Phase 2

Using the provided ERD (evolved from our initial design of the ADVISE database, assignment 3, Phase I), create a relational database. Using a database package and SQL of your choice, create a set of relations which include (but are not limited to) the provided ERD.

The number, type, size and name of the attributes in each relation are provided on our web site, however, you may change them in order to improve the system. If you do change the attributes or other aspect of the schema, you should still be able to accommodate the queries listed in the "Database Queries" section of this assignment. As usual your schema design should seek to minimize redundancy, and improve performance as much as possible. Apply the Normal Forms to each relation and document if they pass 1st, 2nd and 3rd normal forms. If your tables don’t pass normal forms, explain fully why? Justify why you chose to violate the normal forms?

A) Database Creation: (DDL)

- **Review:** Create Schema, Create Table, Drop Schema, Drop Table, Alter Table, Create Domain

- Using the SQL of your choice, create the ADVISE schema. Next, create the above relations. Remember to specify the constraints such as (primary key, foreign key, Not Null, On Delete, On Update, Default, etc.) *If the SQL that you are using does not provide some of these constraints, pencil them in manually in your output.*

  - If your schema is changed, include the new ER in your output, and point out the modifications made.

  - Print the structure of each relation. (Relation name, attributes, type, keys, foreign keys, constraints, indexes, etc.)
Print the constraints which are applied to each relation. (PK, FK, etc.)

B) Database Population:

- Review: Insert into, Delete From, Update-Set-Where

1) Using SQL, populate each table with at least 5 records. (Do not use too much abbreviation when entering your data since it makes it difficult to evaluate the result of your queries.) (Note: I realize that it sometimes takes longer to do this with SQL than it does using the normal user interface provided by your database package. So, insert one or two of the records using SQL and the rest using the interface. (If you like)

   Show the SQL command which was used to populate the relations. (at least for 4 distinct insert statements in at least 2 different tables.)

2) Print the records of each relation after it is completely populated. (Use a highlighter to show the inserted records) (NOTE: Do not print the course table since it is specified by me and it has 1800 records.)

3) Delete a record from at least 2 different tables.

   Show the SQL command which was used to delete the records.

4) Print the corresponding table before and after the delete operation. (Use a highlighter to show the deleted records)

5) Update one or more records from at least 2 tables.

   Show the SQL command which was used to update the records.

6) Print the corresponding tables before and after the update operation. (Use a highlighter to show the updated records)

C) Database Queries:

- Review: Select-From-Where, IN, Group by, Having, Order by, Aggregate functions(Add, Count, Avg, Min, Max, etc.), Sub-string Comparison (Like, %, _)

- In order to test the functionality of your database implement at least 6 of the following queries in SQL.

   Q1: Display the StudentID, First name, Last name and phone number of the Advisees for Hakimzadeh

   Q2: Show the INFO Courses and Their Prerequisites/coreq/equiv etc. (show the subject area, course number (not course id), course title, the relationship (prereq, coreq, etc.), followed by the prereq/coreq, subject area, course no and course title. (First try it for CSCI courses since they are already provided)

   Q3: Display the Basic Degree Requirement Categories for All Degrees at IUSB. Sort
the categories by the order of appearance.

Q4: Display the Basic Degree Requirement Categories, and Show the Corresponding Requirements for the BS in CSCI. Sort the Result by the Order of Appearance in the Degree-requirement-category. (Note that some categories do not appear in the result.)

Q5: Display ALL the Basic Degree Requirement Categories (even those that are not required by BS in CSCI), and Show Which Categories, Have a Corresponding Requirements for the BS in CSCI and which ones do not. The ones that do not have a corresponding CS requirement will appear as NULL values in the CS column. (Hint, think LEFT JOIN)

Q6: DISPLAY THE DEGREE REQUIREMENTS (DegreeID, RequirementText, Detailed RequirementText) for BSINFBS.

Q7: DISPLAY COURSES THAT SATISFY the Degree requirements for BSINFBS and the courses that satisfy them. (DegreeID, RequirementText, Detailed RequirementText, courses which satisfy that requirement)

Q8: Print the Transcript for Studentid = 1000
Q9: Print the Degree Audit for Studentid = 1000
Q10: Your choice of query which includes 7 or more tables.
Q11: Your choice of query which includes 7 or more tables.
Q12: Your choice of query which includes 7 or more tables.

Note: You should put enough data in your database, so that none of the above queries yields an empty set.

