ERM to Relational Model

Kathleen Durant

CS 3200

Lesson 3

Outline for today

- Review of fundamental ERM concepts
- Walk through the process of identifying the components of a entity relational model (ER)
- Introduce the relational model
- Mapping from the ERM to the Relational model
- Introduction to the SQL data definition commands

Entity Relational Model

- It is expressed in terms of **entities** in the environment
- The **relationships** (or associations) among those entities
- The **attributes** (properties) of both the entities and their relationships

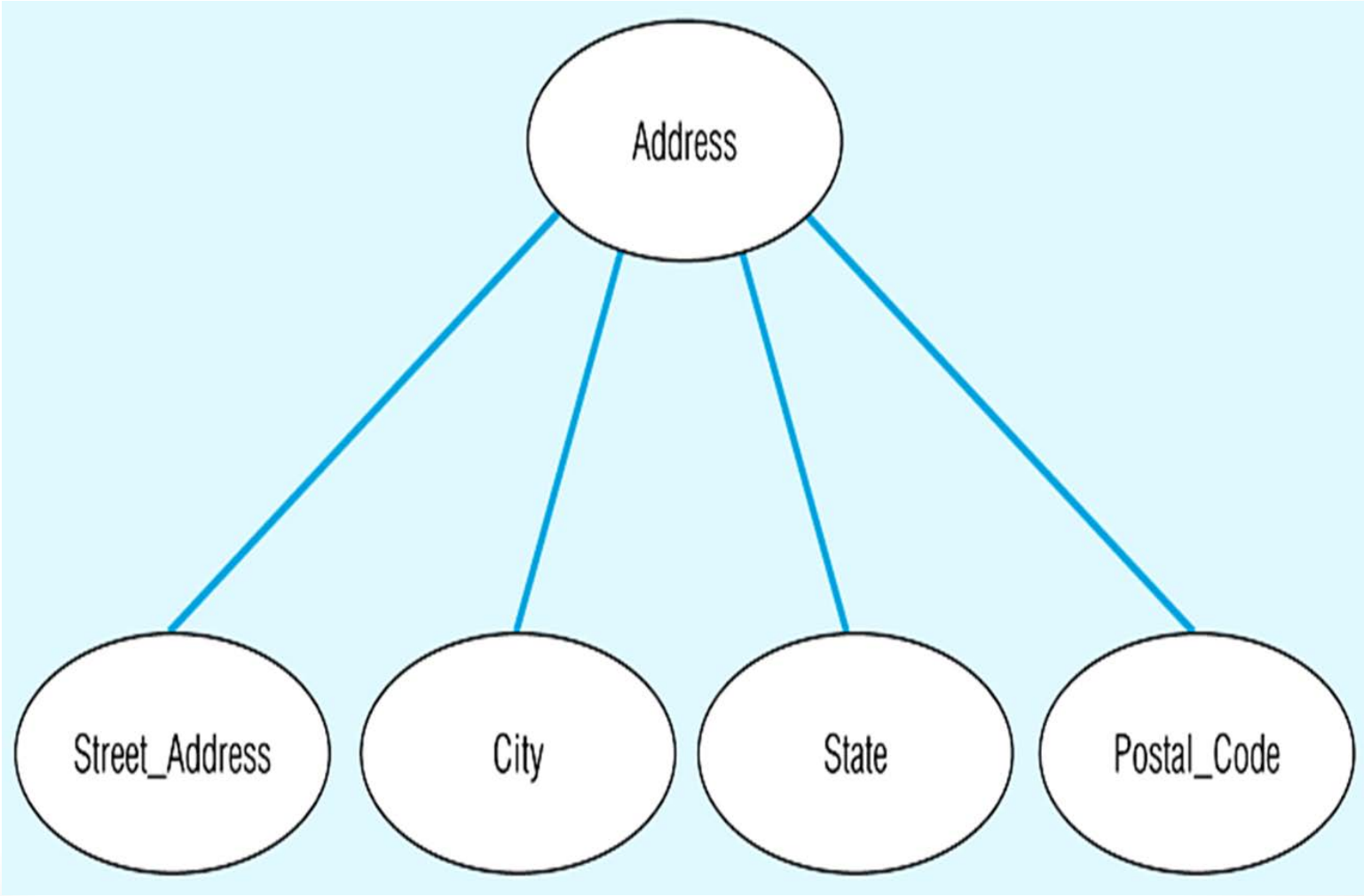
ER Model constructs: Entities

- Entity - person, place, object, event, concept
- Entity Set - is a collection of entities that share common properties or characteristics.
  - Each entity set is given a unique name
  - Since this name represents a set of items, it is always singular
  - The description of an entity set is often referred to as an Entity Type

