Informatics 1: Data & Analysis

Lecture 4: From ER Diagrams to Relational Models

Ian Stark

School of Informatics
The University of Edinburgh

Friday 25 January 2013
Semester 2 Week 2

Lecture Plan

Data Representation

This first course section starts by presenting two common data representation models.

- The *entity-relationship (ER)* model
- The *relational* model

Data Manipulation

This is followed by some methods for manipulating data in the relational model and using it to extract information.

- *Relational algebra*
- The *tuple-relational calculus*
- The query language *SQL*

Lecture Plan

Data Representation

This first course section starts by presenting two common data representation models.

- The *entity-relationship (ER)* model
- The *relational* model

Data Manipulation

This is followed by some methods for manipulating data in the relational model and using it to extract information.

- *Relational algebra*
- The *tuple-relational calculus*
- The query language *SQL*

The Story so Far

Entity-Relationship diagrams

- Entity sets: Rectangles
- Relationships: Diamonds linked to the entity sets
- Attributes: Ovals linked to entity or relationship sets
- Key constraints as arrows; total participation as thick lines
- Weak entities with their identifying relationship; Entity hierarchies

Relational models

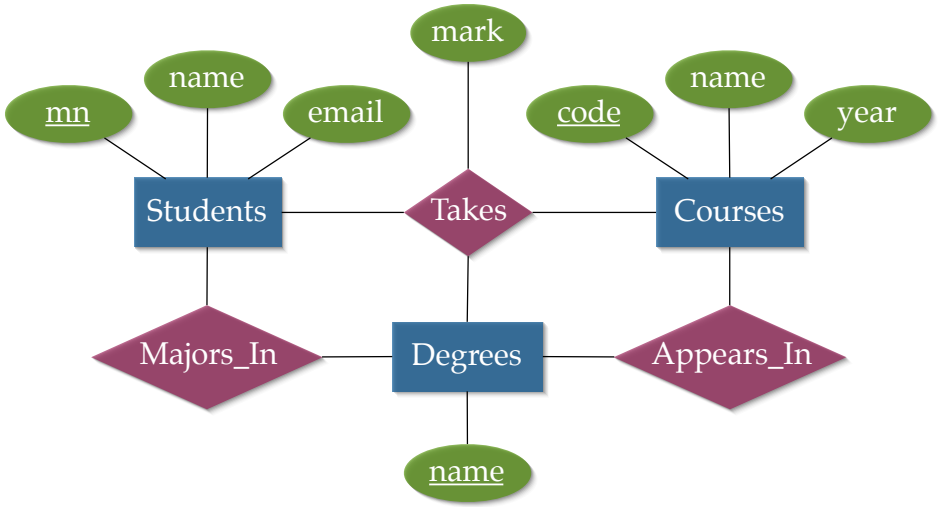
- Relations: Tables matching schemas
- Schema: A set of field names and their domains
- Table: A set of tuples of values matching these fields
- Primary key: Chosen uniquely-valued field or set of fields
- Foreign key: Must be drawn from key values in another table

Translating ER Diagrams to Relational Models

An ER diagram captures a conceptual model of the data to be managed in a database: what there is and how it is connected.

We can use this as a basis for a relational schema, as a step towards implementation in a working RDBMS.

This translation will be approximate: some constraints expressed in an ER diagram might not naturally fit into relational schemas.



CREATE TABLE Students (...)

CREATE TABLE Takes (...)

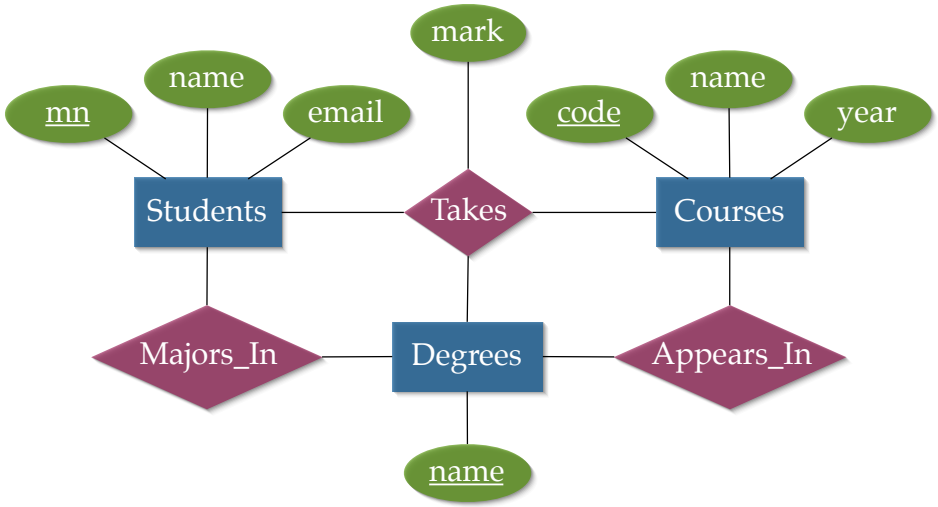
CREATE TABLE Degrees (...)

etc.

