Entity-Relationship Model

1. Entity
2. Attributes
3. Entity Sets
4. Relationship Sets
5. Design Issues
6. Mapping Constraints
7. Weak Entity
8. Keys
9. E-R Diagram
10. Extended E-R Features
11. Design of an E-R Database Schema
12. Reduction of an E-R Schema to Tables

Database Design

- Before we look at how to create and use a database we'll look at how to design one
- Need to consider
  - What tables, keys, and constraints are needed?
  - What is the database going to be used for?
- Conceptual design
  - Build a model independent of the choice of DBMS
- Logical design
  - Create the database in a given DBMS
- Physical design
  - How the database is stored in hardware

Purpose of E/R Model

The E/R model allows us to sketch database schema designs.

Includes some constraints, but not operations.

Designs are pictures called *entity-relationship diagrams*.

Later: convert E/R designs to relational DB designs.

Framework for E/R

- Design is a serious business.
- The “CLIENT” knows they want a database, but they don’t know what they want in it.
- Sketching the key components is an efficient way to develop a working database.

Entities

- Entities represent objects or things of interest
  - Physical things like students, lecturers, employees, products
  - More abstract things like modules, orders, courses, projects
- Entities have
  - A general type or class, such as Lecturer or Module
  - Instances of that particular type, such as Steve Mills, Natasha Alechina (A bunch of names of faculty mebers) are instances of Lecturer
  - Attributes (such as name, email address, to be discussed later again)

Entities Contd..

A *database* can be modeled as:

a collection of entities,

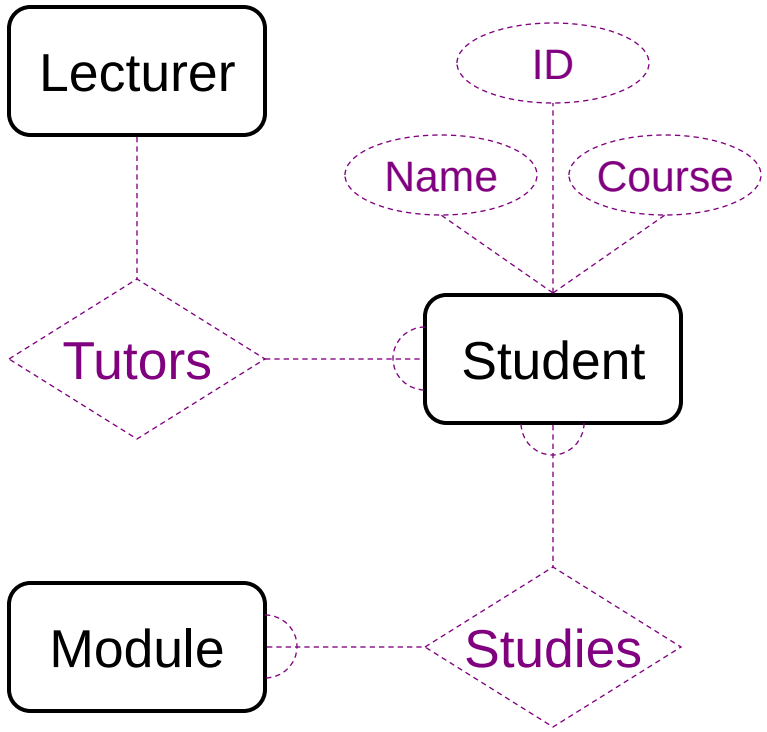
relationship among entities.

An *entity* is also an object that exists and is distinguishable from other objects.

Example: specific person, company, event, plant

Diagramming Entities

- In an E/R Diagram, an entity is usually drawn as a box with rounded corners
- The box is labelled with the name of the class of objects represented by that entity
- We will talk about the rest of the symbols gradually



Entity Sets

- An *entity set* is a set of entities of the same type that share the same properties.
  - Example: set of all persons, companies, trees, holidays

An **entity set** is a **set** of **entities** of the same type (e.g., all persons having an account at a bank). **Entity sets**need not be disjoint. For example, the **entity set**employee (all employees of a bank) and the **entity set**customer (all customers of the bank) may have members in common.Sep 10, 1995

Entities and Entity Sets

An **entity** is an object that exists and is distinguishable from other objects. For instance, John Harris with S.I.N. 890-12-3456 is an entity, as he can be uniquely identified as one particular person in the universe.
An entity may be **concrete** (a person or a book, for example) or **abstract** (like a holiday or a concept).

