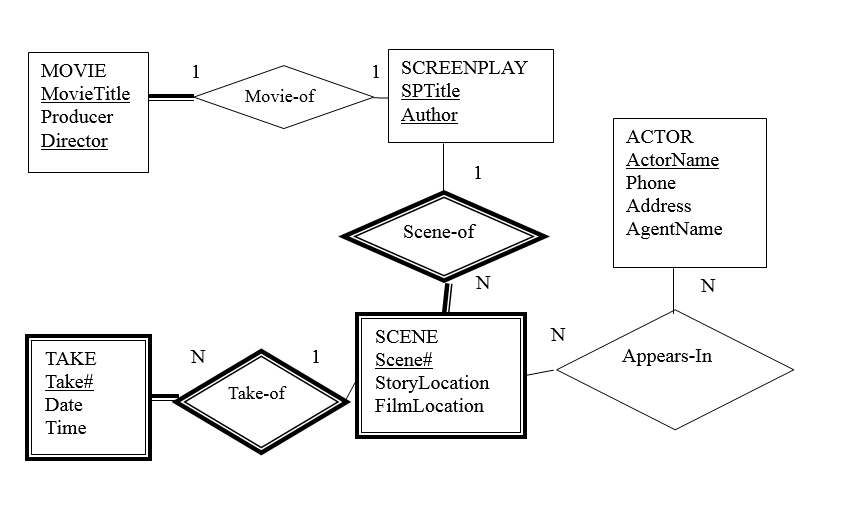
## Problem 1 : Movies ER Model



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

Functional dependencies:

MovieTitle, Director 🡪 Producer

SPTitle, Author 🡪 MovieTitle, Director, Producer

SPTitle, Author, SceneNo🡪StoryLocation, FilmLocation

SPTitle, Author, SceneNo, TakeNo🡪Date,Time

ActorName🡪Phone, Address, AgentName

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.)

F={

MovieTitle, Director 🡪 Producer

SPTitle, Author 🡪 MovieTitle, Director, Producer

SPTitle, Author, SceneNo🡪StoryLocation, FilmLocation

SPTitle, Author, SceneNo, TakeNo🡪Date,Time

ActorName🡪Phone, Address, AgentName

}

F’={

MovieTitle, Director 🡪 Producer,

SPTitle, Author 🡪 MovieTitle,

SPTitle, Author 🡪 Director,

SPTitle, Author 🡪 Producer,

SPTitle, Author, SceneNo🡪StoryLocation,

SPTitle, Author, SceneNo🡪FilmLocation,

SPTitle, Author, SceneNo, TakeNo🡪Date,

SPTitle, Author, SceneNo, TakeNo🡪Time,

ActorName🡪Phone,

ActorName🡪Address,

ActorName🡪AgentName

}

SPTitle, Author 🡪 Producer can be removed for MovieTitle, Director 🡪 Producer and SPTitle, Author 🡪 MovieTitle, Director is transitive, so the minimal cover is:

F’’={

MovieTitle, Director 🡪 Producer,

SPTitle, Author 🡪 MovieTitle,

SPTitle, Author 🡪 Director,

SPTitle, Author, SceneNo🡪StoryLocation,

SPTitle, Author, SceneNo🡪FilmLocation,

SPTitle, Author, SceneNo, TakeNo🡪Date,

SPTitle, Author, SceneNo, TakeNo🡪Time,

ActorName🡪Phone,

ActorName🡪Address,

ActorName🡪AgentName

}

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).

The 3rd normal form set of tables:

(1) Movie(MovieTitle, Director, Producer)

Key is { MovieTitle, Director } ,

The functional dependency is MovieTitle, Director 🡪 Producer

(2) ScreenPlay(SPTitle, Author, MovieTitle, Director)

Key is { SPTitle, Author } ,

The functional dependency is: SPTitle, Author 🡪 MovieTitle, Director

(3) Scene(SPTitle, Author, SceneNo, StoryLocation, FilmLocation)

Key is { SPTitle, Author, SceneNo },

The functional dependency is:

SPTitle, Author, SceneNo🡪StoryLocation, FilmLocation

(4) Take(SPTitle, Author, SceneNo, TakeNo, Date,Time)

Key is { SPTitle, Author, SceneNo, TakeNo },

The functional dependency is:

SPTitle, Author, SceneNo, TakeNo🡪Date,Time

(5) Actor(ActorName, Phone, Address, AgentName)

