DBMS lab -1

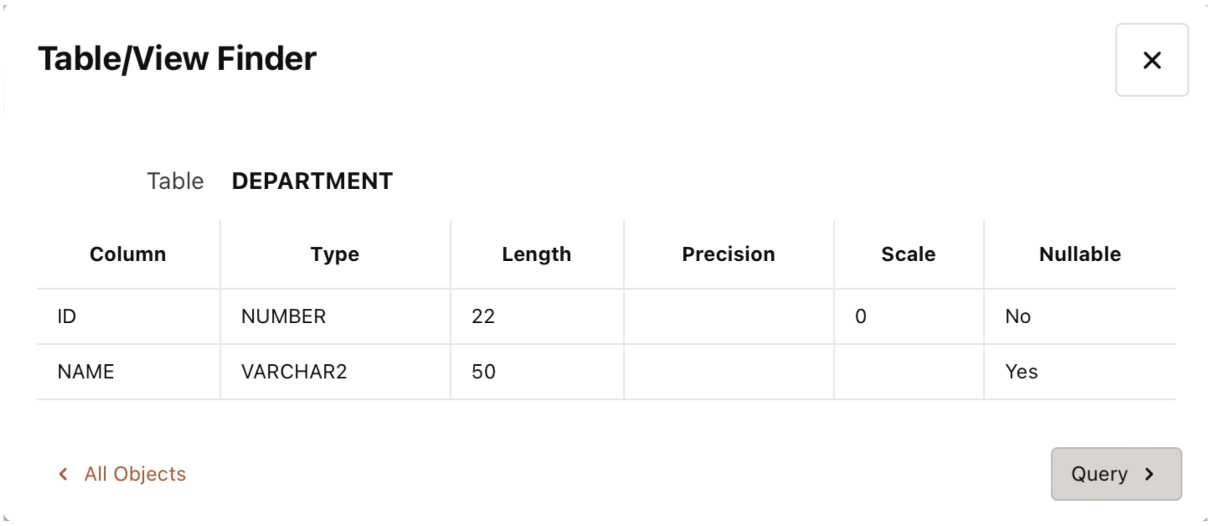
Create Table, Alter, Insert, Update, Delete and Drop

Table Creation

create table department(id int primary key, name varchar(50));

Output

Table Created.



Table/View Finder

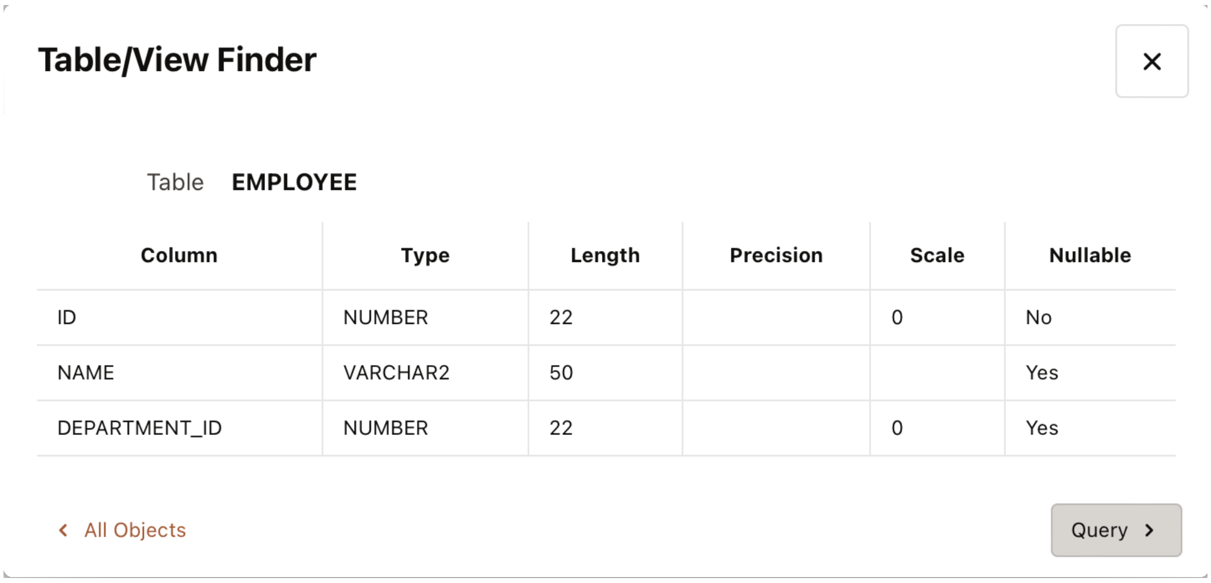
Table **DEPARTMENT**

Column	Type	Length	Precision	Scale	Nullable
ID	NUMBER	22		0	No
NAME	VARCHAR2	50			Yes

create table employee(id int primary key, name varchar(50), department_id int, foreign key (department_id) references department(id));

Output

Table Created.



Table/View Finder

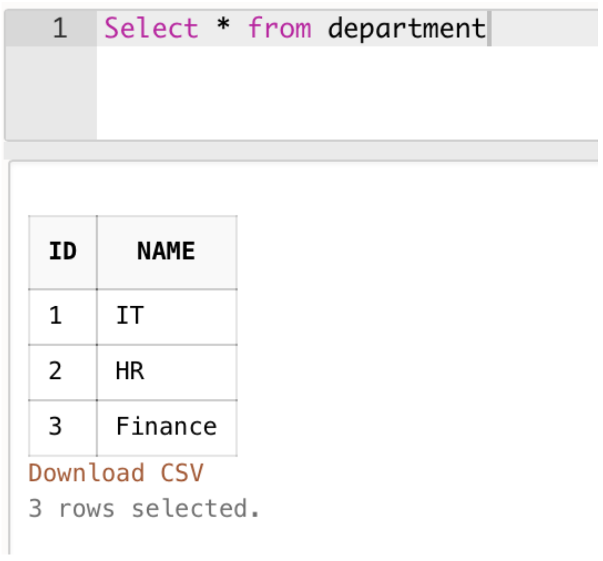
Table **EMPLOYEE**

Column	Type	Length	Precision	Scale	Nullable
ID	NUMBER	22		0	No
NAME	VARCHAR2	50			Yes
DEPARTMENT_ID	NUMBER	22		0	Yes

Insert into Tables

insert into department values(1,'IT');
insert into department values(2,'HR');
insert into department values(3,'Finance');

Output

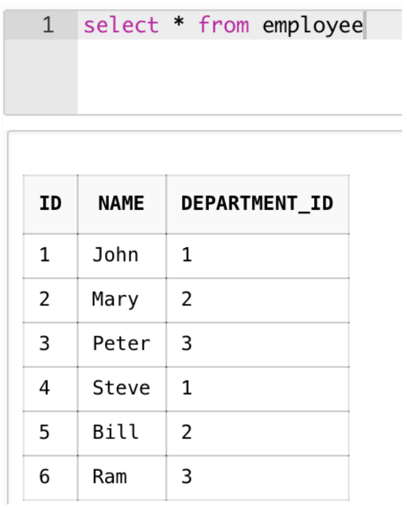


```
Select * from department
```

ID	NAME
1	IT
2	HR
3	Finance

Download CSV
3 rows selected.

insert into employee values(1,'John',1);
insert into employee values(2,'Mary',2);
insert into employee values(3,'Peter',3);
insert into employee values(4,'Steve',1);
insert into employee values(5,'Bill',2);
insert into employee values(6,'Ram',3);

