SQL CASE Study: Student Information System

Introduction

The objective of this SQL Case Study is to provide a frame of reference to show the concepts and syntax of SQL, an Integrated Student Information System for a typical university is considered. To make different type of queries, manipulation, updating and generate different type of reports out of the above application in hand let's consider the four (4) relations tables given below:

a. **STUDENT** table - Each row represents data about a student in the university.
b. **FACULTY** table - Each row represents data about a faculty member employed by the university.
c. **COURSE** table - Each row represents a course being taught during the current term.
d. **REGN** table - Each row represents a student course registration. For example, if a student is enrolled in three courses, three rows would appear in this table.

Description of Student Information System Tables

A. STUDENT TABLE –

Column Desc	Column Name	Data Type	Length	Decimal Places	Nulls Allowed	Codes
Student ID	SID	Char	3		No	
Student Name	SNAME	Char	10		No	
Gender	SEX	Char	3		Yes	F/M
Program of Study	MAJOR	Char	3		Yes	ACC, FIN, MGT, MKT
Grade Point Average	GPA	Decimal	3	2	Yes	

B. FACULTY TABLE -

Column Desc	Column Name	Data Type	Length	Decimal Places	Nulls Allowed	Codes
Faculty ID	FID	Char	3		NO	
Faculty Name	FNAME	Char	10		No	
Phone Ext	EXT	Char	3		Yes	
Department	DEPT	Char	3		Yes	ACC, FIN,MGT,MKT
Rank of Faculty	RANK1	Char	4		Yes	INST, ASST, ASSO, FULL
Salary	SALARY	Integer			Yes	

C. COURSE TABLE -

Column Desc	Column Name	Data Type	Length	Decimal Places	Nulls Allowed	Codes
Course no.	CRSNBR	Char	6		No	
Course title	CNAME	Char	25		No	
No.of Credits	CREDIT	Integer	1		Yes	
Maximum no. allowed	MAXENRL	Integer			Yes	
Faculty ID of Person teaching	FID	Char	3		No	

D. REGN TABLE -

Column Desc	Column Name	Data Type	Length	Decimal Place	Nulls Allowed	Codes
Course Number	CRSNBR	Char	6		No	
Student ID	SID	Char	3		No	
Grade for course	GRADE	Char	1		Yes	

Contents of Each Table

A. STUDENT TABLE

SID	SNAME	SEX	MAJOR	GPA
987	POIRIER	F	MGT	3.2
763	PARKER	F	FIN	2.7
218	RICHARDS	M	ACC	2.4
359	PELNICK	F	FIN	3.6
862	FAGIN	M	MGT	2.2
748	MEGLIN	M	MGT	2.8
506	LEE	M	FIN	2.7
581	GAMBREL	F	MKT	3.8
372	QUICK	F	MGT	3.5
126	ANDERSON	M	ACC	3.7

B. FACULTY TABLE

FID	FNAME	EXT	DEPT	RANK1	SALARY
036	BARGES	325	MGT	ASSO	35000
117	JARDIN	212	FIN	FULL	33000
098	KENEDY	176	ACC	ASSO	30000
075	SAMPLE	171	MKT	ASST	25000
138	WARD	125	MGT	INST	20000
219	PETERS	220	FIN	FULL	45000
151	DARDEN	250	ACC	ASSO	37000
113	PIERCE	205	MGT	INST	22000

C. COURSE TABLE

CRSNBR	CNAME	CREDIT	MAXENRL	FID
MGT630	INTRODUCTION TO MGMT	4	30	138
FIN601	MANAGERIAL FINANCE	4	25	117
MKT610	MARKETING FOR MANAGERS	3	35	075
MKT661	TAXATION	3	30	098
FIN602	INVESTMENT SKILLS	3	25	219
ACC601	BASIC ACCOUNTING	4	25	098

MGT681	INTERNATIONAL MANAGEMENT	3	20	036
MKT670	PRODUCT MARKETING	3	20	075

D. REGN TABLE

CRSNBR	SID	GRADE
MGT630	987	A
FIN602	987	B
MKT610	987	A
FIN601	763	B
FIN602	763	B
ACC610	763	B
ACC610	218	A
ACC661	218	A
MGT630	218	C
MGT630	359	F
MGT681	359	B
MKT610	359	A
MKT610	862	A
MKT670	862	A
ACC610	862	B
MGT630	748	C
MGT681	748	B
FIN601	748	A

Data Associations

The data associations between data elements are provided below for clarity.