What to hand in:
- A binder with the following information:
  - Cover page (title, name, course # and name, assignment #, date)

Section - A
- A copy of the ER used to produce the database.
- Printed Structure of all relations. (Relation name, attributes, type, keys, foreign keys, constraints, etc.) (no data) Show the SQL commands used for create these tables. (annotate if necessary)

Section - B
- Printed copy of relevant tables after insert operations. Show the SQL commands used for inserting. (annotate/highlight which records were inserted)
- Printed copy of relevant tables after each delete operation. Show the SQL commands used for deleting. (annotate/highlight which records were deleted)
- Printed copy of relevant tables after each update operation. Show the SQL commands used for updating. (annotate/highlight which records were updated)

Section - C
Type the query description. Show the SQL commands for performing that query. Print the results of each query. (annotate the output).

For this assignment you have the choice to work alone or work with a classmate. If you do choose to work with a classmate, you must implement the remaining queries provided in the “Database Queries section above. Come and talk to me to seek approval if you are planning to work as a group.
Q1: Display the StudentID, First name, Last name and phone number of the Advisees for Hakimzadeh

SELECT
  student.studentid, student.lastname, student.firstname, student.phone
FROM
  advisor, advise, student
WHERE
  advisor.lastname = "Hakimzadeh"
  AND advisor.advisorid = advise.advisorid
  AND advise.studentid = student.studentid
Q2: Show the INFO courses and their Prerequisites/CoReq/EQUIV etc. (show the subject area, course number (not course id), course title, the relationship (prereq, coreq, etc.), followed by the prereq/coreq, subject area, course no and course title. (First try it for CSCI courses since they are already provided)

```sql
SELECT
    course.subjectarea,
    course.courseno,
    course.coursetitle,
    pre_co_req.relationship,
    prereq.subjectarea as PRE_CO_REQ_SUBJECT_AREA,
    prereq.courseno as PRE_CO_REQ_COURSE_NO
FROM
    course, pre_co_req, course as prereq
WHERE
    course.deptid = 'INFO'
    AND course.courseid = pre_co_req.courseid
    AND pre_co_req.PreCoReqCourseID = prereq.courseid
ORDER BY
    course.subjectarea,
    course.courseno,
    prereq.subjectarea,
    prereq.courseno
```
Q3: Display the Basic Degree Requirement Categories for All Degrees at IUSB. Sort the categories by the order of appearance.

SELECT CategoryID, CategoryText
FROM degree_req_categories
ORDER BY orderofappearance
Q4: Display the Basic Degree Requirement Categories, and Show the Corresponding Requirements for the BS in CSCI. Sort the Result by the Order of Appearance in the Degree-requirement-category. (Note that some categories do not appear in the result.)

```
SELECT
    degree_req_categories.CategoryID,
    degree_req_categories.CategoryText,
    requirements.requirementtext,
    requirements.mincreditsrequired
FROM
    academic_program, degree_req_categories, requirements
WHERE
    academic_program.deptid = 'CSCI'
    AND academic_program.degreetypeid = 'BS'
    AND academic_program.degreeid = requirements.degreeid
    AND degree_req_categories.categoryid = requirements.categoryid
ORDER BY
    degree_req_categories.orderofappearance

Alternative:

SELECT
    degree_req_categories.CategoryID,
    degree_req_categories.CategoryText,
    requirements.requirementtext,
    requirements.mincreditsrequired
FROM
    degree_req_categories, requirements
WHERE
    degreeid = 'CSCIBS'
    AND degree_req_categories.categoryid = requirements.categoryid
ORDER BY
    degree_req_categories.orderofappearance
```
Q5: Display ALL the Basic Degree Requirement Categories (even those that are not required by BS in CSCI), and Show Which Categories, Have a Corresponding Requirements for the BS in CSCI and which ones do not. The ones that do not have a corresponding CS requirement will appear as NULL values in the CS column. (Hint, think LEFT JOIN)

```
SELECT
degree_req_categories.CategoryID,
degree_req_categories.CategoryText,
requirements.requirementtext,
requirements.mincreditsrequired
FROM
degree_req_categories
LEFT JOIN requirements
ON degree_req_categories.categoryid = requirements.categoryid
and degreeid = 'CSCIBS'
ORDER BY
degree_req_categories.orderofappearance
```
Q6: DISPLAY THE GENERAL DEGREE REQUIREMENTS (DegreeID, RequirementText, Detailed RequirementText, Minimum Credits Required, Maximum Credits Required) for BSINFBS.

