COMP 3005 Winter 2022 Assignment 4

Due: Wed. Mar. 16 by 10:00pm submitted to brightspace

Revisions:

Marking: This assignment is based on itemized requirements R1.1 - R 4.5 totaling 28 marks.

Marks are awarded, or deducted, based on requirements as follows:

Req Type	Assignment Grading
R0.x	**Critical Submission and Intent Requirements.** Assignment (or problem in some cases) gets 0 if **any** critical submission requirement (shown in red) is not met.
R0.x	**Good Practice Requirements.** You lose 2 marks for any good practice requirement (shown in amber) not met.
Rx.x	**Design Requirements.** You earn 2 marks for each design requirement (green) satisfied, well implemented, and demonstrated as requested; 1 mark if it's partly met, met but not well implemented, or met but not demonstrated; and 0 if it's not attempted or met.

Submission Requirement R0.1 Submit your answers to this assignment as a single organized .pdf document. Label the problems clearly in your document. Individual loose files will not be graded. 0 marks for the assignment if this requirement is not met.

Submission (what to submit):

Question	Devlverable to Submit
	Hand in a single PDF document for your whole assignment. (Multiple Loose files will not be graded.)
Problem 1,2,3	1)Functional Dependencies 2)Minimal Cover 3)Tables schema for a set of tables all in 3rd normal form. Show what functional dependencies apply to each table.
Problem 4	1)Attribute table with comments 2)Assumptions and constraints discussion. 3)Functional Dependencies 4)Minimal Cover 5)Tables schema for a set of tables all in 3rd

	normal form. Show what functional dependencies apply to each table.

(Marking Instructions in Red)

Total 28 marks

Problem 1 (6 marks)

Problem 2 (6 marks)

Problem 3 (6 marks)

Problems 4 (10 marks)

Functional Dependencies and your Project Database

In this assignment you will get practice with functional dependencies and present your term project database using them.

The assignment is based on the video lectures posted: Feb 16, Feb 28, Mar 7 on functional dependencies, normal forms, minimal covers and representing ER models with functional dependencies.

Problems 1,2, and 3 are to give you practice designing relational tables using only functional dependencies and minimal covers to achieve 3NF tables.

Problems 4 Is to repeat this process but for your term project database.

For each problem model the problem using only functional dependencies and then ensure that, if necessary, they are nomalized to 3NF. It might well happen that your initial dependencies are already a minimal cover representing 3NF if mapped to tables. It might also happen that you need some normalization.

In problem 3 and 4 we are viewing table design as starting with a list of attributes that someone wants to store in a database and then using functional dependencies to express relationships and constraints among those attributes. Then a minimal cover of those dependencies is found and that is used to form the 3NF tables.

These problems is based primarily on sections on functional dependencies and normal forms in the course notes.

An entity A,B,C,D,E can be represented by the functional dependency A,B->C,D,E

A weak entity A,B,C,D that is a weak entitity of strong entity E,F,G can be represented by the functional dependency A,E->B,C,D

There is a little trick to dealing with N:N relationships as a functional dependency. If X and Y are in an N:N relationship and Z is an attribute of that relationship then you can describe it as X,Y->Z. But if X and Y are in an N:N relationship that has no attributes then you can describe that is X,Y->temp. Then do the design and at the end just remove the column temp. For example, if Projects have many Employees and Employees work on many Projects then you can captures that as a functional dependency: ProjNum, EmpNum ->temp. But if the database also keeps track of the number of hours the employees work on each project then you can captures that as: ProjNum,EmpNum -> Hours.

The reason this trick is necessary is because algorithms that do normalization are allowed to remove trivial dependencies. ProjNum,EmpNum->ProjNum,EmpNum is trivial because the RHS is a subset of the LHS and will be removed. But if it is removed a table [ProjNum,EmpNum] will never be formed. So do the design with ProjNum,EmpNum -> temp.