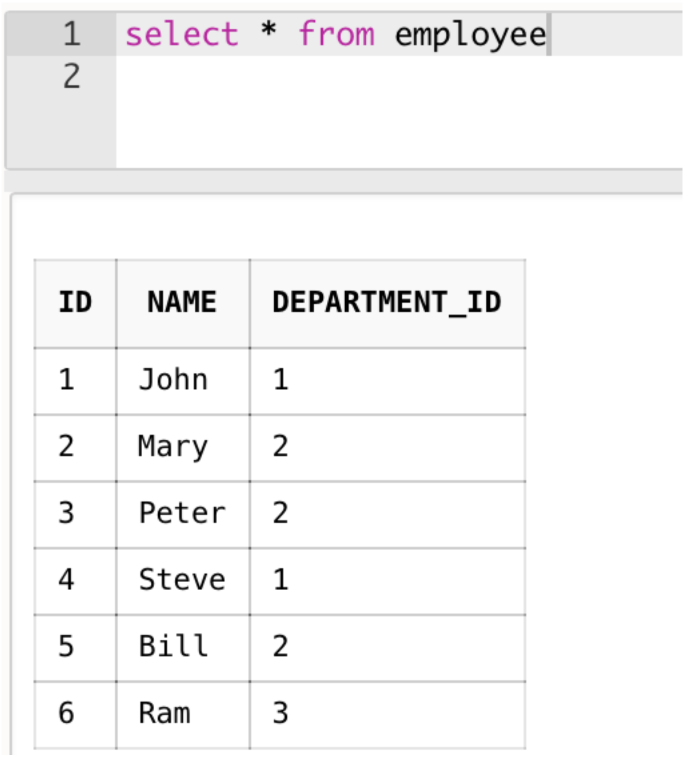


```
select * from employee
```

ID	NAME	DEPARTMENT_ID
1	John	1
2	Mary	2
3	Peter	3
4	Steve	1
5	Bill	2
6	Ram	3

Update the table

update employee set department_id=2 where id=3;

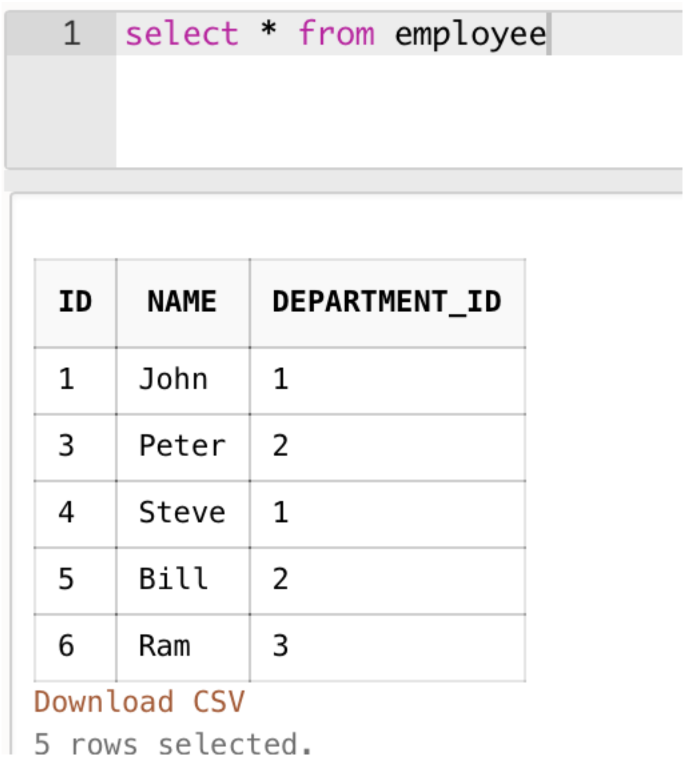


```
select * from employee
```

ID	NAME	DEPARTMENT_ID
1	John	1
2	Mary	2
3	Peter	2
4	Steve	1
5	Bill	2
6	Ram	3

Delete the table

delete from employee where id=2;



```
select * from employee
```

ID	NAME	DEPARTMENT_ID
1	John	1
3	Peter	2
4	Steve	1
5	Bill	2
6	Ram	3

```
Download CSV
5 rows selected.
```

Create a sample table and drop it

```
create table dump(id int primary key, name varchar(50));
```

```
drop table dump;
```

Output

```
Table dropped.
```

DBMS Lab-2

Work with permissions of tables using SQL Queries such as Grant, Revoke, Commit, rollback, savepoint, Select

```
grant select on employee to hr;
revoke select on employee from hr;
```

Grant is used for permitting the users and revoke is used for removing the authorization. Here the Select permission on the employee to hr is granted and revoked next.

```
--commit employee table
commit;
```

commit is used to permanently save all the changes made in the transaction of a database or table **employee.**

```
--rollback employee table
rollback;
```

Rollback is used to undo the transactions that aren’t saved yet in the database.

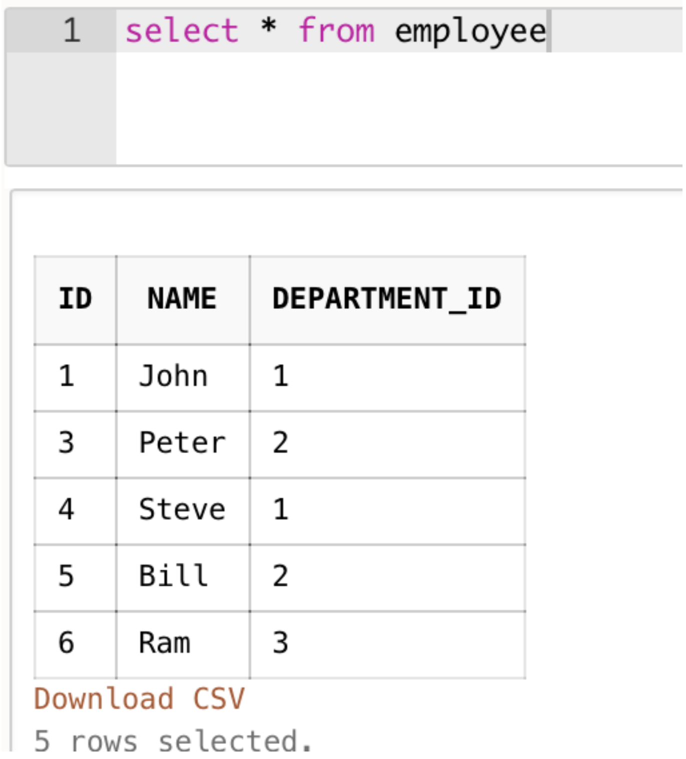
```
--savepoint employee table
savepoint employee;
```

A Savepoint is a point in a transaction in which can roll the transaction back to a certain point without rolling back the entire transaction.

DBMS Lab – 3

View and Visualize all kind of joins

View the table



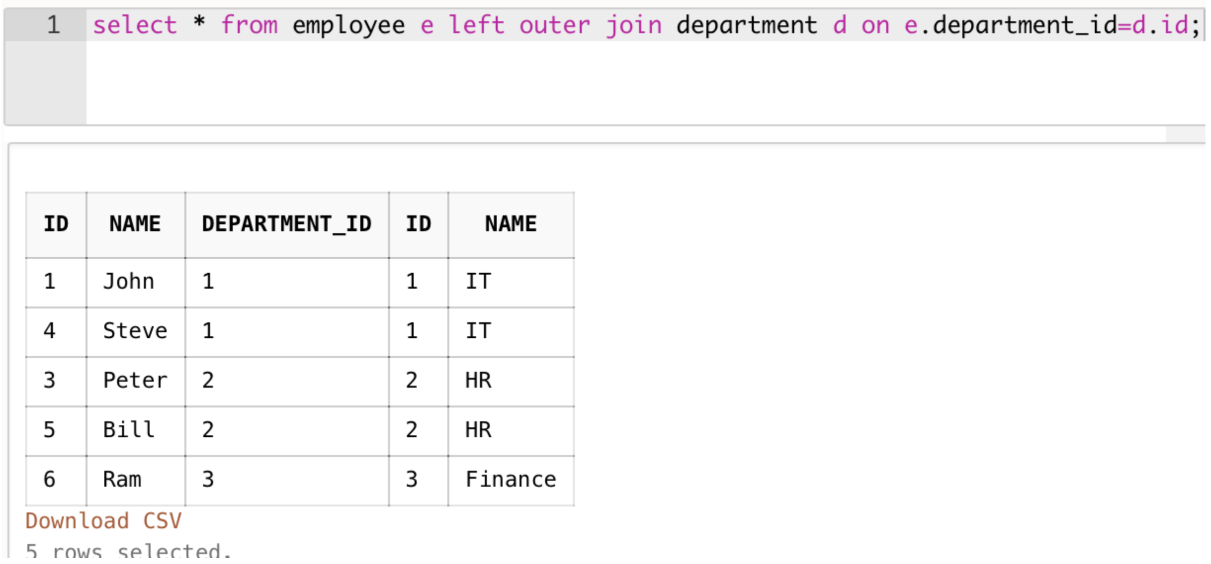
```
select * from employee
```

ID	NAME	DEPARTMENT_ID
1	John	1
3	Peter	2
4	Steve	1
5	Bill	2
6	Ram	3

Download CSV

5 rows selected.

