Informatics 1: Data & Analysis

Lecture 4: From ER Diagrams to Relational Models

Ian Stark

School of Informatics
The University of Edinburgh

Thursday 24 January 2019
Semester 2 Week 2

THE UNIVERSITY *of* EDINBURGH

Outline

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

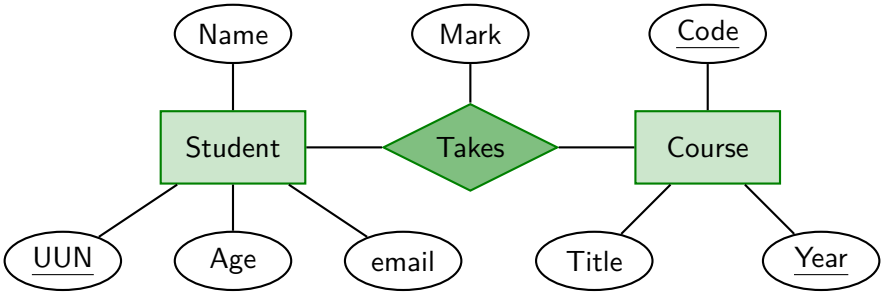
This is followed by some methods for manipulating data in the relational model and using it to extract information.

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

The Story so Far

Entity-Relationship diagrams

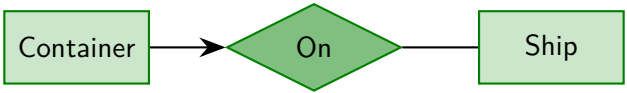
- Entities: Rectangles
- Relationships: Diamonds linked to the entities
- Attributes: Ovals linked to entity or relationship



The Story so Far

Entity-Relationship diagrams

- Entities: Rectangles
- Relationships: Diamonds linked to the entities
- Attributes: Ovals linked to entity or relationship
- Key constraints as arrows



The Story so Far

Entity-Relationship diagrams

- Entities: Rectangles
- Relationships: Diamonds linked to the entities
- Attributes: Ovals linked to entity or relationship
- Key constraints as arrows; total participation as thick/double lines

The Story so Far

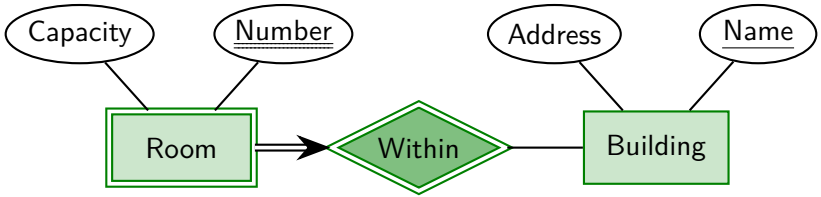
Entity-Relationship diagrams

- Entities: Rectangles
- Relationships: Diamonds linked to the entities
- Attributes: Ovals linked to entity or relationship
- Key constraints as arrows; total participation as thick/double lines

The Story so Far

Entity-Relationship diagrams

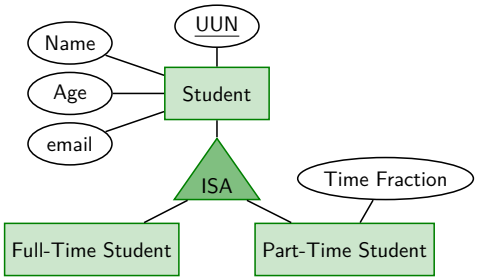
- Entities: Rectangles
- Relationships: Diamonds linked to the entities
- Attributes: Ovals linked to entity or relationship
- Key constraints as arrows; total participation as thick/double lines
- Weak entities with their identifying relationship