Key is { ActorName },

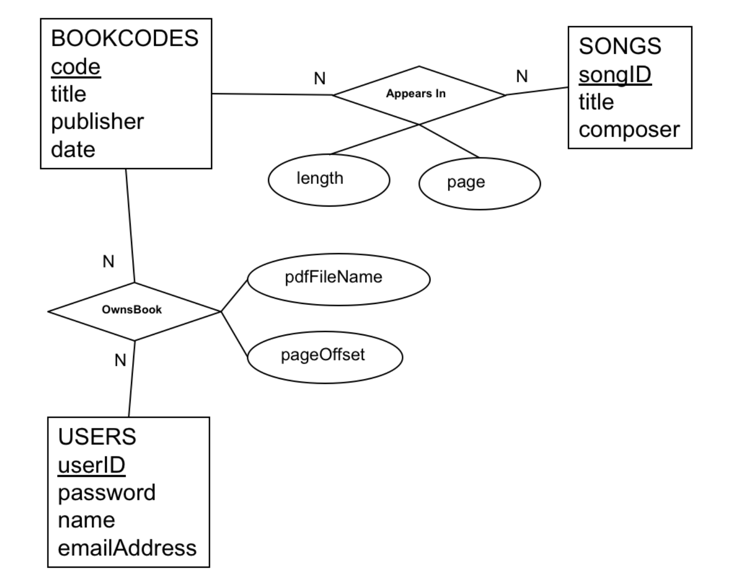
The functional dependency is:

ActorName🡪Phone, Address, AgentName

(6) AppearsIn( ActorName, SPTitle, Author, SceneNo)

Key is { ActorName, SPTitle, Author, SceneNo }

## Problem 2 : Fakebooks ER Model



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

Functional dependencies:

code🡪title, publisher, date

songID🡪title, composer

code, songID🡪length, page

userID🡪password, name, emailAddress

code, userID🡪pdfFileName, pageOffset

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

F={

code🡪title, publisher, date

songID🡪title, composer

code, songID🡪length, page

userID🡪password, name, emailAddress

code, userID🡪pdfFileName, pageOffset

}

F’= {

code🡪title,

code🡪publisher,

code🡪date,

songID🡪title,

songID🡪composer,

code, songID🡪length,

code, songID🡪page,

userID🡪password,

userID🡪name,

userID🡪emailAddress,

code, userID🡪pdfFileName,

code, userID🡪pageOffset

}

There’s no any trivial dependencies.

So F is the minimal cover.

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).

The 3rd normal form set of tables:

(1) BookCodes(code, title, publisher, date)

The key is {code},

The dependency is:code🡪title, publisher, date

(2) Songs(songID, title, composer)

The key is {songID},

The dependency is: songID🡪title, composer

(3) Users(userID, password, name, emailAddress)

The key is {userID},

The dependency is: userID🡪password, name, emailAddress

(4) AppearsIn(code, songID, length, page)

The key is { code, songID } ,

The dependency is: code, songID🡪length, page

(5) OwnsBook(code, userID, pdfFileName, pageOffset)

The key is { code, userID }

The dependency is:code, userID🡪pdfFileName, pageOffset

## 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.

functional dependencies:  
stdnum🡪email, name, city, strnum, street, postcode, area\_code, office\_code, station\_code

email🡪 stdnum,name, city, strnum, street, postcode, area\_code, office\_code, station\_code

area\_code, office\_code🡪 city

postcode 🡪city

course\_section🡪 term

course\_num🡪 course\_name, department\_name

period, room\_num, term 🡪 course\_section

room\_num 🡪 building

stdnum, course\_section,course\_num🡪term, email, name, city, strnum, street, postcode, area\_code, office\_code, station\_code, course\_name, department\_name

stdnum, course\_num 🡪grade

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

A minimal cover for the set of functional dependencies:

F={

stdnum🡪email, name, strnum, street, postcode, area\_code, office\_code, station\_code

area\_code, office\_code🡪 city

postcode 🡪city

course\_section🡪 term

course\_num🡪 course\_name

course\_num 🡪department\_name

period, room\_num, term 🡪 course\_section

room\_num 🡪 building

stdnum, course\_num 🡪grade

}

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).

(1) Student (stdnum, email, name, strnum, street, postcode, area\_code, office\_code, station\_code)

Key is stdnum.

Dependency preserving:

stdnum🡪email, name, strnum, street, postcode, area\_code, office\_code, station\_code

(2) citycode(area\_code, office\_code, city)

Key is { area\_code, office\_code }

Dependency preserving:

area\_code, office\_code🡪 city  
(3) city(postcode, city)

key is postcode.

Dependency preserving:postcode 🡪city

(4) course\_section(course\_section, term)

key is course\_section.

Dependency preserving: course\_section🡪 term

(5) course(course\_num, course\_name, department\_name)

Key is course\_num,

Dependency preserving:

course\_num🡪 course\_name, department\_name

(6)term\_room\_period (period, room\_num, term , course\_section)

Key is { period, room\_num, term}

Dependency preserving:

period, room\_num, term 🡪 course\_section

(7) room(room\_num, building)

Key is {room\_num }

Dependency preserving:

room\_num 🡪 building

(8) grade(stdnum, course\_num, grade)

