TECHNO
INTERNATIONAL
NEW TOWN

ESTD. 2005

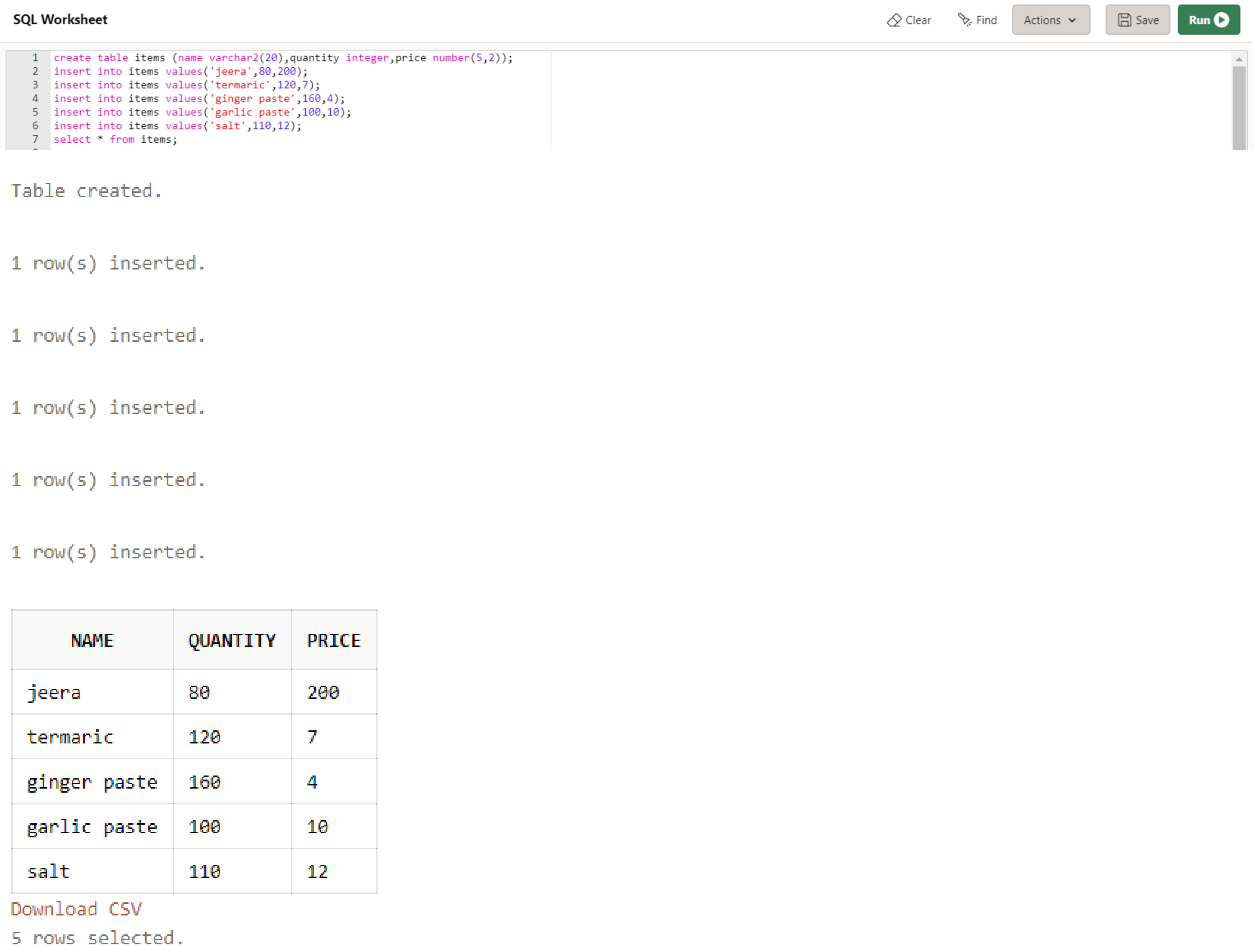
Name : Kartik Paul

Roll : 18701019056

Subject : Advance Database lab

Stream : MCA 4th

1. Suppose there is a table which stores the relevant details of some items. Now from these items, the prices of all the items whose quantity is more than 100 are to be increased by 20%. Write a PL/SQL code to perform this change and also create a separate table which will store the details of these items with changed price values.



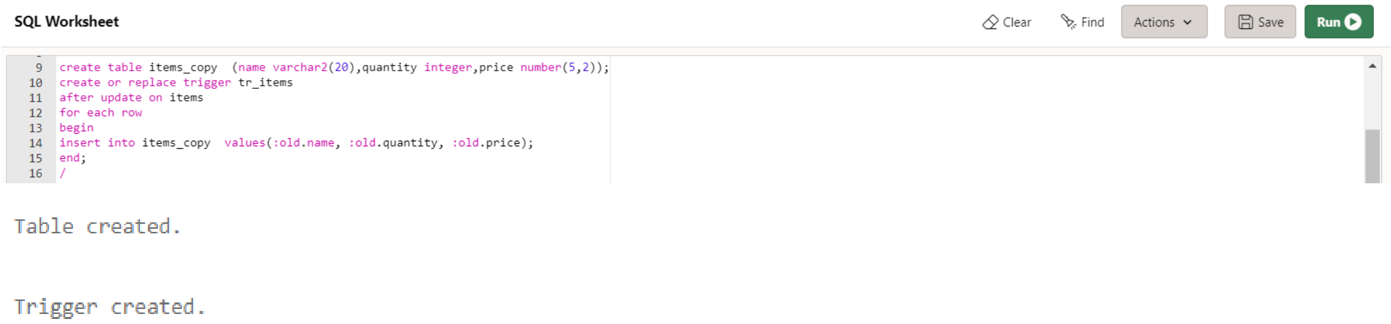
```
create table items (name varchar2(20),quantity integer,price number(5,2));
insert into items values('jeera',80,200);
insert into items values('termaric',120,7);
insert into items values('ginger paste',160,4);
insert into items values('garlic paste',100,10);
insert into items values('salt',110,12);
select * from items;
```

```
Table created.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

NAME	QUANTITY	PRICE
jeera	80	200
termaric	120	7
ginger paste	160	4
garlic paste	100	10
salt	110	12

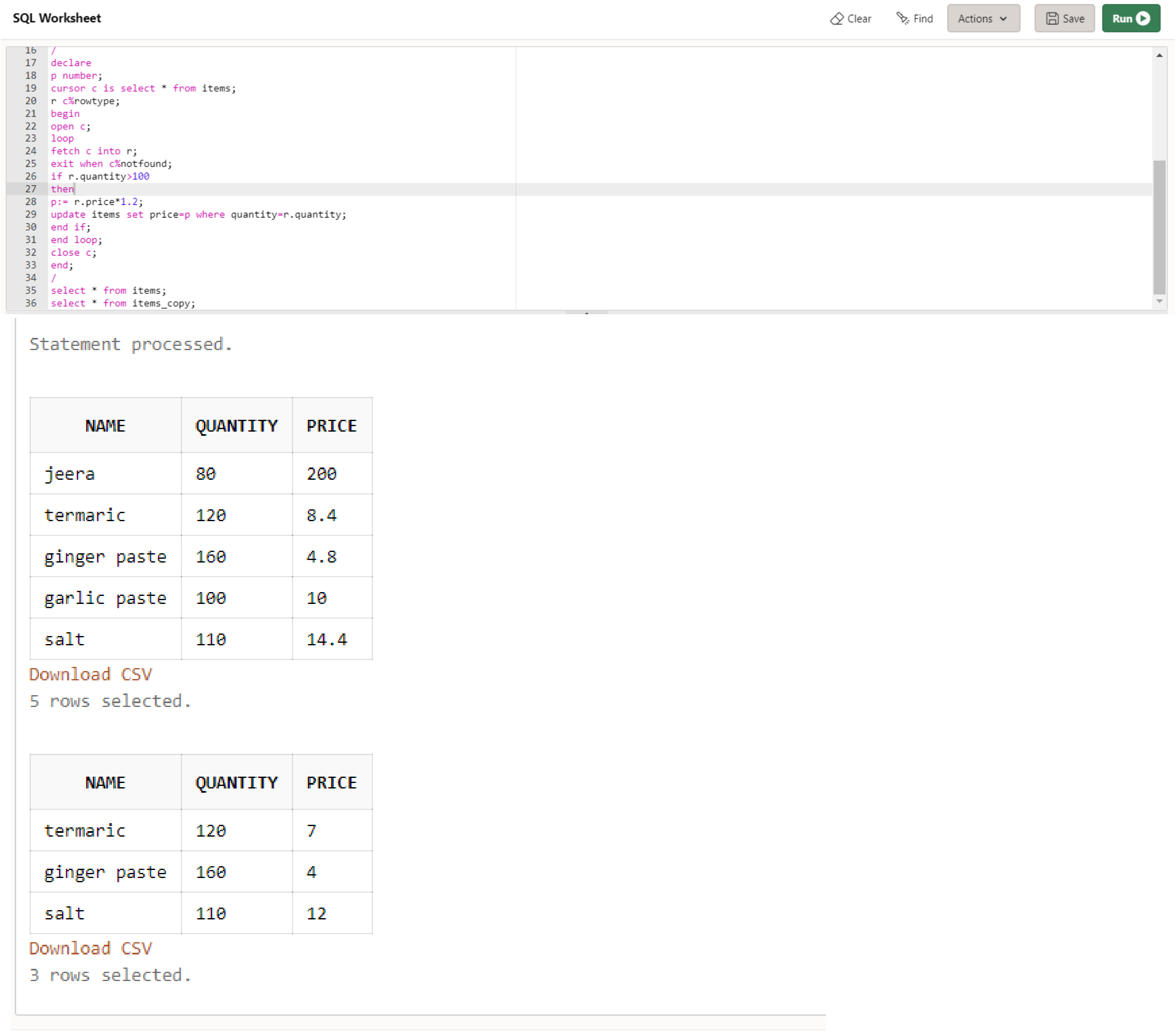
Download CSV

