

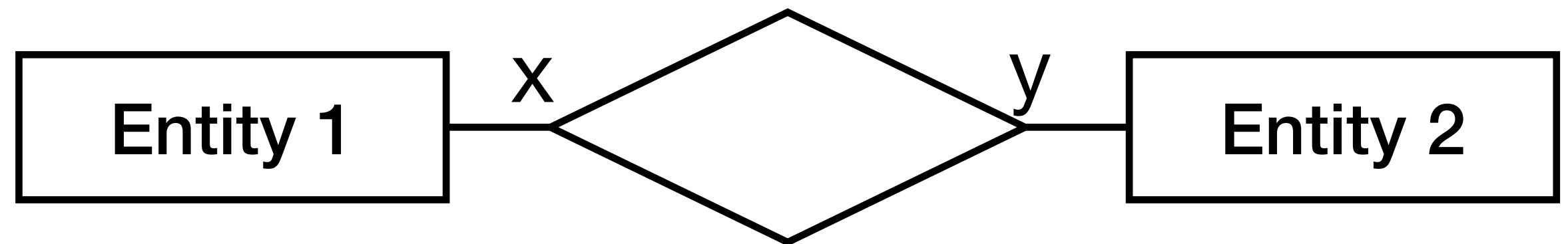
Relational Model

CSCI 220: Database Management and Systems Design

Practice Quiz

- Give the cardinality ratios for the following entities:

Entity 1	Entity 2	x	y
Vehicle	License Plate	1	1
Course	Textbook		
eBay Auction Item	eBay Bid		
Province	Country		
Order	Shipping Label		

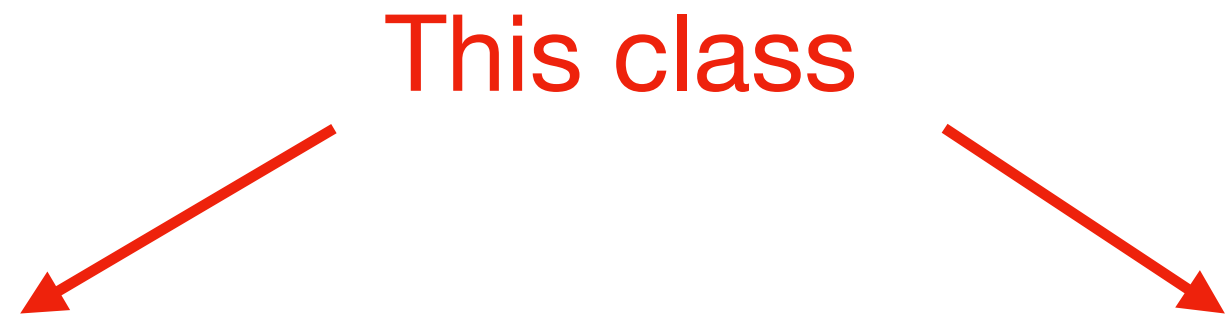


- Then, describe potential candidate keys for each entity

Today you will learn:

- How to design a database with the relational model. In particular:
 - Mathematical foundations of the relational model
 - Basic conversion from an ER model to the relational model
 - Enforcing data integrity with constraints

Review: Where Do Databases Come From?

