Chapter 2: Relational Model

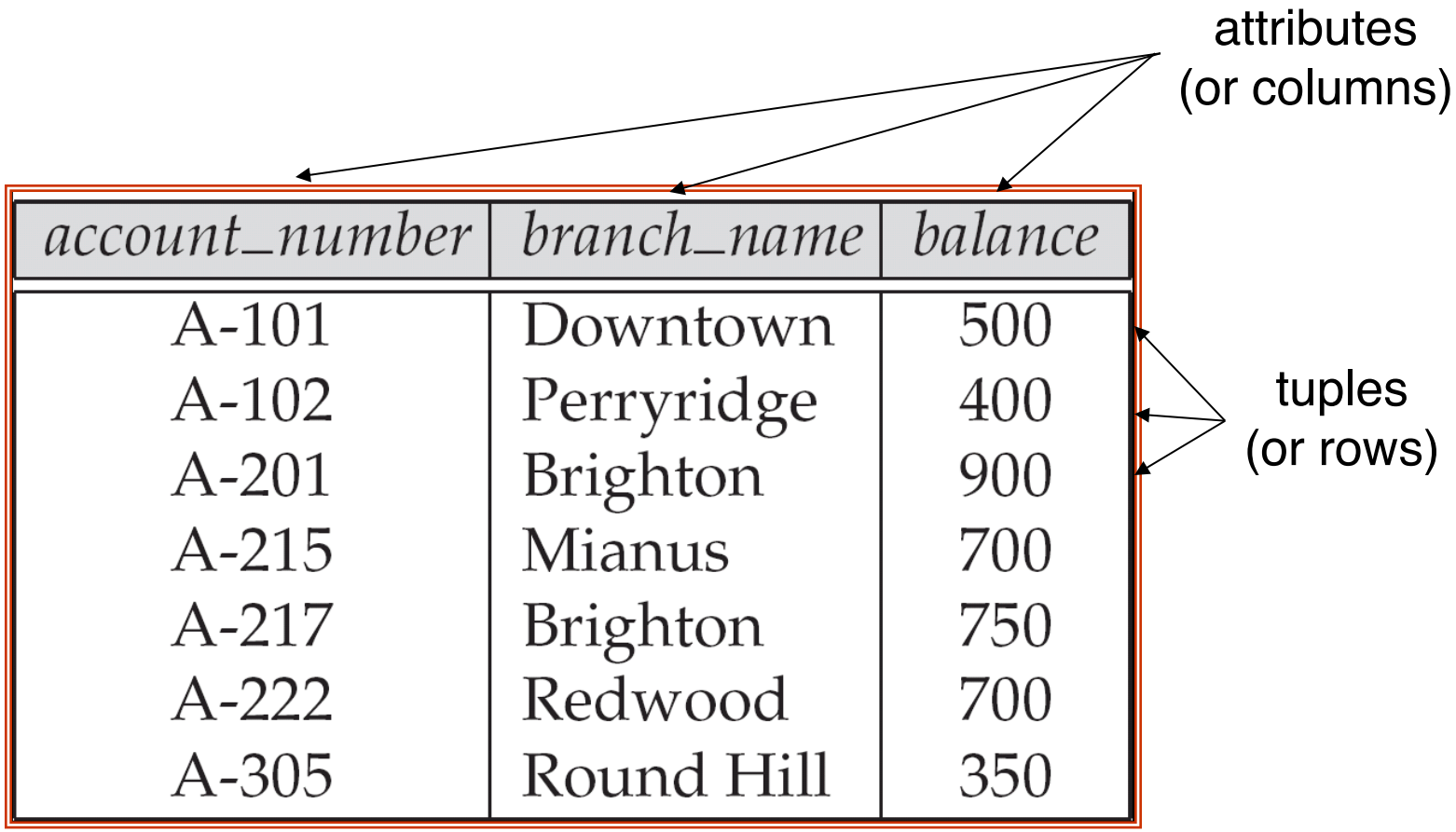
Chapter 2: Relational Model

- Structure of Relational Databases
- Fundamental Relational-Algebra-Operations
- Additional Relational-Algebra-Operations
- Extended Relational-Algebra-Operations
- Null Values
- Modification of the Database

Basic Structure

- **Relational database:** a set of **relations**
- **Relation:** a named data table consisting of two parts:
  - **Schema:** specifies name of relation, consists of a list of attributes and type of each attribute (domains).
    - E.g., Students(*sid*: string, *name*: string, *login*: string, *age*: integer, *gpa*: real).
  - **Instance:** a table of tuples (or called *records, rows*) and attributes (or called *fields, columns*).