Outer Join



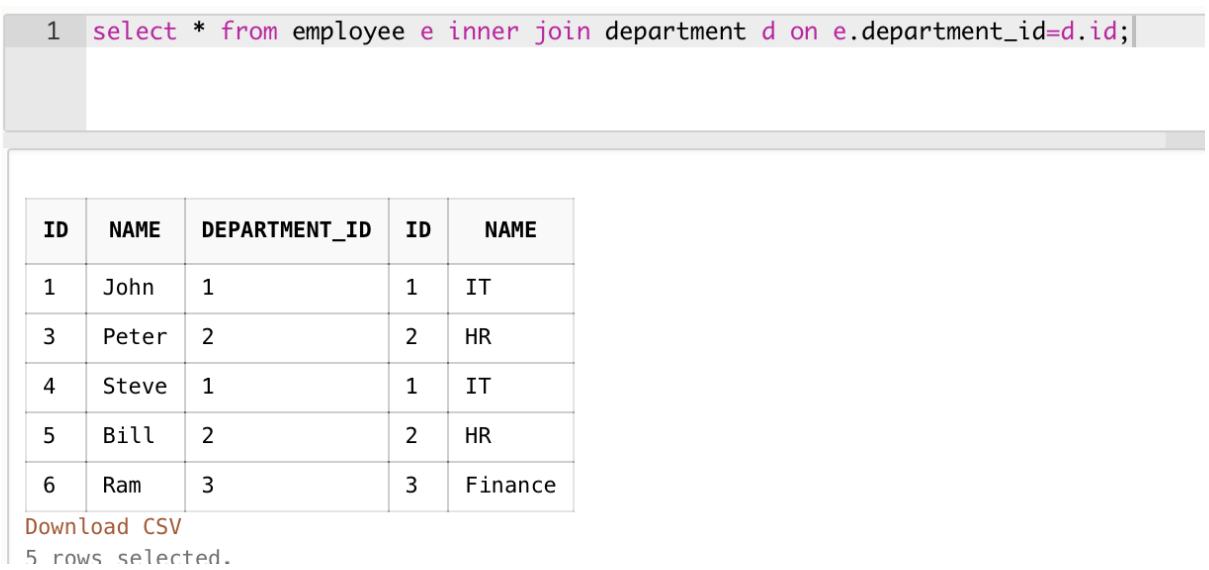
```
select * from employee e left outer join department d on e.department_id=d.id;
```

ID	NAME	DEPARTMENT_ID	ID	NAME
1	John	1	1	IT
4	Steve	1	1	IT
3	Peter	2	2	HR
5	Bill	2	2	HR
6	Ram	3	3	Finance

Download CSV

5 rows selected.

Inner Join

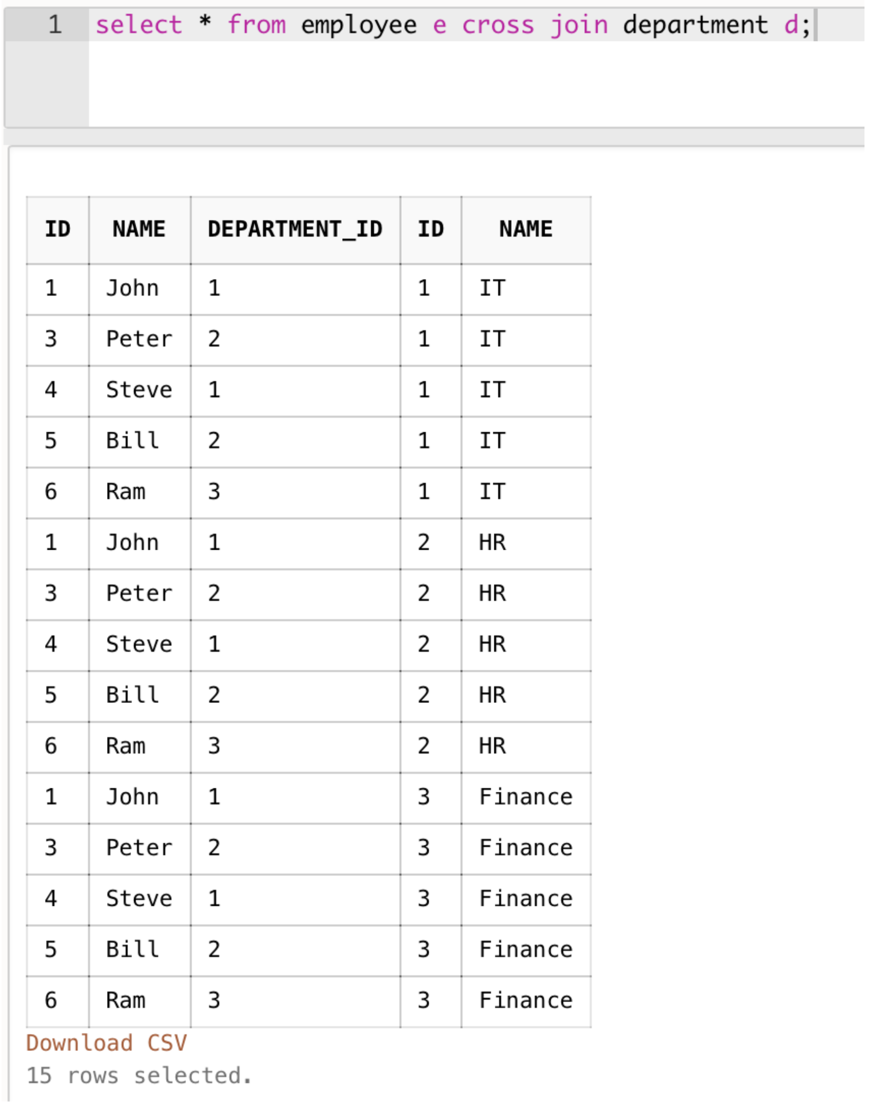


```
select * from employee e inner join department d on e.department_id=d.id;
```

ID	NAME	DEPARTMENT_ID	ID	NAME
1	John	1	1	IT
3	Peter	2	2	HR
4	Steve	1	1	IT
5	Bill	2	2	HR
6	Ram	3	3	Finance

Download CSV
5 rows selected.

Cross Join



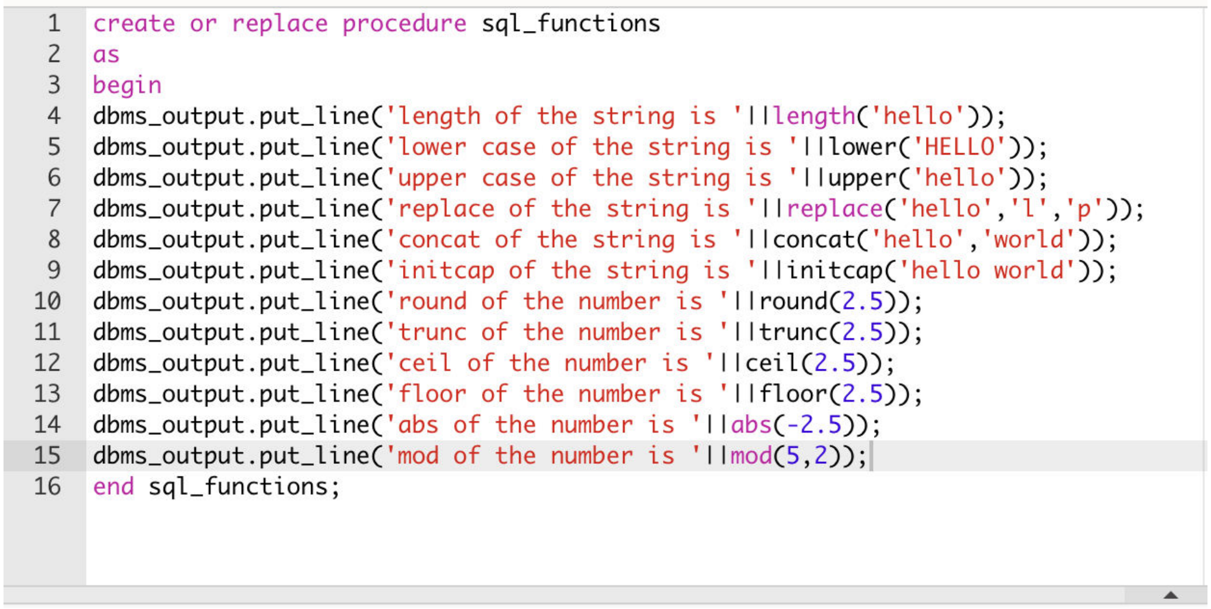
```
select * from employee e cross join department d;
```

