master

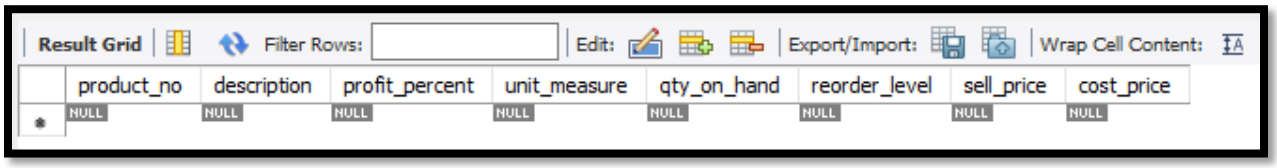
1.] Creating Table

Query:

create table Product_master(
product_no varchar(6), PRIMARY KEY(product_no),
description varchar(15) not null,
profit_percent numeric(4,2),
unit_measure varchar(10),

qty_on_hand numeric(8),
reorder_level numeric(8),
sell_price numeric(8,2),
cost_price numeric(8,2));

Screenshot:



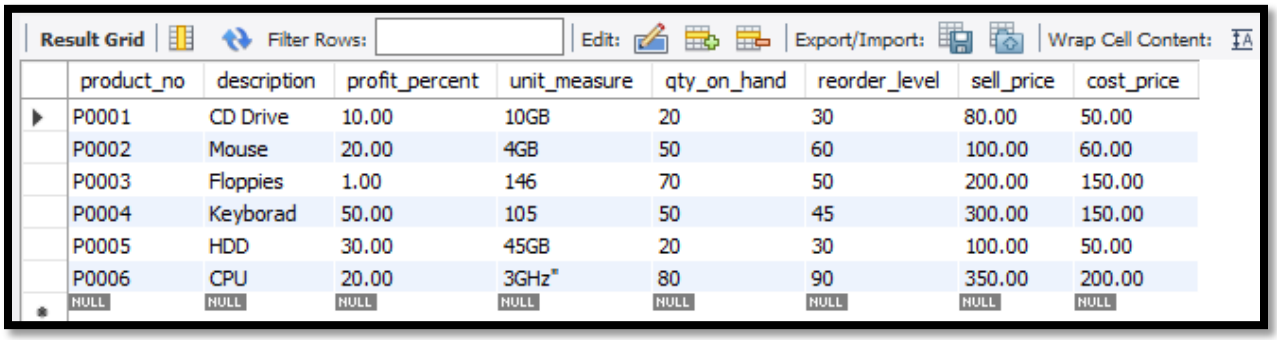
product_no	description	profit_percent	unit_measure	qty_on_hand	reorder_level	sell_price	cost_price
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

2.] Inserting values

Query:

Insert into product_master value('P0001', 'CD Drive', '10', '10GB',20, 30, 80, 50);
Insert into product_master value('P0002', 'Mouse', '20', '4GB',50, 60,100, 60);
Insert into product_master value('P0003', 'Floppies', '1', '146',70, 50, 200, 150);
Insert into product_master value('P0004', 'Keyborad', '50', '105 ',50, 45, 300, 150);
Insert into product_master value('P0005', 'HDD', '30', '45GB',20, 30, 100, 50);
Insert into product_master value('P0006', 'CPU', '20', '3GHz" ',80, 90, 350, 200);

Screenshot:



product_no	description	profit_percent	unit_measure	qty_on_hand	reorder_level	sell_price	cost_price
P0001	CD Drive	10.00	10GB	20	30	80.00	50.00
P0002	Mouse	20.00	4GB	50	60	100.00	60.00
P0003	Floppies	1.00	146	70	50	200.00	150.00
P0004	Keyborad	50.00	105	50	45	300.00	150.00
P0005	HDD	30.00	45GB	20	30	100.00	50.00
P0006	CPU	20.00	3GHz"	80	90	350.00	200.00
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

