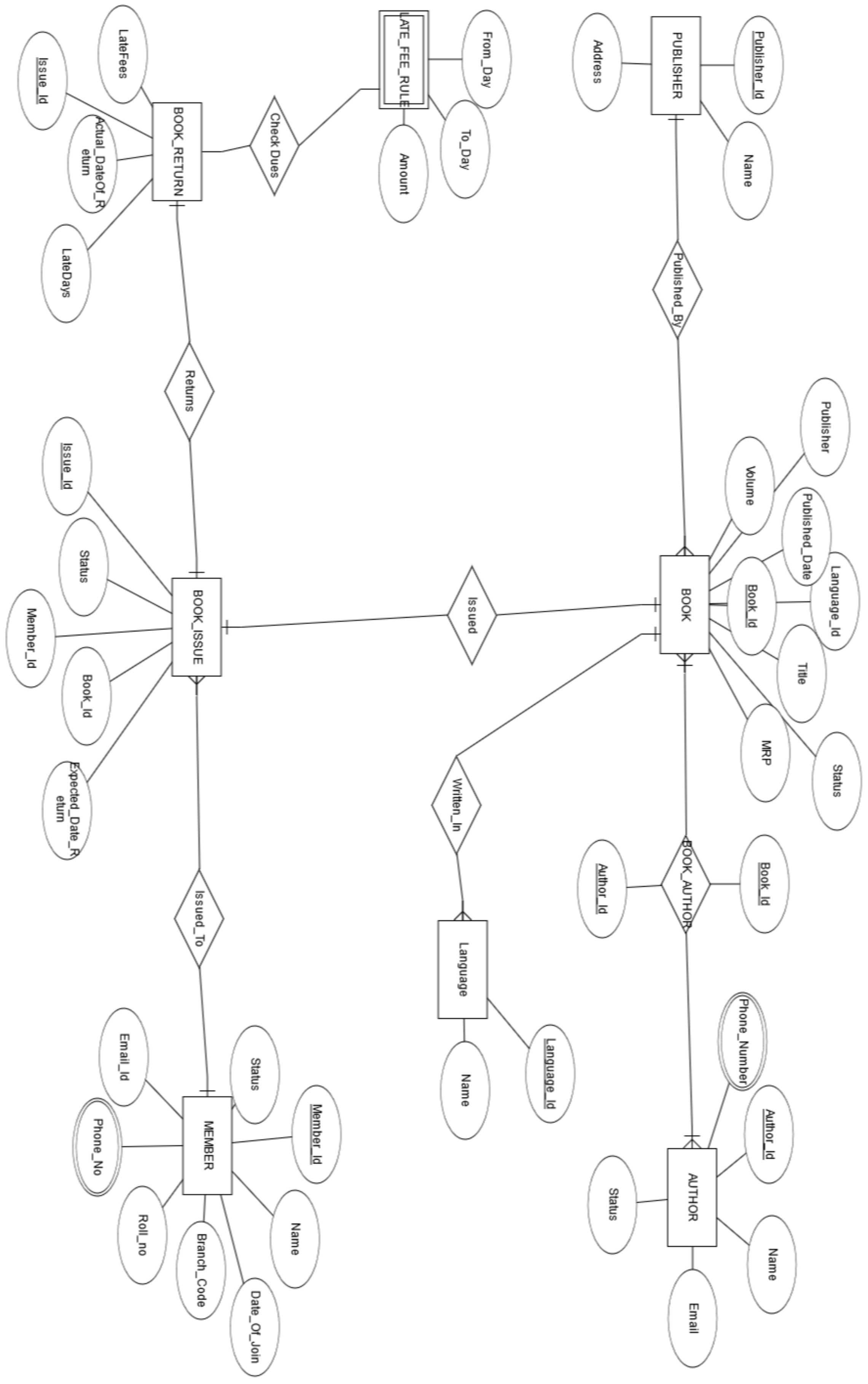
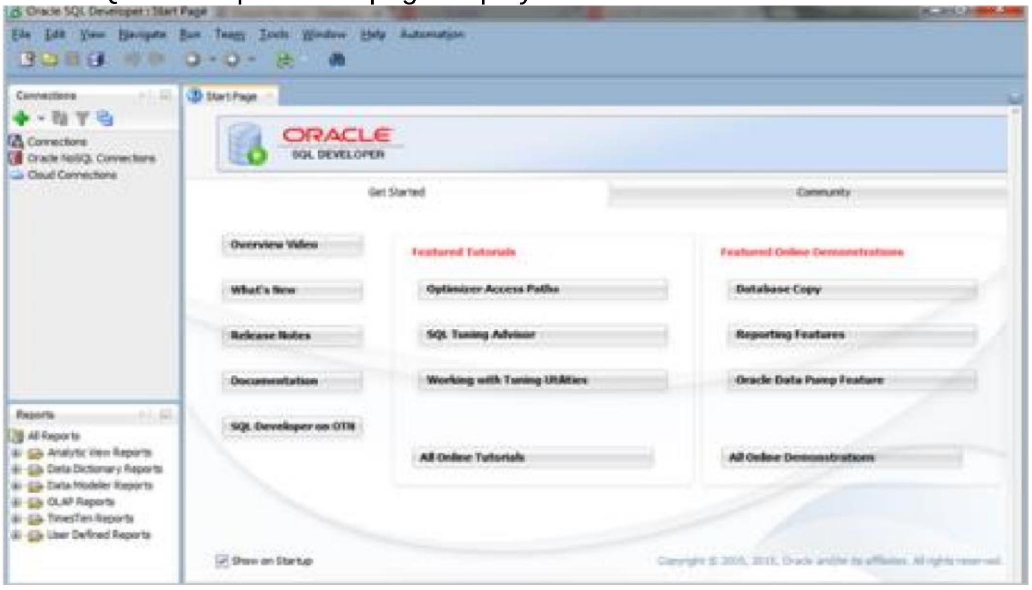

BOOK_RETURN
Issue_Id
LateFees
Actual_DateOf_Return
LateDays
Check Dues
LATE_FEE_RULE
From_Day
To_Day
Amount
PUBLISHER
Publisher_Id
Address
Name
Published_By
Returns
BOOK_ISSUE
Issue_Id
Status
Member_Id
Book_Id
Expected_Date_Return
Issued
BOOK
Publisher
Volume
Published_Date
Book_Id
Language_Id
Title
MRP
Status
Written_In
Language
Name
Language_Id
BOOK_AUTHOR
Author_Id
Book_Id
Issued_To
MEMBER
Email_Id
Status
Member_Id
Phone_No
Roll_no
Branch_Code
Date_Of_Join
Name
AUTHOR
Phone_Number
Author_Id
Status
Name
Email

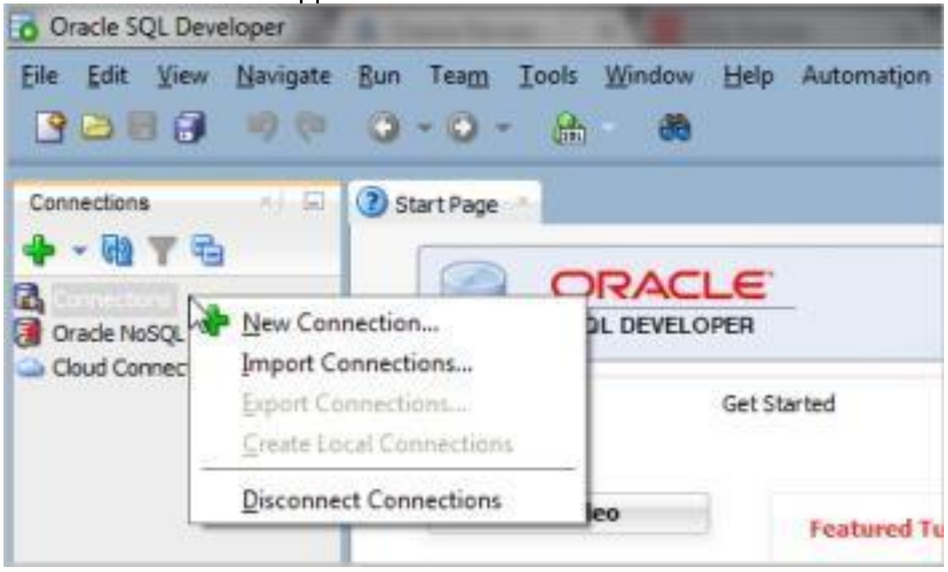
To add an Oracle Cloud connection:

1. Run Oracle SQL Developer locally.

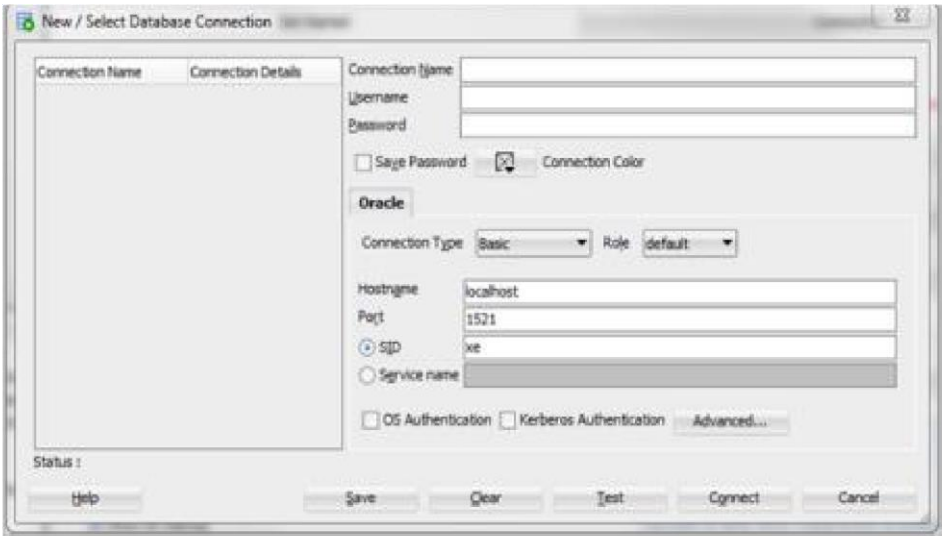
The Oracle SQL Developer home page displays.



2. Under Connections, right click **Connections**.
   The Connection menu appears



3. Select **New Connection**.
   The New/Select Database Connection dialog appears.


New / Select Database Connection

Connection Name	Connection Details

Connection Name:
Username:
Password:
Save Password: ☐
Connection Color

Oracle

Connection Type: Basic
Role: default
Hostname: localhost
Port: 1521
SID: xe (●)
Service name: ○
OS Authentication: ☐
Kerberos Authentication: ☐

Status :

4. On the New/Select Database Connection dialog, make the following entries: ● Connection Name - Enter the name for this cloud connection.

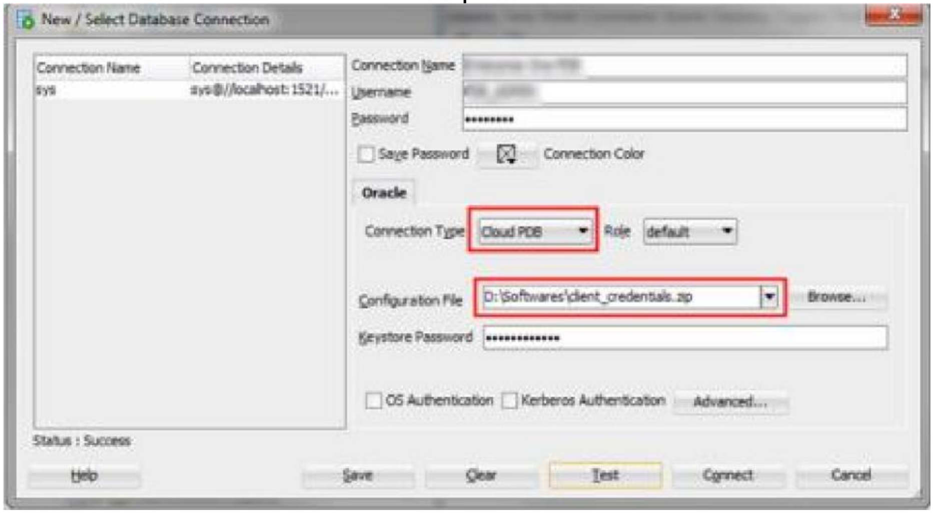
● Username - Enter the **database username** . You can either use the default administrator database account (PDB_ADMIN) provided as part of the service or create a new schema, and use it.

● Password - Enter the **Password** required during sign in when accessing

Exadata Express. ● Connection Type - Select **Cloud PDB**.

● Configuration File - Click **Browse**, and select the **Client Credentials** zip file, downloaded from the Exadata Express service console.

● Keystore Password - Enter the **Password** generated while downloading the Client Credentials from the Exadata Express service console.


New / Select Database Connection

Connection Name	Connection Details
sys	sys@//localhost:1521/...

Connection Name: [redacted]
Username: [redacted]
Password: ••••••••
Save Password: ☐
Connection Color

Oracle

Connection Type: Cloud PDB
Role: default
Configuration File: D:\Softwares\client_credentials.zip
Keystore Password: •••••••••••••
OS Authentication: ☐
Kerberos Authentication: ☐

Status : Success

5. Click **Test**.

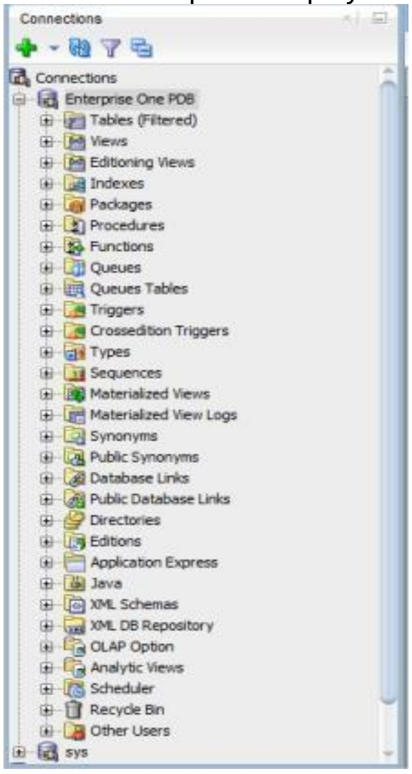
Status: Success displays at the leftmost bottom of the New/Select Database Connection dialog.

6. Click **Connect**.

An entry for the new connection appears under Connections.

7. Open the new connection.

If you have connected successfully, the tables and other objects from Exadata Express display under the new connection.



Deleting Database Connections

You can delete database connections.