ID	NAME	DEPARTMENT_ID	ID	NAME
1	John	1	1	IT
3	Peter	2	1	IT
4	Steve	1	1	IT
5	Bill	2	1	IT
6	Ram	3	1	IT
1	John	1	2	HR
3	Peter	2	2	HR
4	Steve	1	2	HR
5	Bill	2	2	HR
6	Ram	3	2	HR
1	John	1	3	Finance
3	Peter	2	3	Finance
4	Steve	1	3	Finance
5	Bill	2	3	Finance
6	Ram	3	3	Finance

Download CSV
15 rows selected.

DBMS lab-5

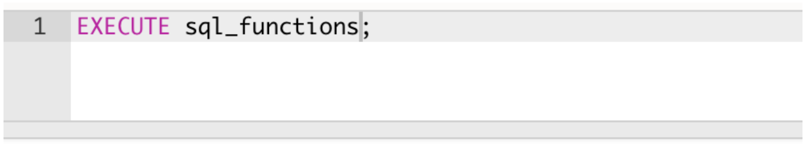
PLSQL Procedures



```
create or replace procedure sql_functions
as
begin
dbms_output.put_line('length of the string is '||length('hello'));
dbms_output.put_line('lower case of the string is '||lower('HELLO'));
dbms_output.put_line('upper case of the string is '||upper('hello'));
dbms_output.put_line('replace of the string is '||replace('hello','l','p'));
dbms_output.put_line('concat of the string is '||concat('hello','world'));
dbms_output.put_line('initcap of the string is '||initcap('hello world'));
dbms_output.put_line('round of the number is '||round(2.5));
dbms_output.put_line('trunc of the number is '||trunc(2.5));
dbms_output.put_line('ceil of the number is '||ceil(2.5));
dbms_output.put_line('floor of the number is '||floor(2.5));
dbms_output.put_line('abs of the number is '||abs(-2.5));
dbms_output.put_line('mod of the number is '||mod(5,2));
end sql_functions;
```

```
Procedure created.
```

Execute the procedure

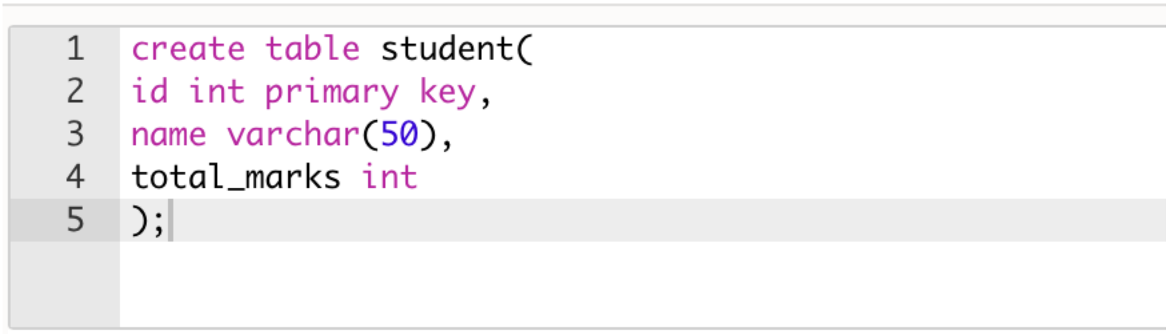


```
EXECUTE sql_functions;
```

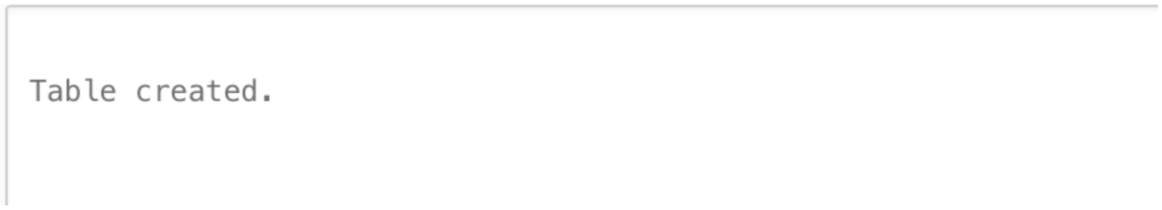
```
Statement processed.
length of the string is 5
lower case of the string is hello
upper case of the string is HELLO
replace of the string is heppo
concat of the string is helloworld
initcap of the string is Hello World
round of the number is 3
trunc of the number is 2
ceil of the number is 3
floor of the number is 2
abs of the number is 2.5
mod of the number is 1
```

DBMS lab-6

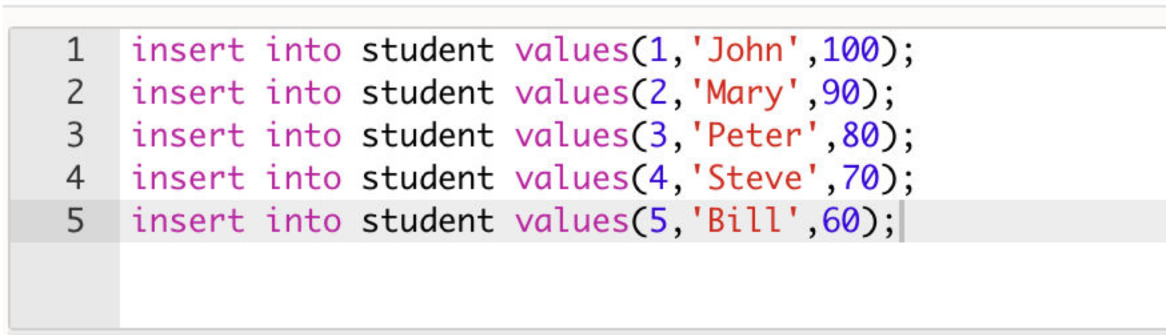
Create Triggers on the table student



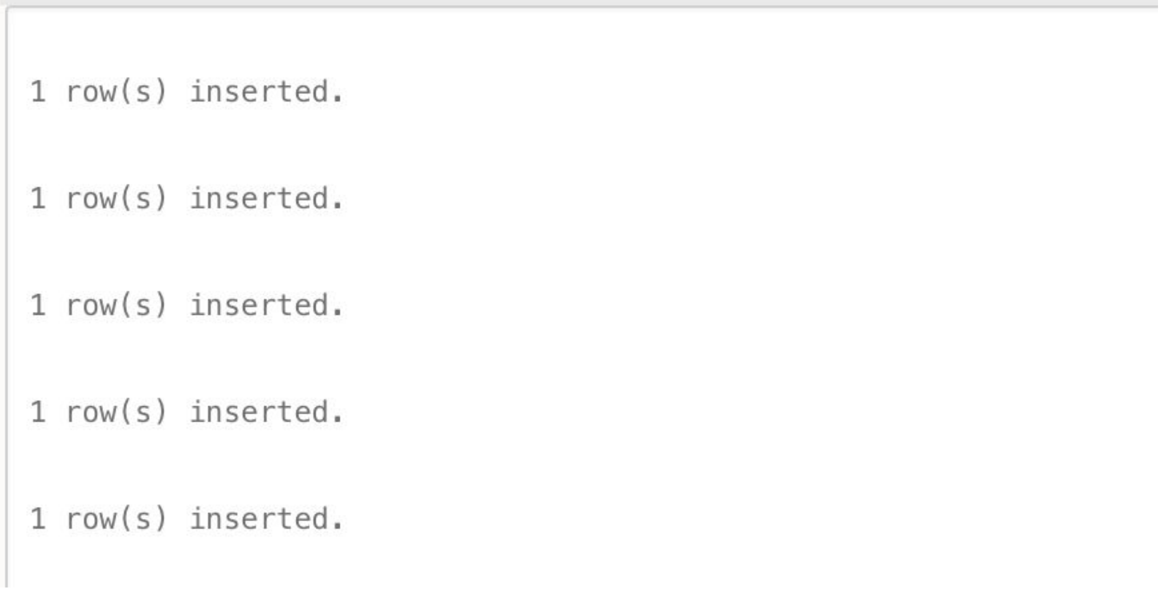
```sql
create table student(
id int primary key,
name varchar(50),
total_marks int
);
```



```
Table created.
```

