BIT358: Advance Database

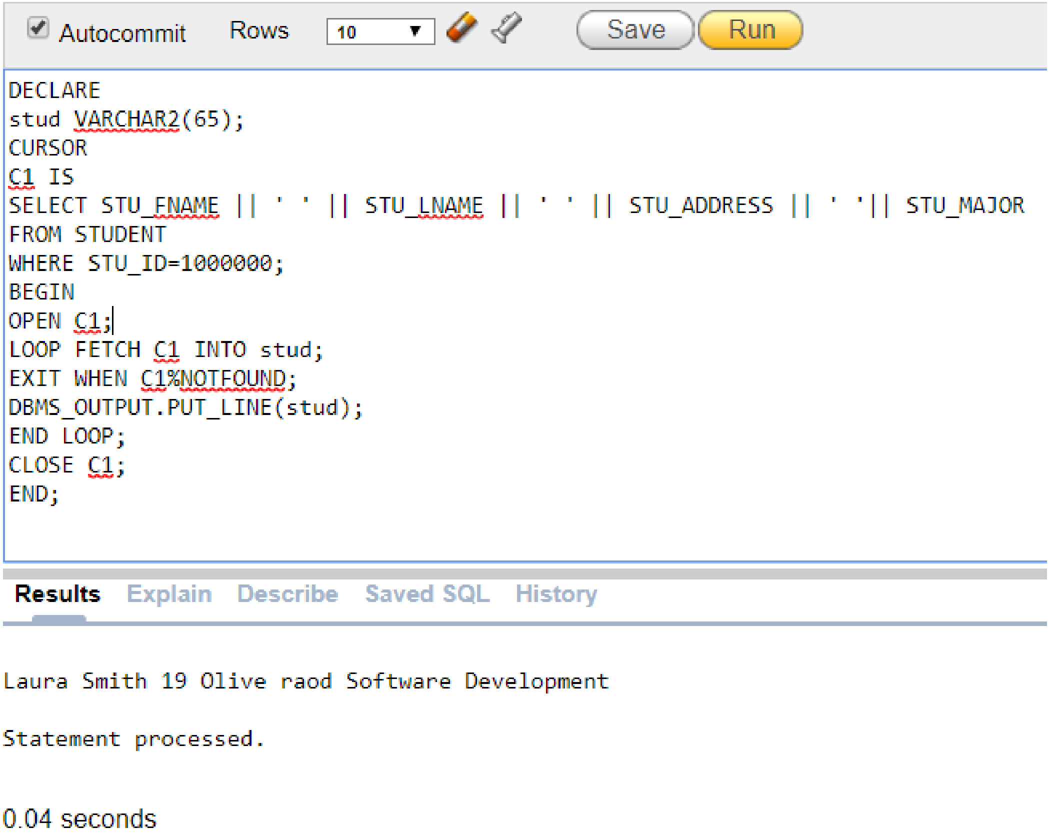
Assessment 3 – Database Application

S1496027 - Harindu

Part B

1.

```
DECLARE
stud VARCHAR2(65);
CURSOR
C1 IS
SELECT STU_FNAME || STU_LNAME || STU_ADDRESS || STU_MAJOR
FROM STUDENT
WHERE STU_ID=1000001;
BEGIN
OPEN C1;
LOOP FETCH C1 INTO stud;
EXIT WHEN C1%NOTFOUND;
DBMS_OUTPUT.PUT_LINE(stud);
END LOOP;
CLOSE C1;
END;
```



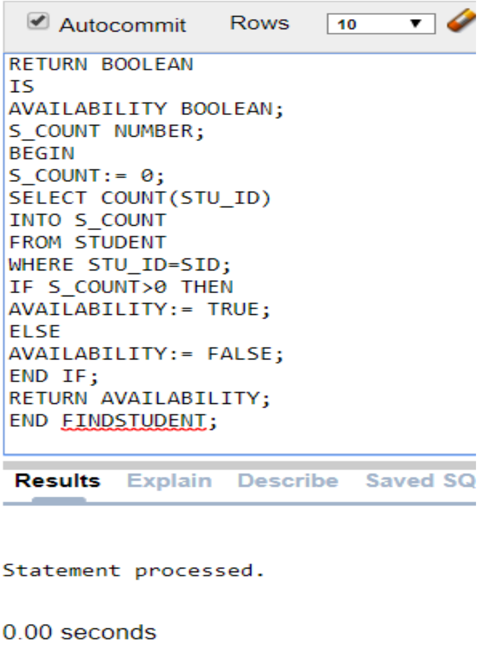
```
DECLARE
stud VARCHAR2(65);
CURSOR
C1 IS
SELECT STU_FNAME || ' ' || STU_LNAME || ' ' || STU_ADDRESS || ' '|| STU_MAJOR
FROM STUDENT
WHERE STU_ID=1000000;
BEGIN
OPEN C1;
LOOP FETCH C1 INTO stud;
EXIT WHEN C1%NOTFOUND;
DBMS_OUTPUT.PUT_LINE(stud);
END LOOP;
CLOSE C1;
END;
```

```
Laura Smith 19 Olive raod Software Development

Statement processed.
```

0.04 seconds

2)

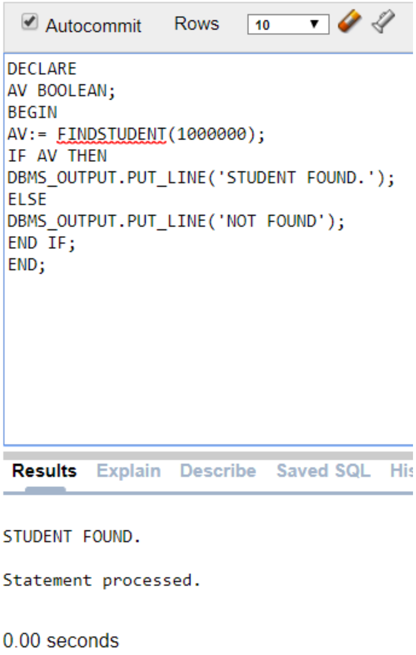
```
CREATE OR REPLACE
FUNCTION FINDSTUDENT(SID IN NUMBER)
RETURN BOOLEAN
IS
AVAILABILITY BOOLEAN;
S_COUNT NUMBER;
BEGIN
S_COUNT:= 0;
SELECT COUNT(STU_ID)
INTO S_COUNT
FROM STUDENT
WHERE STU_ID=SID;
IF S_COUNT>0 THEN
AVAILABILITY:= TRUE;
ELSE
AVAILABILITY:= FALSE;
END IF;
RETURN AVAILABILITY;
END FINDSTUDENT;
```



```
RETURN BOOLEAN
IS
AVAILABILITY BOOLEAN;
S_COUNT NUMBER;
BEGIN
S_COUNT:= 0;
SELECT COUNT(STU_ID)
INTO S_COUNT
FROM STUDENT
WHERE STU_ID=SID;
IF S_COUNT>0 THEN
AVAILABILITY:= TRUE;
ELSE
AVAILABILITY:= FALSE;
END IF;
RETURN AVAILABILITY;
END FINDSTUDENT;
```

```
Statement processed.

0.00 seconds
```

```
DECLARE
AV BOOLEAN;
BEGIN
AV:= FINDSTUDENT(1000000);
IF AV THEN
DBMS_OUTPUT.PUT_LINE('STUDENT FOUND.');
ELSE
DBMS_OUTPUT.PUT_LINE('NOT FOUND');
END IF;
END;
```



```
DECLARE
AV BOOLEAN;
BEGIN
AV:= FINDSTUDENT(1000000);
IF AV THEN
DBMS_OUTPUT.PUT_LINE('STUDENT FOUND.');
ELSE
DBMS_OUTPUT.PUT_LINE('NOT FOUND');
END IF;
END;
```