1. From the Explore Repository, select **Tools**, and then **Database Connection Manager**.

2. In **Database Connection Manager**, select the database connection to remove, and then click **Delete**.

3. A confirmation message is displayed. Click **Yes** to confirm deletion.

Use the SQL Worksheet in SQL Developer to Insert, Update and Delete Data

Introduction

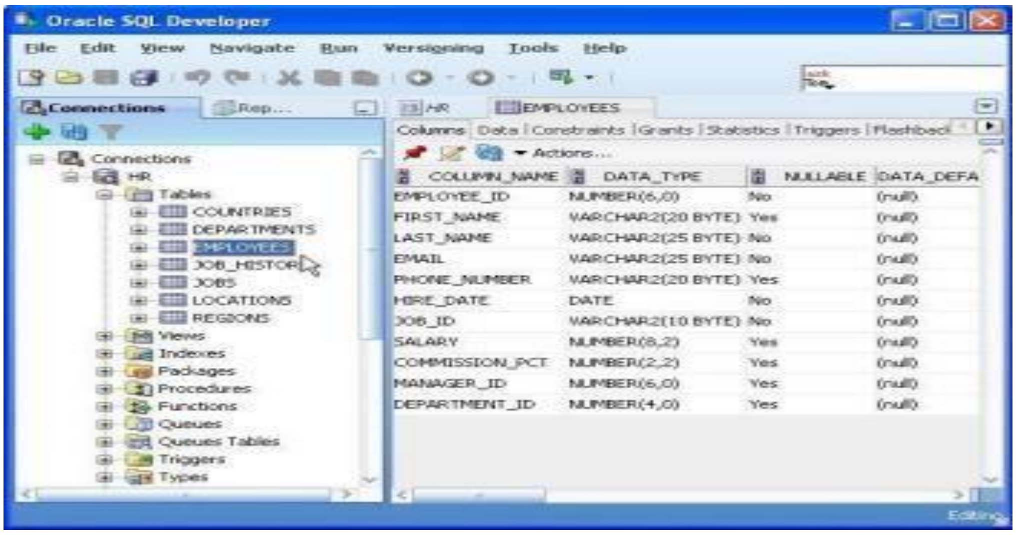
Oracle SQL Developer provides a SQL Worksheet that you can use to update data, by writing simple or complex SQL statements. In this How-To, we look at the most basic of these, inserting a record, updating single and multiple records and deleting single or multiple records.

Adding Data

- Inserting a Row using the Data tab
- Inserting a Row using the SQL Worksheet

SQL Developer has a variety of methods to insert data into your table. We'll start with the most straightforward.

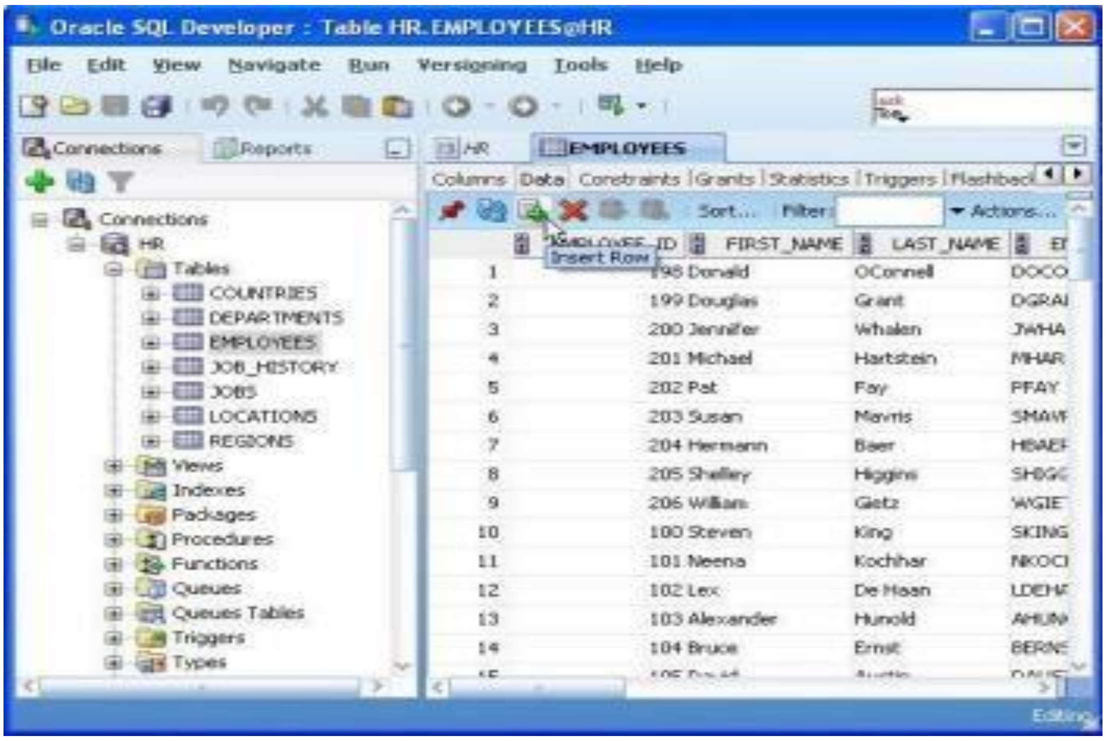
1. SQL Developer makes entering data easily by using the table de nition. Select the EMPLOYEES table in the Connections Navigator.



COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFA
EMPLOYEE_ID	NUMBER(6,0)	No	(null)
FIRST_NAME	VARCHAR2(20 BYTE)	Yes	(null)
LAST_NAME	VARCHAR2(25 BYTE)	No	(null)
EMAIL	VARCHAR2(25 BYTE)	No	(null)
PHONE_NUMBER	VARCHAR2(20 BYTE)	Yes	(null)
HIRE_DATE	DATE	No	(null)
JOB_ID	VARCHAR2(10 BYTE)	No	(null)
SALARY	NUMBER(8,2)	Yes	(null)
COMMISSION_PCT	NUMBER(2,2)	Yes	(null)
MANAGER_ID	NUMBER(6,0)	Yes	(null)
DEPARTMENT_ID	NUMBER(4,0)	Yes	(null)

Notice that some values are required. (Nullable = 'No'). When inserting new rows, at least these values should be populated.

2. Click the Data tab. What you see displayed is the current data you have in that table. Use the scrollbar to view all the rows in your table. To insert a new row click the Insert Row button.

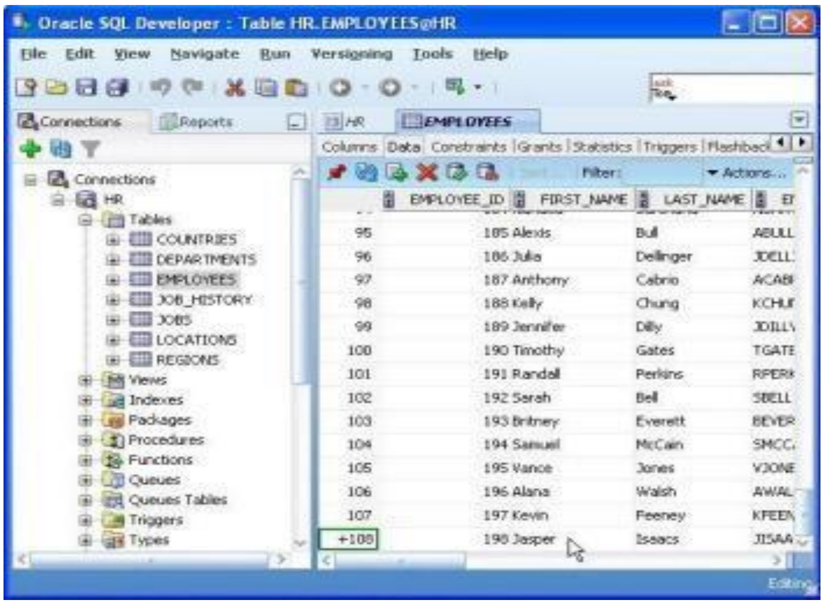


	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EI
1	198	Donald	OConnell	DOCO
2	199	Douglas	Grant	DGRAI
3	200	Jennifer	Whalen	JWHA
4	201	Michael	Hartstein	MHAR
5	202	Pat	Fay	PFAY
6	203	Susan	Mavris	SMAVF
7	204	Hermann	Baer	HBAEF
8	205	Shelley	Higgins	SHIGG
9	206	William	Gietz	WGIET
10	100	Steven	King	SKING
11	101	Neena	Kochhar	NKOCI
12	102	Lex	De Haan	LDEHA
13	103	Alexander	Hunold	AHUN
14	104	Bruce	Ernst	BERNS

Notice the number of rows retrieved is displayed below the Results tab.

3. Fill in values for the required items EMPLOYEE_ID, LAST_NAME, EMAIL,

HIRE_DATE and JOB_ID.

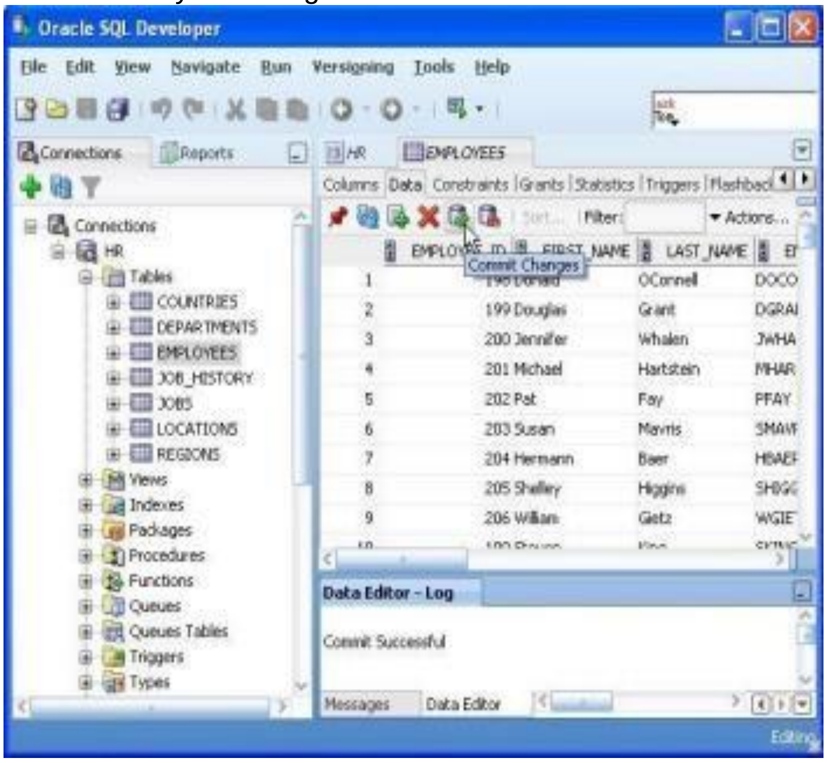


	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EI
95	185	Alexis	Bull	ABULL
96	186	Julia	Dellinger	JDELL
97	187	Anthony	Cabrio	ACABI
98	188	Kelly	Chung	KCHUI
99	189	Jennifer	Dilly	JDILLY
100	190	Timothy	Gates	TGATE
101	191	Randall	Perkins	RPERK
102	192	Sarah	Bell	SBELL
103	193	Britney	Everett	BEVER
104	194	Samuel	McCain	SMCC
105	195	Vance	Jones	VJONE
106	196	Alana	Walsh	AWAL
107	197	Kevin	Feeney	KFEEN
+108	198	Jasper	Isaacs	JISAA

For more complex queries or statements, use the Format function (Ctrl+F7) to make it easier to read the SQL. This can be found in the context menu.

4. To save the record to the database, click the Commit Changes button.

The Data Editor log will show the Commit Successful comment when you have commited your changes.



	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EI
1	198	Donald	OConnell	DOCO
2	199	Douglas	Grant	DGRAI
3	200	Jennifer	Whalen	JWHA
4	201	Michael	Hartstein	MHAR
5	202	Pat	Fay	PFAY
6	203	Susan	Mavris	SMAVI
7	204	Hermann	Baer	HBAEI
8	205	Shelley	Higgins	SHIGG
9	206	William	Gietz	WGIE

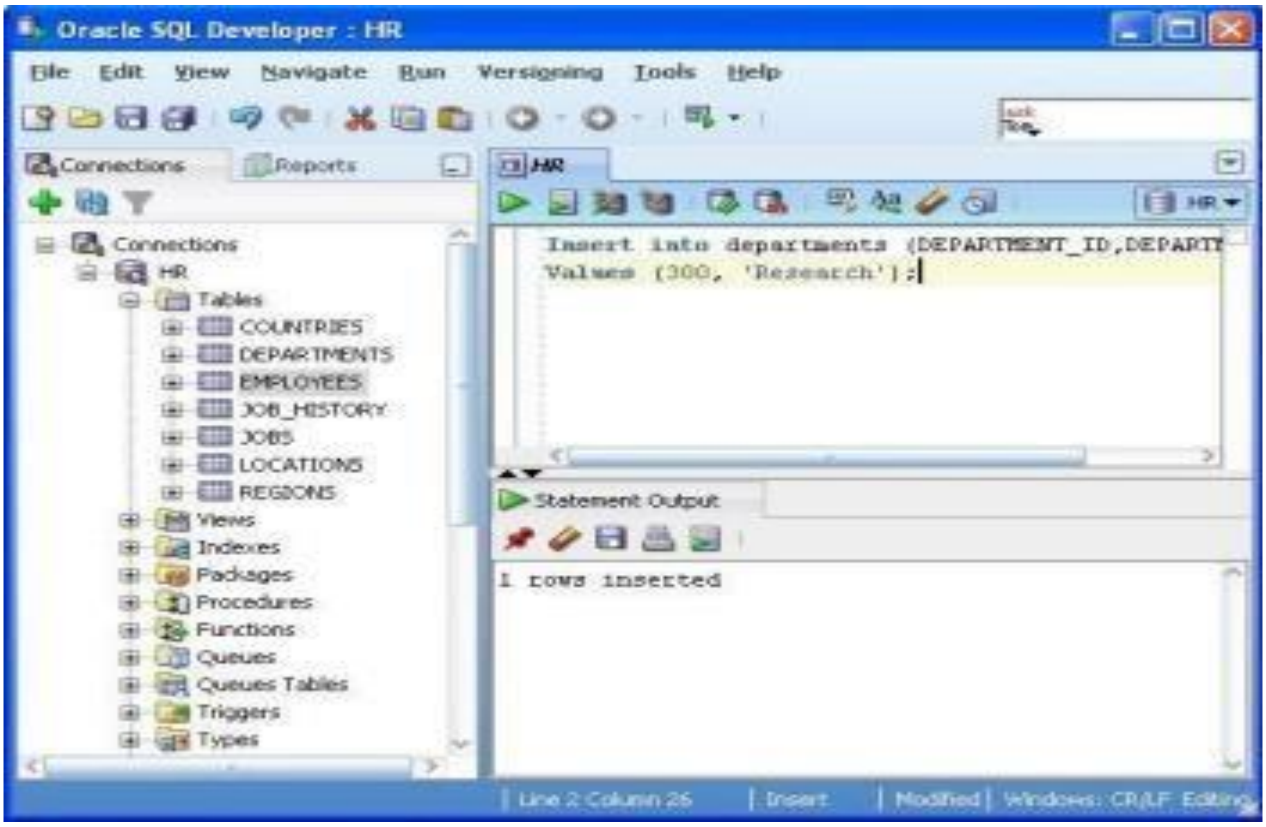
```
Commit Successful
```

5. You can also insert data using the 'traditional' method you'd use when using a command line or SQL Plus. Return to the SQL Worksheet and enter the command:

Insert into departments (DEPARTMENT_ID,DEPARTMENT_NAME) Values (300, 'Research');

Click F9.