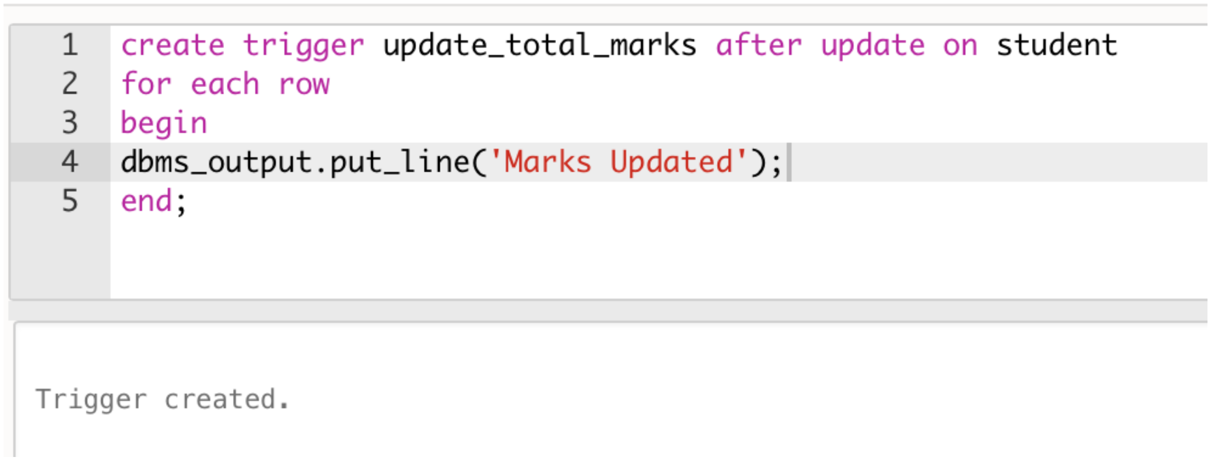


```sql
insert into student values(1,'John',100);
insert into student values(2,'Mary',90);
insert into student values(3,'Peter',80);
insert into student values(4,'Steve',70);
insert into student values(5,'Bill',60);
```



```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

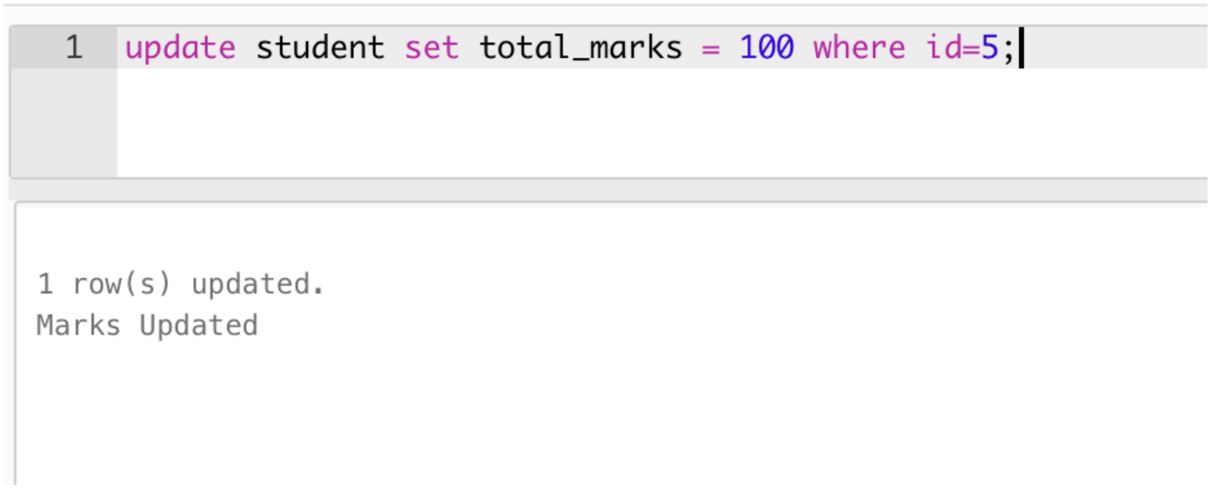
Creating a trigger to update the marks



```
create trigger update_total_marks after update on student
for each row
begin
dbms_output.put_line('Marks Updated');
end;
```

```
Trigger created.
```

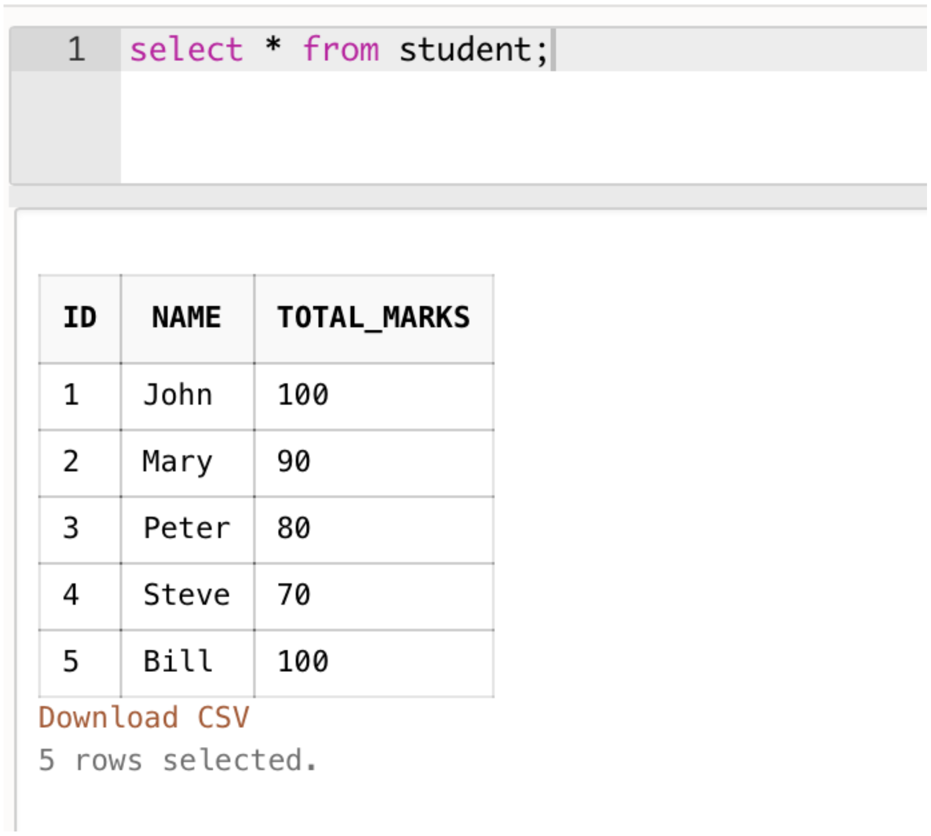
Table row updated