Example of a Relation



account_number	*branch_name*	*balance*
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350

Attribute Types

- Each attribute of a relation has a name
- **Domain** of the attribute: The set of allowed values for each attribute
- Attribute values are (normally) required to be **atomic**; that is, indivisible
  - E.g. the value of an attribute can be an account number, but cannot be a set of account numbers
- Domain is said to be atomic if all its members are atomic
- The special value *null* :
  - Signifies that the value is unknown or does not exist
  - A member of every domain
- The null value causes complications in the definition of many operations
  - We shall ignore the effect of null values in our main presentation and consider their effect later

Relation Schema

- A_1, A_2, …, A_n are *attributes*

- $R = (A_1, A_2, ..., A_n)$ is a *relation schema*

  Example:

  $Customer\_schema = (customer\_name, customer\_street, customer\_city)$

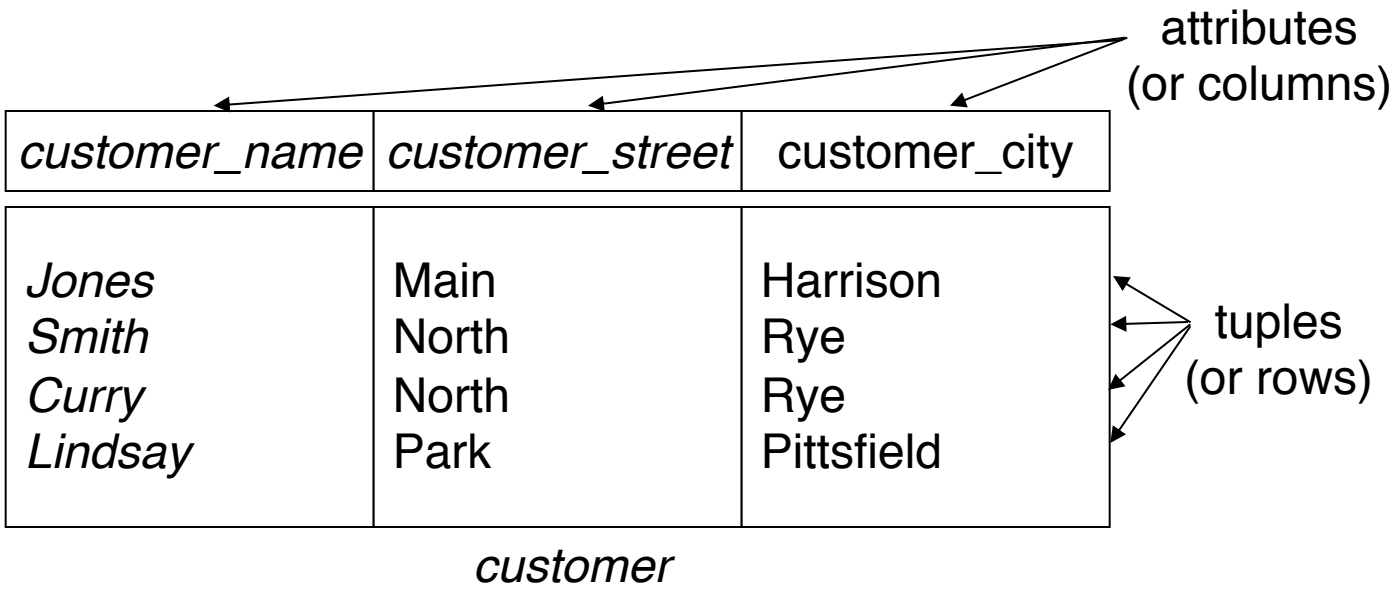
- $r(R)$ denotes a *relation* r on the *relation schema* R

  Example:

  $customer\ (Customer\_schema)$

Relation Instance

- The current values (*relation instance*) of a relation are specified by a table
- An element *t* of *r* is a *tuple*, represented by a *row* in a table



Relations are Unordered

- Order of tuples is irrelevant (tuples may be stored in an arbitrary order)
- Example: *account* relation with unordered tuples

account_number	*branch_name*	*balance*
A-101	Downtown	500
A-215	Mianus	700
A-102	Perryridge	400
A-305	Round Hill	350
A-201	Brighton	900
A-222	Redwood	700
A-217	Brighton	750

Database

- A database consists of multiple relations
- Information about an enterprise is broken up into parts, with each relation storing one part of the information

  account : stores information about accounts
  depositor : stores information about which customer owns which account
  customer : stores information about customers

- Storing all information as a single relation such as
  bank(*account_number, balance, customer_name*, ..)
  results in
  - repetition of information
    - e.g.,if two customers own an account (What gets repeated?)
  - the need for null values
    - e.g., to represent a customer without an account
- Normalization theory (Chapter 7) deals with how to design relational schemas