SID <<----------> SNAME, MAJOR, SEX, GPA

SID <<---------->> CRSNBR

CRSNBR<----------> CNAME

CRSNBR<< -----------> CREDIT, MAXENRL, FID

FID <--------> FNAME, EXT

FID <<--------> DEPT, RANK1, SALARY

(STUDENT#, COURSE#)<<---------->GRADE

Normalized Relations and Business Rules

Based on the above data associations and normalization principles, the normalized relations for the four (4) 3NF relations/ tables are given below:

1. STUDENT (SID, SNAME, SEX, MAJOR, GPA)
2. REGISTRATION (CRSNBR, SID, GRADE)
3. FACULTY (FID, FNAME, EXT, DEPT, RANK1, SALARY)
4. COURSE (CRSNBR, CNAME, CREDIT, MAXENRL, FID)

Here, business rules are: One student can take many courses and one course is taken by many students. Student# is the primary key in Student relation. In the Registration relation, a grade is assigned to a student for a course. Here, SID and CRSNBR together is a composite primary key. In the Faculty relation, FID is unique and considered as a primary key. One faulty can take many courses but one course is taken by one faculty only. FID in Course relation is a foreign key with reference to FID of Faculty relation. A foreign key can be null and can contain repeat values.

Introduction to MySQL

A MySQL database server contains many databases (or schemas). Each database consists of one or more relations or tables. A relation or table is made up of columns (or fields) and rows (records). The SQL keywords and commands are NOT case-sensitive. For clarity, it can be shown in uppercase. The *names* or *identifiers* (database names, table names, column names, etc.) are case-sensitive in some systems, but not in other systems. Hence, it is best to treat *identifiers* as case-sensitive.

Data File of Four Relations and Use

In order to make the participants have knowledge on database creation, table creation, and data insertion/creation (DDL commands), database with tables (with data) are NOT provided directly/ as it is. It is recommended to understand and run Q1 to Q14 to create a

new database, create new tables in the database created and enter data to the four (4) tables: Student, Course, Registration and Faculty. Also, sometimes, without knowledge of Q1 to Q10, participants may face problems in using the data files/tables with data if given. May be one can insert more data by using DDL Insert command or by using Forms.

Note on Questions and Expected Outputs

It is expected that participants must follow the sequence of questions. For example, if the participant has not created one database and in use, then creation or use of tables in the database are not possible and that gives error messages. Similarly, if a table is not created or in use, any operation on it would provide error messages. Creating one table which is already exist in the database will provide error messages. It is recommended that participants follow sequence of questions. If someone wish to by-pass any command or set of commands, he/she must understand whether some requirement of running that command is done or not and then accordingly proceed.

Complexity of the Questions

The questions provided here in this case study are:

- Easy
- Moderate
- Difficult
- Very hard

Participants may differ their interest and knowledge on SQL. Hence category of question into the above 4 are not provided. Complexity is a function of knowledge, interest, and preparation of a participant. There are some questions, where commands might not be covered. In those situations, the participant must search, try, and explore answering those questions. Just by giving some easy questions, the purpose of the case study/ learning will not be achieved.

Questions

Q1) Display list of all the existing databases in the server.

Q2) Create a new database named “ipm”

Q3) Drop the existing database named “ipm”

Q4) Create a database name “ipm” if “ipm” does not exist as database in the server.

Q5) Remove/ delete the database “ipm” if it already exists in the server.

Q6) Create a database “ipm”.

Q7) Check whether the database “ipm” you created now, is available in the database server or not.

Q8) Assuming database “ipm” is available, set “ipm” as the default database for further application.

Q9) Confirm whether the “ipm” database you set as default database, is actually being used as current database or not.

Q10) What are the tables available in the current database “ipm” that is in use?

Q11) Create a table “student” (in the database “ipm”) with the structure/dictionary given above and insert 10 records given in the table created.

Q12) Create a table “faculty” (in the database “ipm”) with the structure/dictionary given above and insert 8 records given in the table created.

Q13) Create a table “course” (in the database “ipm”) with the structure/dictionary given above and insert 8 records given in the table created.

Q14) Create a table “regn” (in the database “ipm”) with the structure/dictionary given above and insert 18 records given in the table created.

Q15) Retrieve a list of all students.

Q16) Display student names, majors, and grade point averages.

Q17) List all data stored in student table.

Q18) What are the different majors offered at the university? List.

Q19) Display all different major (but no duplicates) at the university.

Q20) Display all (but no duplicates) major, sname (combinations) of student table.

