IS 300 — Lectures 10/11

- What are entities and relationships?
- How do we model entities and relationships?
- How is an ER Diagram related to a relational database?
- How are correlation tables and associative objects similar? How are they different?
- How do we find problems with record structure diagrams?
- Some problems for study.

What are entities and relationships?

Entities

- What are they?
  - Tangible Things
  - Roles
  - Incidents/Interactions

- Assumptions
  - Can distinguish between two instances
  - Generally include more information beyond a key

- Attributes
  - Named fields
  - Values change from instance to instance

- Examples
  - Product = { Prod-No + Price } — thing
  - Student = { St-No + Name + Addr } — role
  - Purchase = { Cust-No + Prod-No + Qty } — interaction

Relationships

- What are they?
  - Association between two entities

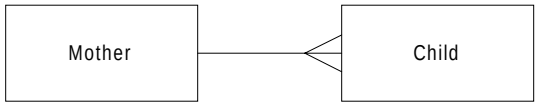
- Cardinality
  - One-to-one (Car / License)
  - One-to-many (Mother / Child)
  - Many-to-many (Student / Course)

How do we model entities and relationships?

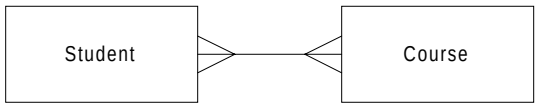
Entity-Relationship (ER) Diagram

1-1

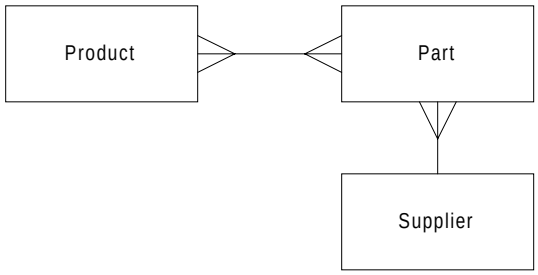
1-M



M-M



Example



How is an ER Diagram related to a relational database?

Record Structure Diagram

A tool that helps convert an ER Diagram into the tables of a relational database.

Finding keys in a table

Think about the actual data

{ St-No + Name + Age }

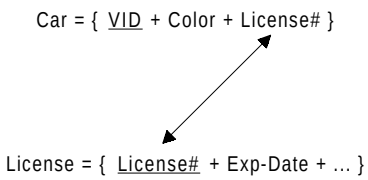
{ Part-No + Supp-No + Qty }

{ St-No + Course-Id + Qtr + Grade }

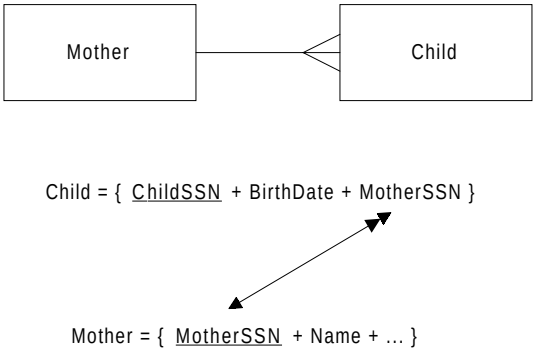
{ Empl-No + ProjNo + Name }

RSD Rules

1-1 Take key field from one of the entities and add it to the other (makes a foreign key).

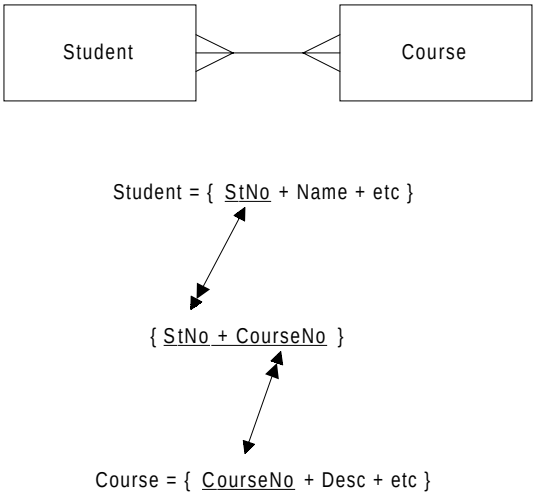


1-M Take the key field from the one entity and make it a foreign key in the many entity.



What does the data look like here?