ER Model Constructs : Attributes

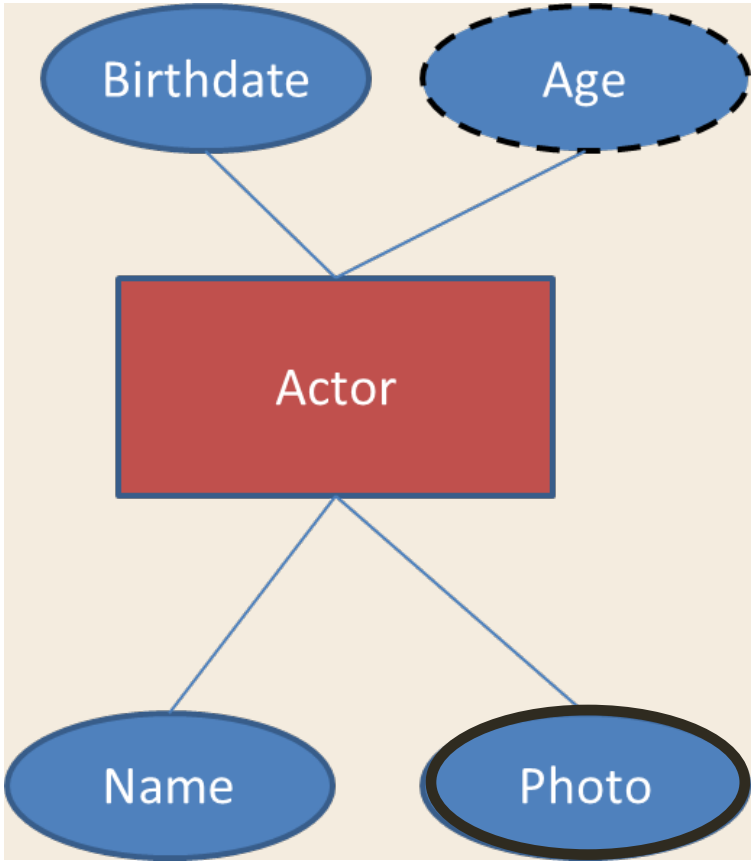
- An attribute is a property or characteristic of an entity type
  - Describes (descriptor) or represents (key) an entity
  - Simple attribute vs. Derived attribute
  - Atomic attribute vs. Composite attribute
  - Single-Valued versus Multi-valued Attribute
- Attributes may also be associated with relationships
- An attribute is associated with exactly one entity or relationship