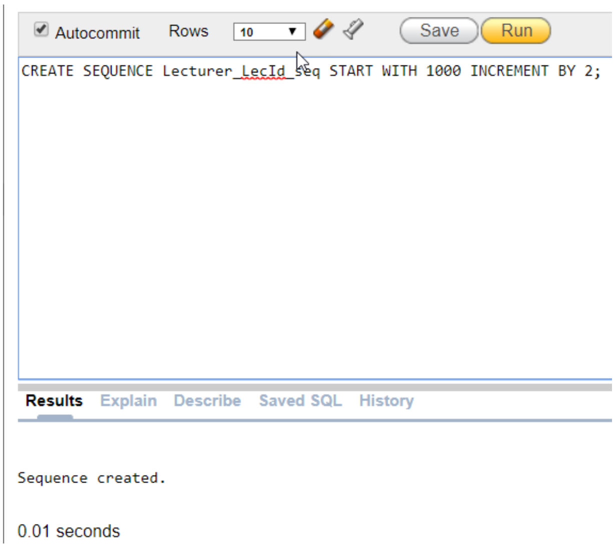
```
STUDENT FOUND.

Statement processed.

0.00 seconds
```

3)

4)

Step 1

CREATE SEQUENCE Lecturer_LecId_seq START WITH 1000 INCREMENT BY 2;



☑ Autocommit Rows 10

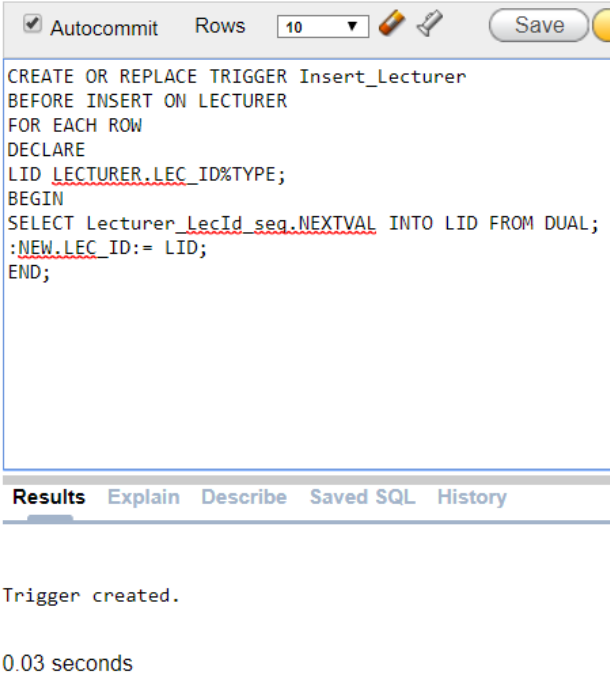
```
CREATE SEQUENCE Lecturer_LecId_seq START WITH 1000 INCREMENT BY 2;
```

```
Sequence created.
```

0.01 seconds

Step 2

CREATE OR REPLACE TRIGGER Insert_Lecturer
BEFORE INSERT ON LECTURER
FOR EACH ROW
DECLARE
LID LECTURER.LEC_ID%TYPE;
BEGIN
SELECT Lecturer_LecId_seq.NEXTVAL INTO LID FROM DUAL;
:NEW.LEC_ID:= LID;
END;



☑ Autocommit Rows 10

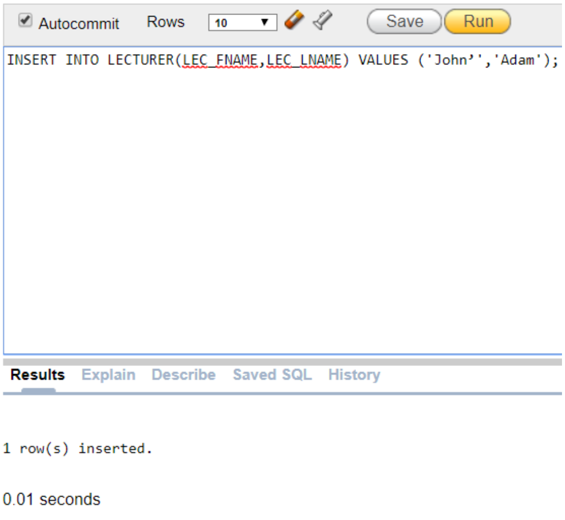
```
CREATE OR REPLACE TRIGGER Insert_Lecturer
BEFORE INSERT ON LECTURER
FOR EACH ROW
DECLARE
LID LECTURER.LEC_ID%TYPE;
BEGIN
SELECT Lecturer_LecId_seq.NEXTVAL INTO LID FROM DUAL;
:NEW.LEC_ID:= LID;
END;
```

```
Trigger created.
```

0.03 seconds

Step 3

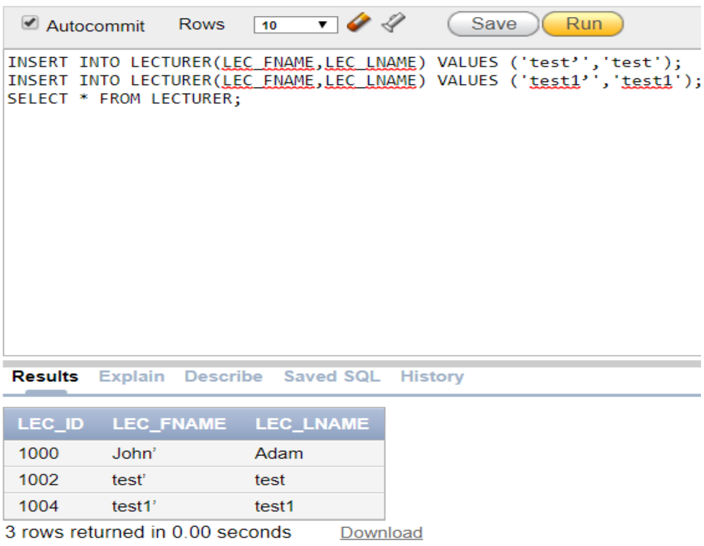
INSERT INTO LECTURER(LEC_FNAME,LEC_LNAME) VALUES ('John’','Adam');
SELECT * FROM LECTURER;



```
INSERT INTO LECTURER(LEC_FNAME,LEC_LNAME) VALUES ('John’','Adam');
```

