CSC 365

Introduction to Database Systems

What is a Data Model?

Three important components of a data model:

1. Structure
2. Constraints
3. Manipulation

Data Models

- Flat file (1960s - 80s)
- Hierarchical (1970s - 80s)
- Network/Graph (1970s -)

- **Relational (1970s -)**

- Object-oriented (1990s -)
- Semi-structured / XML (1990s-)
- Key-value / non-relational / NoSQL (2000s -)

Relational Model - Core Definitions

- **Tuple**: List of attribute values (loose synonyms: row, record)

- **Relation**: A set of tuples, or informally, a named two-dimensional table of data

- **Attribute**: A named column of a relation

- **Domain**: Data type of an attribute, must be scalar-valued (for example: integer, float, string, date)
    - Special value (*null*) is a member of every domain

Aside: Relation vs Relationship

Relation refers to a set of tuples (or, simply, a table of data) This definition is central in the relational model and relational databases.

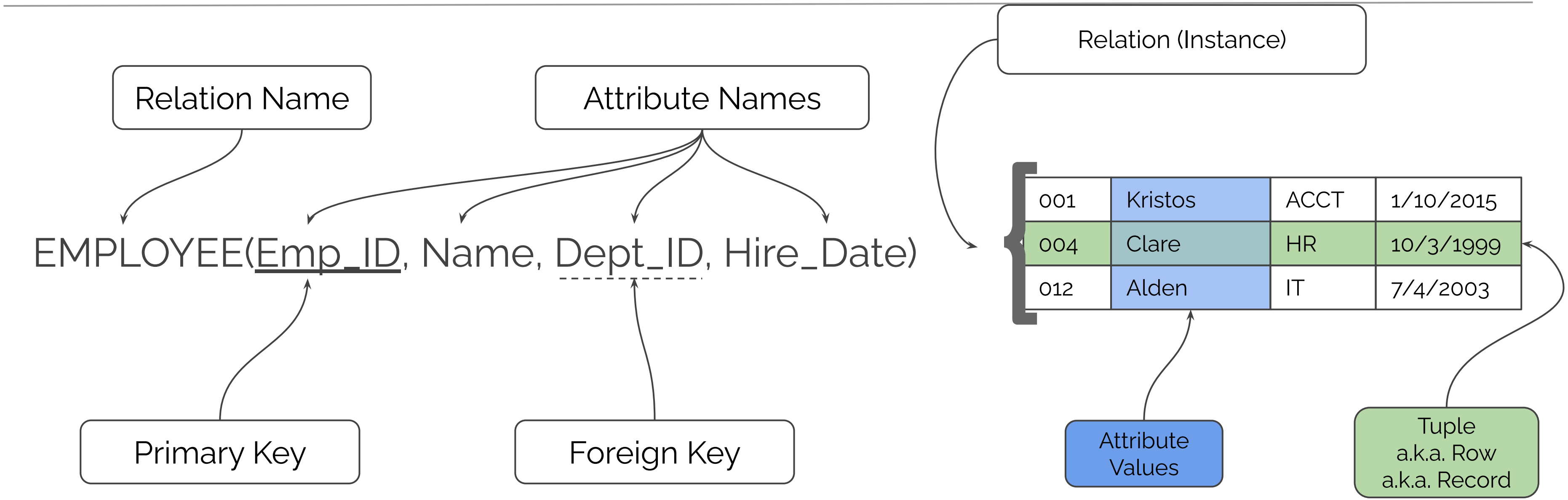
A **relationship** connects tables (or entities) together. Relationships are an important part of the entity-relationship model and E-R diagrams.

Relational Model - Keys

- **[Candidate] Key:** An attribute (or minimal set of attributes) that uniquely identifies a single tuple in a relation.
  - **Composite Key**: A key that consists of more than one attribute. Example: suit and value in a deck of cards.
  - **Primary Key**: A candidate key, chosen by the database designer. Each relation must have one primary key.

- **Foreign Key**: An attribute in a relation that serves as the key of a relation (could be the *same* relation) in the same database. In other words: a reference to a *single* tuple.

