1. Software Development Process

2. Three Different ER Models of the Same Enterprise

3. Cardinalities in ERD’s

4. Relationships with Attributes

5. Developing an ERM from Requirements (3 pages)

6. Mapping ERD to RDM (3 pages)
## Software Development Process

<table>
<thead>
<tr>
<th>Stage</th>
<th>Should include</th>
<th>How we did it for our projects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements</td>
<td>Combination of</td>
<td>one-paragraph description</td>
</tr>
<tr>
<td></td>
<td>• user interviews</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• forms needed</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• reports needed</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• queries anticipated</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• use cases</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• business rules</td>
<td></td>
</tr>
<tr>
<td>Design</td>
<td>One of these:</td>
<td>skipped this part so far, but note phase 11, coming up</td>
</tr>
<tr>
<td></td>
<td>• ER model</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• EER model</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• UML</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• semantic data model</td>
<td></td>
</tr>
<tr>
<td>Implementation</td>
<td>One of these:</td>
<td>a poorly-planned list improved by the team to a set of tables (hence, a relational db)</td>
</tr>
<tr>
<td></td>
<td>• hierarchical db</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• network db</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• relational db</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• object db</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• object-relational db</td>
<td></td>
</tr>
<tr>
<td>Maintenance</td>
<td>• feedback loops</td>
<td>beyond the scope of our course</td>
</tr>
<tr>
<td></td>
<td>• performance measurements</td>
<td></td>
</tr>
</tbody>
</table>
Three different ER approaches to model the same story – equally correct?

Start with this basic situation:

Now suppose we want to add the information about who is the manager of each department.

1) We can add the manager as one more attribute to the Department entity class:

2) We can add another relationship between the two original entity classes:

3) We can add another entity class for just those employees who are managers:
a) For a given doctor, is s/he limited to having appointments with just one patient?
   • If so, then a is 1.
   • If not, then a is N (unless b is N; then use M for a).

b) For a given patient, is s/he limited to having appointments with just one doctor?
   • If so, then b is 1.
   • If not, then b is N (unless a is already N; then use M for b).

c) For a given doctor, is it necessary for her/him to have at least one appointment with some patient?
   • If so, then put a vertical line where c is.
   • If not, then put an oval there.

d) For a given patient, is it necessary for her/him to have at least one appointment with some doctor?
   • If so, then put a vertical line where d is.
   • If not, then put an oval there.
Sometimes Relationships Need Attributes, Too!

Suppose we have this situation:

<table>
<thead>
<tr>
<th>EMPLOYEES</th>
<th>M:N</th>
<th>PROJECTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>empID</td>
<td>WorkOn</td>
<td>pjNo</td>
</tr>
<tr>
<td>empName</td>
<td></td>
<td>startDate</td>
</tr>
<tr>
<td>DoB</td>
<td></td>
<td>budget</td>
</tr>
<tr>
<td>salary</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

And now we'd like to record the number of hours per week each employee works on each of their projects.

The only place that makes sense is to include numHours as an attribute of WorkOn, not of one of the entity classes:
CSC 1035
Developing an ERD from the requirements

We’ve been given a company to create a database for.

The company has employees, who have unique id’s. We also keep track of their name and date of birth. They must work in a unique department, and they might be working on various projects. We note how many hours they spend on each one.

Projects are identified by their project number. They also have a name and a location. Each project is also controlled by a unique department.

Departments have a name and an address. They might control some projects.
The ERD – Initial version

EMPLOYEE
- empID
- empName
- DoB
- key: empID

PROJECT
- projNumber
- location
- projName
- key: projNumber

WORKS_ON
- hoursPerWeek

IS_IN

DEPARTMENT
- deptName
- dAddress
- key: deptName

CONTROLS
The ERD – Final version

- **EMPLOYEE**
  - empID
  - empName
  - DoB
  - key: empID

- **PROJECT**
  - projNumber
  - location
  - projName
  - key: projNumber
  - WORKS_ON
    - hoursPerWeek

- **DEPARTMENT**
  - deptName
  - dAddress
  - key: deptName

- **IS_IN**
- **CONTROLS**
  - 1:N

The diagram shows the relationships between the entities EMPLOYEE, PROJECT, DEPARTMENT, and WORKS_ON.
CSC 1035
Guide for mapping ERD to RDM

Start with an ERD, including attributes and keys for the entity classes, and attributes where appropriate for relationship classes. End up with an RDM: tables, with primary and foreign keys identified.

99% of the time we need consider only the following three situations: entity classes, M:N relationship classes, and other relationship classes.

1 - ENTITY CLASSES) Each entity class is mapped to a table of the same name, the same attributes, and the same primary key.

2 - M:N RELATIONSHIP CLASSES) Each one is mapped to a table of its own. The attributes of this table are the primary keys of the two tables that it connects, plus any indigenous attributes of its own. Its primary key is composite and consists of the two keys of the tables it connects. Each of those keys is a FK referencing its respective table.

3 - OTHER RELATIONSHIP CLASSES) Take the PK from a table next to a “1,” and add it as a FK to the other side. If the relationship is 1:1, just do it on one side.

NOTE: If the same attributes are about to appear in the same table and play different roles there, then we need to do some renaming. This doesn’t occur in the following example.
Example for mapping an ERD to a RDM – the ERD:

EMPLOYEE
- empID
- empName
- DoB
- key: empID

PROJECT
- projNumber
- location
- projName
- key: projNumber

WORKS_ON
- hoursPerWeek

DEPARTMENT
- deptName
- dAddress
- key: deptName

IS_IN
M:N

Example for mapping an ERD to a RDM – the RDM:

DEPARTMENT (deptName, dAddress)

EMPLOYEE (empID, empName, DoB, whichDept)

WORKS_ON (workerID, projNumber, hoursPerWeek)

PROJECT (projNumber, location, projName)