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

0.01 seconds



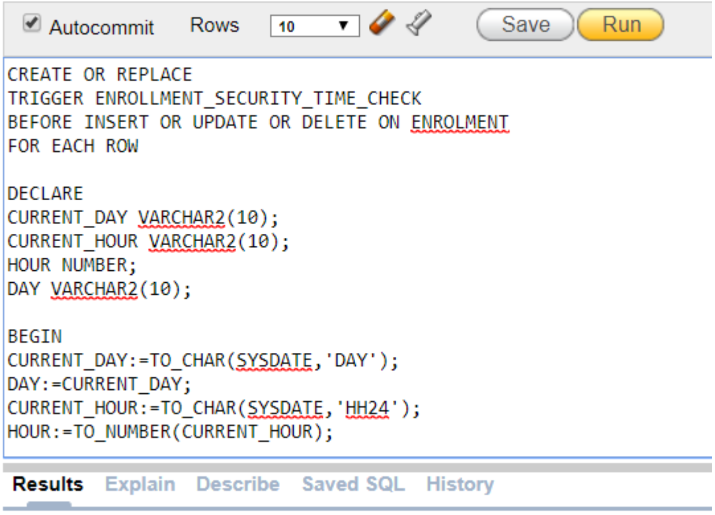
```
INSERT INTO LECTURER(LEC_FNAME,LEC_LNAME) VALUES ('test’','test');
INSERT INTO LECTURER(LEC_FNAME,LEC_LNAME) VALUES ('test1’','test1');
SELECT * FROM LECTURER;
```

LEC_ID	LEC_FNAME	LEC_LNAME
1000	John’	Adam
1002	test’	test
1004	test1’	test1

3 rows returned in 0.00 seconds

5)

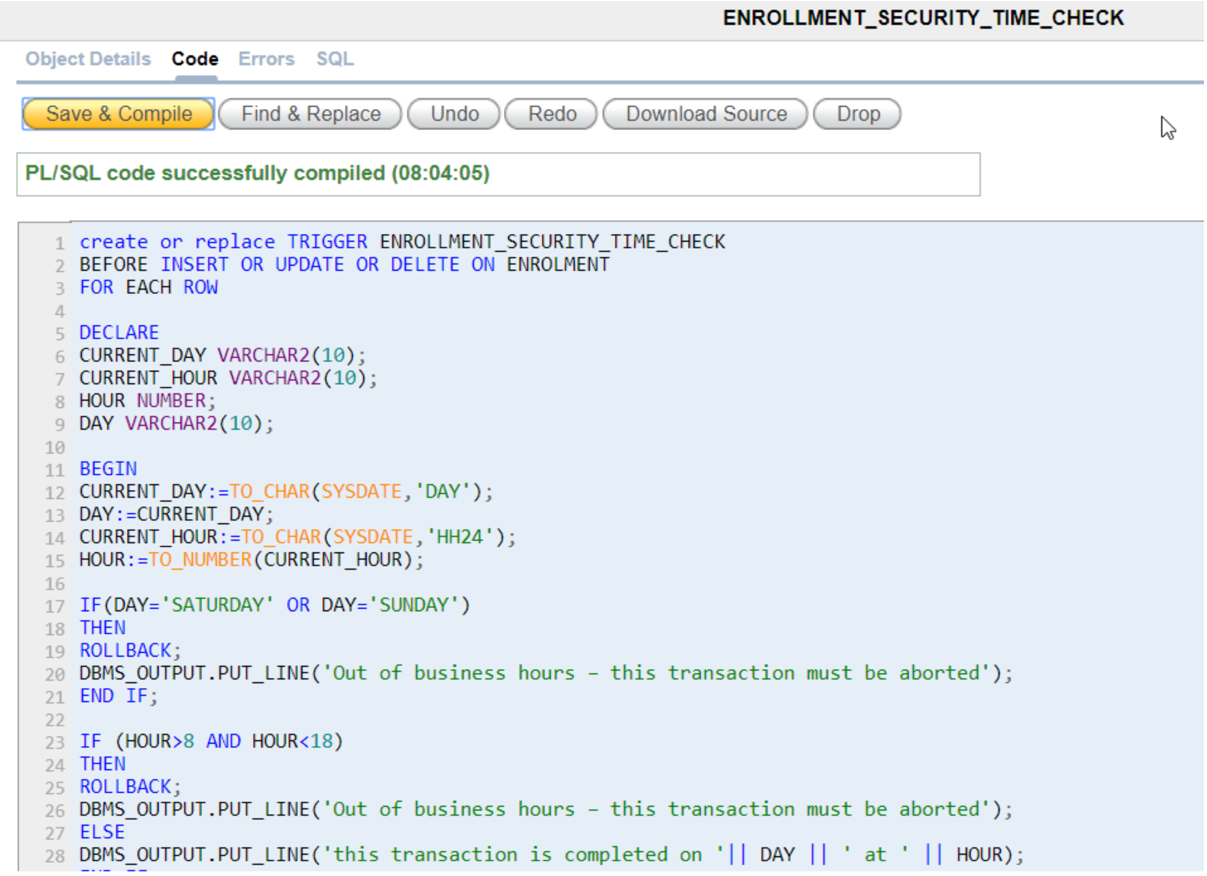
```
create or replace
TRIGGER ENROLLMENT_SECURITY_TIME_CHECK
BEFORE INSERT OR UPDATE OR DELETE ON ENROLMENT
FOR EACH ROW

DECLARE
CURRENT_DAY VARCHAR2(10);
CURRENT_HOUR VARCHAR2(10);
HOUR NUMBER;
DAY VARCHAR2(10);

BEGIN
CURRENT_DAY:=TO_CHAR(SYSDATE,'DAY');
DAY:=CURRENT_DAY;
CURRENT_HOUR:=TO_CHAR(SYSDATE,'HH24');
HOUR:=TO_NUMBER(CURRENT_HOUR);

IF(DAY='SATURDAY' OR DAY='SUNDAY')
THEN
ROLLBACK;
DBMS_OUTPUT.PUT_LINE('Out of business hours – this transaction must be aborted');
END IF;

IF (HOUR>8 AND HOUR<18)
THEN
ROLLBACK;
DBMS_OUTPUT.PUT_LINE('Out of business hours – this transaction must be aborted');
ELSE
DBMS_OUTPUT.PUT_LINE('this transaction is completed on '|| DAY || ' at ' || HOUR);
END IF;
END;
```



```
CREATE OR REPLACE
TRIGGER ENROLLMENT_SECURITY_TIME_CHECK
BEFORE INSERT OR UPDATE OR DELETE ON ENROLMENT
FOR EACH ROW

DECLARE
CURRENT_DAY VARCHAR2(10);
CURRENT_HOUR VARCHAR2(10);
HOUR NUMBER;
DAY VARCHAR2(10);

BEGIN
CURRENT_DAY:=TO_CHAR(SYSDATE,'DAY');
DAY:=CURRENT_DAY;
CURRENT_HOUR:=TO_CHAR(SYSDATE,'HH24');
HOUR:=TO_NUMBER(CURRENT_HOUR);
```

