

**ACHIEVERS CONSULTANCY (*For Better Career*)**

*Jagadeeswari Sambangi, Amruthavarshini Sadineni,* *Vineeth Petnakota, Devanshi Patel*

IS 610 Group 2 Project

UMBC Spring 2023

Version 1 // May 2023

**STUDENT TABLE**

CREATE TABLE student (

s\_id INT PRIMARY KEY,

s\_name VARCHAR(50),

s\_info VARCHAR(50),

s\_doc INT,

s\_major VARCHAR(50),

s\_eid VARCHAR(50),

s\_add VARCHAR(100)

);

**BANK TABLE**

CREATE TABLE bank (

bank\_id INT PRIMARY KEY,

bank\_name VARCHAR(50),

bank\_add VARCHAR(100),

bank\_det INT,

bank\_eid VARCHAR(50),

s\_id INT,

FOREIGN KEY (s\_id) REFERENCES student(s\_id)

);

**INSTITUTION TABLE**

CREATE TABLE institution (

ins\_code INT PRIMARY KEY,

ins\_name VARCHAR(50),

ins\_infopage VARCHAR(100),

ins\_courses VARCHAR(100),

s\_id INT,

bank\_id INT,

FOREIGN KEY (s\_id) REFERENCES student(s\_id),

FOREIGN KEY (bank\_id) REFERENCES bank(bank\_id)

);

**TRAVEL TABLE**

CREATE TABLE travel (

ref\_id INT PRIMARY KEY,

t\_date DATE,

t\_insu VARCHAR(50),

t\_dest VARCHAR(50),

s\_doc INT,

s\_id INT,

bank\_id INT,

FOREIGN KEY (s\_id) REFERENCES student(s\_id),

FOREIGN KEY (bank\_id) REFERENCES bank(bank\_id)

);

**CONSULTANCY TABLE**

CREATE TABLE consultancy (

c\_name\_id INT PRIMARY KEY,

c\_det INT,

c\_agent VARCHAR(50),

c\_cont\_info VARCHAR(50),

c\_eid VARCHAR(50),

s\_id INT,

ins\_code INT,

FOREIGN KEY (s\_id) REFERENCES student(s\_id),

FOREIGN KEY (ins\_code) REFERENCES institution(ins\_code)

);

describe student;

describe travel;

describe bank;

describe institution;

describe consultancy;

NOTE: The above SQL code creates four tables: student, bank, institution, travel, and consultancy.

* The student table has columns for s\_id (primary key), s\_name, s\_info, s\_doc, s\_major, s\_eid, and s\_add.
* The bank table has columns for bank\_id (primary key), bank\_name, bank\_add, bank\_det, bank\_eid, and s\_id (foreign key referencing student table).
* The institution table has columns for ins\_code (primary key), ins\_name, ins\_infopage, ins\_courses, s\_id (foreign key referencing student table), and bank\_id (foreign key referencing bank table).
* The travel table has columns for ref\_id (primary key), t\_date, t\_insu, t\_dest, s\_doc, s\_id (foreign key referencing student table), and bank\_id (foreign key referencing bank table).
* The consultancy table has columns for c\_name\_id (primary key), c\_det, c\_agent, c\_cont\_info, c\_eid, s\_id (foreign key referencing student table), and ins\_code (foreign

**INSERT DATA INTO STUDENT TABLE**

INSERT INTO student (s\_id, s\_name, s\_info, s\_doc, s\_major, s\_eid, s\_add)

VALUES (1001, 'Jagadeeswari', 'Grad Student', 20, 'IS', 'hd76292@umbc.edu', '123 Main Street, Anytown, USA');

INSERT INTO student (s\_id, s\_name, s\_info, s\_doc, s\_major, s\_eid, s\_add)

VALUES (1002, 'Vineeth', 'Senior', 30, 'BA', 'rx82617@umbc.edu', '456 High Street, Anytown, USA');

**INSERT DATA INTO BANK TABLE**

INSERT INTO bank (bank\_id, bank\_name, bank\_add, bank\_det, bank\_eid, s\_id)

VALUES (101, 'ABC Bank', '789 Oak Street, Anytown, USA', 301, 'abcbank@xyz.com', 1001);

INSERT INTO bank (bank\_id, bank\_name, bank\_add, bank\_det, bank\_eid, s\_id)

VALUES (102, 'EFG Bank', '700 Oak Street, Anytown, USA', 302, 'efgbank@xyz.com', 1002);

**INSERT DATA INTO INSTUTUTION TABLE**

INSERT INTO institution (ins\_code, ins\_name, ins\_infopage, ins\_courses, s\_id, bank\_id)

VALUES (1, 'UMBC', 'https://umbc.edu', 'Information Systems', 1001, 101);

INSERT INTO institution (ins\_code, ins\_name, ins\_infopage, ins\_courses, s\_id, bank\_id)

VALUES (2, 'UMBC', 'https://umbc.edu', 'Information Systems', 1002, 102);

**INSERT DATA INTO TRAVEL TABLE**

INSERT INTO travel (ref\_id, t\_date, t\_insu, t\_dest, s\_doc, s\_id, bank\_id)