The Story so Far

Entity-Relationship diagrams

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



The Story so Far

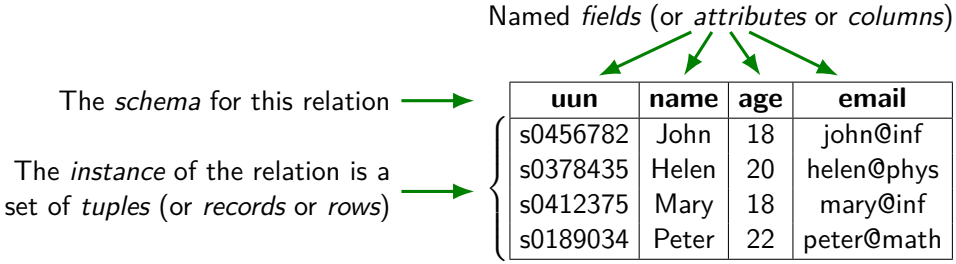
Entity-Relationship diagrams

- Entities: Rectangles
- Relationships: Diamonds linked to the entities
- Attributes: Ovals linked to entity or relationship
- Key constraints as arrows; total participation as thick/double 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

The Story so Far



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

The Story so Far

Student

uun	name	age	email
s0456782	John	18	john@inf
s0378435	Helen	20	helen@phys
s0412375	Mary	18	mary@inf
s0189034	Peter	22	peter@math

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

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

The Story so Far

Student

uun	name	age	email
s0456782	John	18	john@inf
s0378435	Helen	20	helen@phys
s0412375	Mary	18	mary@inf
s0189034	Peter	22	peter@math

Takes

uun	code	mark
s0456782	inf1	71
s0412375	math1	82
s0412375	geo1	64
s0189034	math1	56

Course

code	title	year
inf1	Informatics 1	1
math1	Mathematics 1	1
geo1	Geology 1	1
dbs	Database Systems	3
adbs	Advanced Databases	4

```
CREATE TABLE Takes (
    uun     VARCHAR(8),
    code    VARCHAR(20),
    mark    INTEGER,
    PRIMARY KEY (uun, code),
    FOREIGN KEY (uun) REFERENCES Student,
    FOREIGN KEY (code) REFERENCES Course )
```

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

Aim and Scale

The University of Edinburgh has 41 312 matriculated students studying 8 143 different courses across 20 different schools in 609 buildings.

With a relational database we can ask, and answer, questions like:

- What are the names and email addresses of all students taking a first-year course taught by the School of Informatics?
- Which lectures have been scheduled for rooms which will be at greater than 90% of their seating capacity?

Many databases are much, much larger than this with much more complex queries. However, the aim is the same: how to go beyond answering a single query with a single program to a general system in which many queries can be expressed and answered efficiently.

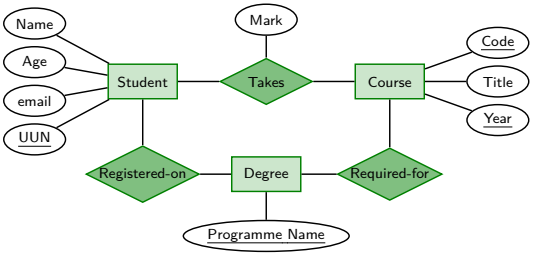
Outline

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 (Relational Database Management System)

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



CREATE TABLE Student (...)

CREATE TABLE Takes (...)

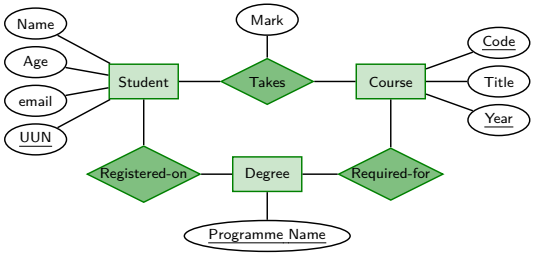
CREATE TABLE Course (...)

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.



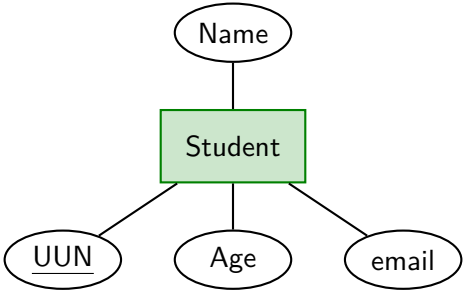
CREATE TABLE Student (...)

CREATE TABLE Takes (...)

CREATE TABLE Course (...)

etc.