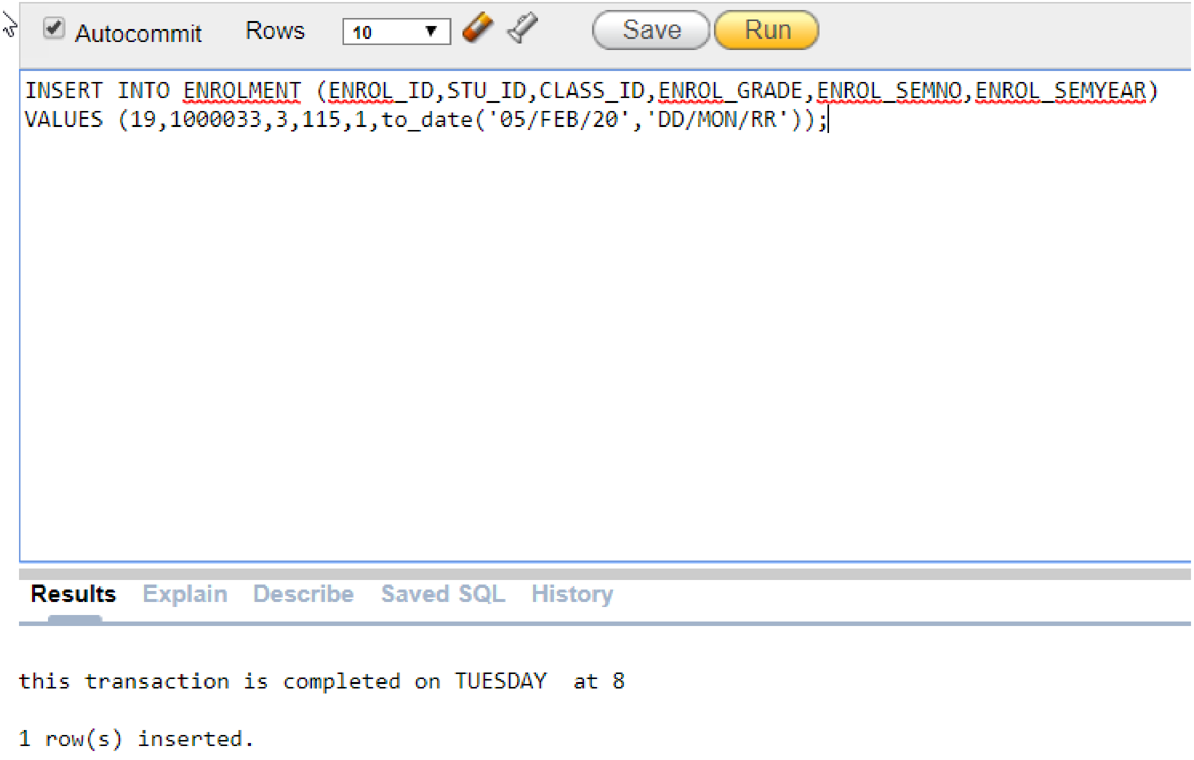
```
Statement processed.
```

0.00 seconds



PL/SQL code successfully compiled (08:04:05)

```
create or replace TRIGGER ENROLLMENT_SECURITY_TIME_CHECK
BEFORE INSERT OR UPDATE OR DELETE ON ENROLMENT
FOR EACH ROW

DECLARE
CURRENT_DAY VARCHAR2(10);
CURRENT_HOUR VARCHAR2(10);
HOUR NUMBER;
DAY VARCHAR2(10);

BEGIN
CURRENT_DAY:=TO_CHAR(SYSDATE,'DAY');
DAY:=CURRENT_DAY;
CURRENT_HOUR:=TO_CHAR(SYSDATE,'HH24');
HOUR:=TO_NUMBER(CURRENT_HOUR);

IF(DAY='SATURDAY' OR DAY='SUNDAY')
THEN
ROLLBACK;
DBMS_OUTPUT.PUT_LINE('Out of business hours - this transaction must be aborted');
END IF;

IF (HOUR>8 AND HOUR<18)
THEN
ROLLBACK;
DBMS_OUTPUT.PUT_LINE('Out of business hours - this transaction must be aborted');
ELSE
DBMS_OUTPUT.PUT_LINE('this transaction is completed on '|| DAY || ' at ' || HOUR);
```

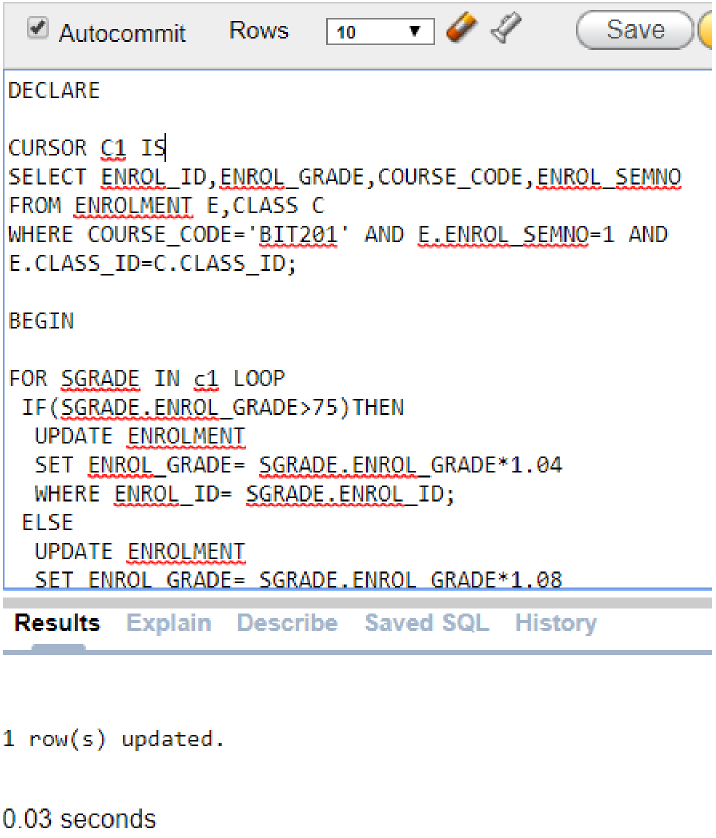


```
INSERT INTO ENROLMENT (ENROL_ID,STU_ID,CLASS_ID,ENROL_GRADE,ENROL_SEMNO,ENROL_SEMYEAR)
VALUES (19,1000033,3,115,1,to_date('05/FEB/20','DD/MON/RR'));
```

```
this transaction is completed on TUESDAY  at 8

1 row(s) inserted.
```

6)

```
DECLARE

CURSOR C1 IS
SELECT ENROL_ID,ENROL_GRADE,COURSE_CODE,ENROL_SEMNO
FROM ENROLMENT E,CLASS C
WHERE COURSE_CODE='BIT201' AND E.ENROL_SEMNO=1 AND
E.CLASS_ID=C.CLASS_ID;

BEGIN

FOR SGRADE IN c1 LOOP
 IF(SGRADE.ENROL_GRADE>75)THEN
  UPDATE ENROLMENT
  SET ENROL_GRADE= SGRADE.ENROL_GRADE*1.04
  WHERE ENROL_ID= SGRADE.ENROL_ID;
 ELSE
  UPDATE ENROLMENT
  SET ENROL_GRADE= SGRADE.ENROL_GRADE*1.08
  WHERE ENROL_ID= SGRADE.ENROL_ID;
 END IF;
END LOOP;


END;
```



