Lecture 6 Notes & Group Practice

**7-step mapping**

* Step 1: Mapping of Regular Entity Types
  + For each regular entity type, create a relation *R* that includes all the **simple** attributes (don’t include multivalue) of *E -* Called **entity relations**
  + If chosen key of E is composite, the set of simple attributes will be the primary key together
* Step 2: Mapping of Weak Entity Types
  + For each weak entity type, create a relation *R* and include all **simple** attributes of the entity type as attributes of *R*
  + Foreign key: **primary key attribute of owner(s)**
  + If a weak entity E2 whose owner is also a weak entity E1, then E1 should first be mapped
  + Primary key: **primary key attribute of owner(s) + weak entity’s partial key (if any)**
* Step 3: Mapping of Binary 1:1 Relationship Types
  + For each binary 1:1 relationship type, Identify **relations** that correspond to **entity types** (S & T) participating in *R*
  + **Foreign key approach – most useful/common**
    - Choose one of the relations S
    - S’s foreign key: T’s primary key
    - S’s primary key: stay the same
    - add R’s **simple** attributes -> S
    - Better to choose an entity type with **total participation** in R in the role of S
* Step 4: Mapping of Binary 1:*N* Relationship Types
  + For each regular binary 1:*N* relationship type
  + Identify relation that represents participating entity type at *N*-side of relationship type (assume S)
  + S’s foreign key: T’ primary key of other entity type as in *S*
  + S’s primary key: stay the same
  + add R’s simple attributes -> S
* Step 4 alternative approach:
  + Use the **relationship relation** (cross-reference) option same as in the third option for binary 1:1 relationships
  + N side PK becomes PK of the **relationship relation**
  + Personal preference: I sometime use it when the relationship has many attributes
* Step 5: Mapping of Binary *M*:*N* Relationship Types
  + For each binary *M*:*N* relationship type
  + Create a new relation *R*
  + R’s foreign keys: primary key of S & T
  + R’s primary key: combination of foreign key attributes
  + Add simple attributes of the relationship to R
* Step 6: Mapping of Multivalued Attributes
  + For each multivalued attribute A
  + Create a new relation M
  + M’s foreign key: K (primary key of the entity type having A)
  + M’s primary key: *A* + *K*
  + If the multivalued attribute is composite, include its simple components
* Step 7: Mapping of *N*-ary Relationship Types
  + For each *n*-ary relationship type *R*
  + Create a new relation *N* to represent *R*
  + N’s foreign keys: primary keys of participating entity types
  + N’s primary key: **usually** combination of all foreign keys; if E side ratio is 1, doesn’t include E’s primary key
  + simple attributes -> N

1. In previous practice, we got 2 possible ER diagrams from the following requirements. Following mapping steps, create relations from both ER diagrams.

The US Film Institute collects statistics about movies and moviegoers in USA. These statistics are stored in a database.

 Each movie has a unique number, and also a name, a production year (movie name and production year combined is unique for movie too), and a country of origin. Each movie theater has a unique number, and also a name, a city and a state where it is located. Each movie theater contains one or more screens (in separate rooms), where the movies are shown. Each such screen has a number, which is unique within that theater, and a name. For each screen, we need to store data about the movie shows that have occurred: which movie that was shown, at what time, and how many visitors there were.

Draw an E/R diagram using the notations we discussed in the class for the database described above. Need to show the cardinality constraints and participation constraint. If you need to make any extra assumptions when making the E/R diagram, then state these assumptions.

ER diagram 1:

Relational data model:

MOVIE



THEATER









ER diagram 2:



Relational data model:

MOVIE



THEATER



Using cross reference approach (as the relationship has many attributes):

ONSCREEN