Example: Composite Attribute



Example: Types of Attributes

Simple Attribute



Derived Attribute

Derived from a subset of the entity's other attributes

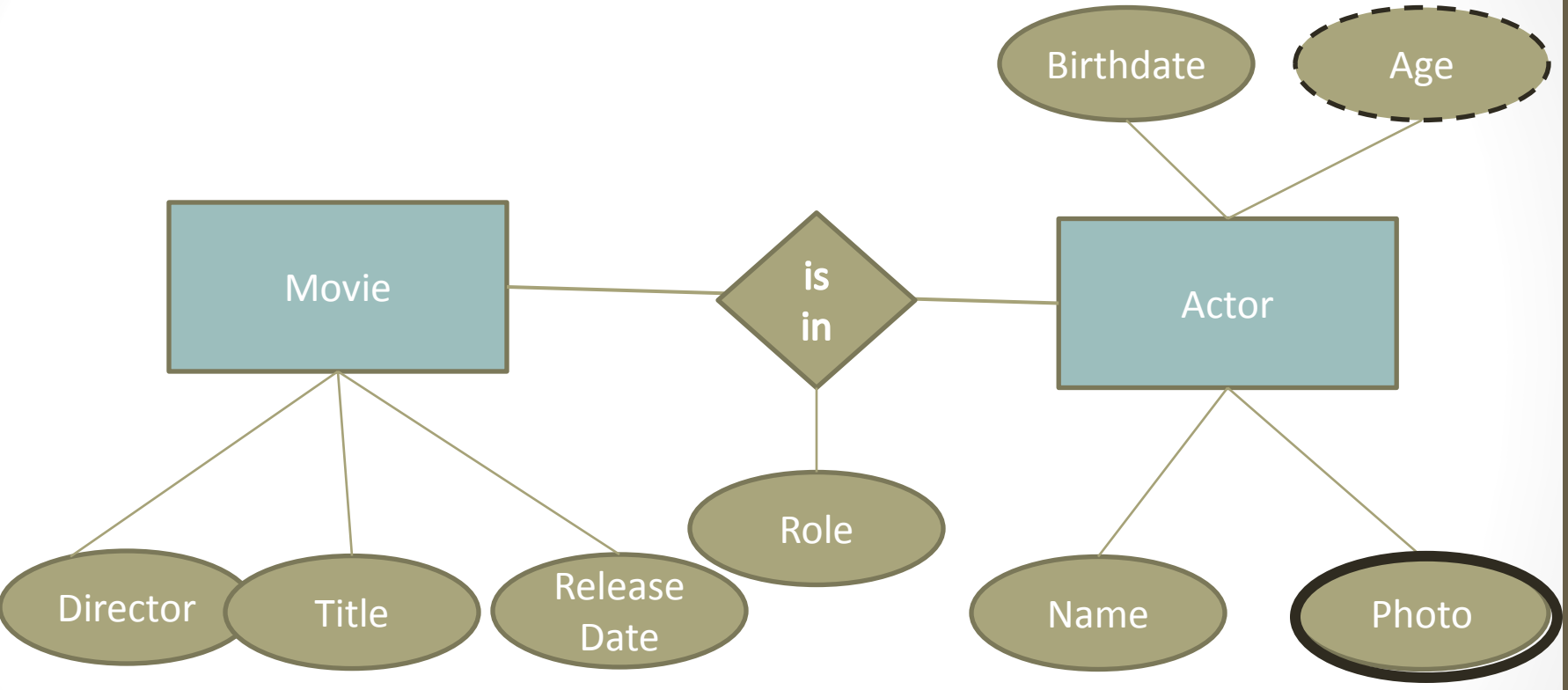
Single-valued Attribute

Multi-valued Attribute

ER Model constructs: Relationships

- A relationship is an association among the instances of one or more entity sets that is of interest
- Relationship Set is a meaningful association between (or among) entity sets
  - Implication: Relationship allows us to answer questions that can not be answered given only the entity sets
  - Set of n-tuples $\{(e_1, \ldots, en) | e_1 \in E_1, \ldots, e_n \in E_n\}$

Example: Relationship set



Constraints : Cardinality

- A relationship's *cardinality* = the maximum number of entities of one type that can be associated with an entity of another type.
  - 1 to 1, 1 to many , or many to many

Examples:

Relationship between *car* and *steering wheel*

Relationship between *building* and *room*

Relationship between *patient and doctor*

ER Process

Where to start

- To make an ER model from a verbal description you need to identify
  - Entities
  - Attributes
  - Relationships
- Cardinality ratios

General guidelines

- Since entities are things or objects they are often nouns in the description
- Attributes are facts or properties, and so are often nouns also
- Verbs often describe relationships between entities

Example to model

A university consists of a number of departments. Each department offers several majors. A number of courses make up each major. Students declare a particular major and take courses towards the completion of that major. Each course is taught by a lecturer from the appropriate department, and each lecturer tutors a group of students

Example: Entities

- A **university** consists of a number of **departments**. Each department offers several **majors**. A number of **courses** make up each **major. Students** declare a particular major and take courses towards the completion of that major. Each course is taught by a **lecturer** from the appropriate department, and each lecturer tutors a group of students