Translating ER Diagrams to Relational Models

There may be more than one possible translation: different alternatives lead to different implementations. These may be efficiency trade-offs, for which we might go back to the requirements to assess their relative impact.

It is possible to make these translations complete (work for any diagram) and automatic (in a push-button tool); but here we shall just consider a few examples illustrating some of the main ideas.



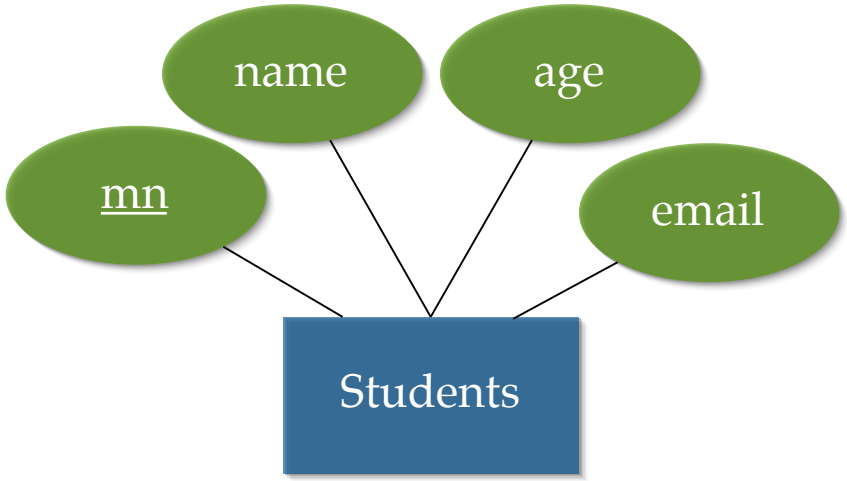
CREATE TABLE Students (...)

CREATE TABLE Takes (...)

CREATE TABLE Degrees (...)

etc.

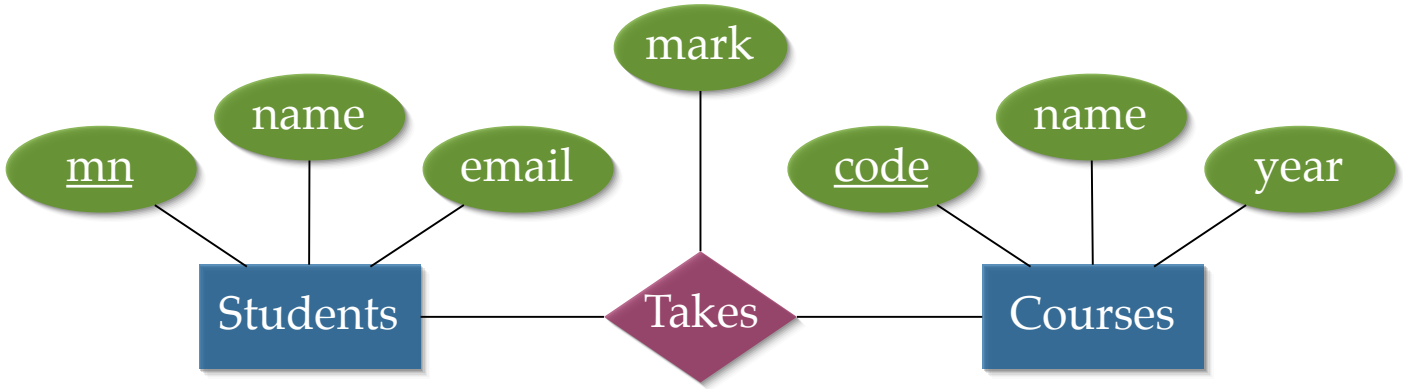
Mapping an Entity



- Create a table for the entity set.
- Make each attribute of the entity set a field of the table, with an appropriate type.
- Declare the field or fields which make up the primary key