Q21) Display a list of all accounting majors.

Q22) Display a list of all students with a GPA above 3.25.

Q23) Display a list of female students with a GPA above 3.25.

Q24) Display information (all attributes) for those students majoring in finance or any student having a GPA above 3.2.

Q25) Display student names either is an accounting major with a GPA above 3.5 or the student is a finance major.

Q26) Retrieve the names, majors, and GPA of all students who have a GPA above 3.5 and who are majoring in either accounting or finance.

Q27) List all students except those majoring in marketing (MKT) and management (MGT).

Q28) Find all students whose GPA is greater than equal to 2.4 and less than equal to 3.5.

Q29) Find all students whose GPA is greater than equal to 2.4 and less than equal to 3.5 without using any “AND” operator.

Q30) List the names of all the students who do not have a GPA in the range of 2.4 to 3.5.

Q31) List all students majoring in either accounting, management or marketing without using logical OR command.

Q32) List all students majoring in either accounting, management or marketing.

Q33) List all students who are not majoring in either accounting or management.

Q34) Retrieve the names of all the students whose last name begins with “P”.

Q35) List the name of any student whose major is unknown.

Q36) Retrieve a list of students in alphabetical order.

Q37) Retrieve a list of all students with a GPA above 3.0 ; placing the student with the highest grade point average first.

Q38) Retrieve a list of students arranged by major and, within major, arranged by the highest grade point average first.

Q39) List the names, GPA and majors of all students on the list (GPA of 3.25 or above). Arrange the output so that the student with the highest GPA appears first.

Q40) Next year every faculty member will receive a 5% salary increase. List the names of each faculty member, his/her current salary, and next years salary.

Q41) List the names of all faculty members earning a monthly salary above 3000.

Q42) What is the average salary paid in each dept?

Q43) Determine the average salary paid to the accounting faculty.

Q44) What is the total salary paid by rank in each dept? Also, within dept, by rank.

Q45) What is the total salary paid to the faculty?

Q46) Which departments have an average salary above 25000? Order the results by average salary, with highest average salary appearing first.

Q47) List the courses, faculty members teaching the courses.

Q48) In which course is FAGIN enrolled? (Assume that you can't remember FAGIN's student ID).

Q49) Provide a class roster of students enrolled in FIN601. The report should include the course no, course name, and student name.

Q50) Provide a class roster of students enrolled in FIN601. The report should include the course no, course name, and student name (Any way other than the way you answer Q.49)

Q51) List the names of all students enrolled in ‘FIN601’.

Q52) List the names of any student enrolled in INTRODUCTION TO MGMT who received an ‘F’ in the course.

Q53) List the names of the faculty members in the same department as PETERS. Assume that you do not know PETER’s department.

Q54) Retrieve the name, major, and GPA of any student whose GPA is above the average for his or her major.

Q55) Retrieve the average GPA from student where major=“MGT’.

Q56) Retrieve a list of students taking courses. Assume that some students in the STUDENT table are not enrolled in any courses this term.

Q57) List the faculty members who are not involved in teaching any courses this term.

Q58) List all management majors and any students taking courses in management (i.e. both majors and not majoring).

Q59) Add a record into student table with the following values only:
SID: 110; SNAME: JONES, SEX: M

Q60) Create a new table stu_stat with the following dictionary: SID Character (3) NOT NULL, SNAME Character (10), SEX Character(1), MAJOR Character (3), and GPA DECIMAL(3,2). Then, copy the data in student table to the stu_stat table.

Q61) A student whose ID is 763 or 748 and course number is ‘FIN601’ are dropped/ removed.

Q62) A student whose ID number is 748 leaves the University. First delete the course in which student 748 is enrolled.

Q63) Drop/remove the records from student when sex="F" and GPA is above 3.0.

Q64) Remove all courses from the REGN table.

Q65) Change the grade to F in REGN table where course no is MGT681.

Q66) All faculty members are to receive a 5% increment in salary.

Q67) To create a view named "Acc_Fac" for accounting department to see faculty Ids, names, and their phone extensions.

Q68) Retrieve fname, and ext from Acc_Fac ascending or descending by fname.

Q69) Create a view "Fac_Sum" that permits the retrieval of department salary totals and number of faculty members without disclosing individual salaries.

Q70) Retrieve department, number of faculty members and total salary from view "Fac_Sum".

Q71) Create a view "Roster" that enables the individual to visualize selected data from the STUDENT, REGN, COURSE and FACULTY tables as being one table, This view includes course number, course name, name of person teaching the course, student ID and student name.