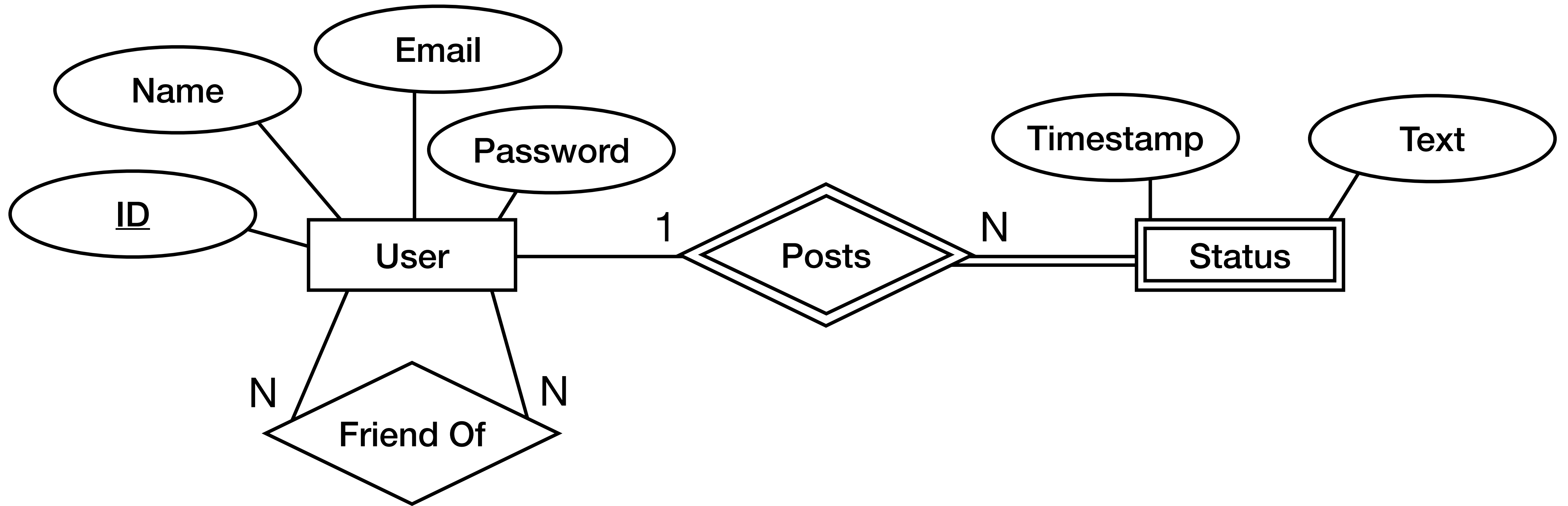
1. Requirements collection
 2. Conceptual design (ER model)
 3. Logical design (first with ER and relational models, then with SQL DDL)
 4. Physical design (DMBS interprets the SQL DDL, storing the DB on disk)
- 
- The diagram consists of the text "This class" in red, positioned above the third item of the list. Two red arrows originate from this text: one points to the phrase "ER and relational models" and the other points to the phrase "SQL DDL".

Often an iterative process (e.g., when ambiguity is discovered, you need to meet with stakeholders again)

Modeling Data Requirements

- Textual descriptions of data requirements are helpful, but language is imprecise
- An **entity-relationship model (ER model)** shows relationships and constraints in a detailed and unambiguous way, and is easy to edit
- An ER model can be converted into the **relational model**
 - Decide which tables to create
 - Then, create the tables with SQL DDL code
 - SQL code retrieves data from tables described in the relational model

Example: Facebook ER Model



Example: Facebook Relational Model

Users

ID	Name	Email	Password
-----------	-------------	--------------	-----------------

Friendship

User ID 1	User ID 2
------------------	------------------

Status Updates

Timestamp	User ID	Text
------------------	----------------	-------------

Example: Facebook Relational Model with Data

Users

ID	Name	Email	Password
111	Peter Story	<u>PeStory@clarku.edu</u>	*****
112	John Magee	<u>JMagee@clarku.edu</u>	*****
113	Li Han	<u>LHan@clarku.edu</u>	*****

Status Updates

Timestamp	User ID	Text
2023-11-29 10:57:01	111	The CMACD building is great!
2023-11-29 11:38:17	113	Welcome back students!
2023-11-29 11:46:29	113	Consider declaring your major!

