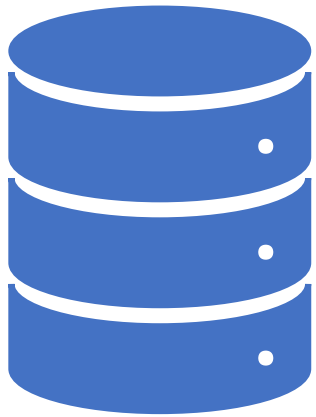


Comp 353 – Databases



Tutorial 1

Summer, 2021

Sections CC and CD

Concordia University - Department of Computer Science & Software
Engineering

Databases

Data ??

At the first steps of designing a Database we should answer two KEY questions (based on the application)?

1. WHAT **concepts (entity sets/classes)** we need to keep data about in the DB? (ex. students, courses, professors,... in a University system)
2. WHAT **relationships among the CONCEPTS** we need to record in the DB? (which students take which courses?)

Databases

Database : an organized collection of data about

- Different kinds of individuals (Entities)
- The relationship between individuals

ER MODEL : an **E**ntity **R**elationship shows how different entity sets in the DB relate to each other, and the constraints that may exist.

Question?

What are the different entities and their relationships in a Bank Database?

Relational Databases

A **relational database** is a set of relations

- **Relations** can be viewed as tables of data

Ex. Database schema

Bank Database (Accounts, Loans, Borrower, Customers)

Accounts

acct_id	branch_name	balance
A-301	New York	350
A-307	Seattle	275
A-318	Los Angeles	550
A-319	New York	80
A-322	Los Angeles	275

Loan

loan_id	branch_name	amount
L-421	San Francisco	7500
L-445	Los Angeles	2000
L-437	Las Vegas	4300
L-419	Seattle	2900

Borrower

cust_name	loan_id
Anderson	L-437
Jackson	L-419
Lewis	L-421
Smith	L-445

Relational Databases

Another example of database schema

- Employee Database (Employees, Departments, Projects)
- Grocery Store Database (Products, Customers, Transactions)

Relations

A relation schema describing its structure includes:

- A set of attributes (ordered columns in a table).
- Each attribute has a domain which specifies the set of valid values for the attribute

Ex. Employee Database (Employees, Departments)

- Employees relation

Schema : (emp_id, emp_name, job_name, manager_id, hire_date, salary, dep_id)

- Department relation

Schema : (dep_id, dep_name, dep_location)

Question: Suggest a domain for each attribute in the above-mentioned relations?

Question?

Considering the Bank Database discussed previously, how we can assume the schema of each relation?

Answer

Considering the Bank Database Application, how to design the schema of each relation? (it is based on the provided Application/Project description!)

- Customers (name, address, phone, acct_id)
- Accounts (acct_id, branch_name, balance)
- Borrowers (cust_name, loan_id)
- Loans (loan_id, branch_name, amount)

Exercise?

Suppose we have a relation HAT with the following schema and provided domains!

HAT(Team, Size, Color)

- domain (Team) = { RedSox, Bruins, Celtics, Patriots, Revolution }
- domain (Size) = { S, M, L, XL }
- domain (Color) = { Black, Blue, White, Red, Green, Yellow }

Show an instance of this relation with two tuples?

Exercise?

How many tuples are possible in HAT relation?

$$|\text{domain}(\text{Team})| \times |\text{domain}(\text{Size})| \times |\text{domain}(\text{Color})| = 5 \times 4 \times 6 = 120$$

Anatomy of an SQL Query

```
SELECT A1, A2, ...  
FROM r1, r2, ...  
[ WHERE C ]  
[... ];
```

- r_i are the relations (tables)
- A_i are attributes (columns)
- C is a condition

SQL Query Example

Query:

Retrieve all tuples for accounts in the Los Angeles branch, with a balance under \$300.

Account

acct_id	branch_name	balance
A-301	New York	350
A-307	Seattle	275
A-318	Los Angeles	550
A-319	New York	80
A-322	Los Angeles	275

SQL Query Example

Query:

Retrieve all tuples for accounts in the Los Angeles branch, with a balance under \$300.

```
SELECT *  
FROM Account  
WHERE branch_name= "Los Angeles" AND balance < 300;
```

Account

acct_id	branch_name	balance
A-301	New York	350
A-307	Seattle	275
A-318	Los Angeles	550
A-319	New York	80
A-322	Los Angeles	275

Example

Query:

Retrieve all branch names that have at least one account.

Account

acct_id	branch_name	balance
A-301	New York	350
A-307	Seattle	275
A-318	Los Angeles	550
A-319	New York	80
A-322	Los Angeles	275

Example

Query:

Retrieve all branch names that have at least one account.

```
SELECT branch_name  
FROM Account;
```

The result is a bag or multiset!

Account

acct_id	branch_name	balance
A-301	New York	350
A-307	Seattle	275
A-318	Los Angeles	550
A-319	New York	80
A-322	Los Angeles	275

Base relation

branch_name
New York
Seattle
Los Angeles
New York
Los Angeles

Derived relation

Example

Query:

Retrieve all branch names that have at least one account.

To eliminate duplicate tuples from the result, we can use **DISTINCT** keyword;

```
SELECT DISTINCT branch_name  
FROM Account;
```

branch_name
New York
Seattle
Los Angeles

An introduction to Join!

How to connect the related data in different tables?

Borrower

cust_name	loan_id
Anderson	L-437
Jackson	L-419
Lewis	L-421
Smith	L-445

Loan

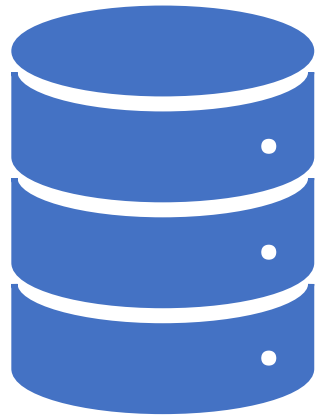
loan_id	branch_name	amount
L-421	San Francisco	7500
L-445	Los Angeles	2000
L-437	Las Vegas	4300
L-419	Seattle	2900

Thanks...

References

- <https://www.w3resource.com/sql-exercises/employee-database-exercise/index.php>
- <https://blog.inf.ed.ac.uk/da18/files/2018/02/da18-7.pdf>
- <http://users.cms.caltech.edu/~donnie/cs121/>

COMP 353 – Databases



Tutorial 2

Summer 2021

Sections CC & CD

Concordia University

Computer Science & Software Engineering

Today's Plan

1. Conceptual DB Design
2. Simple SQL Queries
3. Cartesian Product & Join

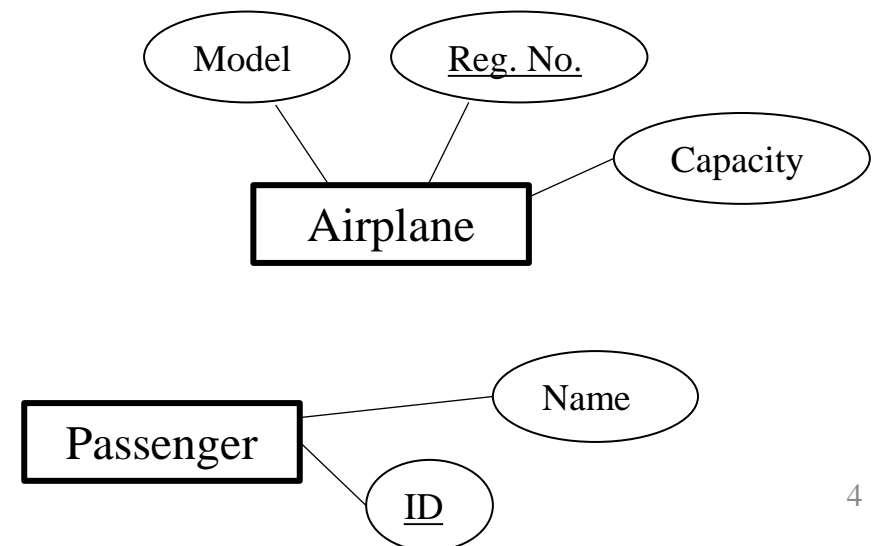
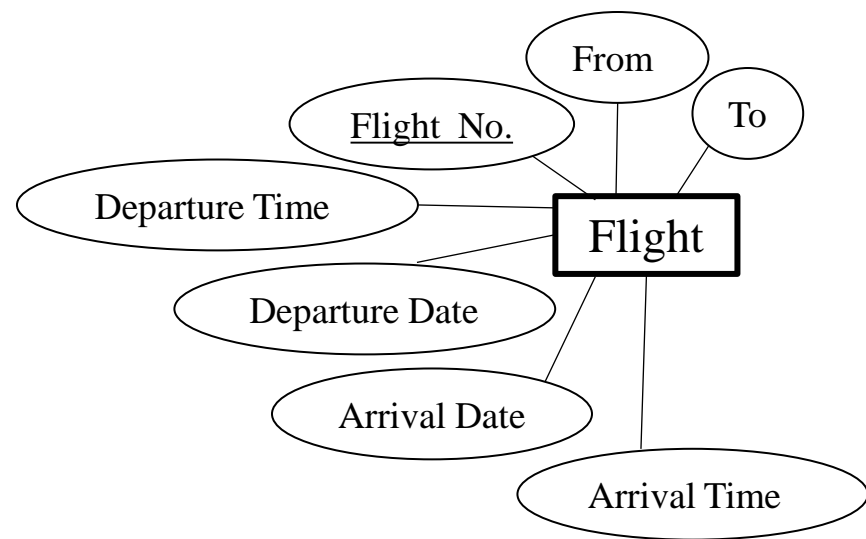
Conceptual DB Design: Example

The Flight Database

The flight database stores details about flights. Each flight has a unique flight number, a departure airport, a destination airport, a departure date and time, and an arrival date and time. Each flight is carried out by a single airplane. An airplane has a model number, a unique registration number, and the capacity to take passengers. Each passenger is identified by his/her name and a unique ID when booking a seat on a flight.

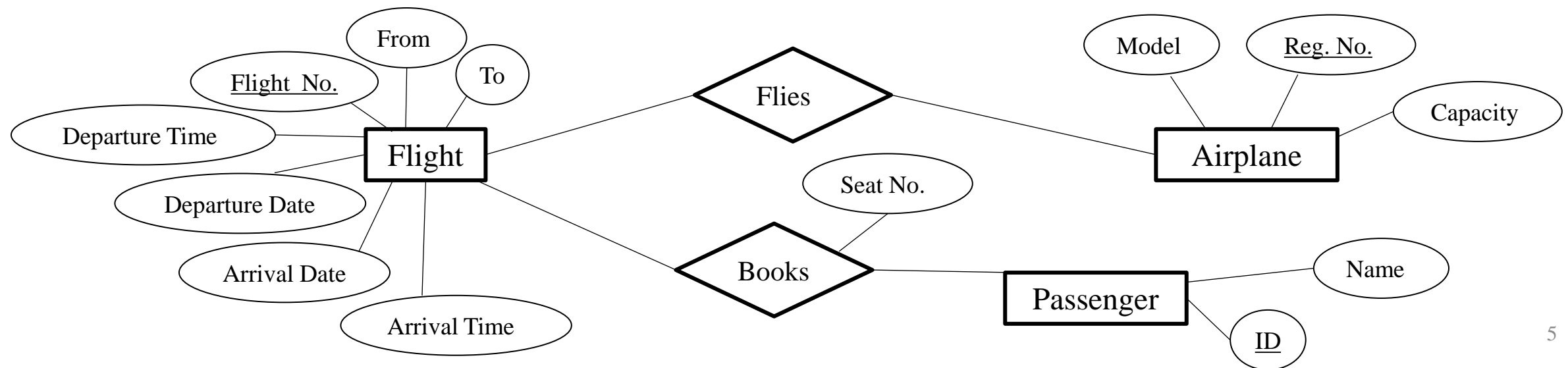
The Flight Database

The flight database stores details about flights. A flight has a unique flight number, a departure airport, a destination airport, a departure date and time, and an arrival date and time. Each flight is carried out by a single airplane. An airplane has a model, a unique registration number, and the capacity to take passengers. Each passenger is identified by his/her name and a unique ID when booking a seat on a flight.



The Flight Database

The flight database stores details about flights. A flight has a unique flight number, a departure airport, a destination airport, a departure date and time, and an arrival date and time. Each flight is carried out by a single airplane. An airplane has a model, a unique registration number, and the capacity to take passengers. A passenger, who has given a name and a unique email address, can book a seat on a flight.



Reminding

Anatomy of an SQL Query

```
SELECT A1, A2, ...  
FROM r1, r2, ...  
[WHERE condition ]  
[... ];
```

- R_i 's are the relations (tables)
- A_i 's are attributes (columns)
- condition is Boolean expression

Example

- Employees relation

Schema : (emp_id, emp_name, job_name, manager_id, hire_date, salary, dep_id)

Query: Write a query that returns the emp_name and salary of the employees who does not belong to department 1234.

Example

- Employees relation

Schema : (emp_id, emp_name, job_name, manager_id, hire_date, salary, dep_id)

Query: Write a query that returns the emp_name and salary, **which is increased by 15%**, for the employees who does not belong to department 1234.

```
SELECT emp_name, 1.15*salary AS "Salary"  
FROM employees  
WHERE dep_id != 1234 ;
```

More Examples

- Employees relation

Schema : (emp_id, emp_name, job_name, manager_id, hire_date, salary, dep_id)

Query: Write a query in SQL to list the employees of department id 3001 or 1001 hired in year 1991.

More Examples

- Employees relation

Schema : (emp_id, emp_name, job_name, manager_id, hire_date, salary, dep_id)

Query: Write a query in SQL to list the employees of department id 3001 or 1001 joined in the year 1991.

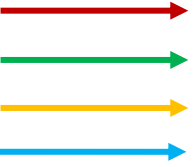
```
SELECT *  
FROM employees  
WHERE (hire_date >= '1991-01-01' AND hire_date < '1992-01-01')  
AND (dep_id =3001 OR dep_id =1001) ;
```

Cartesian product

- Borrower relation
schema : (cust_name, loan_id)
- Loan relation
schema : (loan_id, branch_name, amount)
- Cartesian product of Borrower and Loan (Borrower \times Loan)
`SELECT *`
`FROM Borrower, Loan;`
- The result schema:
(cust_name, borrower.loan_id, loan.loan_id, branch_name, amount)

Cartesian product Example

Borrower



cust_name	loan_id
Anderson	L-437
Jackson	L-419
Lewis	L-421
Smith	L-445


Loan

loan_id	branch_name	amount
L-421	San Francisco	7500
L-445	Los Angeles	2000
L-437	Las Vegas	4300
L-419	Seattle	2900

```
SELECT *  
FROM Borrower, Loan;
```

Result will contain $4 \times 4 = 16$ tuples

- We can use cartesian product to associate related rows between two tables
- But a lot of extra rows are included!



cust_name	borrower. loan_id	loan. loan_id	branch_name	amount
Anderson	L-437	L-421	San Francisco	7500
Anderson	L-437	L-445	Los Angeles	2000
Anderson	L-437	L-437	Las Vegas	4300
Anderson	L-437	L-419	Seattle	2900
Jackson	L-419	L-421	San Francisco	7500
Jackson	L-419	L-445	Los Angeles	2000
Jackson	L-419	L-437	Las Vegas	4300
Jackson	L-419	L-419	Seattle	2900
Lewis	L-421	L-421	San Francisco	7500
Lewis	L-421	L-445	Los Angeles	2000
Lewis	L-421	L-437	Las Vegas	4300
Lewis	L-421	L-419	Seattle	2900
Smith	L-445	L-421	San Francisco	7500
Smith	L-445	L-445	Los Angeles	2000
Smith	L-445	L-437	Las Vegas	4300
Smith	L-445	L-419	Seattle	2900

Join

Borrower

cust_name	loan_id
Anderson	L-437
Jackson	L-419
Lewis	L-421
Smith	L-445

Loan

loan_id	branch_name	amount
L-421	San Francisco	7500
L-445	Los Angeles	2000
L-437	Las Vegas	4300
L-419	Seattle	2900

```
SELECT *  
FROM Borrower, Loan  
WHERE Borrower.loan_id = Loan.loan_id;
```

Result will contain 4 tuples

cust_name	borrower. loan_id	loan. loan_id	branch_name	amount
Anderson	L-437	L-421	San Francisco	7500
Anderson	L-437	L-445	Los Angeles	2000
Anderson	L-437	L-437	Las Vegas	4300
Anderson	L-437	L-419	Seattle	2900
Jackson	L-419	L-421	San Francisco	7500
Jackson	L-419	L-445	Los Angeles	2000
Jackson	L-419	L-437	Las Vegas	4300
Jackson	L-419	L-419	Seattle	2900
Lewis	L-421	L-421	San Francisco	7500
Lewis	L-421	L-445	Los Angeles	2000
Lewis	L-421	L-437	Las Vegas	4300
Lewis	L-421	L-419	Seattle	2900
Smith	L-445	L-421	San Francisco	7500
Smith	L-445	L-445	Los Angeles	2000
Smith	L-445	L-437	Las Vegas	4300
Smith	L-445	L-419	Seattle	2900

Example

Query:

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

Borrower

cust_name	loan_id
Anderson	L-437
Jackson	L-419
Lewis	L-421
Smith	L-445

Loan

loan_id	branch_name	amount
L-421	San Francisco	7500
L-445	Los Angeles	2000
L-437	Las Vegas	4300
L-419	Seattle	2900

Example

Query:

Retrieve the names of all customers with loans at the Seattle branch.

Borrower

cust_name	loan_id
Anderson	L-437
Jackson	L-419
Lewis	L-421
Smith	L-445

Loan

loan_id	branch_name	amount
L-421	San Francisco	7500
L-445	Los Angeles	2000
L-437	Las Vegas	4300
L-419	Seattle	2900

```
SELECT cust_name
FROM Borrower, Loan
WHERE Borrower.loan_id = Loan.loan_id AND branch_name = "Seattle";
```

cust_name
Jackson

- Each table can be followed by an alias Borrower **AS** B, or even just Borrower B.

```
SELECT cust_name
FROM Borrower B, Loan L
WHERE B.loan_id = L.loan_id AND branch_name = "Seattle";
```

More Examples

- Employees relation

Schema : (emp_id, emp_name, job_name, manager_id, hire_date, salary, dep_id)

- Department relation

Schema : (dep_id, dep_name, dep_location)

Query: Write a query in SQL to list the name, job name, annual salary, department id, department name of every employee who earns \$60,000 in a year or does not work as an ANALYST.

More Examples

- Employees relation

Schema : (emp_id, emp_name, job_name, manager_id, hire_date, salary, dep_id)

- Department relation

Schema : (dep_id, dep_name, dep_location)

Query: Write a query in SQL to list the name, job name, annual salary, department id, department name of every employee whose annual salary is \$60,000 is not working as an ANALYST.

```
SELECT e.emp_name, e.job_name, (12*e.salary)"Annual Salary", e.dep_id, d.dep_name
FROM employees e, department d
WHERE e.dep_id = d.dep_id AND (((12*e.salary)>= 60000) OR (e.job_name !=
'ANALYST'));
```

More Examples

- Employees relation

Schema : (emp_id, emp_name, job_name, manager_id, hire_date, salary, dep_id)

Query: Write a query in SQL to list the name, job name, manager id, salary, manager name, manager's salary of every employee.

More Examples

- Employees relation

Schema : (emp_id, emp_name, job_name, manager_id, hire_date, salary, dep_id)

Query: Write a query in SQL to list the name, job name, manager id, salary, manager name, manager's salary for every employee.

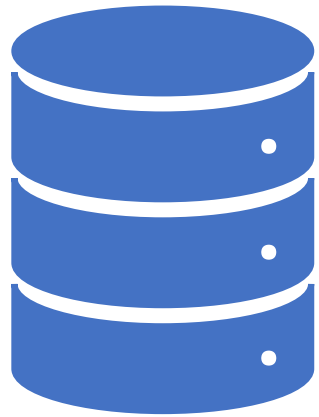
```
SELECT w.emp_name, w.job_name, w.manager_id, w.salary, m.emp_name "Manager",  
m.salary "Manager_Salary"  
FROM employees w, employees m  
WHERE w.manager_id = m.emp_id;
```

Thanks...