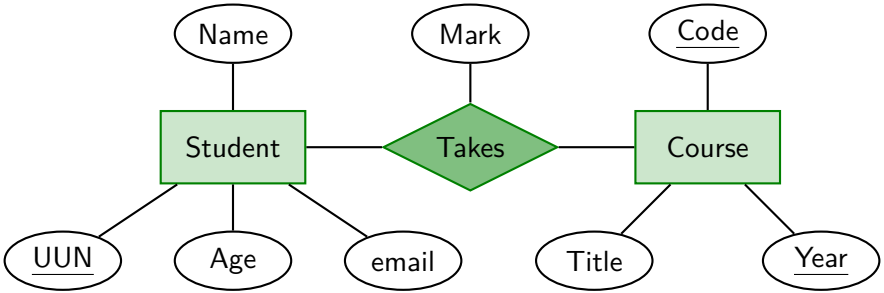
Mapping an Entity



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

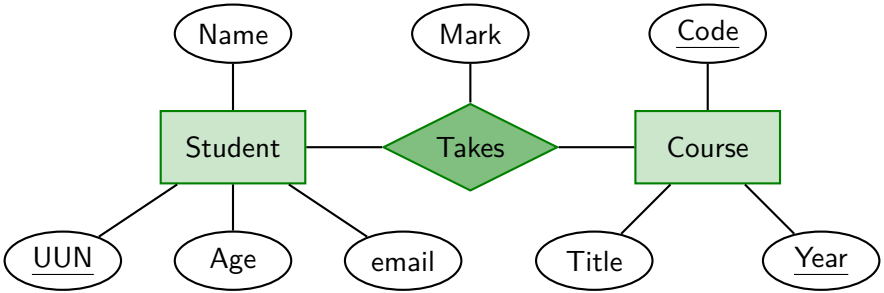
```
CREATE TABLE Student (
    uun     VARCHAR(8),
    name    VARCHAR(20),
    age     INTEGER,
    email   VARCHAR(25),
    PRIMARY KEY (uun) )
```

Mapping a Relationship



- Create tables for each entity, and an additional table for the relationship.
- Add all key attributes of all participating entities as fields in the relationship table.
- Add further fields for any attributes of the relationship itself.
- Declare primary key using all key attributes from participating entities.
- Declare foreign key constraints for all fields in the relationship table that refer to attributes of the entities.

Mapping a Relationship



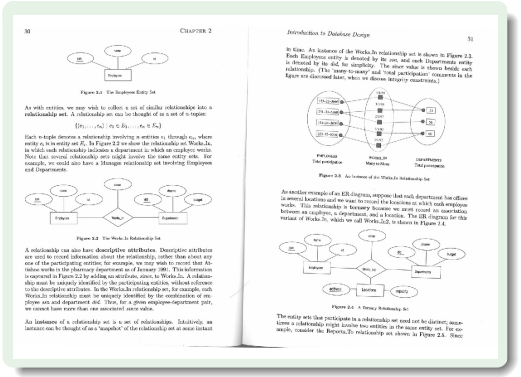

```
CREATE TABLE Takes (
    uun  VARCHAR(8),   year  INTEGER,
    code VARCHAR(20),  mark INTEGER,
    PRIMARY KEY (uun, code, year),
    FOREIGN KEY (uun) REFERENCES Student,
    FOREIGN KEY (code,year) REFERENCES Course )
```

Homework from Tuesday

Read This

Before the next lecture, on Friday, read the remaining sections, §§2.5 onwards, of Ramakrishnan and Gehrke, completing Chapter 2.

These consider trade-offs and choices in the design of Entity-Relationship models, as well as more on the wider context of modelling.



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

Database Textbooks (Page 1 of 2)

R. Ramakrishnan and J. Gehrke. 3 copies in HUB

Database Management Systems. 6 other copies

McGraw-Hill, third edition, 2003.

H. Garcia-Molina, J. Ullman, and J. Widom. 1 copy in HUB

Database Systems: The Complete Book.

Pearson, second edition, 2009.

J. Ullman and J. Widom. 1 copy in HUB

A First Course in Database Systems. 1 other copy

Prentice Hall, third edition, 2008.

M. Kifer, A. Bernstein, and P. M. Lewis. 1 copy in HUB

Database Systems: An Application-Oriented Approach, Introductory Version.

Pearson, second edition, 2005.

Database Textbooks (Page 2 of 2)

A. Silberschatz, H. Korth, and S. Sudarshan. 1 copy in HUB

Database System Concepts.

