The Islamic University of Gaza
Engineering Faculty
Computer Engineering Department

Advance Database Lab

Oracle Project Part 3

Done By :

Ayah Abu Hamra 220064486

Islam Abu Salem 220060769

Submitted To :

Eng. Doa'a Abu Jabal

For your college database created in part 1&2:

- Create a Package college that will contain all your work .

Package specification:-

```
create or replace package college is

PROCEDURE Teachers_insert(id number,
First_Name varchar2,
Last_name varchar2,
Head_ID Number);
---------------------------------------------
PROCEDURE Courses_insert(course_ID Number,
Course_Name varchar2,
course_year Number,
Number_Of_Hours Number,
Description varchar2,
Teacher_id number);
---------------------------------------------
Procedure departments_heads_insert(Department varchar2,
Head_ID NUMBER,
Head_Assisst_ID Number);
---------------------------------------------
PROCEDURE students_insert(ID number,
First_Name varchar2,
Last_name varchar2,
course_ID number,
course_Year number,
number_of_hours number,
Grade number);

---------------------------------------------
PROCEDURE Teacher_Courses_insert2(Teacher_id number,
                                  Teacher_Name       varchar2,
                                  head number,
                                   Course_code        varchar2,
                                   Course_Name        varchar2,
                                   Course_description varchar2);
---------------------------------------------------------------------
PROCEDURE Student_Courses_insert2(stu_id number,
                                  Teacher_id number,
                                  head number,
                                  Student_Name       varchar2,
                                   Teacher_Name       varchar2,
                                   Course_code        varchar2,
                                   Course_Name        varchar2,
                                   Course_description varchar2,
                                   grade              number);
---------------------------------------------
PROCEDURE Tea_Course_Students_insert2(Teacher_id number,
                                      Teacher_Name   varchar2,
                                      Course_code    varchar2,
                                       course_name varchar2,
                                      stu_id number,
                                      Student_Name   varchar2,
                                      grade          number);
-----------------------------------
PROCEDURE Degree_update(New_Degree Number, Student_ID Number);
-------------------------------------
PROCEDURE Teacher_update(teach_id number, id number);
-------------------------------------
PROCEDURE Head_id_update(new_ID NUMBER, dep VARCHAR2);
-------------------------------------
PROCEDURE Delete_teacher(teach_first_name varchar2,
teach_last_name varchar2);
-------------------------------------
PROCEDURE Print_Teacher_Name;
-------------------------------------
PROCEDURE Print_Students(ID NUMBER, cYear NUMBER, Hours NUMBER);
-------------------------------------
```

```
-------------------------------------
PROCEDURE Print_Students(Name varchar2);
-------------------------------------
FUNCTION Teacher_info(teach_id number) RETURN VARCHAR2;
----------------------------------------
FUNCTION Course_description(code VARCHAR2) RETURN VARCHAR2;
-----------------------------------------

end college;
```

Package Body:-

```
SET SERVEROUTPUT ON

create or replace package body college is

PROCEDURE Teachers_insert(id number,
                          First_Name varchar2,
                          Last_name  varchar2,
                          Head_ID Number) Is
begin
insert into Teachers Values (id, First_Name, Last_name, Head_ID);
end;

-----------------------------------------------
PROCEDURE Courses_insert(course_ID Number,
                         Course_Name varchar2,
                         course_year Number,
                         Number_Of_Hours Number,
                         Description varchar2,
                         Teacher_id  number) Is
begin
insert into Courses
values
       (course_ID,
        Course_Name,
        course_year,
        Number_Of_Hours,
        Description,
        Teacher_id);
end;
-----------------------------------------------
Procedure departments_heads_insert(Department varchar2,
                                   Head_ID NUMBER,
                                   Head_Assisst_ID Number) IS
begin
```

```
begin
insert into departments_heads
values
       (Department, Head_ID, Head_Assisst_ID);
end;
-----------------------------------------------
PROCEDURE students_insert(ID number,
                          First_Name       varchar2,
                          Last_name        varchar2,
                          course_ID        number,
                          course_Year      number,
                          number_of_hours  number,
                          Grade            number) IS
begin
insert into students
values
       (ID,
        First_Name,
        Last_name,
        course_ID,
        course_Year,
        number_of_hours,
        Grade);
end;
-----------------------------------------------------------------
PROCEDURE Teacher_Courses_insert2(Teacher_id number,
                                  Teacher_Name        varchar2,
                                  head number,
                                   Course_code         varchar2,
                                   Course_Name         varchar2,
                                   Course_description varchar2) IS

 index2 number;
 Tfname Teachers.First_Name%type;
```

```
 Tlname Teachers.Last_Name%type;
 cid courses.course_id%type;
 cyear courses.Course_Year%type;
 chours courses.Number_Of_Hours%type;


begin
index2:=instr(Teacher_Name,' ');
Tfname:=substr(Teacher_Name,1,index2);
Tlname:=substr(Teacher_Name,index2);

cid:=substr(Course_code,1,1);
cyear:=substr(Course_code,2,1);
chours:=substr(Course_code,3,1);

insert into teachers
values(Teacher_id,
       Tfname,
       Tlname,
       head);

insert into courses
Values
      (cid,
        Course_Name,
        cyear,
        chours,
        Course_description,
        Teacher_id);

end;
-------------------------------------------------------
```

```
PROCEDURE Student_Courses_insert2(stu_id number,
                                  Teacher_id number,
                                  head number,
                                  Student_Name       varchar2,
                                   Teacher_Name       varchar2,
                                   Course_code        varchar2,
                                   Course_Name        varchar2,
                                   Course_description varchar2,
                                   grade              number) IS
 index1 number;
 index2 number;
 fname Students.First_Name%type;
 lname Students.Last_Name%type;
 Tfname Teachers.First_Name%type;
 Tlname Teachers.Last_Name%type;
 cid courses.course_id%type;
 cyear courses.Course_Year%type;
 chours courses.Number_Of_Hours%type;


begin
index1:=instr(Student_Name,' ');
fname:=substr(Student_Name,1,index1);
lname:=substr(Student_Name,index1);
index2:=instr(Teacher_Name,' ');
Tfname:=substr(Teacher_Name,1,index2);
Tlname:=substr(Teacher_Name,index2);

cid:=substr(Course_code,1,1);
cyear:=substr(Course_code,2,1);
chours:=substr(Course_code,3,1);

insert into teachers
values(Teacher_id,
```

```
values(Teacher_id,
       Tfname,
       Tlname,
       head);

insert into courses
Values
      (cid,
        Course_Name,
        cyear,
        chours,
        Course_description,
        Teacher_id);

insert into Students
values
      (stu id,
       fname,
       lname,
       cid,
       cyear,
       chours,
       grade);

end;
-----------------------------------------------------------------
PROCEDURE Tea_Course_Students_insert2(Teacher_id number,
                                      Teacher_Name   varchar2,
                                      Course_code    varchar2,
                                      course_name varchar2,
                                      stu_id number,
                                      Student_Name   varchar2,
                                      grade          number) IS
```

```
 index1 number;
 index2 number;
 fname Students.First_Name%type;
 lname Students.Last_Name%type;
 Tfname Teachers.First_Name%type;
 Tlname Teachers.Last_Name%type;
 cid courses.course_id%type;
 cyear courses.Course_Year%type;
 chours courses.Number_Of_Hours%type;

BEGIN
index1:=instr(Student_Name,' ');
fname:=substr(Student_Name,1,index1);
lname:=substr(Student_Name,index1);
index2:=instr(Teacher_Name,' ');
Tfname:=substr(Teacher_Name,1,index2);
Tlname:=substr(Teacher_Name,index2);

cid:=substr(Course_code,1,1);
cyear:=substr(Course_code,2,1);
chours:=substr(Course_code,3,1);

insert into teachers(id,first_name,last_name)
values(Teacher_id,
       Tfname,
       Tlname
       );

insert into courses(course_id,course_name,course_year,number_of_hours,teacher_id)
Values
      (cid,
      course_name,
        cyear,
        chours,
```

```
        Teacher_id);

insert into Students
values
      (stu_id,
       fname,
       lname,
       cid,
       cyear,
       chours,
       grade);

END;
-----------------------------------------------
PROCEDURE Degree_update(New_Degree Number, Student_ID Number) IS
BEGIN
Update Students set grade = New_Degree where ID = Student_ID;
end;
-----------------------------------------------
PROCEDURE Teacher_update(teach_id number, id number) is
BEGIN
update Courses set Teacher_id = teach_id where course_ID = id;
END;
-----------------------------------------------
PROCEDURE Head_id_update(new_ID NUMBER, dep VARCHAR2) is
    head departments_heads.head_ID%Type;
BEGIN
SELECT head_ID into head FROM departments_heads WHERE Department = dep;
DBMS_OUTPUT.PUT_LINE(head);

update departments_heads set head_ID = NEW_ID where Department = dep;

END;
```

```
PROCEDURE Delete_teacher(teach_first_name varchar2,
                         teach_last_name  varchar2) IS
    teach_F TEACHERS.FIRST_NAME%TYPE;
    teach_L TEACHERS.LAST_NAME%TYPE;

BEGIN
SELECT FIRST_NAME, LAST_NAME INTO teach_F, teach_L
FROM TEACHERS
WHERE FIRST_NAME = teach_first_name
and LAST_NAME = teach_last_name;
DELETE FROM TEACHERS
WHERE FIRST_NAME = teach_first_name
and LAST_NAME = teach_last_name;
DBMS_OUTPUT.PUT_LINE(teach_F);
DBMS_OUTPUT.PUT_LINE(teach_L);
END;
-----------------------------------------------
PROCEDURE Print_Teacher_Name IS
    teach_F TEACHERS.FIRST_NAME%TYPE;
    teach_L TEACHERS.LAST_NAME%TYPE;
    counter number;
BEGIN
counter:=0;
SELECT First_Name, Last_Name
INTO teach_F, teach_L
FROM Teachers
WHERE Head_ID IN (SELECT Head_ID FROM departments_heads);
while counter< SQL%rowcount loop
counter:=counter+1;
DBMS_OUTPUT.PUT_LINE(teach_F);
DBMS_OUTPUT.PUT_LINE(teach_L);
end loop;
end;
```