☑ Autocommit Rows 10

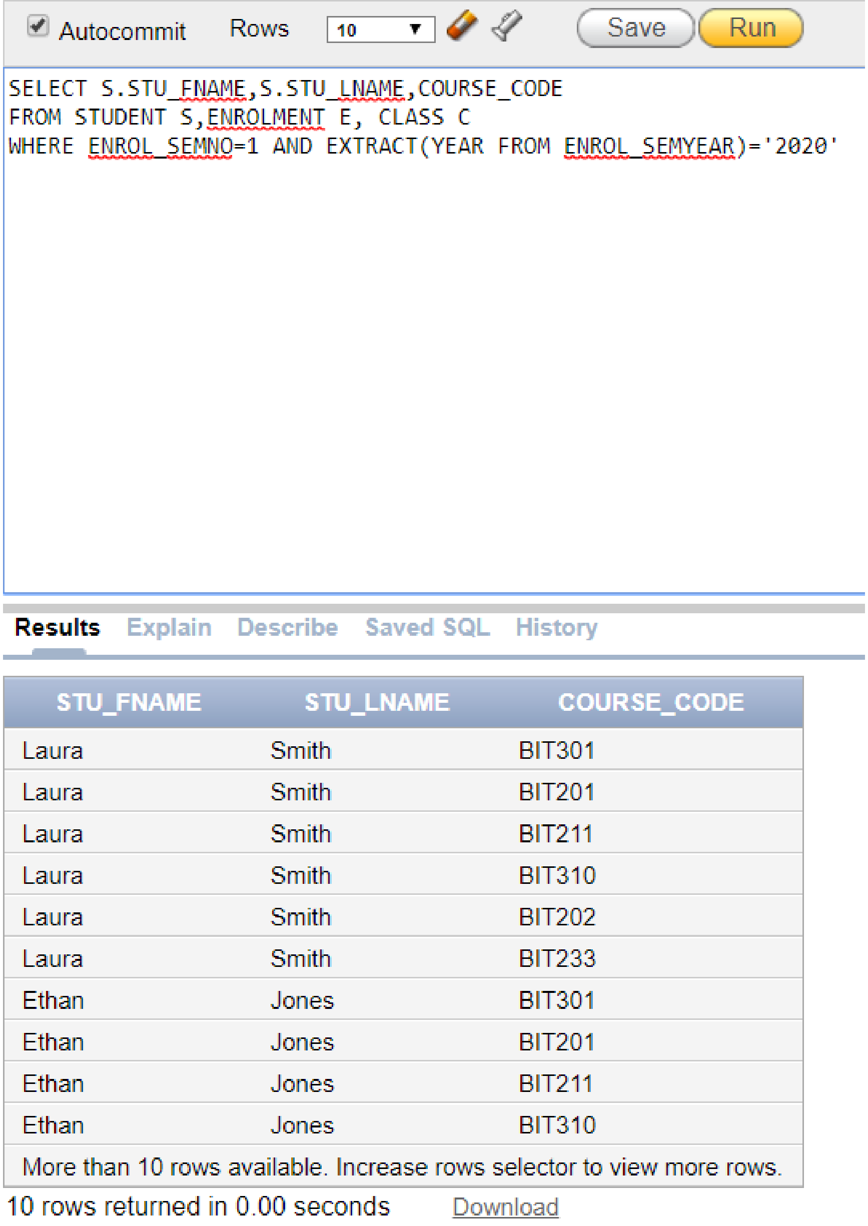
```
DECLARE

CURSOR C1 IS
SELECT ENROL_ID,ENROL_GRADE,COURSE_CODE,ENROL_SEMNO
FROM ENROLMENT E,CLASS C
WHERE COURSE_CODE='BIT201' AND E.ENROL_SEMNO=1 AND
E.CLASS_ID=C.CLASS_ID;

BEGIN

FOR SGRADE IN c1 LOOP
 IF(SGRADE.ENROL_GRADE>75)THEN
  UPDATE ENROLMENT
  SET ENROL_GRADE= SGRADE.ENROL_GRADE*1.04
  WHERE ENROL_ID= SGRADE.ENROL_ID;
 ELSE
  UPDATE ENROLMENT
  SET ENROL_GRADE= SGRADE.ENROL_GRADE*1.08
```

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

0.03 seconds

Part C

1)
SELECT STU_FNAME,S.STU_LNAME,CO.COURSE_CODE
FROM STUDENT S,ENROLMENT E,CLASS C
WHERE ENROL_SEMNO=1 AND EXTRACT(YEAR FROM ENROL_SEMYEAR)='2020'



☑ Autocommit Rows 10

```
SELECT S.STU_FNAME,S.STU_LNAME,COURSE_CODE
FROM STUDENT S,ENROLMENT E, CLASS C
WHERE ENROL_SEMNO=1 AND EXTRACT(YEAR FROM ENROL_SEMYEAR)='2020'
```