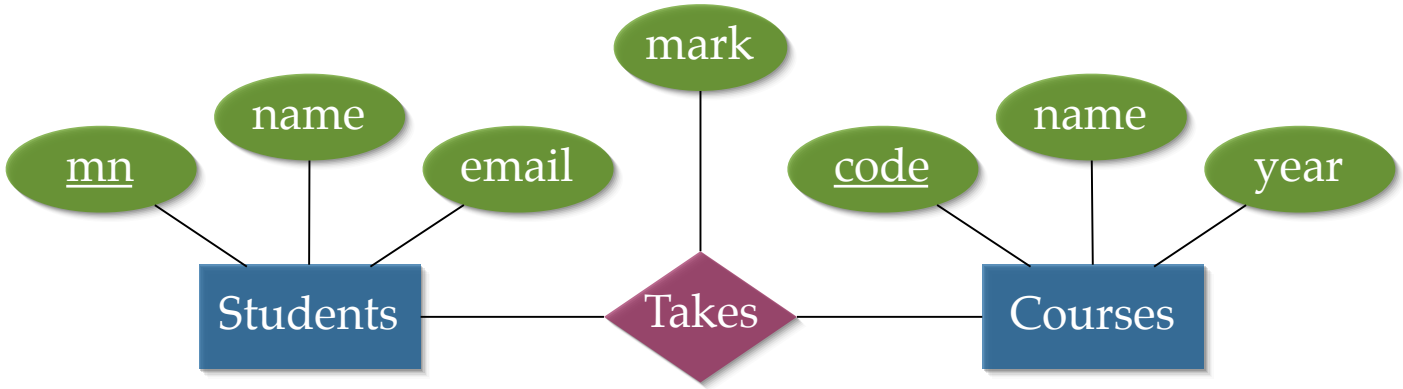
```
CREATE TABLE Students (
    mn      VARCHAR(8),
    name    VARCHAR(20),
    age     INTEGER,
    email   VARCHAR(25),
    PRIMARY KEY (mn) )
```

Mapping a Relationship



- Create a table for the relationship set
- Add all key attributes of all participating entity sets as fields.
- Add fields for each attribute of the relationship.
- Declare the primary key composed of all key attributes from the participating entity sets.
- Declare foreign key constraints for all these fields from the entity sets.

Mapping a Relationship




```
CREATE TABLE Takes (
    mn   VARCHAR(8),
    code VARCHAR(20),
    mark INTEGER,
    PRIMARY KEY (mn, code),
    FOREIGN KEY (mn) REFERENCES Students,
    FOREIGN KEY (code) REFERENCES Courses )
```

Database Textbooks

R. Ramakrishnan and J. Gehrke.
Database Management Systems.
McGraw-Hill, third edition, 2003.

H. Garcia-Molina, J. Ullman, and J. Widom.
Database Systems: The Complete Book.
Pearson, second edition, 2008.

J. Ullman and J. Widom.
A First Course in Database Systems.
Prentice Hall, third edition, 2009.

M. Kifer, A. Bernstein, and P. M. Lewis.
Database Systems: An Application-Oriented Approach, Introductory Version.
Pearson, second edition, 2004.

A. Silberschatz, H. Korth, and S. Sudarshan.
Database System Concepts.
McGraw-Hill, sixth edition, 2010.

Tutorial Exercises

The tutorial exercise sheet was posted to the course website on Tuesday, and announced on the blog.

There are six questions on the sheet, progressively designing and then drawing an Entity-Relationship model for part of a database system.

There is flexibility in how you design the model, and you might come up with several alternative answers.

Write or print out your work and bring it along to the tutorial: you may have to explain it to the other students, or exchange solutions with them.

If you find parts difficult, or have questions about the exercise, ask them on the discussion group or bring them to the tutorials.

I've posted an extension exercise on the discussion group — to find and try out different drawing tools.

Mapping Key Constraints

- Create a table for the relationship set.
- Add all key attributes of all participating entity sets as fields.
- Add fields for each attribute of the relationship.
- Declare a primary key composed of key attributes from the source entity set only.
- Declare foreign key constraints on the fields from all entity sets.

Mapping Key Constraints


```
CREATE TABLE DirectedBy (
    mn       VARCHAR(8),
    staff_id VARCHAR(8),
    PRIMARY KEY (mn),
    FOREIGN KEY (mn) REFERENCES Students,
    FOREIGN KEY (staff_id) REFERENCES DoS )
```

This captures the key constraint, but not the participation constraint.