M-M Create a new table (file/relation) that includes the key fields from each entity. This new table may/may not include other attributes.



What does this data look like?

Is the { StNo + CourseNo } table an object?

Can you distinguish between two entities? Yes

Are there attributes in addition to the key? No

How are correlation tables and associative objects similar? How are they different?

Correlation Tables

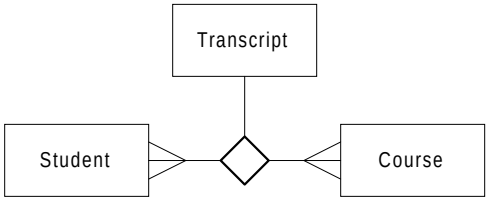
- Contains the data to support M-M relationship (two keys)
- Not shown on ER Diagram because we already know they are there
- Shown on RSD because these diagrams show all data

Associate Objects

- Start out as correlation tables
- Include at least one attribute beyond correlation table keys
- Shown on ER Diagram because they are more that a correlation table
- Examples

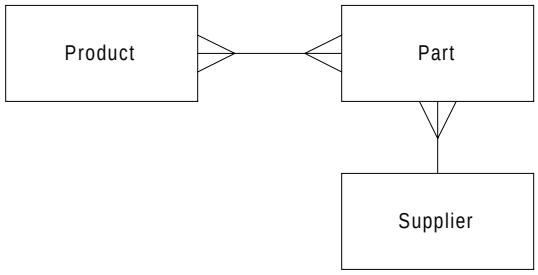
{ StNo + CourseId + Grade }

{ OrderNo + ProdNo + Qty }

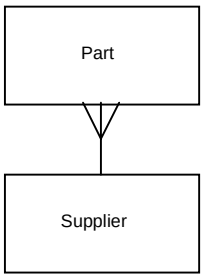


RSD from the earlier problem

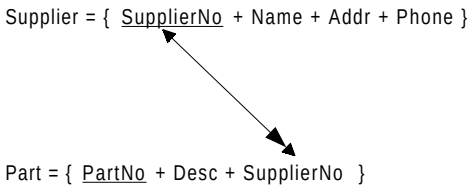
From earlier:



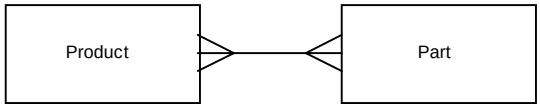
Start with 1-M relationship



1-M Take the key field from the one entity and make it a foreign key in the many entity.



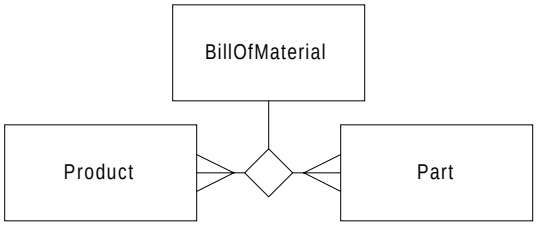
Next do the M-M relationship



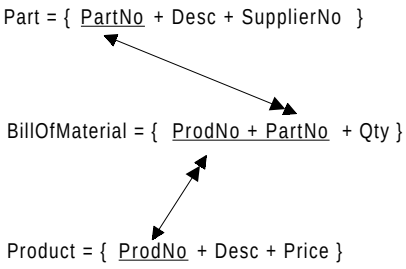
M-M Create a new table (file/relation) that includes the key fields from each entity. This new table may/may not include other attributes.

From problem statement: "Some parts are used more than once in the same product. As a result there is a need to keep track of the quantity of each part found in each product."

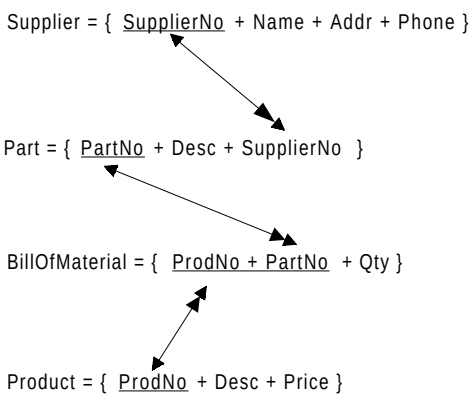
Revised ER Diagram:



RSD



Final RSD


Supplier = { SupplierNo + Name + Addr + Phone }
Part = { PartNo + Desc + SupplierNo }
BillOfMaterial = { ProdNo + PartNo + Qty }
Product = { ProdNo + Desc + Price }

How do we find problems with Record Structure Diagrams?

