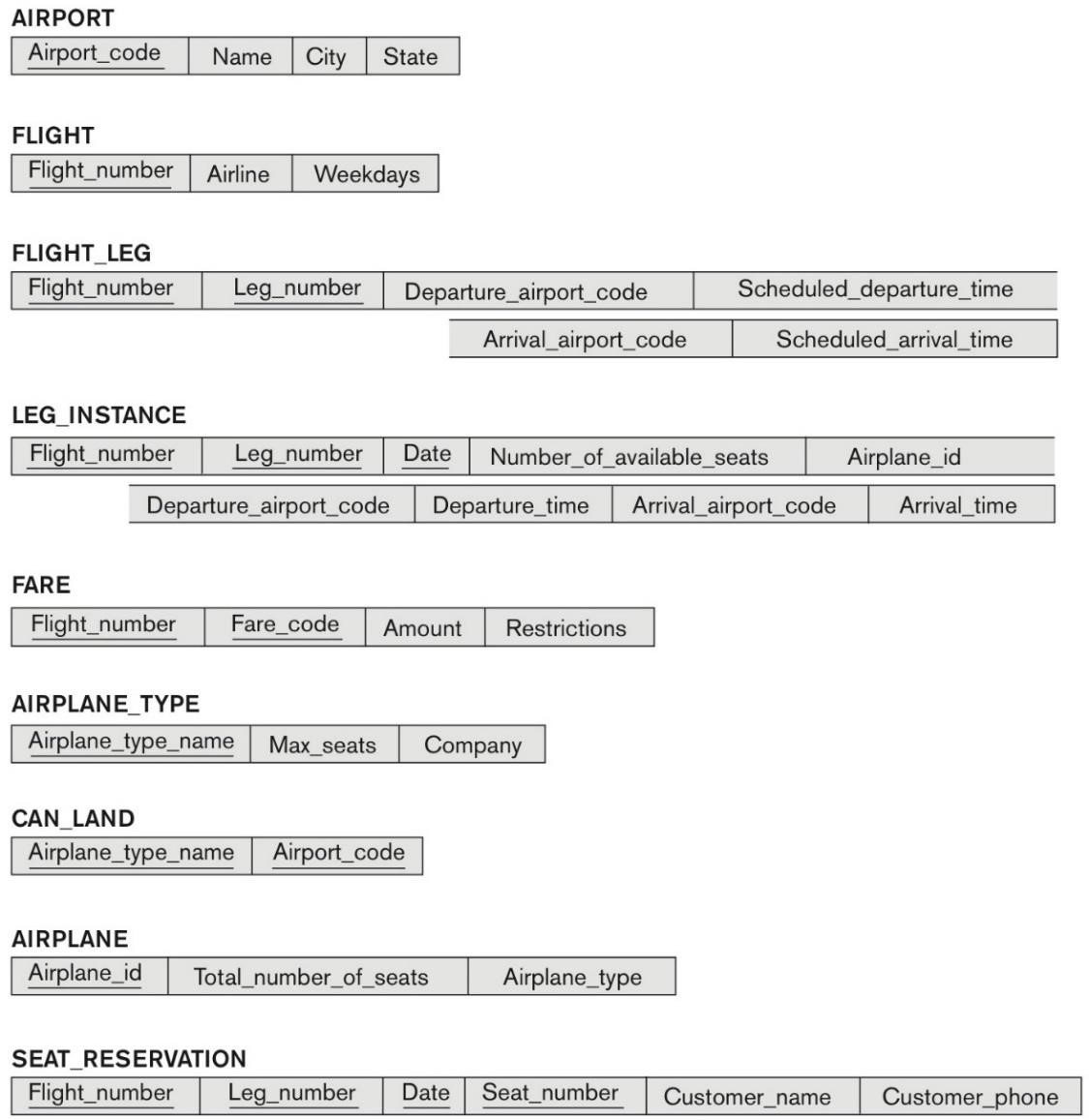
**Introduction to Databases (Spring 2020)**

**Homework #1 (40 Pts, April 29, 2020)**

**Student ID 2016312761**

**Name 여혁수**

**(1) [10 pts]** Consider the AIRLINE relational database schema, which describes a database for airline flight information. Each FLIGHT is identified by a *Flight\_number*, and consists of one or more FLIGHT\_LEGs with *Leg\_number*s 1, 2, 3, and so on. Each FLIGHT\_LEG has scheduled arrival and departure times, airports, and one or more LEG\_INSTANCEs— one for each Date on which the flight travels. FAREs are kept for each FLIGHT. For each FLIGHT\_LEG instance, SEAT\_RESERVATIONs are kept, as are the AIRPLANE used on the leg and the actual arrival and departure times and airports. An AIRPLANE is identified by an *Airplane\_id* and is of a particular AIRPLANE\_TYPE. CAN\_LAND relates AIRPLANE\_TYPEs to the AIRPORTs at which they can land. An AIRPORT is identified by an *Airport\_code*.