NOTE: If you click F5, the detail is shown to the Script Output tab.



```
Insert into departments (DEPARTMENT_ID,DEPART
Values (300, 'Research');
```

```
1 rows inserted
```

Notice the feedback in the message window.
As before you'll need to commit the changes to save them to the database.
Type Commit ; in the SQL Worksheet.

Modifying Data

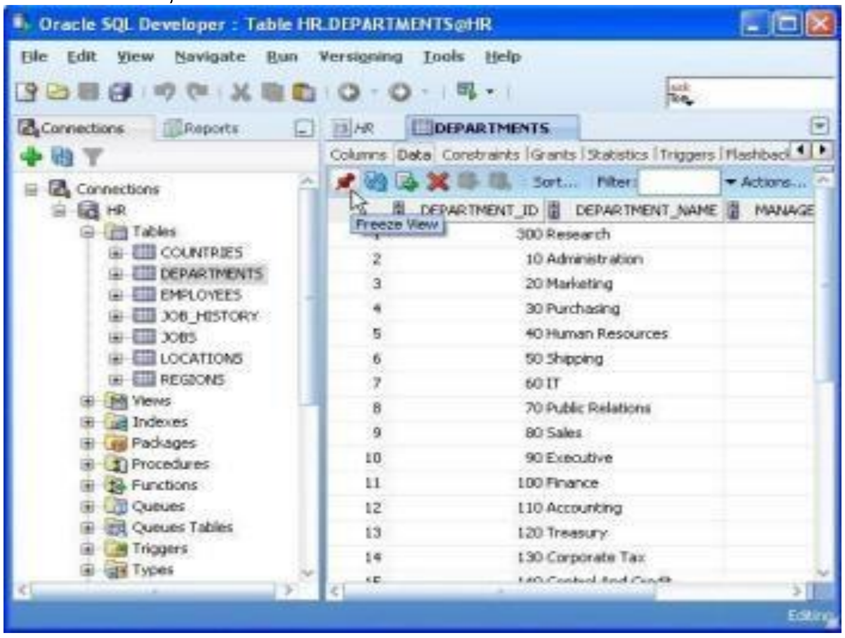
● Updating a Row

● Updating Multiple Rows Using SQL

As in the above example, you can update data using the SQL Worksheet, using SQL commands, or you can use the data tab in the table definition and update individual rows. You'll do both in this next exercise.

1. Once again you can update rows easily by using the Data tab interface.

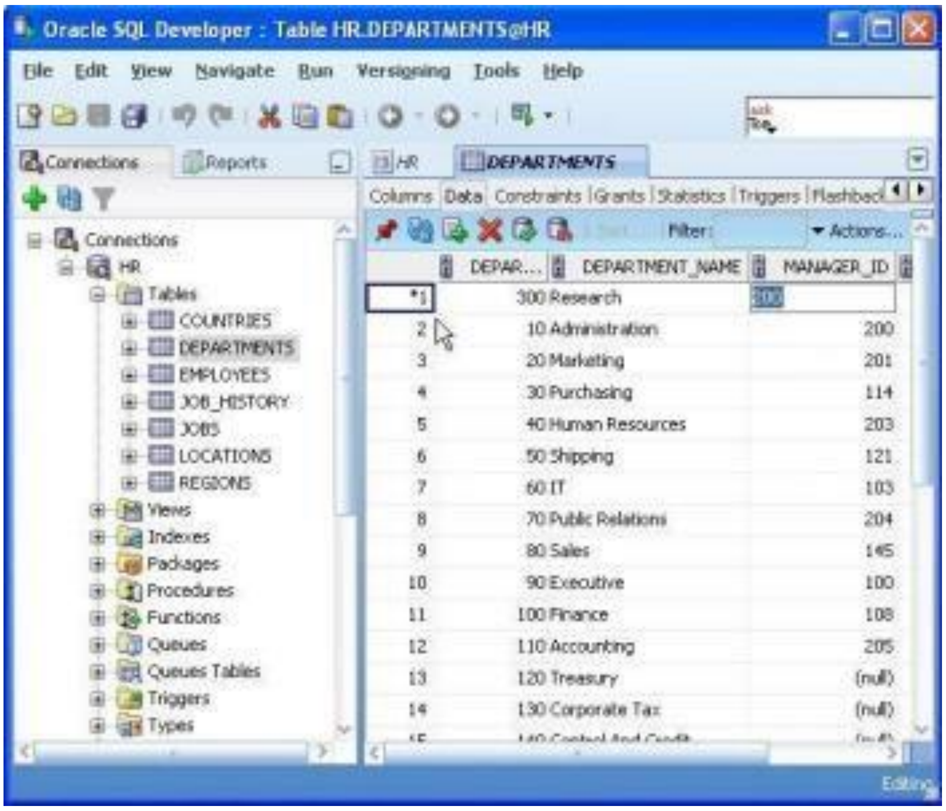
2. As you did in the previous exercise, click on a table in the Connections Navigator. In this exercise, use the DEPARTMENTS table.



	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGE
1	300	Research	
2	10	Administration	
3	20	Marketing	
4	30	Purchasing	
5	40	Human Resources	
6	50	Shipping	
7	60	IT	
8	70	Public Relations	
9	80	Sales	
10	90	Executive	
11	100	Finance	
12	110	Accounting	
13	120	Treasury	
14	130	Corporate Tax	

Notice by clicking on a table di erent from the previous one worked on, the tab is replaced with the new selected table. To keep the EMPLOYEES tab and the DEPARTMENTS tab open, click the Freeze View pin before selecting the new object.
If you always want new tabs to open, you can set a preference to pin tabs.

3. In the last exercise you added a new record. Update that record by clicking on any of the values and changing it.

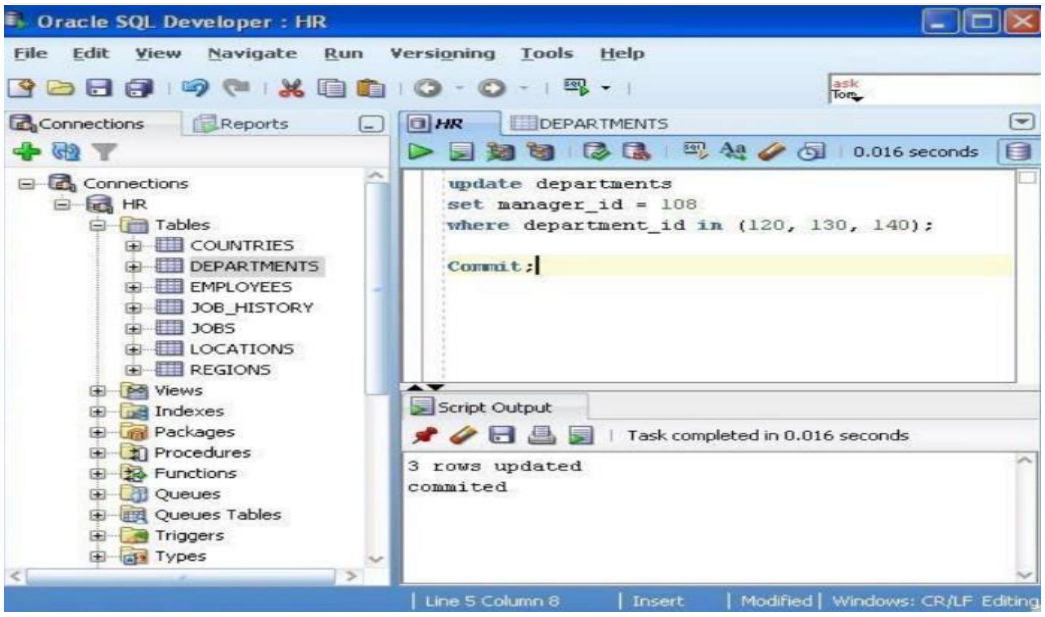


	DEPAR...	DEPARTMENT_NAME	MANAGER_ID
*1	300	Research	300
2	10	Administration	200
3	20	Marketing	201
4	30	Purchasing	114
5	40	Human Resources	203
6	50	Shipping	121
7	60	IT	103
8	70	Public Relations	204
9	80	Sales	145
10	90	Executive	100
11	100	Finance	108
12	110	Accounting	205
13	120	Treasury	(null)
14	130	Corporate Tax	(null)

Notice that once you have updated the record, an asterisk (*) shows next to the record. As before, click the Commit Changes to update the record in the database.

4. You can use this method to update multiple records, but you still need to step through each record and click on the eld to update the record. This can be cumbersome if you have multiple records. To update multiple records, it's easier to use a SQL statement.

```
update departments
set manager_id = 108
where department_id in (120, 130, 140);

Commit;
```



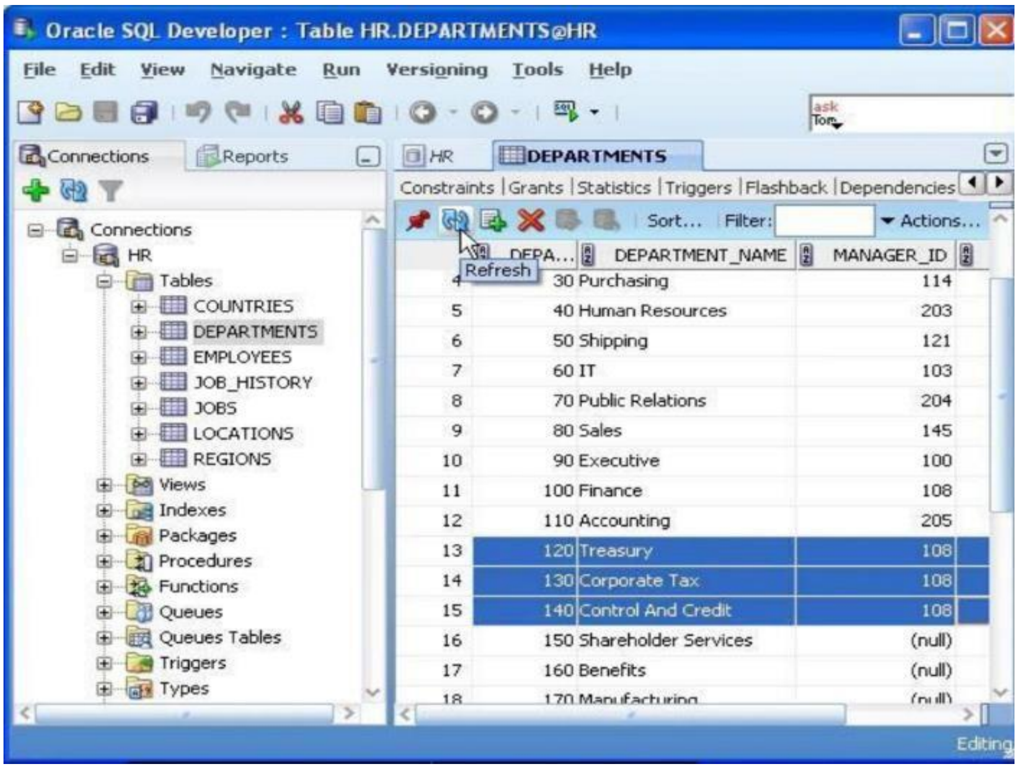
```
update departments
set manager_id = 108
where department_id in (120, 130, 140);

Commit;
```

Task completed in 0.016 seconds

```
3 rows updated
commited
```

The results displayed are the objects your HR schema owns.

5. Review the results of the above by returning to the data tab for the table and select refresh. (or writing a SQL query in the SQL Worksheet)



	DEPA...	DEPARTMENT_NAME	MANAGER_ID
4	30	Purchasing	114
5	40	Human Resources	203
6	50	Shipping	121
7	60	IT	103
8	70	Public Relations	204
9	80	Sales	145
10	90	Executive	100
11	100	Finance	108
12	110	Accounting	205
13	120	Treasury	108
14	130	Corporate Tax	108
15	140	Control And Credit	108
16	150	Shareholder Services	(null)
17	160	Benefits	(null)
18	170	Manufacturing	(null)

The results displayed are the objects your HR schema owns.

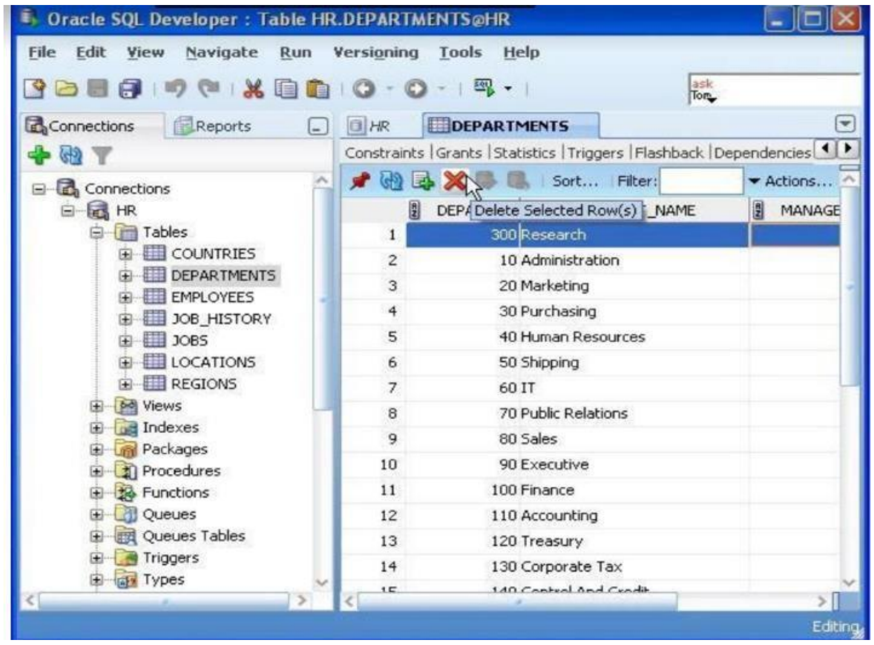
Removing Data

- Deleting a Row
- Deleting multiple rows using SQL

As with the previous two examples, you can use the SQL Worksheet to delete single or multiple rows, or you can use the Data tab.