References

- <https://www.w3resource.com/sql-exercises/employee-database-exercise/index.php>
- <https://www.oreilly.com/library/view/learning-mysql/0596008643/ch04s04.html>

COMP 353 – Databases



Tutorial 3

Summer 2021

Sections CC & CD

Concordia University

Computer Science & Software Engineering

Today's Plan

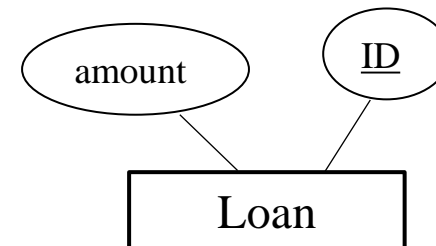
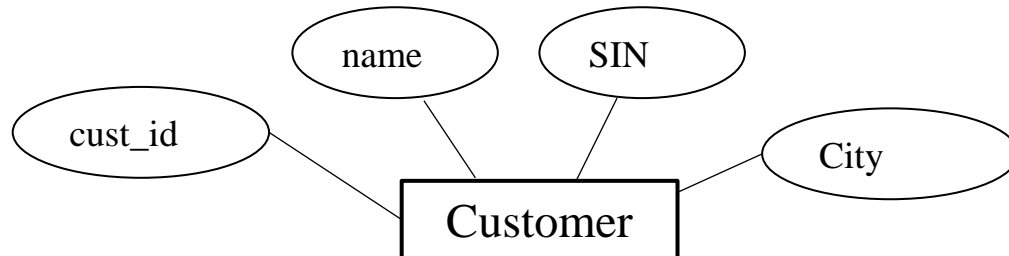
1. Reviewing the concepts related to ER Model
2. Converting ER to Relational Model

Review; Entity sets & Entities

- ✓ An **entity set** is a named collection of **entities** of the same type, i.e., with the same attributes.

Example:

- customers and loans are examples of **entity sets**
- (123, Jackson, 555-555-555, Montreal) is an instance **entity** of customer
- (L-14, 1500) is an instance **entity** of loan

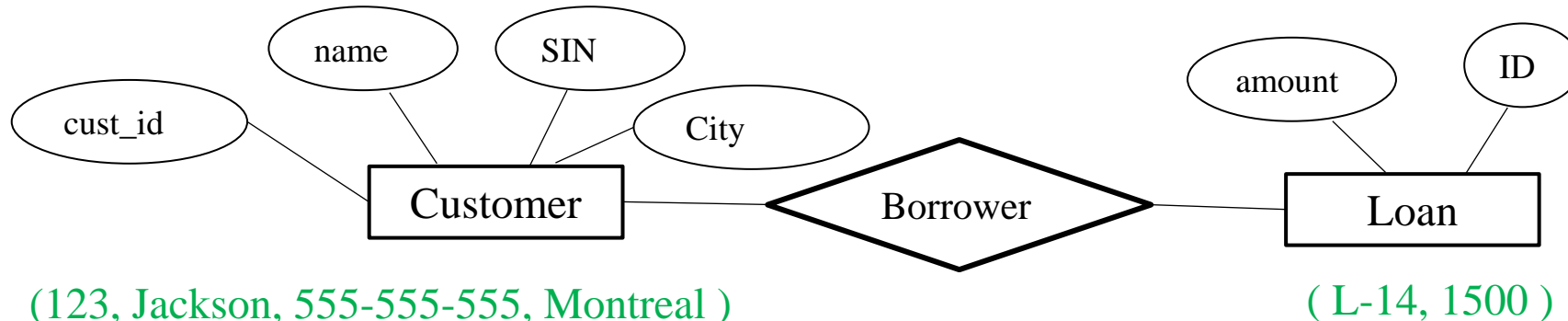


Review; Relationship sets & Relationships

- ✓ Entity sets participate in **relationship sets**. A **relationship set** is a set of relationships of the same type.
- ✓ Specific entities from contributing entity sets participate in a **relationship instance**

Example:

- borrower is a **relationship set** between customer and loan
- borrower contains a **relationship** instance that associates customer “Jackson” and loan “L-14”



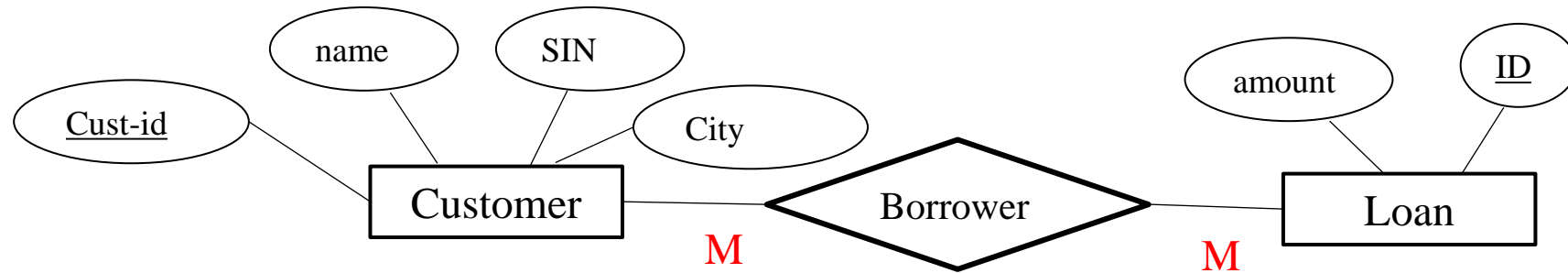
Relationships Multiplicity

For each relationship set R between entity sets A and B, we need to identify the number of entities to which another entity can be associated via R?

Example:

Each customer can have (borrow) **multiple** loans.

Each loan can be borrowed by **several** customers



Borrower is a **Many-to-Many** relationship

Entity set Keys

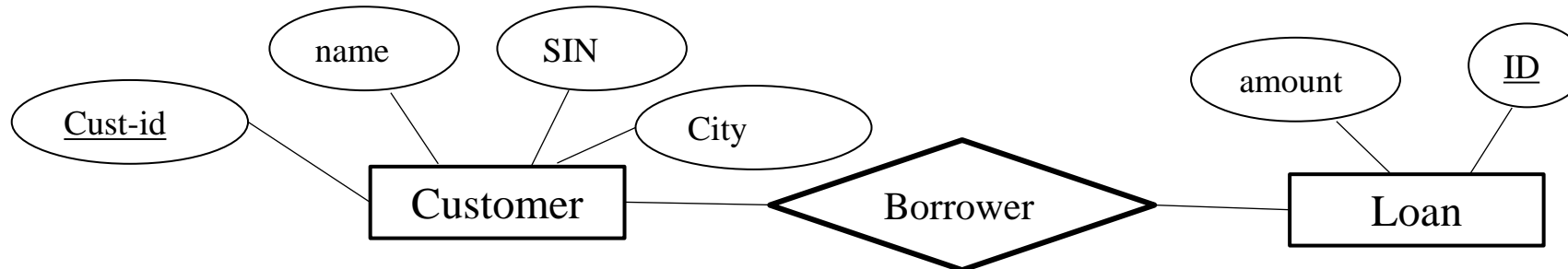
Entities in an entity-set must be uniquely distinguishable using their values

Key: a set of attributes of a relation/class that can uniquely identify an entity

Example:

In customer entity-set

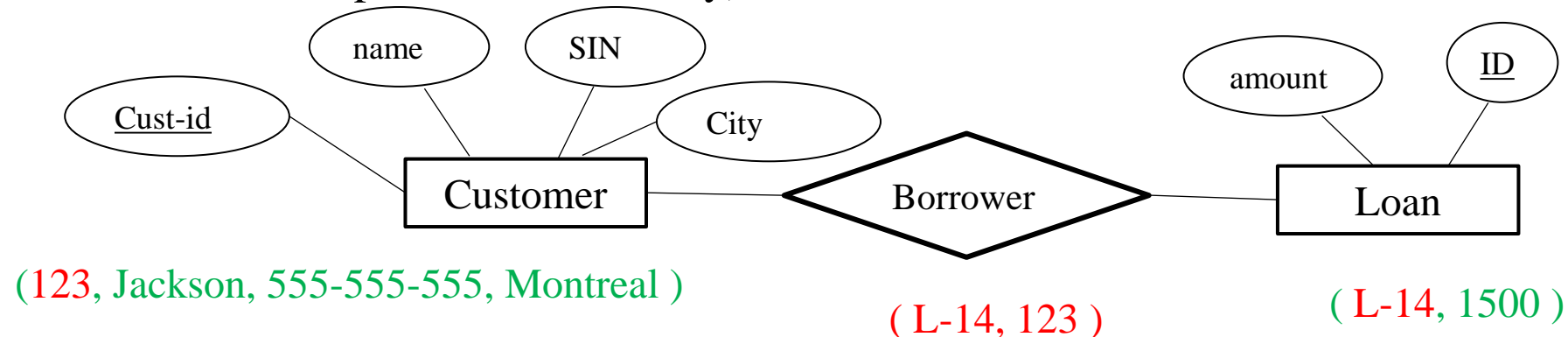
- Using customer name as a key may not be a good DECISION: customers can have the same name
- Either SIN or cust-id can be “considered” as the key attribute. They are called *Candidate* keys. Which one to choose?



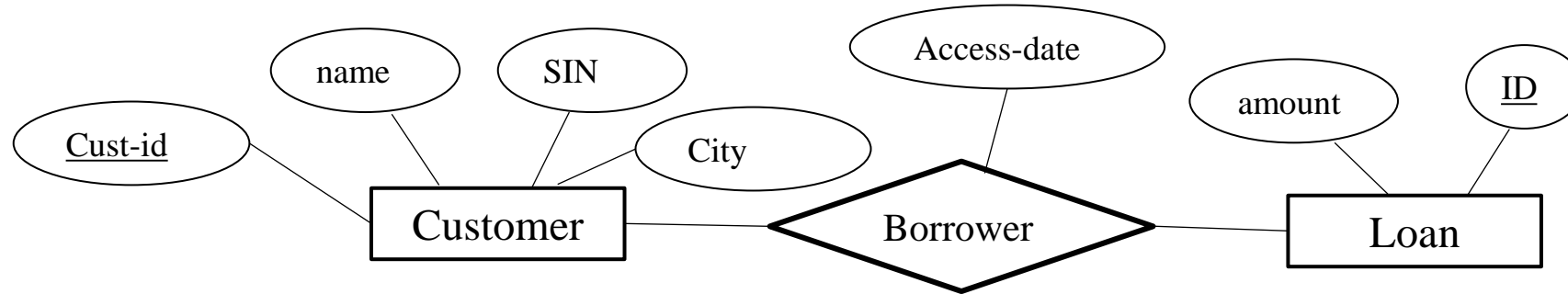
Keys and Relationship Sets

We also need to be able to distinguish between relationships in a relationship-set

- Relationships are identified by the entities participating in the relationship
- The **primary key (or keys) of participating entities**, can help in identifying the relationship instances depending on the multiplicity of the relationship. (Sometimes we need to include more attributes from the relationship set to define a key)



Converting ER to Relational Model



Relational Schema

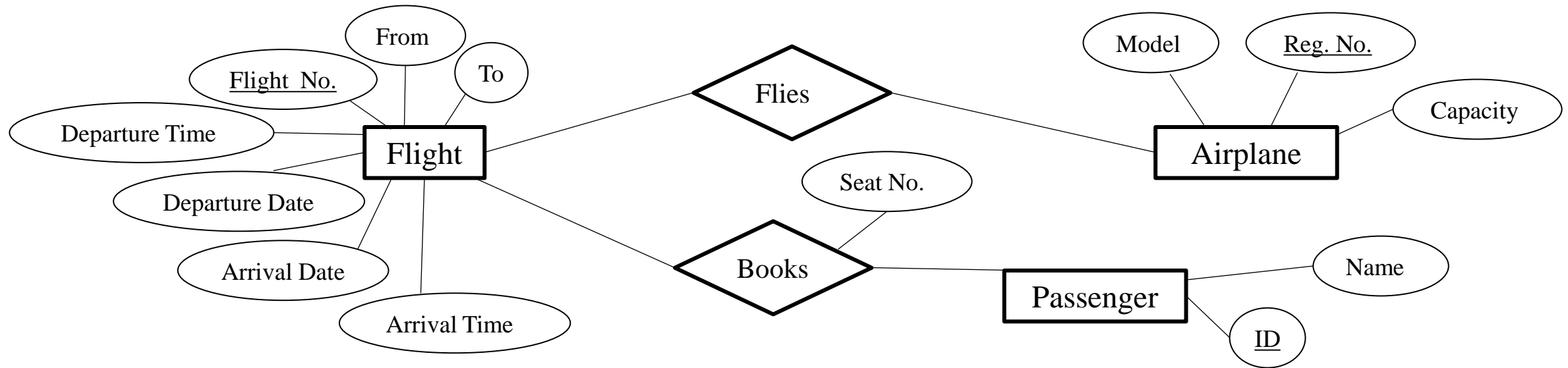
customer(cust-id, name, city, SIN)

loan (loan-id, amount)

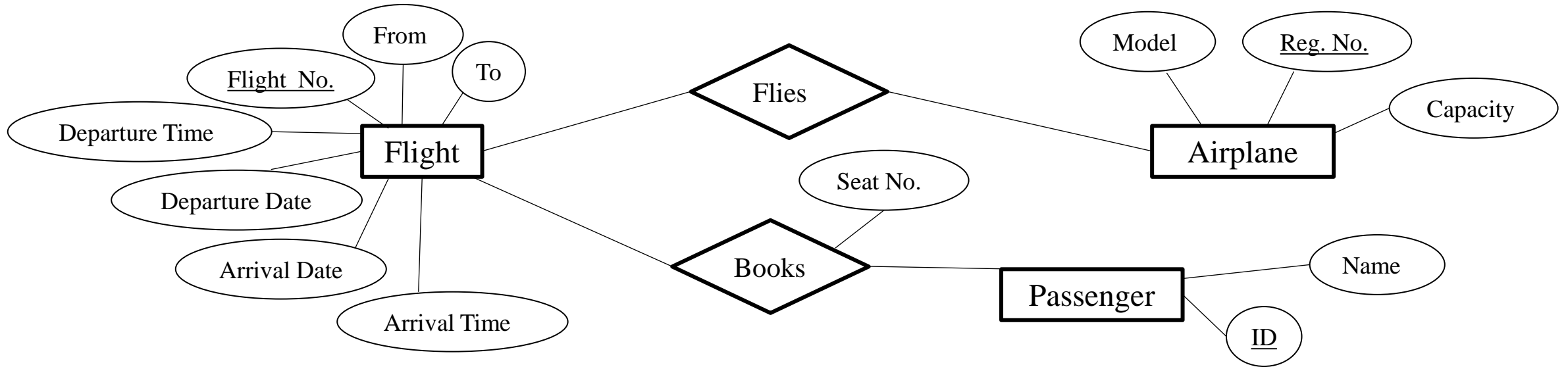
borrower(cust-id,loan-id, access-date)

Converting ER to Relational Model

Ex. Convert the following ER Diagram of a simplified Flight Database to a relational database design/model. That is, to a set of tables/relations.



Converting ER to Relational Model



Step 1. Converting entity sets.

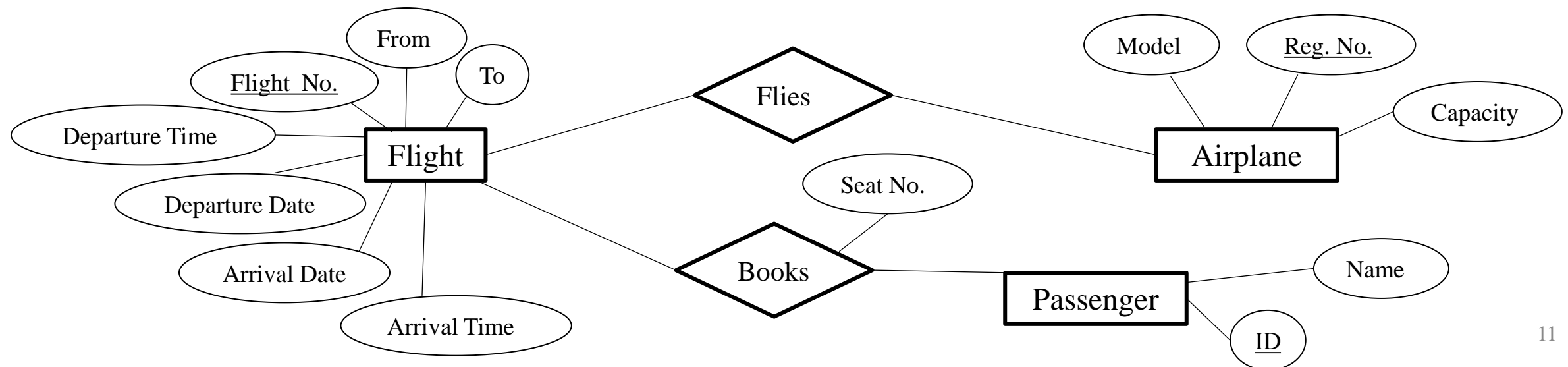
Step 2. Converting relationship sets.

Step 3. Identifying the Key (attributes) of each table/relation obtained.

Converting ER to Relational Model

Ex. To identify the key attribute(s) of each relation, we need to know about the following blank spaces.

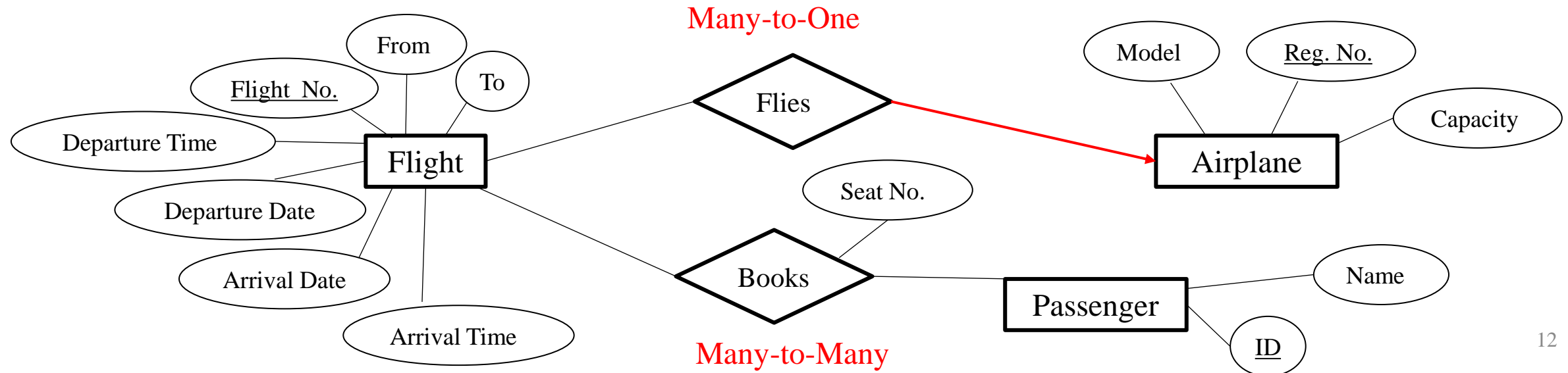
- One Passenger can book Flight(s).
- Each Flight can be booked by Passenger(s).
- Each airplane can carry out Flight(s).
- Each flight can be carried out by Airplane(s).



Converting ER to Relational Model

Regarding the following ER Diagram, fill the following blank spaces;

- One Passenger can book **several** Flight(s).
- Each Flight can be booked by **several** Passenger(s).
- Each airplane can carry out **several** Flight(s).
- Each flight can be carried out by **one** Airplane.



Converting ER to Relational Model

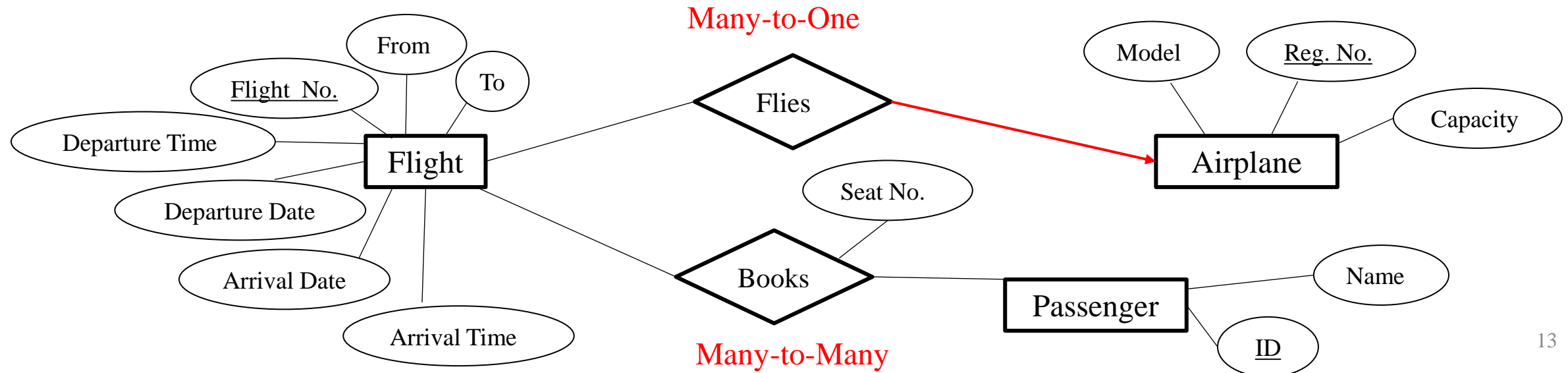
Flights (Flight No. , From, To, Departure Date, Departure Time, Arrival Date, Arrival Time)

Airplanes (Reg. No. , Model, Capacity)

Passengers (ID , Name)

Books (Flight No. , Passenger ID, Seat No.)