Specify all the referential integrity constraints that hold on the schema.

**Ex) Table\_name.column (FK) -> Table\_name for PK**

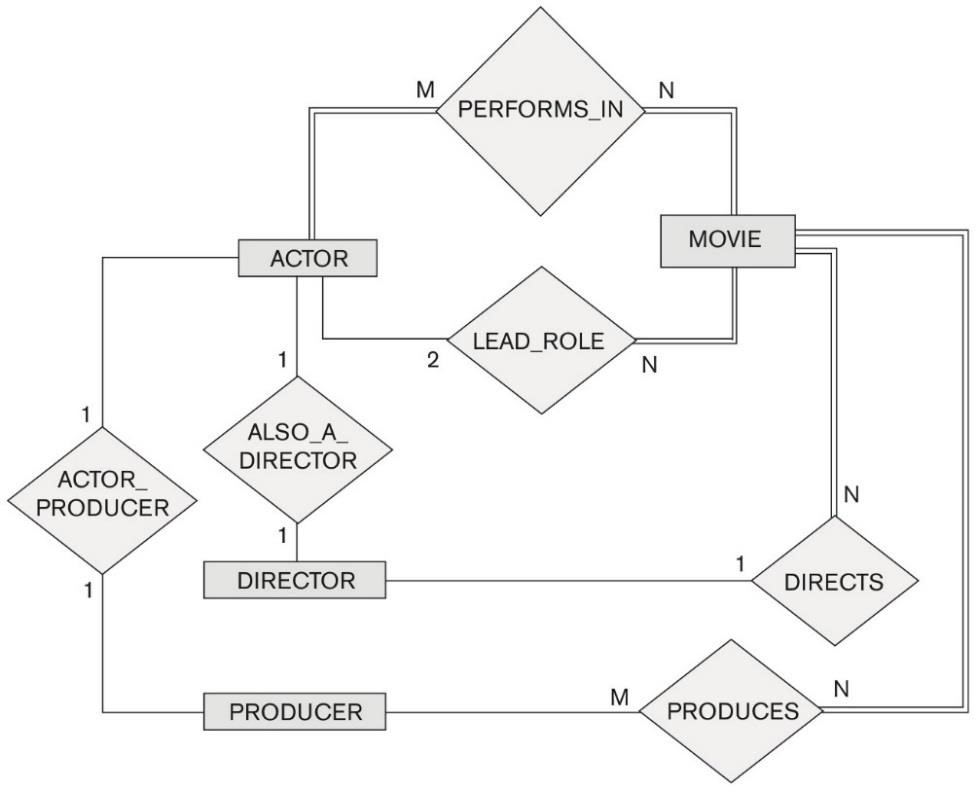
FLIGHT\_LEG.FLIGHT\_NUMBER --> FLIGHT

FLIGHT\_LEG.DEPARTURE\_AIRPORT\_CODE --> AIRPORT

# Answer

|  |
| --- |
| **FLIGHT\_LEG.Flight\_number->FLIGHT**  **LEG\_INSTANCE.Flight\_number->FLIGHT**  **LEG\_INSTANCE.Leg\_number->FLIGHT\_LEG**  **LEG\_INSTANCE.Departure\_airport\_code->FLIGHT\_LEG**  **LEG\_INSTANCE.Departure\_time->FLIGHT\_LEG**  **LEG\_INSTANCE.Arrival\_airport\_cod->FLIGHT\_LEG**  **LEG\_INSTANCE.Arrival\_time->FLIGHT\_LEG**  **FARE.Flight\_number->FLIGHT**  **SEAT\_RESERVATION.Flight\_number->LEG\_INSTANCE**  **SEAT\_RESERVATION.Leg\_number->LEG\_INSTANCE**  **SEAT\_RESERVATION.Date->LEG\_INSTANCE**  **AIRPLANE.Airplane\_type->AIRPLANE\_TYPE**  **LEG\_INSTANCE.Airplane.id->AIRPLANE**  **CAN\_LAND.Airplane\_type\_name->AIRPLANE\_TYPE**  **CAN\_LAND.Airport\_code->AIRPORT** |

**(2)** [**10 pts**] Consider the ER schema for the MOVIES database as below. In the ER schema, ACTOR is used as a generic term and includes actresses. Given the constraints shown in the ER schema, answer the following statements with *true* or *false*. Explain why.