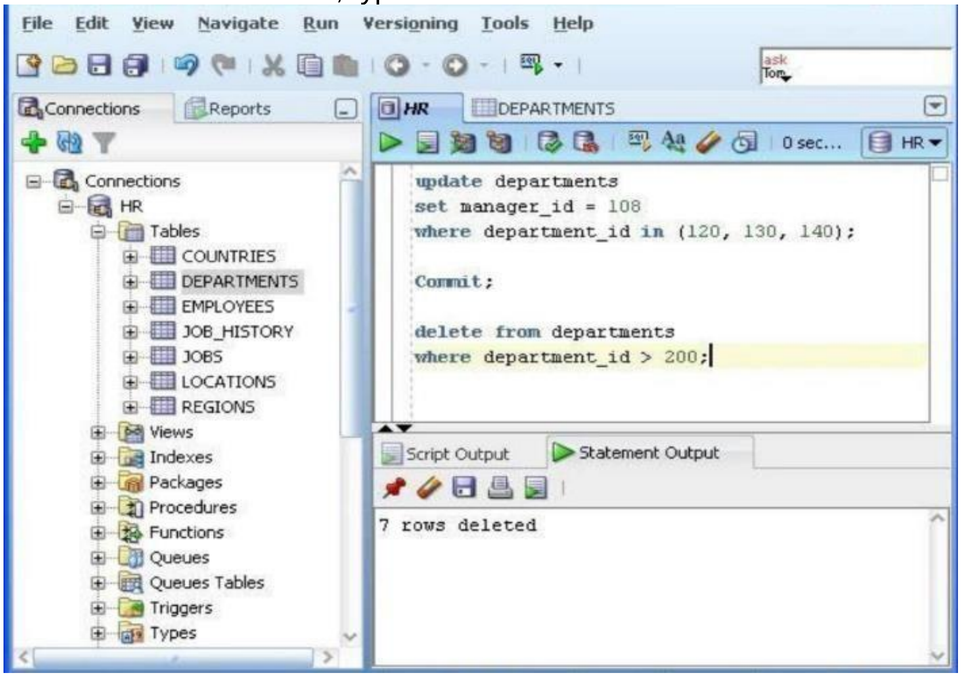
1. Return to the DEPARTMENTS data tab and select and delete the new record you inserted.

```
delete from departments
where department_id > 200;
```



	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGE
1	300	Research	
2	10	Administration	
3	20	Marketing	
4	30	Purchasing	
5	40	Human Resources	
6	50	Shipping	
7	60	IT	
8	70	Public Relations	
9	80	Sales	
10	90	Executive	
11	100	Finance	
12	110	Accounting	
13	120	Treasury	
14	130	Corporate Tax	

2.This row is not deleted, i.e. the changes are not commited to the database, until you click the Commit Changes button 3. Finally, return to the SQL Worksheet and delete a selection of rows, type



```
update departments
set manager_id = 108
where department_id in (120, 130, 140);

Commit;

delete from departments
where department_id > 200;
```

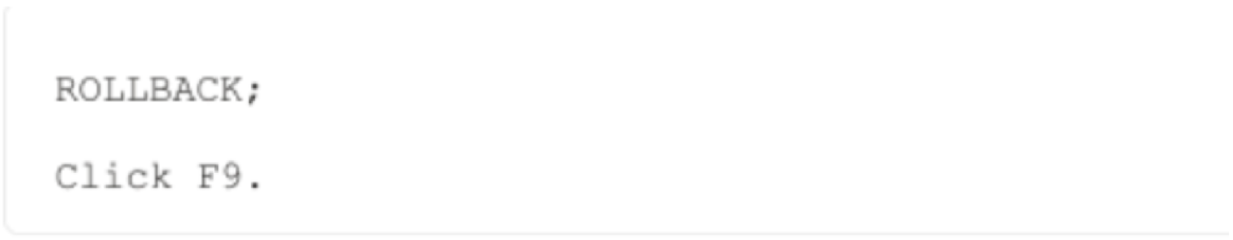
```
7 rows deleted
```

Note: You can use F9 to execute the last statement, or F5 to execute all in the SQL Worksheet. If you want to use F5 for a single statement then you can select the statement and click F5.

HINT : CTRL + Enter will execute the single statement your cursor is on.

4. As before, these changes are not saved to the database. In order to undo any changes you have made, type

The delete action you issued, has now been reversed.

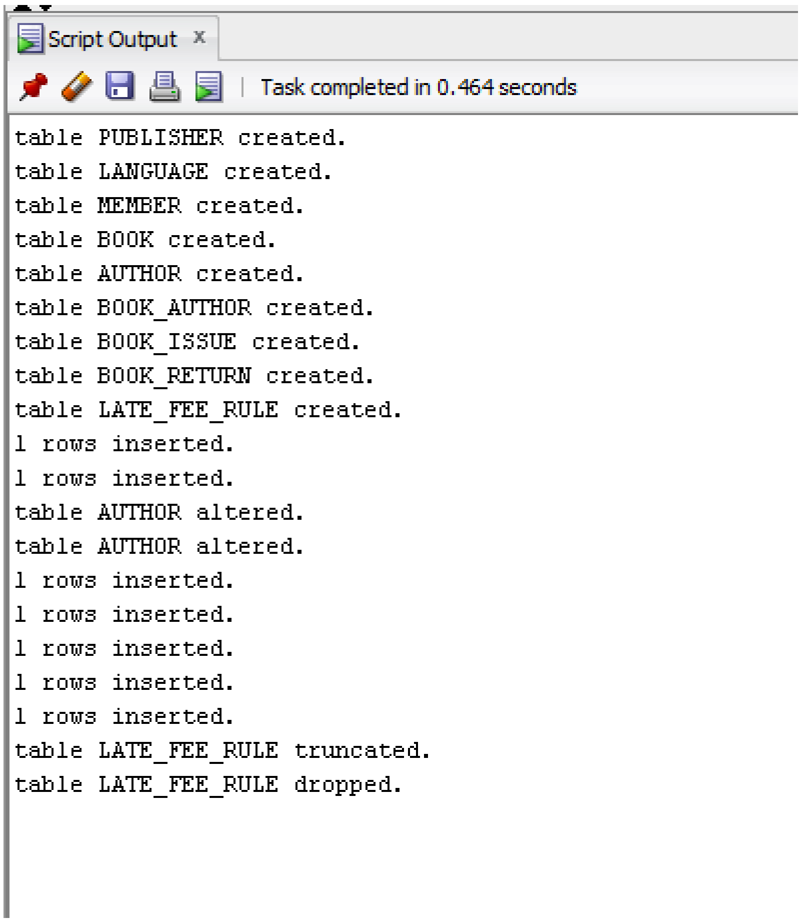


```
ROLLBACK;

Click F9.
```

Exp 3
Queries

```
create table PUBLISHER(
Publisher_id varchar(6),
Name varchar(20),
Address varchar(20),
primary key (Publisher_id));

create table LANGUAGE(
language_id varchar(6),
name varchar(20),
primary key (language_id));

create table MEMBER(
Member_Id varchar(6),
name varchar(20),
Branch_Code varchar(3),
Roll_Number int,
Phone_Number varchar(10),
Email_Id varchar(30),
Date_of_Join date,
Status varchar(20),
primary key (Member_Id));

create table BOOK (
Book_Id varchar(6),
Title varchar(30),
Language_Id varchar(6),
MRP number(6,2), Publisher_Id varchar(6), Published_Date date,
Volume varchar(3), Status varchar(20),
primary key (Book_Id),
foreign key(Language_Id) references LANGUAGE(language_id),
foreign key(Publisher_Id) references PUBLISHER(Publisher_id));

create table AUTHOR(
Author_id varchar(6),
name varchar(20),
phone_number int,
email varchar(30),
status varchar(20),
primary key(Author_id));

create table BOOK_AUTHOR(
Book_Id varchar(6),
Author_Id varchar(6),
primary key (Book_Id,Author_Id),
foreign key(Book_Id) references BOOK(Book_ID),
foreign key (Author_id) references Author(Author_id));

create table BOOK_ISSUE(
Issue_Id varchar(6),
Date_Of_Issue date,
Book_Id varchar(6),
Member_Id varchar(6),Expected_Date_Of_Return date, Status varchar(20),
primary key(Issue_Id),foreign key(Book_id) references Book(book_id),
foreign key(member_id) references member(member_id));

create table BOOK_RETURN(
Issue_Id varchar(6),
Actual_Date_Of_Return date,
LateDays int,
LateFee int,
primary key(Issue_id),
foreign key(Issue_id) references Book_issue(Issue_id));

create table LATE_FEE_RULE(
FromDays int,
ToDays int,
Amount number(5,2));

insert into late_fee_rule values(1,7,10);
insert into late_fee_rule values (8,30,100);
alter table AUTHOR drop column phone_number;
alter table AUTHOR ADD (phone_number varchar(10));
insert into author values('aut123','wikie','authorwikie@gmail.com','working','9494949494');
insert into author
values('aut124','walky','authorwalky@gmail.com','working','9494565656');
insert into language values('la34','english');
insert into publisher values('pub12','DC Books','India');
insert into book values('b1','lord of the rings','la34',1234.50,'pub12','20-11-
2001','0ne','published');

truncate table late_fee_rule;

drop table late_fee_rule;
```

Output



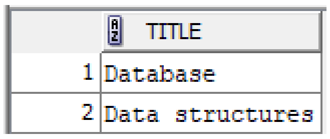
```
table PUBLISHER created.
table LANGUAGE created.
table MEMBER created.
table BOOK created.
table AUTHOR created.
table BOOK_AUTHOR created.
table BOOK_ISSUE created.
table BOOK_RETURN created.
table LATE_FEE_RULE created.
1 rows inserted.
1 rows inserted.
table AUTHOR altered.
table AUTHOR altered.
1 rows inserted.
1 rows inserted.
1 rows inserted.
1 rows inserted.
1 rows inserted.
table LATE_FEE_RULE truncated.
table LATE_FEE_RULE dropped.
```

Exp 5

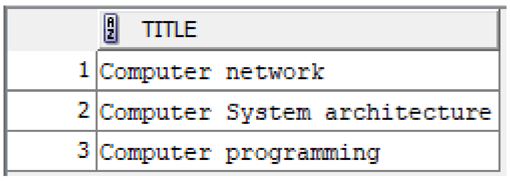
Queries

select * from book1;
select title from book1 where title like 'D%';
select title from book1 where title like '%Computer%';
select upper(title) as title from book1;
select title from book1 where title like 'D_t%struct%';
select title from book1 where title like '%e';
select reverse(title) from book1;

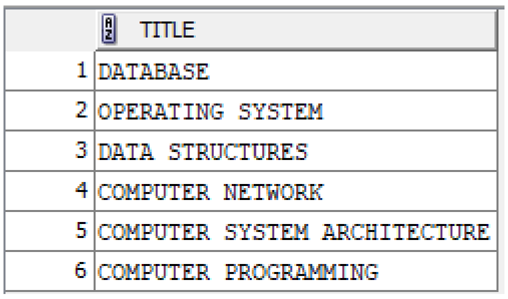
Output



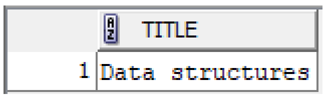
	TITLE
1	Database
2	Data structures



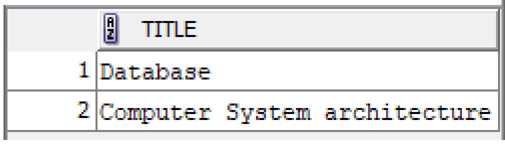
	TITLE
1	Computer network
2	Computer System architecture
3	Computer programming



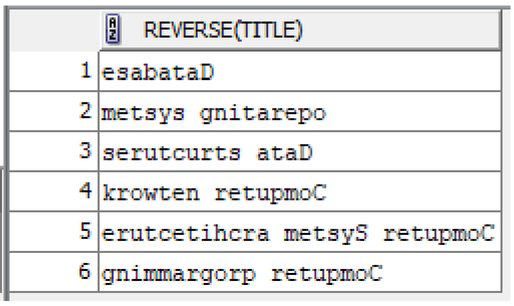
	TITLE
1	DATABASE
2	OPERATING SYSTEM
3	DATA STRUCTURES
4	COMPUTER NETWORK
5	COMPUTER SYSTEM ARCHITECTURE
6	COMPUTER PROGRAMMING



	TITLE
1	Data structures



	TITLE
1	Database
2	Computer System architecture



	REVERSE(TITLE)
1	esabataD
2	metsys gnitarepo
3	serutcurts ataD
4	krowten retupmoC
5	erutcetihcra metsyS retupmoC
6	gnimmargorp retupmoC

Exp 6:

Queries

```
--Q1
select ceil(8.29) from dual;

--Q2
select floor(9.76) from dual;

--Q3
select sqrt(100) from dual;

--Q4
SELECT GREATEST('tomin','vivek','tanya') from dual;
SELECT least('tomin','vivek','tanya') from dual;

--Q5
select current_date from dual;

--Q6
select to_char(current_date,'MM-DD-YYYY') from dual;

--Q7
select abs(8.29) from book;

--Q8
create table angle(
ANGLE float,
sin float,
cos float,
tan float,
cot float,
sec float,
primary key (angle));

insert into angle(angle) values(0);
insert into angle(angle) values(30);
insert into angle(angle) values(45);
insert into angle(angle) values(60);
insert into angle(angle) values(90);
update angle set
sin=round(sin((angle*3.14)/180),2),
cos=round(cos((angle*3.14)/180),2),
tan=round(tan((angle*3.14)/180),2);
```

```
update angle set cot=round(1/tan,2) where tan!=0;
update angle set sec=round(1/cos,2) where cos!=0;

select * from angle;

--Q9
select reverse( 'nmutuAotedOehT') from dual;
select ltrim('123231xyzTech','123xyz') from dual;
select rtrim('Computer     ') from dual;
select rpad('computer',12,'x') from dual;
select length('Database Management Systems') as length from dual;
select concat('Julius ','Ceasur') as string from dual;
select substr('India is my country',7,2) from dual;
```