Repeat Groups - Bad

{ ASUW# + Name + MajCode + MajDesc + CourseId + CourseGrade }

The course information repeats for same student for each course taken.

Solution – break into two tables (results in a 1-M relationship here):

{ ASUW# + Name + MajCode + MajDesc }

{ ASUW# + CourseId + CourseGrade }

Partial/Full Dependencies

A functional dependency between fields A and B exists if, for a given value of A, you can determine at most one value for B. For example, for ASUW# and Name, a functional dependency exists if you can determine at most one Name given a specific value for ASUW#.

A partial dependency (bad) – a non-key value is determined by a subset of a compound key.

{ FlightNo + Date + DepartTime + CurrentCount }

Here DepartTime is determined by just the FlightNo so there is a partial dependency.

A full dependency (good) – a non-key value is determined by the entire compound key.

In the example above, the CurrentCount is determined by both fields in the compound key.

Solution to a partial dependency – break into two tables (1-M):

Before: { FlightNo + Date + DepartTime + CurrentCount }

After:

{ FlightNo + Date + CurrentCount }
{ FlightNo + DepartTime }

Transitive Dependencies - Bad

A transitive dependency exists when one non-key field is dependent on another non-key field.

{ ASUW# + Name + MajCode + MajDesc }

Here if you know the value for major code, you can determine the major description.

Solution to transitive dependencies – break into two tables (1-M).

{ ASUW# + Name + MajCode }
{ MajCode + MajDesc }

A set of data with all the problems:

{ ASUW# + Name + MajCode + MajDesc + CourseId + CourseDesc +
Date + CourseGrade }

First remove repeat group:
File1: { ASUW# + Name + MajCode + MajDesc }
File2: {ASUW# + CourseId + CourseDesc + Date + CourseGrade }

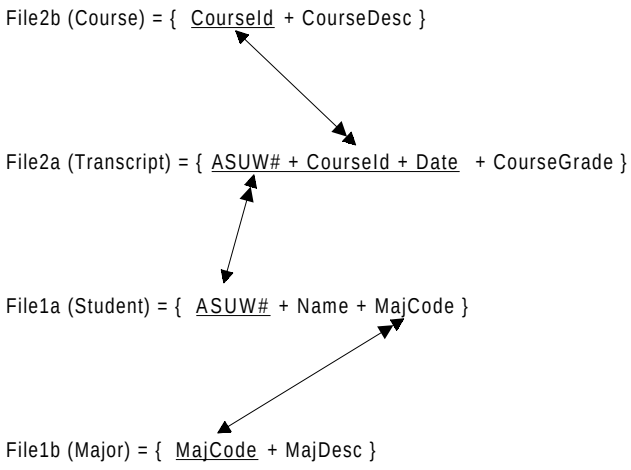
Now remove the partial dependencies (look at File1 and File2):
File1: *no partial dependencies*

File2a: {ASUW# + CourseId + Date + CourseGrade }
File2b: { CourseId + CourseDesc }

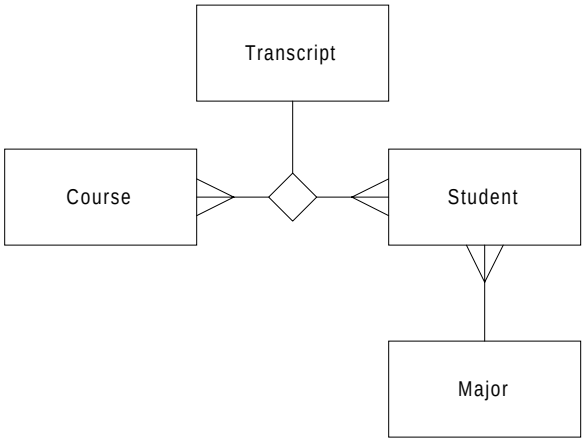
Finally remove transitive dependencies (look at files 1, 2a, 2b):
File1a: { ASUW# + Name + MajCode }
File1b: { MajCode + MajDesc }

File2a: *no transitive dependencies*
File2b: *no transitive dependencies*

Final RSD:



Final ER Diagram:



- Some problems for study.

(see problem set discussed in class)