McGraw-Hill, sixth edition, 2011.

T. Connolly and C. Begg. 1 copy in HUB

Database Systems: A Practical Approach to Design, Implementation and Management.

Addison-Wesley, fourth edition, 2005.

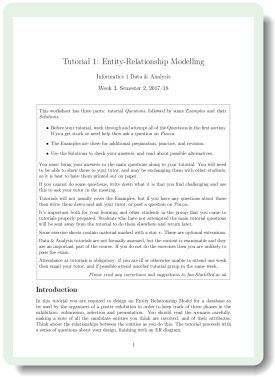
R. Elmasri and S. Navathe. 3 copies

Fundamentals of Database Systems.

Addison-Wesley, third edition, 2000.

Do This

Download from Learn the worksheet with the first set of tutorial exercises. Read the instructions on the front page, follow them.



Tutorial 1: Entity-Relationship Modelling

Informatics 1 Data & Analysis

Week 3, Semester 2, 2017/18

This worksheet has three parts: tutorial *Questions*, followed by some *Examples* and their *Solutions*.

- Before your tutorial, work through and attempt all of the Questions in the first section. If you get stuck or need help then ask a question on *Piazza*.
- The Examples are there for additional preparation, practice, and revision.
- Use the Solutions to check your answers, and read about possible alternatives.

You must bring your answers to the main questions along to your tutorial. You will need to be able to show them to your tutor, and may be exchanging them with other students, so it is best to have them printed out on paper.

If you cannot do some questions, write down what it is that you find challenging and use this to ask your tutor in the meeting.

Tutorials will not usually cover the Examples, but if you have any questions about them write them down and ask your tutor, or post a question on *Piazza*.

It's important both for your learning and other students in the group that you come to tutorials properly prepared. Students who have not attempted the main tutorial questions will be sent away from the tutorial to do them elsewhere and return later.

Some exercises sheets contain material marked with a star ⋆. These are optional extensions.

Data & Analysis tutorials are not formally assessed, but the content is examinable and they are an important part of the course. If you do not do the exercises then you are unlikely to pass the exam.

Attendance at tutorials is obligatory: if you are ill or otherwise unable to attend one week then email your tutor, and if possible attend another tutorial group in the same week.

Please send any corrections and suggestions to Jan.Hidders@ed.ac.uk

Introduction

In this tutorial you are required to design an Entity Relationship Model for a database to be used by the organisers of a poster exhibition in order to keep track of three phases in the exhibition: submission, selection and presentation. You should read the scenario carefully, making a note of all the candidate entities you think are involved, and of their attributes. Think about the relationships between the entities as you do this. The tutorial proceeds with a series of questions about your design, finishing with an ER-diagram.

1

Each tutorial worksheet also has several more example problems and notes on solving them. You don't have to do them for tutorials, but you can use them as guidance. After the tutorial, try to solve some of them without consulting the solutions.

The sheet has six questions where you progressively design and then draw an Entity-Relationship model for part of a database system.

There is flexibility in how you design the model, and there is more than one possible solution to the problem.

How to Tutorial !

Before the tutorial, work through the exercise sheet. Ask and answer questions on Piazza, talk to other students, drop into InfBASE, write up your work.

Bring your solutions and your working. You will need to be able to show these to your tutor and exchange them with other students.

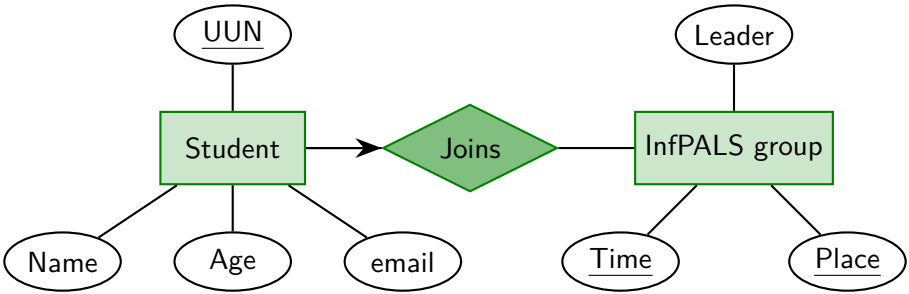
Come to tutorials properly prepared. Students who have not attempted the exercises will be sent away to spend the time working on them.

