JAVED, NAHID, SAIF, TARIK, MANISH

DATABASE COURSEWORK 2

Relational Schema

FOREIGN KEY = *EXAMPLE*

PRIMARY KEY = **EXAMPLE**

Student

STUDENTID	FIRSTNAME	LASTNAME	DOB	YEARENROLLED	*SUBJECTID*

Subject

SUBJECTID	NAME	NUMBEROFCOURSEWORKS	SEMESTER	ACADEMICYEAR	*LECTURERID*

Lecturer

LECTURERID	FIRSTNAME	LASTNAME

Coursework

COURSEWORKID	MAXMARKS	MARKSGAINED	*SUBJECTID*	COURSEWORK NUMBER	STATUSOFSUBMISSION	MODULEPERCENTAGE

Exam

EXAMID	MAXMARK	MARKSGAINED	ATTEMPTNUMBER	*LECTURERID*	*SUBJECTID*

Questions

QUESTIONID	MAXMARK	QUESTIONNUMBER	*EXAMID*	MARKSGAINED	*LECTURERID*

EXPLAINATION OF THE CODE

Table Subject –

Created a subject table that consists of SUBJECTID, NAME, NUMBEROFCOURSEOWRKS, SEMESTER, ACADEMICYEAR and LECTURERID. LECTURERID is a Foreign key from the table lecturer. SUBEJCTID is the primary key. LECTURERID would teach the contents of the subject.

Table Student –

Created a student table that consists of STUDENTID, FIRSTNAME, LASTNAME, DOB, YEARENROLLED and SUBJECTID. The foreign keys in the table is SUBJECTID and the primary key is STUDENTID. SUBJECTID would show what the student has taken as a subject.

Table Exam –

Created an exam table that consists of EXAMID, MAXMARKS, MARKSGAINED, ATTEMPTNUMBER, LECTURERID and SUBJECTID. The foreign keys are SUBJECTID and LECTURERID. The primary key is EXAMID. Its linked to SUBJECTID because a module would have an exam and LECTURERID would have marked the papers.

Table Question –

Created a question table that consists of QUESTIONID, MAXMARKS, QUESTIONNUMBER, EXAMID, MARKSGAINED and LECTURERID. The foreign keys are EXAMID and LECTURERID. The primary key is QUESTIONID. EXAMID is linked because the exam consists of questions and the LECTURERID is showing what question they mark.

Table Coursework –

Created a coursework table that consists of COURSEWORKID, MAXMARKS, MARKSGAINED, SUBJECTID, COURSEWORKNUMBER, STATUSOFSUBMISSION an MODULEPERCENTAGE. The foreign key is SUBJECTID and the primary key is COURSEWORKID. SUBJECTID is linked because the subject must include coursework as another form of assessment other than doing an exam.

Table Lecturer –

Created a lecturer table that consist of LECTUERID, FIRSTNAME AND LASTNAME. The primary key used is LECTURERID.

DATA INSERT –

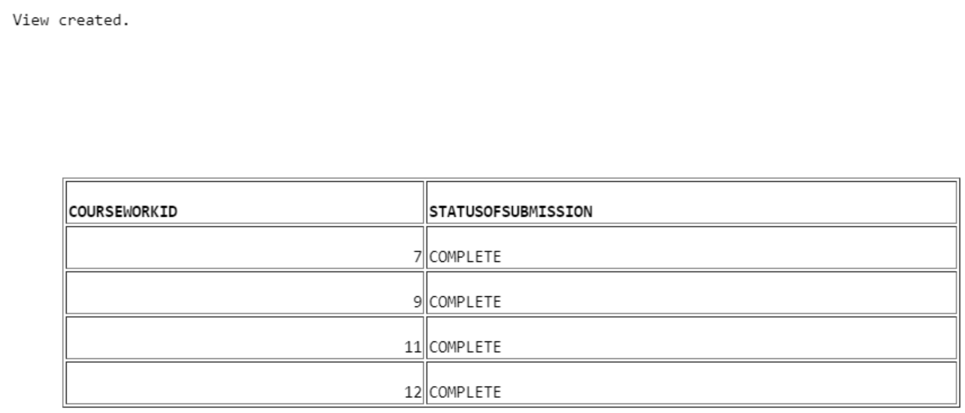
Created multiple datasets that link to each attribute to their respective tables. They are all linked to each other to create any possible statement.