```
5 rows selected.
```



```
create table items_copy  (name varchar2(20),quantity integer,price number(5,2));
create or replace trigger tr_items
after update on items
for each row
begin
insert into items_copy  values(:old.name, :old.quantity, :old.price);
end;
/
```

```
Table created.

Trigger created.
```



```
/
declare
p number;
cursor c is select * from items;
r c%rowtype;
begin
open c;
loop
fetch c into r;
exit when c%notfound;
if r.quantity>100
then
p:= r.price*1.2;
update items set price=p where quantity=r.quantity;
end if;
end loop;
close c;
end;
/
select * from items;
select * from items_copy;
```

Statement processed.

NAME	QUANTITY	PRICE
jeera	80	200
termaric	120	8.4
ginger paste	160	4.8
garlic paste	100	10
salt	110	14.4

Download CSV

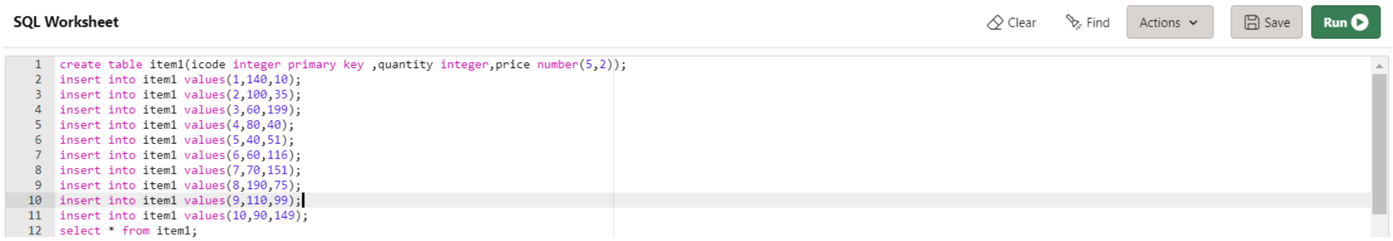
5 rows selected.

NAME	QUANTITY	PRICE
termaric	120	7
ginger paste	160	4
salt	110	12

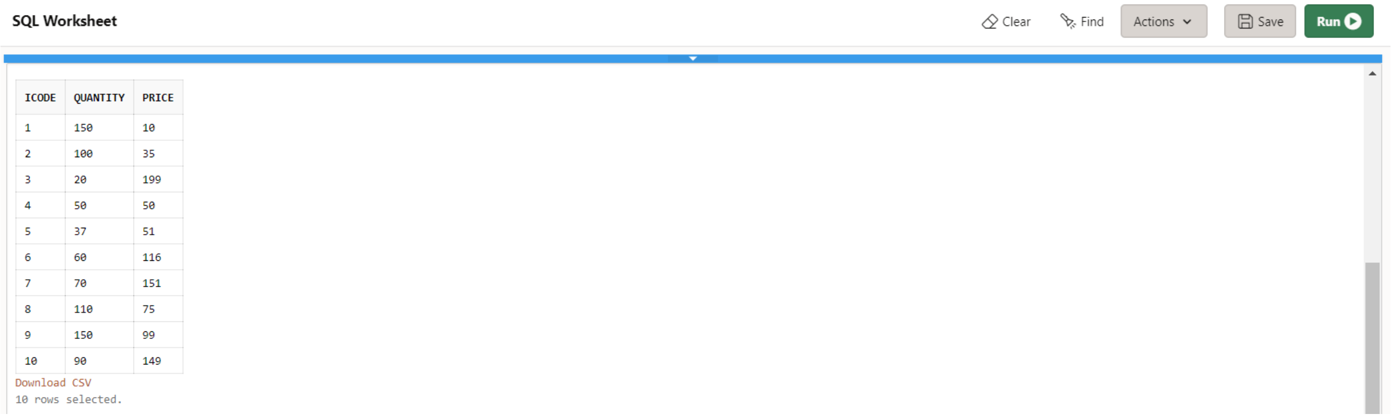
Download CSV

3 rows selected.

2. Suppose there is a table which stores the relevant details of some items. Table entries are ordered by “item code”. Now write a PL/SQL code which will display the details of the first 5 most expensive items.

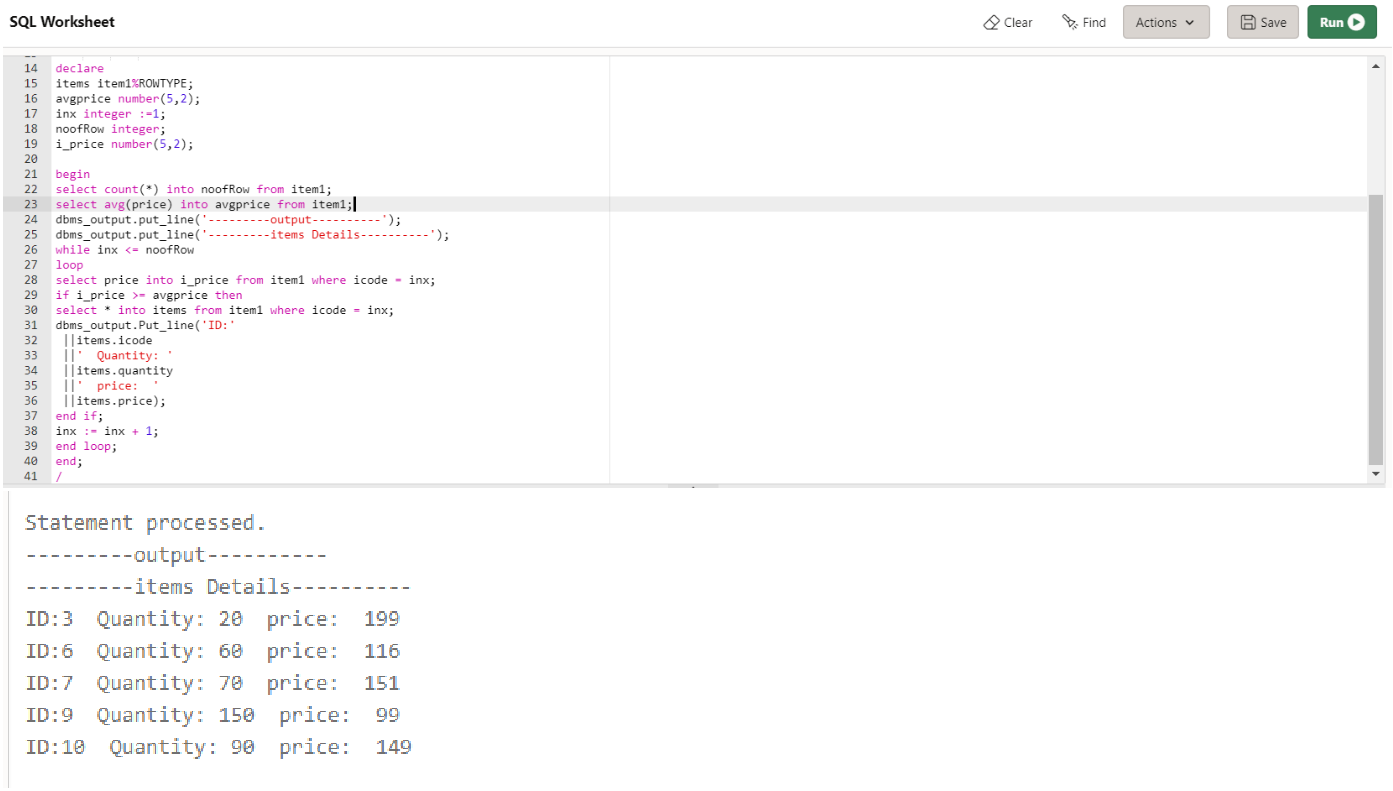