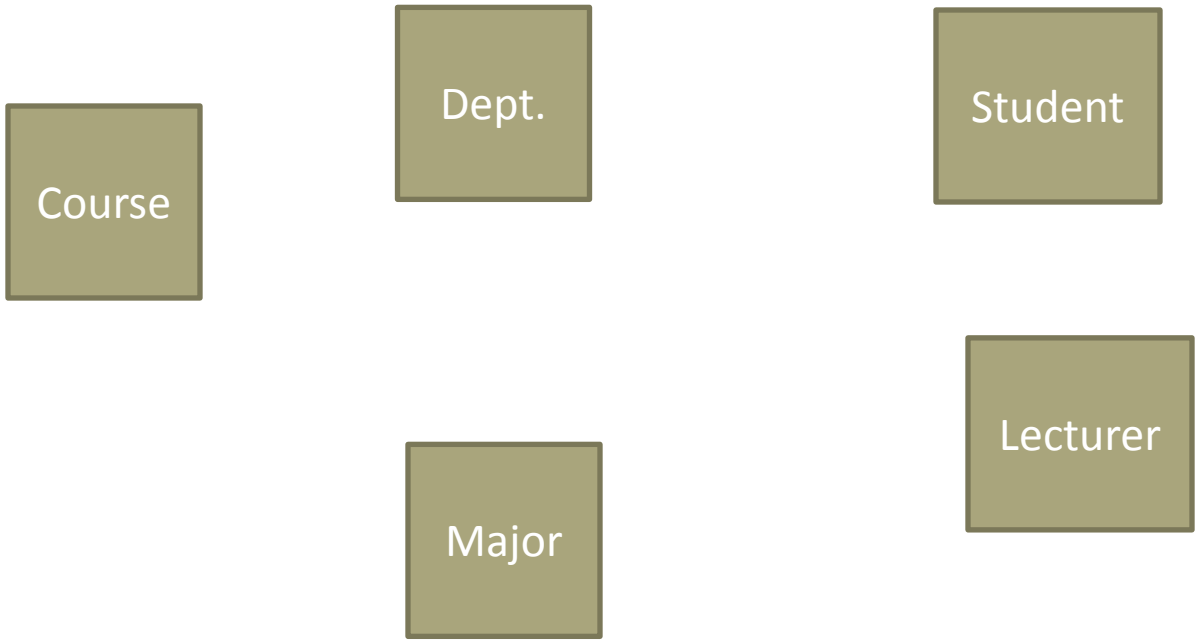
Example: Relationships

- A **university** consists of a number of departments. Each department **offers** several **majors**. A number of **courses** **make up** each major. **Students** **declare** a particular major and **take** courses towards the completion of that major. Each course is **taught** by a **lecturer** from the appropriate department, and each lecturer **tutors** a group of students

Entities

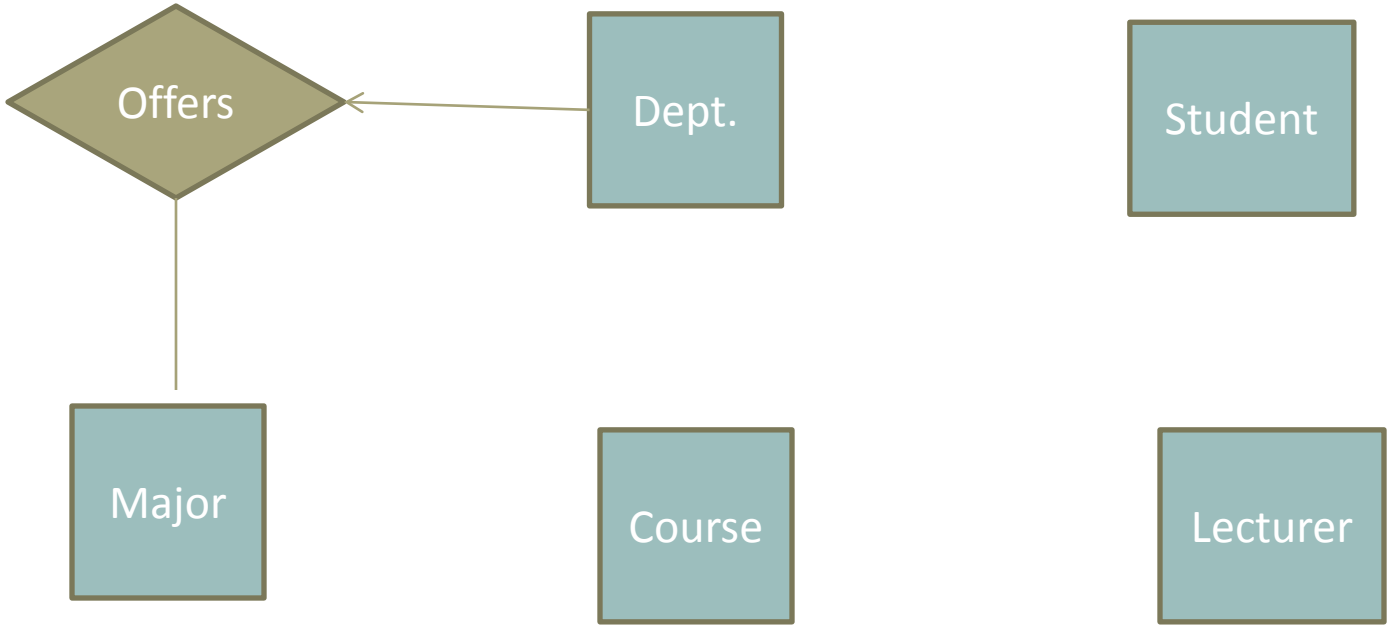
How do we add:

Department offers several majors



Entities – add a relationship

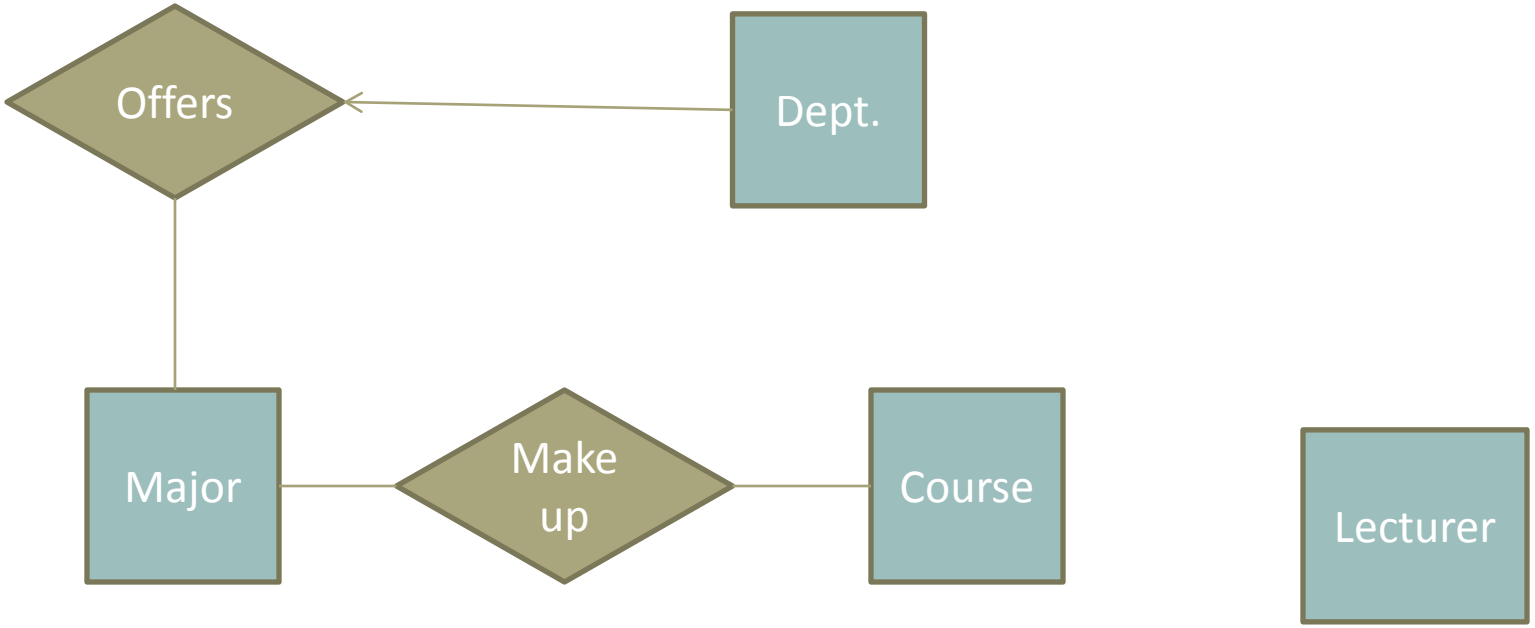
Department offers a major



A number of courses make up each major.

Entities – add a relationship

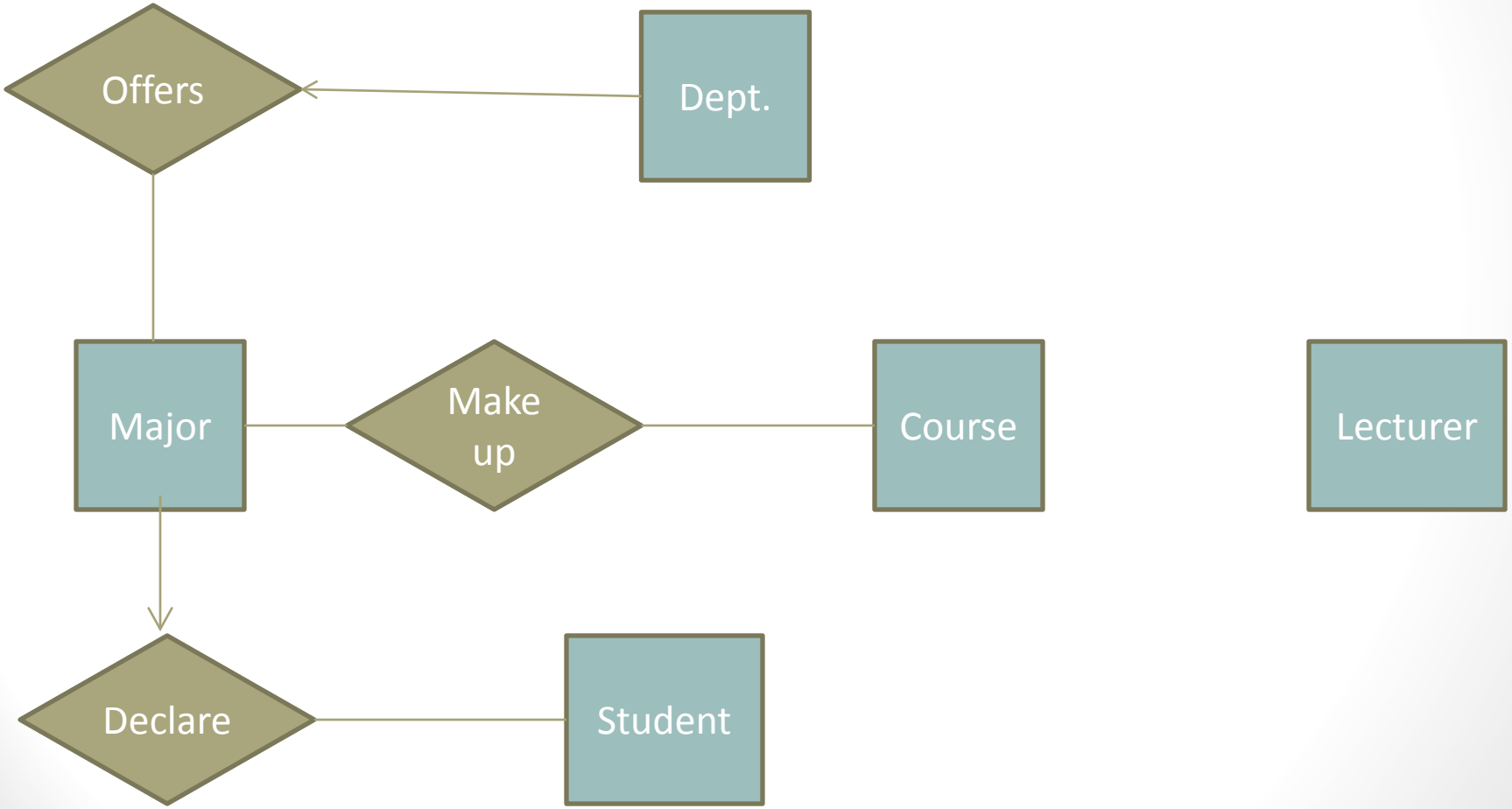
A number of courses make up each major.