CREATE VIEW –

```
CREATE VIEW COMPLETED_COURSEWORK AS
SELECT COURSEWORKID, STATUSOFSUBMISSION
FROM COURSEWORK
WHERE STATUSOFSUBMISSION = 'COMPLETE';

SELECT * FROM COMPLETED_COURSEWORK;
```

This VIEW is creating COMPLETED_COURSEWORK and it is selecting the COURSEWORKID plus the STATUSOFSUBMISSION from subject. However, it has a condition where it should only print out the status that of the coursework that is Complete. It will not print out the incomplete ones. So, when it says from coursework, it is getting the attributes that are COURSEWORKID and STATUSOFSUBMISSION from the class COURSEWORK. Then I created a select statement so that it can print the table by getting the data from the columns.



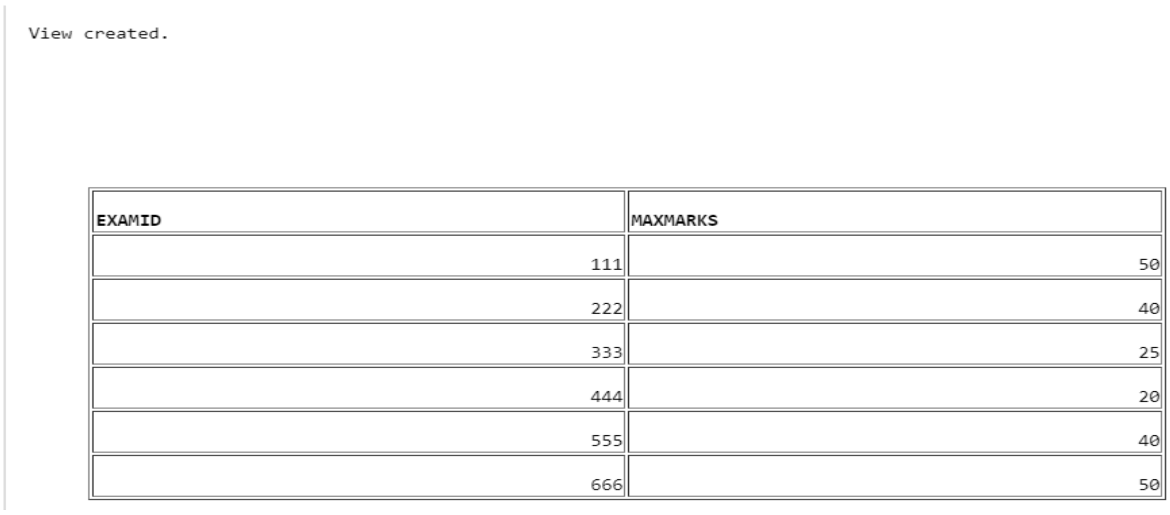
```
View created.
```

COURSEWORKID	STATUSOFSUBMISSION
7	COMPLETE
9	COMPLETE
11	COMPLETE
12	COMPLETE

```
CREATE VIEW EXAM_MARKS AS
SELECT EXAMID, MAXMARKS
FROM EXAM
WHERE MAXMARKS > '15';

SELECT * FROM EXAM_MARKS;
```

This statement created a view called EXAM_MARKS and it is selecting EXAMID and MAXMARKS from the class EXAM. But the condition is that it should display all the MAXMARKS that is greater than 15. Then to show if it displays the table, I created a select statement that it should get all the data from the two columns.



```
View created.
```

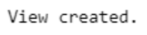
EXAMID	MAXMARKS
111	50
222	40
333	25
444	20
555	40
666	50

