CMSC 424 – Database design
Lecture 5:
Relational Model
Queries
SQL...maybe

Book: Chap. 2
Chap. 3...maybe

Mihai Pop

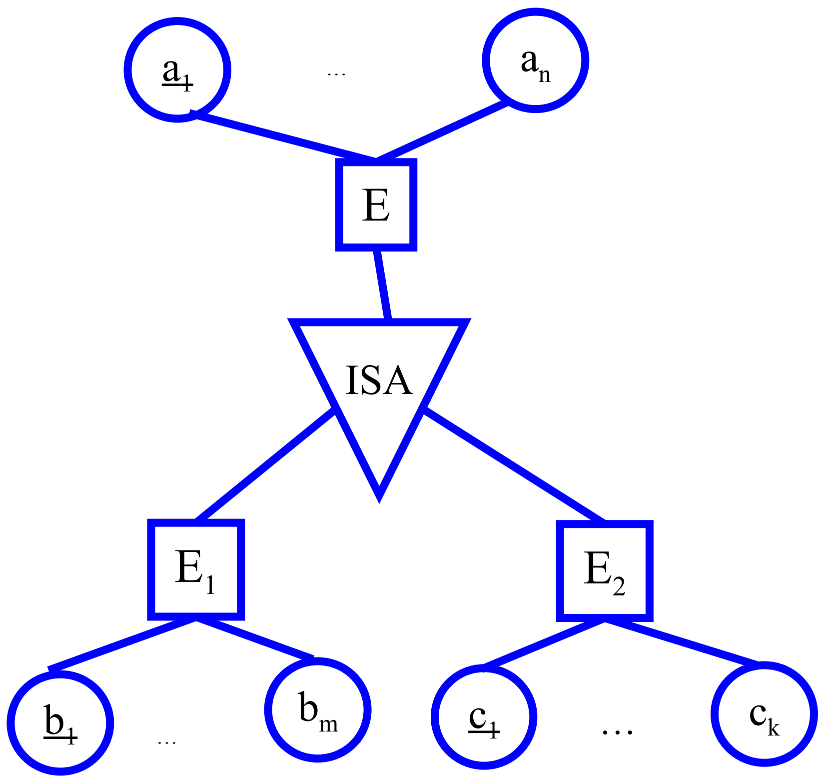
Logistics

- Homework…NOW
- No office hours tomorrow!
- Oracle accounts… have you tried them?
- SQL assignment will be issued Thursday
  - part 1 due Tuesday
  - part 2 due a week from Tuesday (February 26)
  - NO EXTENSIONS

E/R Diagrams & Relations

E/R	Relational Schema

Subclasses



Method 1:

$E = (\underline{a_1}, \ldots, a_n)$

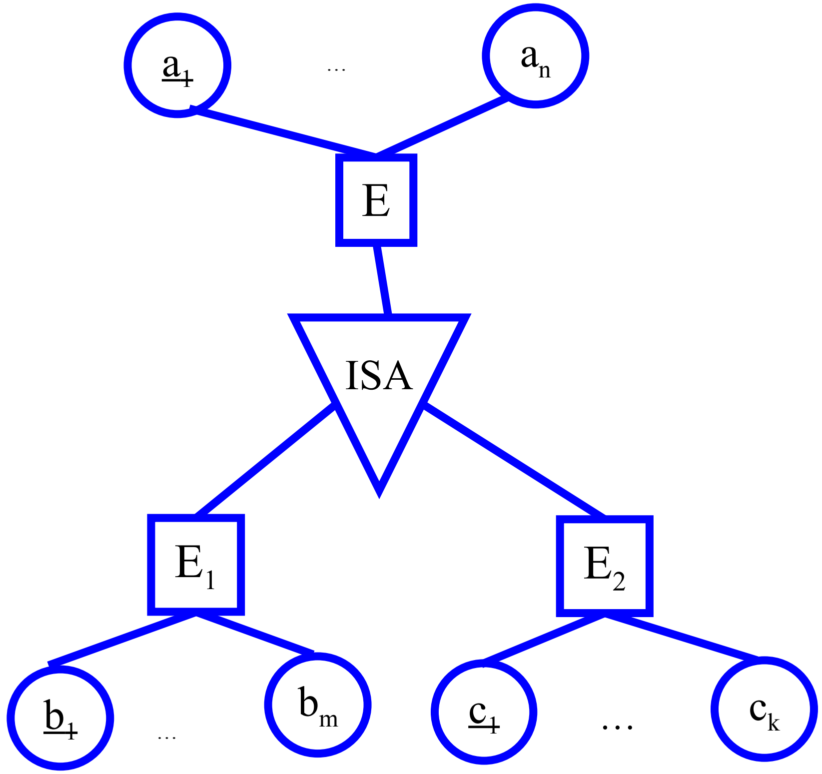
$E_1 = (\underline{a_1}, b_1, \ldots, b_m)$