```sql
create table item1(icode integer primary key ,quantity integer,price number(5,2));
insert into item1 values(1,140,10);
insert into item1 values(2,100,35);
insert into item1 values(3,60,199);
insert into item1 values(4,80,40);
insert into item1 values(5,40,51);
insert into item1 values(6,60,116);
insert into item1 values(7,70,151);
insert into item1 values(8,190,75);
insert into item1 values(9,110,99);
insert into item1 values(10,90,149);
select * from item1;
```



ICODE	QUANTITY	PRICE
1	150	10
2	100	35
3	20	199
4	50	50
5	37	51
6	60	116
7	70	151
8	110	75
9	150	99
10	90	149

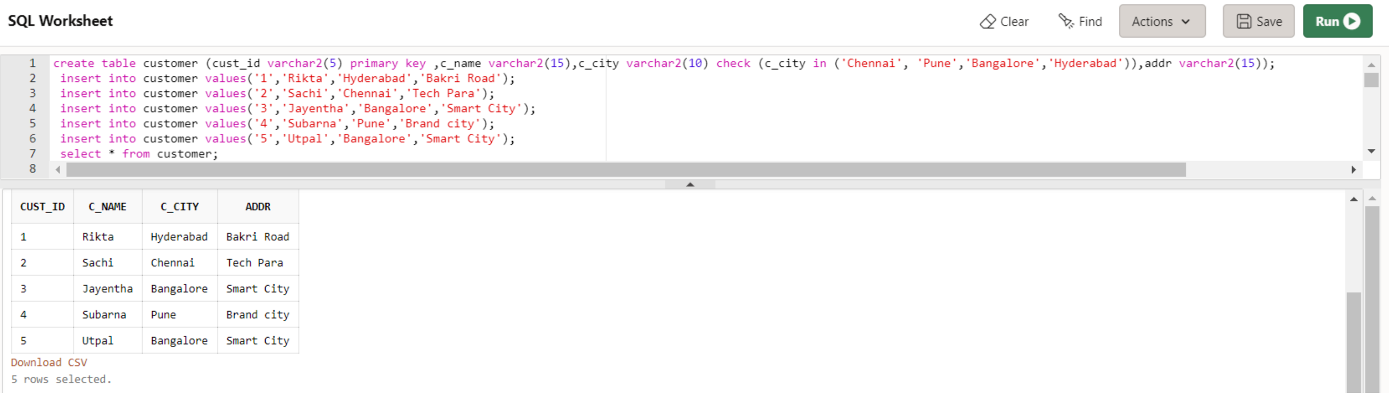
10 rows selected.



```sql
declare
items item1%ROWTYPE;
avgprice number(5,2);
inx integer :=1;
noofRow integer;
i_price number(5,2);

begin
select count(*) into noofRow from item1;
select avg(price) into avgprice from item1;
dbms_output.put_line('---------output----------');
dbms_output.put_line('---------items Details----------');
while inx <= noofRow
loop
select price into i_price from item1 where icode = inx;
if i_price >= avgprice then
select * into items from item1 where icode = inx;
dbms_output.Put_line('ID:'
 ||items.icode
 ||'  Quantity: '
 ||items.quantity
 ||'  price:  '
 ||items.price);
end if;
inx := inx + 1;
end loop;
end;
/
```

```
Statement processed.
---------output----------
---------items Details----------
ID:3  Quantity: 20  price:  199
ID:6  Quantity: 60  price:  116
ID:7  Quantity: 70  price:  151
ID:9  Quantity: 150  price:  99
ID:10  Quantity: 90  price:  149
```

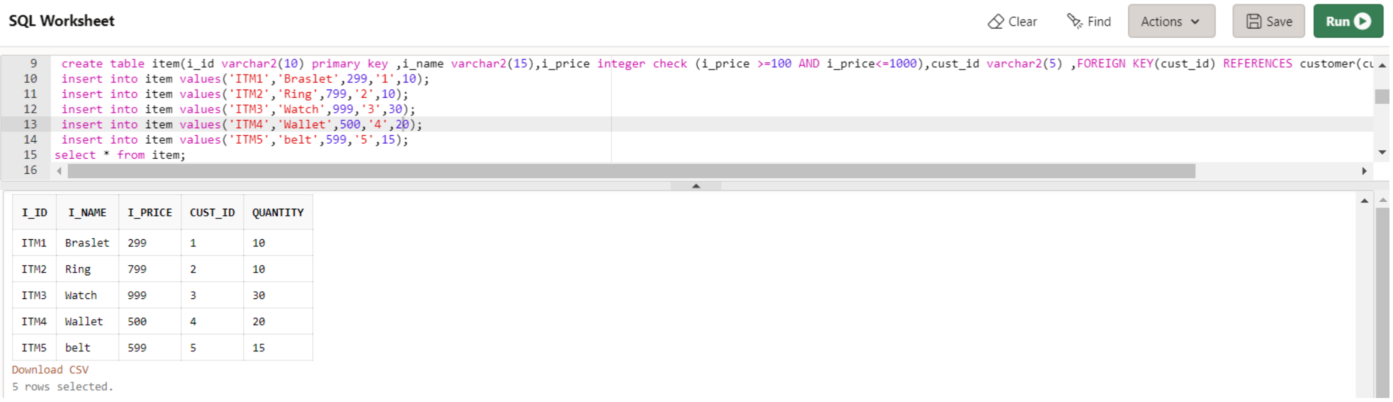
3. Every item id must begin with ‘ITM’, the price range of all the items is from Rs.100/- to Rs.1000/-. The customer’s city must be either from one of the following: Chennai, Pune, Bangalore and Hyderabad. Write a PL/SQL code using Cursor to input the id of an item and to check whether the number of this item present is less than 50. If so, add 30% more items, otherwise add 15% more items.



```sql
create table customer (cust_id varchar2(5) primary key ,c_name varchar2(15),c_city varchar2(10) check (c_city in ('Chennai', 'Pune','Bangalore','Hyderabad')),addr varchar2(15));
 insert into customer values('1','Rikta','Hyderabad','Bakri Road');
 insert into customer values('2','Sachi','Chennai','Tech Para');
 insert into customer values('3','Jayentha','Bangalore','Smart City');
 insert into customer values('4','Subarna','Pune','Brand city');
 insert into customer values('5','Utpal','Bangalore','Smart City');
 select * from customer;
```

CUST_ID	C_NAME	C_CITY	ADDR
1	Rikta	Hyderabad	Bakri Road
2	Sachi	Chennai	Tech Para
3	Jayentha	Bangalore	Smart City
4	Subarna	Pune	Brand city
5	Utpal	Bangalore	Smart City

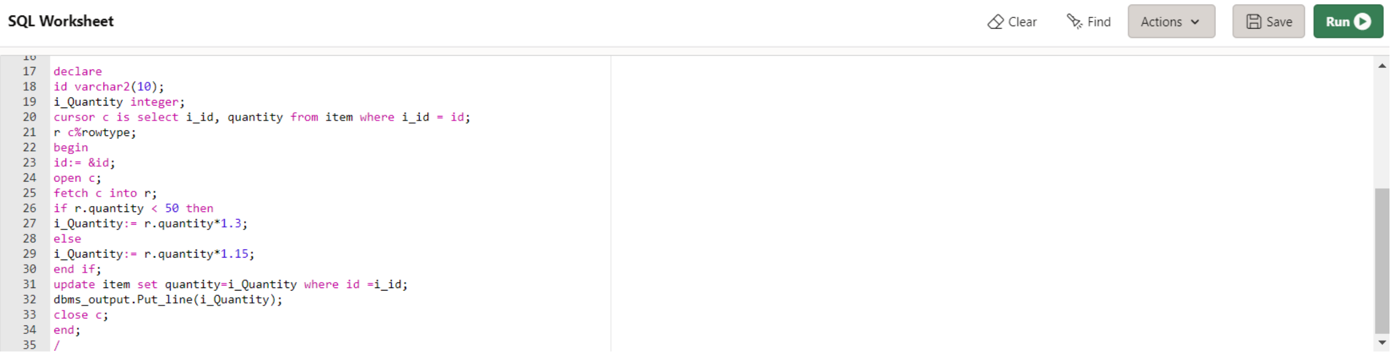
Download CSV
5 rows selected.