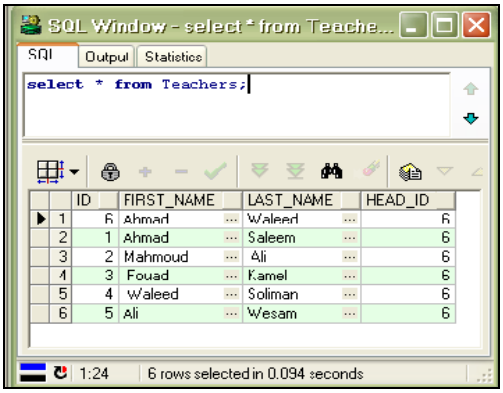
```
PROCEDURE Print_Students(ID NUMBER, cYear NUMBER, Hours NUMBER) IS
    std_F Students.FIRST_NAME%TYPE;
    std_L Students.LAST_NAME%TYPE;
    grd   Students.Grade%TYPE;
    counter number;
    Cursor students_cursor is
    SELECT First_Name, Last_Name, GRADE
         FROM Students
         WHERE Course_ID = ID
          and Course_Year = cYear
          and Number_Of_Hours = Hours;

BEGIN
open students_cursor;
counter:=0;
while counter<students_cursor%rowcount loop
fetch students_cursor into std_F,std_L,grd;
DBMS_OUTPUT.PUT_LINE(std_F);
DBMS_OUTPUT.PUT_LINE(std_L);
DBMS_OUTPUT.PUT_LINE(grd);
counter:=counter+1;
end loop;

END;
-----------------------------------------------
PROCEDURE Print_Students(Name varchar2) IS
    std_name Teacher_Course_Students."Student Name"%type;
    std_cr   Teacher_Course_Students."Course Code"%type;
    std_grd  Teacher_Course_Students.Grade%type;
    counter number;
    Cursor students_cursor is
    SELECT "Student Name", "Course Code", Grade
         FROM Teacher_Course_Students
         WHERE "Teacher Name" = Name;
```

```
BEGIN
open students_cursor;
counter:=0;
while counter<students_cursor%rowcount loop
fetch students_cursor into std_name,std_cr,std_grd;
DBMS_OUTPUT.PUT_LINE(std_name);
DBMS_OUTPUT.PUT_LINE(std_cr);
DBMS_OUTPUT.PUT_LINE(std_grd);
counter:=counter+1;
end loop;
END;
-----------------------------------------------
FUNCTION Teacher_info(teach_id number) RETURN VARCHAR2 IS
first TEACHERS.FIRST_NAME%TYPE;
last  TEACHERS.LAST_NAME%TYPE;

BEGIN
SELECT FIRST_NAME, LAST_NAME
INTO first, last
FROM TEACHERS
WHERE ID = teach_id;
DBMS_OUTPUT.PUT_LINE(first || ' ' || last);
RETURN first || ' ' || last;
END;
-----------------------------------------------
FUNCTION Course_description(code VARCHAR2) RETURN VARCHAR2 IS
    des Teacher_Courses.Description%TYPE;
    id number(1);
    cyear number(1);
    hours number(1);

BEGIN
id:=substr(code,1,1);
cyear:=substr(code,2,1);
```

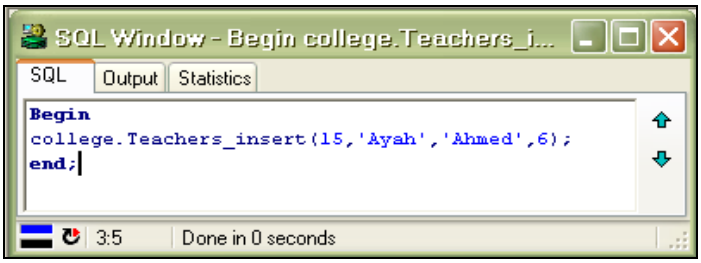
```
hours:=substr(code,3,1);

SELECT Description
INTO des
FROM Courses
WHERE Course_id=id and
Course_year=cyear and
Number_of_Hours=hours;
DBMS_OUTPUT.PUT_LINE( The description of the course code '||code| '  is   '||des);
RETURN des;
END;
-----------------------------------------------
Begin
college.Teachers_insert(12,'Islam','Salem',6);
end college;
```

- Procedures Test :-

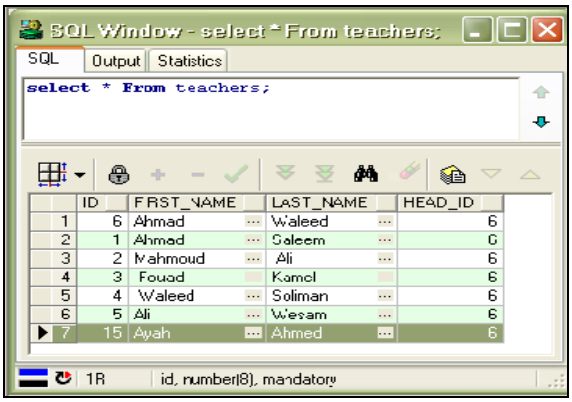
1. Teachers_insert



Before calling "Teacher_insert" procedure.

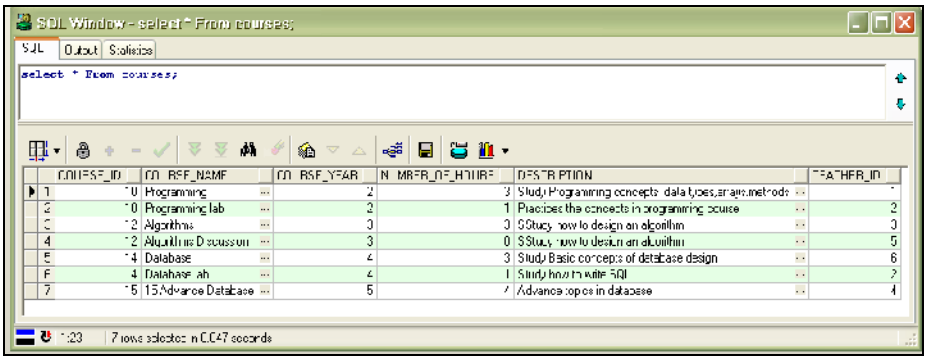


calling "Teacher_insert" procedure.

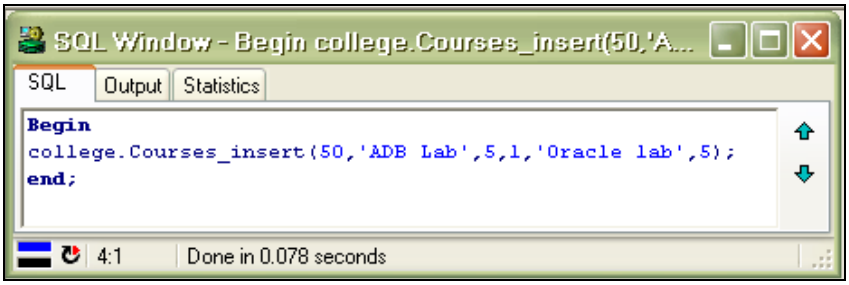


After calling "Teacher_insert" procedure.

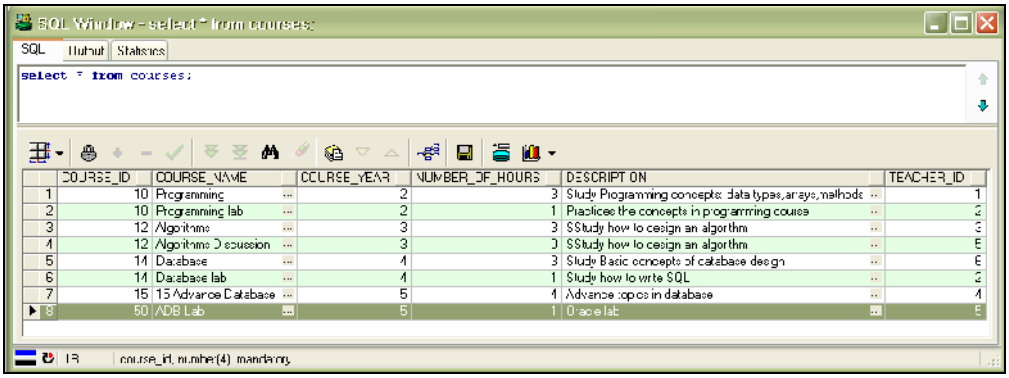
2. Courses_insert



before calling "course_insert" procedure.

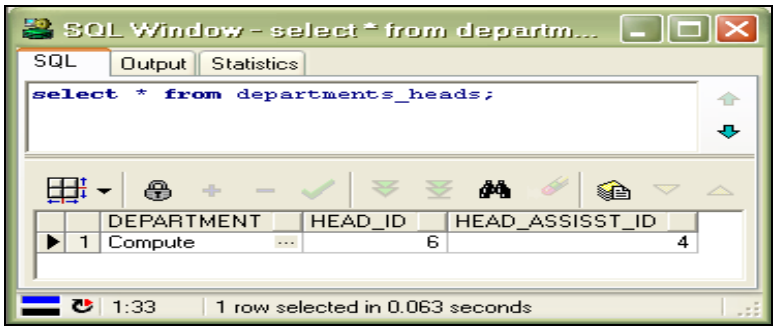


calling " course _insert" procedure.

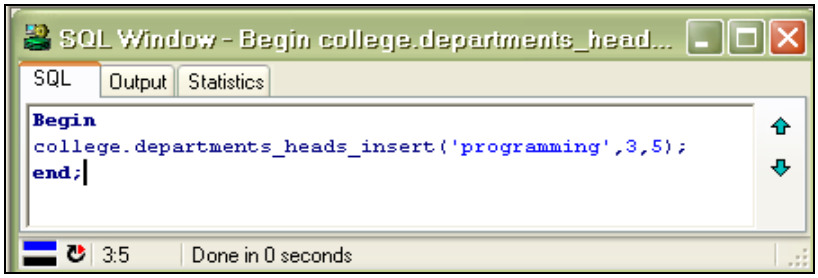


After calling " course _insert" procedure.

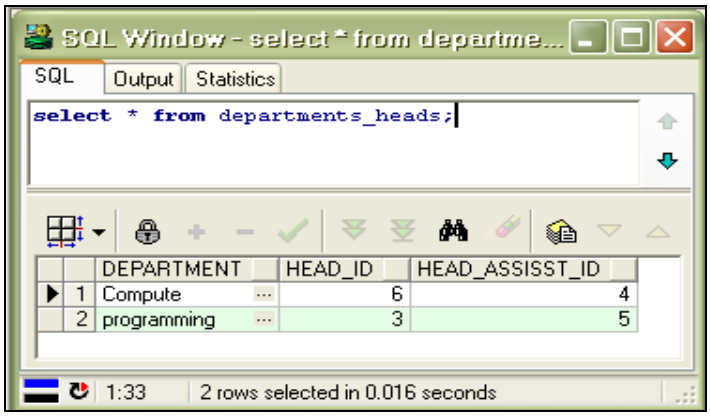
3. departments_heads_inser
t



Before calling "department_heads_insert" procedure.

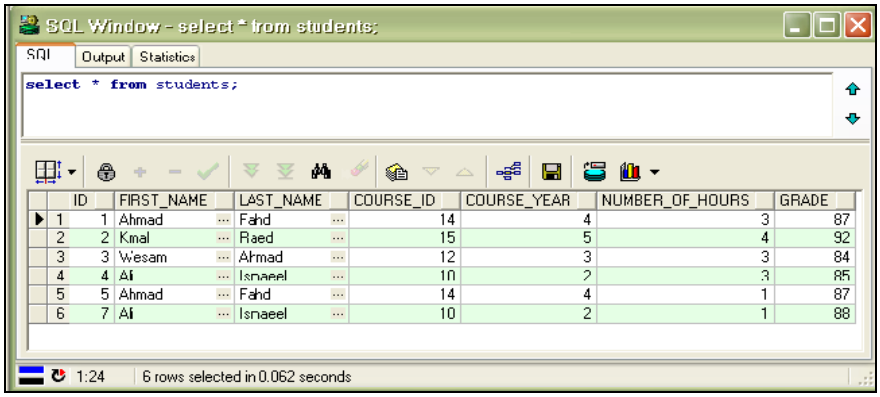


calling " department_heads _insert" procedure.