Data & Analysis tutorials and exercises are not given marks or grades, but their content is examinable and they are an important part of the course. If you do not do the exercises then you are unlikely to pass the exam.

Course participation requires attendance at tutorials: if you are ill or otherwise unable to attend one week then email your tutor, and if possible attend another tutorial group in the same week.

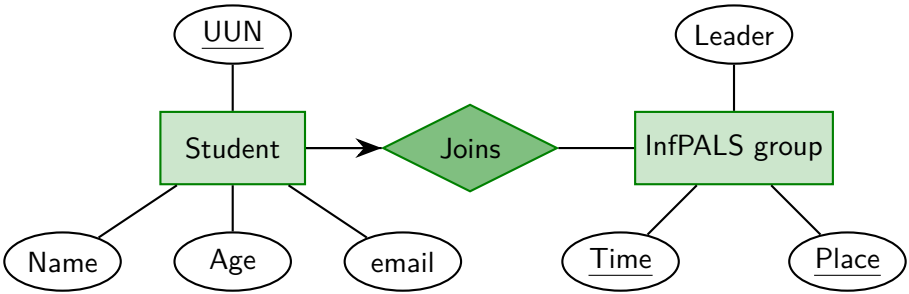
Outline

Mapping Key Constraints, Method 1



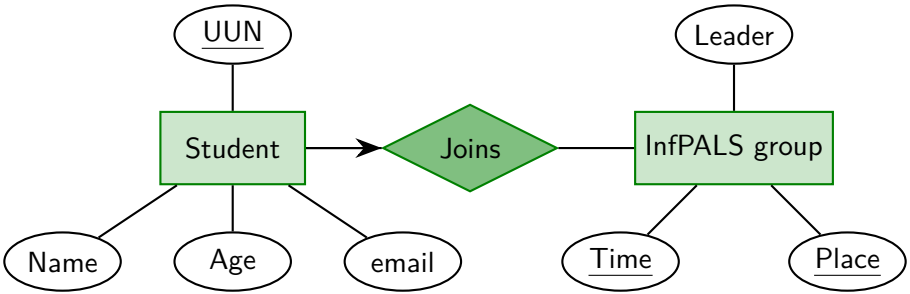
- Create tables for each entity, and an additional table for the relationship.
- Add all key attributes of all participating entities as fields in the relationship table.
- Add further fields for any attributes of the relationship itself.
- Declare primary key using only key attributes of the source entity.
- Declare foreign key constraints for all fields in the relationship table that refer to attributes of the entities.

Mapping Key Constraints, Method 1




```
CREATE TABLE Joins (
    uun   VARCHAR(8),
    time  VARCHAR(20),   place  VARCHAR(32),
    PRIMARY KEY (uun),
    FOREIGN KEY (uun) REFERENCES Student,
    FOREIGN KEY (time,place) REFERENCES InfPALSGroup )
```

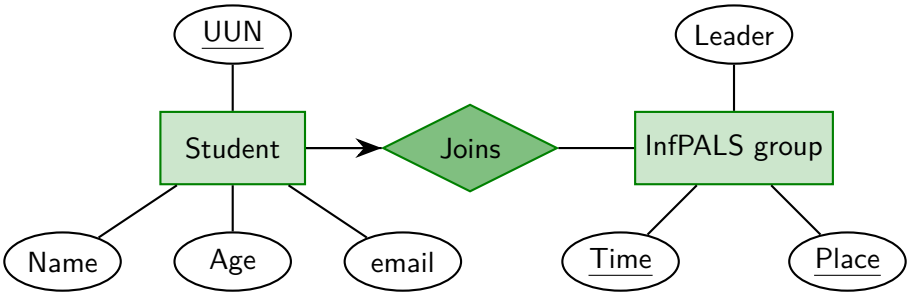
Mapping Key Constraints, Method 2



In fact, because the Joins relationship is many-to-one, we don't need a whole table for the relationship itself: information about InfPALS group membership can go in the Student table.