An **entity set** is a set of entities of the same type (e.g., all persons having an account at a bank).

Entity sets **need not be disjoint**. For example, the entity set *employee* (all employees of a bank) and the entity set *customer* (all customers of the bank) may have members in common.

An entity is represented by a set of **attributes**.

E.g. name, S.I.N., street, city for ``customer'' entity.

The **domain** of the attribute is the set of permitted values (e.g. the telephone number must be seven positive integers).

Formally, an attribute is a **function** which maps an entity set into a domain.

Every entity is described by a set of (attribute, data value) pairs.

There is one pair for each attribute of the entity set.

E.g. a particular *customer* entity is described by the set {(name, Harris), (S.I.N., 890-123-456), (street, North), (city, Georgetown)}.

An analogy can be made with the programming language notion of type definition.The concept of an **entity set** corresponds to the programming language **type definition**.
A variable of a given type has a particular value at a point in time.
Thus, a programming language variable corresponds to an **entity** in the E-R model.

An *entity set* is a logical container for instances of an entity type and instances of any type derived from that entity type. (For information about derived types, see Entity Data Model: Inheritance.) The relationship between an entity type and an entity set is analogous to the relationship between a row and a table in a relational database: Like a row, an entity type describes data structure, and, like a table, an entity set contains instances of a given structure. An entity set is not a data modeling construct; it does not describe the structure of data. Instead, an entity set provides a construct for a hosting or storage environment (such as the common language runtime or a SQL Server database) to group entity type instances so that they can be mapped to a data store.

An entity set is defined within an entity container, which is a logical grouping of entity sets and association sets.

For an entity type instance to exist in an entity set, the following must be true:

The type of the instance is either the same as the entity type on which the entity set is based, or the type of the instance is a subtype of the entity type.

The entity key for the instance is unique within the entity set.

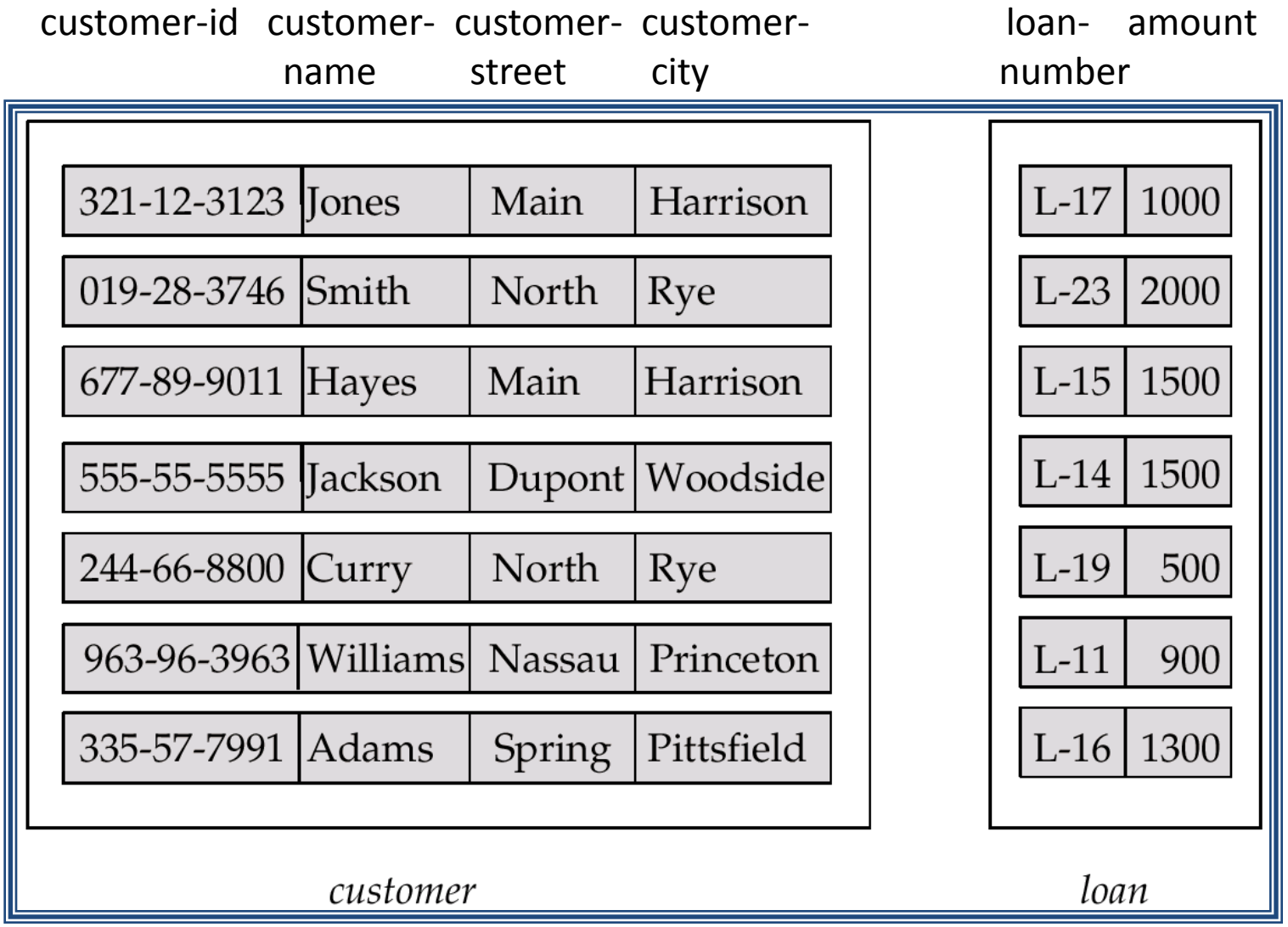
The instance does not exist in any other entity set.