Output

```
CEIL(8.29)
----------
9
FLOOR(9.76)
-----------
9
SQRT(100)
----------
10
GREATEST('TOMIN','VIVEK','TANYA')
---------------------------------
vivek
LEAST('TOMIN','VIVEK','TANYA')
------------------------------
tanya
CURRENT_DATE
------------
26-10-22
TO_CHAR(CURRENT_DATE,'MM-DD-YYYY')
----------------------------------
10-26-2022
ABS(8.29)
----------
8.29
table ANGLE created.
```

```
1 rows inserted.
1 rows inserted.
1 rows inserted.
1 rows inserted.
1 rows inserted.
5 rows updated.
4 rows updated.
4 rows updated.
ANGLE      SIN      COS      TAN      COT      SEC
---------- ---------- ---------- ---------- ---------- ----------
0        0        1        0                 1
30       0.5      0.87      0.58      1.72      1.15
45      0.71      0.71         1         1      1.41
60      0.87       0.5      1.73      0.58         2
90         1         0   1255.77         0
REVERSE('NMUTUAOTEDOEHT')
------------------------
TheOdetoAutumn
LTRIM('123231XYZTECH','123XYZ')
------------------------------
Tech
RTRIM('COMPUTER')
----------------
Computer
RPAD('COMPUTER',12,'X')
----------------------
computerxxxx
LENGTH
----------
27

STRING
------------
Julius Ceasur

SUBSTR('INDIAISMYCOUNTRY',7,2)
-----------------------------
is
```

Exp 7:

Queries

```
create table student(
rollNo int,name varchar(20),
physics numeric(3,1),
chemistry numeric(3,1),
maths numeric(3,1),
primary key (rollNo)
);

--Q1
select avg(physics) from student;

--Q2
select max(maths) from student;

--Q3
select min(chemistry) from student;

--Q4
select count(physics) from student where physics>12;

--Q5
select rollNo,name from student where maths>25 and physics>12 and chemistry>12;

--Q6
select rollNo,name,physics+chemistry+maths as total from student order by total desc ;

--Q7
select (count(name)*10) as pass_maths from student where maths>=25;

--Q8
select (count(name)*10) as pass_all from student where STATUS='pass';

--Q9
select avg(maths+physics+chemistry) as class_avg from student;

--Q10
select count(name) as total_pass from student where status='pass'
```

Output

```
AVG(PHYSICS)
------------
16
MAX(MATHS)
----------
41
MIN(CHEMISTRY)
--------------
7
COUNT(PHYSICS)
--------------
8
ROLLNO NAME
---------- --------------------
1 ABC
ROLLNO NAME                 TOTAL
---------- -------------------- ----------
1 ABC                    73
2 CMD                     70
10 YWA                     67
4 RZT                    63
5 UNM                     60
9 QZX                    58
8 WLE                     57
3 PQY                    56
6 HGT                     38
7 XYZ                    35
10 rows selected
PASS_MATHS
----------
50
PASS_ALL
----------
20
CLASS_AVG
----------
57.7
```

Exp 8:

Queries

```
create table Customer(
customerID varchar(6),
customerName varchar(30),
contactName varchar(30),
address varchar(50),
city varchar(20),
postalCode varchar(6),
country varchar(20),
primary key (customerID));

create table employees(
employeeID varchar(6),
lastname varchar(10),
firstname varchar(10),
birthDate date,
primary key(employeeID));

create table Orders(
orderId varchar(6),
customerID varchar(6),
employeeID varchar(6),
orderDate date,
shipperID varchar(6),
primary key(orderID),
foreign key (employeeID) references employees(employeeID),
foreign key (customerID) references Customer(customerID));

select count(customerID) as no_of_customers,country from customer
group by country
having count(customerID)>5;

select count(customerID) as no_of_customers,country from customer
group by country
having count(customerID)>5
order by count(customerID) desc;

select o.employeeid,e.firstname,e.lastname from orders o
join employees e on o.employeeid=e.employeeid
group by o.employeeid,e.firstname,e.lastname having count(o.employeeid)>=9;
```

Output

```
NO_OF_CUSTOMERS COUNTRY
--------------- --------------------
6 pakisthan
7 India

NO_OF_CUSTOMERS COUNTRY
--------------- --------------------
7 India
6 pakisthan

EMPLOYEEID FIRSTNAME  LASTNAME
---------- ---------- ----------
3          joseph     jacob
1          johns      raju
```

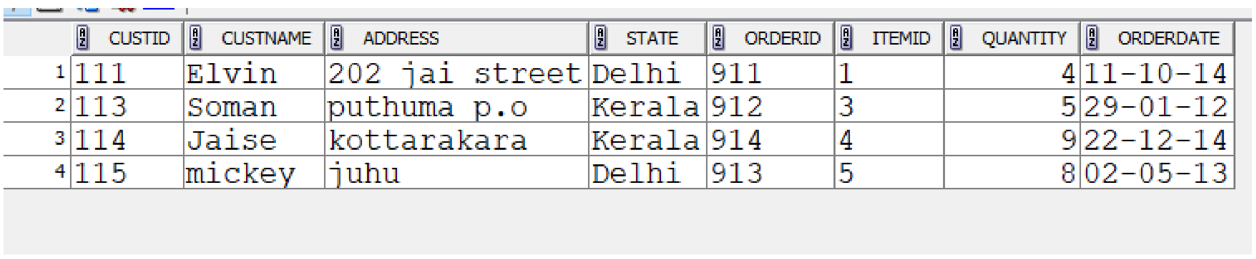
Exp 9:

Queries

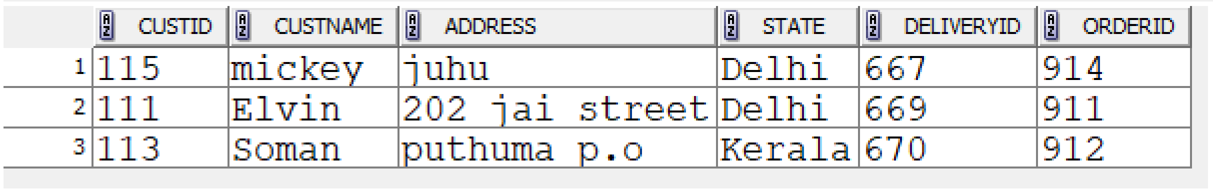
```
--Q1
Select * from customers natural join orders ;

--q2
Select * from customers natural join delivery ;

--q3
Select orderdate from customers natural join orders
Where custname like 'j%';

--q4
Select itemname,price from items natural join orders
Natural join customers where custname='mickey';

--q5
Select * from
Customers c join orders o on o.custid = c.custid
Left join delivery d on d.custid = c.custid
Where o.orderdate >= '01-02-14'
And d.deliveryid is null;

--q6
Select itemid from orders
Union
Select o.itemid from orders o left join delivery d
On d.custid=o.custid where d.deliveryid is null;

--q7
Select customers.custname
From customers inner join
Delivery on delivery.custid=customers.custid;

--q8
Select customers.custname from customers
Join orders on customers.custid=orders.custid
Minus (select customers.custname from customers
Join orders on customers.custid=orders.custid
Join delivery on delivery.custid=orders.custid);

--q9
Select customers.custname from customers
Inner join orders on orders.custid=customers.custid
Where orders.quantity=(select max(orders.quantity) from orders);
```

```
--q10
Select * from customers inner join orders on orders.custid=customers.custid
Join items on items.itemid=orders.itemid where items.price >5000;

--q11
Select custname,address from customers minus
Select custname,address from customers c
Join orders o on c.custid=o.custid
Join items i on i.itemid=o.itemid
Where i.itemname like 'galaxy';

--q12
Select * from customers right join orders on customers.custid=orders.custid;
Select * from customers left join orders on customers.custid=orders.custid;

--q13
Select * from customers order by state;

--q14
Select * from items where price >(select avg(price)from items) order by category;
```

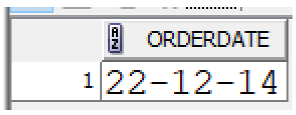
Output



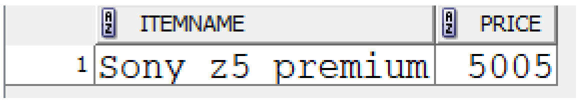
	CUSTID	CUSTNAME	ADDRESS	STATE	ORDERID	ITEMID	QUANTITY	ORDERDATE
1	111	Elvin	202 jai street	Delhi	911	1	4	11-10-14
2	113	Soman	puthuma p.o	Kerala	912	3	5	29-01-12
3	114	Jaise	kottarakara	Kerala	914	4	9	22-12-14
4	115	mickey	juhu	Delhi	913	5	8	02-05-13



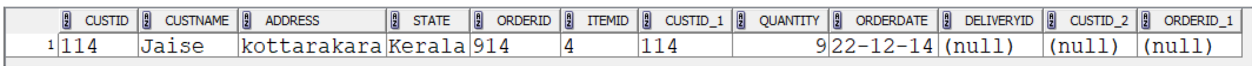
	CUSTID	CUSTNAME	ADDRESS	STATE	DELIVERYID	ORDERID
1	115	mickey	juhu	Delhi	667	914
2	111	Elvin	202 jai street	Delhi	669	911
3	113	Soman	puthuma p.o	Kerala	670	912



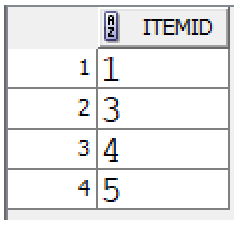
	ORDERDATE
1	22-12-14



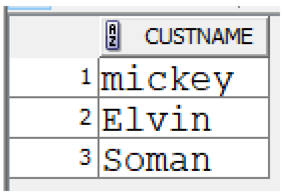
	ITEMNAME	PRICE
1	Sony z5 premium	5005



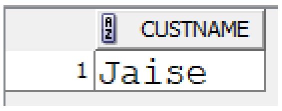
	CUSTID	CUSTNAME	ADDRESS	STATE	ORDERID	ITEMID	CUSTID_1	QUANTITY	ORDERDATE	DELIVERYID	CUSTID_2	ORDERID_1
1	114	Jaise	kottarakara	Kerala	914	4	114	9	22-12-14	(null)	(null)	(null)



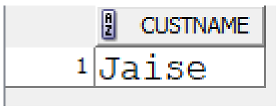
	ITEMID
1	1
2	3
3	4
4	5



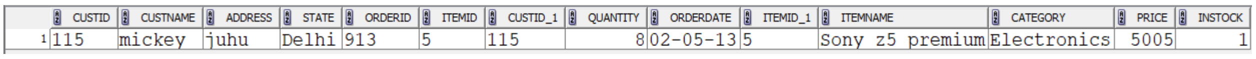
	CUSTNAME
1	mickey
2	Elvin
3	Soman



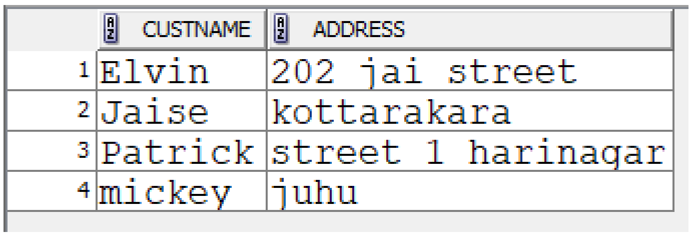
	CUSTNAME
1	Jaise



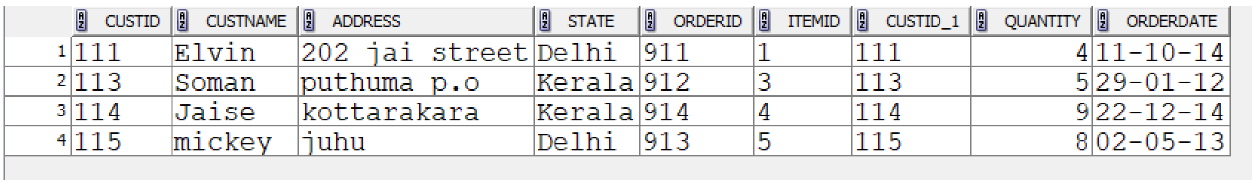
	CUSTNAME
1	Jaise



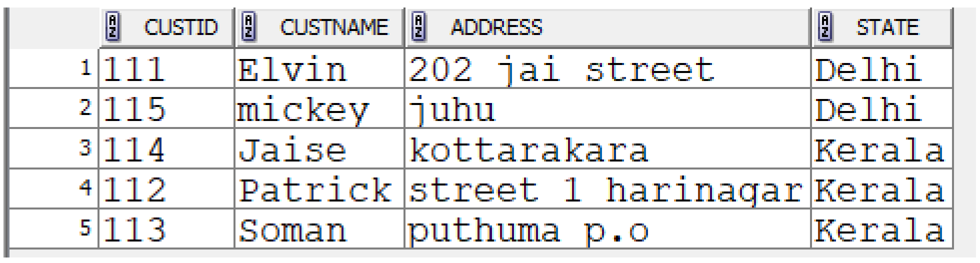
	CUSTID	CUSTNAME	ADDRESS	STATE	ORDERID	ITEMID	CUSTID_1	QUANTITY	ORDERDATE	ITEMID_1	ITEMNAME	CATEGORY	PRICE	INSTOCK
1	115	mickey	juhu	Delhi	913	5	115	8	02-05-13	5	Sony z5 premium	Electronics	5005	1



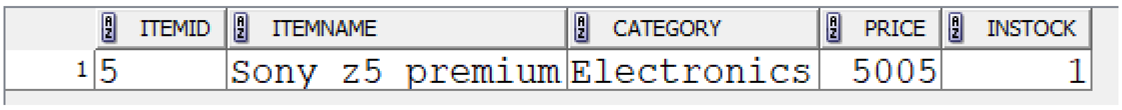
	CUSTNAME	ADDRESS
1	Elvin	202 jai street
2	Jaise	kottarakara
3	Patrick	street 1 harinagar
4	mickey	juhu



	CUSTID	CUSTNAME	ADDRESS	STATE	ORDERID	ITEMID	CUSTID_1	QUANTITY	ORDERDATE
1	111	Elvin	202 jai street	Delhi	911	1	111	4	11-10-14
2	113	Soman	puthuma p.o	Kerala	912	3	113	5	29-01-12
3	114	Jaise	kottarakara	Kerala	914	4	114	9	22-12-14
4	115	mickey	juhu	Delhi	913	5	115	8	02-05-13



	CUSTID	CUSTNAME	ADDRESS	STATE
1	111	Elvin	202 jai street	Delhi
2	115	mickey	juhu	Delhi
3	114	Jaise	kottarakara	Kerala
4	112	Patrick	street 1 harinagar	Kerala
5	113	Soman	puthuma p.o	Kerala