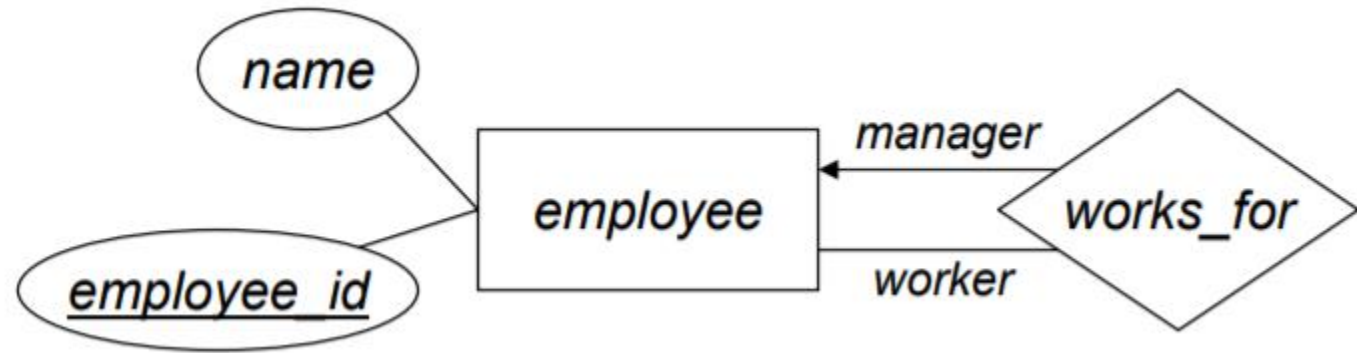
Flies (Flight No. , Reg. No.)



Converting ER to Relational Model

Example: a relationship-set named `works_for`, specifying employee/manager assignments

Relationship involves two copies of EMPLOYEE entity set.



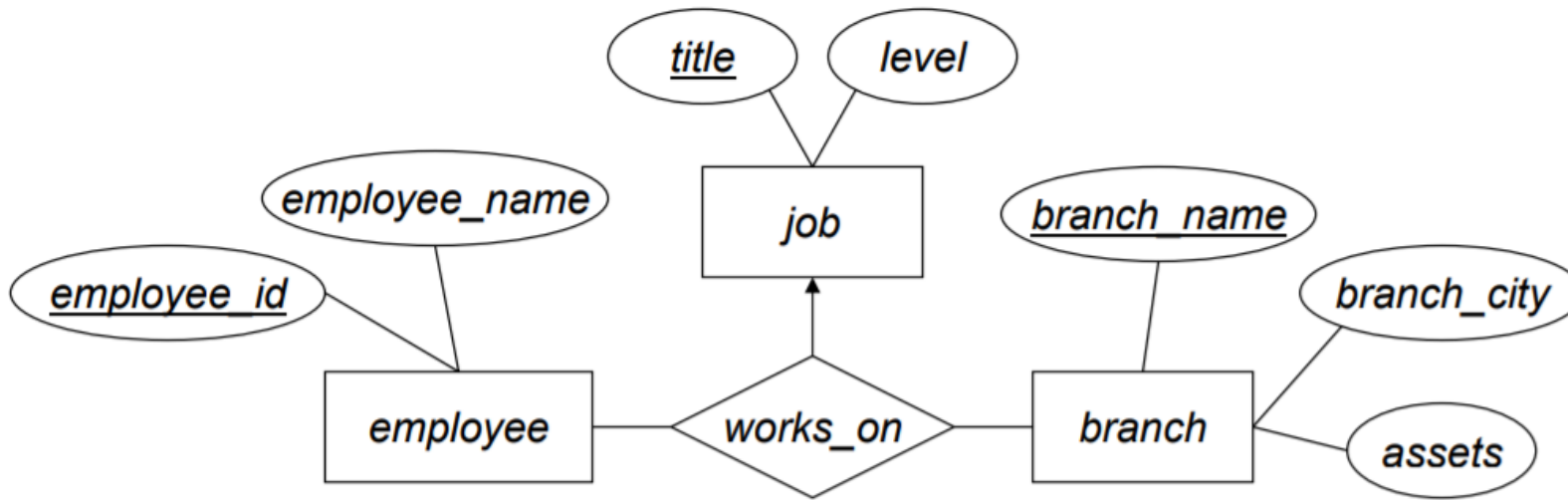
Relational Schema

`employee (employee_id, name)`

`works_for(employee_id, manager_id)`

“Many” side is used for primary key.

Example



Relational Schema

job (title, level)

employee (employee-id, employee-name)

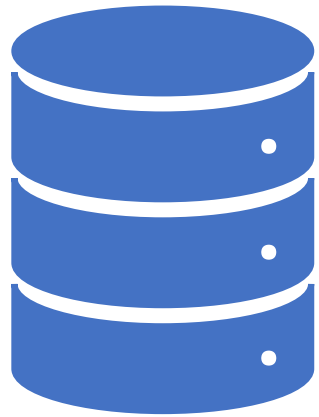
branch (branch-name, branch-city, assets)

Works-on (employee-id, branch-name, job-title)

What is the primary key of Works-on??

WELCOME!

COMP 353 – Databases



Tutorial 4

Summer 2021

Sections CC , CD

Concordia University

Computer Science & Software Engineering

Today's Plan

1. Data Representation
2. Data definition
3. Data Manipulation

The story so far

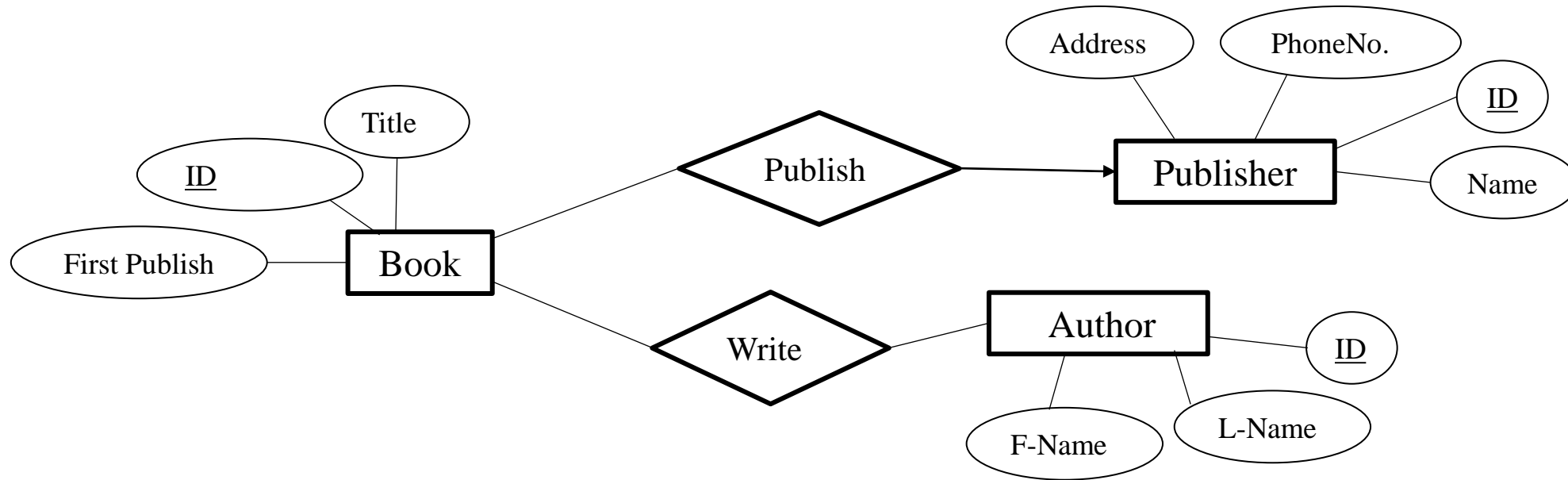
Data Representation

Two common data representation models:

- The **entity-relationship (ER) model** (conceptual Model) : entities with attributes and the relationships between them.
- The **Relational model** (Logical Model) : tables with fields that can refer to other tables.
 - Obtained by converting an ER model to relations/tables

We also need to declare/define the schema of the relations/tables, store data in them and manipulate (work with) the data

Data Representation Example



Books (BID, Title, First Publish)

Publishers (PubID, Name, PhoneNo, Address)

Authors (WID, F-Name, L-Name)

Publishing (BID, PubID)

Writing (BID , WID)

DDL & DML

Data Definition Language (DDL)

Used to declare the schema of relations and create new tables.

Data Manipulation Language (DML)

Used to;

- Insert, delete and update rows in existing tables
- Query the database.

Create Database & Tables

First, create a new database

```
CREATE DATABASE `COMP353`;
```

Use the database

```
USE comp353;
```

Creating books table

```
CREATE TABLE Books (  
    book_id INT,  
    book_title VARCHAR(30),  
    publish_date DATE );
```

Inserting Rows

Tables are initially empty. Use **INSERT** statement to add rows

```
INSERT INTO Books (book_id, book_title, publish_date) VALUES (1, 'book1', '2020-01-01');
```

```
INSERT INTO Books VALUES (1, 'book2', '1990-01-01');
```

- Values appear in same order as table's attributes

	book_id	book_title	publish_date
▶	1	book1	2020-01-01
	1	book2	1990-01-01

Problem: We can add multiple books with the same book id!

➤ To force the database not to allow two rows with same account ID;

Need to change the column id to primary key (**Primary key constraint**)!

Inserting Rows

```
ALTER TABLE Books ADD PRIMARY KEY (book_id);
```

It returns an error! Why??

The Solution : We have to delete rows or update the ids first and then run the above command.

```
DELETE FROM Books;
```

```
ALTER TABLE Books ADD PRIMARY KEY (book_id);
```

Creating other Tables

Publishers (PubID, Name, PhoneNo, Address)

```
CREATE TABLE Publishers (  
    publisher_id INT PRIMARY KEY,  
    publisher_name VARCHAR(30),  
    publisher_address VARCHAR(255),  
    publisher_phone_number VARCHAR(12) );
```

Authors (WID, F-Name, L-Name)

```
CREATE TABLE Writers (  
    writer_id INT PRIMARY KEY,  
    writer_first_name VARCHAR(30),  
    writer_last_name VARCHAR(255) );
```

The CREATE TABLE syntax allows integrity constraints to be specified.

What happen if we execute these commands?

```
INSERT INTO writers VALUES (1, null, 'ln1');
```

```
INSERT INTO writers VALUES(null, 'fn2', 'ln2');
```

Creating other Tables

Publishing (BID, PubID)

```
CREATE TABLE Publisher_Books(  
    publisher_id INT,  
    book_id INT PRIMARY KEY ,  
    FOREIGN KEY (publisher_id) REFERENCES Publishers(publisher_id),  
    FOREIGN KEY (book_id) REFERENCES Books(book_id));
```

Writing (BID , WID)

```
CREATE TABLE Writers_Books(  
    book_id INT,  
    writer_id INT,  
    PRIMARY KEY (book_id, writer_id),  
    FOREIGN KEY (book_id) REFERENCES Books(book_id),  
    FOREIGN KEY (writer_id) REFERENCES Writers(writer_id));
```

Altering Relation Schemas

- Add birth date column to writers table

```
ALTER TABLE writers ADD writer_birth_date DATE;
```

- Add second address to publishers table

```
ALTER TABLE publishers ADD publisher_address_2 VARCHAR(255);
```

- Change publisher_phone_number to publisher_phone

```
ALTER TABLE publishers CHANGE publish_phone_number publihser_phone VARCHAR(20);
```

Altering Relation Schemas

Change the writers table so that it stores first name and last name in one column using the following steps;

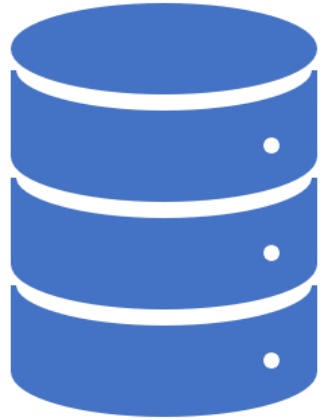
1. Update first name by concatenating last name to it
2. Rename first name column to name
3. Delete last name column.

```
Update writers SET writer_first_name = CONCAT(writer_first_name, " ", writer_last_name);
```

```
ALTER TABLE writers CHANGE writer_first_name writer_name VARCHAR(30);
```

```
ALTER TABLE writers DROP COLUMN writer_last_name;
```

WELCOME!



COMP 353 – Databases

Tutorial 5

Summer 2021

Sections CC, CD

Concordia University

Computer Science & Software Engineering

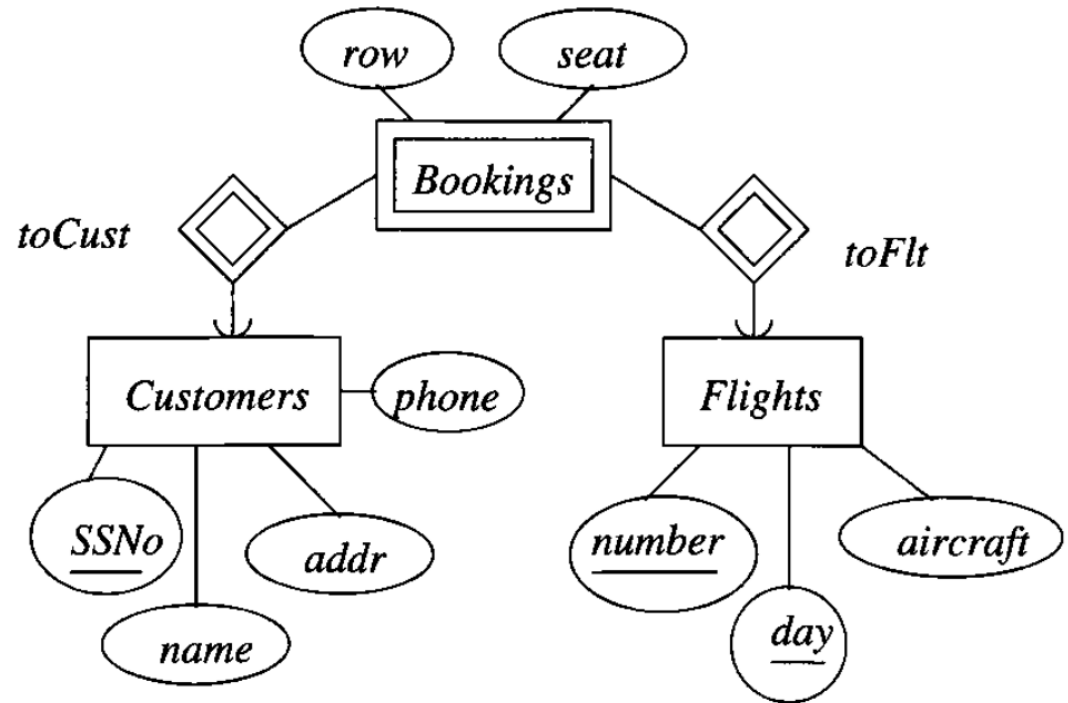
Today's Plan

From E /R Diagrams to Relational Designs/Models

- ❖ Converting Weak Entity Sets
- ❖ Converting isa-hierarchy (of entity sets)
 1. Straight E/R-Style Conversion
 2. OO Approach
 3. Nulls Approach

E/R Diagram Conversions (Weak Entity Sets)

Convert this E/R diagram to relational database (Textbook, Ex. 4.5.1):



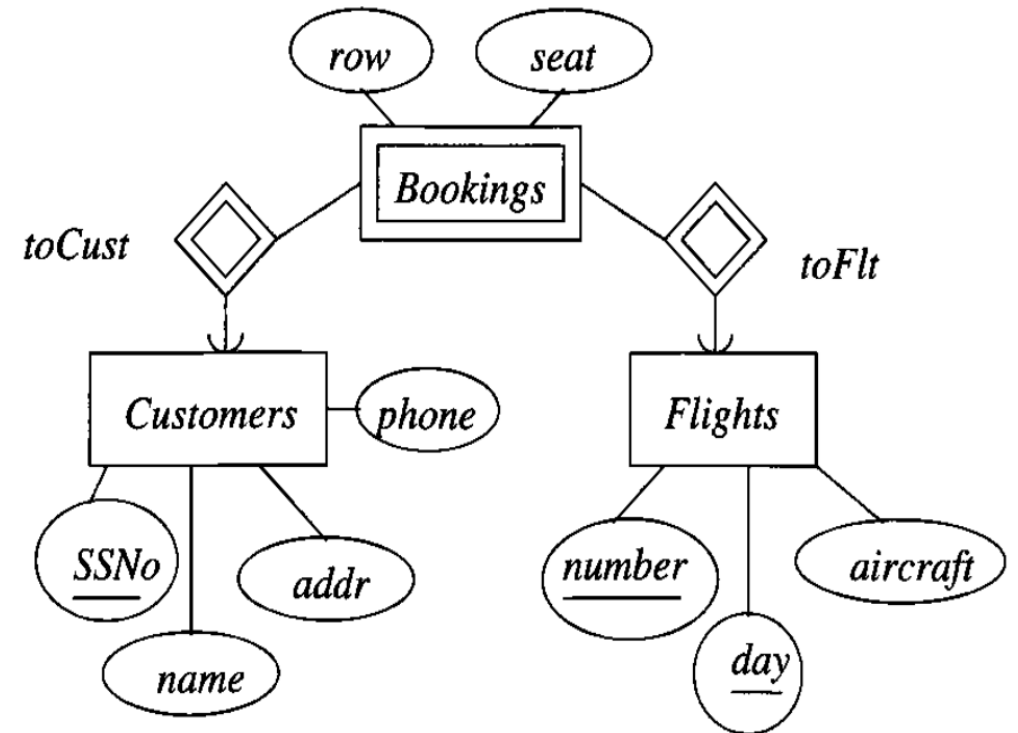
E/R Conversion > Weak Entity Sets

Relational Database:

Customers(SSNo, name, addr, phone)

Flights(number, day, aircraft)

Bookings(SSNo, FltNumber, day, row, seat)

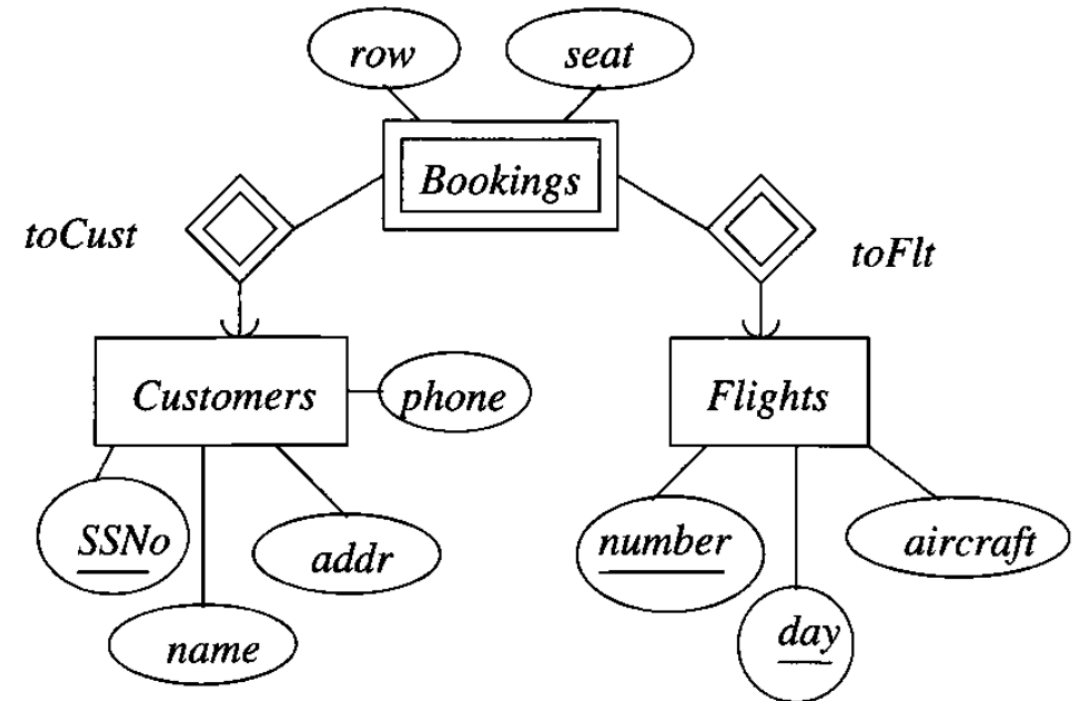


What are the Candidate Keys for Bookings?