```
update student set total_marks = 100 where id=5;
```

```
1 row(s) updated.
Marks Updated
```

Viewing the table

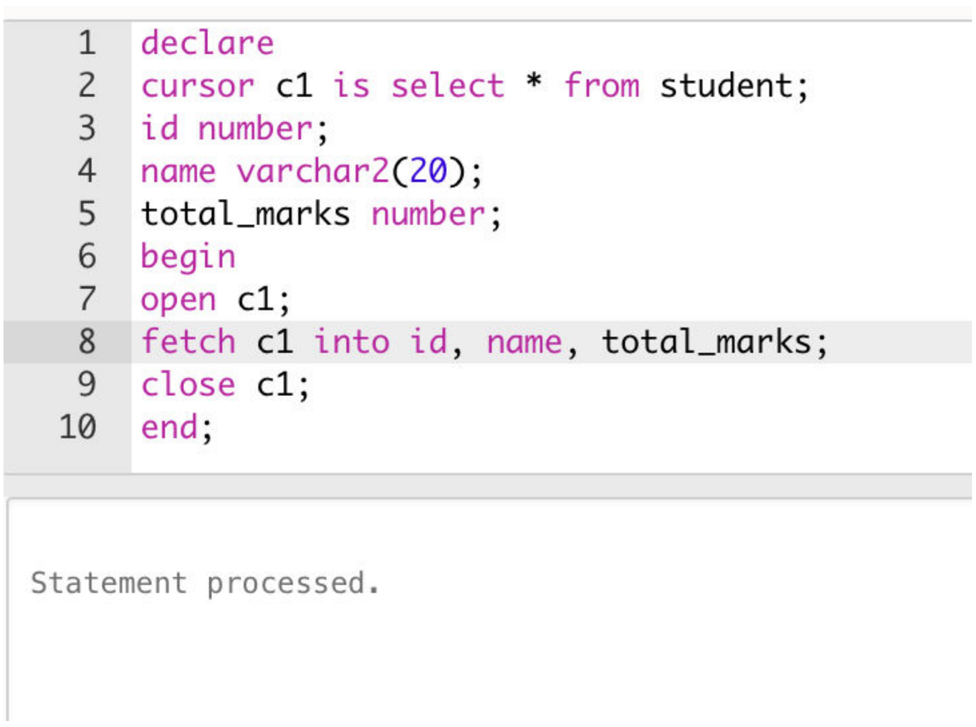


```
select * from student;
```

ID	NAME	TOTAL_MARKS
1	John	100
2	Mary	90
3	Peter	80
4	Steve	70
5	Bill	100

```
Download CSV
5 rows selected.
```

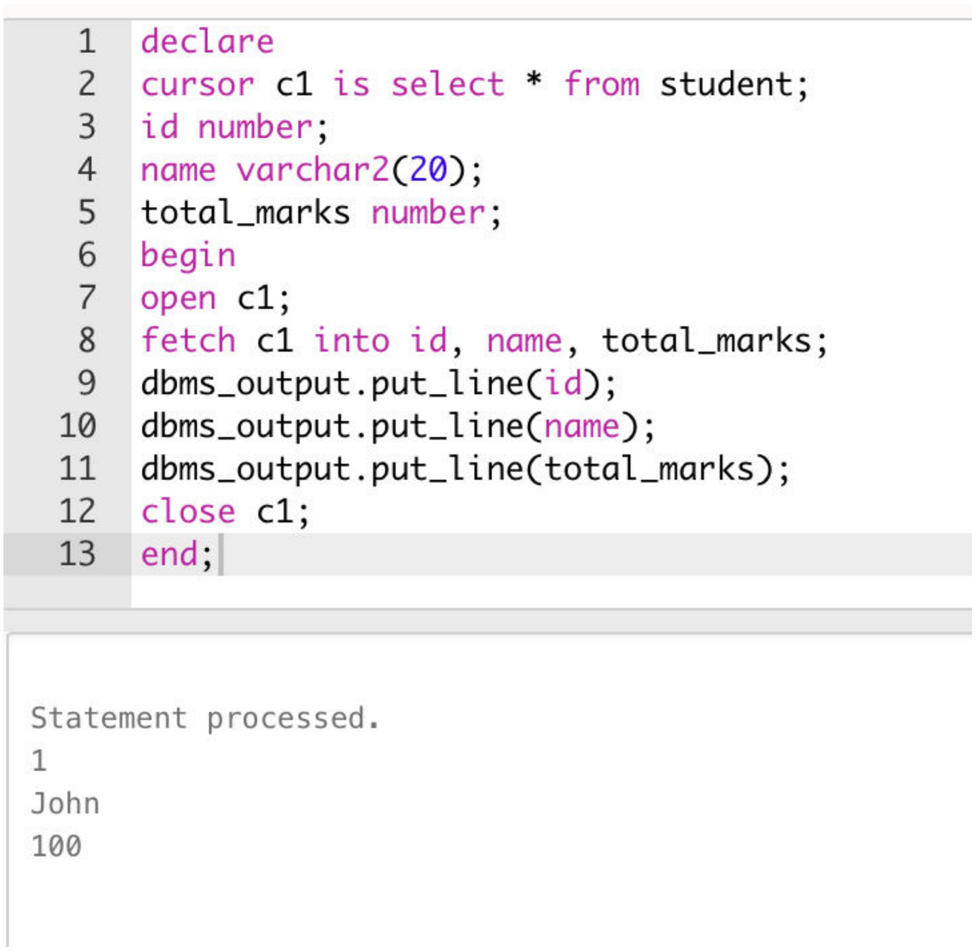
Creating a simple cursor



```
declare
cursor c1 is select * from student;
id number;
name varchar2(20);
total_marks number;
begin
open c1;
fetch c1 into id, name, total_marks;
close c1;
end;
```

```
Statement processed.
```

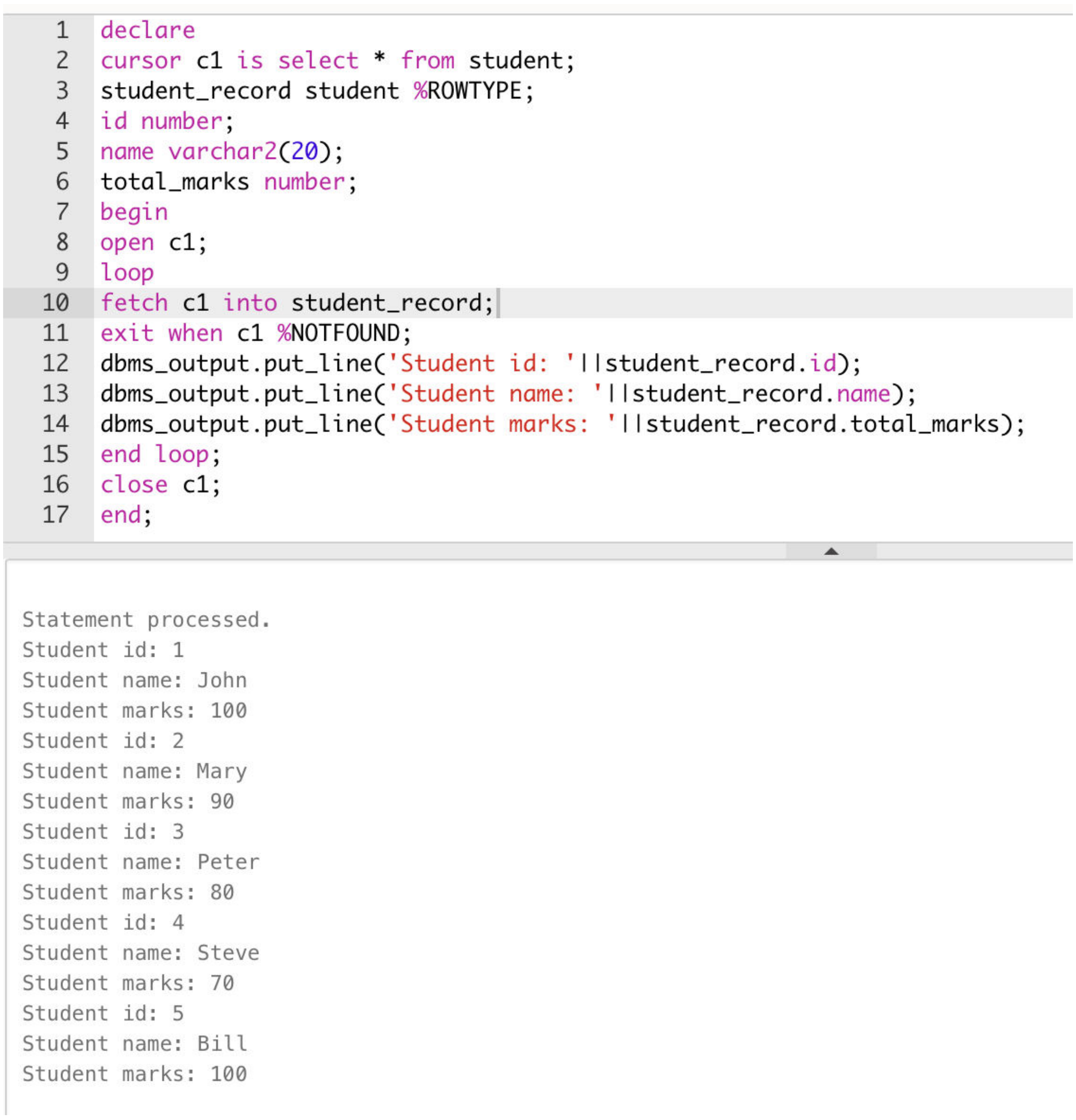
Viewing through cursors