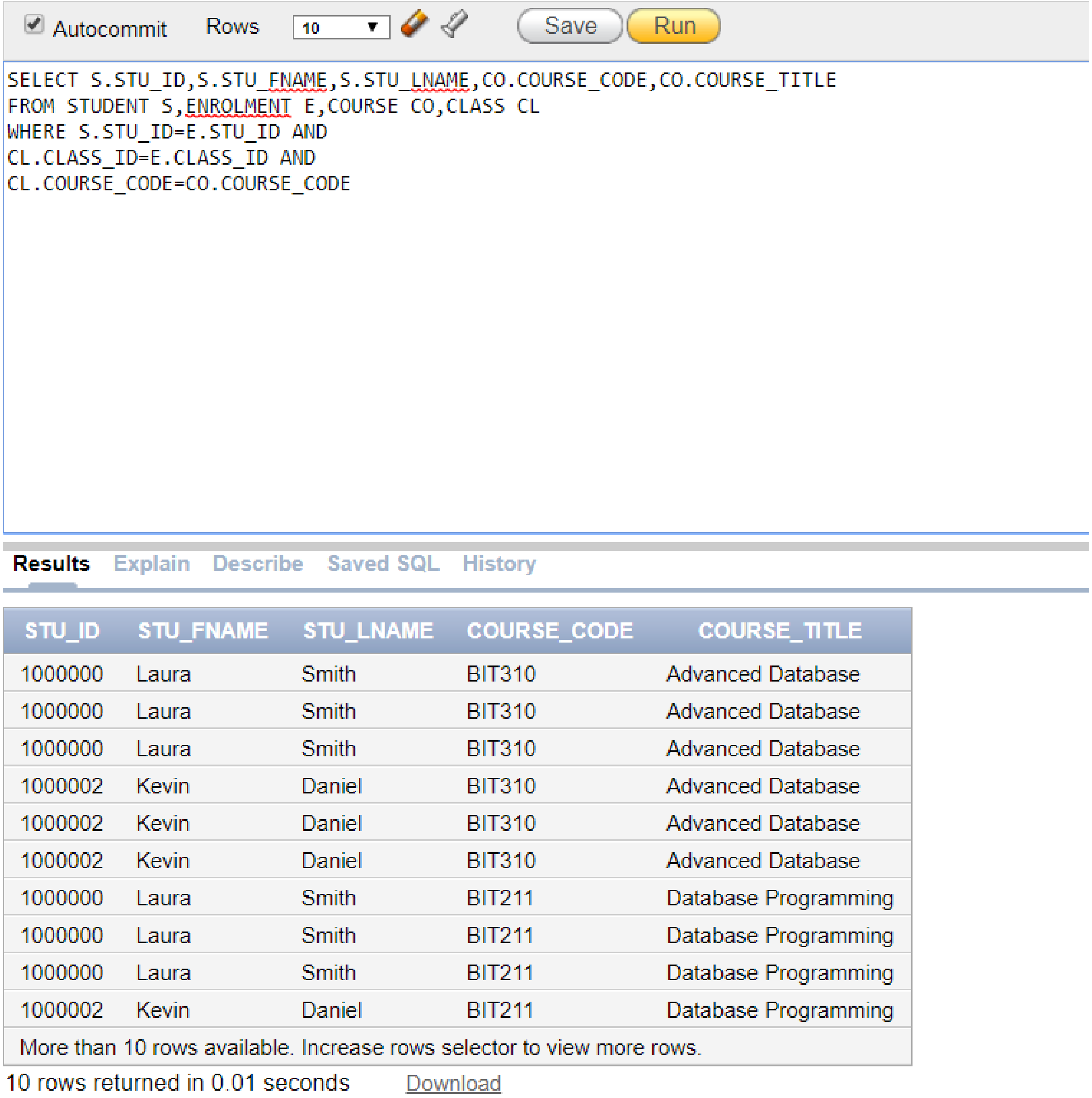
STU_FNAME	STU_LNAME	COURSE_CODE
Laura	Smith	BIT301
Laura	Smith	BIT201
Laura	Smith	BIT211
Laura	Smith	BIT310
Laura	Smith	BIT202
Laura	Smith	BIT233
Ethan	Jones	BIT301
Ethan	Jones	BIT201
Ethan	Jones	BIT211
Ethan	Jones	BIT310

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.00 seconds Download

2)

```
SELECT
S.STU_ID,S.STU_FNAME,S.STU_LNAME,CO.COURSE_CODE,CO.COURSE_TITLE
FROM STUDENT S,ENROLMENT E,COURSE CO,CLASS CL
WHERE S.STU_ID=E.STU_ID AND
CL.CLASS_ID=E.CLASS_ID AND
CL.COURSE_CODE=CO.COURSE_CODE
```



```
SELECT S.STU_ID,S.STU_FNAME,S.STU_LNAME,CO.COURSE_CODE,CO.COURSE_TITLE
FROM STUDENT S,ENROLMENT E,COURSE CO,CLASS CL
WHERE S.STU_ID=E.STU_ID AND
CL.CLASS_ID=E.CLASS_ID AND
CL.COURSE_CODE=CO.COURSE_CODE
```

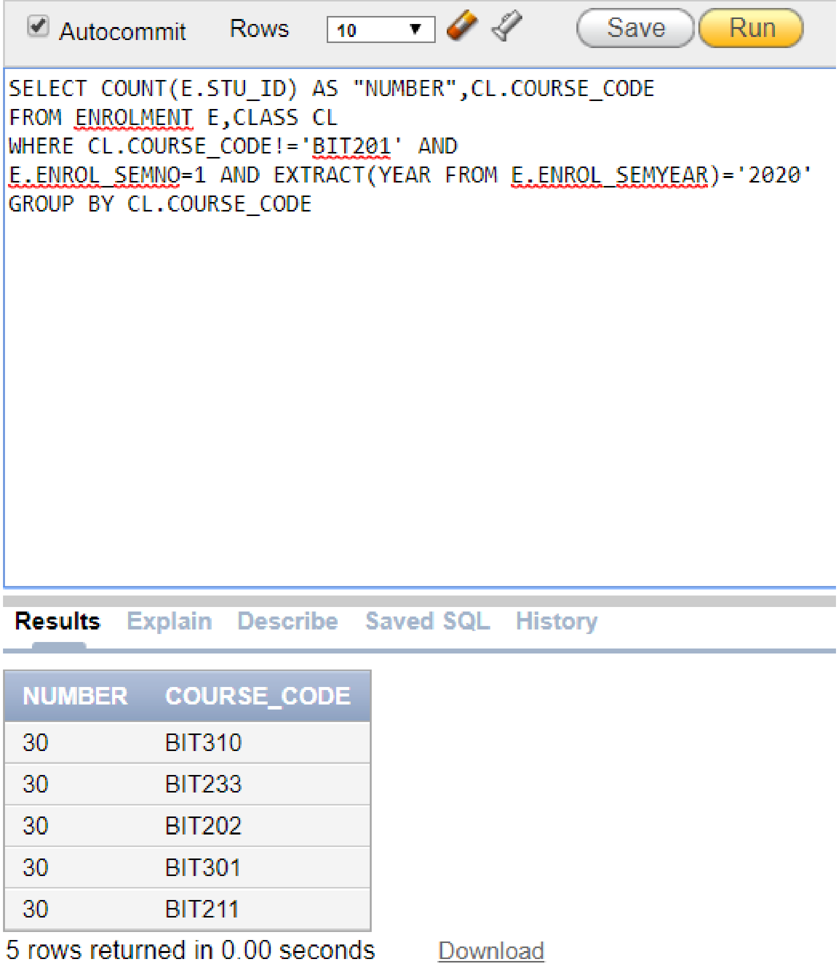
STU_ID	STU_FNAME	STU_LNAME	COURSE_CODE	COURSE_TITLE
1000000	Laura	Smith	BIT310	Advanced Database
1000000	Laura	Smith	BIT310	Advanced Database
1000000	Laura	Smith	BIT310	Advanced Database
1000002	Kevin	Daniel	BIT310	Advanced Database
1000002	Kevin	Daniel	BIT310	Advanced Database
1000002	Kevin	Daniel	BIT310	Advanced Database
1000000	Laura	Smith	BIT211	Database Programming
1000000	Laura	Smith	BIT211	Database Programming
1000000	Laura	Smith	BIT211	Database Programming
1000002	Kevin	Daniel	BIT211	Database Programming

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.01 seconds

3)

```
SELECT COUNT(E.STU_ID) AS "NUMBER",CL.COURSE_CODE
FROM ENROLMENT E,CLASS CL
WHERE CL.COURSE_CODE!='BIT201' AND
E.ENROL_SEMNO=1 AND EXTRACT(YEAR FROM E.ENROL_SEMYEAR)='2020'
GROUP BY CL.COURSE_CODE
```