- {SSNo, FltNumber, day}
- {FltNumber, day, row, seat}

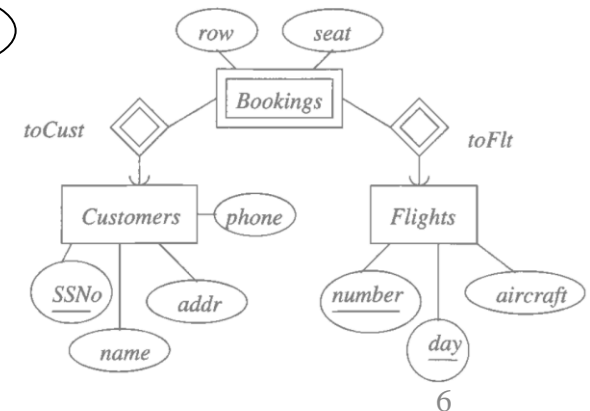
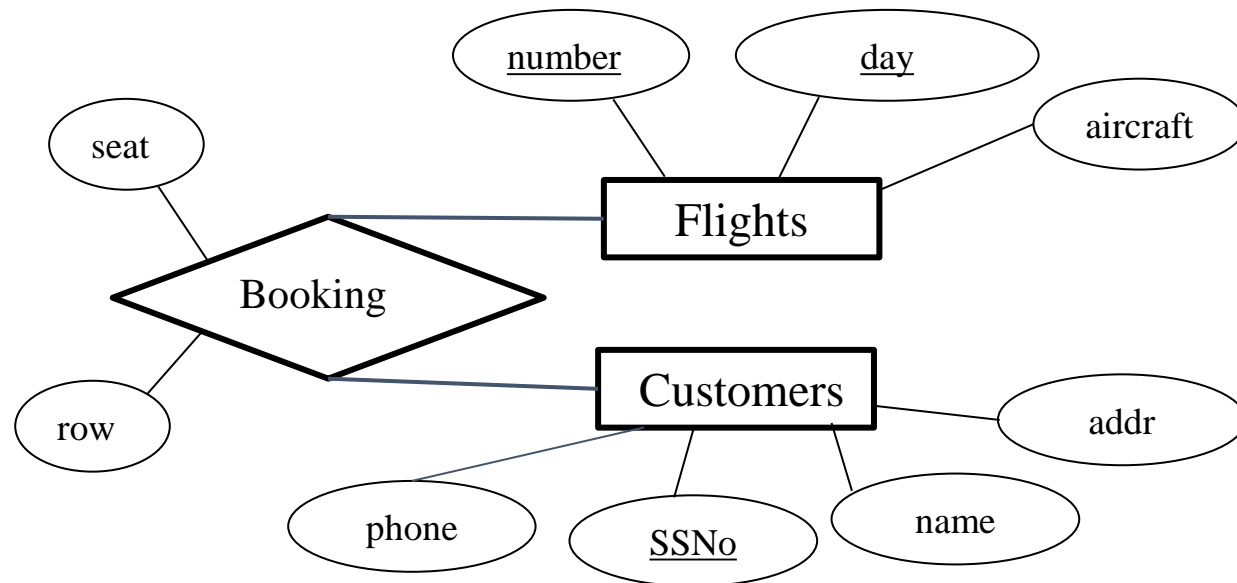
E/R Diagram Conversion

Can we describe the weak entity set Bookings using a different approach?



E/R Diagram Conversion

Convert the following diagram to a relational database:



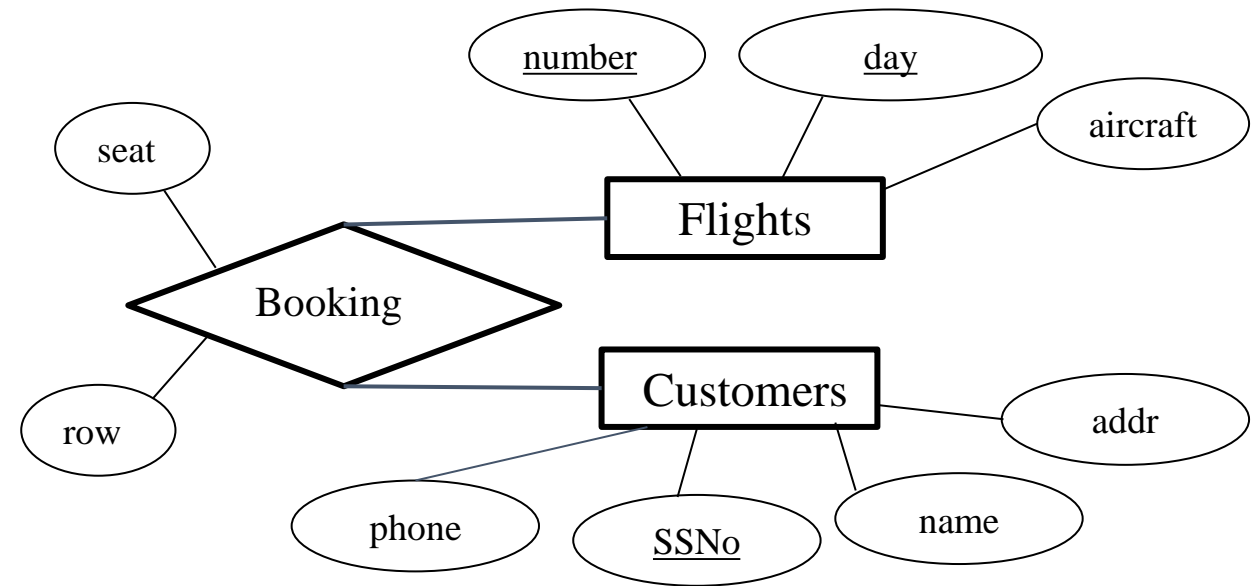
E/R Diagram Conversion

Convert the following diagram to a relational database:

Customers(SSNo, name, addr, phone)

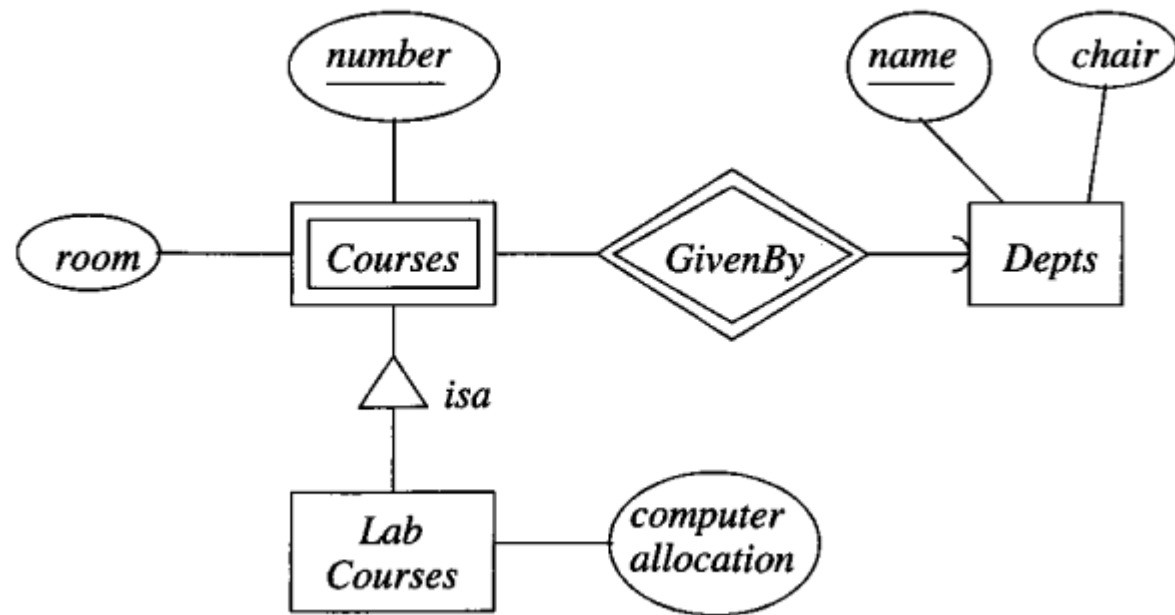
Flights(number, day, aircraft)

Bookings(SSNo, FltNumber, day, row, seat)



E/R Diagram Conversion

Convert the following diagram to a relational database:



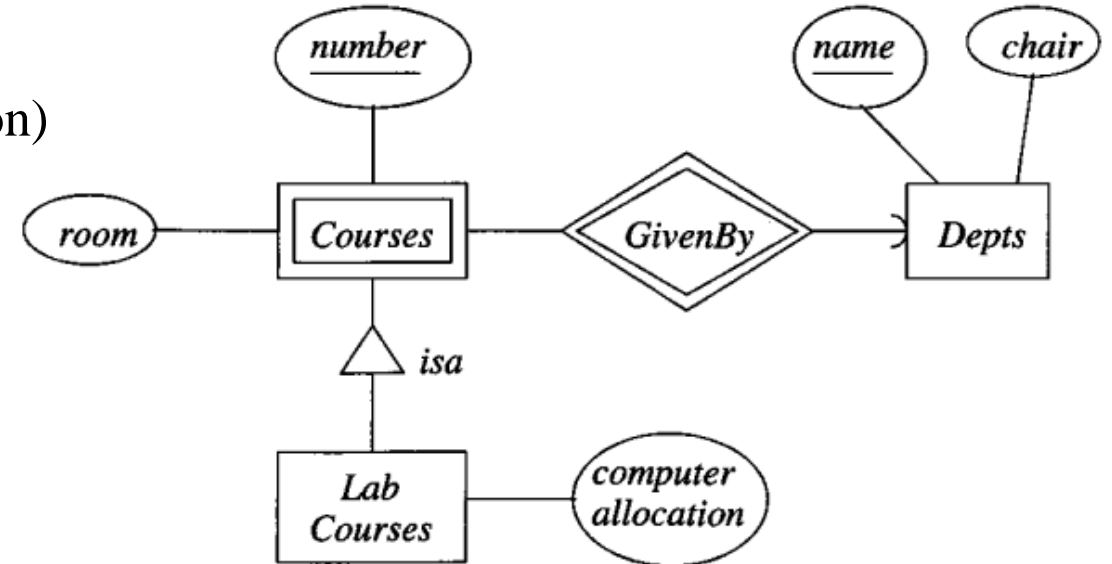
E/R Diagram Conversion

1. Isa-hierarchy using the straight E/R-Style

Depts (name, chair)

Courses (number, deptName, room)

LabCourses (number, deptName, computerAllocation)



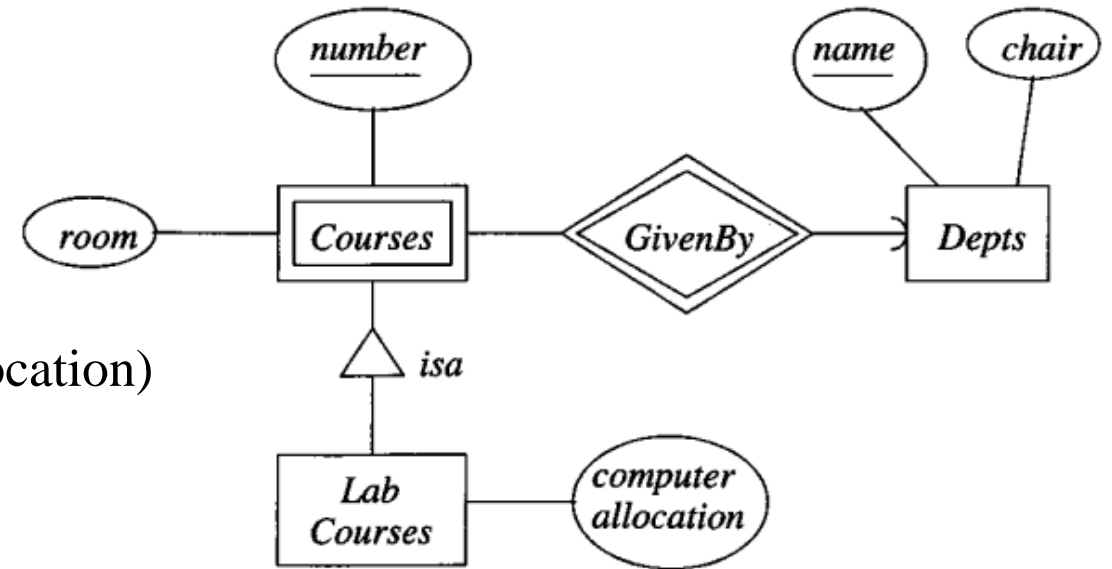
E/R Diagram Conversion

2. Isa-hierarchy using the OO Approach

Depts (name, chair)

Courses (number, deptName, room)

LabCourses (number, deptName, room, computerAllocation)

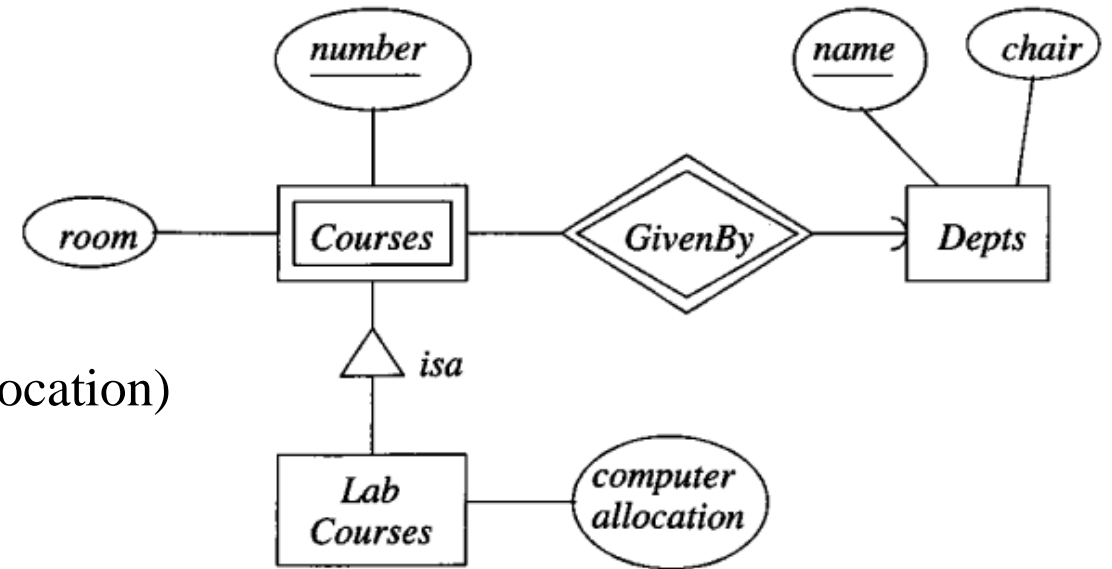


E/R Diagram Conversion

3. Isa-hierarchy using the NULLs Approach:

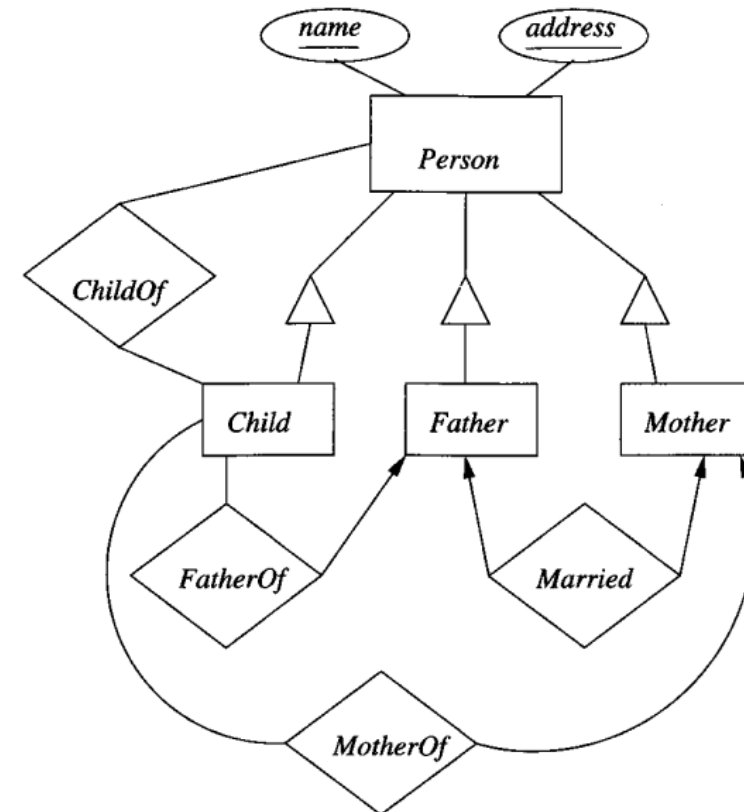
Depts (name, chair)

Lab-Courses (number, deptName, room, computerAllocation)



E/R Conversion

Convert the following diagram to a relational database:



E/R Diagram Conversion

1. Isa conversion using E/R –Style

Person (name, address)

Father (name, address)

Mother (name, address)

Child (name, address)

Married (Wname, Waddress, Hname, Haddress) **PK?**

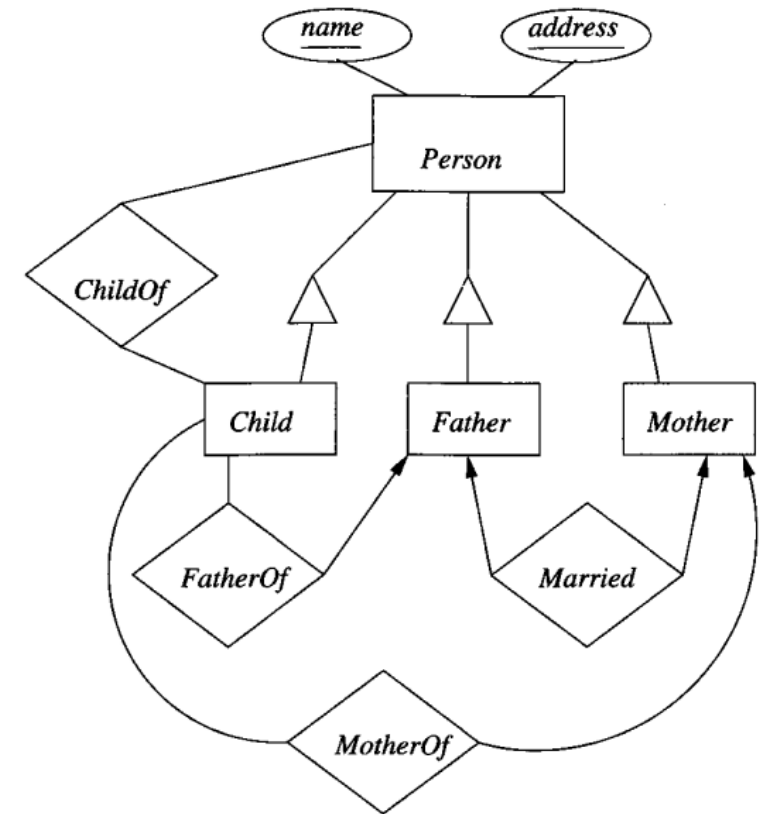
Either {Wname, Waddress} OR {Hname, Haddress}

FatherOf (Cname, Caddress, Fname, Faddress)

PK {Cname, Caddress}

MotherOf (Cname, Caddress, Mname, Maddress)

ChildOf (Cname, Caddress, Pname, Paddress) **PK?**



E/R Diagram Conversion

2. Isa conversion using OO Approach

Person (name, address)

PersonFather (name, address)

PersonMother (name, address)

PersonChild (name, address)

PersonMotherChild (name, address)

PersonFatherChild (name, address)

PersonMotherFather (name, address) Dose it mean??

PersonMotherFatherChild (name, address) Dose it mean??

Add 4 tables for the 4 diamonds:

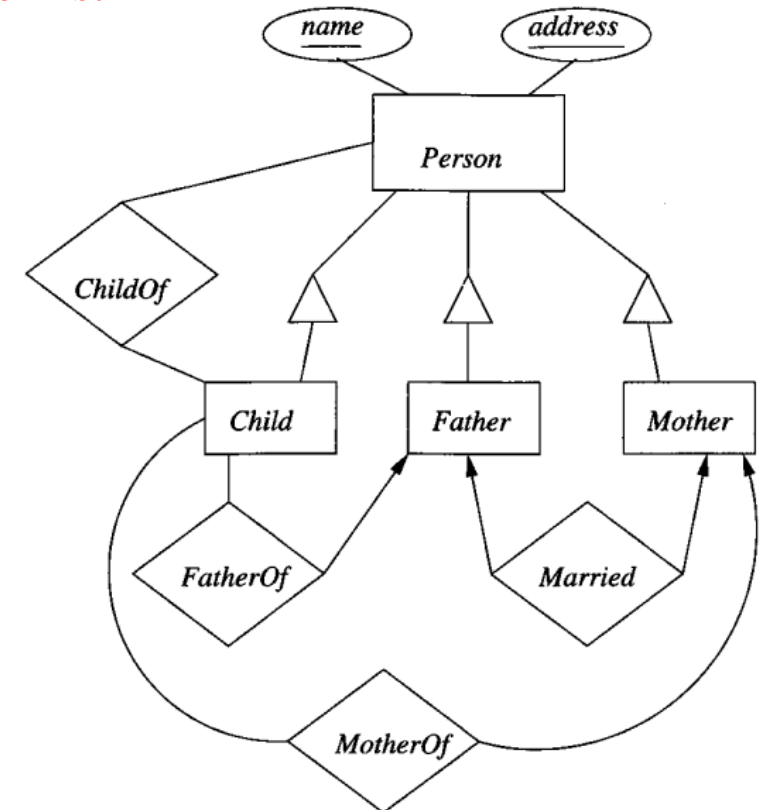
ChildOf(personName,personAddress,childName,childAddress)

FatherOf(childName,childAddress,fatherName,fatherAddress)

MotherOf(childName,childAddress,motherName,motherAddress)

Married(husbandName,husbandAddress,wifeName,wifeAddress)

Question: Can we combine the first four tables??