```
declare
cursor c1 is select * from student;
id number;
name varchar2(20);
total_marks number;
begin
open c1;
fetch c1 into id, name, total_marks;
dbms_output.put_line(id);
dbms_output.put_line(name);
dbms_output.put_line(total_marks);
close c1;
end;
```

```
Statement processed.
1
John
100
```

Creating a cursor and use loops to view it



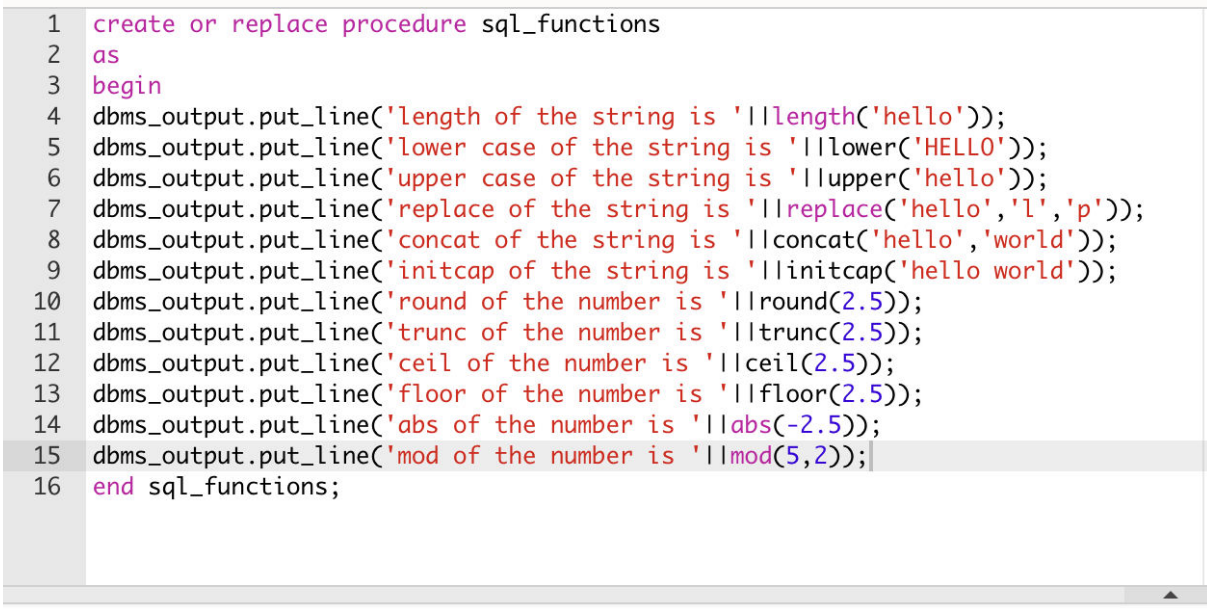
```
declare
cursor c1 is select * from student;
student_record student %ROWTYPE;
id number;
name varchar2(20);
total_marks number;
begin
open c1;
loop
fetch c1 into student_record;
exit when c1 %NOTFOUND;
dbms_output.put_line('Student id: '||student_record.id);
dbms_output.put_line('Student name: '||student_record.name);
dbms_output.put_line('Student marks: '||student_record.total_marks);
end loop;
close c1;
end;
```

```
Statement processed.
Student id: 1
Student name: John
Student marks: 100
Student id: 2
Student name: Mary
Student marks: 90
Student id: 3
Student name: Peter
Student marks: 80
Student id: 4
Student name: Steve
Student marks: 70
Student id: 5
Student name: Bill
Student marks: 100
```

DBMS lab-7

Prepare using SQL Builtin Functions and write a program that works with PL/SQL Functions

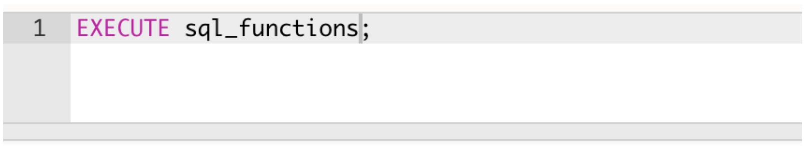
PLSQL Procedures



```
create or replace procedure sql_functions
as
begin
dbms_output.put_line('length of the string is '||length('hello'));
dbms_output.put_line('lower case of the string is '||lower('HELLO'));
dbms_output.put_line('upper case of the string is '||upper('hello'));
dbms_output.put_line('replace of the string is '||replace('hello','l','p'));
dbms_output.put_line('concat of the string is '||concat('hello','world'));
dbms_output.put_line('initcap of the string is '||initcap('hello world'));
dbms_output.put_line('round of the number is '||round(2.5));
dbms_output.put_line('trunc of the number is '||trunc(2.5));
dbms_output.put_line('ceil of the number is '||ceil(2.5));
dbms_output.put_line('floor of the number is '||floor(2.5));
dbms_output.put_line('abs of the number is '||abs(-2.5));
dbms_output.put_line('mod of the number is '||mod(5,2));
end sql_functions;
```

```
Procedure created.
```

Execute the procedure



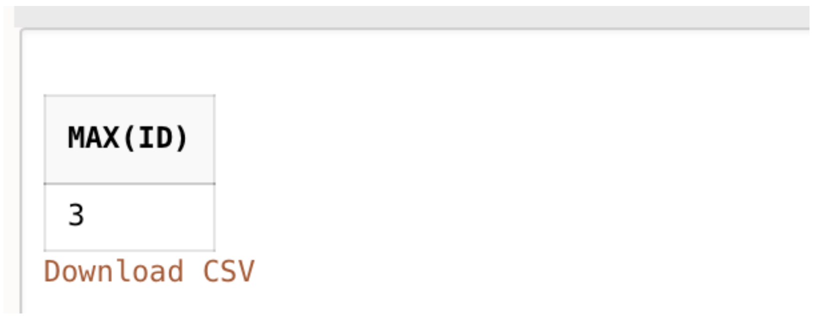
```
EXECUTE sql_functions;
```

```
Statement processed.
length of the string is 5
lower case of the string is hello
upper case of the string is HELLO
replace of the string is heppo
concat of the string is helloworld
initcap of the string is Hello World
round of the number is 3
trunc of the number is 2
ceil of the number is 3
floor of the number is 2
abs of the number is 2.5
mod of the number is 1
```

PLSQL Function