JAVED, NAHID, SAIF, TARIK, MANISH

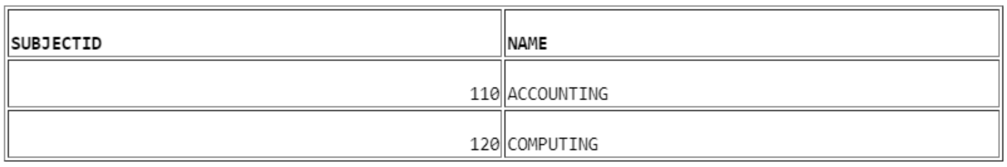
```
CREATE VIEW SEM_SUBJECT AS
SELECT SUBJECTID, NAME
FROM SUBJECT
WHERE SEMESTER = '1';

SELECT * FROM SEM_SUBJECT;
```

This statement creates a view called SEM_SUBJECT and it is selecting SUBJECTID and NAME from the class SUBJECT. But the condition is that it should display only semester 1 and not any other semester. Then to show if it displays the table, I created a select statement that it gets all the data from the two columns.



```
View created.
```

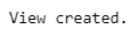


SUBJECTID	NAME
110	ACCOUNTING
120	COMPUTING

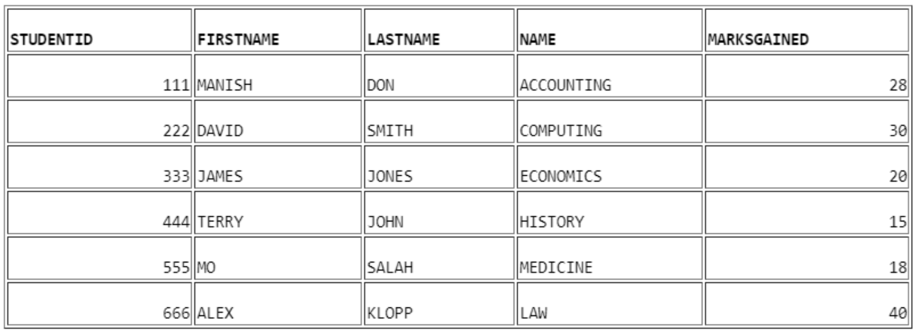
```
CREATE VIEW STUDENT_DETAILS AS
SELECT STUDENT.STUDENTID, STUDENT.FIRSTNAME, STUDENT.LASTNAME,
SUBJECT.NAME,EXAM.MARKSGAINED
FROM STUDENT
INNER JOIN SUBJECT ON STUDENT.SUBJECTID=SUBJECT.SUBJECTID
INNER JOIN EXAM ON EXAM.SUBJECTID=SUBJECT.SUBJECTID;

SELECT * FROM STUDENT_DETAILS;
```

This statement creates a view called STUDENT_DETAILS and it is selecting the columns from STUDENT. However, I've used something unique called INNER JOIN, which merges more than 2 classes together. In this case it is merging the table STUDENT, SUBJECT and EXAM together. It is getting the attributes from SUBJECT and EXAM. Then to show if it displays the table, I created a select statement that it gets all the data from the columns selected.



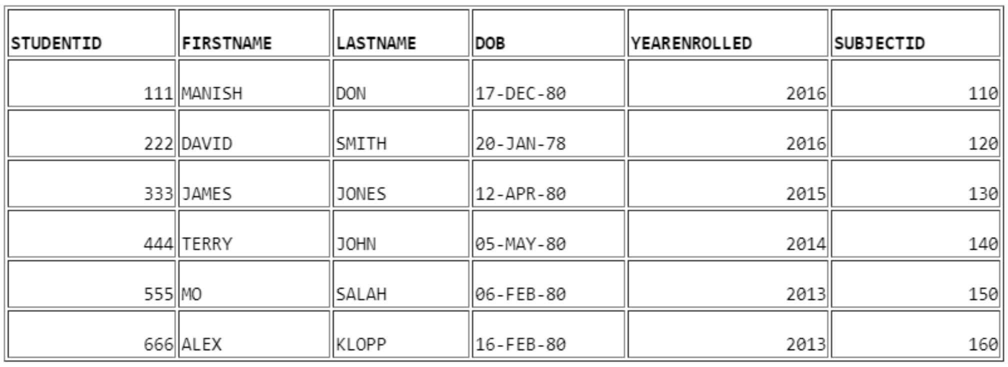
```
View created.
```



STUDENTID	FIRSTNAME	LASTNAME	NAME	MARKSGAINED
111	MANISH	DON	ACCOUNTING	28
222	DAVID	SMITH	COMPUTING	30
333	JAMES	JONES	ECONOMICS	20
444	TERRY	JOHN	HISTORY	15
555	MO	SALAH	MEDICINE	18
666	ALEX	KLOPP	LAW	40