E/R Diagram Conversion

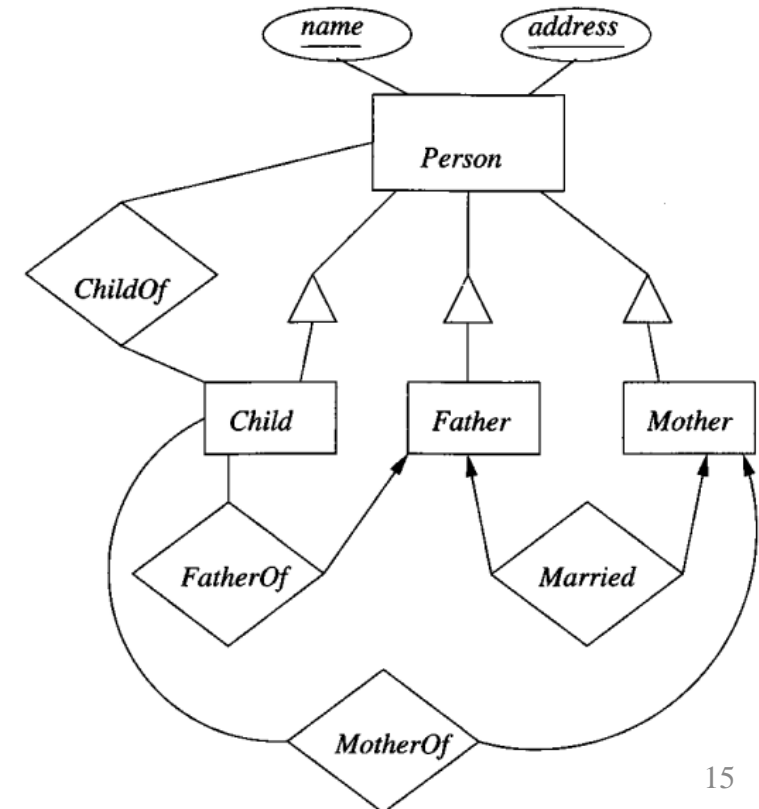
2. Isa conversion using OO Approach

- ❖ Since in OO each entity can belong to one class only, then the m:1 relationships *MotherOf* and *FatherOf* can be added as attributes to the relations:

PersonChild (name, address, Fname, Faddress, Mname, Maddress)

PersonMotherChild (name, address, Fname, Faddress, Mname, Maddress)

PersonFatherChild (name, address, Fname, Faddress, Mname, Maddress)



E/R Diagram Conversion

2. Isa conversion using OO Approach

- ❖ Also, *Married* can be added as attributes to 4 tables; PersonFather, PersonMother, PersonMotherChild, PersonFatherChild.
- ❖ Should add spouse name and address to these tables (Sname, Saddress)!

PersonFather (name, address, Sname, Saddress)

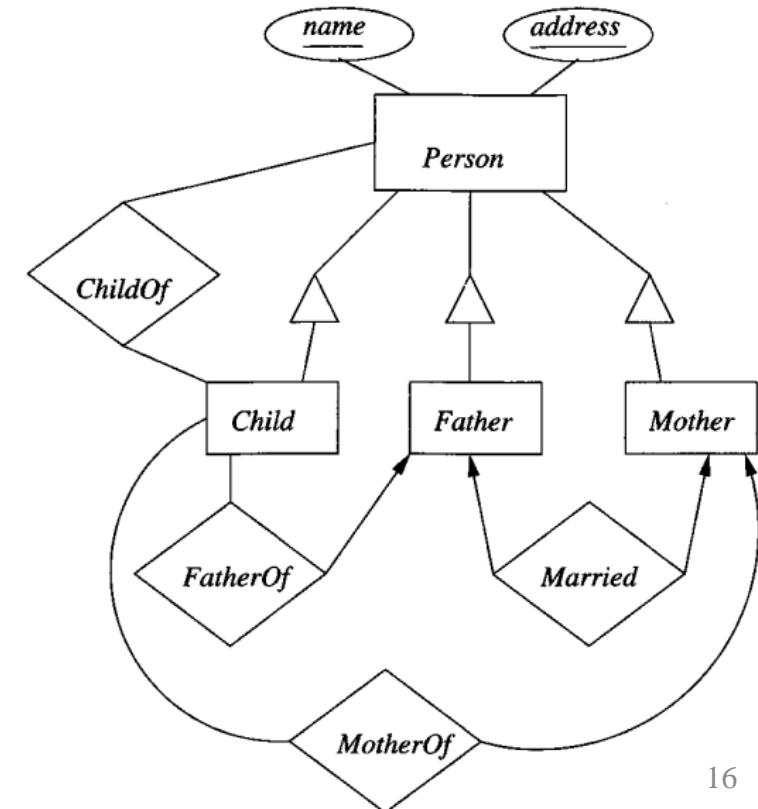
PersonMother (name, address, Sname, Saddress)

PersonMotherChild (name, address, Fname, Faddress, Mname, Maddress, Sname, Saddress)

PersonFatherChild (name, address, Fname, Faddress, Mname, Maddress, Sname, Saddress)

- ❖ *ChildOf*

ChildOf (Cname, Caddress, Pname, Paddress)



E/R Diagram Conversion

3. Isa conversion using Nulls Approach

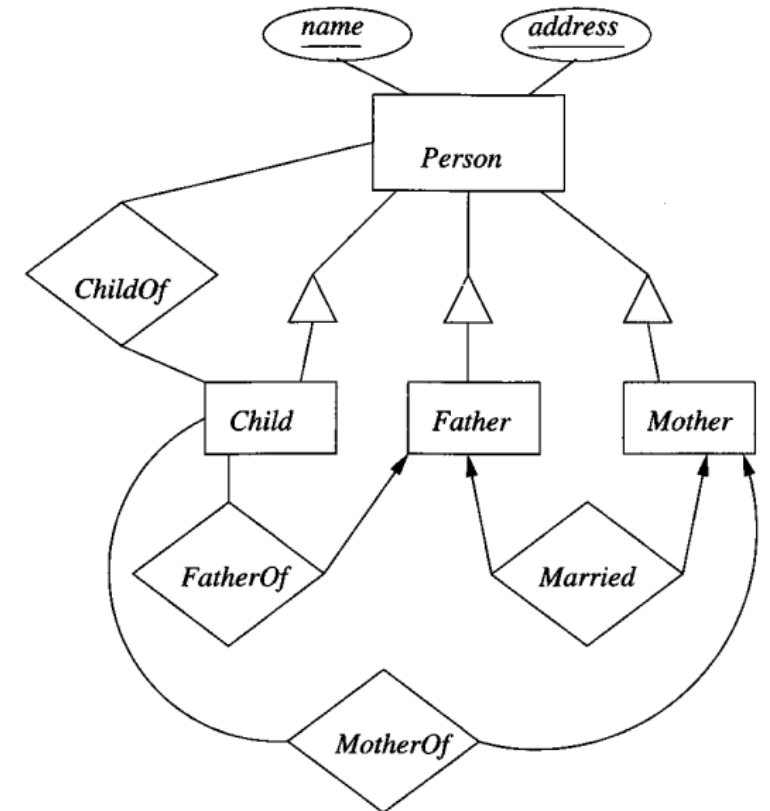
Person (name, address)

Married (Wname, Waddress, Hname, Haddress)

FatherOf (Cname, Caddress, Fname, Faddress)

MotherOf (Cname, Caddress, Mname, Maddress)

ChildOf (Cname, Caddress, Pname, Paddress)



Welcome!



COMP 353 – Databases

Tutorial 6

Summer 2021

Sections CC, CD

Concordia University

Computer Science & Software Engineering

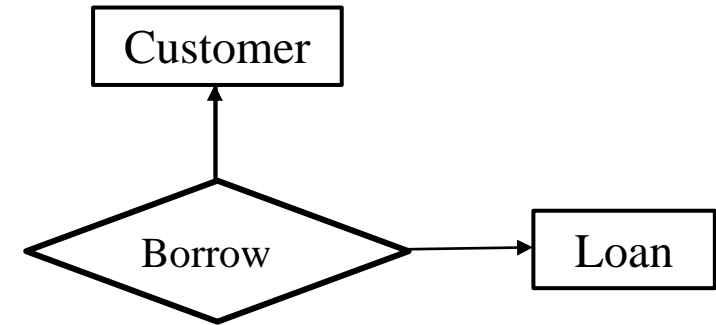
Today's Plan

- ❖ Cardinality of Relationship Sets
- ❖ Multiway Relationships
- ❖ Keys
- ❖ An Introduction to Normalization

Example

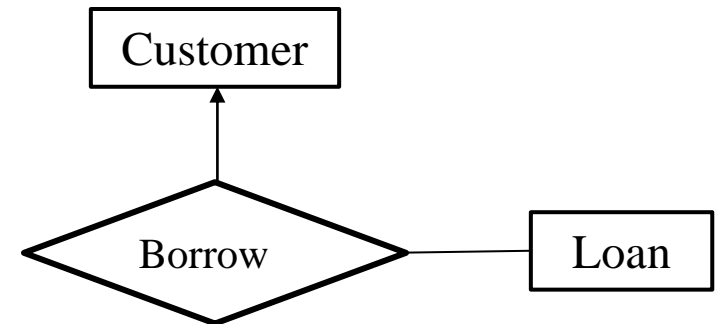
One-to-one relationship

- Each customer can have at most one loan
- Customers can't share loans (e.g. with spouse or business partner)



One-to-many relationship

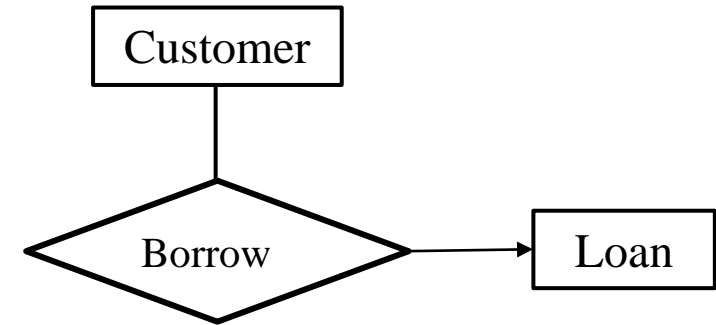
- A customer can have multiple loans
- Customers still can't share loans



Example

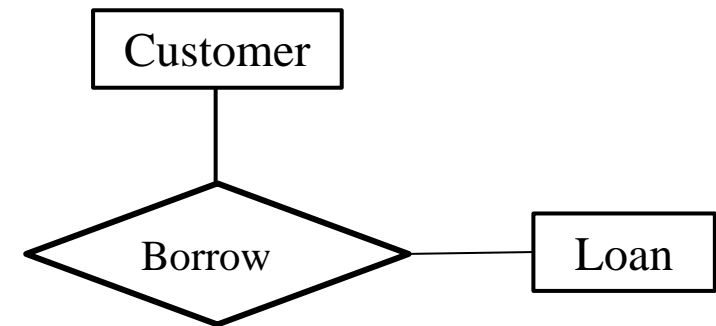
Many-to-one relationship

- Each customer can have AT MOST one loan
- Customers can share loans



Many-to-many relationship

- A customer can have multiple loans
- Customers can share loans



The Best Cardinality for Borrow!

Which mapping cardinality is most appropriate for a given relationship?

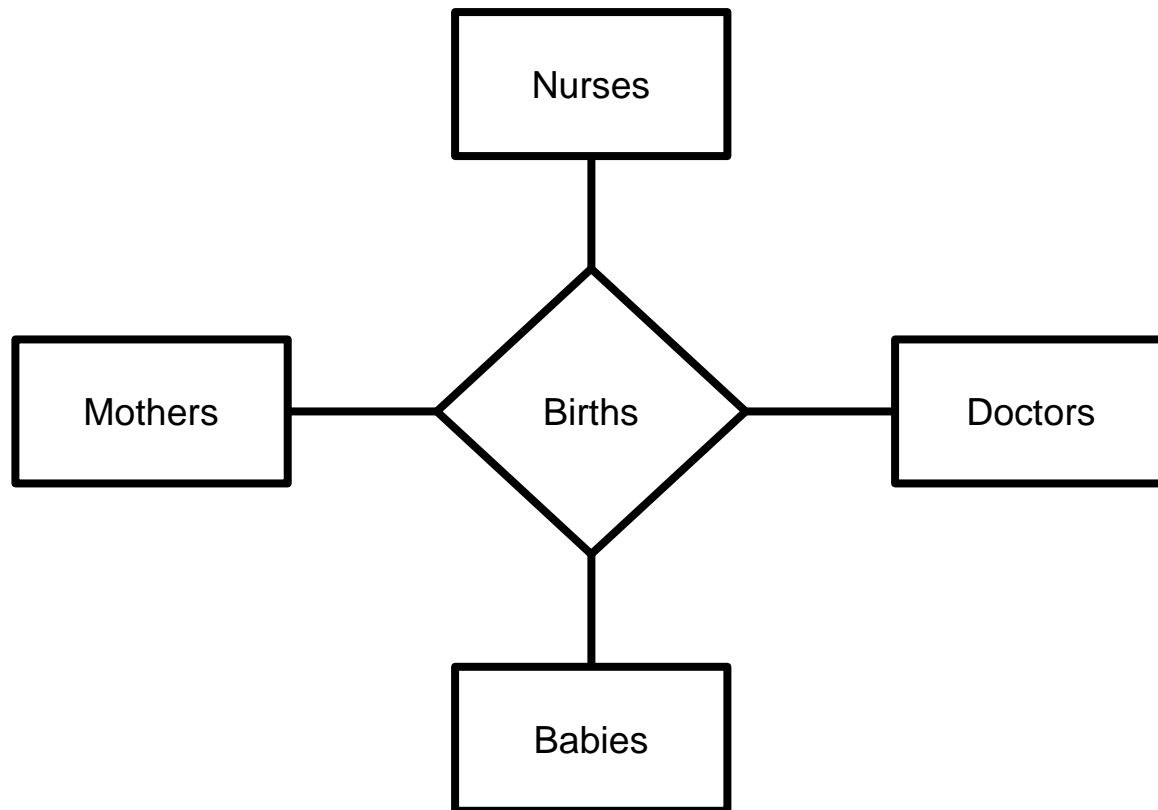
- The answer depends on what we are trying to model

Could just use many-to-many relationships everywhere, but may not reflect the reality we want to model

The more reasonable and realistic choice for the relationship **Borrow** would be **many-to-many**

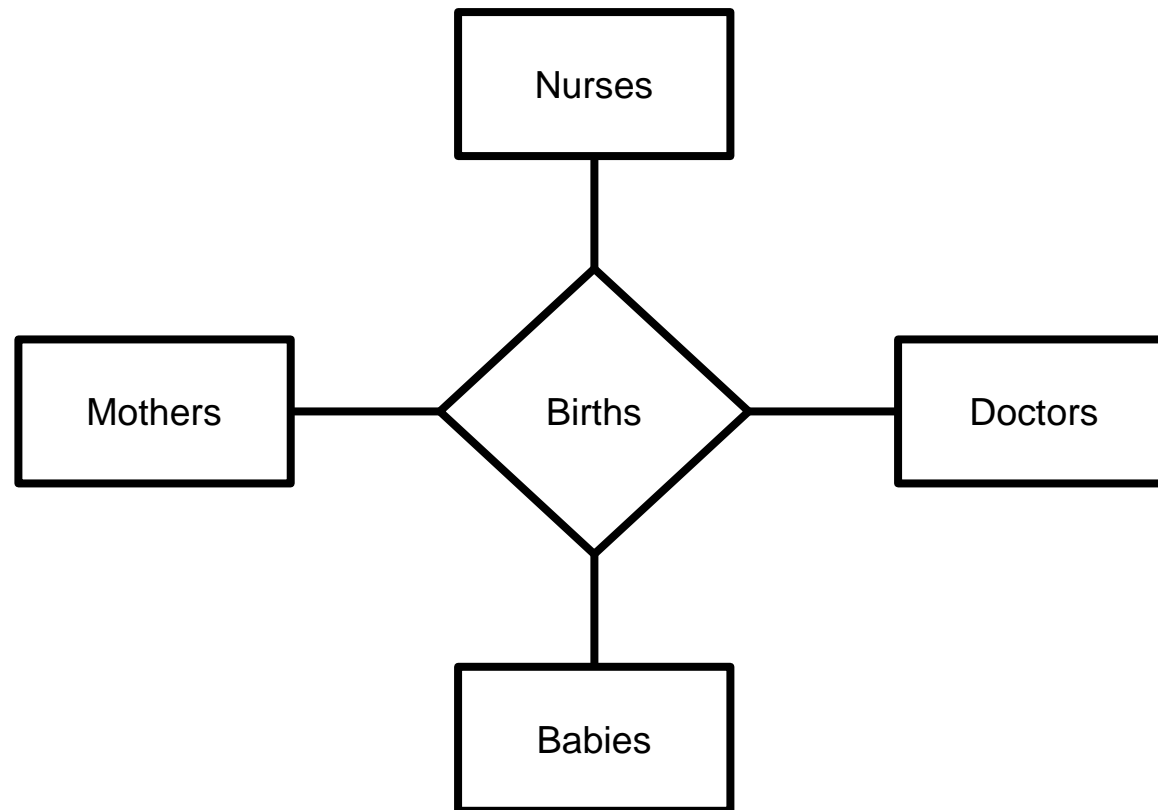
Multiway Relationships

Exercise: Describe the information modeled/represented by this E/R diagram.



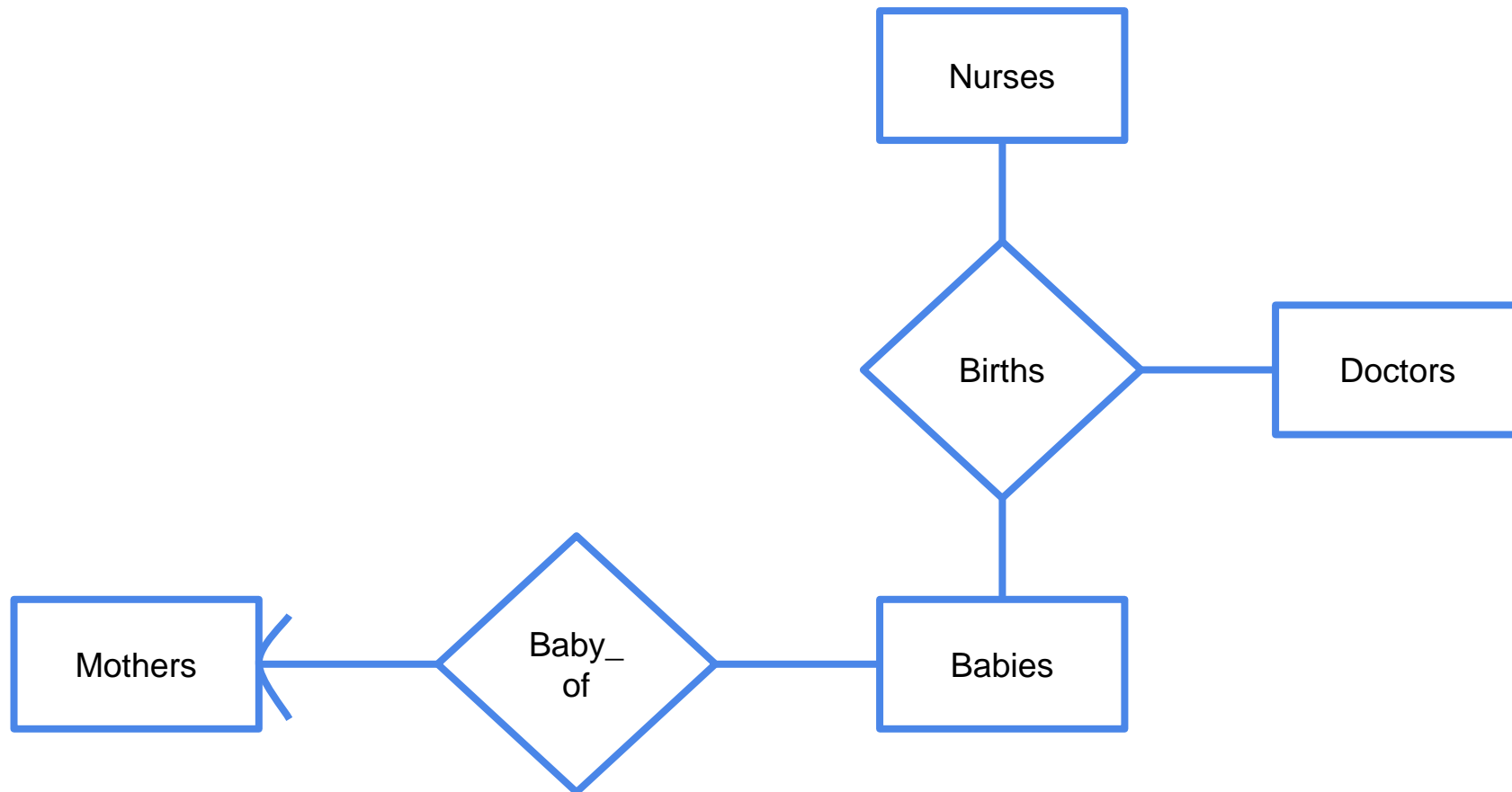
Exercise

- How to modify the diagram to reflect the fact that each baby has one mother?