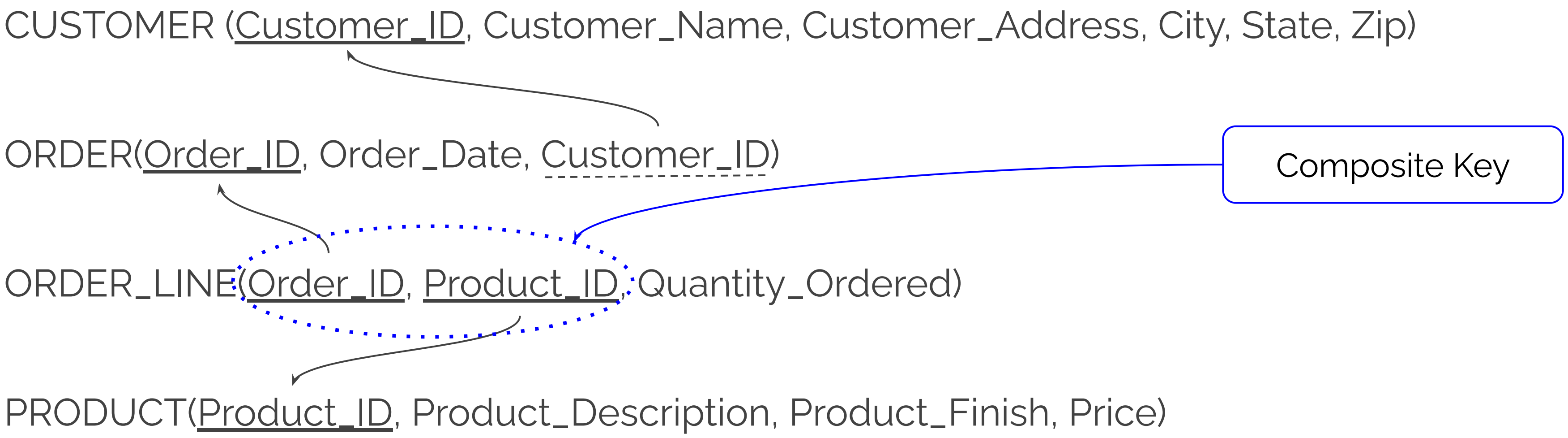
Relational Model - Notation



EMPLOYEE(Emp_ID, Name, Dept_ID, Hire_Date)

001	Kristos	ACCT	1/10/2015
004	Clare	HR	10/3/1999
012	Alden	IT	7/4/2003

Sample Relations



CUSTOMER (Customer_ID, Customer_Name, Customer_Address, City, State, Zip)

ORDER(Order_ID, Order_Date, Customer_ID)

ORDER_LINE(Order_ID, Product_ID, Quantity_Ordered)

PRODUCT(Product_ID, Product_Description, Product_Finish, Price)

Composite Key

Relation Instance

A relation **instance** is the set of tuples for a given relation, often shown with a header.

Relation Schema: EMPLOYEE(Emp_ID, FirstName, LastName, Department)

Relation Instance:

0045600	Raphael	Worsham	Human Resources
0399299	Maureen	Mahedy	Accounting

A Note About Sets

- Attribute names in a relation schema are a **set**, not a list
- Relations themselves are **sets** of rows (tuples) not lists
- Implications?

More Keys

Every relation must have a *single* primary key. We also define:

- **Superkey** (or "superset of a key"): A set of attributes that uniquely identifies a tuple. This set *need not be minimal*
- **[Candidate] Key:** A single attribute (or *minimal* set of attributes) that uniquely identifies each tuple in a relation. Examples: EmployeeID, SSN.
    - A relation's primary key is chosen by the database designer from the identified candidate keys

Relational Model: Structure Recap

- Relation
- Tuple
- Attribute
- Keys: Primary Key, Foreign Key
  - Superkey
  - Candidate key
  - Composite key

Relational Model - Data Integrity

- Rules or constraints that are applied to maintain consistency and correctness of data

- Some are structural rules which are defined universally by the relational model (primary key, foreign key)

- Other rules flow from application-specific rules (Domain Constraints, Assertions) The SQL language includes tools to define these rules.

Domain Constraints

Domain: All the values which a data element may contain.

All values in a column must be taken from the same **domain**

Example: A domain could be expressed as a primitive data type along with a list of allowed values. For example: Attribute `OrderStatus` must be a string and must be one of the values: “Pending”, “Shipped”, “Canceled”, or “Returned”

Primary Key Constraint / Entity Integrity

Every relation must have a primary key. No primary key attribute (or component attribute of a primary key) may be null.

Given the relation:
EMPLOYEE(Emp_ID, Name, Dept_ID, Hire_Date)

These rows violate the primary key (Emp_ID) constraint

Emp_ID	Name	Dept_ID	Hire_Date
002	Simpson, Angela	ACCT	2/29/2016
null	Lucero, Sebastian	HR	6/7/2008
004	Martin, Susan	*null*	6/14/2017
004	Medina, Sebastian	IT	9/1/2019

Referential Integrity Constraint

If there is a foreign key in one relation, either each foreign key value must match a primary key value in some relation, or the foreign key value must be *null*.

Given the two relations:
EMPLOYEE(Emp_ID, Name, Dept_ID, Hire_Date)
DEPARTMENT(Dept_ID, Dept_Name)

Violates referential integrity rule (unless there exists a department with ID “Lucero”)

Emp_ID	Name	Dept_ID	Hire_Date
002	Simpson, Angela	ACCT	2/29/2016
001	Lucero, Sebastian	Lucero	6/7/2008
004	Martin, Susan	*null*	6/14/2017

Dept_ID	Dept_Name
ACCT	Accounting
HR	Human Resources

Assertions

A statement of a constraint or control on the actions of an organization. At the implementation level: a boolean-valued expression that must be true at all times.

Examples:

- Course enrollment must be less than or equal to the capacity of the room in which it is taught
- A person may purchase a ticket to the all-star game only if that person is a season ticket holder

Relational Model - Constraints / Integrity

- Domain Constraints

- Entity Integrity

- Referential Integrity

- Assertions

Relational Model - Data Manipulation

- Theoretical foundations: set theory, relational algebra
    - Five primitive operators: selection, projection, Cartesian product (a.k.a. cross product or cross join), set union, set difference

- Special-purpose programming language built on concepts from relational algebra: Structured Query Language (SQL)
    - SQL will be covered in depth during this course

Relational Model Summary

- Structure
  - Set-based concepts that form the basis of relational databases
- Constraints
  - Enforce a consistent representation of data that aligns with real-world rules
- Manipulation
  - Structured Query Language (SQL) backed by sound theory (Relational Algebra)

Data Independence

One important advantage offered by the relational model is **data independence**, defined by EF Codd as:

> "the independence of application programs and terminal activities from growth in data types and changes in data representation."