**(a)** There are no actors in this database that have been in no movies.

# Answer

|  |
| --- |
| **True, because ACTOR entity has a relationship that is total participation with MOVIE entity.** |

**(b)** A movie can have no lead actor.

# Answer

|  |
| --- |
| **False, In LEAD\_ROLE relationship, MOVIE entity has a relationship that is total participation with ACTOR entity. That means all tuples of MOVIE indicates some of tuples of ACTOR.** |

**(c)** Every director has been an actor in some movie.

# Answer

|  |
| --- |
| **False, In ALSO\_A\_DIRECTOR relationship, it is 1:1 cardinality ratio relationship, and DIRECTOR entity doesn’t have total participation to ACTOR entity. So there are some director who have not been an actor.** |

**(d)** An actor can be a director and producer at the same time.

# Answer

|  |
| --- |
| **True, reason why an actor can be a director explained by question(c), and also ACTOR entity has 1:1 relationship with PRODUCER entity, so an actor can be a director and producer at the same time.** |

**(e)** Every producer has been an actor.

# Answer

|  |
| --- |
| **False, In ACTOR\_PRODUCER relationship, PRODUCER entity is not total participation to ACTOR entity. So some producers might have not been an actor.** |

**(f)** A movie has one director, but many producers.

# Answer

|  |
| --- |
| **True, it is true that whichever we chose a movie, it has only one director because MOVIE entity has N:1 & total participation relationship with DIRECTOR relationship. Also, a movie can have many producers, because MOVIE and PRODUCER entities have M:N relationship.** |

**(g)** Every movie has a director who also acted in that movie.

# Answer

|  |
| --- |
| **False, by solution of question(c), there can be some director who is not actor. In same way, every movie can’t have a director who also acted in that movie.** |

**(h)** A producer cannot be an actor in some other movie.

# Answer

|  |
| --- |
| **False, PRODUCER & ACTOR entities have 1:1 relationship, so a producer can be an actor and by solution of question(a), all actors participate in some movie. So, a producer can be an actor in some other movie.** |

**(i)** Every producer should produce one or more movies.

# Answer

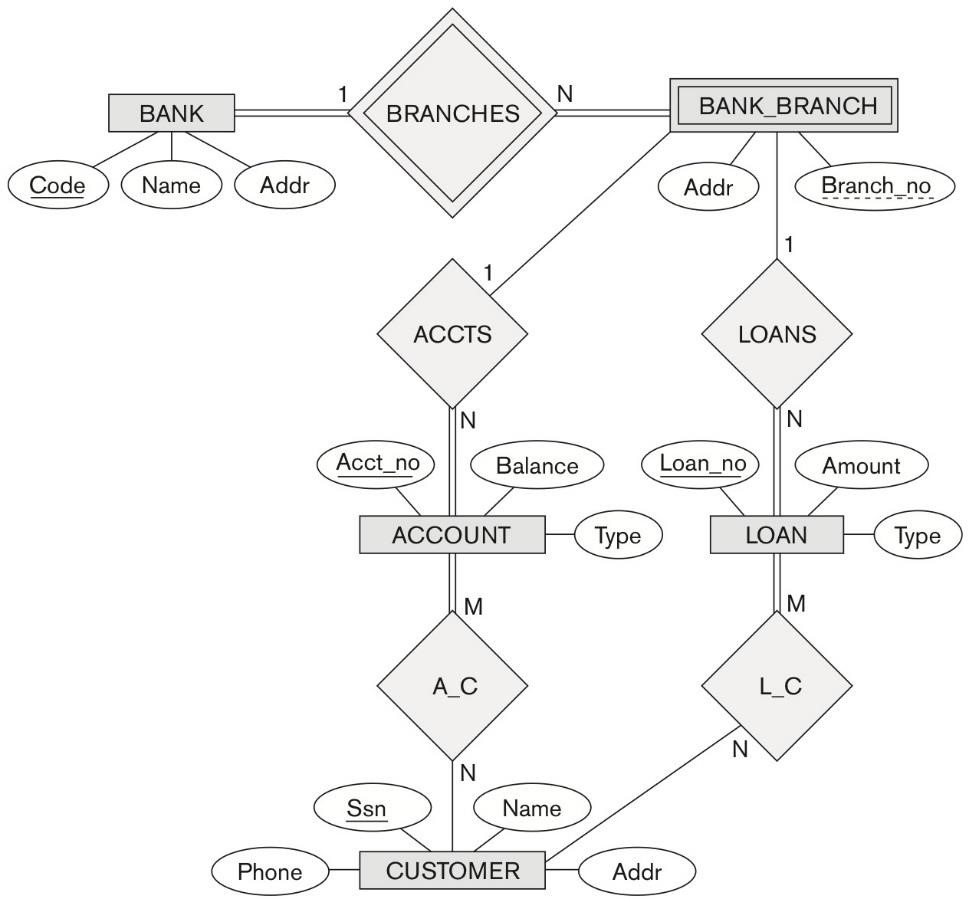
|  |
| --- |
| **False, PRODUCER entity has partial participation with MOVIE entity. So there is a producer who does not make any movie.** |

**(j)** A movie can have two lead actors.

# Answer

|  |
| --- |
| **True, because ACTOR & MOVIE entities have 2:N relationship with total participation of MOVIE entity in LEAD\_ROLE relationship. So there is a movie which has two lead actors.** |

**(3) [10 pts]** Consider the following ER diagram for the BANK database. Each bank can have multiple branches, and each branch can have multiple accounts and loans.