```sql
 create table item(i_id varchar2(10) primary key ,i_name varchar2(15),i_price integer check (i_price >=100 AND i_price<=1000),cust_id varchar2(5) ,FOREIGN KEY(cust_id) REFERENCES customer(cu
 insert into item values('ITM1','Braslet',299,'1',10);
 insert into item values('ITM2','Ring',799,'2',10);
 insert into item values('ITM3','Watch',999,'3',30);
 insert into item values('ITM4','Wallet',500,'4',20);
 insert into item values('ITM5','belt',599,'5',15);
select * from item;
```

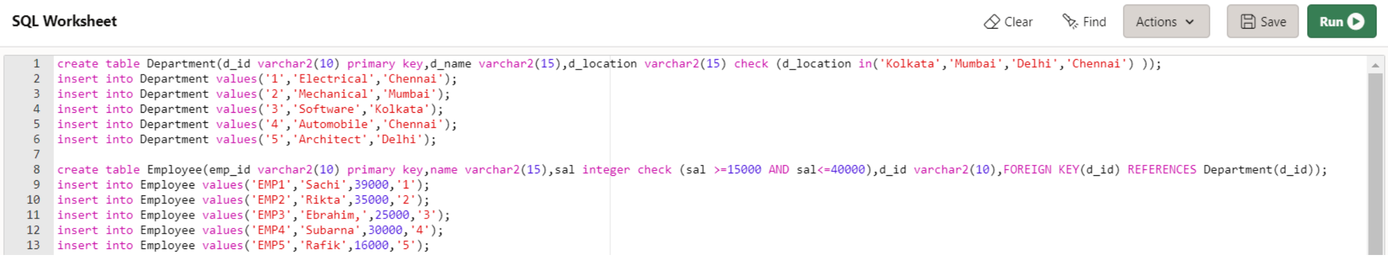
I_ID	I_NAME	I_PRICE	CUST_ID	QUANTITY
ITM1	Braslet	299	1	10
ITM2	Ring	799	2	10
ITM3	Watch	999	3	30
ITM4	Wallet	500	4	20
ITM5	belt	599	5	15

Download CSV
5 rows selected.

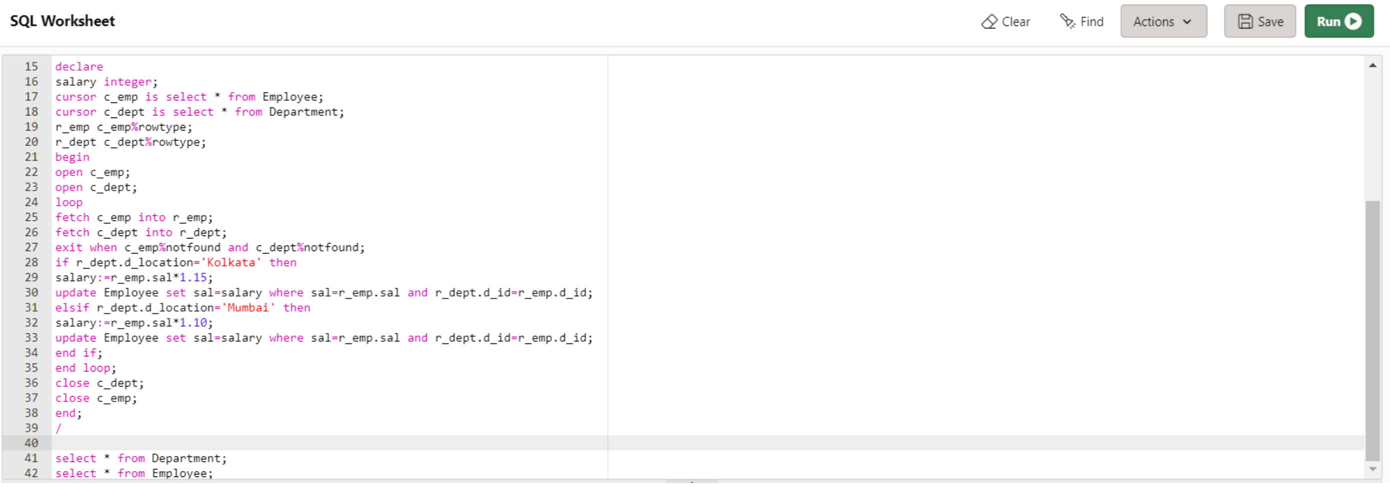


```sql
declare
id varchar2(10);
i_Quantity integer;
cursor c is select i_id, quantity from item where i_id = id;
r c%rowtype;
begin
id:= &id;
open c;
fetch c into r;
if r.quantity < 50 then
i_Quantity:= r.quantity*1.3;
else
i_Quantity:= r.quantity*1.15;
end if;
update item set quantity=i_Quantity where id =i_id;
dbms_output.Put_line(i_Quantity);
close c;
end;
/
```

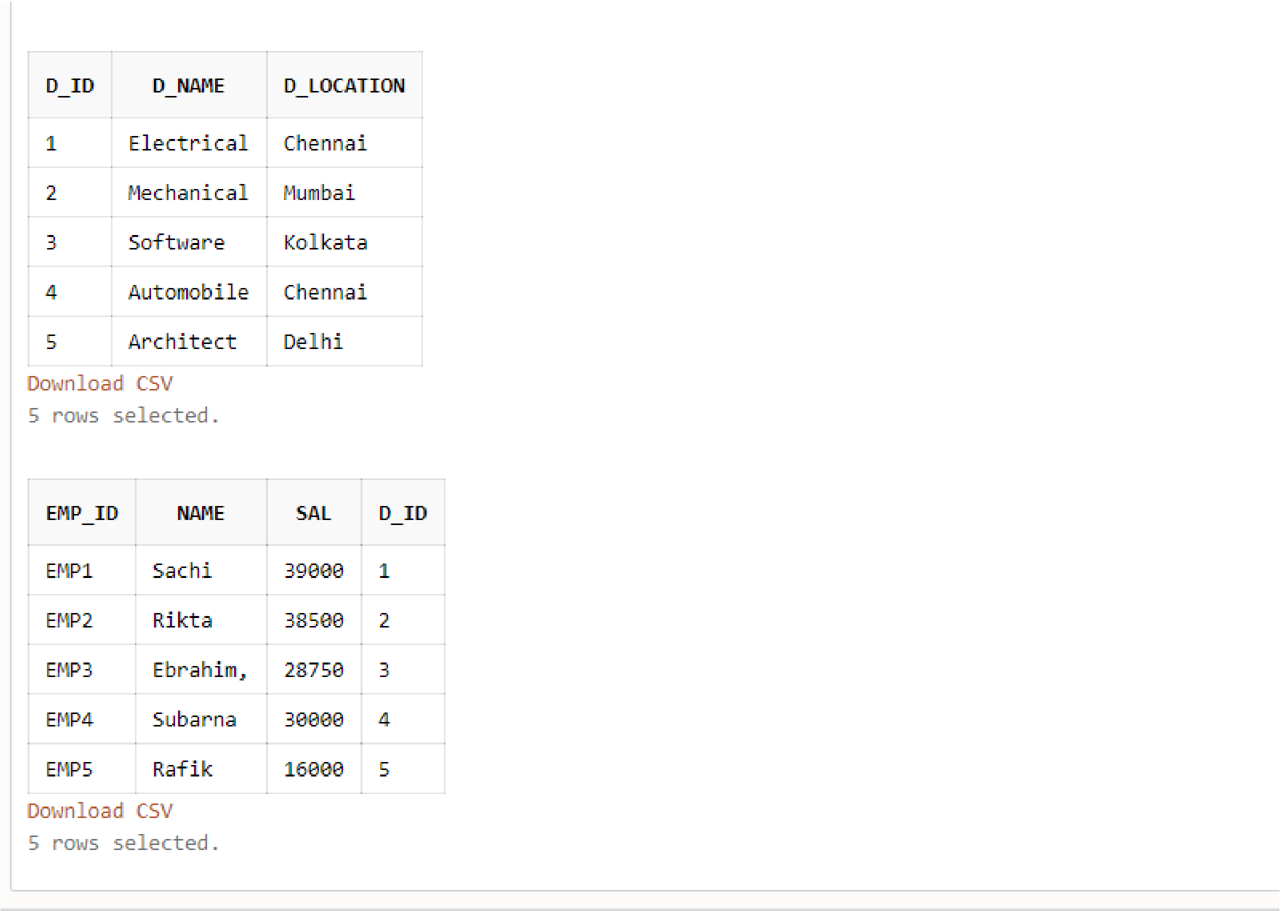
4. Create the following tables with proper integrity constraints: Employee (emp_id, name, sal, d_id) Department (d_id, d_name, d_location) Every employee id must begin with „EMP‟, the salary range of an employee should be between 15000 and 40000, and the departments are located in one of the following locations: Kolkata, Mumbai, Delhi and Chennai. Write a PL/SQL code using Cursor to increase the salary of all the employees of Kolkata by 15% and Mumbai by 10%.