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

Mapping Key Constraints, Method 2




```
CREATE TABLE Student (
    uun     VARCHAR(8),    age     INTEGER,
    name    VARCHAR(20),   email   VARCHAR(25),
    time    VARCHAR(10),   place   VARCHAR(32),
    PRIMARY KEY (uun),
    FOREIGN KEY (time,place) REFERENCES InfPALSGroup )
```

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.

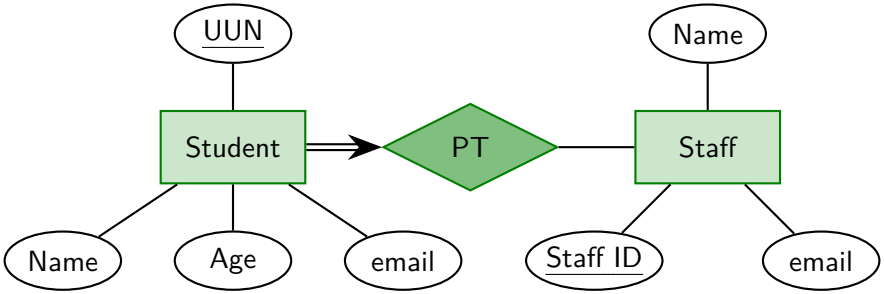
Some RDBMS forbid **NULL** from appearing in any field declared as a primary key.

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.

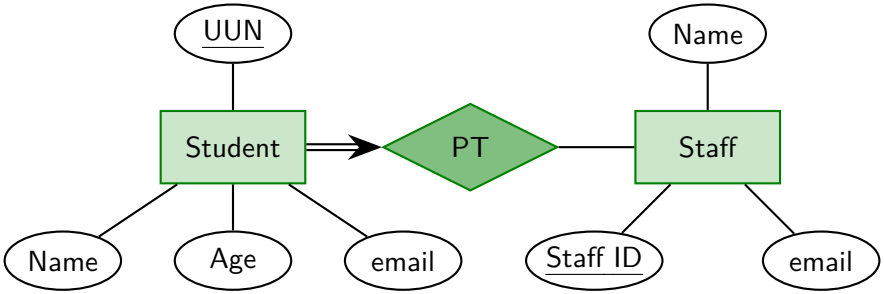
Forbidding **NULL** is in some cases a way to enforce total participation.

Mapping Key and Participation Constraints



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

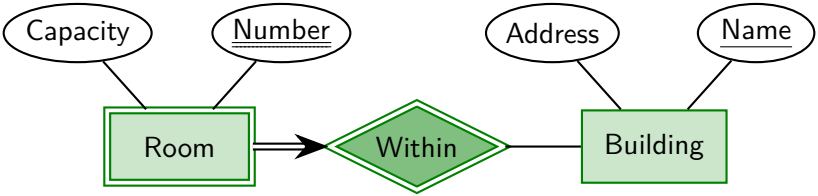
Mapping Key and Participation Constraints




```
CREATE TABLE Student (
    uun     VARCHAR(8),    age    INTEGER,
    name    VARCHAR(20),   email  VARCHAR(25),
    pt      VARCHAR(8) NOT NULL,
    PRIMARY KEY (uun),
    FOREIGN KEY (pt) REFERENCES Staff(staff_id) )
```

Outline

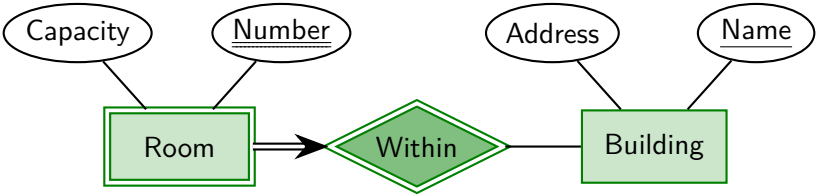
Mapping Weak Entities and Identifying Relationships



A weak entity always has a participation and key constraint with its identifying relationship, which makes things similar to the previous case.

- Create a tables for the weak entity and its identifying owner.
- Add every key attribute of the identifying owner as a field in the weak entity table.
- Declare a composite key using key attributes from both entities.
- Declare a foreign key constraint on the identifying owner fields.
- Declare that records in the table must be deleted when their identifying owners are.