$E_2 = (\underline{a_1}, c_1, \ldots, c_k)$

E/R Diagrams & Relations

E/R	Relational Schema

Subclasses



Method 1:

$E = (\underline{a_1}, \ldots, a_n)$

$E_1 = (\underline{a_1}, b_1, \ldots, b_m)$

$E_2 = (\underline{a_1}, c_1, \ldots, c_k)$

Method 2:

$E_1 = (\underline{a_1}, \ldots, a_n, b_1, \ldots, b_m)$

$E_2 = (\underline{a_1}, \ldots, a_n, c_1, \ldots, c_k)$

E/R Diagrams & Relations

Subclasses example:

Method 1:

Account = (<u>acct_no</u>, balance)
SAccount = (<u>acct_no</u>, interest)
CAccount = (<u>acct_no</u>, overdraft)

Method 2:

SAccount = (<u>acct_no</u>, balance, interest)
CAccount = (<u>acct_no</u>, balance, overdraft)

Q: When is method 2 not possible?

A: When subclassing is <u>partial</u>

Keys and Relations

- Recall:
  - Keys: Sets of attributes that allow us to identify entities
  - Very loosely speaking, tuples === entities
- Just as in E/R Model:
  - Superkeys, candidate keys, and primary keys

Keys

- Superkey
  - set of attributes of table for which every row has distinct set of values
- Candidate key
  - Minimal such set of attributes
- Primary key
  - DB Chosen Candidate key
  - Plays a very important role
    - E.g. relations typically sorted by this

Keys

- Also act as integrity constraints
  - i.e., guard against illegal/invalid instance of given schema

e.g., Branch = (bname, bcity, assets)