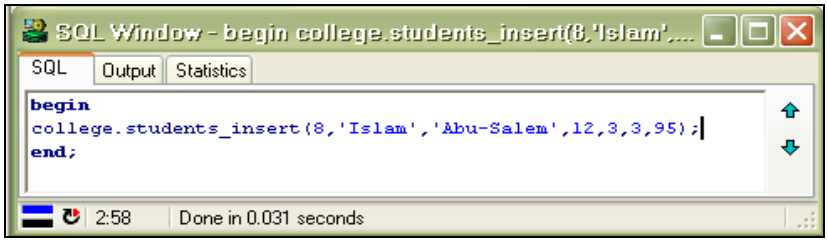


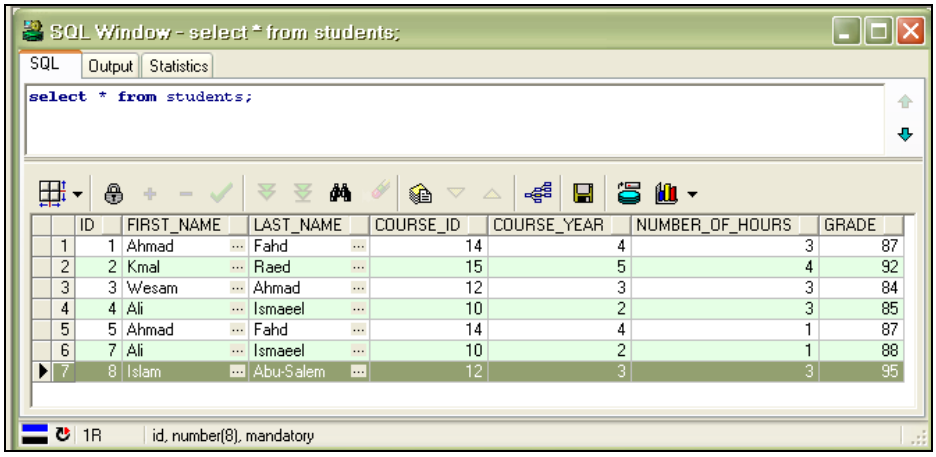
After calling " department_heads _insert" procedure.

4. students_insert



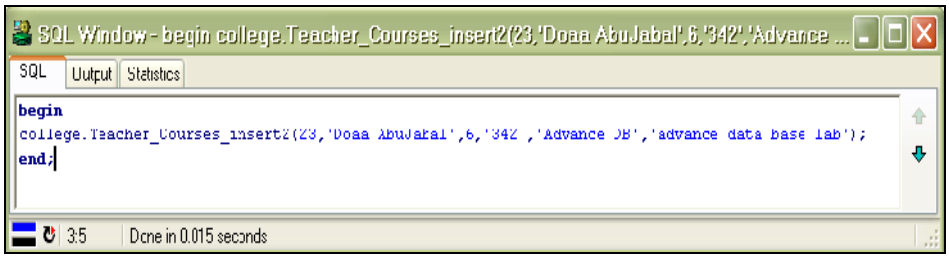
	ID	FIRST_NAME	LAST_NAME	COURSE_ID	COURSE_YEAR	NUMBER_OF_HOURS	GRADE
1	1	Ahmad	Fahd	14	4	3	87
2	2	Kmal	Raed	15	5	4	92
3	3	Wesam	Ahmad	12	3	3	84
4	4	Ali	Ismaeel	10	2	3	85
5	5	Ahmad	Fahd	14	4	1	87
6	7	Ali	Ismaeel	10	2	1	88

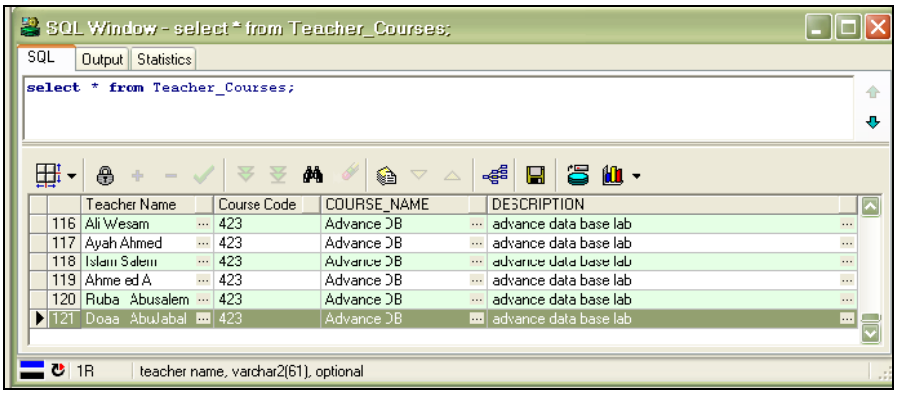




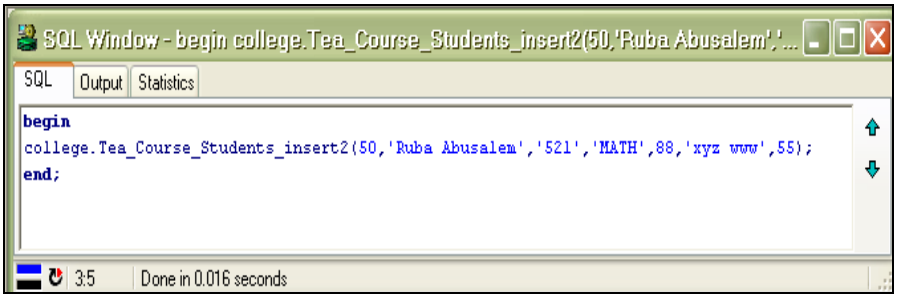
	ID	FIRST_NAME	LAST_NAME	COURSE_ID	COURSE_YEAR	NUMBER_OF_HOURS	GRADE
1	1	Ahmad	Fahd	14	4	3	87
2	2	Kmal	Raed	15	5	4	92
3	3	Wesam	Ahmad	12	3	3	84
4	4	Ali	Ismaeel	10	2	3	85
5	5	Ahmad	Fahd	14	4	1	87
6	7	Ali	Ismaeel	10	2	1	88
7	8	Islam	Abu-Salem	12	3	3	95

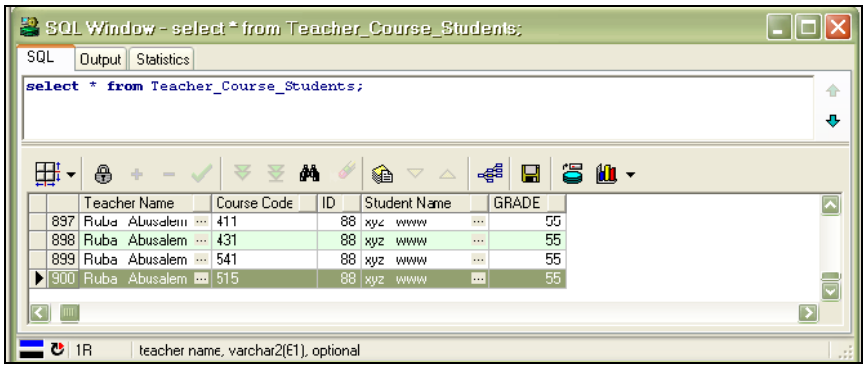
5. Teacher_Courses_insert2



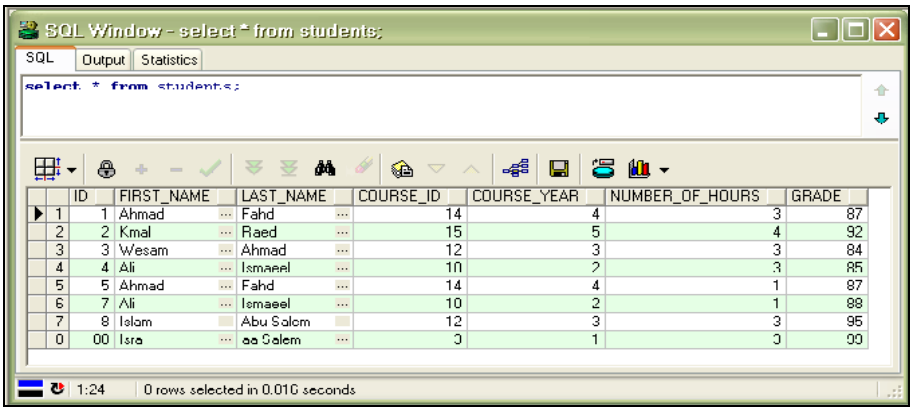


6. Tea_Course_Students_insert2

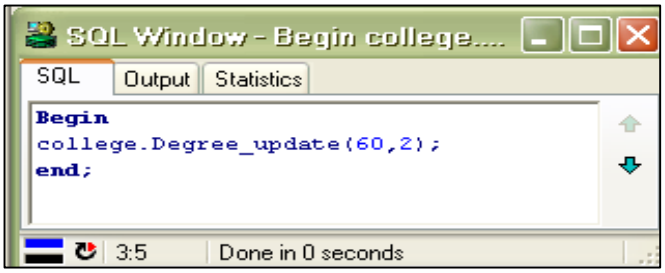


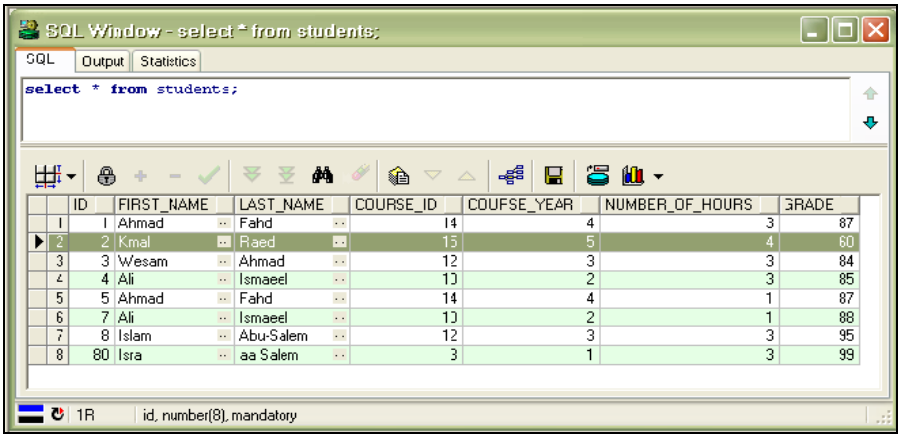


7. Degree_update



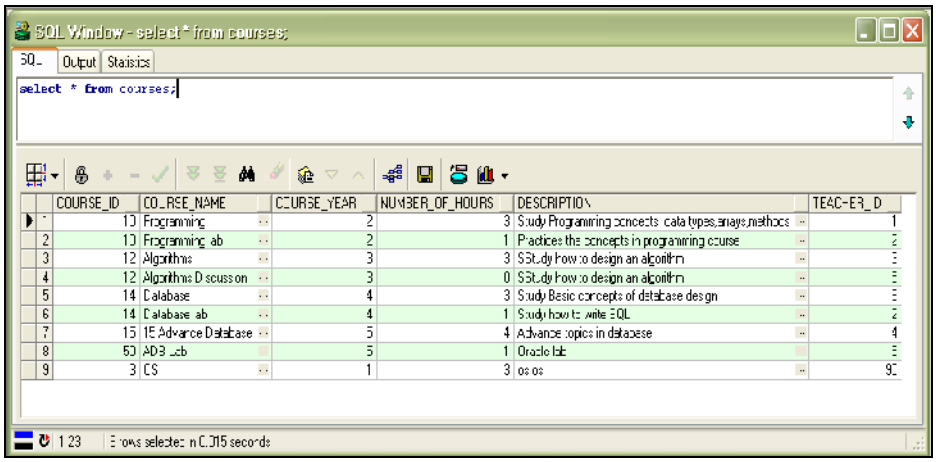
	ID	FIRST_NAME	LAST_NAME	COURSE_ID	COURSE_YEAR	NUMBER_OF_HOURS	GRADE
1	1	Ahmad	Fahd	14	4	3	87
2	2	Kmal	Raed	15	5	4	92
3	3	Wesam	Ahmad	12	3	3	84
4	4	Ali	Ismaeel	10	2	3	85
5	5	Ahmad	Fahd	14	4	1	87
6	7	Ali	Ismaeel	10	2	1	88
7	8	Islam	Abu Salem	12	3	3	95
0	00	Isra	aa Salem	3	1	3	99