--DISPLAY TABLE OUTPUT
SELECT * FROM STUDENT;

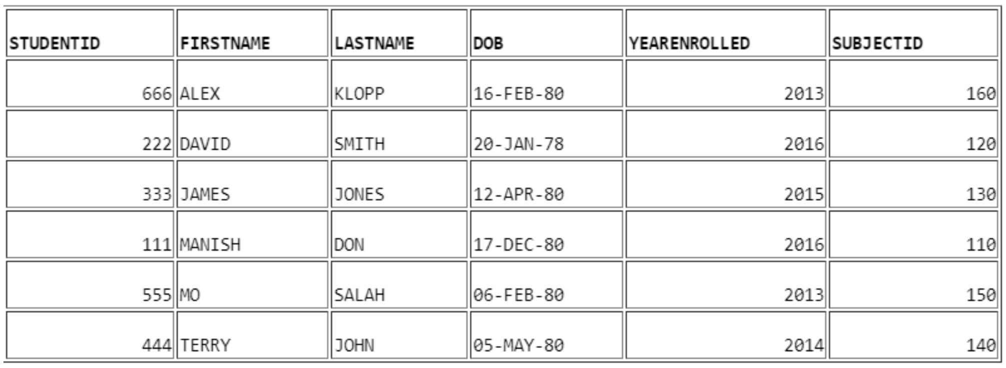
This statement shows that it is selecting all the columns in the table and printing it out. The table below shows what the output consists of.



STUDENTID	FIRSTNAME	LASTNAME	DOB	YEARENROLLED	SUBJECTID
111	MANISH	DON	17-DEC-80	2016	110
222	DAVID	SMITH	20-JAN-78	2016	120
333	JAMES	JONES	12-APR-80	2015	130
444	TERRY	JOHN	05-MAY-80	2014	140
555	MO	SALAH	06-FEB-80	2013	150
666	ALEX	KLOPP	16-FEB-80	2013	160

--DISPLAY RECORDS IN ORDER
SELECT * FROM STUDENT ORDER BY FIRSTNAME ASC;

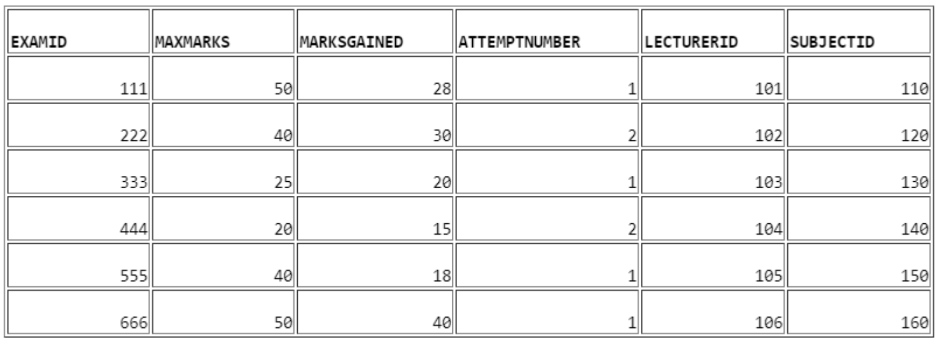
This select statement shows that it is getting all the attributes from table STUDENT and it is ordering the table ascendingly by FIRSTNAME only. I am using a unique keyword called ASC that allows the table to sorted ascendingly.