```sql
create table Department(d_id varchar2(10) primary key,d_name varchar2(15),d_location varchar2(15) check (d_location in('Kolkata','Mumbai','Delhi','Chennai') ));
insert into Department values('1','Electrical','Chennai');
insert into Department values('2','Mechanical','Mumbai');
insert into Department values('3','Software','Kolkata');
insert into Department values('4','Automobile','Chennai');
insert into Department values('5','Architect','Delhi');

create table Employee(emp_id varchar2(10) primary key,name varchar2(15),sal integer check (sal >=15000 AND sal<=40000),d_id varchar2(10),FOREIGN KEY(d_id) REFERENCES Department(d_id));
insert into Employee values('EMP1','Sachi',39000,'1');
insert into Employee values('EMP2','Rikta',35000,'2');
insert into Employee values('EMP3','Ebrahim,',25000,'3');
insert into Employee values('EMP4','Subarna',30000,'4');
insert into Employee values('EMP5','Rafik',16000,'5');
```



```sql
declare
salary integer;
cursor c_emp is select * from Employee;
cursor c_dept is select * from Department;
r_emp c_emp%rowtype;
r_dept c_dept%rowtype;
begin
open c_emp;
open c_dept;
loop
fetch c_emp into r_emp;
fetch c_dept into r_dept;
exit when c_emp%notfound and c_dept%notfound;
if r_dept.d_location='Kolkata' then
salary:=r_emp.sal*1.15;
update Employee set sal=salary where sal=r_emp.sal and r_dept.d_id=r_emp.d_id;
elsif r_dept.d_location='Mumbai' then
salary:=r_emp.sal*1.10;
update Employee set sal=salary where sal=r_emp.sal and r_dept.d_id=r_emp.d_id;
end if;
end loop;
close c_dept;
close c_emp;
end;
/

select * from Department;
select * from Employee;
```



D_ID	D_NAME	D_LOCATION
1	Electrical	Chennai
2	Mechanical	Mumbai
3	Software	Kolkata
4	Automobile	Chennai
5	Architect	Delhi

Download CSV

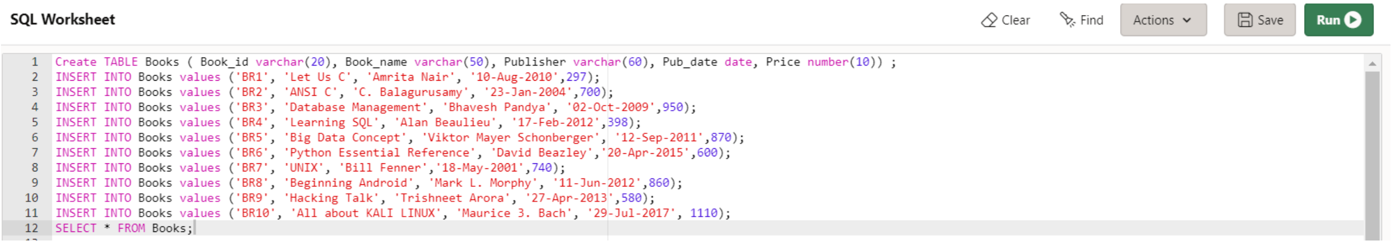
5 rows selected.

EMP_ID	NAME	SAL	D_ID
EMP1	Sachi	39000	1
EMP2	Rikta	38500	2
EMP3	Ebrahim,	28750	3
EMP4	Subarna	30000	4
EMP5	Rafik	16000	5

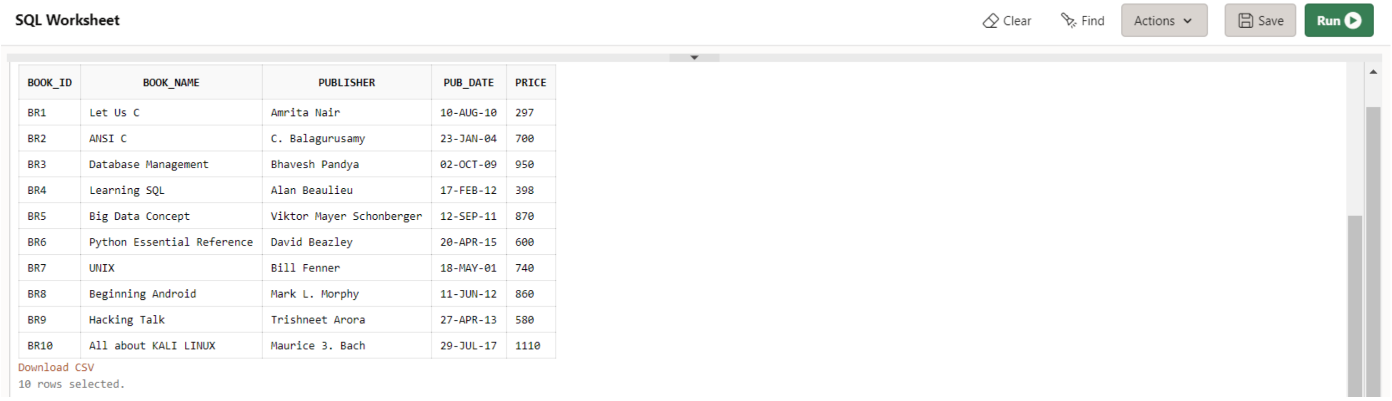
Download CSV

5 rows selected.

5. Create a product table, named Books having the following fields:(Book_id, Book_Name, Publisher, Pub_date, Price)Insert at least 10 records.Write a PL/SQL code to fetch the first 5 books according to their priceand store their details in a separate table called Costly_books.



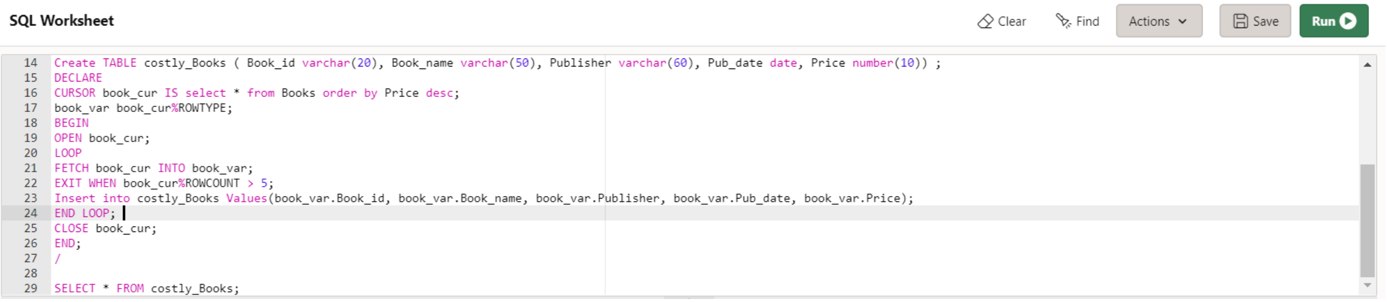
```
Create TABLE Books ( Book_id varchar(20), Book_name varchar(50), Publisher varchar(60), Pub_date date, Price number(10)) ;
INSERT INTO Books values ('BR1', 'Let Us C', 'Amrita Nair', '10-Aug-2010',297);
INSERT INTO Books values ('BR2', 'ANSI C', 'C. Balagurusamy', '23-Jan-2004',700);
INSERT INTO Books values ('BR3', 'Database Management', 'Bhavesh Pandya', '02-Oct-2009',950);
INSERT INTO Books values ('BR4', 'Learning SQL', 'Alan Beaulieu', '17-Feb-2012',398);
INSERT INTO Books values ('BR5', 'Big Data Concept', 'Viktor Mayer Schonberger', '12-Sep-2011',870);
INSERT INTO Books values ('BR6', 'Python Essential Reference', 'David Beazley','20-Apr-2015',600);
INSERT INTO Books values ('BR7', 'UNIX', 'Bill Fenner','18-May-2001',740);
INSERT INTO Books values ('BR8', 'Beginning Android', 'Mark L. Morphy', '11-Jun-2012',860);
INSERT INTO Books values ('BR9', 'Hacking Talk', 'Trishneet Arora', '27-Apr-2013',580);
INSERT INTO Books values ('BR10', 'All about KALI LINUX', 'Maurice 3. Bach', '29-Jul-2017', 1110);
SELECT * FROM Books;
```



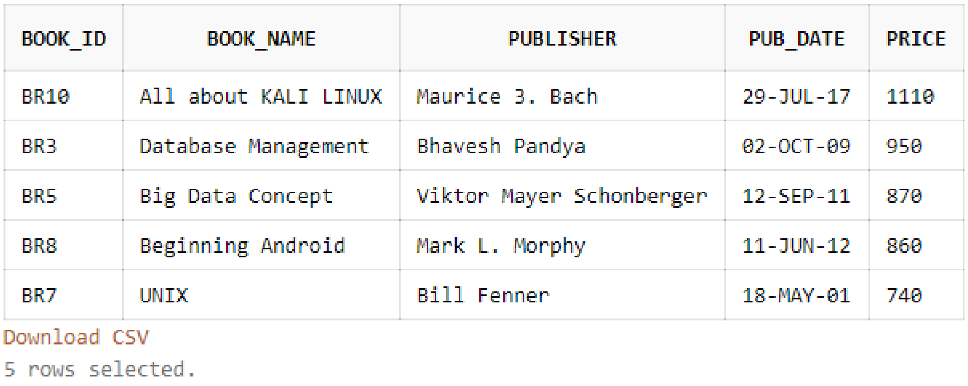
BOOK_ID	BOOK_NAME	PUBLISHER	PUB_DATE	PRICE
BR1	Let Us C	Amrita Nair	10-AUG-10	297
BR2	ANSI C	C. Balagurusamy	23-JAN-04	700
BR3	Database Management	Bhavesh Pandya	02-OCT-09	950
BR4	Learning SQL	Alan Beaulieu	17-FEB-12	398
BR5	Big Data Concept	Viktor Mayer Schonberger	12-SEP-11	870
BR6	Python Essential Reference	David Beazley	20-APR-15	600
BR7	UNIX	Bill Fenner	18-MAY-01	740
BR8	Beginning Android	Mark L. Morphy	11-JUN-12	860
BR9	Hacking Talk	Trishneet Arora	27-APR-13	580
BR10	All about KALI LINUX	Maurice 3. Bach	29-JUL-17	1110

Download CSV

10 rows selected.