Entity Sets In coordination with Entities

- *Entity* = “thing” or object.

- *Entity set* = collection of similar entities.
  - Similar to a class in object-oriented languages.

Entity Sets *customer* and *loan*

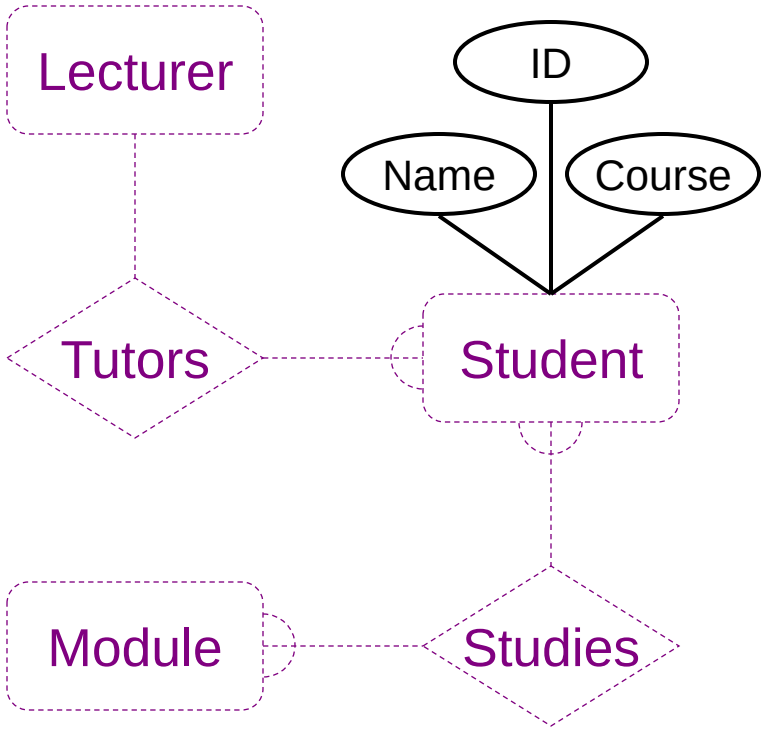


Attributes

- Attributes are facts, aspects, properties, or details about an entity
  - Students have IDs, names, courses, addresses, …
  - Modules have codes, titles, credit weights, levels, …
- Attribute = property of (the entities) and also an entity set.
  - Attributes are simple values, e.g. integers or character strings, not structs, sets, etc.
- Attributes have
  - A name
  - An associated entity
  - Domains of possible values
  - Values from the domain for each instance of the entity they are belong to
- Entities HAVE *attributes*
  - Example: people have *names* and *addresses*

Diagramming Attributes

- In an E/R Diagram attributes may be drawn as ovals
- Each attribute is linked to its entity by a line
- The name of the attribute is written in the oval



Attributes in Reln to Entities

- An entity is represented by a set of **attributes**, that is descriptive properties possessed by all members of an entity set.

Example:

customer = (customer-id, customer-name, customer-street, customer-city)
loan = (loan-number, amount)

- *Domain* – the set of permitted values for each attribute