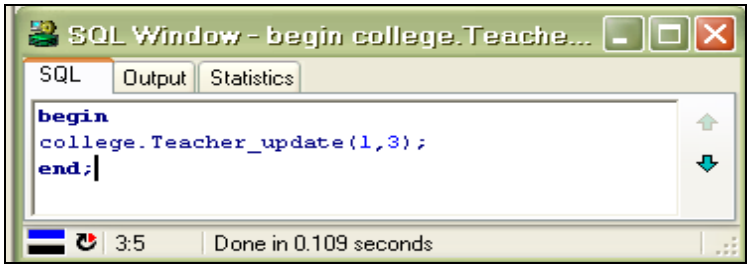


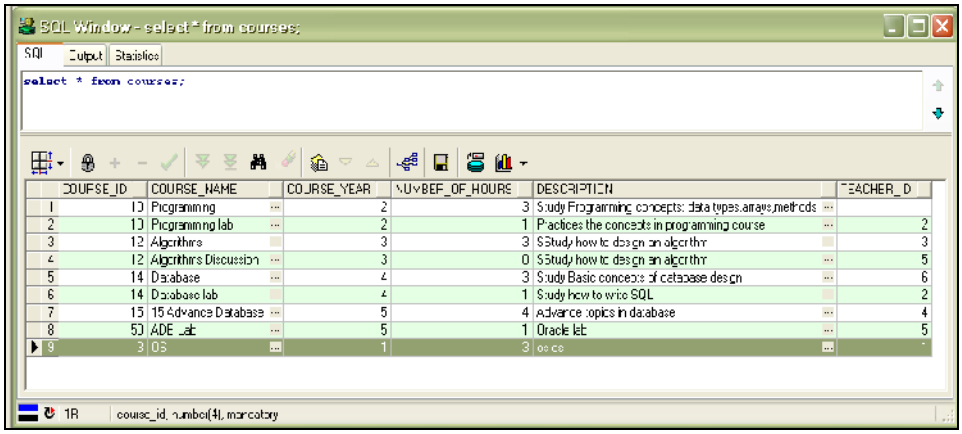


	ID	FIRST_NAME	LAST_NAME	COURSE_ID	COUFSE_YEAR	NUMBER_OF_HOURS	GRADE
1	1	Ahmad	Fahd	14	4	3	87
2	2	Kmal	Raed	15	5	4	60
3	3	Wesam	Ahmad	12	3	3	84
4	4	Ali	Ismaeel	10	2	3	85
5	5	Ahmad	Fahd	14	4	1	87
6	7	Ali	Ismaeel	10	2	1	88
7	8	Islam	Abu-Salem	12	3	3	95
8	80	Isra	aa Salem	3	1	3	99

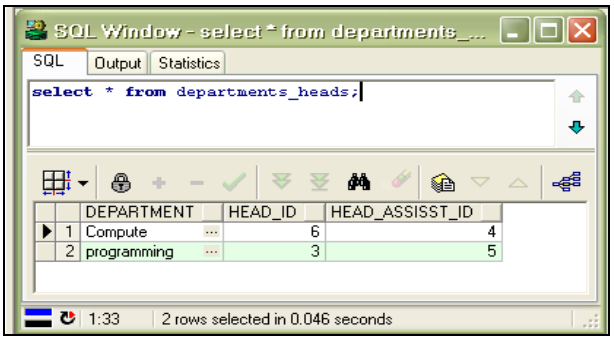
8. Teacher_update

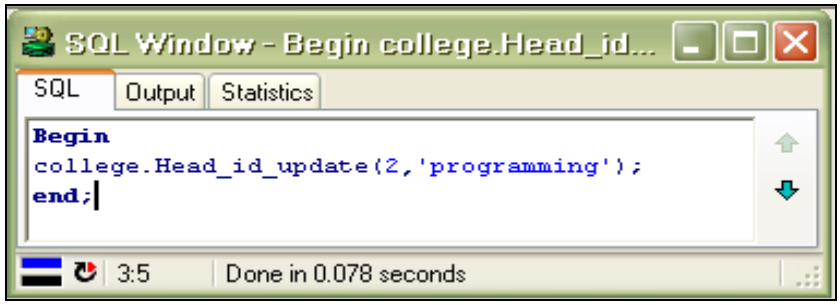


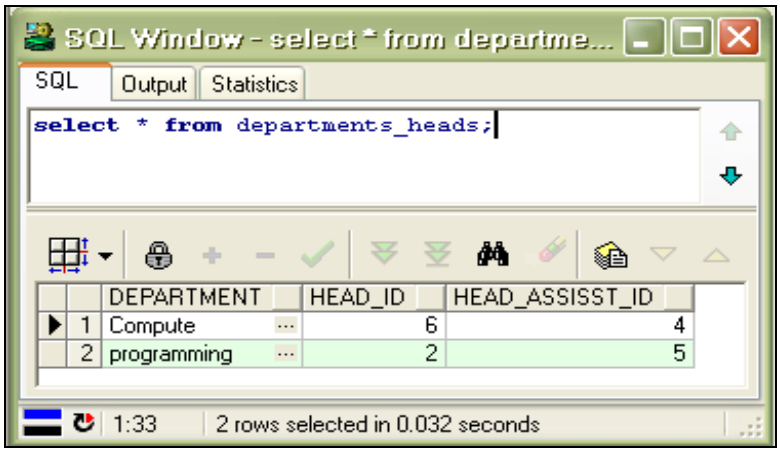




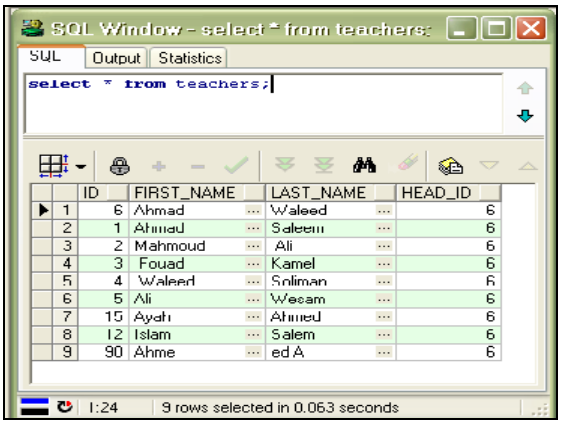
9. Head_id_update

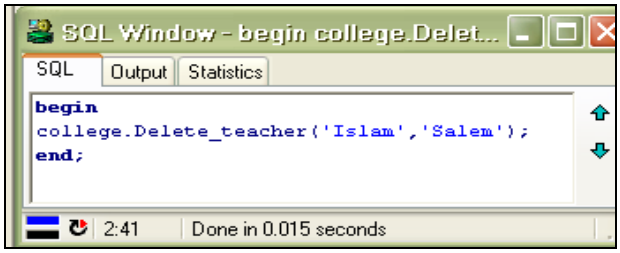


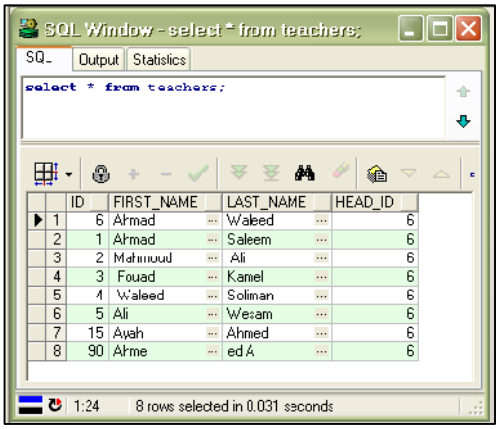




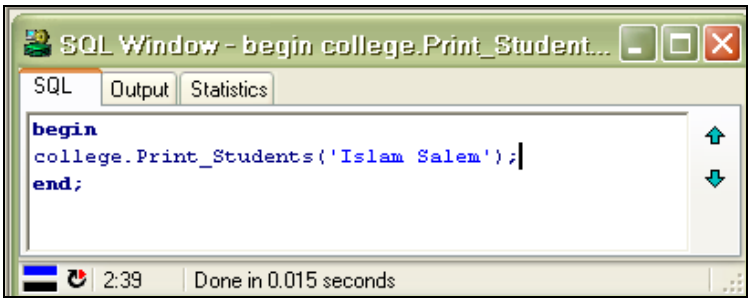
10. Delete_teacher

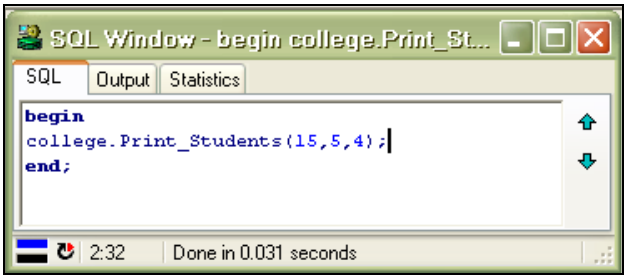




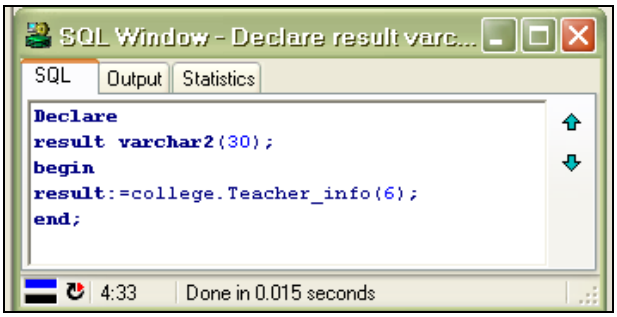


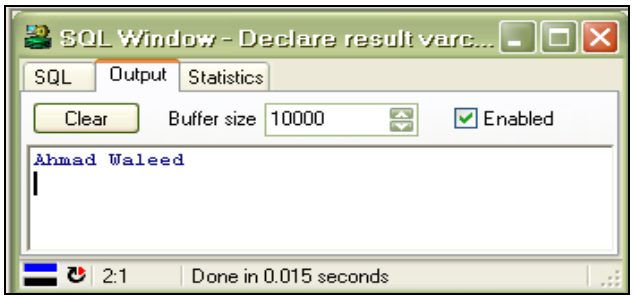
11. Print_Students



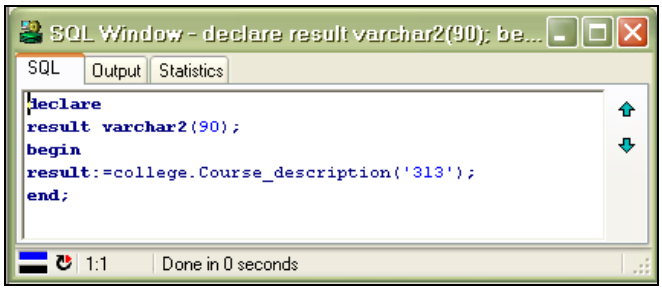


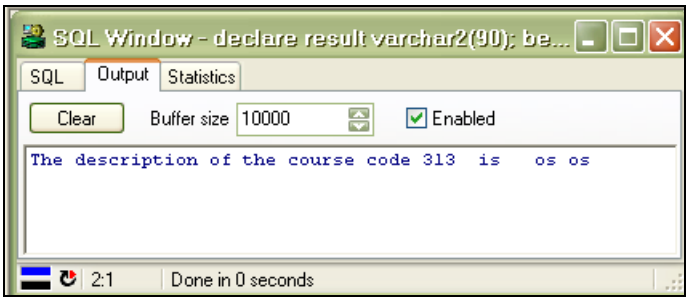
12. Teacher_info





13. Course_description



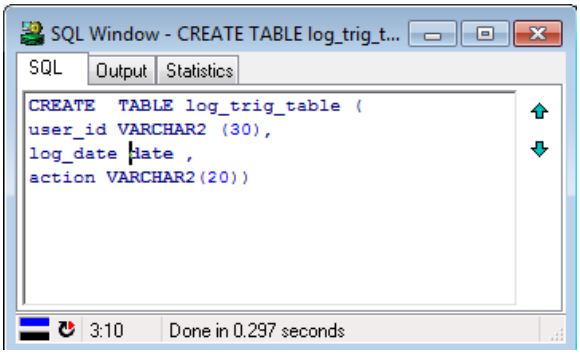


Triggers Part:

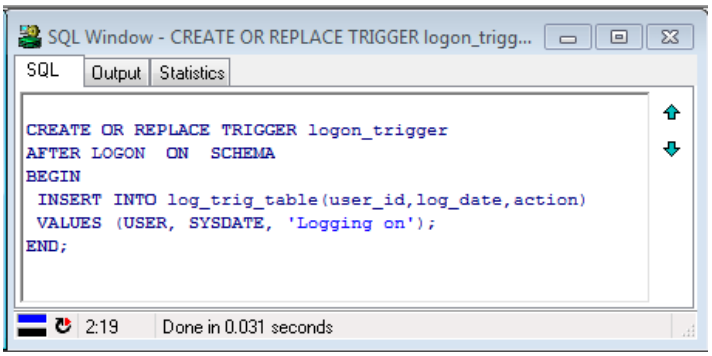
Logon and logoff triggers

Create logon_trigger that add "user_id, log_date and action(**'Logging on')**" on log_trig_table

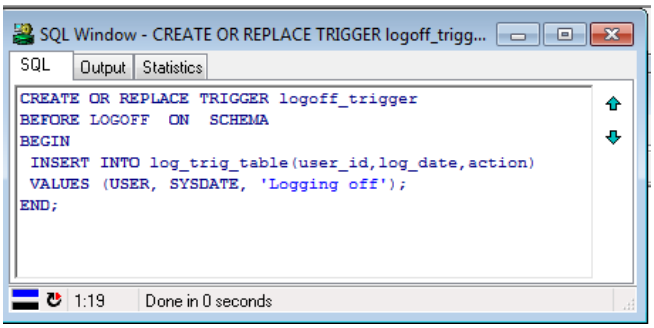
- First, we have to create log_trig_table



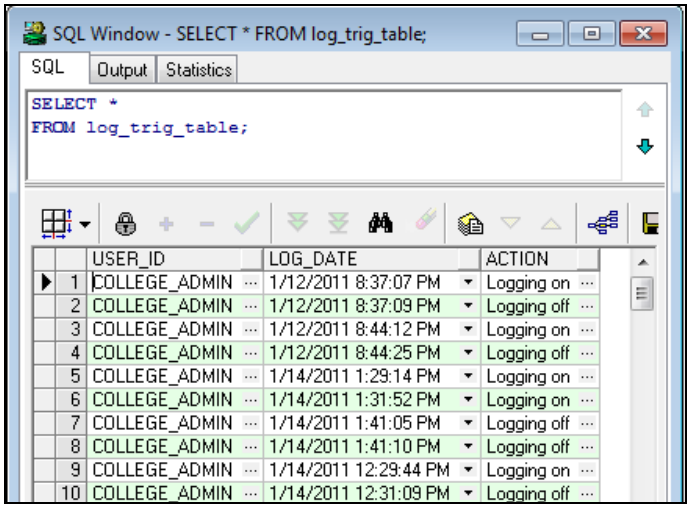
- **logon_trigger**



Create logoff_trigger that add "user_id, log_date and action(**'Logging off'**)" on log_trig_table .



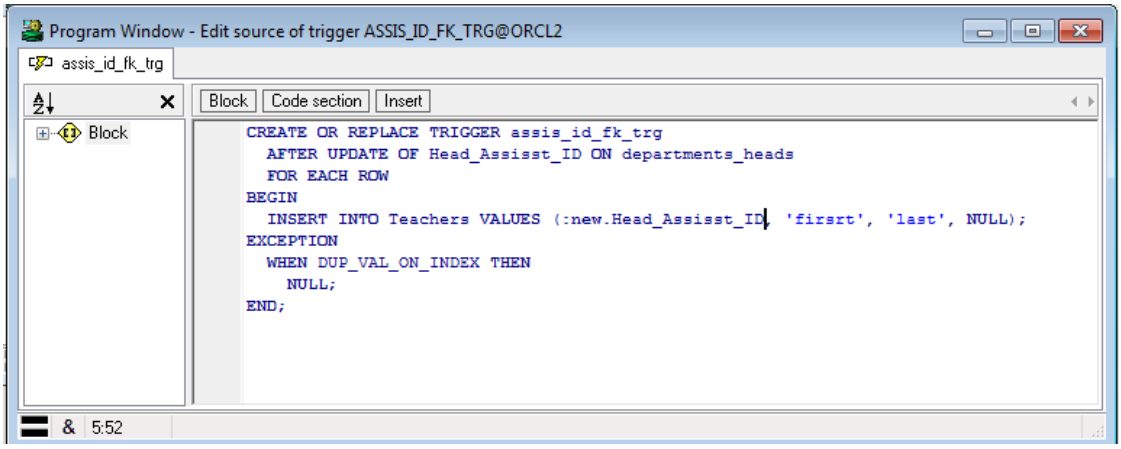
- The Result of the two previous triggers



	USER_ID	LOG_DATE	ACTION
1	COLLEGE_ADMIN	1/12/2011 8:37:07 PM	Logging on
2	COLLEGE_ADMIN	1/12/2011 8:37:09 PM	Logging off
3	COLLEGE_ADMIN	1/12/2011 8:44:12 PM	Logging on
4	COLLEGE_ADMIN	1/12/2011 8:44:25 PM	Logging off
5	COLLEGE_ADMIN	1/14/2011 1:29:14 PM	Logging on
6	COLLEGE_ADMIN	1/14/2011 1:31:52 PM	Logging on
7	COLLEGE_ADMIN	1/14/2011 1:41:05 PM	Logging off
8	COLLEGE_ADMIN	1/14/2011 1:41:10 PM	Logging off
9	COLLEGE_ADMIN	1/14/2011 12:29:44 PM	Logging on
10	COLLEGE_ADMIN	1/14/2011 12:31:09 PM	Logging off

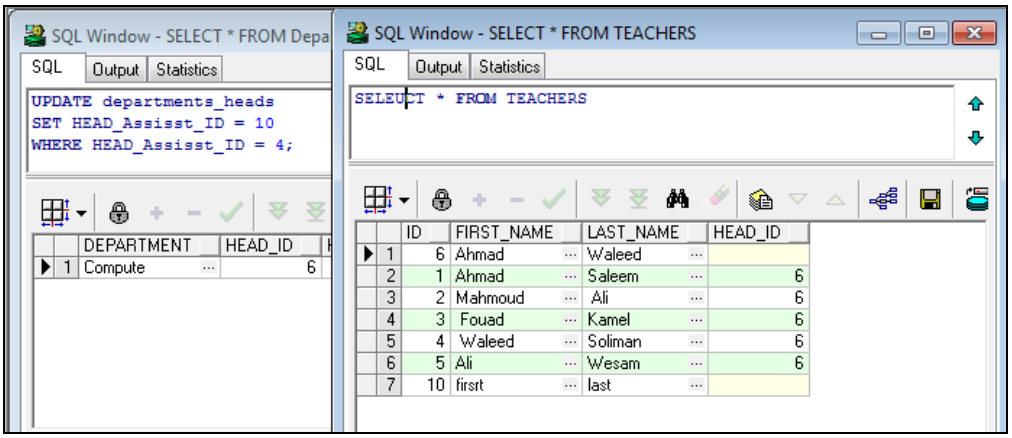
Foreign Keys Triggers

1- Head_Assisstant_ID in department_heads table is a foreign key for ID in the teacher table.

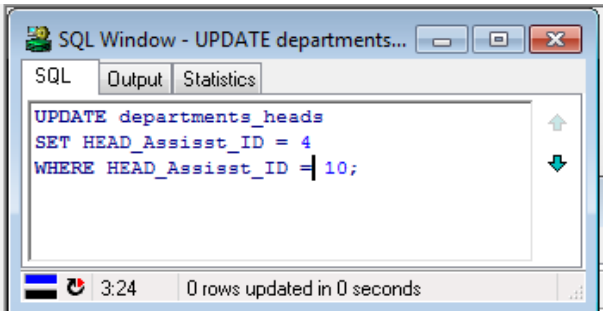


To test the trigger we will update the assisstant id to 10 which is not exsits in the teachers table, so the trigger wil be fired.

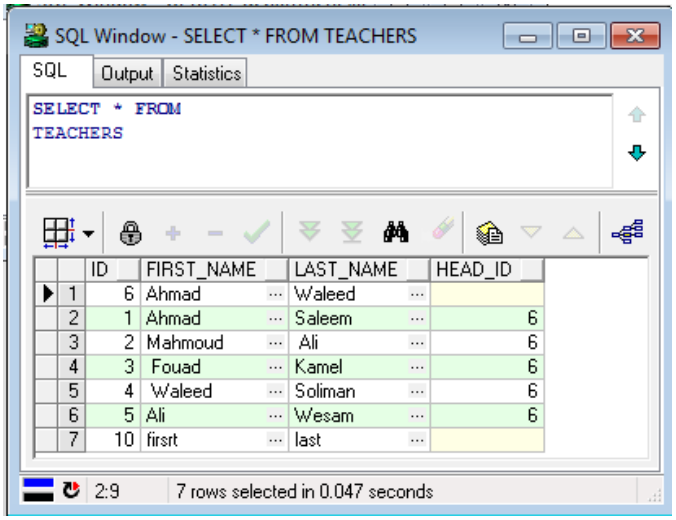
Here the teachers table after firing the trigger



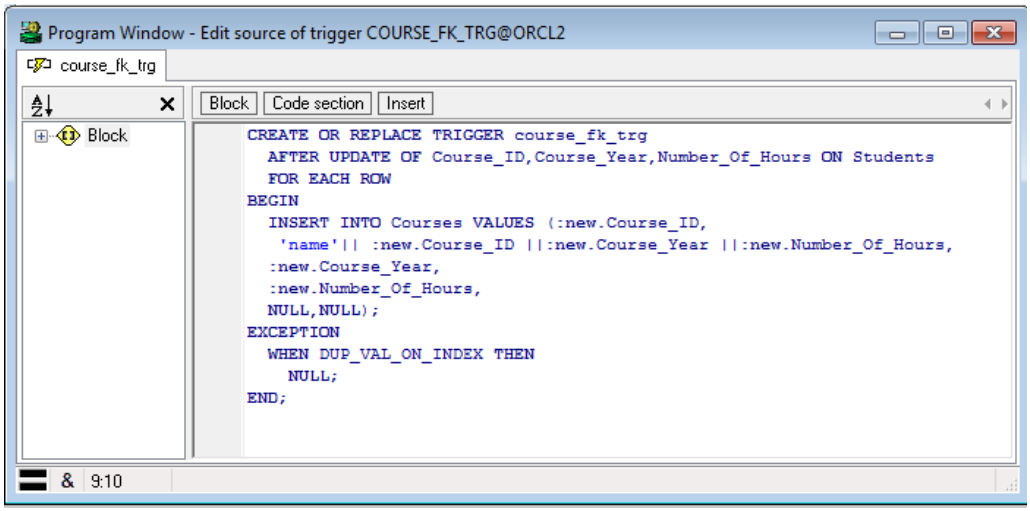
Here we will again update the assisstant id to 4 which is in the teachers id so the trigger do nothing (NULL)