SELECT
    academic_program.DegreeID,
    requirements.RequirementText,
    detailed_requirement.DetailedRequirementText,
    detailed_requirement.MinCreditsRequired,
    detailed_requirement.MaxCreditsRequired
FROM
    academic_program, requirements, detailed_requirement
WHERE
    academic_program.DegreeID = 'BSINFBS'
    AND academic_program.DegreeID = requirements.Degreeid
    AND requirements.requirementid = detailed_requirement.RequirementID
ORDER BY
    requirements.requirementid, requirements.orderofappearance
Q7: DISPLAY COURSES THAT SATISFY the Degree requirements for BSINFBS and the courses that satisfy them. (DegreeID, RequirementText, Detailed RequirementText, courses which satisfy that requirement)

SELECT
  academic_program.DegreeID,
  requirements.requirementtext,
  requirements.mincreditsrequired,
  detailed_requirement.detailedrequirementtext,
  satisfied_by.CourseID,
  course.subjectarea,
  course.courseno,
  course.coursetitle,
  course.mincredits as CREDITS,
  satisfied_by.MinGradeRequired
FROM
  academic_program, requirements, detailed_requirement, course, satisfied_by
WHERE
  academic_program.DegreeID = 'BSINFBS'
  AND academic_program.DegreeID = requirements.Degreeid
  AND requirements.requirementid = detailed_requirement.RequirementID
  AND detailed_requirement.DetailedRequirementID = satisfied_by.DetailedRequirementID
  AND satisfied_by.CourseID = course.CourseID
ORDER BY
  requirements.requirementid, requirements.orderofappearance,
  detailed_requirement.orderofappearance, course.courseno
Q8: Print the Transcript for Studentid = 1000

```
SELECT
    enrollment.academicterm,
    semester.semester,
    semester.year,
    student.studentid,
    student.Lastname,
    student.firstname,
    course.subjectarea,
    course.courseno,
    enrollment.grade,
    completion_method.description
FROM
    student, enrollment, course, completion_method, semester
WHERE
    student.studentid = '1000'
    AND student.studentid = enrollment.studentid
    AND enrollment.courseid = course.courseid
    AND enrollment.completionmethodid = completion_method.completionmethodID
    AND semester.academicterm= enrollment.academicterm
ORDER BY
    enrollment.academicterm, course.courseno
```
Q9: Print the Degree Audit for Studentid = 1000

```sql
SELECT
    student.studentid as STUD_ID,
    academic_program.DegreeID,
    requirements.requirementtext,
    requirements.mincreditsrequired as Credits,
    detailed_required.detaileddescription,
    satisfied_by.CourseID,
    course.subjectarea as Subject,
    course.courseno as CRS_NO,
    course.coursetitle,
    course.mincredits as CRED,
    satisfied_by.MinGradeRequired as Min_GRD,
    enrollment.grade,
    enrollment.completionmethodid
FROM
    declared_program, academic_program, requirements, detailed_required, satisfied_by, course
LEFT JOIN (student, enrollment)
ON
    (student.studentid = enrollment.studentid
    AND enrollment.courseid = course.courseid)
WHERE
    student.studentid = '1000'
    AND student.studentid = declared_program.studentid
    AND declared_program.DegreeID = academic_program.degreeid
    AND academic_program.DegreeID = requirements.Degreeid
    AND requirements.requirementid = detailed_required.RequirementID
    AND detailed_required.DetailedRequirementID = satisfied_by.DetailedRequirementID
    AND satisfied_by.CourseID = course.CourseID
ORDER BY
    requirements.requirementid, requirements.orderofappearance,
    detailed_required.orderofappearance, course.courseno
```
SELECT
    student.studentid as STUD_ID,
    academic_program.DegreeID,
    requirements.requirementtext,
    requirements.mincreditsrequired as Credits,
    detailed_requirement.detailedrequirementtext,
    satisfied_by.CourseID,
    course.subjectarea as Subject,
    course.courseno as CRS_NO,
    course.coursetitle,
    course.mincredits as CRED,
    satisfied_by.MinGradeRequired as Min_GRD,
    enrollment.grade,
    completion_method.description
FROM
    declared_program, academic_program, requirements, detailed_requirement, satisfied_by, course
LEFT JOIN (student, enrollment)
ON
    (student.studentid = enrollment.studentid
     AND enrollment.courseid = course.courseid)
LEFT JOIN (completion_method)
ON
    (enrollment.completionmethodid = completion_method.completionmethodID )
WHERE
    student.studentid = '1000'
    AND student.studentid = declared_program.studentid
    AND declared_program.DegreeID = academic_program_degreeID
    AND academic_program_degreeID = requirements.DegreeID
    AND requirements.requirementid = detailed_requirement_detailID
    AND detailed_requirement.DetailedRequirementID = satisfied_by.DetailedRequirementID
    AND satisfied_by.CourseID = course.CourseID
ORDER BY
    requirements.requirementid, requirements.orderno
    detailed_requirement.orderno, course.courseno