Students declare a major.

Entities – add a relationship

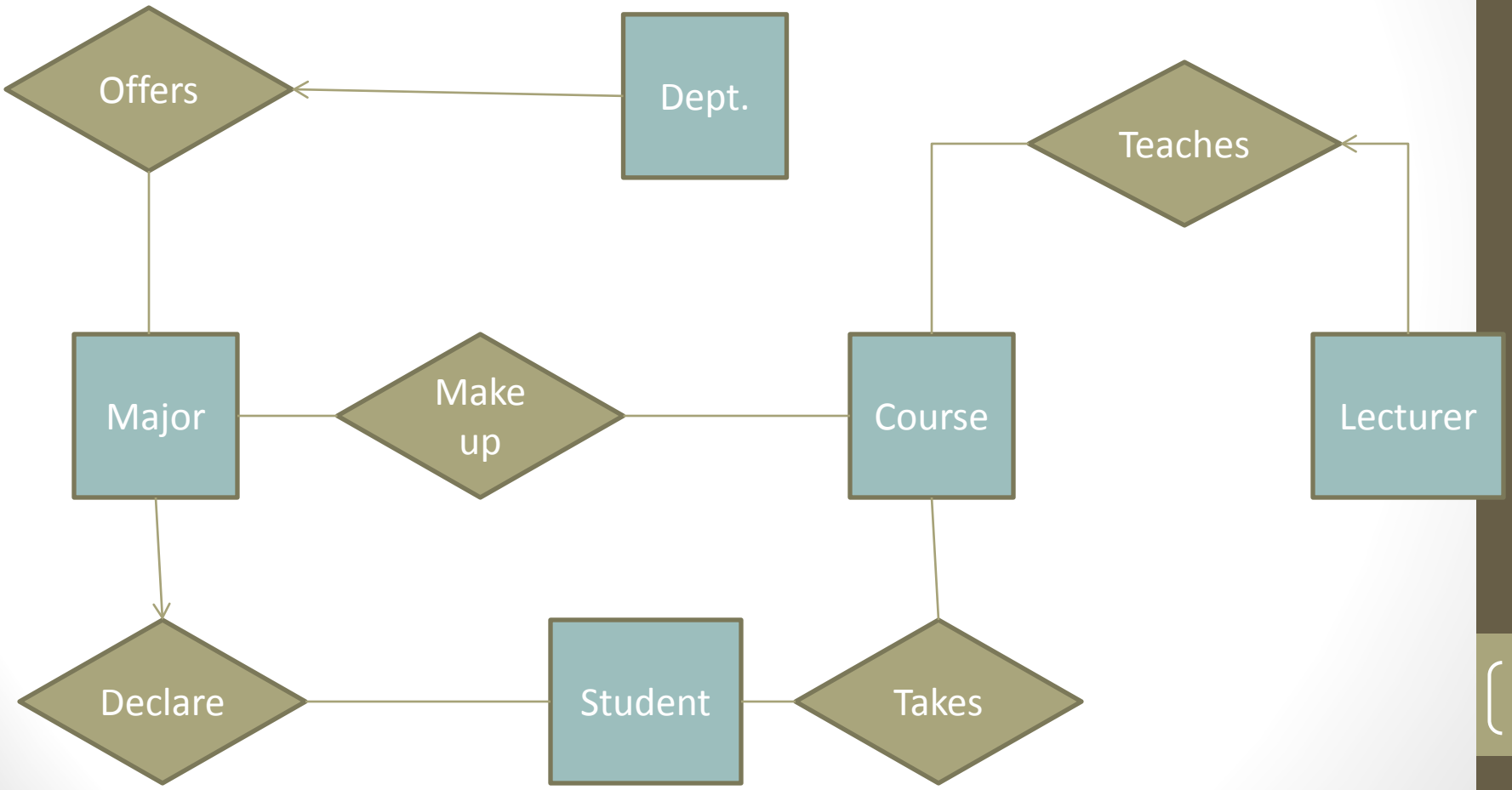
Students declare a major.