bname	bcity	assets
Brighton	Brooklyn	5M
Brighton	Boston	3M

← *Invalid*

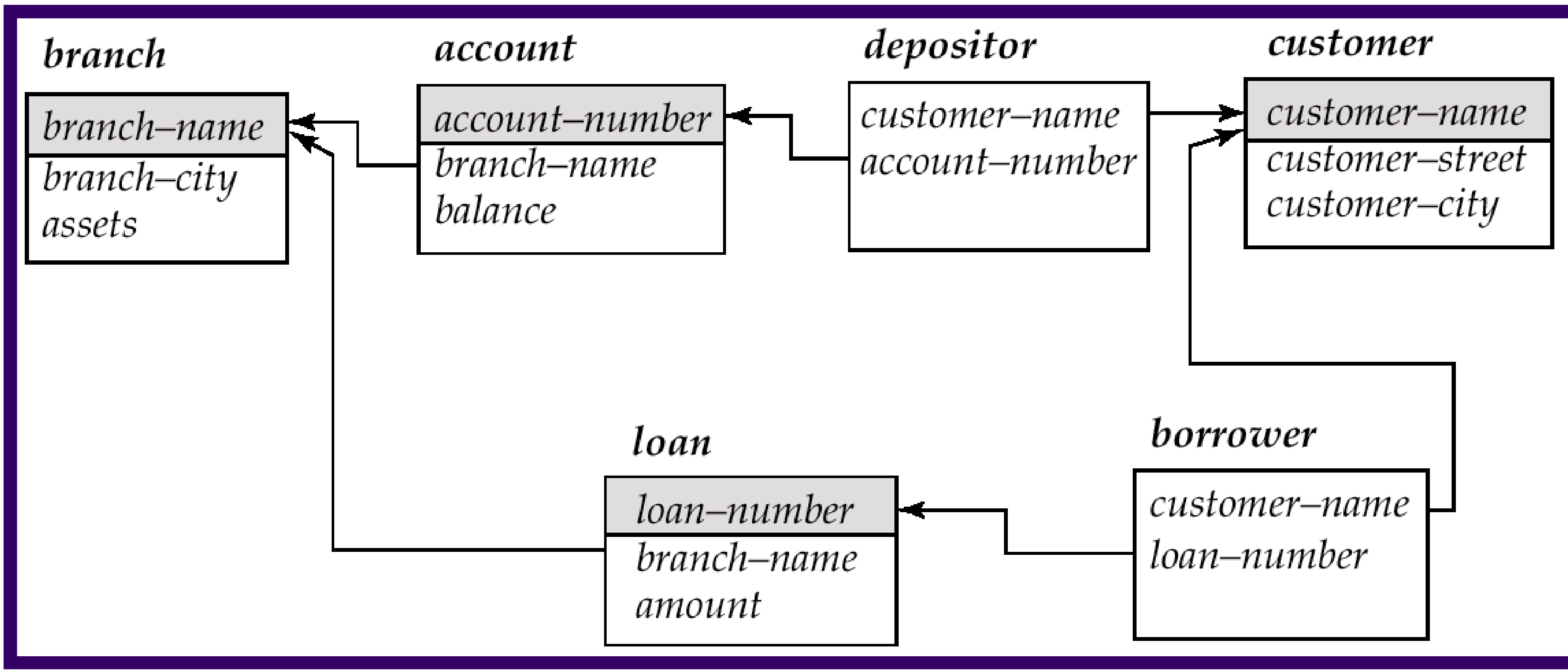
Keys

- In fact, keys are one of the primary ways to enforce constraints/structure
- Consider a one-to-many relationship e.g.
  - Between customers and accounts
  - The relational model will be:
    - Customers(custid, custname,…)
    - Accounts(accountid, custid, balance,…)
  - Allows for multiple accounts per customer, but not multiple customers per account
    - Not possible to store such information
- In other words, constraints will lead to less representation power
  - Contrast with:
    - Customers(custid, custname,…)
    - Accounts(accountid, balance,…)
    - CustomerHasAccounts(custid, accountid)

More on Keys

- Determining Primary Keys
  - If relation schema derived from E-R diagrams, we can determine the primary keys using the original entity and relationship sets
  - Otherwise, same way we do it for E-R diagrams
    - Find candidate keys (minimal sets of attributes that can uniquely identify a tuple)
    - Designate one of them to be primary key
- Foreign Keys
  - If a relation schema includes the primary key of another relation schema, that attribute is called the *foreign key*

Schema Diagram for the Banking Enterprise



Next

- Query language for operating on the relations
- Theoretical:
  - Relational Algebra
  - Tuple Relational Calculus
  - Domain Relational
- Practical:
  - SQL (loosely based on TRC)
  - Datalog

Next…

- Query language for operating on the relations
- Theoretical:
  - Relational Algebra
  - Tuple Relational Calculus
  - Domain Relational
- Practical:
  - SQL (loosely based on TRC)
  - Datalog

Account

bname	acct_no	balance
Downtown	A-101	500
Mianus	A-215	700
Perry	A-102	400
R.H.	A-305	350

Branch

bname	bcity	assets
Downtown	Brooklyn	9M
Redwood	Palo Alto	2.1M
Perry	Horseneck	1.7M
Mianus	Horseneck	0.4M

Depositor

cname	acct_no
Johnson	A-101
Smith	A-215
Hayes	A-102
Turner	A-305

Borrower

cname	lno
Jones	L-17
Smith	L-23
Hayes	L-15
Jackson	L-14

Customer

cname	cstreet	ccity
Jones	Main	Harrison
Smith	North	Rye
Hayes	Main	Harrison
Curry	North	Rye
Lindsay	Park	Pittsfield
Turner	Putnam	Stanford

Loan

bname	lno	amt
Downtown	L-17	1000
Redwood	L-23	2000
Perry	L-15	1500
Downtown	L-14	1500
Mianus	L-93	500
R.H.	L-11	900
Perry	L-16	1300

Example Queries

Find the names of all customers who have a loan at the Perryridge branch.

$$\Pi_{customer\_name} (\sigma_{branch\_name=\text{“Perryridge”}} (\sigma_{borrower.loan\_number = loan.loan\_number}(borrower \times loan)))$$

Find the names of all customers who have a loan at the Perryridge branch but do not have an account at any branch of the bank.

$$\Pi_{customer\_name} (\sigma_{branch\_name = \text{“Perryridge”}} (\sigma_{borrower.loan\_number = loan.loan\_number}(\text{borrower x loan}))) - \Pi_{customer\_name}(\text{depositor})$$

Example Queries

Find the names of all customers who have a loan at the Perryridge branch.

- Query 1

$$\Pi_{customer\_name} (\sigma_{branch\_name = \text{"Perryridge"}} (\sigma_{borrower.loan\_number = loan.loan\_number} (borrower \times loan)))$$

- Query 2

$$\Pi_{customer\_name}(\sigma_{loan.loan\_number = borrower.loan\_number} ((\sigma_{branch\_name = \text{"Perryridge"}} (loan)) \times borrower))$$

Example Queries

Find the largest account balance

Strategy:

- Find those balances that are *not* the largest
  - Rename *account* relation as *d* so that we can compare each account balance with all others
- Use set difference to find those account balances that were *not* found in the earlier step.

The query is:

$$\Pi_{balance}(account) - \Pi_{account.balance}(\sigma_{account.balance < d.balance}(account \times \rho_d(account)))$$