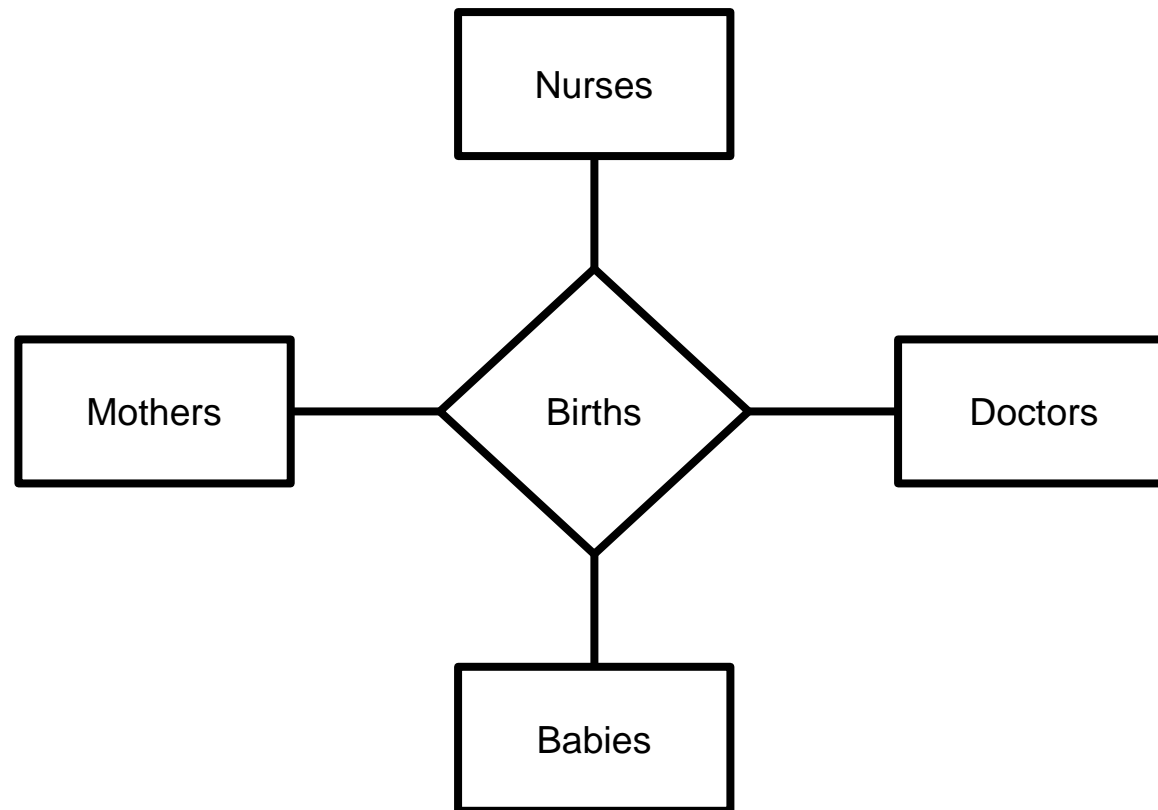
Exercise

- How can we modify the diagram to reflect the fact that each baby has only one mother?



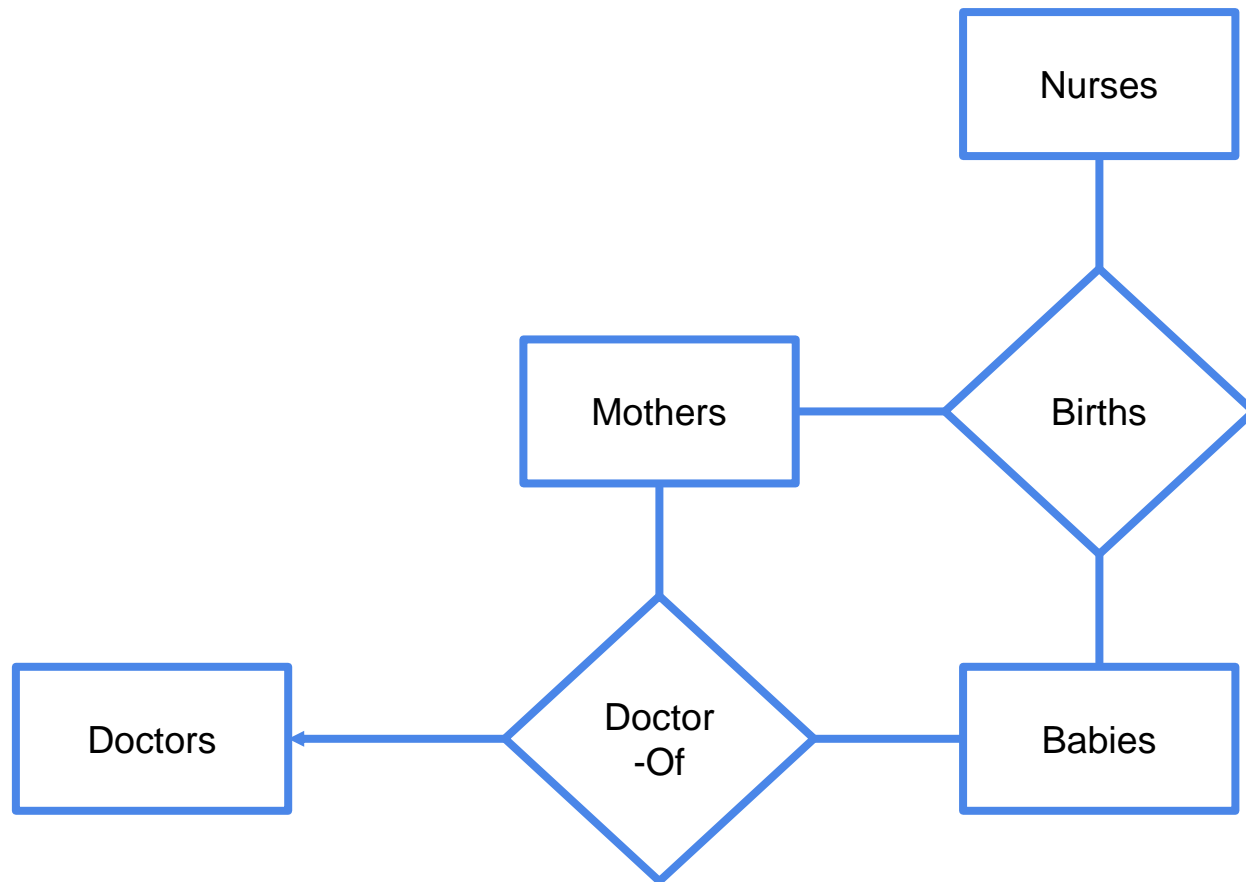
Exercise

- Change the diagram such that each mother-child pair has only one doctor.



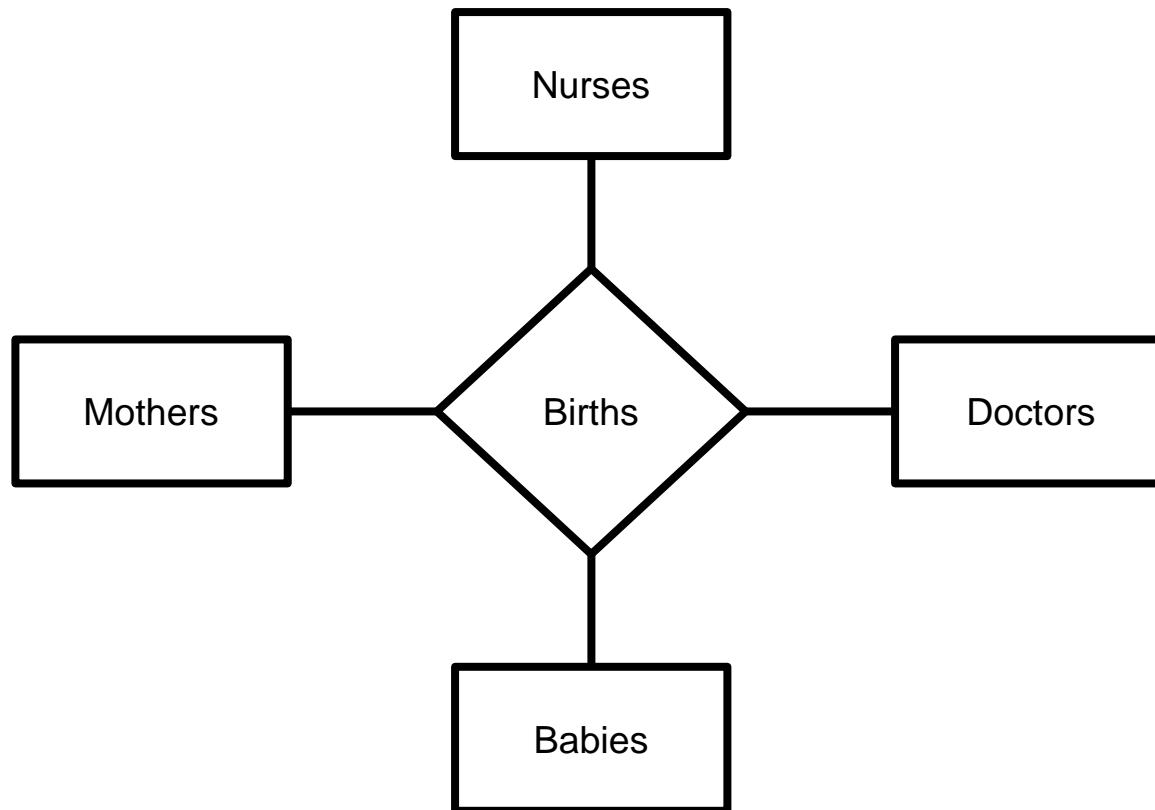
Exercise

- Change the diagram such that each mother-child pair has only one doctor.



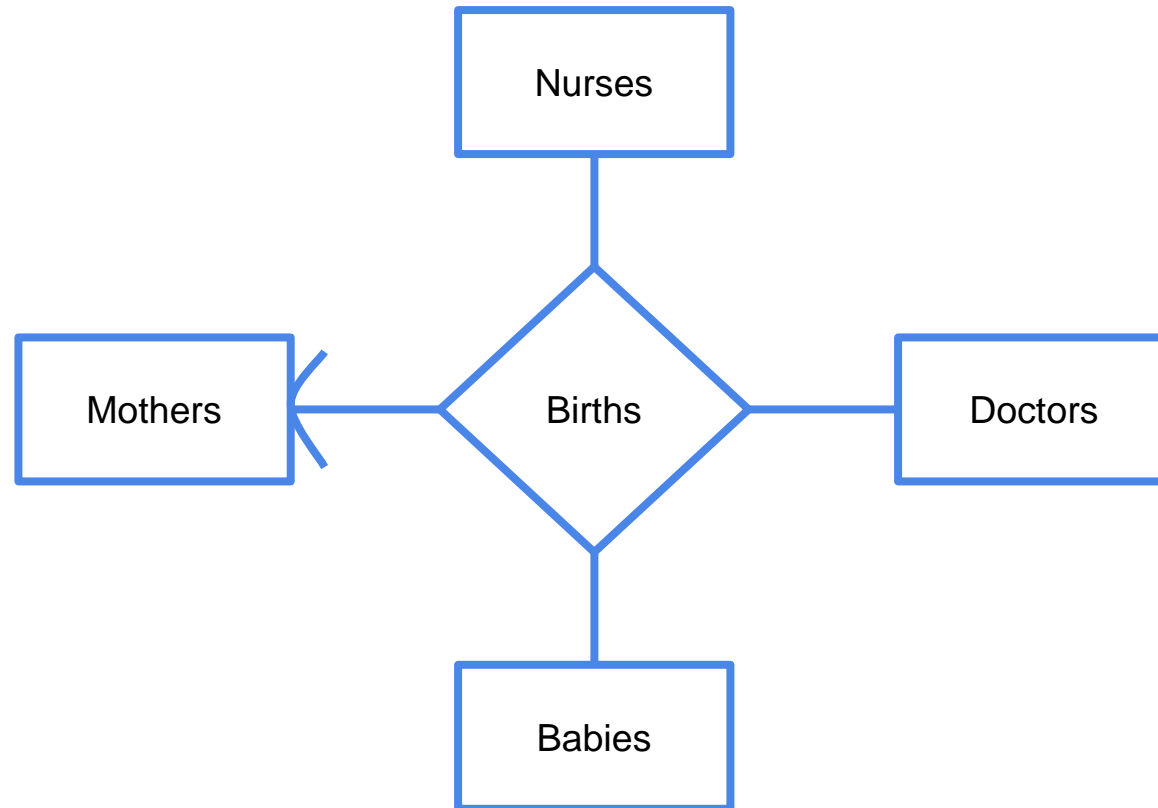
Exercise

- Change the diagram such that for each combination of doctor, baby and nurse, there exists a unique mother.



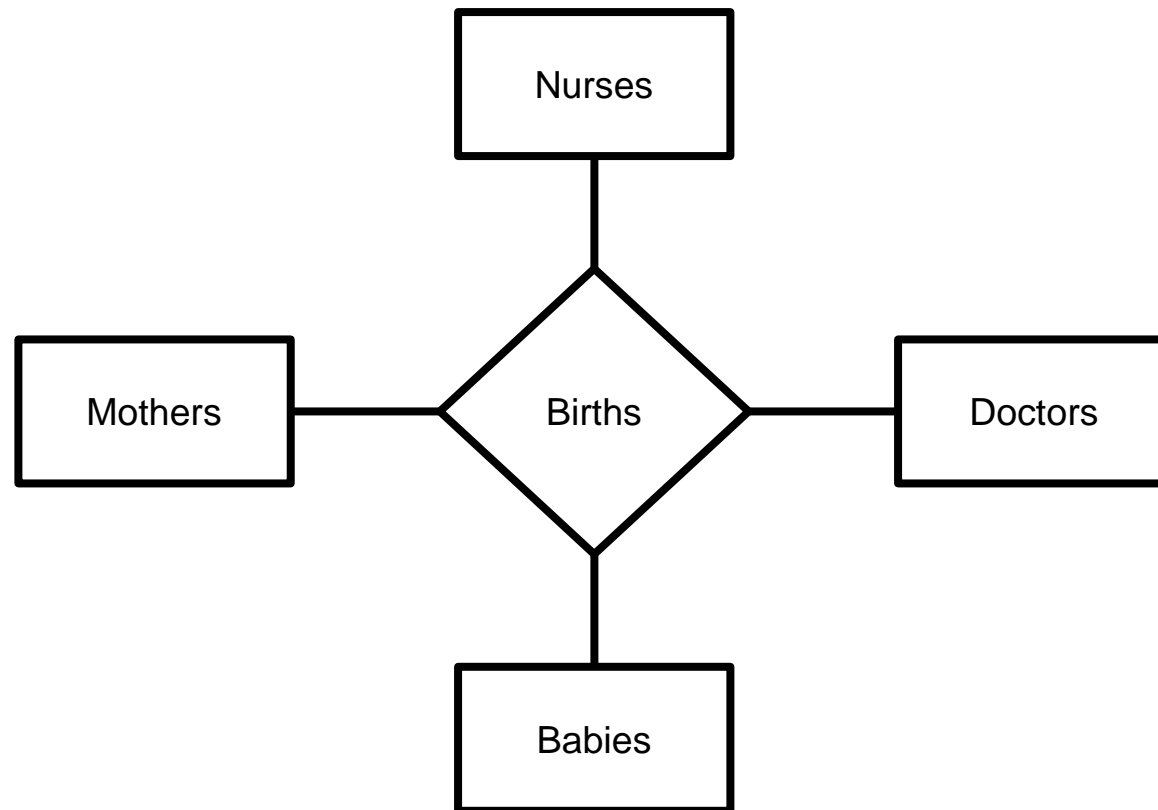
Exercise

- Change the diagram such that for any combination of doctor, baby and nurse, there exists a unique mother.



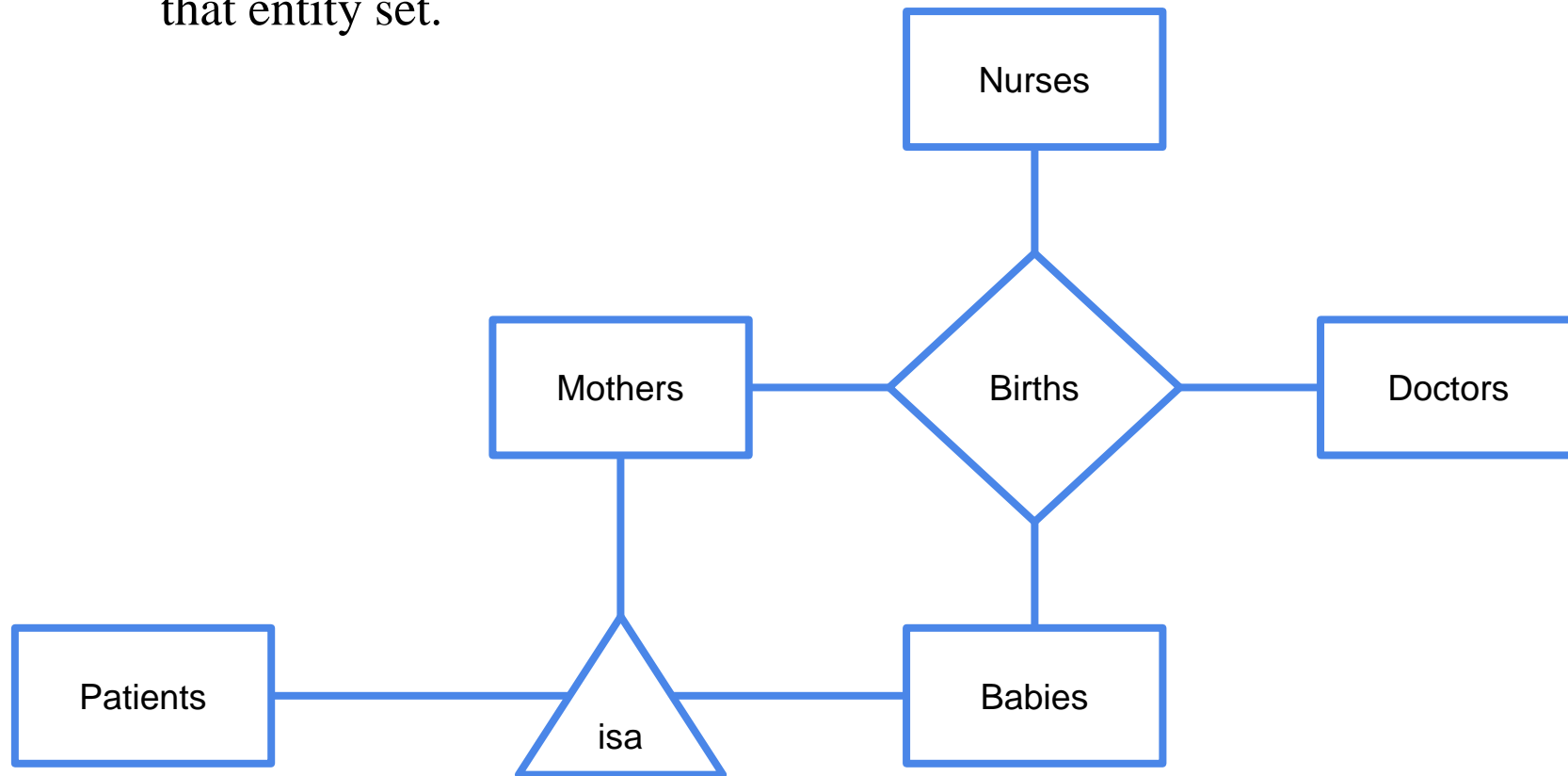
Exercise

- Add a list of patients to the diagram such that both babies and mothers are the children of that entity set.