Mapping Key Constraints, Alternative

Because the DirectedBy relation is many-to-one, we don't need a whole table for the relation itself. However, this does slightly pollute the source entity table.

- Create a table for the source and target entity sets as usual.
- Add every key field of the target as a field in the source.
- Declare these fields as foreign keys.

Mapping Key Constraints, Alternative


```
CREATE TABLE Students (
    mn     VARCHAR(8),
    name   VARCHAR(20),
    age    INTEGER,
    email  VARCHAR(25),
    dos_id VARCHAR(8),
    PRIMARY KEY (mn),
    FOREIGN KEY (dos_id) REFERENCES DoS(staff_id) )
```

This may still not include the total participation constraint on Students in DirectedBy, but we are now nearer to doing so...

Null Values

A field in SQL can have the special value **NULL**.

NULL can mean many things: that a field is unknown, or missing, or unavailable; or that this field may not apply in certain situations.

Some RDBMS forbid **NULL** from appearing in any field declared as a primary key.

Some of these may still allow **NULL** to appear in foreign key fields.

A schema can state that certain fields may not contain **NULL** using the **NOT NULL** declaration.

Forbidding **NULL** is in some cases a way to enforce total participation.

Mapping Key and Participation Constraints

- Create a table for the source and target entity sets as usual.
- Add every key field of the target as a field in the source.
- Declare these fields as **NOT NULL**
- Declare these fields as foreign keys.

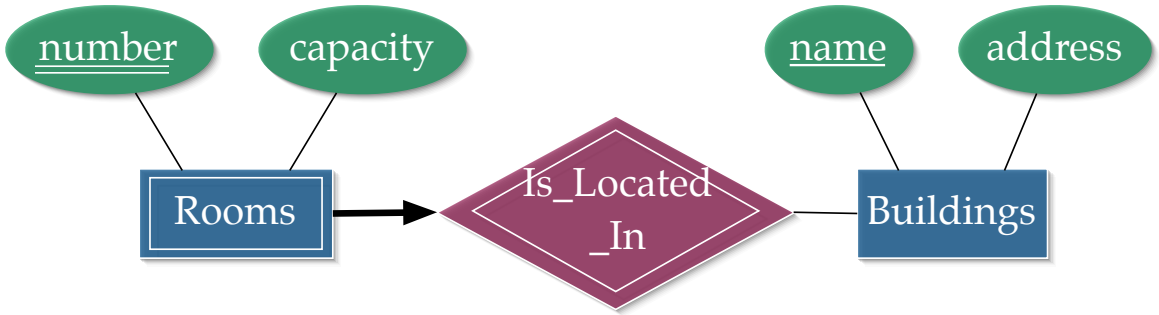
Mapping Key and Participation Constraints

- Create a table for the source and target entity sets as usual.
- Add every key field of the target as a field in the source.
- Declare these fields as **NOT NULL**
- Declare these fields as foreign keys.

Mapping Key and Participation Constraints

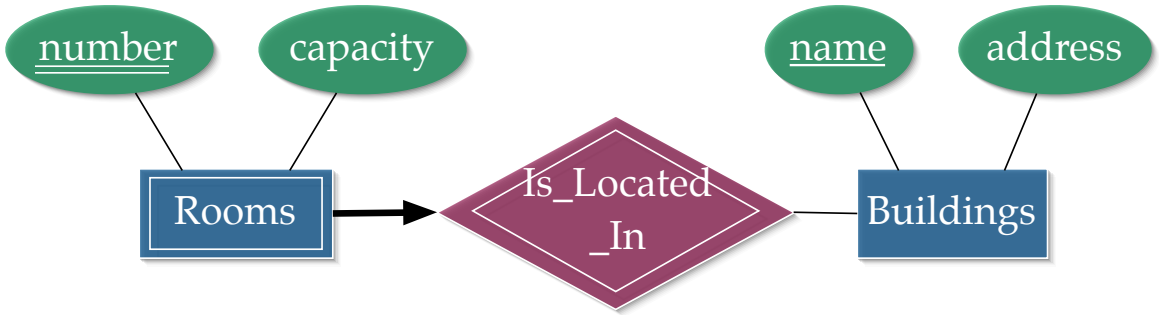

```
CREATE TABLE Students (
    mn      VARCHAR(8),
    name    VARCHAR(20),
    age     INTEGER,
    email   VARCHAR(25),
    dos_id VARCHAR(8) NOT NULL,
    PRIMARY KEY (mn),
    FOREIGN KEY (dos_id) REFERENCES DoS(staff_id) )
```