```
create or replace function max_department
return number
as
begin
return (select max(id) from department);
end;
```

Execute Function

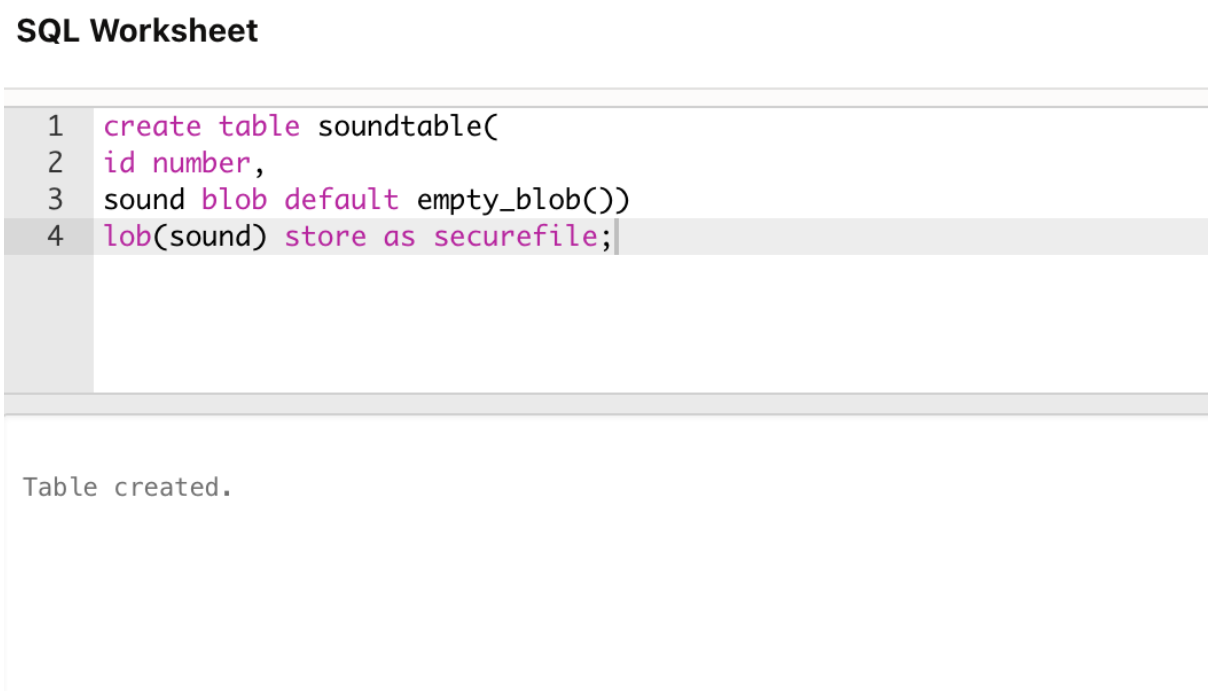


MAX(ID)
3

Download CSV

```
Create a Multimedia table with multimedia datatypes image and
video. Use Search and Filters in it.

-- create a table using image and video data types
CREATE TABLE multimedia (
    id serial PRIMARY KEY,
    image image,
    video video
);
-- insert a row into the table
INSERT INTO multimedia (image, video)

VALUES (

'http://www.postgresql.org/media/img/about/press/elephant.png',

'http://www.postgresql.org/media/mov/about/press/elephant.ogv'
);

-- use search to find the row
SELECT * FROM multimedia WHERE image @> 'elephant.png';

-- use filter to find the row
SELECT * FROM multimedia WHERE video <@ 'elephant.ogv';

-- create a plsql function to return the image and video data
CREATE OR REPLACE FUNCTION get_multimedia(id integer)
RETURNS multimedia AS $$
DECLARE
    m multimedia;
BEGIN
    SELECT * INTO m FROM multimedia WHERE id = $1;
    RETURN m;
END;
$$ LANGUAGE plpgsql;

-- call the function
SELECT get_multimedia(1);
```

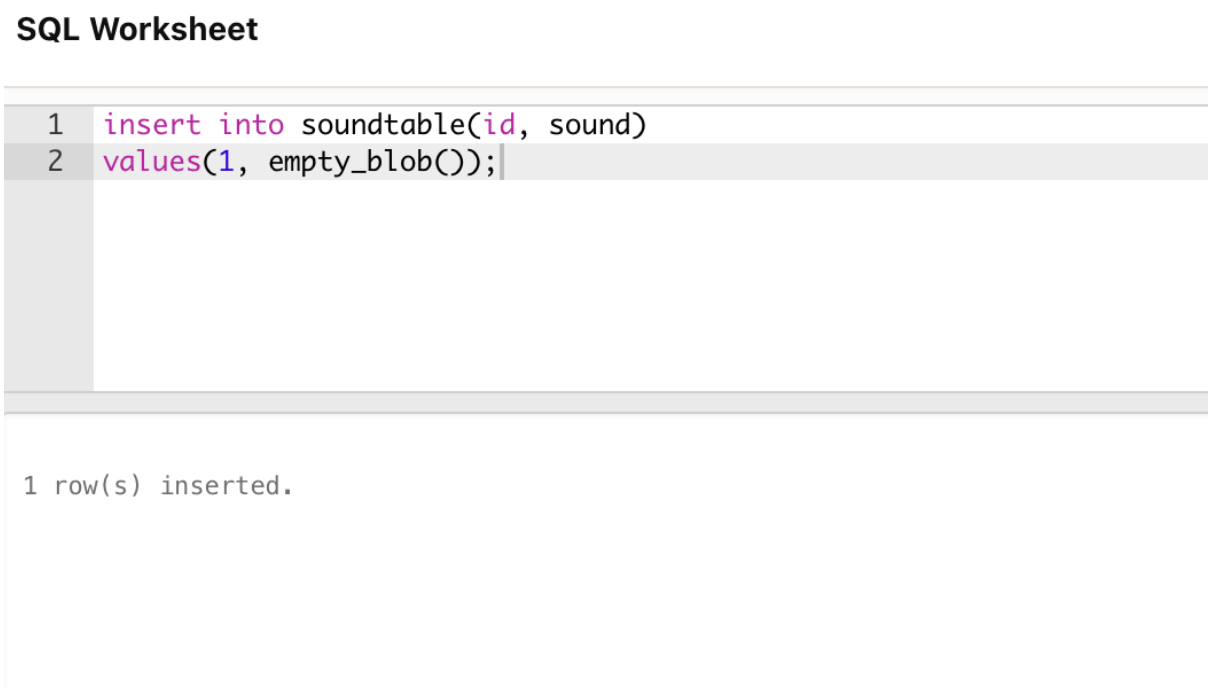
Create Multimedia table using Multimedia datatypes and Create a View for the Sound table



SQL Worksheet

```
create table soundtable(
id number,
sound blob default empty_blob())
lob(sound) store as securefile;
```

```
Table created.
```

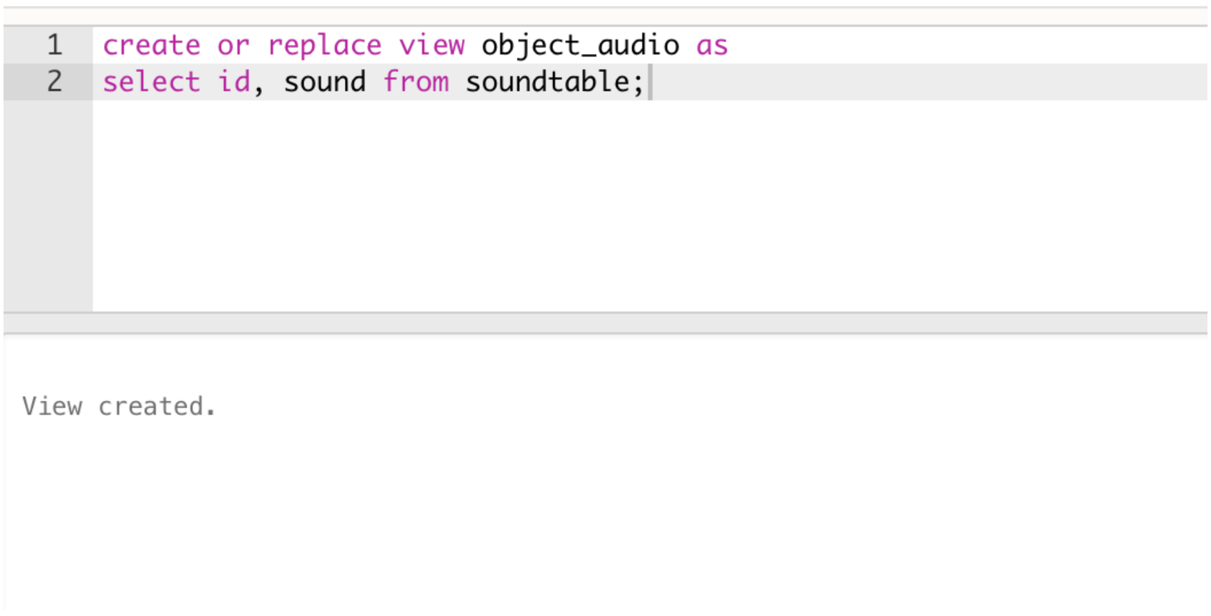


SQL Worksheet

```
insert into soundtable(id, sound)
values(1, empty_blob());
```

```
1 row(s) inserted.
```

SQL Worksheet



```sql
create or replace view object_audio as
select id, sound from soundtable;
```

```
View created.
```