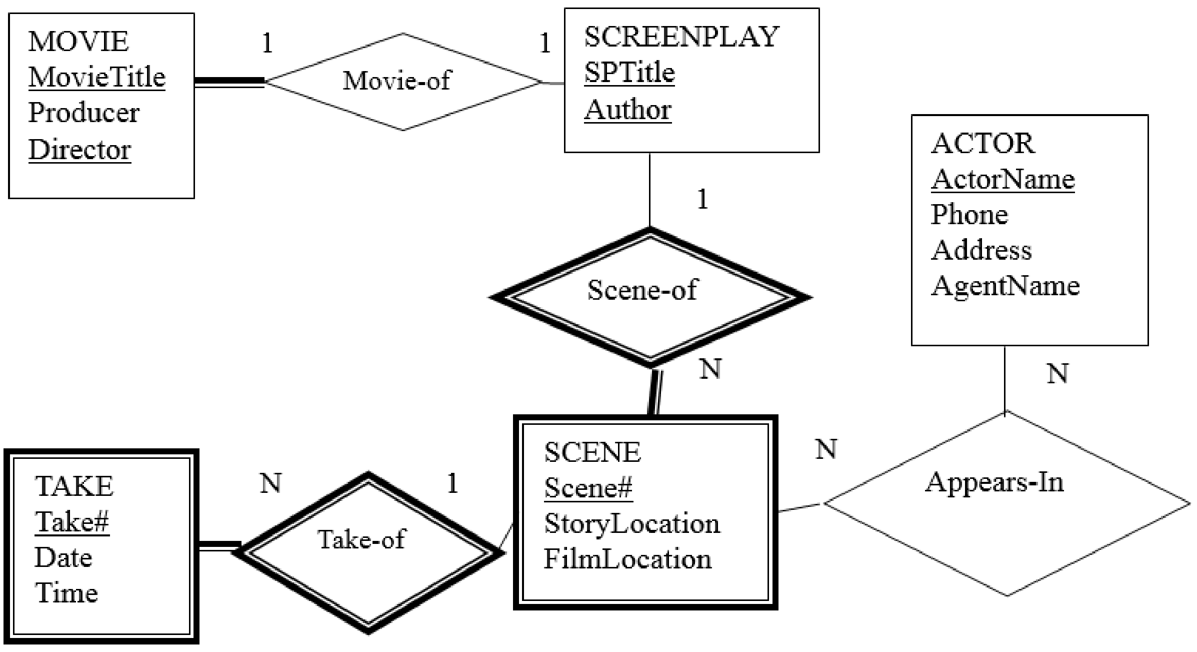
Finally be careful because if several entities have the same attribute name you will have to assign different names in the functional dependencies. When establishing a foreign key relationship to implement a relationship by "adding columns to an existing table" you might have to choose non-confilicting attribute names.

Problem 1 : Movies ER Model

Consider a Database that keeps track of scenes filmed for different movies. A movie uses a screenplay (or story) which is broken down into scenes. The movie will also have the same scenes because it is a movie of that screenplay. Not all screenplays in the database become movies, but every movie is of a particular screenplay.
Also, a screenplay is used for only one movie. That is, there are not two different movies made of the same screenplay. Scenes have a story-location where the story takes place and a filming-location where the filming will actually be done. Each scene has some actors that appear in that scene. Actors have a name, phone number, address and agent that represents them. A scene can be filmed more than once (maybe the actor forgot their lines). Each filming of a scene is called a “Take”. The movie is typically created by using the best take of each scene and putting them together. Below is an E-R diagram that captures these requirements.

Using the proposed E-R diagram provided, answer the questions related to designing the tables. (Note: thick lines denote **weak** entities or **mandatory participation** in relationships.)

Also in your functional depdencies use TakeNo and SceneNo instead of Take# and Scene# (i.e. avoid the # character).



R1.1 [2 marks] Provide a set of Functional dependencies that completely captures all the features in the situation depicted by the ER model.

2 marks if the answer is correct and complete (e.g. functional dependencies completely capture what is represented in the ER model), 1 mark if it's incomplete, 0 marks if its wrong.

R1.2 [2 marks] Provide a minimal cover for the set of functional dependencies. (Your answer from R1.1 might already be a minimal cover but you need to check and make sure.)