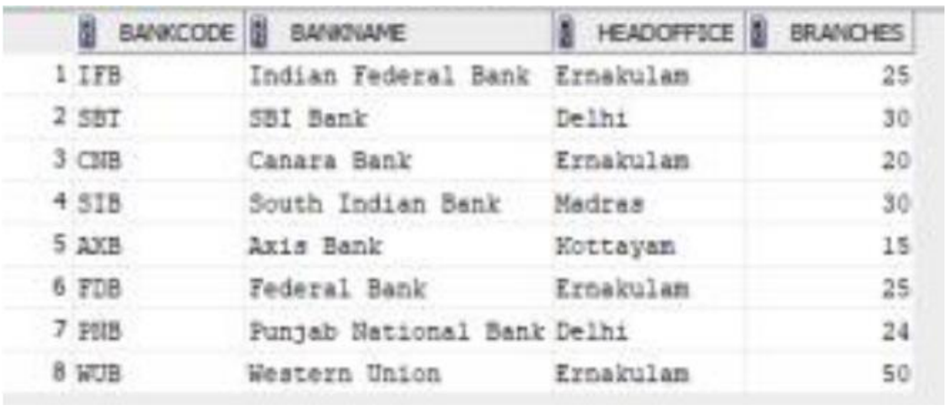
	ITEMID	ITEMNAME	CATEGORY	PRICE	INSTOCK
1	5	Sony z5 premium	Electronics	5005	1

Exp 10:

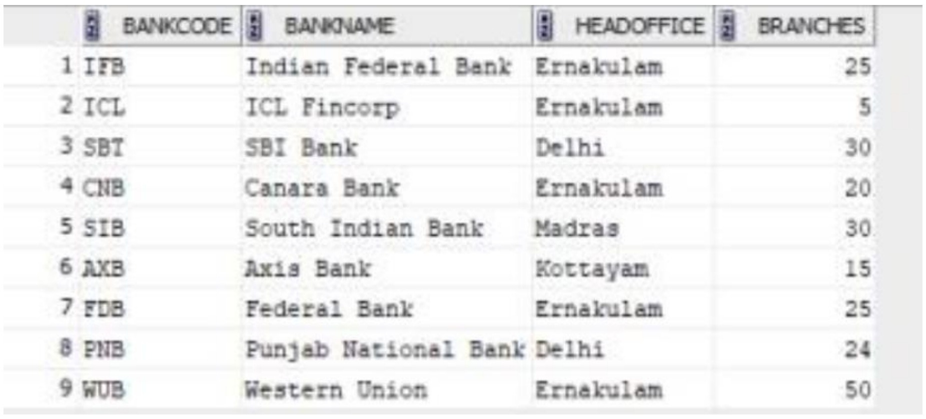
Queries

```
CREATE TABLE BANK(
bankcode varchar(3),
bankname varchar(30) not null,
headoffice varchar(30),
branches int not null check(branches>0),
primary key(bankcode) );

insert into BANK values('SBT','SBI Bank','Delhi',30);
insert into BANK values('CNB','Canara Bank','Ernakulam',20);
insert into BANK values('SIB','South Indian Bank','Madras',30);
insert into BANK values('AXB','Axis Bank','Kottayam',15);
insert into BANK values('FDB','Federal Bank','Ernakulam',25);

insert into BANK values('IFB','Indian Federal Bank','Ernakulam',25);
commit;
select * from BANK


insert into BANK values('ICL','ICL Fincorp','Ernakulam',5);
SAVEPOINT A;
insert into BANK values('GMB','Grameen Bank','Pune',15);
SAVEPOINT B;
select * from BANK


ROLLBACK TO B;
select * from BANK
```

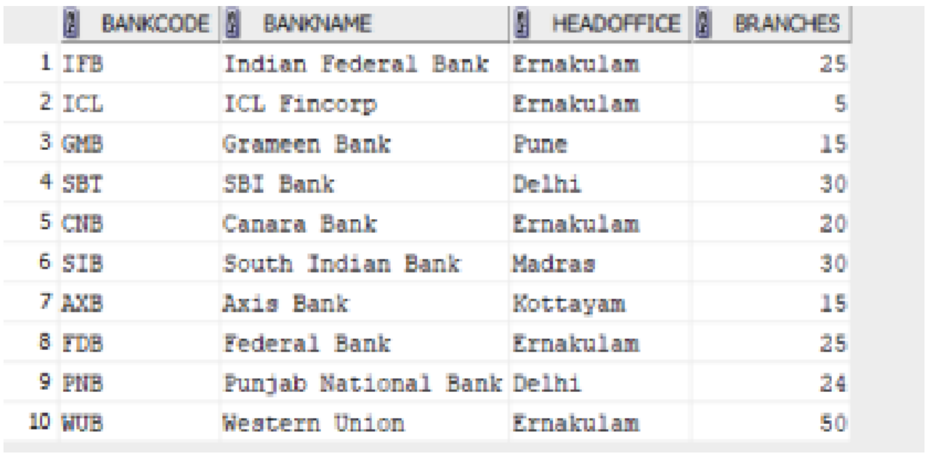
Output



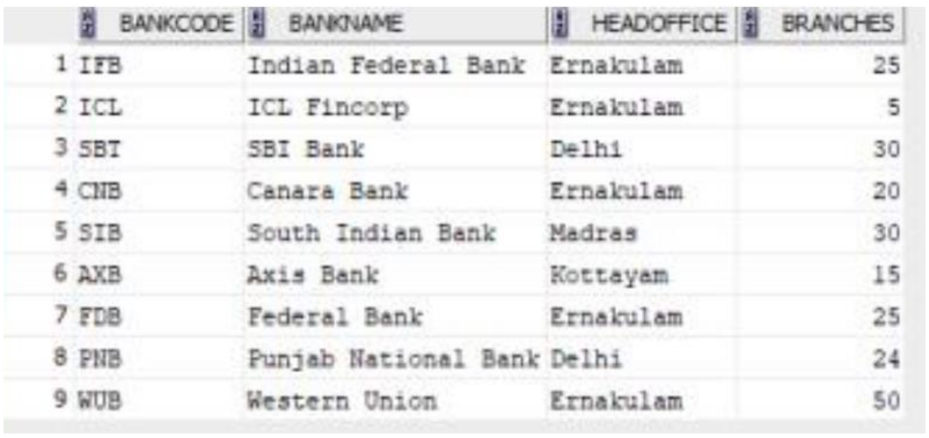
	BANKCODE	BANKNAME	HEADOFFICE	BRANCHES
1	IFB	Indian Federal Bank	Ernakulam	25
2	SBT	SBI Bank	Delhi	30
3	CNB	Canara Bank	Ernakulam	20
4	SIB	South Indian Bank	Madras	30
5	AXB	Axis Bank	Kottayam	15
6	FDB	Federal Bank	Ernakulam	25
7	PNB	Punjab National Bank	Delhi	24
8	WUB	Western Union	Ernakulam	50



	BANKCODE	BANKNAME	HEADOFFICE	BRANCHES
1	IFB	Indian Federal Bank	Ernakulam	25
2	ICL	ICL Fincorp	Ernakulam	5
3	SBT	SBI Bank	Delhi	30
4	CNB	Canara Bank	Ernakulam	20
5	SIB	South Indian Bank	Madras	30
6	AXB	Axis Bank	Kottayam	15
7	FDB	Federal Bank	Ernakulam	25
8	PNB	Punjab National Bank	Delhi	24
9	WUB	Western Union	Ernakulam	50



	BANKCODE	BANKNAME	HEADOFFICE	BRANCHES
1	IFB	Indian Federal Bank	Ernakulam	25
2	ICL	ICL Fincorp	Ernakulam	5
3	GMB	Grameen Bank	Pune	15
4	SBT	SBI Bank	Delhi	30
5	CNB	Canara Bank	Ernakulam	20
6	SIB	South Indian Bank	Madras	30
7	AXB	Axis Bank	Kottayam	15
8	FDB	Federal Bank	Ernakulam	25
9	PNB	Punjab National Bank	Delhi	24
10	WUB	Western Union	Ernakulam	50



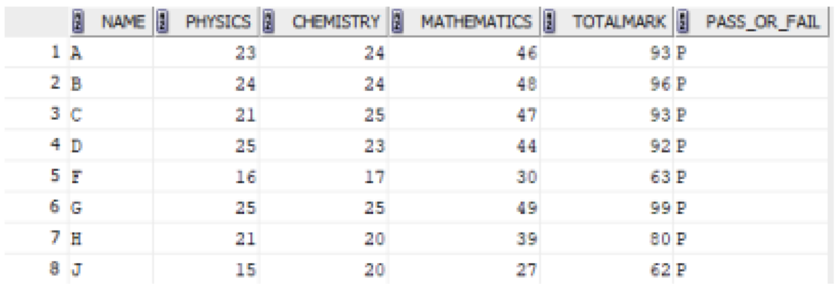
	BANKCODE	BANKNAME	HEADOFFICE	BRANCHES
1	IFB	Indian Federal Bank	Ernakulam	25
2	ICL	ICL Fincorp	Ernakulam	5
3	SBT	SBI Bank	Delhi	30
4	CNB	Canara Bank	Ernakulam	20
5	SIB	South Indian Bank	Madras	30
6	AXB	Axis Bank	Kottayam	15
7	FDB	Federal Bank	Ernakulam	25
8	PNB	Punjab National Bank	Delhi	24
9	WUB	Western Union	Ernakulam	50

Exp 11:

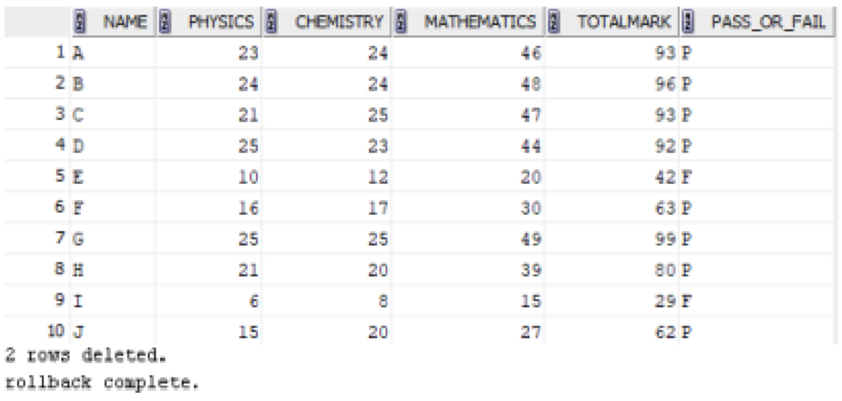
Queries

```
select *from STUDENT
delete from STUDENT where PASS_OR_FAIL='F';
ROLLBACK;

select *from STUDENT
commit
SAVEPOINT
savepoint SP1;

delete from STUDENT where TOTALMARK>85;
ROLLBACK TO SP1;

grant select on STUDENT to C19CSB14
select *from C19CSB14.STUDENT

revoke select
on student from c19csb14
```

Output

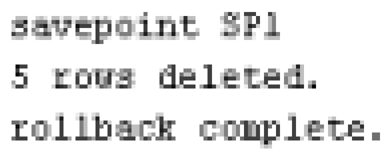


	NAME	PHYSICS	CHEMISTRY	MATHEMATICS	TOTALMARK	PASS_OR_FAIL
1	A	23	24	46	93	P
2	B	24	24	48	96	P
3	C	21	25	47	93	P
4	D	25	23	44	92	P
5	F	16	17	30	63	P
6	G	25	25	49	99	P
7	H	21	20	39	80	P
8	J	15	20	27	62	P

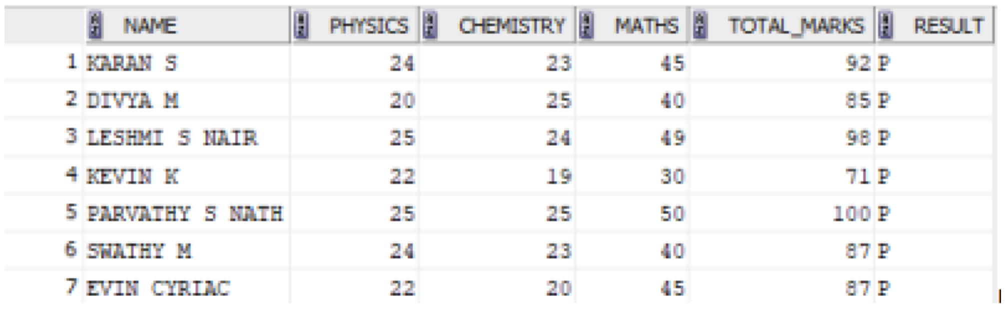


	NAME	PHYSICS	CHEMISTRY	MATHEMATICS	TOTALMARK	PASS_OR_FAIL
1	A	23	24	46	93	P
2	B	24	24	48	96	P
3	C	21	25	47	93	P
4	D	25	23	44	92	P
5	E	10	12	20	42	F
6	F	16	17	30	63	P
7	G	25	25	49	99	P
8	H	21	20	39	80	P
9	I	6	8	15	29	F
10	J	15	20	27	62	P

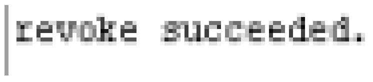
```
2 rows deleted.
rollback complete.
```



```
savepoint SP1
5 rows deleted.
rollback complete.
```



	NAME	PHYSICS	CHEMISTRY	MATHS	TOTAL_MARKS	RESULT
1	KARAN S	24	23	45	92	P
2	DIVYA M	20	25	40	85	P
3	LESHMI S NAIR	25	24	49	98	P
4	KEVIN K	22	19	30	71	P
5	PARVATHY S NATH	25	25	50	100	P
6	SWATHY M	24	23	40	87	P
7	EVIN CYRIAC	22	20	45	87	P



```
revoke succeeded.
```

Exp 12:

Queries

```
CREATE TABLE Bank(bankcode VARCHAR(3),
bankname varchar(30) not null,
headoffice varchar(15),
branches int not null check(branches>0),
PRIMARY KEY(bankcode));

INSERT INTO Bank VALUES('SBI','STATE BANK INDIA','Delhi','50');

INSERT INTO Bank VALUES('FB','FEDERAL BANK','Eranakulam','40');

INSERT INTO Bank VALUES('SBT','STATE BANK TRAVANCORE','Kottayam','35');

INSERT INTO Bank VALUES('AX','AXIS BANK ','Eranakulam','25');

INSERT INTO Bank VALUES('SIB','SOUTH INDIAN BANK ','Eranakulam','38');
SELECT *FROM Bank;

create table Branch1(branchid varchar(3),
branchname varchar(30) DEFAULT'New Delhi',
bankcode varchar(3),
PRIMARY KEY(branchid) );

ALTER TABLE Branch1
ADD FOREIGN KEY(bankcode)
REFERENCES Bank(bankcode);

INSERT INTO Branch1 VALUES('101', DEFAULT,'SBI');
INSERT INTO Branch1 VALUES('102', DEFAULT,'FB');
INSERT INTO Branch1 VALUES('103', DEFAULT,'SBT');
INSERT INTO Branch1 VALUES('104', DEFAULT,'AX');
INSERT INTO Branch1 VALUES('105', DEFAULT,'SIB');


SELECT * FROM branch1;

DELETE FROM BRANCH1  WHERE bankcode LIKE 'SBT';
DELETE FROM BANK  WHERE bankcode LIKE 'SBT';
INSERT INTO Branch1 VALUES('106','Kottayam','SBI');
SELECT * FROM branch1;

ALTER TABLE branch1
DROP PRIMARY KEY;

CREATE VIEW bank_head_office AS SELECT bankcode,bankname,headoffice,branches FROM
bank WHERE headoffice LIKE 'Eranakulam';
SELECT * FROM bank_head_office;

CREATE VIEW bank_branch AS SELECT branchid,branchname,bankcode FROM branch1
WHERE branchname LIKE 'Kottayam';
select * from bank_branch;
```

Output

```
table BANK created.
1 rows inserted.
1 rows inserted.
1 rows inserted.
1 rows inserted.
1 rows inserted.

BANKCODE BANKNAME                     HEADOFFICE       BRANCHES
-------- ------------------------------ --------------- ----------
SBI      STATE BANK INDIA               Delhi                50
FB       FEDERAL BANK                   Eranakulam           40
SBT      STATE BANK TRAVANCORE          Kottayam             35
AX       AXIS BANK                      Eranakulam           25
SIB      SOUTH INDIAN BANK              Eranakulam           38

table BRANCH1 created.
table BRANCH1 altered.
1 rows inserted.
1 rows inserted.
1 rows inserted.
1 rows inserted.
1 rows inserted.

BRANCHID BRANCHNAME                     BANKCODE
```

```
-------- ----------------------------- --------
101      New Delhi                   SBI
102      New Delhi                   FB
103      New Delhi                   SBT
104      New Delhi                   AX
105      New Delhi                   SIB

1 rows deleted.
1 rows deleted.
1 rows inserted.

BRANCHID BRANCHNAME                    BANKCODE
-------- ----------------------------- --------
101      New Delhi                   SBI
102      New Delhi                   FB
104      New Delhi                   AX
105      New Delhi                   SIB
106      Kottayam                    SBI

table BRANCH1 altered.
view BANK_HEAD_OFFICE created.

BANKCODE BANKNAME                     HEADOFFICE       BRANCHES
-------- ----------------------------- --------------- ----------
FB      FEDERAL BANK                 Eranakulam          40
AX      AXIS BANK                    Eranakulam          25
SIB     SOUTH INDIAN BANK            Eranakulam          38

view BANK_BRANCH created.
BRANCHID BRANCHNAME                    BANKCODE
-------- ----------------------------- --------
106      Kottayam                    SBI
```

Exp 13:

Queries

a) FACTORIAL PROGRAM:

```
declare
f number:=1;
a number;
i number:=1;
begin
a:=:a;
if(a=0)
then
dbms_output.put_line('Factorial is: 1');
else
while(i<=a)
loop
f:=f*i;
i:=i+1;
end loop;
dbms_output.put_line('Factorial is '||f);
end if;
end;
```

b) TO FIND THE GREATEST OF THREE NUMBERS PROGRAM:

```
declare
a number:=7;
b number:=1;
c number:=5;
begin

dbms_output.put_line('a='||a||' b='||b||' c='||c);
if a>b AND a>c
then
dbms_output.put_line('a is greatest');
else
if b>a AND b>c
then
dbms_output.put_line('b is greatest');
else
dbms_output.put_line('c is greatest');
end if;
end if;
end;
```

c) TO IMPLEMENT A CALCULATOR PROGRAM:

```
declare
a number(3);
b number(3);
c number(3);
d number(3);
begin
dbms_output.put_line('Enter 2 numbers');
a:=:a;
b:=:b;
dbms_output.put_line('Enter 1 for Addition');
dbms_output.put_line('Enter 2 for Substraction');
dbms_output.put_line('Enter 3 for Multiplication');
dbms_output.put_line('Enter 4 for Division');
dbms_output.put_line('Enter your Choice');
c:=:c;
case c
WHEN 1 THEN d:=:a+b ;
WHEN 2 THEN d:=:a-b ;
WHEN 3 THEN d:=:a*b ;
WHEN 4 THEN d:=:a/b ;
end case;
dbms_output.put_line('output'||d);
end;
```

d) TO GENERATE FIBONACCI SERIES PROGRAM:

```
declare
first number := 0;
second number := 1;
temp number;
n number := 5;
i number;
begin
dbms_output.put_line('Fibonacci Series:');
dbms_output.put_line(first);
dbms_output.put_line(second);
for i in 2..n
loop
temp:=first+second;
first := second;
second := temp;
dbms_output.put_line(temp);
end loop;
end;
```

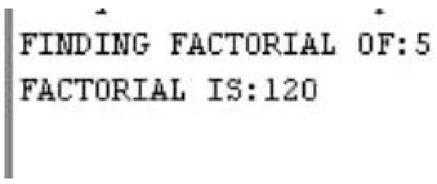
e) To show if a number is divided by zero

```
DECLARE
a int := 10;
b int := 0;
answer int;
BEGIN
answer:=a/b;
dbms_output.put_line('The Result after Division is '||answer);
exception
WHEN zero_divide THEN
dbms_output.put_line('Dividing by zero please check the values again!');
dbms_output.put_line('The value of a is '||a);
dbms_output.put_line('The value of b is '||b);
END;
```

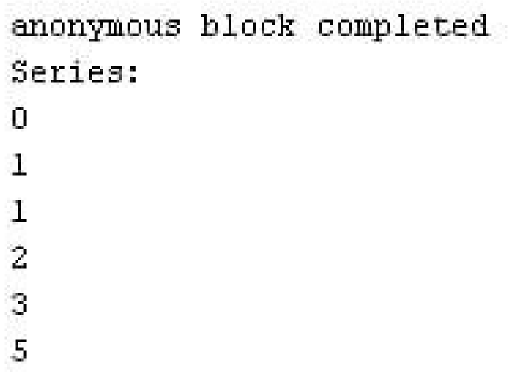
f) To show no data found exception.

```
set serveroutput on;
CREATE TABLE ebill(
cname varchar(20),
prevreading varchar(20),
currreading varchar(20)
);
DECLARE
x integer;
y integer;
z varchar2(20);
ex exception;
BEGIN
x := :prevreading;
y := :currreading;
z := :cname;
if(x = y) then
raise ex;
else
INSERT INTO ebill VALUES(z,y,x);
end if;
EXCEPTION
WHEN ex then
dbms_output.put_line('Data EntryError');
END
```

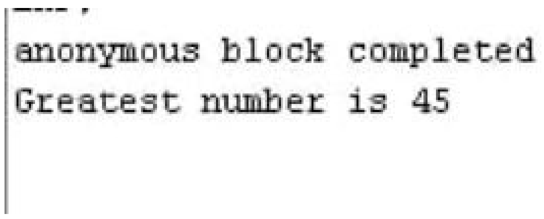
Output



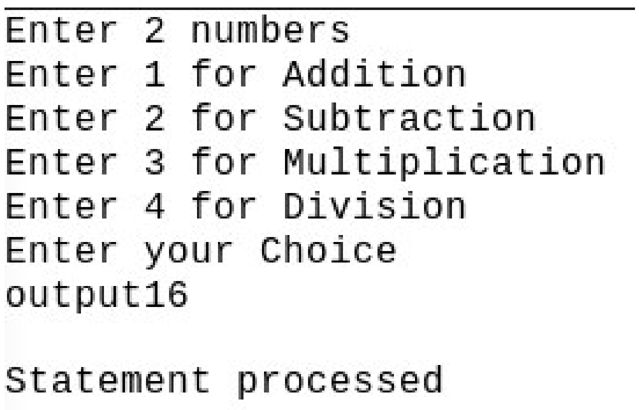
```
FINDING FACTORIAL OF:5
FACTORIAL IS:120
```



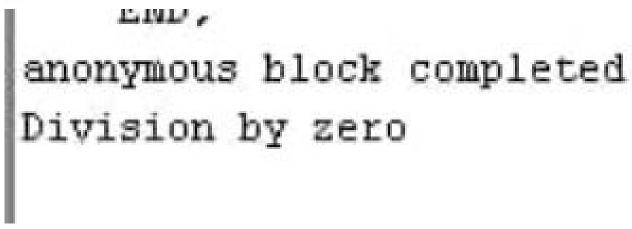
```
anonymous block completed
Series:
0
1
1
2
3
5
```



```
anonymous block completed
Greatest number is 45
```



```
Enter 2 numbers
Enter 1 for Addition
Enter 2 for Subtraction
Enter 3 for Multiplication
Enter 4 for Division
Enter your Choice
output16

Statement processed
```



```
anonymous block completed
Division by zero
```

Exp 14:

Queries

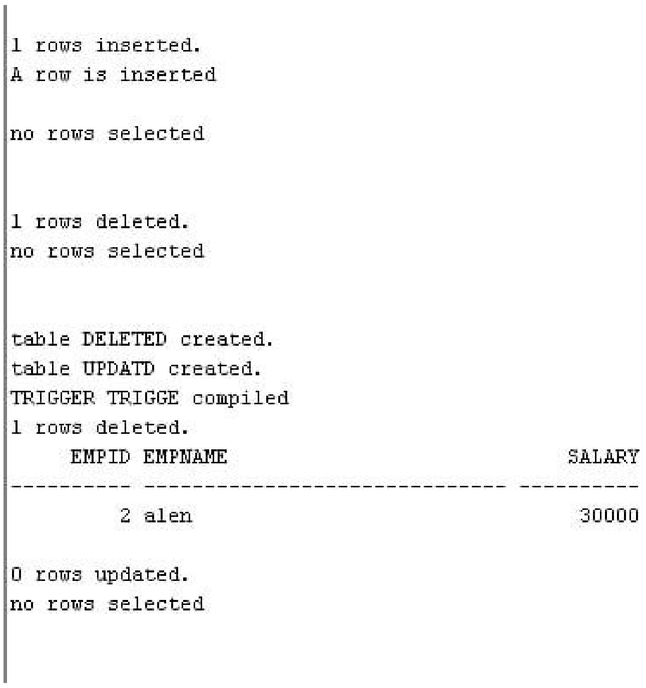
```
factorial(n int)
return int as
fact int;
begin
fact:=1;
for i in 1..n loop
fact:=fact*i;
end loop;
return fact;
end;
DECLARE
c int;
BEGIN
c := factorial(5);
DBMS_OUTPUT.PUT_LINE(' Factorial '|| NUM || ' is ' ||FACTORIAL);
END;
```

```
CREATE TABLE student_details(roll int,marks int, phone int);
INSERT INTO student_details
VALUES(1,70,9496947423);
INSERT INTO student_details
VALUES(1,47,987654321);
INSERT INTO student_details
VALUES(2,48,912365478);
INSERT INTO student_details
VALUES(3,48,6258741138);

SELECT * FROM student_details;

create or replace procedure pr1
as
begin
update student_details
set marks=marks+(marks*0.05);
end;
begin
pr1;
end;
SELECT * FROM student_details order by roll;

CREATE TABLE students(id int,name varchar(10),m1 int,m2 int,m3 int,total int,grade
varchar(1));
declare
id int;
name varchar(30);
m1 int;
m2 int;
m3 int;
t int;
begin
id:=:id;
name:=:name;
m1:=:m1;
m2:=:m2;
m3:=:m3;
t:=m1+m2+m3;
insert into students(id,name,m1,m2,m3,total)
values(id,name,m1,m2,m3,t); end;

select * from students
delete from students where id=1;
create or replace function f1
return varchar as
cursor c is select total from students;
t int;
a int;
begin
open c;
loop
fetch c into t;
exit when c%notfound;
a:=t/3;
if(a>=90) then return('A+');
elsif(a>=80) then return('A');
elsif(a>=70) then return('B');
elsif(a>=60) then return('C');
else
return('D');
end if;
end loop;
close c;
end;
create or replace procedure p1
as
c varchar(30);
begin
c:=f1();
update students
```

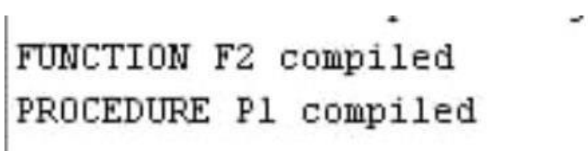
```
set grade=c;
end;
begin
p1;
end;
create table customer_details(cust_id int,cust_name varchar(30),address varchar(30));
create table emp_details(empid int,empname varchar(30),salary int); create table
cust_count(count_row int);

create trigger tri
after insert on customer_details
for each row
begin
dbms_output.put_line('A row is inserted');
end;
insert into customer_details(cust_id,cust_name,address)
,'abc');

create trigger trig
after insert on emp_details
for each row
when(new.salary>20000)
begin
dbms_output.put_line('Salary is greater than 20000');
end;
insert into emp_details(empid,empname,salary)
values(2,'alen',30000);

insert into cust_count values(0);
create trigger trigg
before insert or delete on customer_details
for each row
begin
if deleting then
update cust_count
set count_row=count_row-1;
else
update cust_count
set count_row=count_row+1;
end if;
end;
insert into customer_details(cust_id,cust_name,address)

values(1,'bic','xyz'); select * from cust_count;
delete from customer_details where cust_id=4;
select * from cust_count;
```

```
create table deleted(empid int,empname varchar(30),salary int);
create table updatd(empid int,empname varchar(30),salary int);
create or replace trigger trigge
before delete or update on emp_details
for each row
begin
if deleting then
insert into deleted
values(:old.empid,:old.empname,:old.salary);
else
insert into updatd
values(:new.empid,:new.empname,:new.salary);
end if;
end;
delete from emp_details where empid=2;
select * from deleted
update emp_details
set salary=10000 where empid=1;
select * from updatd
```