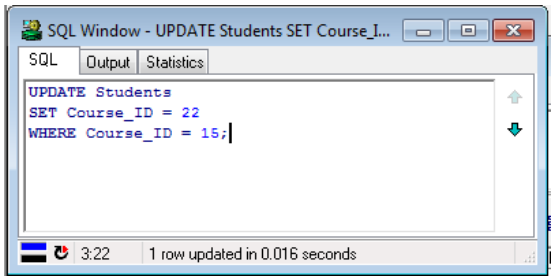
No change on the teachers table



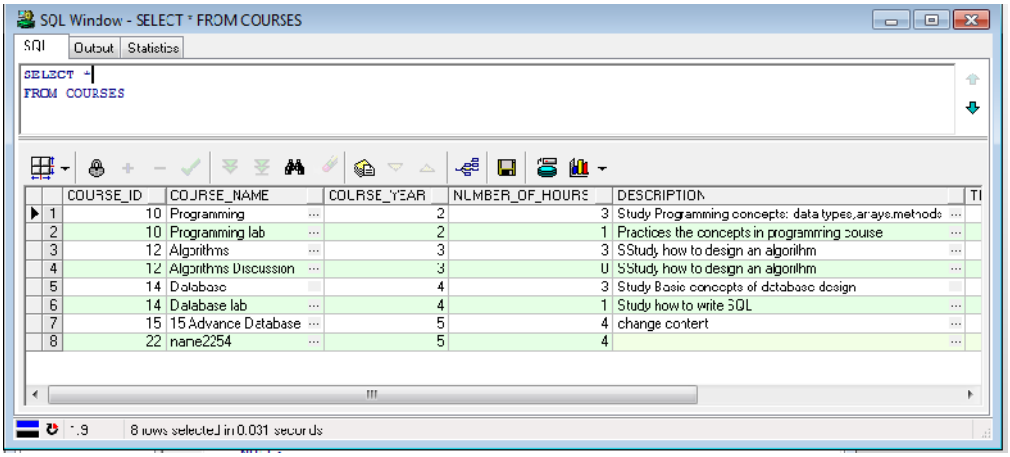
2- Trigger for Course_ID,Year_ID, Number_Of_Hours in Students table which are foreign keys for the same columns in Courses table.



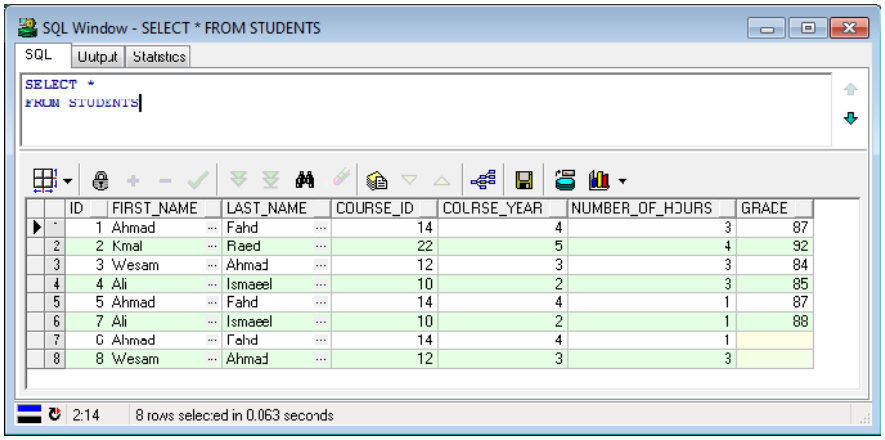
Here the query that will fire the trigger. We update the Course id in the Students table to 22 which is not in the Courses table



Here we can see that 22 was added to the courses table



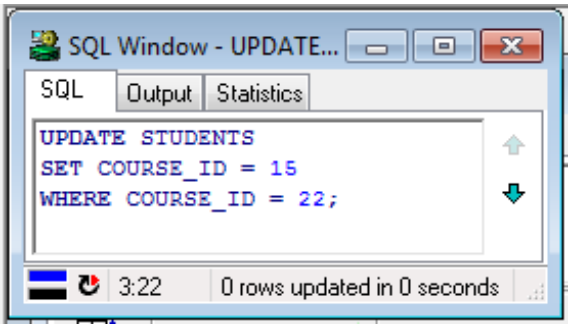
Here the Update in the Students table



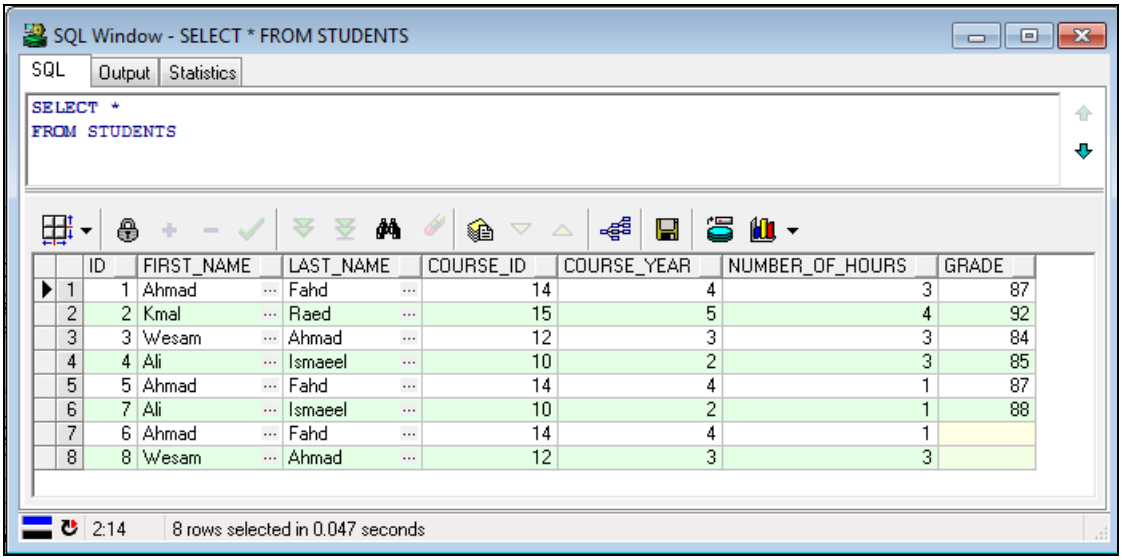
	ID	FIRST_NAME	LAST_NAME	COURSE_ID	COURSE_YEAR	NUMBER_OF_HOURS	GRADE
1	1	Ahmad	Fahd	14	4	3	87
2	2	Kmal	Raed	22	5	4	92
3	3	Wesam	Ahmad	12	3	3	84
4	4	Ali	Ismaeel	10	2	3	85
5	5	Ahmad	Fahd	14	4	1	87
6	7	Ali	Ismaeel	10	2	1	88
7	6	Ahmad	Fahd	14	4	1	
8	8	Wesam	Ahmad	12	3	3	

The following query will make the trigger to do nothing (NULL)

The update will not make any problem because the 15 is already in the Courses table

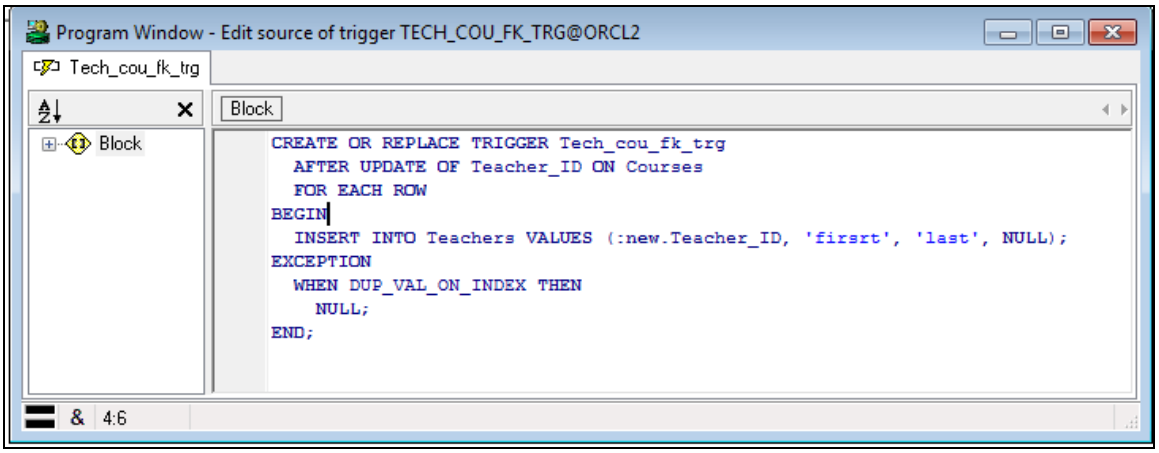


Here the resul of the update

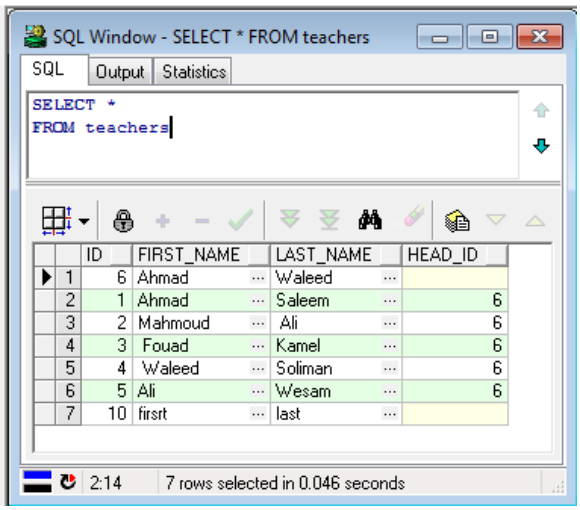


	ID	FIRST_NAME	LAST_NAME	COURSE_ID	COURSE_YEAR	NUMBER_OF_HOURS	GRADE
1	1	Ahmad	Fahd	14	4	3	87
2	2	Kmal	Raed	15	5	4	92
3	3	Wesam	Ahmad	12	3	3	84
4	4	Ali	Ismaeel	10	2	3	85
5	5	Ahmad	Fahd	14	4	1	87
6	7	Ali	Ismaeel	10	2	1	88
7	6	Ahmad	Fahd	14	4	1	
8	8	Wesam	Ahmad	12	3	3	

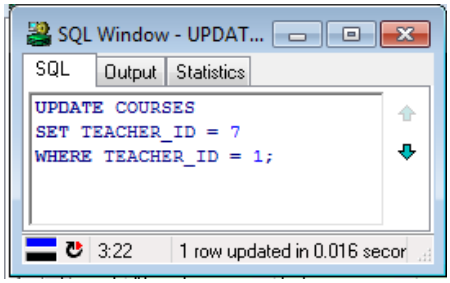
3- Trigger for the Teacher_ID in the Courses Table which references to the ID in the teachers table.