Friendship

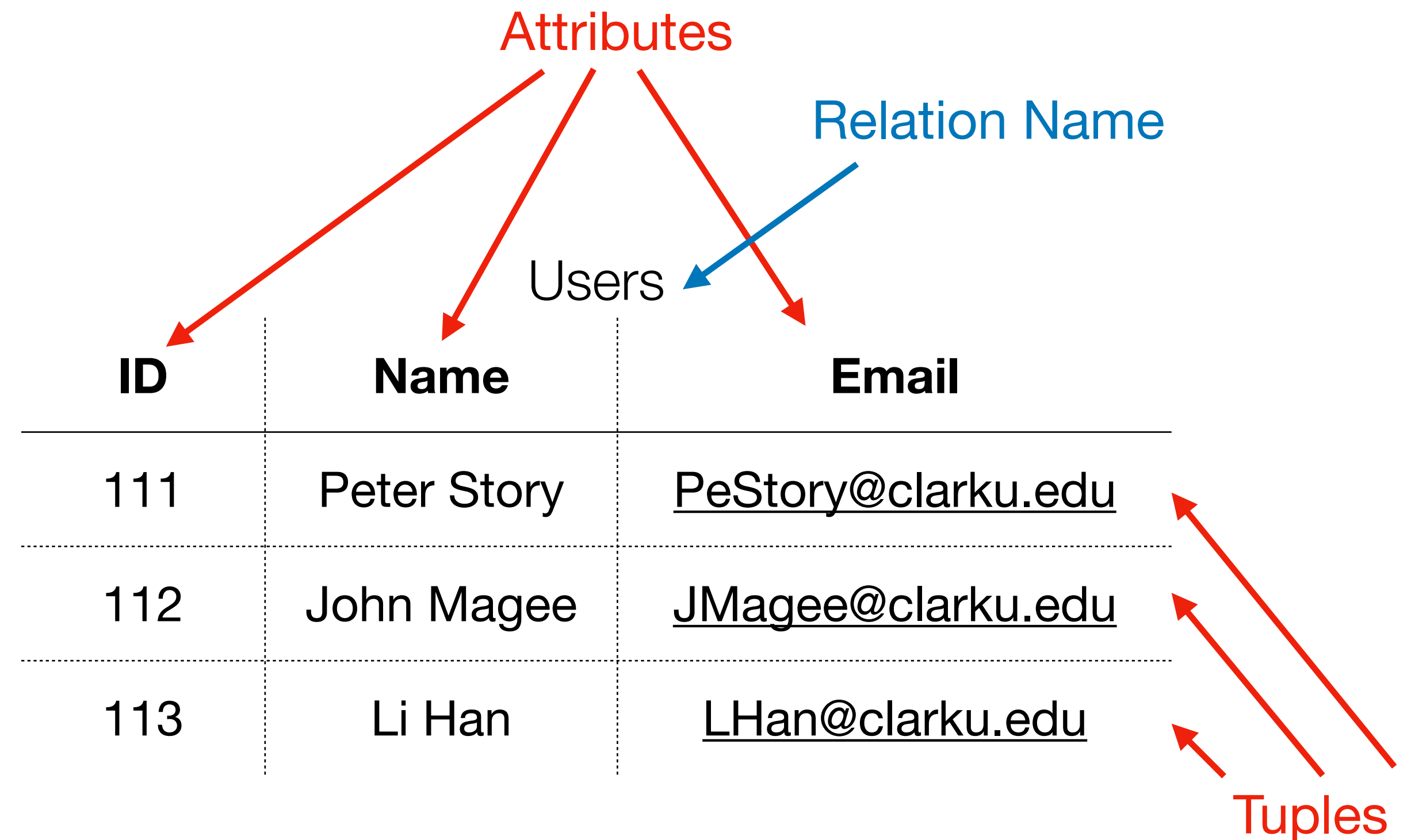
User ID 1	User ID 2
111	112
111	113
112	113

Relational Model History

- Described by researcher Ted Codd of IBM Research in 1970
 - Simple mathematical foundations
- Commercial implementations in the 1980s
- “The relational model is old! Why would I want to use it?”
 - Answer: relational databases were fast in the 1980s. Computers are ten thousand times faster now. Today, relational databases have incredible performance!