Key is { stdnum, course\_num}

Dependency preserving:

stdnum, course\_num 🡪grade

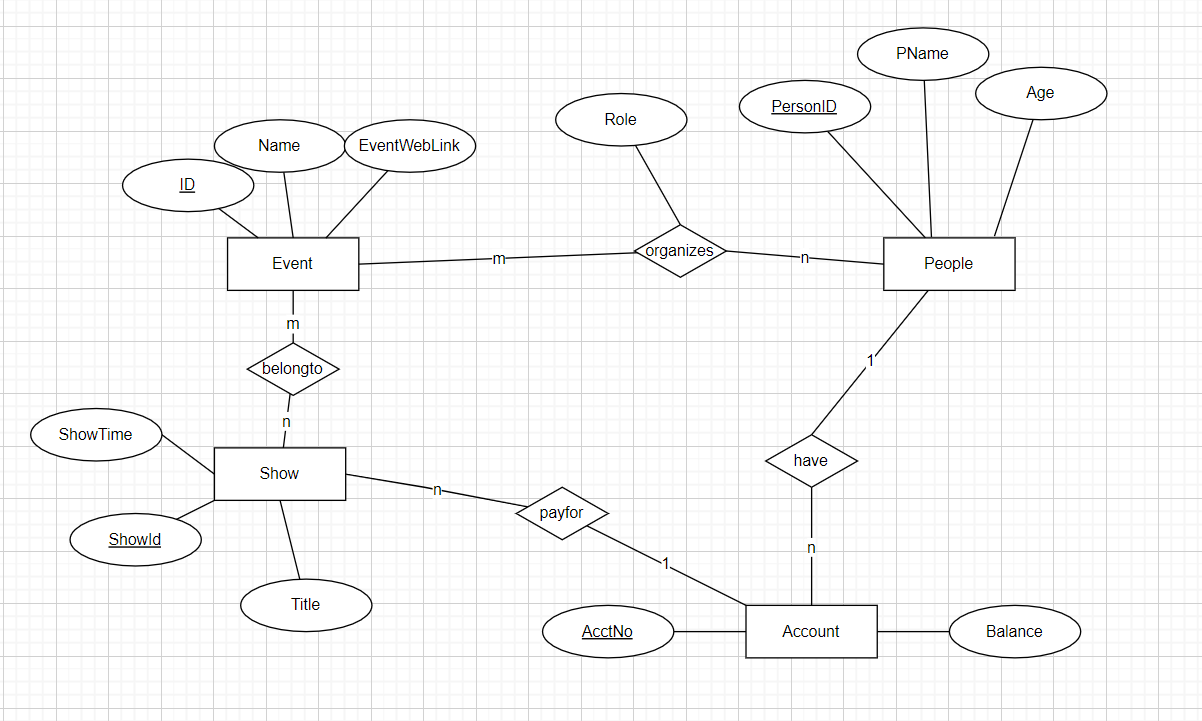
(9) enroll(stdnum, course\_section, course\_num)

Key is { stdnum, course\_num, course\_section }

## Problem 4 : Attribute-Based Project Database

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

For my term project in assignment 2 lack of N:N relationships, so here I revise my term project design, and the ER diagram is as following:



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.

|  |  |
| --- | --- |
| attribute | comment |
| ID | Event Id |
| Name | Event Name |
| EventWebLink | Event Web Link |
| Role | People organize the events, the role of people. |
| PersonID | Id of people |
| PName | Name of people |
| Age | Age of people |
| ShowID | The Id of show |
| Title | The title of show |
| ShowTime | Show time of the show |
| AcctNo | The number of account |
| Balance | The balance of account |

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.

An Event is determined by its unique ID.

Everyone is identified by PersonID.

Every person organize the event has a role. And a person can organize many events, an event can be organized by many persons.

A show is identified by ShowID.

A show can belong to many events, and an event may have many shows.

A person can have may accounts, An account can only belong to a person.

Account is identified by AcctNo.

An account can pay for many shows, and a show can only be paid by an account.

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

ID🡪Name, EventWebLink

PersonID🡪PName, Age

AcctNo🡪Balance,PersonID

ShowID🡪Title, ShowTime,AcctNo

ID, PersonID🡪Role

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

F={

ID🡪Name, EventWebLink

PersonID🡪PName, Age

AcctNo🡪Balance,PersonID

ShowID🡪Title, ShowTime,AcctNo

ID, PersonID🡪Role

}

F’={

ID🡪Name,

ID🡪EventWebLink,

PersonID🡪PName,

PersonID🡪Age,

AcctNo🡪Balance,

AcctNo🡪PersonID,

ShowID🡪Title,

ShowID🡪ShowTime,

ShowID🡪AcctNo,

ID, PersonID🡪Role

}

The F is minimal cover.

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).

Event(ID, Name, EventWebLink)

The key is {ID}.

The dependency is: ID🡪Name, EventWebLink

People( PersonID,PName, Age)

The key is {PersonID}

The dependency is: PersonID🡪PName, Age

Account (AcctNo, Balance,PersonID)

The key is {AcctNo}.

The dependency is: AcctNo🡪Balance,PersonID

Show(ShowID, Title, ShowTime, AcctNo)

The key is {ShowID},

The dependency is:ShowID🡪Title, ShowTime,AcctNo

Organizes(ID, PersonID, Role)

The key is {ID, PersonID},

The dependency is:ID, PersonID🡪Role

EventShow(ID, ShowID)

The key is {ID, ShowID},

There’s no dependency.