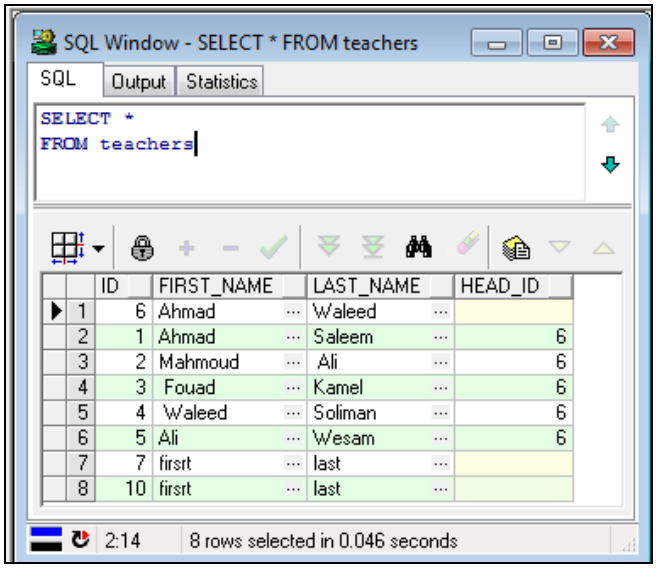
Here the teachers table before firing the trigger



Here the query that will fire the trigger

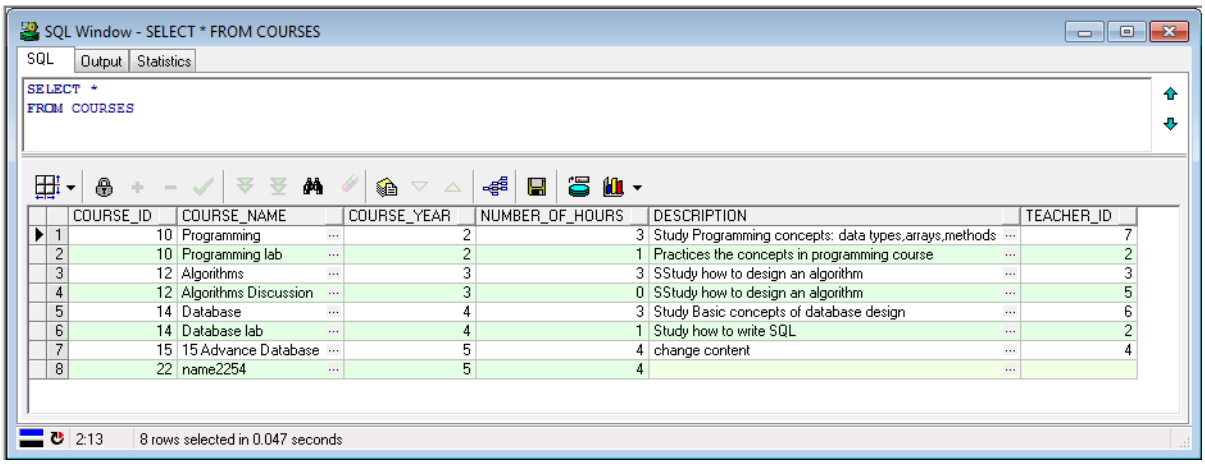


Here the teachers table after the trigger, we see tha ID = 7 was added to the table



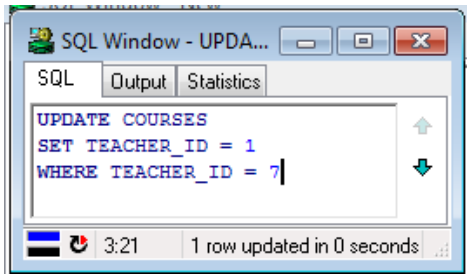
	ID	FIRST_NAME	LAST_NAME	HEAD_ID
1	6	Ahmad	Waleed	
2	1	Ahmad	Saleem	6
3	2	Mahmoud	Ali	6
4	3	Fouad	Kamel	6
5	4	Waleed	Soliman	6
6	5	Ali	Wesam	6
7	7	firsrt	last	
8	10	firsrt	last	

Here the update in the courses table

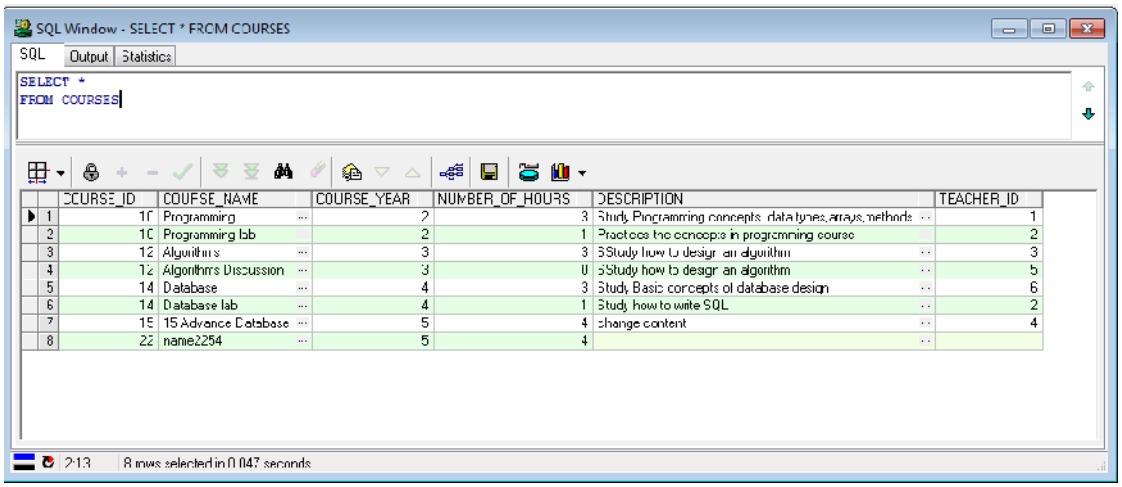


	COURSE_ID	COURSE_NAME	COURSE_YEAR	NUMBER_OF_HOURS	DESCRIPTION	TEACHER_ID
1	10	Programming	2	3	Study Programming concepts: data types,arrays,methods	7
2	10	Programming lab	2	1	Practices the concepts in programming course	2
3	12	Algorithms	3	3	SStudy how to design an algorithm	3
4	12	Algorithms Discussion	3	0	SStudy how to design an algorithm	5
5	14	Database	4	3	Study Basic concepts of database design	6
6	14	Database lab	4	1	Study how to write SQL	2
7	15	15 Advance Database	5	4	change content	4
8	22	name2254	5	4		

Here the update will cause nothing in the trigger and the update will be done without any change in the teachers table.

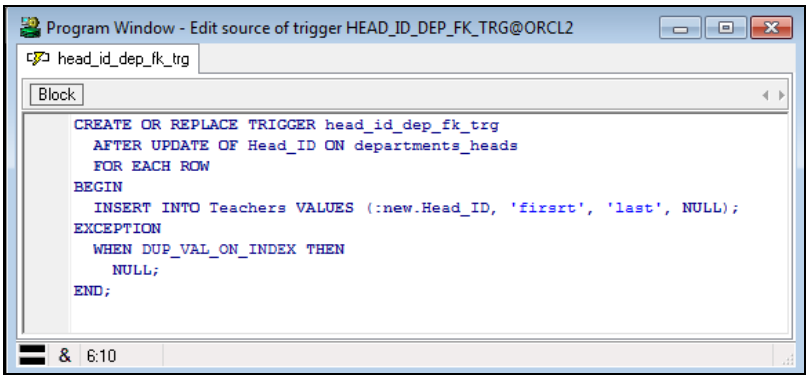


Here the courses table after update

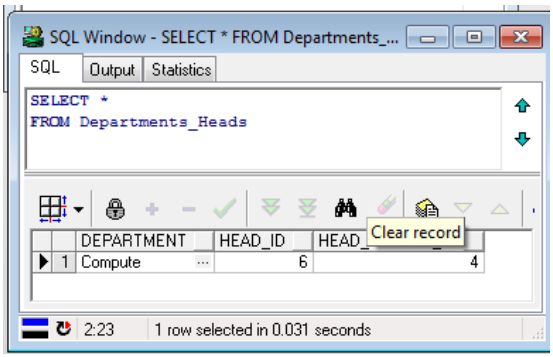


	COURSE_ID	COURSE_NAME	COURSE_YEAR	NUMBER_OF_HOURS	DESCRIPTION	TEACHER_ID
1	10	Programming	2	3	Study Programming concepts, data types, arrays, methods	1
2	10	Programming lab	2	1	Practices the concepts in programming course	2
3	12	Algorithms	3	3	3Study how to design an algorithm	3
4	12	Algorithms Discussion	3	0	3Study how to design an algorithm	5
5	14	Database	4	3	Study Basic concepts of database design	6
6	14	Database lab	4	1	Study how to write SQL	2
7	15	15 Advance Database	5	4	change content	4
8	22	name2254	5	4		

4- Triggers for the foreign key Head_ID in the departments_head table which references to the Head_ID in the teachers table

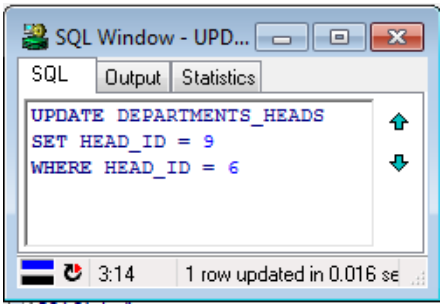


The departments_head table before the firing of the trigger

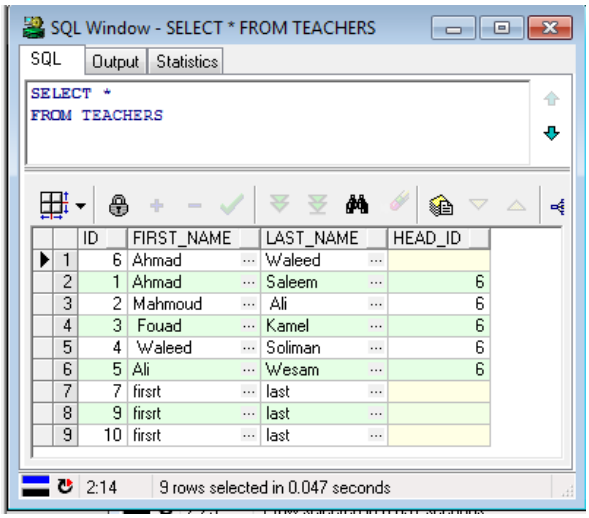


	DEPARTMENT	HEAD_ID	HEAD_
1	Compute	6	4

Now we will update the head_id to 9 which is not in the teachers table which cause firing the trigger that will add the teacher with id = 9 to the teachers table

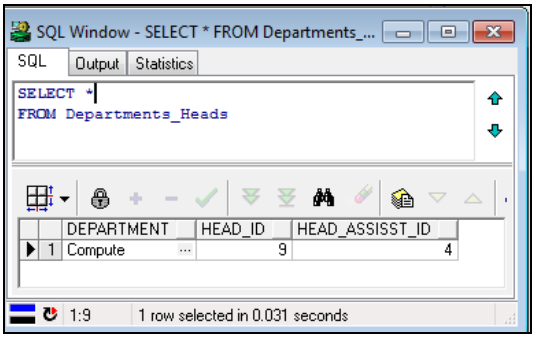


The result of firing the trigger on the teachers table (insertion of teacher with id = 9)