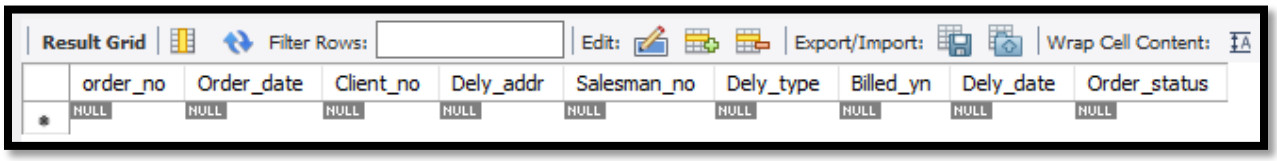
C.] Sales_order

1.] Creating table

Query:

```
create table sales_order(
order_no varchar(6), PRIMARY KEY(order_no),
Order_date date NOT NULL,
Client_no varchar(6) NOT NULL,
Dely_addr varchar(25),
Salesman_no varchar(6),
Dely_type char(1),
Billed_yn char(1),
Dely_date date,
Order_status varchar(10));
```

Screenshot:



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

order_no	Order_date	Client_no	Dely_addr	Salesman_no	Dely_type	Billed_yn	Dely_date	Order_status
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

2.] Inserting values

Query:

```
Insert into sales_order value('101','2021-02-26', 'C0001', 'Mumbai', '1A', 's', 'y',
'2021-01-17', 'delivered');
Insert into sales_order value('102','2021-02-21', 'C0002', 'Chennai', '2B', 's', 'y',
'2021-01-15', 'shipped');
Insert into sales_order value('103','2021-02-02', 'C0003', 'Chennai', '3C', 'r', 'n',
'2021-01-09', 'delivered');
Insert into sales_order value('104','2021-02-06', 'C0004', 'Mumbai', '4D', 's', 'y',
'2021-01-13', 'delivered');
Insert into sales_order value('105','2021-02-15', 'C0005', 'Kottayam', '5E', 'r', 'n',
'2021-01-19', 'shipped');
Insert into sales_order value('106','2021-02-10', 'C0006', 'Mumbai', '6F', 's', 'y',
'2021-01-18', 'delivered');
```

Insert into sales_order value('107','2021-01-10', 'C0007', 'Chennai', '6F', 's', 'y', '2021-01-18', 'delivered');
Insert into sales_order value('108','2021-01-10', 'C0008', 'Mumbai', '6F', 's', 'y', '2021-01-18', 'shipped');
Insert into sales_order value('109','2021-01-10', 'C0009', 'Madras', '6F', 's', 'y', '2021-01-18', 'shipped');
Insert into sales_order value('110','2021-01-10', 'C00010', 'Mumbai', '6F', 's', 'y', '2021-01-18', 'delivered');

Screenshot:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

order_no	Order_date	Client_no	Dely_addr	Salesman_no	Dely_type	Billed_yn	Dely_date	Order_status
101	2021-02-26	C0001	Mumbai	1A	s	y	2021-01-17	delivered
102	2021-02-21	C0002	Chennai	2B	s	y	2021-01-15	shipped
103	2021-02-02	C0003	Chennai	3C	r	n	2021-01-09	delivered
104	2021-02-06	C0004	Mumbai	4D	s	y	2021-01-13	delivered
105	2021-02-15	C0005	Kottayam	5E	r	n	2021-01-19	shipped
106	2021-02-10	C0006	Mumbai	6F	s	y	2021-01-18	delivered
107	2021-01-10	C0007	Chennai	6F	s	y	2021-01-18	delivered
108	2021-01-10	C0008	Mumbai	6F	s	y	2021-01-18	shipped
109	2021-01-10	C0009	Madras	6F	s	y	2021-01-18	shipped
110	2021-01-10	C00010	Mumbai	6F	s	y	2021-01-18	delivered

D.] Sales_order_details

1.] Creating Table

Query:

Create table sales_order_details(
order_no char(6) references sales_order,
product_no char(6) references product_master,
qtyordered numeric(8),
qtydisp numeric(8),
productrate numeric(10,2));

Screenshot:

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

order_no	product_no	qtyordered	qtydisp	productrate

2.] Inserting values

Query:

Insert into sales_order_details value('101', 'P0001', 10, 5, 5000);
Insert into sales_order_details value('102', 'P0002', 23, 20, 2000);
Insert into sales_order_details value('103', 'P0003', 5, 5, 1000);
Insert into sales_order_details value('105', 'P0004', 10, 2, 1500);
Insert into sales_order_details value('106', 'P0005', 18, 15, 5500);
Insert into sales_order_details value('110', 'P0006', 10, 9, 2500);