```
Create TABLE costly_Books ( Book_id varchar(20), Book_name varchar(50), Publisher varchar(60), Pub_date date, Price number(10)) ;
DECLARE
CURSOR book_cur IS select * from Books order by Price desc;
book_var book_cur%ROWTYPE;
BEGIN
OPEN book_cur;
LOOP
FETCH book_cur INTO book_var;
EXIT WHEN book_cur%ROWCOUNT > 5;
Insert into costly_Books Values(book_var.Book_id, book_var.Book_name, book_var.Publisher, book_var.Pub_date, book_var.Price);
END LOOP;
CLOSE book_cur;
END;
/

SELECT * FROM costly_Books;
```

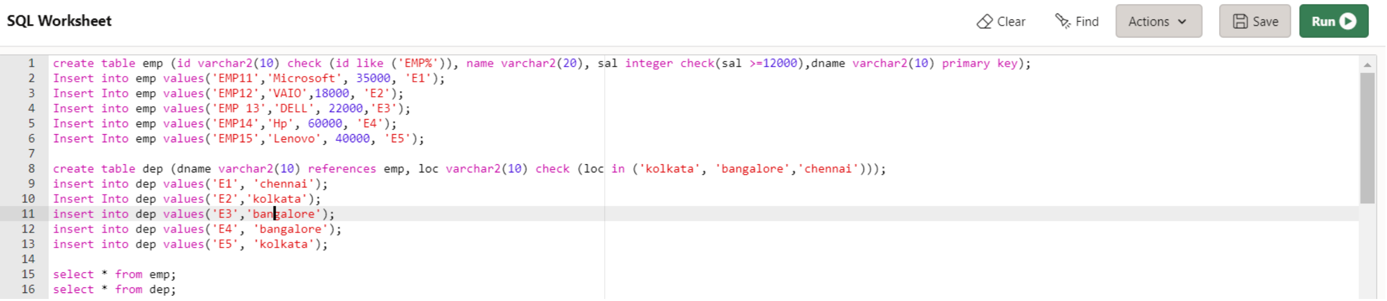


BOOK_ID	BOOK_NAME	PUBLISHER	PUB_DATE	PRICE
BR10	All about KALI LINUX	Maurice J. Bach	29-JUL-17	1110
BR3	Database Management	Bhavesh Pandya	02-OCT-09	950
BR5	Big Data Concept	Viktor Mayer Schonberger	12-SEP-11	870
BR8	Beginning Android	Mark L. Morphy	11-JUN-12	860
BR7	UNIX	Bill Fenner	18-MAY-01	740

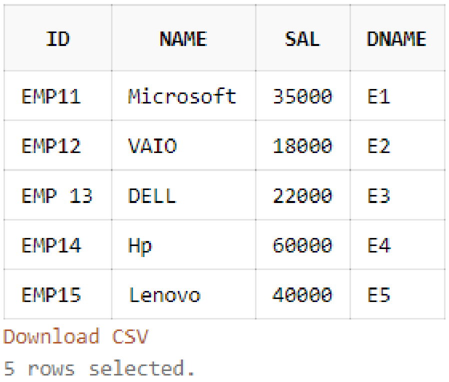
Download CSV

5 rows selected.

6. Create the following tables with proper integrity constraints: Employee (id, name, sal, dname) Department (dname,loc) Every employee id must begin with ‘EMP’, the minimum salary of an employee should be 12000 and all departments are located in one of the following locations: Kolkata, Bangalore and Chennai. Write a PL/SQL code using cursor to increase the salary of all the employees from Kolkata by 10% and decrease the salary of all the employees from Bangalore by 5%. Ensure that the updation is properly



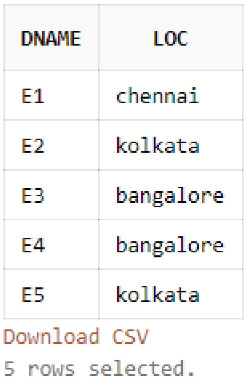
```
create table emp (id varchar2(10) check (id like ('EMP%')), name varchar2(20), sal integer check(sal >=12000),dname varchar2(10) primary key);
Insert into emp values('EMP11','Microsoft', 35000, 'E1');
Insert Into emp values('EMP12','VAIO',18000, 'E2');
Insert into emp values('EMP 13','DELL', 22000,'E3');
Insert into emp values('EMP14','Hp', 60000, 'E4');
Insert Into emp values('EMP15','Lenovo', 40000, 'E5');

create table dep (dname varchar2(10) references emp, loc varchar2(10) check (loc in ('kolkata', 'bangalore','chennai')));
insert into dep values('E1', 'chennai');
Insert Into dep values('E2','kolkata');
insert into dep values('E3','bangalore');
insert into dep values('E4', 'bangalore');
insert into dep values('E5', 'kolkata');

select * from emp;
select * from dep;
```



ID	NAME	SAL	DNAME
EMP11	Microsoft	35000	E1
EMP12	VAIO	18000	E2
EMP 13	DELL	22000	E3
EMP14	Hp	60000	E4
EMP15	Lenovo	40000	E5

Download CSV

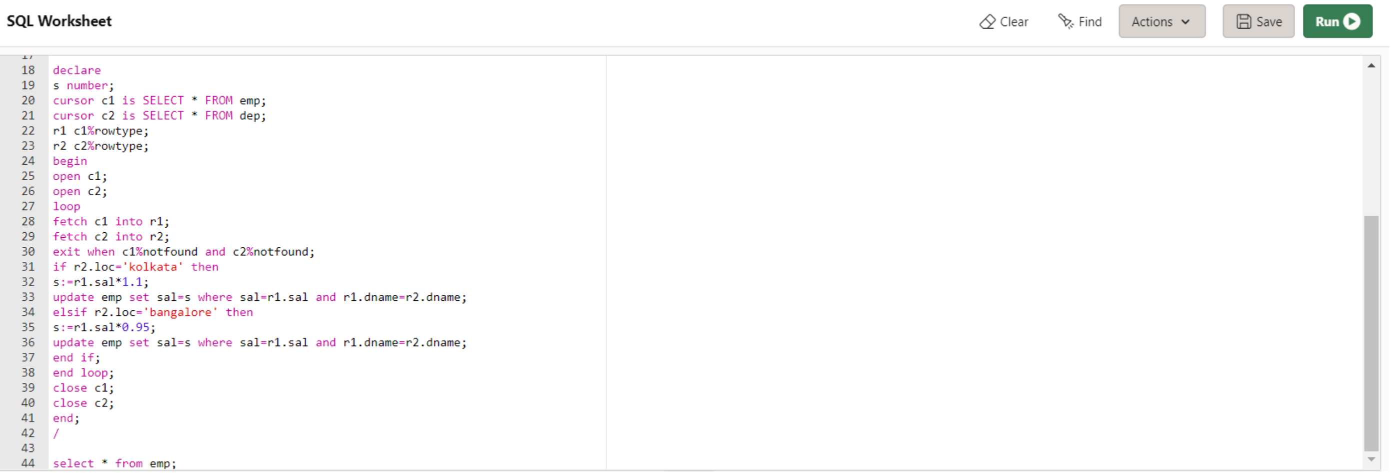
5 rows selected.



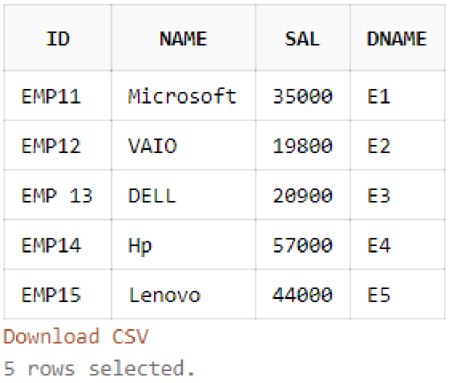
DNAME	LOC
E1	chennai
E2	kolkata
E3	bangalore
E4	bangalore
E5	kolkata

Download CSV

5 rows selected.