STUDENTID	FIRSTNAME	LASTNAME	DOB	YEARENROLLED	SUBJECTID
666	ALEX	KLOPP	16-FEB-80	2013	160
222	DAVID	SMITH	20-JAN-78	2016	120
333	JAMES	JONES	12-APR-80	2015	130
111	MANISH	DON	17-DEC-80	2016	110
555	MO	SALAH	06-FEB-80	2013	150
444	TERRY	JOHN	05-MAY-80	2014	140

--DISPLAY RECORDS DEPENDING ON VALUE
SELECT * FROM EXAM WHERE MAXMARKS > 10;

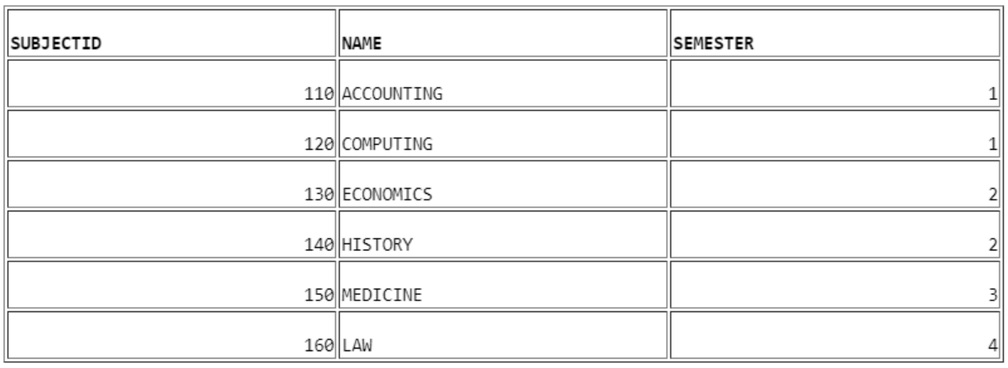
The statement shows that it is selecting everything that is in the EXAM table, but the condition says that the MAXMARKS must be greater than 10. Therefore, it will only print out a table that only consists of MAXMARKS only greater than 10



EXAMID	MAXMARKS	MARKSGAINED	ATTEMPTNUMBER	LECTURERID	SUBJECTID
111	50	28	1	101	110
222	40	30	2	102	120
333	25	20	1	103	130
444	20	15	2	104	140
555	40	18	1	105	150
666	50	40	1	106	160

--DISPLAY SPECIFIC COLUMNS FROM TABLE
SELECT SUBJECTID, NAME, SEMESTER FROM SUBJECT;

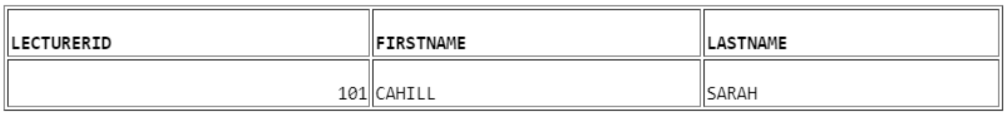
This statement shows that it is selecting the specific attributes from SUBJECT. So, it should only have three columns printing out. So, it prints out SUBJECTID, NAME and SEMESTER.



SUBJECTID	NAME	SEMESTER
110	ACCOUNTING	1
120	COMPUTING	1
130	ECONOMICS	2
140	HISTORY	2
150	MEDICINE	3
160	LAW	4

--DISPLAY SPECIFIC COLUMNS BASED ON CONDITION
SELECT LECTURERID, FIRSTNAME, LASTNAME FROM LECTURER WHERE LASTNAME = 'SARAH';

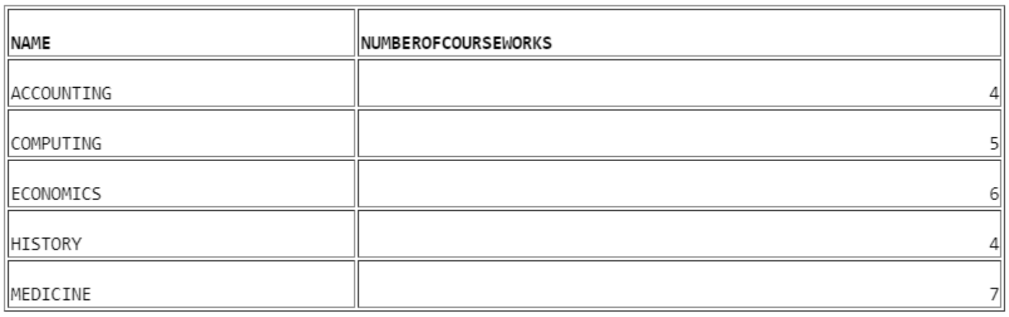
This statement selects the specific attributes which are lecturerid, FIRSTNAME and lastname from the table lecturer but the condition is that it should only print out any name that has sarah in it.