Screenshot:

Result Grid | Filter Rows: | Export:

order_no	product_no	qtyordered	qtydisp	productrate
101	P0001	10	5	5000.00
102	P0002	23	20	2000.00
103	P0003	4	3	1000.00
103	P0003	2	2	500.00
105	P0004	10	2	1500.00
106	P0005	18	15	5500.00
110	P0006	10	9	2500.00

2. Find the products which has been sold to client 'Ivan Bayross'

Query:

select Name,product_master.product_no,description,sales_order.order_no
from client_master,product_master,sales_order,sales_order_details
where client_master.Name = 'Ivan Bayross'
and client_master.client_no = sales_order.client_no
and sales_order.order_no = sales_order_details.order_no
and sales_order_details.product_no = product_master.product_no;

Screenshot:

Result Grid | Filter Rows: | Export:

Name	product_no	description	order_no
Ivan Bayross	P0006	CPU	110

3. Find out product and their quantities that is to be delivered.

Query:

```
select dely_date, sales_order_details.product_no,qty_on_hand,description
from sales_order,sales_order_details,product_master
where Order_status ='delivered'
and sales_order.order_no = sales_order_details.order_no
and sales_order_details.product_no = product_master.product_no ;
```

Screenshot:

Result Grid | Filter Rows: | Export:

dely_date	product_no	qty_on_hand	description
2021-01-17	P0001	20	CD Drive
2021-01-09	P0003	70	Floppies
2021-01-18	P0005	20	HDD
2021-01-18	P0006	80	CPU

4. Find out the product number and description of Moving products.

Query:

```
select distinct product_master.product_no, product_master.description
from product_master, sales_order_details
where product_master.product_no = sales_order_details.product_no;
```

Screenshot:

Result Grid | Filter Rows: | Export:

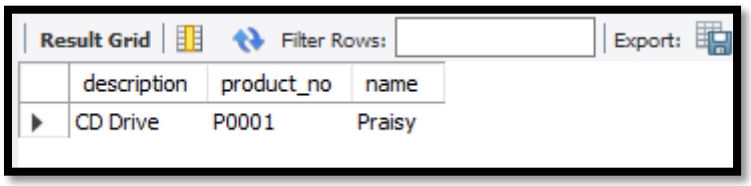
product_no	description
P0001	CD Drive
P0002	Mouse
P0003	Floppies
P0004	Keyborad
P0005	HDD
P0006	CPU

5. Find out the names of clients who have purchased ‘CD Drive’.

Query:

```
select description,product_master.product_no,name
from product_master,client_master,sales_order,sales_order_details
where description = 'CD Drive'
and product_master.product_no = sales_order_details.product_no
and sales_order_details.order_no = sales_order.order_no
and sales_order.client_no = client_master.client_no;
```

Screenshot:



Result Grid | Filter Rows: | Export:

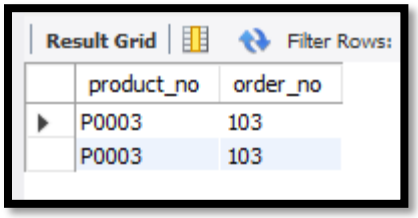
description	product_no	name
CD Drive	P0001	Praisy

6. List the product_no and order_no of customers having quantity ordered less than 5 form sales_order_details table for the product ‘floppies’.

Query:

```
select product_master.product_no,order_no
from product_master,sales_order_details
where description = 'Floppies'
and product_master.product_no = sales_order_details.product_no
and qtyordered < 5;
```

Screenshot:



Result Grid | Filter Rows:

product_no	order_no
P0003	103
P0003	103

7. **Find the products and their quantities for the orders placed by client_no ‘C0001’ and ‘C0002’.**

Query:

```
select description,qtyordered
from product_master, sales_order_details, sales_order
where product_master.product_no = sales_order_details.product_no
and sales_order_details.order_no = sales_order.order_no
and client_no in('C0001','C0002');
```

Screenshot:

Result Grid | Filter Rows:

description	qtyordered
CD Drive	10
Mouse	23

8. **Find the description and total quantity sold for each products.**

Query:

```
select description ,reorder_level, qty_on_hand "SOLD"
from product_master;
```