```
declare
s number;
cursor c1 is SELECT * FROM emp;
cursor c2 is SELECT * FROM dep;
r1 c1%rowtype;
r2 c2%rowtype;
begin
open c1;
open c2;
loop
fetch c1 into r1;
fetch c2 into r2;
exit when c1%notfound and c2%notfound;
if r2.loc='kolkata' then
s:=r1.sal*1.1;
update emp set sal=s where sal=r1.sal and r1.dname=r2.dname;
elsif r2.loc='bangalore' then
s:=r1.sal*0.95;
update emp set sal=s where sal=r1.sal and r1.dname=r2.dname;
end if;
end loop;
close c1;
close c2;
end;
/

select * from emp;
```

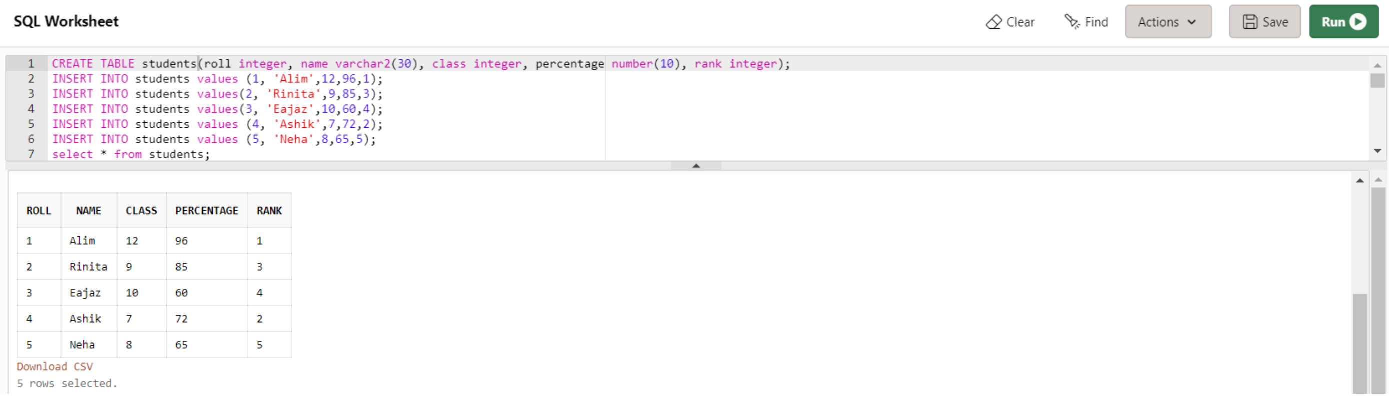


ID	NAME	SAL	DNAME
EMP11	Microsoft	35000	E1
EMP12	VAIO	19800	E2
EMP 13	DELL	20900	E3
EMP14	Hp	57000	E4
EMP15	Lenovo	44000	E5

Download CSV

5 rows selected.

7. Create a table named Students to store the detailed information of the students of a school. The table should contain the following fields: (Sroll, Sname, Class, Percentage, Rank) Write a PL/SQL code to separately store the records having even numbered ranks and odd-numbered ranks to two different tables.

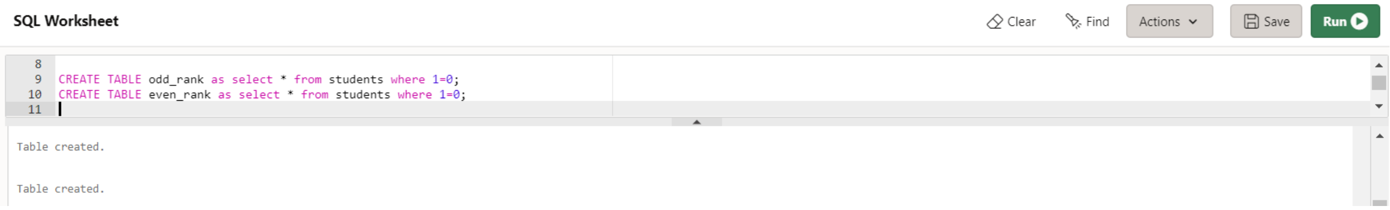


```
CREATE TABLE students(roll integer, name varchar2(30), class integer, percentage number(10), rank integer);
INSERT INTO students values (1, 'Alim',12,96,1);
INSERT INTO students values(2, 'Rinita',9,85,3);
INSERT INTO students values(3, 'Eajaz',10,60,4);
INSERT INTO students values (4, 'Ashik',7,72,2);
INSERT INTO students values (5, 'Neha',8,65,5);
select * from students;
```

ROLL	NAME	CLASS	PERCENTAGE	RANK
1	Alim	12	96	1
2	Rinita	9	85	3
3	Eajaz	10	60	4
4	Ashik	7	72	2
5	Neha	8	65	5

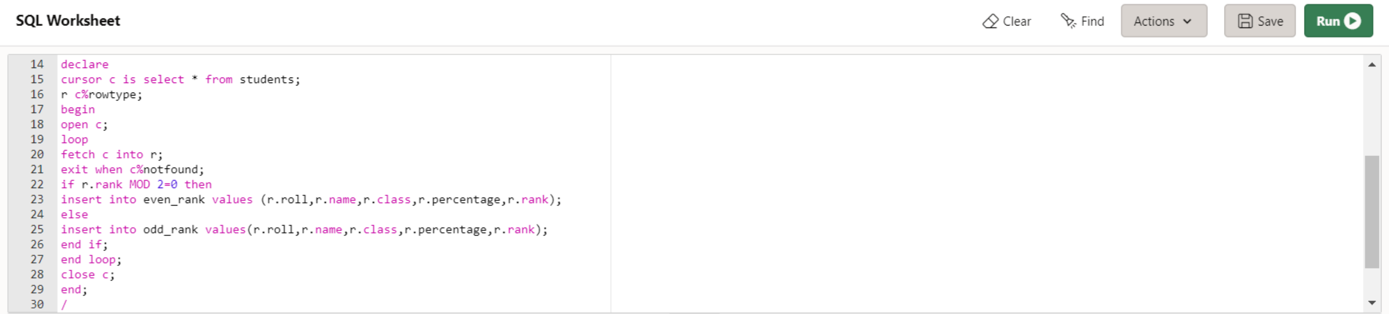
Download CSV

5 rows selected.

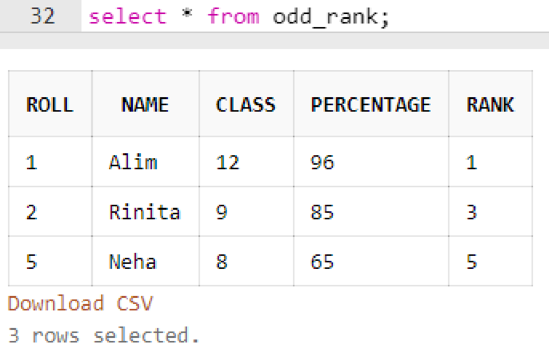


```
CREATE TABLE odd_rank as select * from students where 1=0;
CREATE TABLE even_rank as select * from students where 1=0;
```

```
Table created.