R1.3 [2 marks] Based on your minimal cover find a dependency preserving, 3rd normal form set of tables to use for your database that captures all of the data intended by the E-R model. Show for each table in the decomposition, its key and the functional dependencies that apply to it (that is, map to it).

Problem 2 : Fakebooks ER Model

Here is a scenario and ER model similar to the fakebook example modelled in Assignment 2 (there might be some minor changes though).

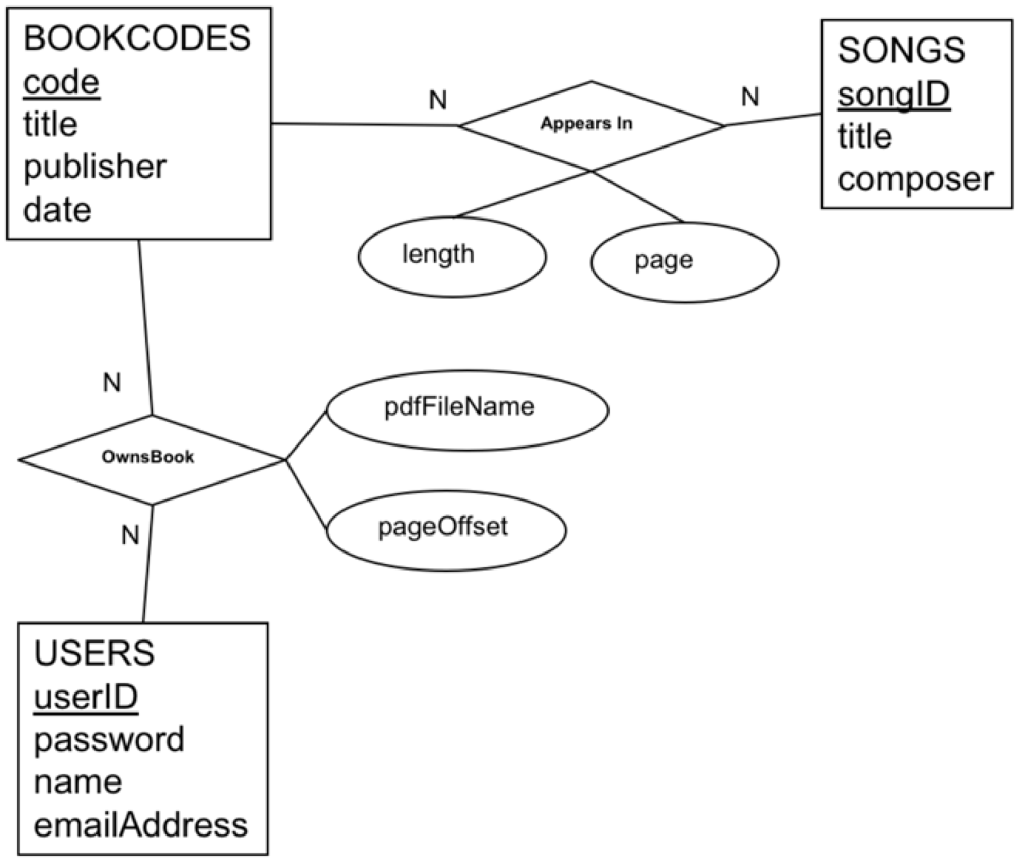
Musicians use fake book charts to play and improvise from. We want to create a database that will use the indexing information provided in assignment 1 an support the following.

The users (musicians) will upload their own copies of .pdf books to the application. The books supported by the database will be those referred to by the BOOKCODES entity. When a musician has uploaded a book file they will have the right to be shown pages of from their copy of the book using the indexing data in the database. The musicians will only be allowed to see the page of books they have uploaded, but they can search all of the indexing data. For copyright reasons they will not be allowed to see contents from other books.

The indexing information should provide the books and page numbers for the various songs. Book files (.pdf) uploaded by users book file should also have some kind of offset information to account for introductory pages in the uploaded book file. That is, if the song is indexed to be on page 1 but that is the 10th page of a particluar .pdf book file then an offset should be stored in the database to account for this. The database should store information about the books including their book code (unique), title, publisher, and date of publication.