Exercise

- Add a list of patients to the diagram such that the babies and the mothers are the children of that entity set.



Keys

A **Superkey** of a table T is a subset of its attributes that uniquely identifies tuples in T.

- If a set of attributes $K \subseteq R$ is a superkey, so is any superset S of K (where $S \subseteq R$)

Example:

Customers (cust-id, name, city, SIN)

Loans (loan-id, amount)

Borrowers (cust-id, loan-id, access-date)

How many superkeys does Customers have?

Keys

A minimal superkey is called a **Candidate key**. That is, a candidate key is:

- A superkey for which no proper subset is a superkey

Thus, for Customers, {cust-id} and {SIN} are two candidate keys

Primary Key: A relation might have several candidate keys. In that case, ONE candidate key is chosen as the P.K.

Foreign Key: A relation schema can include attributes which are the primary keys of other schemas

Example: The F.Ks of Borrowers in the following DB?

Customers(cust-id, name, city, SIN)

Loans (loan-id, amount)

Borrowers (cust-id,loan-id, access-date)

Introduction to Normalization

What is this table about?

- Employees?
- Departments?

Redundancy

EMP_DEPT

Ename	<u>Ssn</u>	Bdate	Address	Dnumber	Dname	Dmgr_ssn
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975 FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5	Research	333445555
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1	Headquarters	888665555

Example

Some possible difficulties/ANOMALIES in NOT normalized tables

- Add a new employee with unknown department
- Add a new department
- Change the department name/manager
- Delete James E. Borg

EMP_DEPT

Ename	<u>Ssn</u>	Bdate	Address	Dnumber	Dname	Dmgr_ssn
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975 FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5	Research	333445555
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1	Headquarters	888665555

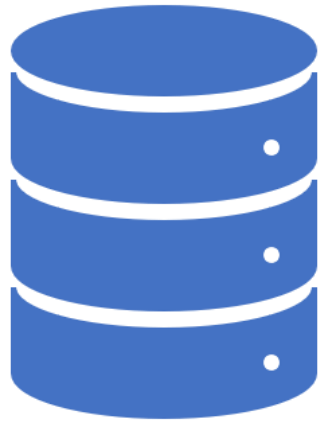
Future Plan

- ✓ Normal forms indicate certain degrees of design quality
- ✓ We are going to build up to a set of “tests” (to detect certain violation of NFs and algorithms to fix the problems)
- ✓ To do so, we use Functional Dependencies to detect and fix NF of relations.

EMP_DEPT

Ename	<u>Ssn</u>	Bdate	Address	Dnumber	Dname	Dmgr_ssn
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975 FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	56	5	Research	333445555
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1	Headquarters	888665555

Welcome!



Comp 353 – Databases

Tutorial 7

Summer 2021

Sections CC,CD

Concordia University - Department of Computer Science & Software
Engineering

Today's Plan

- ❖ Keys
- ❖ Functional Dependency
- ❖ Functional Dependency & Superkeys

Superkey

- Why we need a key?
- Let's assume we have a table of Students:
- Can we use a combination of columns to be a superkey?
 - {Department, Course} ?
 - {Name, Mark} ?
 - {Name, Mark, Department, Course} ?

Name	Mark	Department	Course
Mike	75	CS	C1
John	65	INDS	C2
Ann	85	INDS	C1
Mike	75	CS	C3
Nancy	90	IT	C4

Superkey

- Superkey is an attribute or a set of attributes which can uniquely identify each record of a table.
 - Id is a Superkey, for example
 - What about {Id, Name} ?

Id	Name	Mark	Department	Course
1	Mike	75	CS	C1
2	John	65	INDS	C2
3	Ann	85	INDS	C1
4	Mike	75	CS	C3
5	Nancy	90	IT	C4

Maximum Number of Superkeys

- Let's calculate the maximum number of possibilities for superkeys in this table.

Id	Name	Mark	Department	Course
1	Mike	75	CS	C1
2	John	65	INDS	C2
3	Ann	85	INDS	C1
4	Mike	75	CS	C3
5	Nancy	90	IT	C4

Candidate Key

Question?

What are the superkeys of relation= $\{A, B, C\}$, if A is a candidate key?

- $\{A\}$ $\{A, B\}$ $\{A, C\}$ $\{A, B, C\}$

Functional Dependency

- Functional Dependency?
 - What does $X \rightarrow Y$ mean?
 - X determines Y
 - Y is functionally dependent on X
 - X is Determinant and Y is Dependent

X	Y
1	1
2	1
3	15
4	14
5	6

X	Y
1	1
2	1
3	15
4	14
2	5

X	Y
1	1
2	1
3	15
4	14
2	1

Functional Dependency

- Find which FDs hold on this relation instance?
 - $\text{Id} \rightarrow \text{Name}$?
 - $\text{Id, Name} \rightarrow \text{Course}$?
 - $\text{Name, Mark} \rightarrow \text{Course}$?
 - $\text{Name, Mark} \rightarrow \text{Department, Course}$?

Id	Name	Mark	Department	Course
1	Mike	70	CS	Course1
2	Rose	60	INDS	Course1
3	Mike	70	CS	Course2
4	Rose	60	INDS	Course3
5	Daniel	90	IT	Course3
6	Mary	90	EE	Course2

Functional Dependency

- Find which FDs hold?
 - $\text{Id} \rightarrow \text{Name}$? Yes
 - $\text{Id, Name} \rightarrow \text{Course}$? Yes
 - $\text{Name, Mark} \rightarrow \text{Course}$? No (WHY?)
 - $\text{Name, Mark} \rightarrow \text{Department, Course}$? No (WHY?)

Id	Name	Mark	Department	Course
1	Mike	70	CS	Course1
2	Rose	60	INDS	Course1
3	Mike	70	CS	Course2
4	Rose	60	INDS	Course3
5	Daniel	90	IT	Course3
6	Mary	90	EE	Course2

Trivial Functional Dependency

- An FD $X \rightarrow Y$ is TRIVIAL if its right-hand side is a subset of its left-hand side.

If $Y \subseteq X$, then $X \rightarrow Y$ is trivial

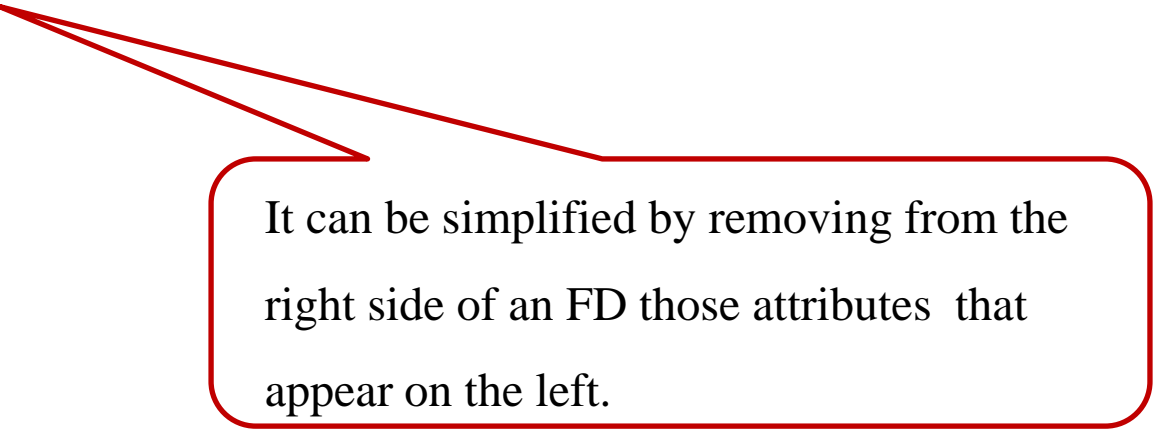
- $(\text{Id}, \text{Name}) \rightarrow \text{Name}$
- $\text{Name} \rightarrow \text{Name}$

It says that “two tuples that agree in both Id and Name, MUST agree in Name.”

Id	Name	Mark	Department	Course
1	Mike	70	CS	Course1
2	Rose	60	INDS	Course1
3	Mike	70	CS	Course2
4	Rose	60	INDS	Course3
5	Daniel	90	IT	Course3
6	Mary	90	EE	Course2

Non-Trivial Functional Dependency

- Are the following FDs trivial?
 - $\text{Id} \rightarrow \text{Name}$
 - $\text{Id, Name} \rightarrow \text{Name, department}$
 - Both FDs are **nontrivial**



It can be simplified by removing from the right side of an FD those attributes that appear on the left.

Armstrong Axiom, Inference Rules

- **Reflexivity:** If $Y \subseteq X$, then $X \rightarrow Y$ (trivial FD)
 - $\text{Ename, Bdate} \rightarrow \text{Bdate}$
 - $\text{Ename, Bdate} \rightarrow \text{Ename}$
 - $\text{Ename, Bdate} \rightarrow \text{Ename, Bdate}$
- **Augmentation:** If $X \rightarrow Y$, then $XZ \rightarrow YZ$, for every $Z \subseteq R$
 - $\text{Ename} \rightarrow \text{Bdate}$ then $(\text{Ename, Address}) \rightarrow \text{Bdate, Address}$
- **Transitivity:** If $X \rightarrow Y$ & $Y \rightarrow Z$ then $X \rightarrow Z$ (for any subsets $X, Y, Z \subseteq R$)
 - $\text{Ename} \rightarrow \text{Dname}$ & $\text{Dname} \rightarrow \text{Manager}$ then $\text{Ename} \rightarrow \text{Manager}$

EMP_DEPT

Ename	<u>Ssn</u>	Bdate	Address	Dnumber	Dname	Dmgr_ssn
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321

Armstrong Axiom, Inference Rules

- **Union:** If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$
 - $SSN \rightarrow Bdate$ & $SSN \rightarrow Address$ then $SSN \rightarrow Bdate, Address$
- **Decomposition or Splitting:** If $X \rightarrow YZ$, then $X \rightarrow Z$ & $X \rightarrow Y$
 - $SSN \rightarrow Bdate, Address$ then $SSN \rightarrow Bdate$ & $SSN \rightarrow Address$
- **Pseudo-Transitivity:** If $X \rightarrow Y$ and $YZ \rightarrow A$, then $XZ \rightarrow A$
 - $Ename \rightarrow Dname$ & $Dnumber, Dname \rightarrow Manager$ then $Ename, Dnumber \rightarrow Manager$

$XY \rightarrow Z$ **CANNOT** be decomposed to $X \rightarrow Z$ & $Y \rightarrow Z$

EMP_DEPT

Ename	<u>Ssn</u>	Bdate	Address	Dnumber	Dname	Dmgr_ssn
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321

Attribute Closure, Closure Set

Example: Given the relation $R(A, B, C, D, E)$ and Functional Dependencies

$F\{A \rightarrow B, B \rightarrow C, C \rightarrow D, D \rightarrow E\}$, find other hidden FDs?

➤ using the **transitivity** rule:

- $A \rightarrow B$ & $B \rightarrow C$, then $A \rightarrow C$
- $A \rightarrow C$ & $C \rightarrow D$ results in $A \rightarrow D$
- $A \rightarrow D$ & $D \rightarrow E$ we will have $A \rightarrow E$

➤ Now we can use the **union** rule to combine the mentioned attributes as follows:

- $A \rightarrow \mathbf{ABCDE}$
- We can also show that $\mathbf{B} \rightarrow \mathbf{BCDE}$ but B cannot determine A.
- The same for C by showing that $\mathbf{C} \rightarrow \mathbf{CDE}$

Attribute Closure, Closure Set

Question ?

Knowing that $A^+ = \{ A, D, B, C, E \}$, what can we say about the results of following closures?

- AB^+
- AC^+
- AD^+

Attribute Closure, Closure Set

Example: Let us consider the FDs: $F = \{AB \rightarrow C, BC \rightarrow AD, D \rightarrow E, \text{ and } CG \rightarrow B\}$ for relation $R(A, B, C, D, E, G)$.

Does $AB \rightarrow D$ follow from F ?

To check whether an FD: $X \rightarrow Y$ follows from a set of FDs, we first find **the closure of the set of attributes X** , denoted X^+ , which includes every attribute determined by X . If the right-hand side Y is included in X^+ , then $X \rightarrow Y$ follows from F . Otherwise, the FD is not followed by F .

Attribute Closure, Closure Set

Example: Let us consider the FDs $F = \{AB \rightarrow C, BC \rightarrow AD, D \rightarrow E, \text{ and } CG \rightarrow B\}$ for relation $R(A, B, C, D, E, G)$.

Does $AB \rightarrow D$ follow from F ?

We find $\{A, B\}^+$:

Initialization: $X^+ = \{A, B\}^+ = \{A, B\}$

Round I: $X^+ = \{A, B\}$

using $AB \rightarrow C$, $X^+ = \{A, B, C\}$

using $BC \rightarrow AD$, $X^+ = \{A, B, C, D\}$

using $D \rightarrow E$, $X^+ = \{A, B, C, D, E\}$

using $CG \rightarrow B$, no more attributes get into X^+ .

From this point, by rechecking all the FDs the closure is not changed, so the algorithm terminates.

$X^+ = \{A, B, C, D, E\}$

Functional Dependency & Superkeys

If the closure of a set of attributes X includes all attributes of the relation R , then X is a superkey of R .

In our previous example,

➤ $\{A, B\}^+ = \{A, B, C, D, E\}$ does not include attribute G , so AB is not a superkey.

➤ To find a superkey,

Use the trivial FD $G \rightarrow G$ with augmentation to get $\{A, B, G\}^+ = \{A, B, C, D, E, G\}$ resulting in the superkey $\{A, B, G\}$

From Super keys to Candidate keys

To get a candidate key from the superkey $\{A, B, G\}$, we must test if we can remove any attribute from the set WITHOUT changing its power of being a key.

- We eliminate attributes from the superkey one at a time and find the closure of the new set
- If the new set still functionally determines all attributes of the relation, we continue the process by this new set.
- The resulting smallest set(s) is a candidate key.

If we **remove A**: $\{B, G\}^+ = \{B, G\} \leftarrow$ not a full cover, A is needed

If we **remove B**: $\{A, G\}^+ = \{A, G\} \leftarrow$ not a full cover, B is needed

If we **remove F**: $\{A, B\}^+ = \{A, B, C, D, E\} \leftarrow$ not a full cover, F is needed

Thus we know that $\{A, B, G\}$ is truly a candidate key for the relation!

Attribute Closure, Closure Set

- Now it's your time to find the superkeys and the candidate keys for the following relation and functional dependency:
 - $R(A, B, C, D, E)$
 - FD: $\{ A \rightarrow B, D \rightarrow E \}$

Welcome!



Comp 353 – Databases

Tutorial 8

Summer 2021

Sections CC & CD

Concordia University - Department of Computer Science & Software Engineering

Today's Plan

- ❖ Projecting FDs
- ❖ Prime and Non-Prime Attributes
- ❖ Canonical Cover (*Minimal Basis*)

Projecting FDs

Given a set F of FDs for a relation R ,

Question: If we remove some attributes from R to get R_1 , what FDs still follow in R_1 ?

Answer: By computing the projection of FD's F on R_1 , we get all FD's that follow from F and involve only the attributes of R_1 .

Algorithm ;

1. Initialize $F_1 = \{\}$.
2. Construct a set of all subsets of attributes of R_1 called X .
3. Compute X_i^+ for all members of X under F . X_i^+ may consists of attributes that are not in R_1 .
4. Add to F_1 all nontrivial FD's $X \rightarrow A$ such that A is both in X_i^+ and an attributes of R_1 .
5. Now, F_1 is a basis for the FD's that hold in R_1 but may not be a minimal basis.

Boyce-Codd Normal Form (BCNF)

Let R be a relation schema with FD's F . R is in BCNF if every FD $X \rightarrow A$ in F ($X \subseteq R$, and $A \in R$)

- $A \in X$, (meaning it is a trivial FD)

OR

- X is a superkey

To determine whether R with a given set of FD's F is in BCNF

Ignore all trivial FDs

Check whether the LHS X of each of the other FD's is a superkey using the closure: check that $X^+ = R$

Pick every FD $X \rightarrow A$ that violates the BCNF requirement:

1. Decompose R into two relations: XA and $R - A$, and project the FDs onto the new relations
2. If either $R - A$ or XA is not in BCNF decompose it further (recursion)

Boyce-Codd Normal Form (BCNF)

Example 1: $R = \{A, B, C, D, E\}$ and $F = \{A \rightarrow BC, C \rightarrow DE\}$, decompose R to BCNF. First step, find the candidate keys via closures

First step, find the candidate keys via closures:

- $A^+ = \{A, B, C, D, E\}$ $D^+ = \{D\}$
 $B^+ = \{B\}$ $E^+ = \{E\}$ $\{A\}$ is a superkey
 $C^+ = \{C, D, E\}$
- Pick every FD $X \rightarrow A$ that violates the BCNF requirement:
- $A \in X$, (meaning it is a trivial FD)
- OR
- X is a superkey
- We determined that $\{A\}$ is a superkey
- $A \rightarrow BC$: The LHS is a superkey, so no violation here
- $C \rightarrow DE$: This FD is not trivial and $\{C\} \neq \{A\}$, so it is a violation

Finding All Candidate Keys

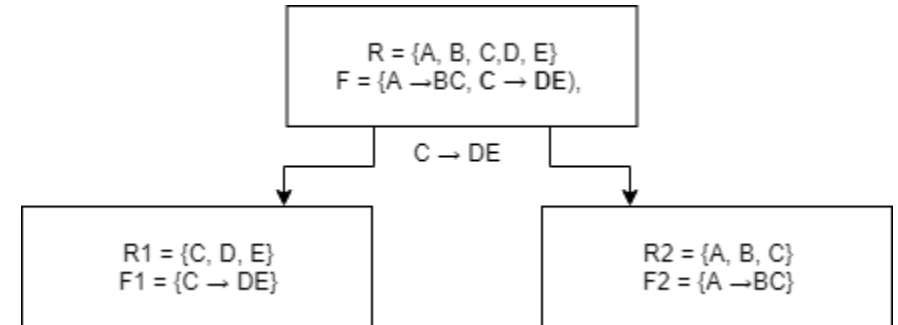
Example 1: ick every FD $X \rightarrow A$ that violates the BCNF requirement:

1. Decompose R into two relations: XA and $R - A$, and project the FDs onto the new relations
2. If either $R-A$ or XA is not in BCNF decompose it further (recursion)

$C \rightarrow DE$ becomes:

$R1 = \{C, D, E\}$ and $F1 = \{C \rightarrow DE\}$

$R2 = \{A, B, C\}$ and $F2 = \{A \rightarrow BC\}$



Candidate Key

Example 1: Find all the candidate keys for the following relation and its set of FDs:

- $R(A, B, C, D, E)$
- FD: $\{ A \rightarrow B, D \rightarrow E \}$

Candidate Key

Example 1: Find all the candidate keys for the following relation and its set of FDs:

- $R(A, B, C, D, E)$
- FD: $\{ A \rightarrow B, D \rightarrow E \}$

Top-Down Approach

First, we need to find all the subset of the R with size 4 and see if their closure will produce the whole relation:

- $ABCD^+ = \{A, B, C, D, E\} \Rightarrow$ A superkey
- $ABCE^+ = \{A, B, C, E\} \Rightarrow$ Not a superkey
- $ACDE^+ = \{A, B, C, D, E\} \Rightarrow$ A superkey
- $ABDE^+ = \{A, B, D, E\} \Rightarrow$ Not a superkey
- $ABCE^+ = \{A, B, C, E\} \Rightarrow$ Not a superkey
- $BCDE^+ = \{B, C, D, E\} \Rightarrow$ Not a superkey

Candidate Key

Example 1: Find all the candidate keys for the following relation and its set of FDs:

- $R(A, B, C, D, E)$
- FD: $\{A \rightarrow B, D \rightarrow E\}$

Now we have to follow the same approach for the superkeys which are found in the previous slide:

First, we need to check $ABCD^+ = \{A, B, C, D, E\}$.

- $ABC^+ = \{A, B, C\} \Rightarrow$ Not a superkey
- $ABD^+ = \{A, B, D, E\} \Rightarrow$ Not a superkey
- $ACD^+ = \{A, B, C, D, E\} \Rightarrow$ A superkey
- $BCD^+ = \{B, C, D, E\} \Rightarrow$ Not a superkey

Now, we have find ACD as a superkey.

Candidate Key

Example 1: Find all the candidate keys for the following relation and its set of FDs:

- $R(A, B, C, D, E)$
- FD: $\{ A \rightarrow B, D \rightarrow E \}$

Now we have to follow the same approach for the superkeys which are found in the previous slide:

Second, we need to check $ACDE^+ = \{A, B, C, D, E\}$.