A course is taught by a lecturer.

Entities – add a relationship

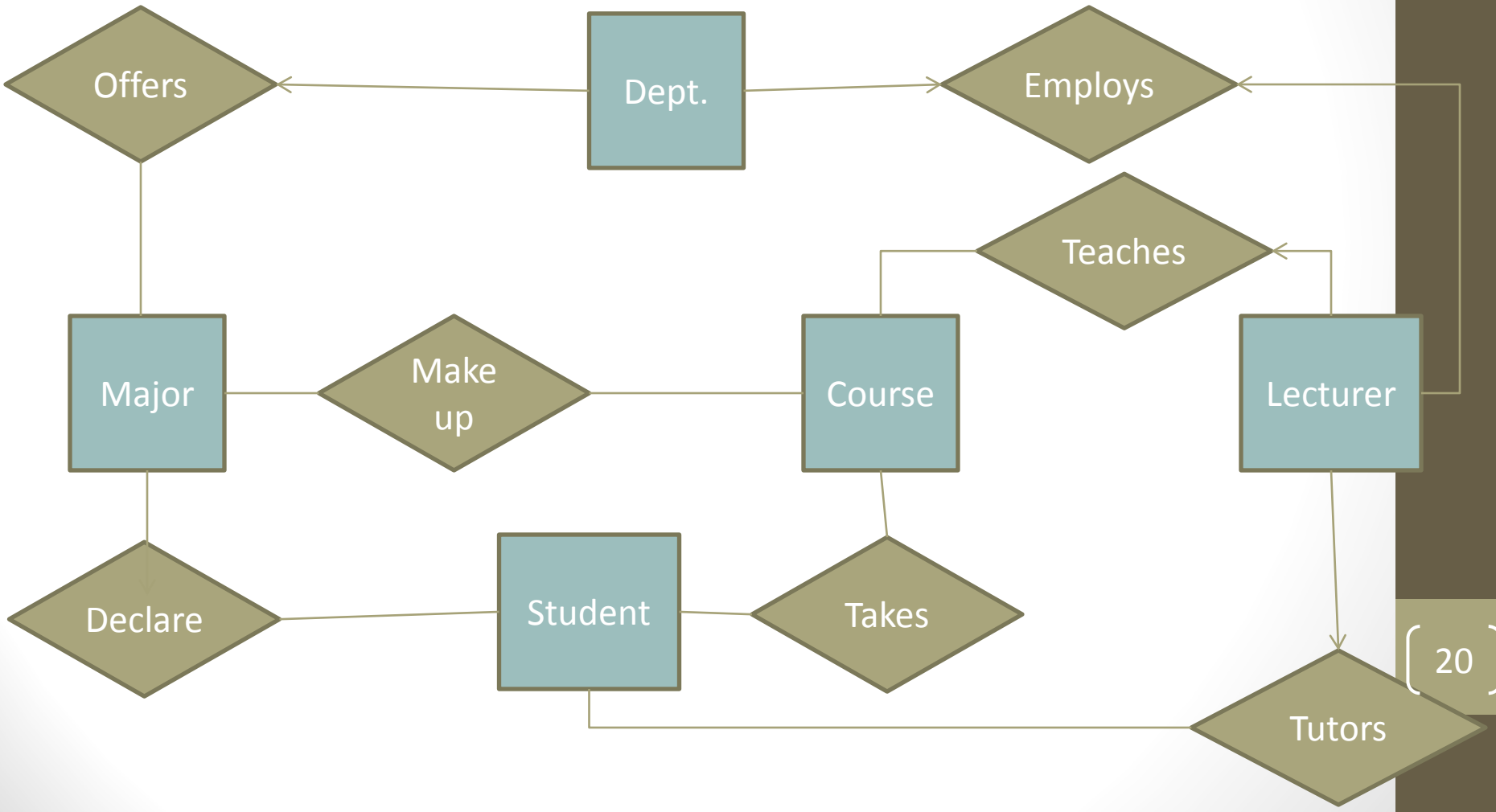
Each course is taught by a lecturer.