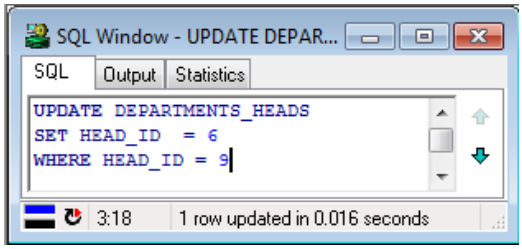
	ID	FIRST_NAME	LAST_NAME	HEAD_ID
1	6	Ahmad	Waleed	
2	1	Ahmad	Saleem	6
3	2	Mahmoud	Ali	6
4	3	Fouad	Kamel	6
5	4	Waleed	Soliman	6
6	5	Ali	Wesam	6
7	7	firsrt	last	
8	9	firsrt	last	
9	10	firsrt	last	

Here the result of updating the departments_head table

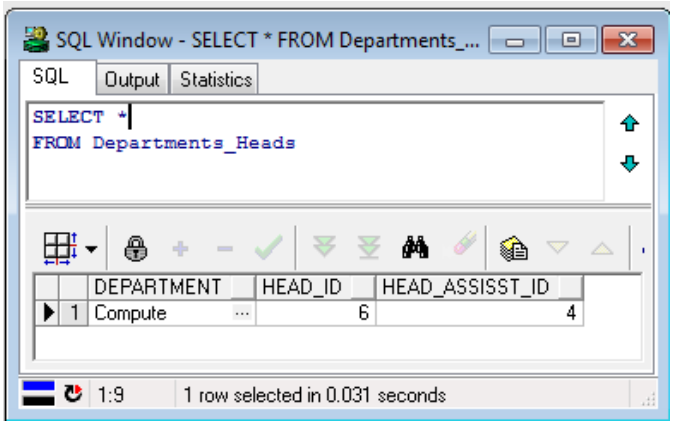


	DEPARTMENT	HEAD_ID	HEAD_ASSISST_ID
1	Compute	9	4

If we update the head_id with a value that already in the teachers table the trigger will make nothing in the teachers (NULL action)

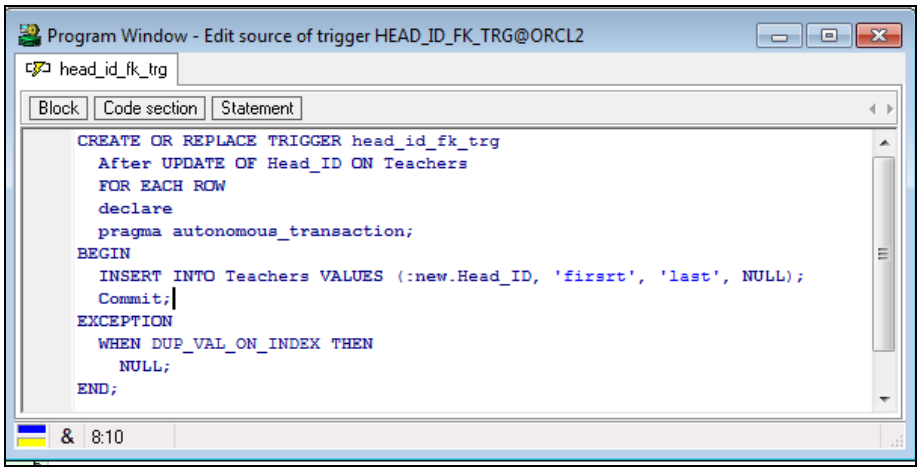


The update in the departments_head table



5- Trigger for the Head_id in the teachers table which is foreign key for the ID in the same table.

Here we will face a mutating problem which can be solved as the following



Here a brief explaination

Pragma Autonomous transaction:-

Whenever we commit all the transactions before the last commit in that session get saved. Suppose that you have a scenario you just want a particular transaction only to be save. In order to achieve that you make this as a separate transaction which can be committed but prior to this the transactions are not commit.

eg:

insert into tab1 () values ();

update tab2 set values.. some more stmts..

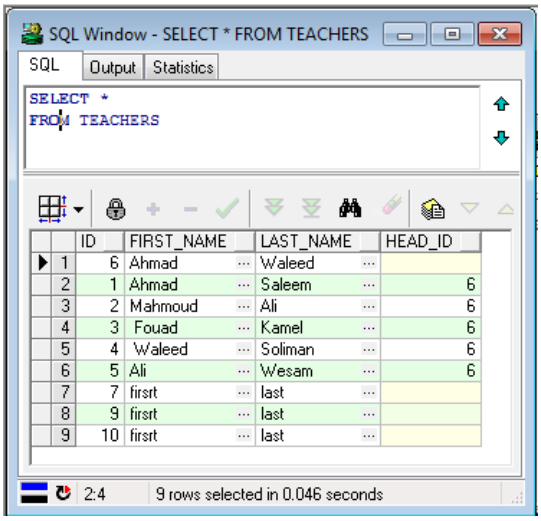
pragma autonomous transaction

insert into tab3 values();

commit;

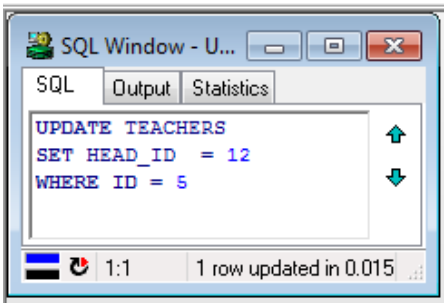
here.. insert into tab1 is only saved but not tab1 and tab2.

Here the teachers table before the trigger

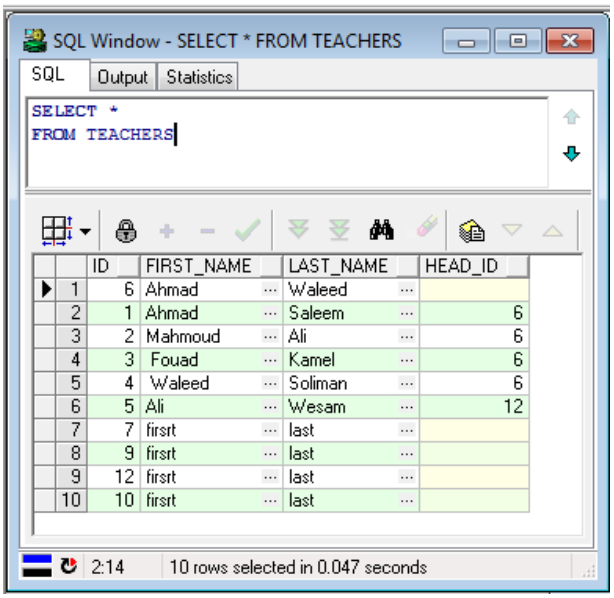


	ID	FIRST_NAME	LAST_NAME	HEAD_ID
1	6	Ahmad	Waleed	
2	1	Ahmad	Saleem	6
3	2	Mahmoud	Ali	6
4	3	Fouad	Kamel	6
5	4	Waleed	Soliman	6
6	5	Ali	Wesam	6
7	7	firsrt	last	
8	9	firsrt	last	
9	10	firsrt	last	

Here the query that will fire the trigger because the id = 12 is not in the teachers table



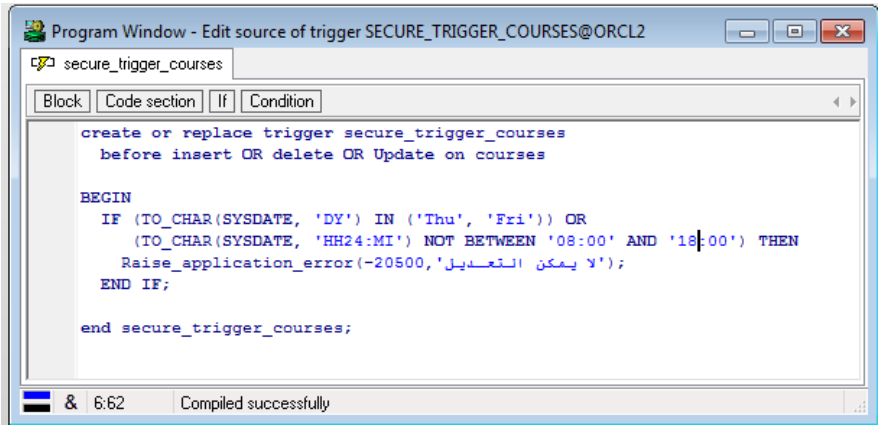
Here the table after firing the table and inserting the teacher with id = 12



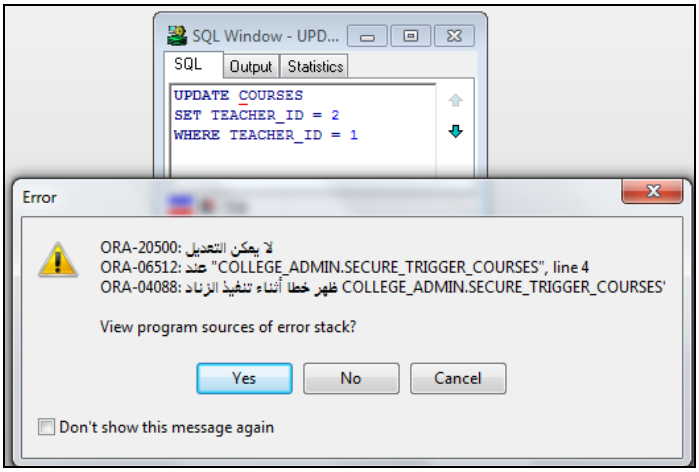
	ID	FIRST_NAME	LAST_NAME	HEAD_ID
1	6	Ahmad	Waleed	
2	1	Ahmad	Saleem	6
3	2	Mahmoud	Ali	6
4	3	Fouad	Kamel	6
5	4	Waleed	Soliman	6
6	5	Ali	Wesam	12
7	7	firsrt	last	
8	9	firsrt	last	
9	12	firsrt	last	
10	10	firsrt	last	

Security triggers

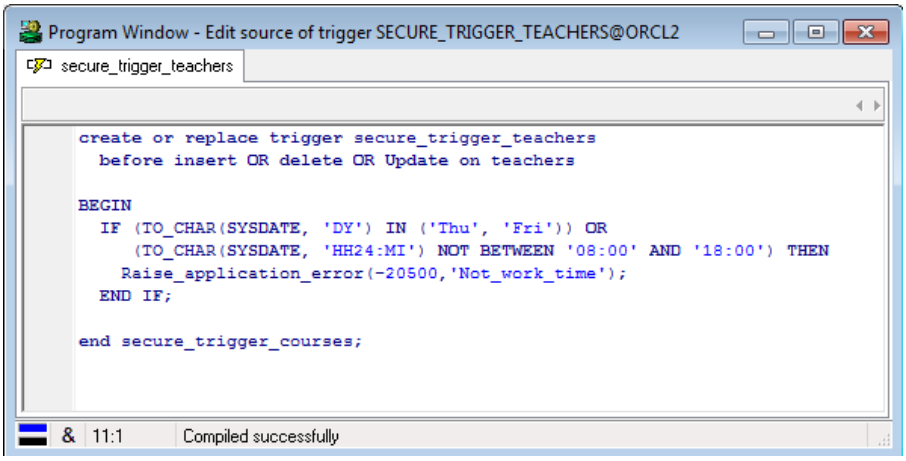
1- Trigger on the Courses table in order not to update or delete or insert on the table after the end of the work times



Here the test of the trigger



2- Trigger on the Teachers table in order not to update or delete or insert on the table after the end of the work times



Here the result of the test of this trigger