Screenshot:

Result Grid | Filter Rows:

description	reorder_level	SOLD
CD Drive	30	20
Mouse	60	50
Floppies	50	70
Keyborad	45	50
HDD	30	20
CPU	90	80

9. Find the value of each product sold.

Query:

```
select description,(reorder_level - qty_on_hand) * sell_price "VALUES"
from product_master;
```

Screenshot:

Result Grid | Filter Rows:

description	VALUES
CD Drive	800.00
Mouse	1000.00
Floppies	-4000.00
Keyborad	-1500.00
HDD	1000.00
CPU	3500.00

10. Find out the name of customers who have given the order of more than 10 qty.

Query:

```
SELECT
   client_master.Name
FROM
   ((sales_order_details
   INNER JOIN sales_order ON sales_order.order_no =
sales_order_details.order_no)
   INNER JOIN client_master ON client_master.client_no =
sales_order.client_no)
WHERE
   sales_order_details.qtyordered > 10;
```

Screenshot:

Result Grid

Name
Ashley
Jerry

POSTLAB QUESTIONS:

1. **What is the difference between inner Join and outer Join.**

INNER JOIN	OUTER JOIN
1. It returns the combined tuple between two or more tables.	1. It returns the combined tuple from a specified table even join condition will fail.
2. Used clause INNER JOIN and JOIN.	3. Used clause LEFT OUTER JOIN, RIGHT OUTER JOIN, FULL OUTER JOIN, etc.
3. When any attributes are not common then it will return nothing.	3. It does not depend upon the common attributes. If the attribute is blank then here already placed NULL.
4. If tuples are more. Then INNER JOIN works faster than OUTER JOIN.	4. Generally, The OUTER JOIN is slower than INNER JOIN. But except for some special cases.
5. It is used when we want detailed information about any specific attribute.	5. It is used when we want to complete information.
6. JOIN and INNER JOIN both clauses work the same.	6. FULL OUTER JOIN and FULL JOIN both clauses work the same.
7. SQL Syntax: select * from table1 INNER JOIN / JOIN table2 ON table1.column_name = table2.column_name;	7. SQL Syntax: select * from table1 LEFT OUTER JOIN / RIGHT OUTER JOIN / FULL OUTER JOIN / FULL JOIN table2 ON table1.column_name = table2.column_name;

2. **Give one example for equi_join and non equi_join.**

Ans:

1. EQUI JOIN :

EQUI JOIN creates a JOIN for equality or matching column(s) values of the relative tables. EQUI JOIN also create JOIN by using JOIN with ON and then providing the names of the columns with their relative tables to check equality using equal sign (=).

Syntax :

SELECT column_list

FROM table1, table2....

WHERE table1.column_name = table2.column_name;

Example –

SELECT student.name, student.id, record.class, record.city
FROM student, record
WHERE student.city = record.city;

2. NON EQUI JOIN :

NON EQUI JOIN performs a JOIN using comparison operator other than equal(=) sign like >, <, >=, <= with conditions.

Syntax:

SELECT *

FROM table_name1, table_name2

WHERE table_name1.column [> | < | >= | <=]
table_name2.column;

Example –

SELECT student.name, record.id, record.city
FROM student, record
WHERE Student.id < Record.id ;

3. Complete online exercise and add screen shots

Exercise:

Insert the missing parts in the `JOIN` clause to join the two tables `Orders` and `Customers`, using the `CustomerID` field in both tables as the relationship between the two tables.

```
SELECT *
FROM Orders
LEFT JOIN Customers
ON Orders.CustomerID=Customers.CustomerID;
```

Show Answer

Exercise:

Choose the correct `JOIN` clause to select all records from the two tables where there is a match in both tables.

```
SELECT *
FROM Orders
INNER JOIN Customers
ON Orders.CustomerID=Customers.CustomerID;
```

Show Answer

Exercise:

Choose the correct `JOIN` clause to select all the records from the `Customers` table plus all the matches in the `Orders` table.

```
SELECT *
FROM Orders
RIGHT JOIN Customers
ON Orders.CustomerID=Customers.CustomerID;
```

Show Answer