A lecturer tutors students

Entities – add a relationship

A lecturer tutors students



Introduction to Relational Model

- How does a relational database conceptually represent data?
  - How can we access specific values in a database?
  - How do we map an ER diagram to an actual database

Top level definitions

- Relational database: a set of relations
- Relation: made up of two parts
  - **Instance**: a table, with rows and columns.
    - #Rows = cardinality of the relation
    - #Fields = degree / arity of the relation
  - **Schema**: specifies name of relation, plus name and type of each column.
    - E.g., Students(sid: string, name: string, login: string, dob: date, gpa: real).
- One can think of a relation as a set of rows or tuples
  - All rows are distinct. (Not necessarily true for DBMS tables.)
  - Order of tuples is irrelevant (tuples may be stored in an arbitrary order)

Example of Relation

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98
55517	Ali	ali@math	Sep 22, 1989	3.11
55518	Smith	smith@math	Nov 30, 1991	3.32

- Cardinality = 4, degree = 5, all rows distinct
- Do all columns in a relation instance have to be distinct?

Relational Query Languages

- A major strength of the relational model: supports simple, powerful querying of data.
- Queries can be written intuitively, and the DBMS is responsible for efficient evaluation.
  - Specify WHAT you want, not HOW to get it efficiently
  - Declarative query language plus automatic optimizer
- How can it optimize different queries: precise semantics for relational queries.
  - Simplicity and elegance of relational model and operators also crucial
- Allows the optimizer to extensively re-order operations and still ensure that the answer does not change.

A SQL History

- Developed by IBM (System R) in the 1970s
- Need for a standard since it is used by many vendors
- Standards:
- SQL-86
- SQL-89 (minor revision)
- SQL-92 (major revision)
- SQL-99 (major extensions, current standard)
- However, not all vendors implement the complete standard and often there are vendor-specific extensions

Retrieving data from a table

Select command

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98
55517	Ali	ali@math	Sep 22, 1989	3.11
55518	Smith	smith@math	Nov 30, 1991	3.32

Select * from Students S where S.name = 'Smith'

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55518	Smith	smith@math	Nov 30, 1991	3.32

Selecting fields from multiple tables

- Select S.name, S.ssid, E.cid from Students S join Enrolled E on S.ssid = E.ssid where E.grade ='A'

SID	Name	Login	DoB	GPA
55515	Smith	smith@ccs	Jan 10,1990	3.82
55516	Jones	jones@hist	Feb 11, 1992	2.98
55517	Ali	ali@math	Sep 22, 1989	3.11
55518	Smith	smith@math	Nov 30, 1991	3.32

Sid	CId	Grade
55515	History 101	C
55516	Biology 220	A
55517	Anthro 320	B
55518	Music 101	A

name	ssid	cid
Jones	55516	Biology 220
Smith	55518	Music 101

Create a table

- Specify name of table, names of fields (columns) as well as date type for each field
  - Type (domain) of each field is enforced by the DBMS whenever tuples are added or modified.

- **Create table Students (ssid int, sname char(20), slogin char(40), dob date, gpa real)**

- **Create table Enrolled (ssid int, cid int, grade char(2))**

- **Create table <tablename> (fieldname type , …)**

Destroying and Altering tables

- The DROP TABLE statement allows you to remove tables from your schema
- *Drop* table <tablename> to remove a table from a database
  - Example:
  - *Drop* table Students

- ALTER TABLE statement will change the schema of a table
- *ALTER* TABLE <name> add column <column name> <column type>
  - Increase the –arity of the table
  - Example:

  Alter table Students Add column GradYear int

Adding and Deleting Tuples

- *INSERT into <TableName> (f_1,…,f_n) VALUES (v_1,…,v_n)*
  - *INSERT* INTO Students (sid, name, login, dob, gpa) VALUES (53688, ‘Chen’, ‘Chen@ee’, ‘Jan 03, 1992’, 3.2)
- Delete from <TableName> conditional
- Can delete all tuples satisfying some condition
  - *DELETE* FROM Students S WHERE S.name = ‘Smith’
- *Simple Introduction to SQL Commands more functionality described later*

Relational Model: Summary

- A tabular representation of data.
  - Simple and intuitive, currently the most widely used.
  - Integrity constraints can be specified by the DBA, based on application semantics. DBMS checks for violations.
- Powerful and natural query languages exist.