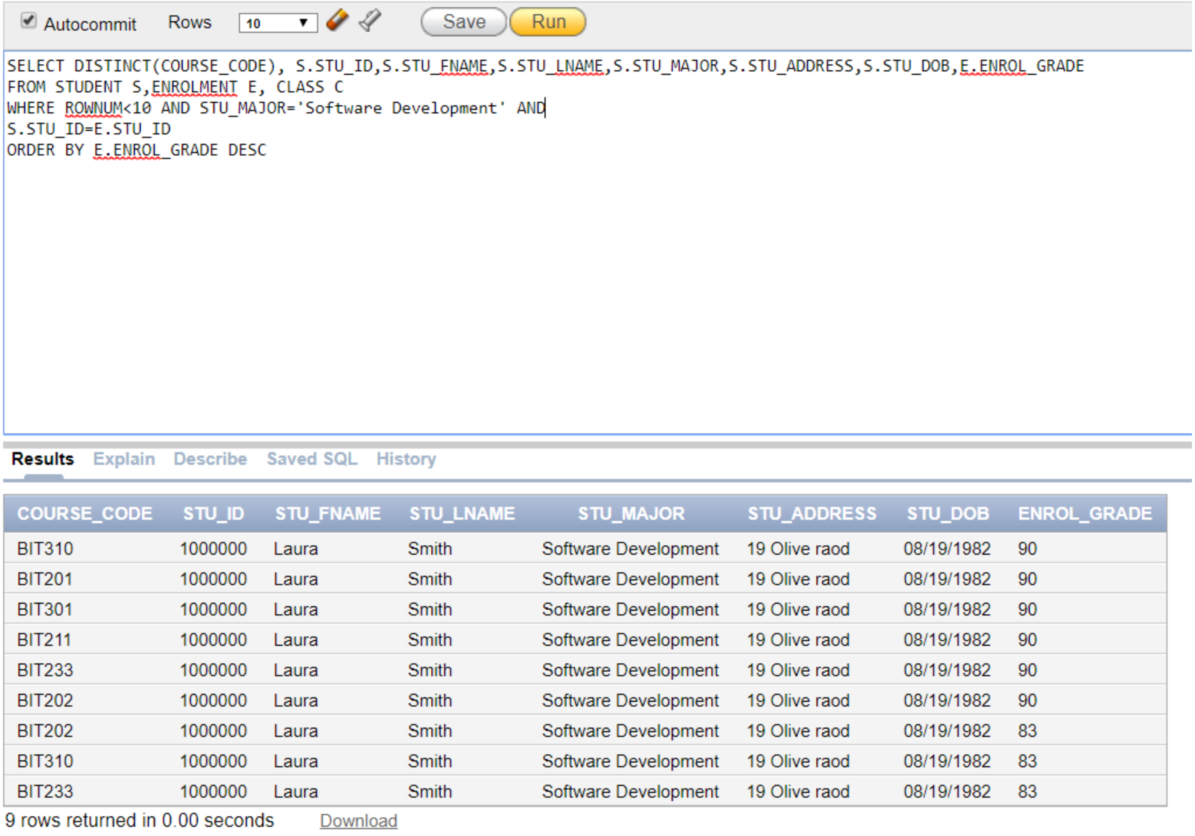
```
SELECT COUNT(E.STU_ID) AS "NUMBER",CL.COURSE_CODE
FROM ENROLMENT E,CLASS CL
WHERE CL.COURSE_CODE!='BIT201' AND
E.ENROL_SEMNO=1 AND EXTRACT(YEAR FROM E.ENROL_SEMYEAR)='2020'
GROUP BY CL.COURSE_CODE
```

NUMBER	COURSE_CODE
30	BIT310
30	BIT233
30	BIT202
30	BIT301
30	BIT211

5 rows returned in 0.00 seconds

4)

```
SELECT DISTINCT(COURSE_CODE),
S.STU_ID,S.STU_FNAME,S.STU_LNAME,S.STU_MAJOR,S.STU_ADDRESS,S.STU_DOB,E.ENROL_GRADE
FROM STUDENT S,ENROLMENT E, CLASS C
WHERE ROWNUM<10 AND STU_MAJOR='Software Development' AND
S.STU_ID=E.STU_ID
ORDER BY E.ENROL_GRADE DESC
```



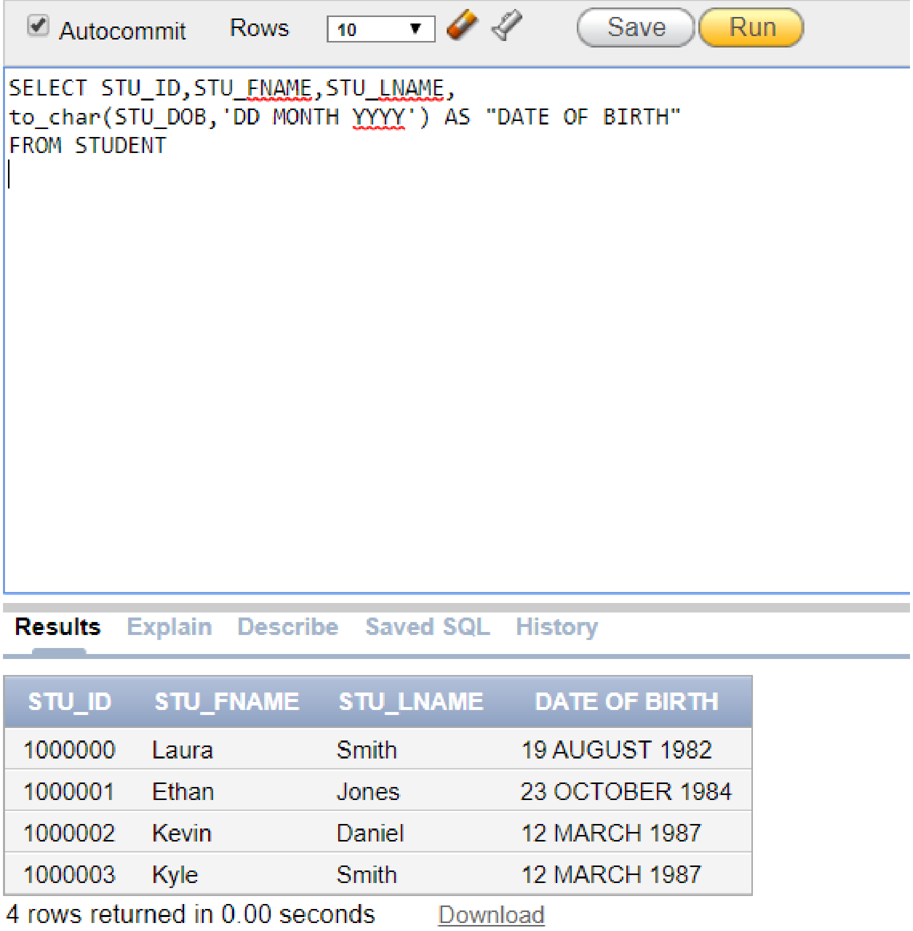
☑ Autocommit Rows 10

```
SELECT DISTINCT(COURSE_CODE), S.STU_ID,S.STU_FNAME,S.STU_LNAME,S.STU_MAJOR,S.STU_ADDRESS,S.STU_DOB,E.ENROL_GRADE
FROM STUDENT S,ENROLMENT E, CLASS C
WHERE ROWNUM<10 AND STU_MAJOR='Software Development' AND
S.STU_ID=E.STU_ID
ORDER BY E.ENROL_GRADE DESC
```