The *customer* Relation

customer_name	*customer_street*	*customer_city*
Adams	Spring	Pittsfield
Brooks	Senator	Brooklyn
Curry	North	Rye
Glenn	Sand Hill	Woodside
Green	Walnut	Stamford
Hayes	Main	Harrison
Johnson	Alma	Palo Alto
Jones	Main	Harrison
Lindsay	Park	Pittsfield
Smith	North	Rye
Turner	Putnam	Stamford
Williams	Nassau	Princeton

The *depositor* Relation

customer_name	*account_number*
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

Keys

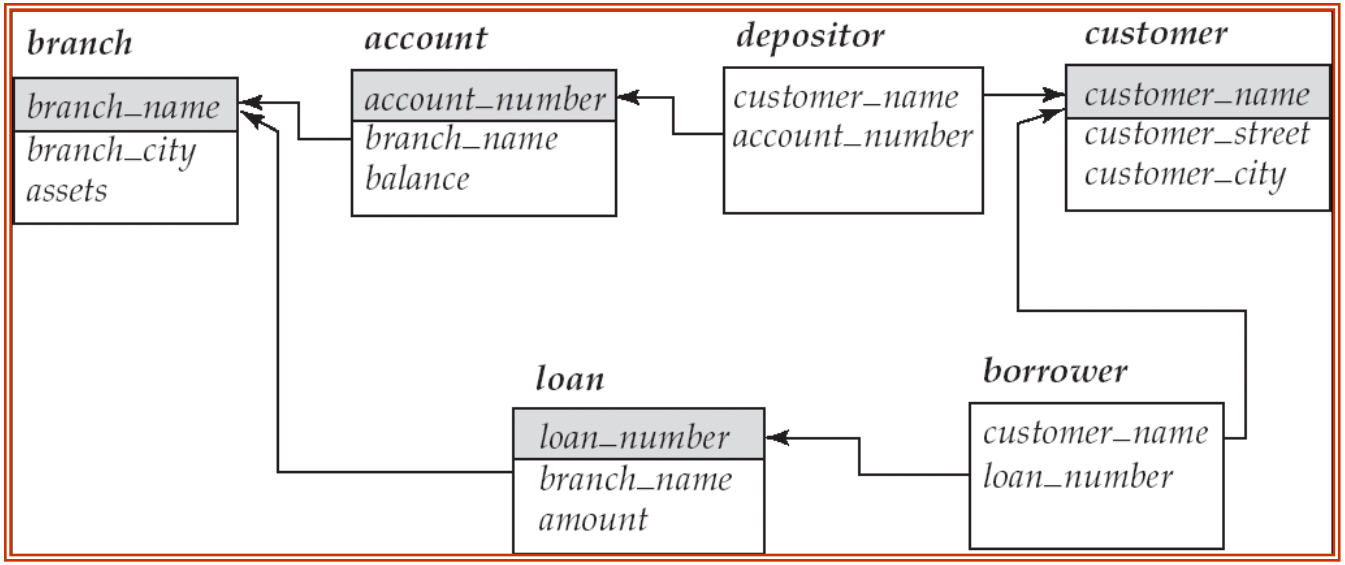
- A set of attribute K is a **superkey** of R if:
  - No two tuples can have same values in all these attributes
- Example: Customer(*customer_name, customer_street, customer_city*)
  - {*customer_name, customer_street*} and {*customer_name*} are both superkeys
  - What about *name*?
  - PS: In real life, an attribute such as *customer_id* would be used instead of *customer_name* to uniquely identify customers, but we omit it to keep our examples small, and instead assume customer names are unique.

Keys (Cont.)

- *K* is a **candidate key** if *K* is minimal
  Example: {*customer_name*} is a candidate key for *Customer*, since it is a superkey and no subset of it is a superkey.
- **Primary key:** If there's more than one candidate keys, one is chosen as the primary key
  - Should choose an attribute whose value never, or very rarely, changes.
  - E.g., email address is unique, but may change

Foreign Keys

- **Foreign key:** A relation schema may have an attribute that corresponds to the primary key of another relation.
  - E.g. *customer_name* and *account_number* attributes of *depositor* are foreign keys to *customer* and *account* respectively.
  - Can refer to itself
  - Only values occurring in the primary key attribute of the **referenced relation** may occur in the foreign key attribute of the **referencing relation**.
- **Schema diagram**



Review of last lecture

- Concepts:
  - Schema
  - Table
  - Relation
  - Attribute
  - Domain
  - Super key
  - Candidate key
  - Primary key

Class Exercise

- 1. Given a relation r defined over the schema R, which of the following can always uniquely identify the tuples in r?

  A. any non-null attributes of R

  B. super key of R

  C. the first attribute in R

  D. R itself

- 2. Given the following relation, list all candidate keys and superkeys.

A	B	C	D
A1	B1	C1	D1
A1	B2	C2	D1
A2	B1	C2	D1