The database should support a collection of users. Users have a name, email address, userid, password. The database must keep track of which books which users are allowed to access.

Here is a proposed E-R diagram provided for the situation described above.



R2.1 [2 marks] Provide a set of Functional dependencies that completely captures all the features in the situation depicted in the ER diagram.

R2.2 [2 marks] Provide a minimal cover for the set of functional dependencies.

R2.3 [2 marks] Based on your minimal cover find a dependency preserving, 3rd normal form set of tables use for your database that captures all of the data intended by the E-R model. Show for each table in the decomposition, its key and the dependencies that apply to it (that is, map to it).

Problem 3 : Attribute-Based Design

For this problem we want to design database tables for the following proposed set of attributes. A very common starting point for a database design is list a "catalog" of attributes that you want the database to store. The design is to be done entirely using functional dependencies. That is, capture all constraints in the form of a dependency that will eventually result in appropriate tables.

Consider the following attributes to be stored in a relational database.

attribute	comment
stdnum	student number
email	student email address
name	student name
city	student address city
strnum	student address street number
street	student address street
postcode	student address postal code
area_code	student phone number area code
office_code	student phone number office code
station_code	student phone number station code
course_num	course number e.g. COMP3005
course_name	course name e.g. Fundamentals of Databases

course_section	course section e.g. F2018-A
department_name	name of department offering course
room_num	location of a course e.g. MC2000
building	building name e.g. Minto Centre
period	time table period e.g. Tuesday 10:00-11:30
term	academic term e.g. fall2018
grade	grade student received in a course

Here are the known assumptions and constraints that need to be reflected in the functional dependencies:

1) Students can have only one name, address, phone number and email address in the database.
2) Both a student's student number and their email address uniquely identify the student.
2) A phone number area code and office code is associated with a particular city. That is, the city is a unique fact about the phone number's area code and office code combination.
3) A postal code uniquely identifies a city. That is, knowing just someone's postal code uniquely determines the city part of their address.
4) A course number does not identify a term, time or offering of the course, but a course section identifies a course offering including the year, term, and section letter. A course number identifies the department offering the course.
5) The combination of period, room, and term identifies a course section. The database should provide a way to see this scheduling information.
6) The room number identifies which building the room is in. That is, building is a unique fact about the the room number.
7) Students are enrolled in many course sections and course sections have many students enrolled in them. The database needs to keep track of this enrollment information.
8) The combination of student number and course number uniquely idenfities a grade the student received in some offering (section) of the course. Student's grades are reported for the course number. For example: a student enrolls in COMP3005W2015-B but on their transcript it just shows that they got a grade of B+ in COMP3005.

R3.1 [2 marks] Write down the functional dependencies that would capture all the requirements of the database. You may add more constraints or assumptions if needed but if so write them down with the functional dependencies.

R3.2 [2 marks] Provide a minimal cover for the set of functional dependencies.

R3.3 [2 marks] Based on your minimal cover find a dependency preserving, 3rd normal form set of tables use for your database that captures all of the intended data. Show for each table in the decomposition, its key and the dependencies that apply to it (that is, map to it).

Problem 4 : Attribute-Based Project Database

Repeat the steps and deliverables of problem 3 but for your term project database.

Specifically:

R4.1 [2 marks] Provide a table (as shown for problem 3) that lists all the attributes that need to be stored in your database. The table should list the attributes and provide enough comments with each for us to understand what they represent.

R4.2 [2 marks] Provide an **Assumptions and Constraints** section that explains all assumptions that apply to the attributes listed in your table from the previous requirement.

R4.3 [2 marks] Write down the functional dependencies that would capture all the attributes and constraints of your project database.

R4.4 [2 marks] Provide a minimal cover for the set of functional dependencies.

R4.5 [2 marks] Based on your minimal cover find a dependency preserving, 3rd normal set of tables use for your database that captures all of the intended data. Show for each table in the decomposition, its key and the dependencies that apply to it (that is, map to it).