Mapping Weak Entity Sets and Identifying Relationships



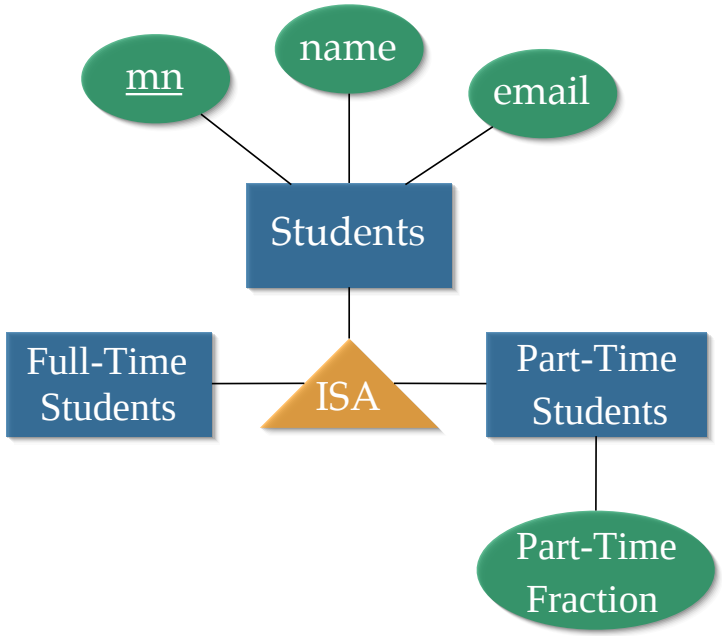
- Create a table for the weak entity set.
- Make each attribute of the weak entity set a field of the table.
- Add fields for the key attributes of the identifying owner.
- Declare the composite key using attributes from both entities.
- Declare a foreign key constraint on the identifying owner fields.
- Instruct the system to automatically delete any tuples in the table when their identifying owners are deleted.

Mapping Weak Entity Sets and Identifying Relationships



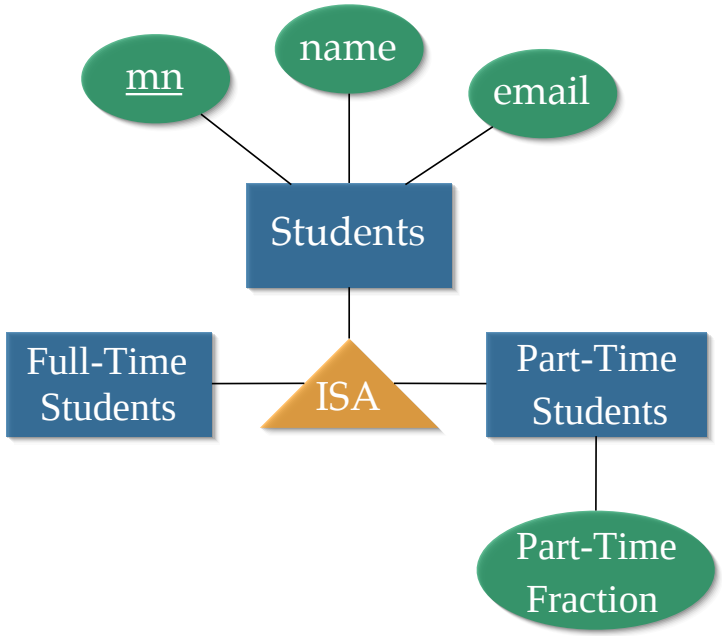

```
CREATE TABLE Rooms (
    number          VARCHAR(8),
    capacity        INTEGER,
    building_name   VARCHAR(20),
    PRIMARY KEY (number, building_name),
    FOREIGN KEY (building_name)
        REFERENCES Buildings(name)
        ON DELETE CASCADE )
```

Mapping Entity Hierarchies



- Declare a table for the superclass entity.
- For each subclass entity declare another table using the primary key of the superclass and the extra attributes of the subclass.
- Declare the primary key from the superclass as the primary key of the subclass, with a foreign key constraint.

Mapping Entity Hierarchies




```
CREATE TABLE PTStudents (
    mn          VARCHAR(8),
    pt_frac     REAL,
    PRIMARY KEY (mn),
    FOREIGN KEY (mn) REFERENCES Students )
```