LECTURERID	FIRSTNAME	LASTNAME
101	CAHILL	SARAH

--DISPLAY SPECIFIC RECORDS
SELECT NAME , NUMBEROFCOURSEWORKS FROM SUBJECT WHERE NUMBEROFCOURSEWORKS > '3';

This statement displays NUMBEROFCOURSEWORKS that each SUBJECT has with the condition that the NUMBEROFCOURSEWORKS are greater than 3. All modules that are under the requirement are not displayed.

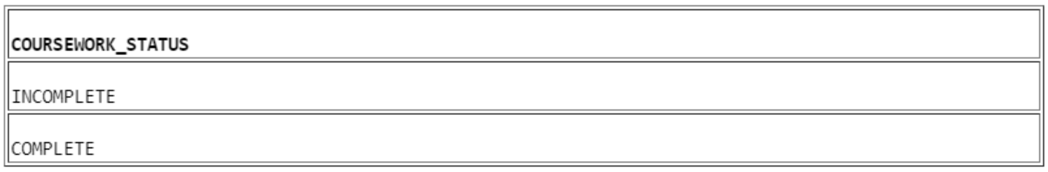


NAME	NUMBEROFCOURSEWORKS
ACCOUNTING	4
COMPUTING	5
ECONOMICS	6
HISTORY	4
MEDICINE	7

--DISTINCT RECORDS

```
SELECT DISTINCT(STATUSOFSUBMISSION) AS COURSEWORK_STATUS FROM
COURSEWORK GROUP BY STATUSOFSUBMISSION;
```

This statement uses a DISTINCT key that returns the values Incomplete and Complete. However, it does not duplicate any other values that are the same as those two values mentioned above. So, it is getting the attribute that is labelled as COURSEWORK_STATUS from the table COURSEWORK. Using the group key allows you to have one or more columns, which in this case we have one column that stores two results.

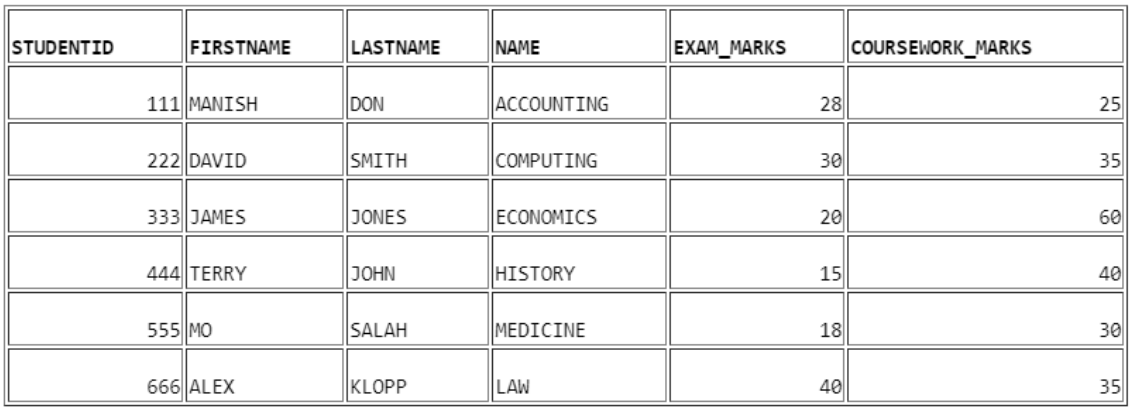


COURSEWORK_STATUS
INCOMPLETE
COMPLETE

--SELECT WITH JOIN

```
SELECT STUDENT.STUDENTID, STUDENT.FIRSTNAME, STUDENT.LASTNAME,
SUBJECT.NAME,EXAM.MARKSGAINED as EXAM_MARKS, COURSEWORK.MARKSGAINED as
COURSEWORK_MARKS
FROM STUDENT
INNER JOIN SUBJECT ON STUDENT.SUBJECTID=SUBJECT.SUBJECTID
INNER JOIN COURSEWORK ON COURSEWORK.SUBJECTID=SUBJECT.SUBJECTID
INNER JOIN EXAM ON EXAM.SUBJECTID=SUBJECT.SUBJECTID;
```