VALUES (10001, date '2023-06-06', 'TravelSafe', 'Baltimore', 20, 1001, 101);

**INSERT DATA INTO CONSULTANCY TABLE**

INSERT INTO consultancy (c\_name\_id, c\_det, c\_agent, c\_cont\_info, c\_eid, s\_id, ins\_code)

VALUES (1, 301, 'ABC Agency', 'abcagency@email.com', 'abcagency@xyz.com', 1001,1);

* NOTE: For example: The above query is an INSERT statement that inserts a new row into the institution table. The values being inserted are specified in the VALUES clause. The values being inserted are: - ins\_code - 610 - ins\_name - 'UMBC' - ins\_infopage - '[https://umbc.edu](https://umbc.edu/)' - ins\_courses - 'Information Systems' - s\_id - 1001 - bank\_id - 101 These values correspond to the columns specified in the institution table. The ins\_code column is the primary key of the table, so it must have a unique value for each row. The other columns are non-key columns that contain additional information about the institution.

select \* from student;

select \* from consultancy;

select \* from bank;

select \* from travel;

select s\_id,s\_name from student;

select bank\_id, bank\_name from bank;

select \* from student

where s\_id=1001;

select \* from student

where s\_info='Grad Student' and s\_major='IS';

* NOTE:

The first five queries are simple SELECT statements that retrieve all the data from the respective tables.

* The first query select \* from student; retrieves all the records and columns from the student table.
* The second query select \* from consultancy; retrieves all the records and columns from the consultancy table.
* The third query select \* from bank; retrieves all the records and columns from the bank table. The fourth query select \* from travel; retrieves all the records and columns from the travel table.
* The fifth query select s\_id,s\_name from student; retrieves only the s\_id and s\_name columns from the student table.
* The sixth query select bank\_id, bank\_name from bank; retrieves only the bank\_id and bank\_name columns from the bank table.
* The seventh query select \* from student where s\_id=1001; retrieves all columns of the records in the student table where the s\_id is equal to 1001.
* The eighth query select \* from student where s\_info='Grad Student' and s\_major='IS'; retrieves all columns of the records in the student table where the s\_info is equal to 'Grad Student' and the s\_major is equal to 'IS'.

**UPDATE STUDENT “AMRUTHA” DATA WITH STUDENT “JAGADEESWARI” NAME**

update student

set s\_name='Amrutha'

where s\_id=1001;

select \* from student;

* NOTE: Updates the value of s\_name to "Amrutha" for the row in the student table where the s\_id value is 1001. The UPDATE statement modifies an existing record in a table. The SET clause is used to specify the column name and the new value. The WHERE clause is used to filter the records that will be updated. In this case, only the record where s\_id is 1001 will be updated.

**UPDATE STUDENT “VINEETH” DATA WITH STUDENT “DEVANSHI” NAME**

update student

set s\_name='Devanshi',s\_major='CS'

where s\_major='BA';

select \* from student;

* NOTE: Updates the value of s\_name to "Devanshi" and s\_major to "CS" for all the rows in the student table where the value of s\_major is "BA". This UPDATE statement updates multiple records in the student table. The SET clause is used to specify the column names and the new values. The WHERE clause is used to filter the records that will be updated. In this case, only the records where s\_major is "BA" will be updated. The last statement SELECT \* FROM student is used to view the updated records in the student table.

**JOIN: INNER JOIN & LEFT JOIN**

SELECT ins\_name, s\_name, c\_agent

FROM institution

INNER JOIN student ON institution.s\_id = student.s\_id

INNER JOIN consultancy ON student.s\_id = consultancy.s\_id;

* NOTE: The above SQL statement selects the institution name, student name, and consultancy agent for all records where the institution's s\_id matches the student's s\_id and the student's s\_id matches the consultancy's s\_id, using INNER JOIN.

SELECT s.s\_name, c.c\_agent

FROM student s

LEFT JOIN consultancy c ON s.s\_id = c.s\_id;

* NOTE: The above SQL statement selects the student’s name and consultancy agent for all students and any associated consultancy agents using a LEFT JOIN, which means that all records from the "left" (student) table are included in the result, even if there is no matching record in the "right" (consultancy) table.

**GROUP COMMAND**

SELECT s\_major, COUNT(\*)

FROM student

GROUP BY s\_major;

* NOTE: The above code is a SQL query that retrieves the number of students in each major from the "student" table. The query uses the GROUP BY clause to group the students by their major and then the COUNT function is used to count the number of students in each group. The output of this query will show the count of students in each major.

SELECT bank\_name, MIN(bank\_det) AS min\_det, MAX(bank\_det) AS max\_det

FROM bank

GROUP BY bank\_name;

* NOTE: The above SQL query selects the bank\_name, along with the minimum and maximum bank\_det values, from the bank table. It groups the rows by bank\_name, which means that for each unique bank\_name, the query calculates the minimum and maximum bank\_det values. The results are returned as a table.

SELECT s\_major, AVG(s\_doc) AS avg\_doc

FROM student

GROUP BY s\_major;

* NOTE: This SQL query selects the s\_major and the average s\_doc (a numeric column in the student table) for each unique s\_major value. It groups the rows by s\_major and calculates the average s\_doc for each group. The results are returned as a table.