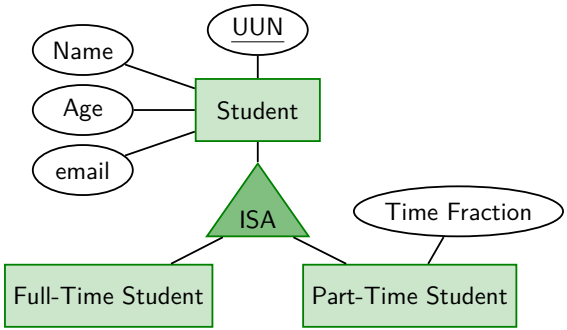
Mapping Weak Entities and Identifying Relationships




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

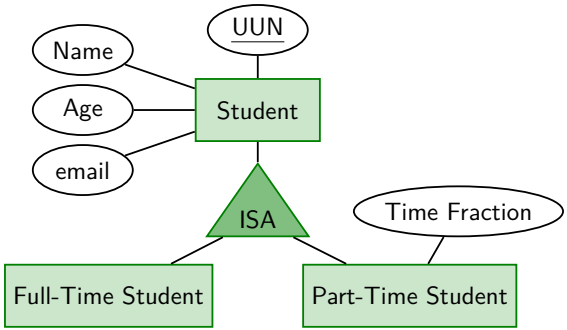
(Don't use **ON DELETE SET NULL** here)

Mapping Entity Hierarchies and Inheritance



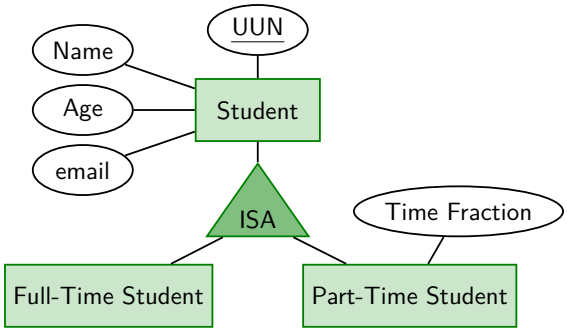
- Declare a table for the superclass entity with all attributes.
- For each subclass entity declare another table using the primary key of the superclass and any extra attributes of the subclass.
- Declare the primary key from the superclass as the primary key of each subclass, with a foreign key constraint.

Mapping Entity Hierarchies and Inheritance



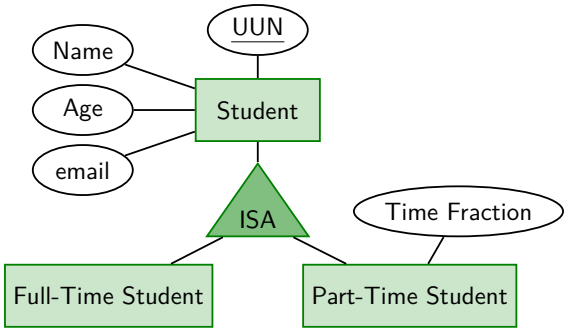

```
CREATE TABLE Student (
    uun   VARCHAR(8),   age   INTEGER,
    name  VARCHAR(20),  email VARCHAR(25),
    PRIMARY KEY (uun) )
```

Mapping Entity Hierarchies and Inheritance




```
CREATE TABLE FullTimeStudent (
    uun   VARCHAR(8),
    PRIMARY KEY (uun),
    FOREIGN KEY (uun) REFERENCES Student )
```

Mapping Entity Hierarchies and Inheritance




```
CREATE TABLE PartTimeStudent (
    uun  VARCHAR(8),  fraction FLOAT,
    PRIMARY KEY (uun),
    FOREIGN KEY (uun) REFERENCES Student )
```

Summary

Conceptual Model

ER-diagrams: entities with attributes and the relationships between them.

Logical Model

Relational models: tables with fields that can refer to other tables.

Translation

We can use an ER-diagram to guide the design of an appropriate relational model. This may require choosing between alternatives and making compromises: possibly involving simplicity of the design or its efficient implementation.

EDINBURGH GLOBAL GAME JAM® 2019 25-27 JAN

Artists, Musicians, coders, writers...all welcome!

48 HOUR GAME JAM
FREE & OPEN TO ALL

REGISTER ONLINE AT bit.ly/EDINGGJ

EDINBURGH NAPIER UNIVERSITY