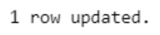
Inner join is method of joining data together that have matching values in multiple tables. This allows linked data to be combined into one large table instead of several small tables. The table USES INNER JOIN to link the module name, exam marks and the coursework marks of the student which all respective are attributes in their own tables.

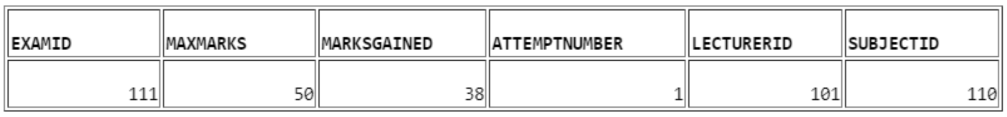


STUDENTID	FIRSTNAME	LASTNAME	NAME	EXAM_MARKS	COURSEWORK_MARKS
111	MANISH	DON	ACCOUNTING	28	25
222	DAVID	SMITH	COMPUTING	30	35
333	JAMES	JONES	ECONOMICS	20	60
444	TERRY	JOHN	HISTORY	15	40
555	MO	SALAH	MEDICINE	18	30
666	ALEX	KLOPP	LAW	40	35

--UPDATE A RECORD
UPDATE EXAM SET MARKSGAINED = '38' WHERE EXAMID = '111';
SELECT * FROM EXAM WHERE EXAMID = '111';

This statement is updating the EXAM SET MARKSGAINED to 38 to the following EXAMID that is 111. Then the select statement gets all the data from Exam table and changes the MARKSGAINED to 38 in EXAMID 111. The following output is shown below.

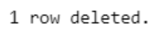

```
1 row updated.
```

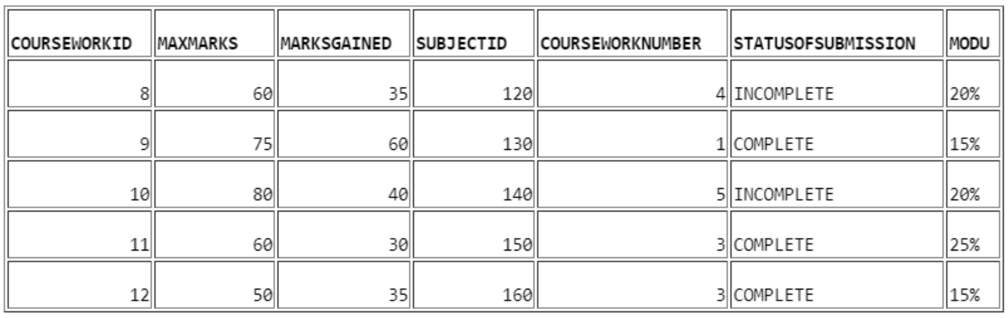


EXAMID	MAXMARKS	MARKSGAINED	ATTEMPTNUMBER	LECTURERID	SUBJECTID
111	50	38	1	101	110

--DELETE A RECORD
DELETE FROM COURSEWORK WHERE COURSEWORKID = '7';
SELECT * FROM COURSEWORK

This statement allows you to delete a record, but the condition is that COURSEWORKID must equal to 7. The select statement would get all the attributes from COURSEWORK. The output shows that one row has been deleted so instead of 6 rows there are now 5.


```
1 row deleted.
```



COURSEWORKID	MAXMARKS	MARKSGAINED	SUBJECTID	COURSEWORKNUMBER	STATUSOFSUBMISSION	MODU
8	60	35	120	4	INCOMPLETE	20%
9	75	60	130	1	COMPLETE	15%
10	80	40	140	5	INCOMPLETE	20%
11	60	30	150	3	COMPLETE	25%
12	50	35	160	3	COMPLETE	15%