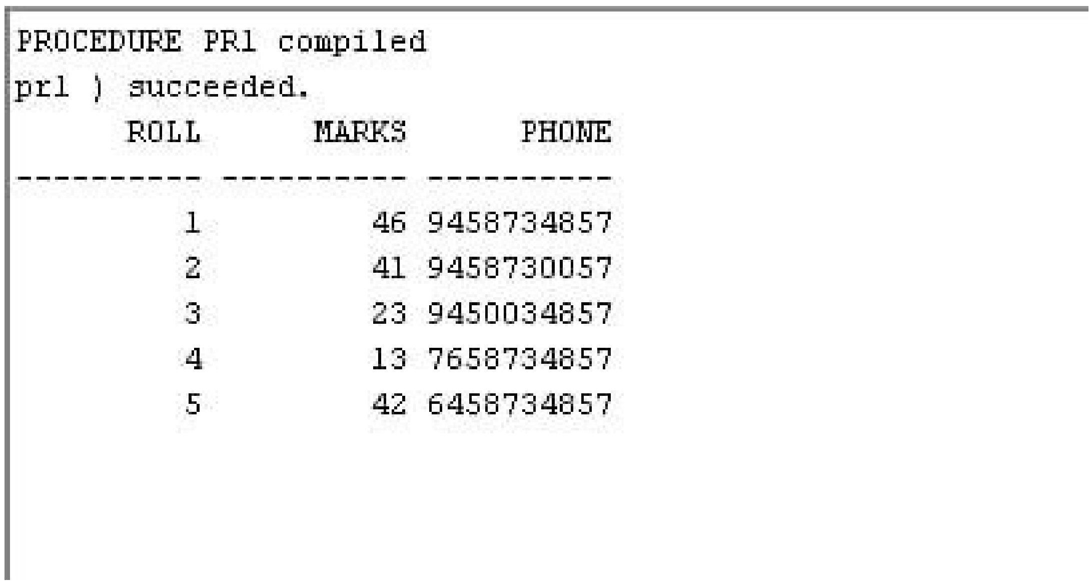
Output



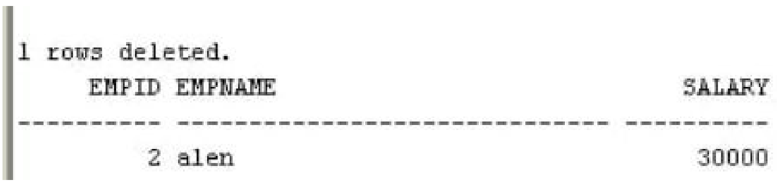
```
1 rows inserted.
A row is inserted

no rows selected


1 rows deleted.
no rows selected


table DELETED created.
table UPDATD created.
TRIGGER TRIGGE compiled
1 rows deleted.
     EMPID EMPNAME                            SALARY
---------- ------------------------------ ----------
         2 alen                                30000

0 rows updated.
no rows selected
```



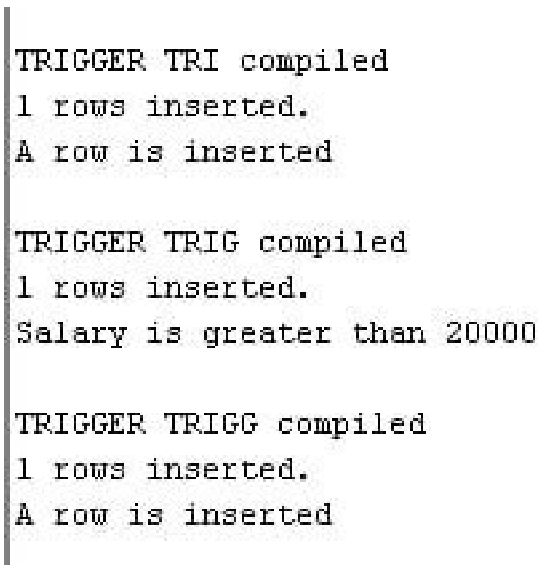
```
FUNCTION F2 compiled
PROCEDURE P1 compiled
```



```
PROCEDURE PR1 compiled
prl ) succeeded.
      ROLL      MARKS      PHONE
---------- ---------- ----------
         1         46 9458734857
         2         41 9458730057
         3         23 9450034857
         4         13 7658734857
         5         42 6458734857
```



```
1 rows deleted.
     EMPID EMPNAME                          SALARY
---------- ---------------------------- ----------
         2 alen                              30000
```



```
TRIGGER TRI compiled
1 rows inserted.
A row is inserted

TRIGGER TRIG compiled
1 rows inserted.
Salary is greater than 20000

TRIGGER TRIGG compiled
1 rows inserted.
A row is inserted
```

Exp 15:

Queries

```
--step 1
CREATE OR REPLACE PACKAGE Pk1 AS
PROCEDURE proc1(a int,b int);
PROCEDURE proc2(a int);
FUNCTION fn11(a int) return varchar2;
FUNCTION fn22(a int,b int,c int) return INT;
END Pk1;
--step 2
CREATE OR REPLACE PACKAGE BODY Pk1 AS
PROCEDURE proc1(a int,b int) IS
BEGIN
dbms_output.put_line('Sum:'||(a+b));
dbms_output.put_line('AVG:'||(a+b)/2);
dbms_output.put_line('Product:'||(a*b));
END proc1;
PROCEDURE proc2(a int) IS
BEGIN
dbms_output.put_line('Square root of '||a||' is '||sqrt(a));
END proc2;
FUNCTION fn11(a int) return varchar2 is b varchar2(4);
BEGIN
IF(MOD(a,2)=0) THEN
RETURN 'even';
ELSE
RETURN 'odd';
END IF;
END;
FUNCTION fn22(a int,b int,c int)
return int is d int;
BEGIN
d := a+b+c;
return d;
END fn22;
END Pk1;
--step 3
DECLARE
p int;
q int;
r int;
s int;
result varchar2(4);
sum1 int;
```

```
BEGIN
p:=&p;
q:=&q;
r:=&r;
s:=&s;
Pk1.proc1(p,q);
Pk1.proc2(r);
RESULT:=Pk1.fn11(s);
sum1:=Pk1.fn22(p,q,s);
dbms_output.put_line(s ||' is '||result);
dbms_output.put_line('Sum of '||p||','||q||' and '||s||' is '||sum1);
END;
```

Output

```
PACKAGE PK1 compiled
PACKAGE BODY PK1 compiled
Sum:34
AVG:17
Product:225
Square root of 16 is 4
4 is even
Sum of 9,25 and 4 is 38
Sum:34
AVG:17
Product:225
Square root of 25 is 5
9 is odd
Sum of 4,16 and 9 is 29
```

Exp 16:

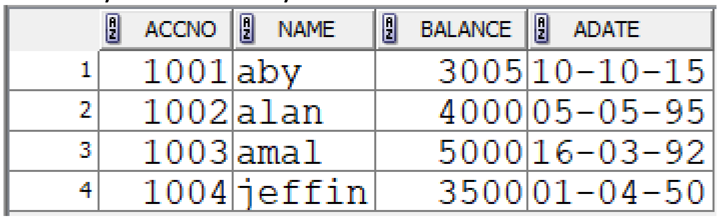
Queries

```
--Q1
create table bank_details(accno int,name varchar(30),balance int,adate date);
insert into bank_details values(1001,'aby',3005,'10-oct-15');
insert into bank_details values(1002,'alan',4000,'05-may-95');
insert into bank_details values(1003,'amal',5000,'16-mar-92');
insert into bank_details values(1004,'jeffin',3500,'01-apr-50');

select *from bank_details;
create table bank1(accno int,intesrest int);
declare
cursor temp is select accno,name,balance,adate from bank_details;
t temp%rowtype;
interest int;
begin
open temp;
loop
fetch temp into t;
interest:=.08*t.balance;
insert into bank1 values(t.accno,interest);
exit when temp%notfound;
end loop;
close temp;
end;
select * from bank1;

--Q2
Create table student (id int,name varchar(30),m1 int,m2 int,m3 int,grade varchar(10));
insert into student(id,name,m1,m2,m3) values(1,'allen',40,60,50);
insert into student(id,name,m1,m2,m3) values(2,'adi',47,54,34);
insert into student(id,name,m1,m2,m3) values(3,'binu',70,90,80);
insert into student(id,name,m1,m2,m3) values(4,'ciju',76,32,46);
insert into student(id,name,m1,m2,m3) values(5,'dinu',49,70,67);
declare
cursor temp is select id,m1,m2,m3 from student;
t temp%rowtype;
tot int;
grad varchar(10);
begin
open temp;
loop
fetch temp into t;
tot:=t.m1+t.m2+t.m3;
```

```
if((tot/3)>80)then
grad:='A';
elsif((tot/3)>70)then
grad:='B';
elsif((tot/3)>60)then
grad:='C';
else
grad:='D';
end if;
update student set grade=grad where id=t.id;
exit when temp%notfound;
end loop;
close temp;
end;
select * from student;

--Q3
Create table people_list(id int,name varchar(30),dt_joining date,place varchar(30));
insert into people_list values(100,'allen','10-04-2003','kerala');
insert into people_list values(102,'adi','11-3-2018','tamilnadu');
insert into people_list values(103,'binu','02-2-2017','japan');
insert into people_list values(104,'ciju','01-4-2001','america');
insert into people_list values(105,'dinu','06-5-2016','paris');

select * from people_list;
create table exp_list(id int,name varchar(30),ex int);
declare
cursor temp is select id,name,dt_joining from people_list;
t temp%rowtype;
ex int;
begin
open temp;
loop
fetch temp into t;
ex:=months_between(sysdate,t.dt_joining)/12;
if ex>10 then
insert into exp_list values(t.id,t.name,ex);
end if;
exit when temp%notfound;
end loop;
end;
select * from exp_list;

--Q4
Create table employee_list(id int,name varchar(30),monthly_salary int);
insert into employee_list values(1,'allen',3000);
insert into employee_list values(2,'adi',6000);
insert into employee_list values(3,'binu',20000);
insert into employee_list values(4,'ciju',50000);
insert into employee_list values(5,'dinu',35000);
select * from employee_list;

declare
cursor temp is select id,name,monthly_salary from employee_list;
t temp%rowtype;
a int;
m int;
begin
open temp;
loop
fetch temp into t;
a:=t.monthly_salary*12;
if(a<60000)then
m:=t.monthly_salary+(t.monthly_salary*0.25);
elsif(a<200000)then
m:=t.monthly_salary+(t.monthly_salary*0.20);
elsif(a<500000)then
m:=t.monthly_salary+(t.monthly_salary*0.15);
else
m:=t.monthly_salary+(t.monthly_salary*0.10);
end if;
update employee_list set monthly_salary=m where id=t.id;
exit when temp%notfound;
end loop;
close temp;
end;
select * from employee_list;
```

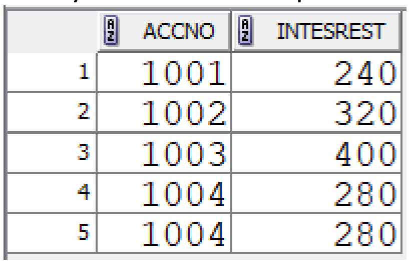
Output

```
table BANK_DETAILS created.
4 rows inserted.
>>Query Run In:Query Result 1
```



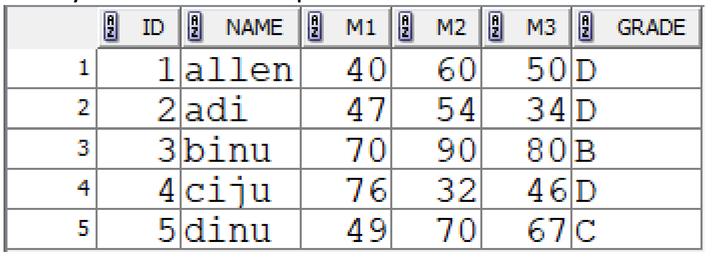
	ACCNO	NAME	BALANCE	ADATE
1	1001	aby	3005	10-10-15
2	1002	alan	4000	05-05-95
3	1003	amal	5000	16-03-92
4	1004	jeffin	3500	01-04-50

table BANK1 created.
anonymous block completed



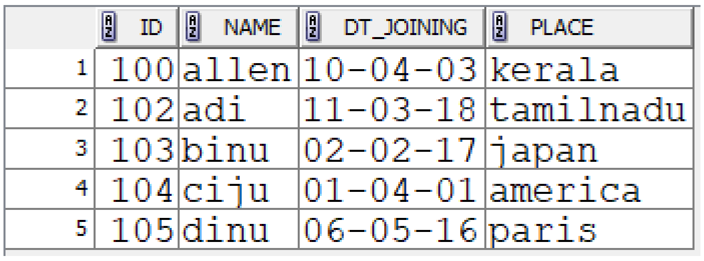
	ACCNO	INTESREST
1	1001	240
2	1002	320
3	1003	400
4	1004	280
5	1004	280

table STUDENT created.
5 rows inserted.
anonymous block completed



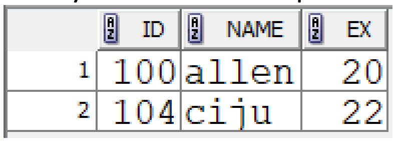
	ID	NAME	M1	M2	M3	GRADE
1	1	allen	40	60	50	D
2	2	adi	47	54	34	D
3	3	binu	70	90	80	B
4	4	ciju	76	32	46	D
5	5	dinu	49	70	67	C

table PEOPLE_LIST created.
5 rows inserted.



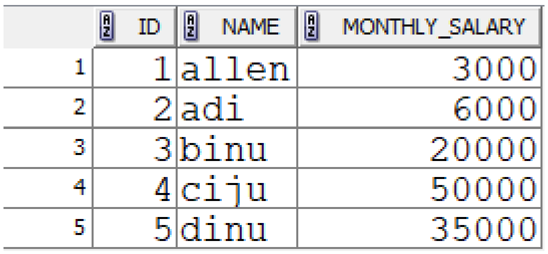
	ID	NAME	DT_JOINING	PLACE
1	100	allen	10-04-03	kerala
2	102	adi	11-03-18	tamilnadu
3	103	binu	02-02-17	japan
4	104	ciju	01-04-01	america
5	105	dinu	06-05-16	paris

table EXP_LIST created.
anonymous block completed



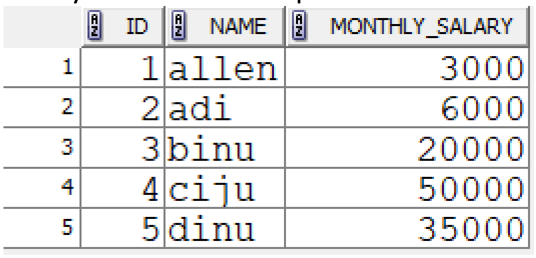
	ID	NAME	EX
1	100	allen	20
2	104	ciju	22

table EMPLOYEE_LIST created.
5 rows inserted.



	ID	NAME	MONTHLY_SALARY
1	1	allen	3000
2	2	adi	6000
3	3	binu	20000
4	4	ciju	50000
5	5	dinu	35000

anonymous block completed



	ID	NAME	MONTHLY_SALARY
1	1	allen	3000
2	2	adi	6000
3	3	binu	20000
4	4	ciju	50000
5	5	dinu	35000