**(a) [1 pt]** List the regular entity types in the ER diagram.

# Answer

|  |
| --- |
| **[BANK, ACCOUNT, LOAN, CUSTOMER]** |

**(b) [2 pt]** Is there a weak entity type? If so, specify the primary key for the weak entity type.

# Answer

|  |
| --- |
| **Yes, BANK\_BRANCH entity is weak entity. It has weak relation with BANK entity, so primary key is an attribute named Code and Branch\_no, which is primary key of BANK entity and BANK\_BRANCH entity each.** |

**(c)** [**7 pts**] Map the BANK diagram into a relational model. For each relation, specify all primary keys and foreign keys.

**Answer:**

BANK

|  |  |  |
| --- | --- | --- |
| Code | Name | Addr |

BANK\_BRANCH

|  |  |  |
| --- | --- | --- |
| BCode | Branch\_No | Addr |

ACCOUNT

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AcctNo | Balance | Type | BCode | Branch\_No |

Customer

|  |  |  |  |
| --- | --- | --- | --- |
| SSN | Name | Phone | Addr |

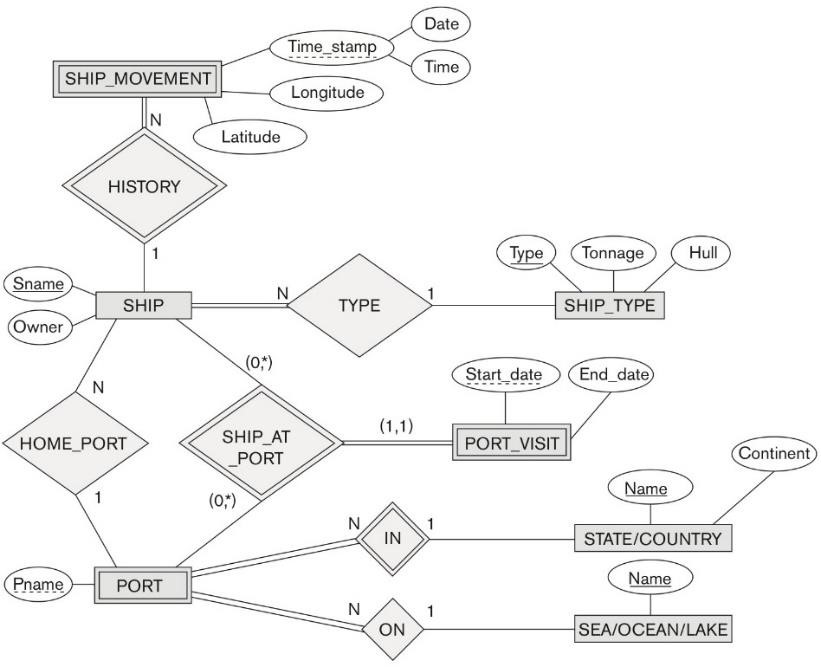
LOAN

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| LoanNo | Amount | Type | BCode | Branch\_No |

A\_C

|  |  |
| --- | --- |
| SSN | ACCTNo |
| L\_C |  |
| SSN | LoanNo |

**(4) [10 pts]** Consider the ER diagram for SHIP\_TRACKING database to keep track of transport ships and their locations for maritime authorities. Map this diagram to a relational schema (SHIP, SHIP TYPE, PORT, PORT\_VISIT, SHIP\_MOVEMENT, STATE\_COUNTRY, SEA\_OCEAN\_LAKE), and specify all primary keys and foreign keys for each relation.



# Answer

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SHIP\_MOVEMENT**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | Sname | Longitude | Latitude | Date | Time |   **SHIP**   |  |  |  |  | | --- | --- | --- | --- | | SName | Owner | Type | Pname |   **SHIP\_TYPE**   |  |  |  | | --- | --- | --- | | Type | Tonnage | Hull |   **PORT**   |  |  |  | | --- | --- | --- | | Pname | State\_Name | Sea\_Name |   **PORT\_VISIT**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | Start\_date | End\_date | Sname | Pname | State\_name |   **STATE/COUNTRY**   |  |  | | --- | --- | | Name | Continent |   **SEA/OCEAN/LAKE**   |  | | --- | | Name | |