Table created.
```



```
declare
cursor c is select * from students;
r c%rowtype;
begin
open c;
loop
fetch c into r;
exit when c%notfound;
if r.rank MOD 2=0 then
insert into even_rank values (r.roll,r.name,r.class,r.percentage,r.rank);
else
insert into odd_rank values(r.roll,r.name,r.class,r.percentage,r.rank);
end if;
end loop;
close c;
end;
/
```

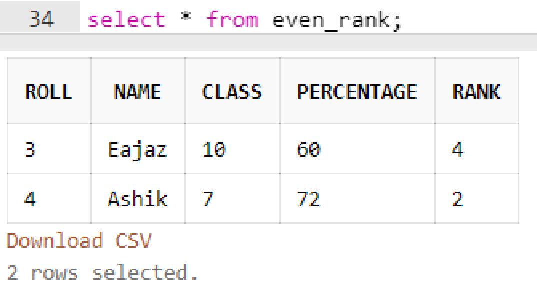


```
select * from odd_rank;
```

ROLL	NAME	CLASS	PERCENTAGE	RANK
1	Alim	12	96	1
2	Rinita	9	85	3
5	Neha	8	65	5

Download CSV

3 rows selected.



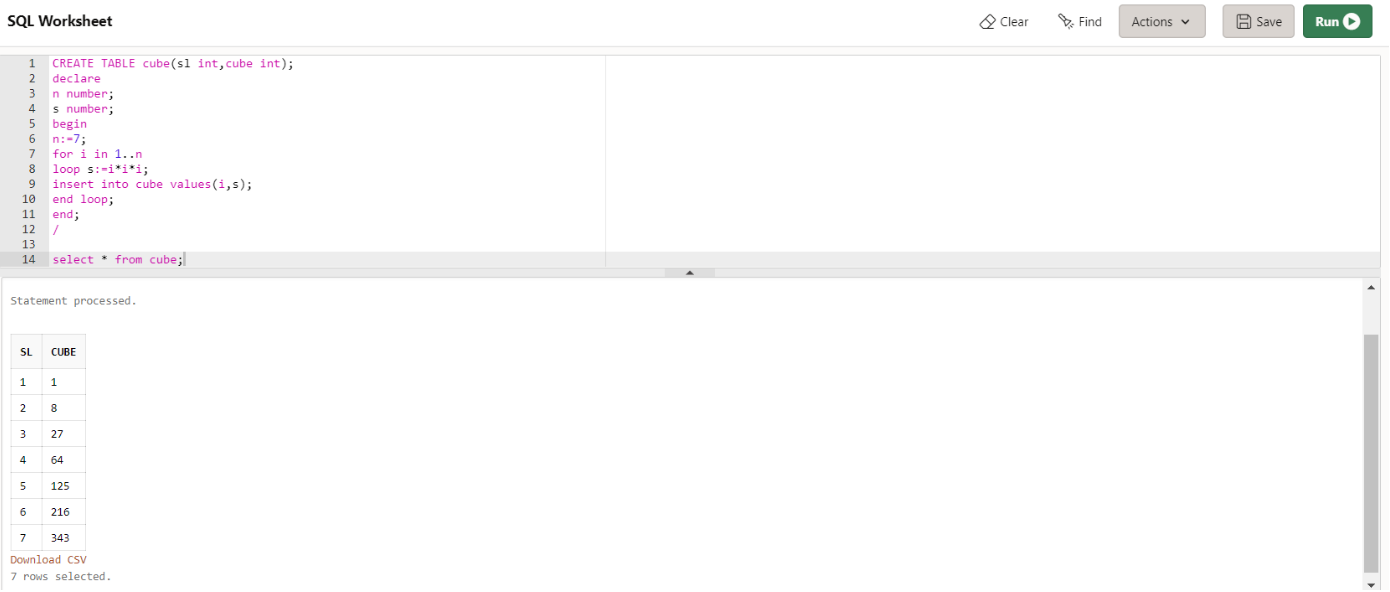
```
select * from even_rank;
```

ROLL	NAME	CLASS	PERCENTAGE	RANK
3	Eajaz	10	60	4
4	Ashik	7	72	2

Download CSV

2 rows selected.

8. (a) Write a PL/SQL code to store the first n positive integers along with their cubes in an already created table. ‘n’ should be taken as an input from the user. The program will also display the output of the table



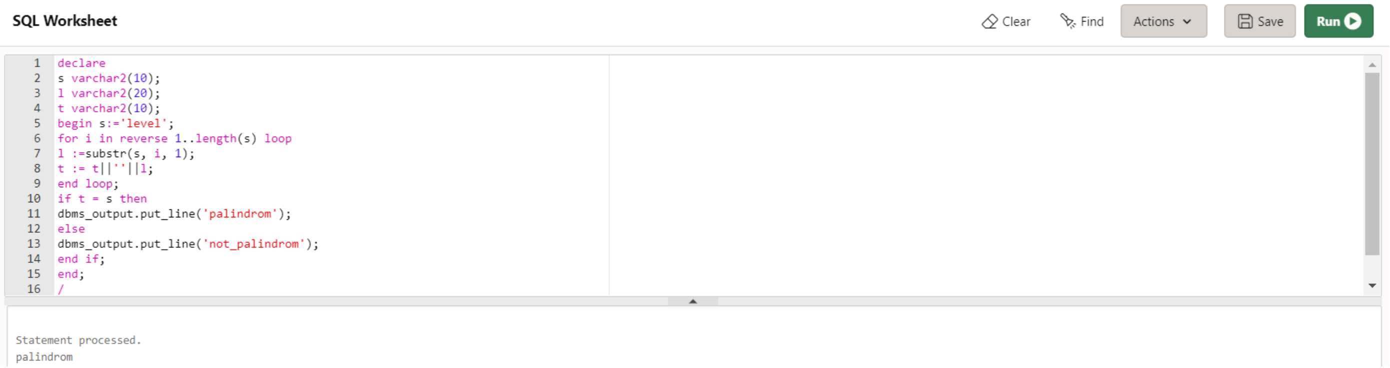
```
CREATE TABLE cube(sl int,cube int);
declare
n number;
s number;
begin
n:=7;
for i in 1..n
loop s:=i*i*i;
insert into cube values(i,s);
end loop;
end;
/

select * from cube;
```

```
Statement processed.
```

SL	CUBE
1	1
2	8
3	27
4	64
5	125
6	216
7	343

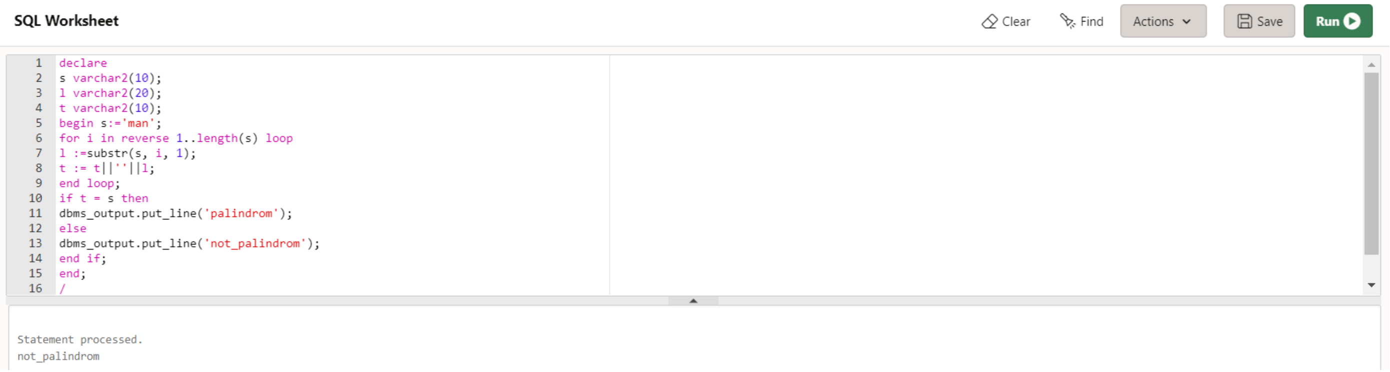
Download CSV
7 rows selected.

8.(b) Write a PL/SQL code to check whether a given string is a palindrome or not. The string should be taken as an input from the user.



```
declare
s varchar2(10);
l varchar2(20);
t varchar2(10);
begin s:='level';
for i in reverse 1..length(s) loop
l :=substr(s, i, 1);
t := t||''||l;
end loop;
if t = s then
dbms_output.put_line('palindrom');
else
dbms_output.put_line('not_palindrom');
end if;
end;
/
```

```
Statement processed.
palindrom
```



```
declare
s varchar2(10);
l varchar2(20);
t varchar2(10);
begin s:='man';
for i in reverse 1..length(s) loop
l :=substr(s, i, 1);
t := t||''||l;
end loop;
if t = s then
dbms_output.put_line('palindrom');
else
dbms_output.put_line('not_palindrom');
end if;
end;
/
```

```
Statement processed.
not_palindrom
```