Relational Model Concepts

- A **relation** (AKA, table) consists of:
 - **Tuples** (AKA, rows or instances)
 - **Attributes** (AKA, columns)



Domain

- Each attribute has a defined **domain** of allowed values
- Values should be **atomic** (i.e., always considered in their entirety)
- Are these domains atomic?
 - A person's name?
 - A person's phone number?
 - A person's age?

Domains and Data Types

- A domain can be specified as a data type (e.g., integer, string, floating point, etc.)
- Additional restrictions are also possible (e.g., age must be an integer greater than 18)

Domains and NULL

- Databases offer a special value, NULL
- NULL can be used to represent unknown or inapplicable values
- For example, a new employee's HIRE_DATE may initially be NULL
- Only allow NULL if you need to
- Always document what NULL means

Aside: NULL != "NULL" != ""

How a 'NULL' License Plate Landed One Hacker in Ticket Hell

Security researcher Joseph Tartaro thought NULL would make a fun license plate. He's never been more wrong.

Brian Barrett • Aug 13, 2019 8:51 PM

Elena Lacey; Getty Images

Joseph Tartaro never meant to cause this much trouble. Especially for himself.

In late 2016, Tartaro decided to get a [vanity license plate](#). A security researcher by trade, he ticked down possibilities that related to his work: SEGFAULT, maybe, or something to do with vulnerabilities. Sifting through his options, he started typing "null pointer," but caught himself after the first word: NULL. Funny. "The idea was I'd get VOID for my wife's car, so our driveway would be NULL and VOID," Tartaro says.

<https://www.wired.com/story/null-license-plate-landed-one-hacker-ticket-hell/>

Mathematical Formalisms

- Relation schema R
 - Denoted by $R(A_1, A_2, \dots, A_n)$
 - Made up of a relation name R and a list of attributes, A_1, A_2, \dots, A_n
- Attribute A_i : name of a role played by some domain D in the relation schema R
- Domain $\text{dom}(A_i)$: the set of possible values for A_i
- Degree (or arity) of a relation: number of attributes n of its relation schema

Mathematical Formalisms

- Relation $r(R)$ (AKA relation state)
 - Set of n -tuples $r = \{t_1, t_2, \dots, t_m\}$
 - Each n -tuple t
 - Ordered list of n values $t = \langle v_1, v_2, \dots, v_n \rangle$
 - Each value v_i , $1 \leq i \leq n$, is an element of $\text{dom}(A_i)$ or is a special NULL value

Mathematical Formalisms

- Relation $r(R)$ (AKA relation state)
 - Mathematical *relation* of degree n on the domains $\text{dom}(A_1)$, $\text{dom}(A_2)$, ..., $\text{dom}(A_n)$.
 - Meaning, a subset of the Cartesian product of the domains of R 's attributes:
 - $r(R) \subseteq (\text{dom}(A_1) \times \text{dom}(A_2) \times \dots \times \text{dom}(A_n))$

Relation State Example

- Attribute ID has domain {1, ..., MAX_INT}
- Attribute Name has domain {a, aa, aaa, ..., ZZZZZZZ...}
- Attribute Email has domain {a@a.aaa, aa@a.aaa, ..., Z@Z.ZW}
- The relation state (pictured) is a subset of the cartesian product of these domains

Users

ID	Name	Email
111	Peter Story	<u>PeStory@clarku.edu</u>
112	John Magee	<u>JMagee@clarku.edu</u>
113	Li Han	<u>LHan@clarku.edu</u>