COURSE_CODE	STU_ID	STU_FNAME	STU_LNAME	STU_MAJOR	STU_ADDRESS	STU_DOB	ENROL_GRADE
BIT310	1000000	Laura	Smith	Software Development	19 Olive raod	08/19/1982	90
BIT201	1000000	Laura	Smith	Software Development	19 Olive raod	08/19/1982	90
BIT301	1000000	Laura	Smith	Software Development	19 Olive raod	08/19/1982	90
BIT211	1000000	Laura	Smith	Software Development	19 Olive raod	08/19/1982	90
BIT233	1000000	Laura	Smith	Software Development	19 Olive raod	08/19/1982	90
BIT202	1000000	Laura	Smith	Software Development	19 Olive raod	08/19/1982	90
BIT202	1000000	Laura	Smith	Software Development	19 Olive raod	08/19/1982	83
BIT310	1000000	Laura	Smith	Software Development	19 Olive raod	08/19/1982	83
BIT233	1000000	Laura	Smith	Software Development	19 Olive raod	08/19/1982	83

9 rows returned in 0.00 seconds Download

5)
SELECT STU_ID,STU_FNAME,STU_LNAME,
to_char(STU_DOB,'DD MONTH YYYY') AS "DATE OF BIRTH"
FROM STUDENT



☑ Autocommit Rows 10

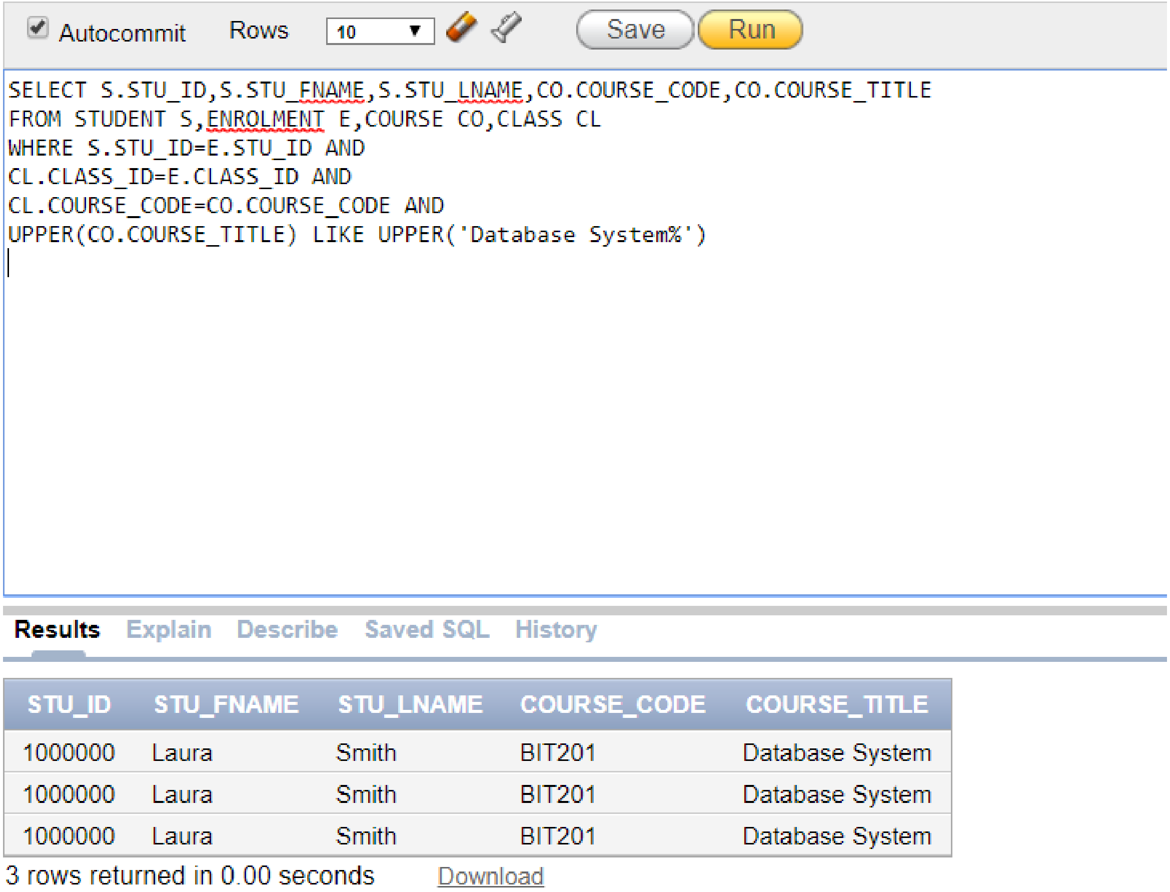
```
SELECT STU_ID,STU_FNAME,STU_LNAME,
to_char(STU_DOB,'DD MONTH YYYY') AS "DATE OF BIRTH"
FROM STUDENT
```

STU_ID	STU_FNAME	STU_LNAME	DATE OF BIRTH
1000000	Laura	Smith	19 AUGUST 1982
1000001	Ethan	Jones	23 OCTOBER 1984
1000002	Kevin	Daniel	12 MARCH 1987
1000003	Kyle	Smith	12 MARCH 1987

4 rows returned in 0.00 seconds

6)

```
SELECT
S.STU_ID,S.STU_FNAME,S.STU_LNAME,CO.COURSE_CODE,CO.COURSE_TITLE
FROM STUDENT S,ENROLMENT E,COURSE CO,CLASS CL
WHERE S.STU_ID=E.STU_ID AND
CL.CLASS_ID=E.CLASS_ID AND
CL.COURSE_CODE=CO.COURSE_CODE AND
UPPER(CO.COURSE_TITLE) LIKE UPPER('Database System%')
```



```
SELECT S.STU_ID,S.STU_FNAME,S.STU_LNAME,CO.COURSE_CODE,CO.COURSE_TITLE
FROM STUDENT S,ENROLMENT E,COURSE CO,CLASS CL
WHERE S.STU_ID=E.STU_ID AND
CL.CLASS_ID=E.CLASS_ID AND
CL.COURSE_CODE=CO.COURSE_CODE AND
UPPER(CO.COURSE_TITLE) LIKE UPPER('Database System%')
```

STU_ID	STU_FNAME	STU_LNAME	COURSE_CODE	COURSE_TITLE
1000000	Laura	Smith	BIT201	Database System
1000000	Laura	Smith	BIT201	Database System
1000000	Laura	Smith	BIT201	Database System

3 rows returned in 0.00 seconds

References

techonthenet, 2020. *Oracle / PLSQL: Functions.* [Online]
Available at: https://www.techonthenet.com/oracle/functions.php
[Accessed 27 May 2020].

tutorialspoint, 2020. *PL/SQL - Triggers.* [Online]
Available at: https://www.tutorialspoint.com/plsql/plsql_triggers.htm
[Accessed 01 June 2020].