Q72) Display course number, course name, student ID, and student name from view "Roster" for the course number "FIN601"

Q73) A faculty member, DARDEN, changes office and receives a new phone extension. The accounting secretary, using the view ACC_FAC, updates the phone extension. DARDEN's new extension is 943.

Q74) Eliminate the view "Acc_Fac" for accounting secretary.

Q75) Add a faculty advisor column "FID" with Character (3) to the STUDENT table.

Q76) Modify FID column to the STUDENT table: from Character (3) to Character (9).

Q77) Create a new table STUDENT2 with the following revised columns: SID Character (3), FNAME Character (10), LNAME Character (15), SEX Character (1), MAJOR Character (3), GPA Decimal (3,2). And then load the existing student data into the new table (STUDENT2).

Q78) Create one index named "MAJORIND" using the MAJOR column of Student to improve performance.

Q79) Create another index "MAJORIND1" using the MAJOR column of Student to improve performance, MAJOR descending.

Q80) Create another index "STUINDEX" for the student table that prevents records with the same student ID from being stored in the table.

Q81) Remove the index "STUINDEX" linked to Student table upon SID.

Q82) Combine and display the result of list SID of all female students and SID of all students who are having grade "A". In this case there may be some female students having grade "A".

Q83) Describe structure of table "student".

Q84) Write a stored procedure named “Getstudents” : list all the sname of table student.

Q85) Run the procedure Getstudents created before/ i.e. existing.

Q86) Remove the procedure Getstudents.

Q87) Without using stored procedure, display database metadata such as detail schema of the table “Student”.

Q88) Without using stored procedure, display database metadata such as detail schema of the table “Student” with ordinal position of columns.

Q89) Display columns/structure of table faculty.

Q90) Display all the column information (full information, not part) of table ‘Faculty”.

Q91) Show the execution plan (how MySQL executes the query) for select * from faculty.

Q92) Delete/ remove the able “STUDENT” from the database.

Q93) Create one table for Faculty who are in probation (FACPROB) with following structure/ dictionary and 5 records/tuples
FID Character (3) where null values are not allowed,
PROBYEARS integer values
Records:
098,4
138,8
151,10
988,11
056,2

Q94) To show the complete CREATE TABLE statement used to create table “Student”.

Q95) Can you use a select command without a table? Please write.

Q96) Display only 1st 3 rows order by Rank1 of the records from faculty table.

Q97) Skip one row followed by display of 3 rows order by Rank1 of the records from faculty table.

Q98) Define an *alias* “CourseID” for crsnbr and an alias “MaximumEnrollment” for maxenrl of the table course and use it.

Q99) Retrieve/ display the data starting from the left table and matching rows in the right table. The left join returns all rows from the left table and the matching rows from the right table. If a row in the left table does not have a matching row in the right table, the columns of the right table will have nulls. Take any two tables and write the command

Q100) Can you write Q99 any other way?

Q101) Retrieve/ display the data starting from the right table. The command must result a set that contains all rows from the right table and the matching rows in the left table. If a row in the right table that does not have a matching row in the left table, all columns in the left table will contain nulls. Take any two tables and write the command.

Q102) Can you write Q101 any other way?

Q103) Create a table IPMFA with the following structure:

FID Character (3) where null values are not allowed; FNAME Character (10) where null values are not allowed, EXT Character (3) where null values are not allowed, DEPT Character (3), RANK1 Character (4), SALARY as integer. In this table, FID is the primary key.

Q104) Create a table IPMCO with the following structure:

CRSNBR Character (6) with null values not allowed, CNAME Character (25) with null values not allowed, CREDIT as integer, MAXENRL as integer, FID Character (3) with null values not allowed. Now, introduce FID as Foreign Key and then reference to IPMFAC table considering FID of IPMFAC table and FID of IPMCO as common field.

Q105) Remove the FOREIGN KEY constraint, created in table IPMCO: See the table IPMCO.

Q106) Remove the FOREIGN KEY constraint, created in table IPMCO: Remove the foreign key constraint.

Q107) Remove the FOREIGN KEY constraint, created in table IPMCO: Execute the command to remove the key.

Q108) Introduce FID as a Primary Key to the existing Table Faculty. Note that Faculty table already exists with certain records.

Q109) Introduce FID as Foreign Key to the existing Table Course and then reference to FID of Faculty table. Course table already exists with certain records.

Q110) Display course name and faculty name who is teaching that course.

***