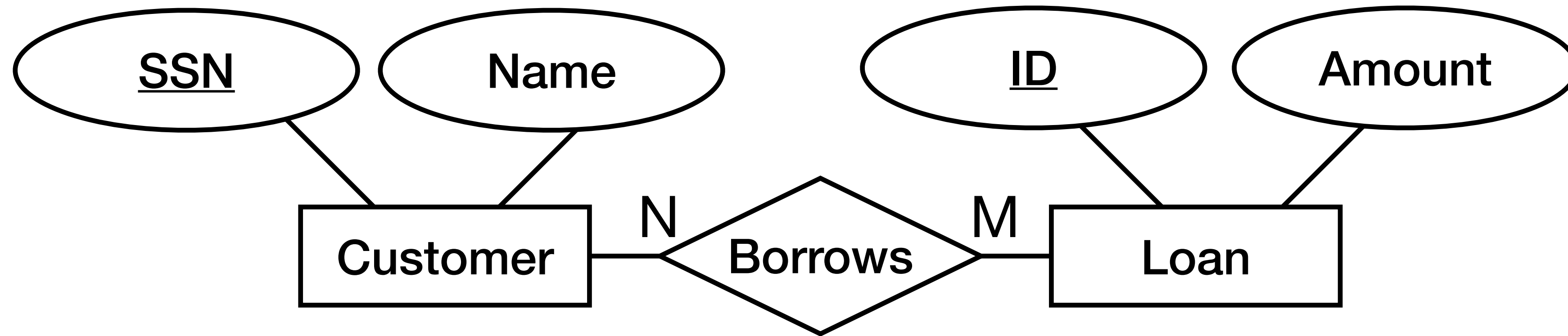
Takeaways from Mathematical Foundations

- Tuples (AKA, rows) don't have a particular order, since the relation state consists of a **set** of tuples
 - In practice, data *is* stored on disk in a particular order, but that order can be unpredictable
 - When we retrieve data, we can sort the data if order matters
- We can use set operations on relations (e.g., union, intersection, etc.)

Takeaways from Mathematical Foundations

- Fun fact: the “relational” in “relational databases” comes from the concept of a mathematical relation
 - Different than the “relationships” between entities in our ER model!

Simple ER to Relational Mapping



Customer

<u>SSN</u>	Name
------------	------

Borrows

<u>SSN</u>	<u>Loan ID</u>
------------	----------------

Loan

<u>ID</u>	Amount
-----------	--------

Key, Superkey, Candidate Key, Primary Key

- **Key attributes:** unique constraint
- **Superkey:** any attribute set that distinguishes identities. Superkeys are possible even if no single attribute is unique.
- **Candidate key:** “minimal superkey” (can’t remove unnecessary attributes)
- **Primary key:** the candidate key chosen to uniquely identifying entities. Underlined.

Customer

<u>SSN</u>	Name
------------	------

Borrows

<u>SSN</u>	<u>Loan ID</u>
------------	----------------

Loan

<u>ID</u>	Amount
-----------	--------

Schema-Based Constraints

- Domain constraints
- Key constraints
- Referential integrity constraints
- The DBMS prevents the database from entering an invalid state by checking each constraint before “committing” attempted changes

Domain Constraints

- Numeric Types
 - Integer Types
 - Arbitrary Precision Numbers
 - Floating-Point Types
- Monetary Types
- Character Types
- Binary Data Types
- Date/Time Types
- Boolean Type
- Enumerated Types
- Geometric Types
- Network Address Types
- Bit String Types
- Text Search Types
- UUID Type
- JSON Types
- ...

<https://www.postgresql.org/docs/current/datatype.html>

Key Constraints

- Primary Key Constraint: each row's primary key must be unique
- Entity Integrity Constraint: no primary key can be NULL

Referential Integrity Constraint

- Foreign keys must exist in the referenced relation
- Foreign keys should be depicted with arrows

ERROR!



Customer

<u>SSN</u>	Name
111-22-3333	Jane Smith
111-22-4444	John Smith

Borrows

<u>SSN</u>	<u>Loan ID</u>
111-22-3333	2

Loan

<u>ID</u>	Amount
1	1000

Application-Based Constraints

- Some constraints require general-purpose logic to enforce
- Sometimes, these are enforced by advanced DBMS features (e.g., triggers and assertions)
 - For example, limiting an order to \$5000 of products
- Other times, these are enforced by application logic
 - For example, enforcing password strength

Relational Model Practice

- Ideas for real world things we can create a relational model for?
- We'll define domain, key, and referential integrity constraints

Future Topics

- Querying and modifying the database with SQL
- Mathematical formalisms of SQL
- Advanced ER to relational mapping