- $ACD^+ = \{A, B, C, D, E\} \Rightarrow$ **A superkey**
- $ADE^+ = \{A, B, D, E\} \Rightarrow$ **Not a superkey**
- $ACE^+ = \{A, B, C, E\} \Rightarrow$ **Not a superkey**
- $CDE^+ = \{C, D, E\} \Rightarrow$ **Not a superkey**

Now, we have find ACD as a superkey.

Candidate Key

Example 1: Find all the candidate keys for the following relation and its set of FDs:

- $R(A, B, C, D, E)$
- FD: $\{ A \rightarrow B, D \rightarrow E \}$

And we can state that ACD is a superkey but it may **NOT** be a candidate key.

Is ACD a candidate key?

$$AC^+ = \{A, B, C\}$$

$$AD^+ = \{A, B, D, E\}$$

$$CD^+ = \{C, D, E\}$$

Since no subset of this superkey is a key, we conclude that ACD is the only candidate key.

Prime and Non-Prime Attributes

Example 1: Find all the candidate keys for the following relation and its set of FDs:

- $R(A, B, C, D, E)$
- FD: $\{ A \rightarrow B, D \rightarrow E \}$

What are the prime attributes of R?

Prime Attributes = $\{A, C, D\}$

What are the non-prime attributes of R?

Non-Prime Attributes = $\{B, E\}$

Candidate Key

Example2: Find all the candidate keys of relation R with the given FDs:

- R(A, B, C, D, E, F)
- FD: { $AB \rightarrow C$, $C \rightarrow DE$, $E \rightarrow F$, $D \rightarrow A$, $C \rightarrow B$ }

Candidate Key

Example 2: Find all the candidate keys of R relation with the FD's:

- $R(A, B, C, D, E, F)$
- FD: $\{AB \rightarrow C, C \rightarrow DE, E \rightarrow F, D \rightarrow A, C \rightarrow B\}$

First, we the whole relation and then we need to remove the attributes which can be determined from the other present attributes by using the functional dependencies.

$$ABCDEF^+ = \{A, B, C, D, E, F\}$$

From $AB \rightarrow C \Rightarrow ABDEF^+$

From $AB \rightarrow C, C \rightarrow DE \Rightarrow AB \rightarrow DE \Rightarrow ABF^+$

From $AB \rightarrow DE \Rightarrow AB \rightarrow D \ \& \ AB \rightarrow E$

From $AB \rightarrow E \ \& \ E \rightarrow F \Rightarrow AB^+ = \{A, B, C, D, E, F\}$

Thus, AB is a superkey but may **NOT** be a candidate key. (Need to test if any subset of it is a key or not)

Candidate Key

Example 2: Find all the candidate keys of R with the FD's:

- $R(A, B, C, D, E, F)$
- FD: $\{AB \rightarrow C, C \rightarrow DE, E \rightarrow F, D \rightarrow A, C \rightarrow B\}$

AB is a superkey but it may **NOT** be a candidate key.

$$A^+ = \{A\}$$

$$B^+ = \{B\}$$

Since no proper subset of AB is a superkey, we conclude that AB is a candidate key.

But is it the only candidate key of R?

Candidate Key

Example 2: Find all the candidate keys of R with the FD's:

- $R(A, B, C, D, E, F)$
- FD: $\{AB \rightarrow C, C \rightarrow DE, E \rightarrow F, D \rightarrow A, C \rightarrow B\}$

Is AB the only candidate key?

we need to see if any proper subset of AB is in the RHS of any functional dependency?

From AB & $C \rightarrow B \Rightarrow AC$ is another superkey.

From AB & $D \rightarrow A \Rightarrow DB$ is another superkey.

From AB & $D \rightarrow A$ & $C \rightarrow B \Rightarrow DC$ is another superkey.

Now we have to check if any one of AC, DB, and DC is a candidate key.

Candidate Key

Example 2: Find all the candidate keys for relation R with the FD's:

- $R(A, B, C, D, E, F)$
- FD: $\{AB \rightarrow C, C \rightarrow DE, E \rightarrow F, D \rightarrow A, C \rightarrow B\}$

From $AB \ \& \ C \rightarrow B \Rightarrow AC$ is another superkey.

Is AC a candidate key as well?

$$A^+ = \{A\}$$

$$C^+ = \{A, B, C, D, E, F\}$$

And since C itself is a superkey, then AC is not a candidate key.

Similarly, DC is **not** a candidate key as well.

Candidate Key

Example 2: Find all the candidate keys of R with the FD's:

- R(A, B, C, D, E, F)
- FD: { $AB \rightarrow C$, $C \rightarrow DE$, $E \rightarrow F$, $D \rightarrow A$, $C \rightarrow B$ }

From AB & $D \rightarrow A \Rightarrow DB$ is another superkey.

Is DB a candidate key?

$$D^+ = \{D, A\}$$

$$B^+ = \{B\}$$

From the above, we conclude that DB is also candidate key.

Candidate Key

Example 2: Find all the candidate keys of R with the FD's:

- $R(A, B, C, D, E, F)$
- FD: $\{AB \rightarrow C, C \rightarrow DE, E \rightarrow F, D \rightarrow A, C \rightarrow B\}$

The candidate keys are:

AB, C, BD

Prime Attributes of R = $\{A, B, C, D\}$

Non-Prime Attributes of R = $\{E, F\}$

Canonical Cover

Let F be a set of FD's. A canonical cover (or minimal basis) of F is a set G of FD's that satisfies the following conditions:

- 1- G is equivalent to F , that is, $G \equiv F$
- 2- Every FD in G has a single attribute on the right hand side.
- 3- G is minimal, that is, if we obtain a set H of FD's from G by deleting some FD's in G or by reducing the left hand side of some FD's, then H won't be equivalent to F (that is, $H \not\equiv F$)

Canonical Cover

Example 3: Find a canonical cover G for $F = \{A \rightarrow BC, B \rightarrow C, A \rightarrow B, AB \rightarrow C\}$ over $R(A, B, C)$
(Note that we remove ANY trivial FD in F right away, if there is any)

Step 1: Simplify each FD via decomposition to have only one attribute on the RHS

- $G = \{A \rightarrow B, A \rightarrow C, B \rightarrow C, A \rightarrow B, AB \rightarrow C\}$

Step 2: Minimize the LHS X of every FD in $G = \{A \rightarrow B, A \rightarrow C, B \rightarrow C, A \rightarrow B, AB \rightarrow C\}$

- A single attribute LHS is obviously minimal.

$$\begin{aligned} AB \rightarrow C \quad & \{A, B\}^+ = \{A, B, C\} \\ & \{A\}^+ = \{A, B, C\} \\ & \{B\}^+ = \{C\} \end{aligned}$$

Since $\{A\}^+ = \{A, B\}^+$, we can replace the FD $AB \rightarrow C$ by $A \rightarrow C$.

Canonical Cover

Example 3: Find a canonical cover G for $F = \{A \rightarrow BC, B \rightarrow C, A \rightarrow B, AB \rightarrow C\}$ over $R(A, B, C)$

Step 3: Delete redundant FDs, if any

$$G = \{A \rightarrow B, A \rightarrow C, B \rightarrow C, A \rightarrow B, A \rightarrow C\}$$

✓ First, we can safely delete all duplicate FDs.

- $G = \{A \rightarrow B, A \rightarrow C, B \rightarrow C\}$

✓ Then we check if G can be reduced further?

For every FD $X \rightarrow Y$ in G ,

1. Remove $X \rightarrow Y$ from G ; call the result G' .
2. Compute X^+ under G' . If $Y \in X^+$, then $X \rightarrow Y$ is redundant and can be removed from G

I. $G = \{A \rightarrow B, A \rightarrow C, B \rightarrow C\}$ & $G' = \{A \rightarrow C, B \rightarrow C\}$

Under G' , $\{A\}^+ = \{A, C\}$

Because we cannot get B from A without the FD $A \rightarrow B$, it is not redundant

Canonical Cover

Example 3: Find a canonical cover G for $F = \{A \rightarrow BC, B \rightarrow C, A \rightarrow B, AB \rightarrow C\}$ over $R(A, B, C)$

Step 3: Delete redundant FDs, if any

- $G = \{A \rightarrow B, A \rightarrow C, B \rightarrow C\}$

For every FD $X \rightarrow Y$ in G ,

1. Remove $X \rightarrow Y$ from G ; call the result G' .
2. Compute X^+ under G' . If $Y \in X^+$, then $X \rightarrow Y$ is redundant and can be removed from G

II. $G = \{A \rightarrow B, A \rightarrow C, B \rightarrow C\}$ & $G' = \{A \rightarrow B, B \rightarrow C\}$

- Under G' , $\{A\}^+ = \{A, B, C\}$
- Because $C \in A^+$, the FD $A \rightarrow C$ is redundant.
- $G = \{A \rightarrow B, B \rightarrow C\}$

Canonical Cover

Example 1: Find a canonical cover G for $F = \{A \rightarrow BC, B \rightarrow C, A \rightarrow B, AB \rightarrow C\}$ over $R(A, B, C)$

Step 3: Delete redundant FDs, if any

- $G = \{A \rightarrow B, B \rightarrow C\}$

For every FD $X \rightarrow Y$ in G ,

1. Remove $X \rightarrow Y$ from G ; call the result G' .
2. Compute X^+ under G' , If $Y \in X^+$, then $X \rightarrow Y$ is redundant and can be removed from G

III. $G = \{A \rightarrow B, B \rightarrow C\}$ & $G' = \{A \rightarrow B\}$

- Under G' , $\{B\}^+ = \{B\}$
- Because C is not in B^+ , the FD $B \rightarrow C$ is not redundant.

$G = \{A \rightarrow B, B \rightarrow C\}$ is a canonical for $F = \{A \rightarrow BC, B \rightarrow C, A \rightarrow B, AB \rightarrow C\}$

G and F are functionally equivalent.

Canonical Cover

- **Example 4:** Find a canonical cover of $F = \{ A \rightarrow B, BC \rightarrow D, AC \rightarrow D \}$

- **Step 1:** Simplify each FD to have one attribute on the RHS

- **Step 2:** Minimize the LHS X of every FD: $X \rightarrow A$

- A single attribute LHS is obviously minimal.

I. $BC \rightarrow D$
 $\{B,C\}^+ = \{B,C, D\}$
 $\{B\}^+ = \{B\}$
 $\{C\}^+ = \{C\}$

II. $AC \rightarrow D$
 $\{A, C\}^+ = \{A, C, B, D\}$
 $\{A\}^+ = \{A, B\}$
 $\{C\}^+ = \{C\}$

- Still F hasn't changed.

Canonical Cover

Example 4: Find a canonical cover of $F = \{ A \rightarrow B, BC \rightarrow D, AC \rightarrow D \}$

Step 3: Remove redundant FDs, if any

For every FD $X \rightarrow Y$ in G ,

1. Remove $X \rightarrow Y$ from G ; call the result G' .
2. Compute X^+ under G' , If $Y \in X^+$, then $X \rightarrow Y$ is redundant and can be removed from G

I. $G = \{ A \rightarrow B, BC \rightarrow D, AC \rightarrow D \}$ & $G' = \{ BC \rightarrow D, AC \rightarrow D \}$

Under G' , $\{A\}^+ = \{A\}$

Because B is not in A^+ , the FD $A \rightarrow B$, it is not redundant

II. $G = \{ A \rightarrow B, BC \rightarrow D, AC \rightarrow D \}$ & $G' = \{ A \rightarrow B, AC \rightarrow D \}$

Under G' , $\{BC\}^+ = \{B, C\}$

Because D is not in $\{BC\}^+$, then $BC \rightarrow D$ is not a redundant FD.

Canonical Cover

Example 4: Find a canonical cover of $F = \{ A \rightarrow B, BC \rightarrow D, AC \rightarrow D \}$

Step 3: Remove redundant FDs, if any

For every FD $X \rightarrow Y$ in G ,

1. Remove $X \rightarrow Y$ from G ; call the result G' .
2. Compute X^+ under G' , If $Y \in X^+$, then $X \rightarrow Y$ is redundant and can be removed from G

III. $G = \{ A \rightarrow B, BC \rightarrow D, AC \rightarrow D \}$ & $G' = \{ A \rightarrow B, BC \rightarrow D \}$

Under G' , $\{AC\}^+ = \{A, B, C, D\}$

Because $D \in \{AC\}^+$, then the FD $AC \rightarrow D$ is redundant

$G = \{ A \rightarrow B, BC \rightarrow D \}$

Canonical Cover

A set F of functional dependencies can have multiple canonical covers

Example 3: $F = \{ A \rightarrow BC, B \rightarrow AC, C \rightarrow AB \}$

Each of the following sets of FD's is a canonical cover for F :

$G_1 = \{ A \rightarrow B, B \rightarrow C, C \rightarrow A \}$

$G_2 = \{ A \rightarrow B, B \rightarrow AC, C \rightarrow B \}$

$G_3 = \{ A \rightarrow C, C \rightarrow B, B \rightarrow A \}$

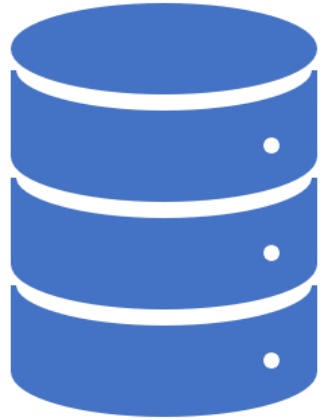
$G_4 = \{ A \rightarrow C, B \rightarrow C, C \rightarrow AB \}$

$G_5 = \{ A \rightarrow BC, B \rightarrow A, C \rightarrow A \}$

References

- <https://youtu.be/AGFUfLPFJ7w>
- <https://youtu.be/jFlxFODMLYw>
- <https://www.geeksforgeeks.org/canonical-cover/>
- <http://www.mathcs.emory.edu/~cheung/Courses/377/Syllabus/9-NormalForms/FindKeys.html>

Welcome!



Comp 353 – Databases

Tutorial 9

Summer 2021

Sections CC & CD

Concordia University - Department of Computer Science & Software Engineering

Today's Plan

- ❖ Projecting FDs
- ❖ Boyce-Codd Normal Form & 3NF
- ❖ Decomposition to BCNF or 3NF
- ❖ Chase Test

Projecting FDs

We are given a relation R with a set F of FDs.

Question: If we remove some attributes from R to get $R1$, what FDs will hold on $R1$?

Answer: Find the **projection of F on $R1$** . That is, find all (NON-trivial) FD's that follow from F and involve only the attributes of $R1$.

Algorithm ;

1. Initialize $F1 = \{\}$.
2. Construct a set of all subsets X of attributes of $R1$.
3. For each X_i in X , compute X_i^+ under F . Note that X_i^+ may consist of attributes that are not in $R1$.
4. Add to $F1$, every nontrivial FD's $X \rightarrow A$ such that **A is both in X_i^+ and an attribute of $R1$** .

Projecting FDs

Example 1: Let $R = \{A, B, C, D\}$ with $F = \{A \rightarrow B, B \rightarrow C, C \rightarrow D\}$. Project F onto $R_1(A, C, D)$, call the result F_1 .

1. $F_1 = \{\}$.
2. Construct a set of all subsets of attributes of R_1 called X .
$$X = \{\emptyset, \{A\}, \{C\}, \{D\}, \{A, C\}, \{A, D\}, \{C, D\}, \{A, C, D\}\}$$
3. Compute X_i^+ for all members of X under F . X_i^+ may consists of attributes that are not in R_1 .
4. Add to F_1 all nontrivial FD's $X \rightarrow A$ such that A is both in X_i^+ and an attributes of R_1 .

Projecting FDs

Example 1: Let $R = \{A, B, C, D\}$ with $F = \{A \rightarrow B, B \rightarrow C, C \rightarrow D\}$. Project F onto $R_1(A, C, D)$, call the result F_1 .

1. $F_1 = \{\}$.
2. Construct a set of all subsets of attributes of R_1 called X .

$$X = \{\emptyset, \{A\}, \{C\}, \{D\}, \{A, C\}, \{A, D\}, \{C, D\}, \{A, C, D\}\}$$

3. Compute X_i^+ for all members of X under F . X_i^+ may consists of attributes that are not in R_1 .

$$\{\emptyset\}^+ = \emptyset \quad \{A\}^+ = \{A, B, C, D\} \quad \{C\}^+ = \{C, D\} \quad \{D\}^+ = \{D\} \quad \{C, D\}^+ = \{C, D\}$$

Since $\{A\}^+ = \{A, B, C, D\}$ so we can ignore all supersets of $\{A\}$

4. Add to F_1 all nontrivial FD's $X \rightarrow A$ such that A is both in X_i^+ and an attributes of R_1 .

$$\{A\}^+ = \{A, B, C, D\} \Rightarrow A \rightarrow C, A \rightarrow D$$

$$\{C\}^+ = \{C, D\} \Rightarrow C \rightarrow D$$

$$\{C, D\}^+ = \{C, D\} \Rightarrow C \rightarrow D$$

$$F_1 = \{A \rightarrow C, A \rightarrow D, C \rightarrow D\}$$

Third Normal Form - 3NF

To identify a 3NF relation we need to take a few steps:

Each **nontrivial** functional dependency $X \rightarrow A$ should have **at least one** of the following characteristics:

- X is a superkey.

OR

- A is a prime attribute.

Third Normal Form - 3NF

Example 2: Is the following relation in 3NF?

R (A, B, C, D, E, F)

FD = {AB \rightarrow CDEF, BD \rightarrow F}

First find the candidate keys (from which we get the prime attributes):

Third Normal Form - 3NF

Example 2: Is the following relation in the 3NF?

$R(A, B, C, D, E, F)$

$FD = \{AB \rightarrow CDEF, BD \rightarrow F\}$

Candidate keys : $\{AB\}$

- Prime Attributes = $\{A, B\}$
- Non-Prime Attributes = $\{C, D, E, F\}$

- $AB \rightarrow CDEF$ satisfies the 3NF requirement
- $BD \rightarrow F$ does not satisfy the 3NF requirement since BD is not a superkey and F is not a prime attribute.

Boyce-Codd Normal Form (BCNF)

Let R be a relation schema with FD's F . We say that R is in BCNF w.r.t. F if in each **nontrivial** functional dependency $X \rightarrow A$:

- X is a superkey

To determine whether R with F is in BCNF

- Ignore all trivial FDs
- Check whether the LHS X of each FD's is a superkey. That is check if $X^+ = R$

Boyce-Codd Normal Form (BCNF)

Example 3: Is the following relation in BCNF (w.r.t. F)?

$R(A, B, C, D, E)$

$F = \{A \rightarrow BCDE, BC \rightarrow ACE, D \rightarrow E\}$

Boyce-Codd Normal Form (BCNF)

Example 3: Is the following relation in BCNF?

$R(A, B, C, D, E)$

$FD = \{A \rightarrow BCDE, BC \rightarrow ACE, D \rightarrow E\}$

To answer the question, we compute the closures of the LHS X of every FD to see if $X^+ = R$ or not.

$A^+ = \{A, B, C, D, E\} = R$

$BC^+ = \{A, B, C, D, E\} = R$

$D^+ = \{D, E\} \neq R$, and hence R is not in BCNF (since the LHS D is not a superkey).

Normal Forms

Example 4: Suppose R (A, B, C, D, E). Is R in BCNF or 3NF, if the FDs are as follows?

FD = { $AE \rightarrow BC$, $AC \rightarrow D$, $CD \rightarrow BE$, $D \rightarrow E$ }

Candidate keys: {AD, AC, AE}

	$AE \rightarrow BC$	$AC \rightarrow D$	$CD \rightarrow BE$	$D \rightarrow E$
BCNF				
3NF				

Normal Forms

Example 4: Suppose R (A, B, C, D, E). Is R in BCNF or 3NF, if the FDs are as follows?

FD = { $AE \rightarrow BC$, $AC \rightarrow D$, $CD \rightarrow BE$, $D \rightarrow E$ }

Candidate keys: {AD, AC, AE}

	$AE \rightarrow BC$	$AC \rightarrow D$	$CD \rightarrow BE$	$D \rightarrow E$
BCNF	Yes	Yes	No	No
3NF	Yes	Yes	No	Yes

Decomposition into BCNF

Example 5: Suppose $R = \{A, B, C, D, E\}$ and $F = \{A \rightarrow BC, C \rightarrow DE\}$. If R is not in BCNF w.r.t. F , decompose R into BCNF relations.

➤ BCNF checking:

$A^+ = \{A, B, C, D, E\}$ $\{A\}$ is a superkey
 $C^+ = \{C, D, E\}$ $\{C\}$ is NOT a superkey

➤ For every FD $A \rightarrow B$ in F that violates the BCNF requirement:

1. Decompose R into two relations: AB and $R - B$, and project F onto the new relations
2. Repeatedly do the following

If either $R-B$ or AB is not in BCNF w.r.t their corresponding FDs, decompose it further.

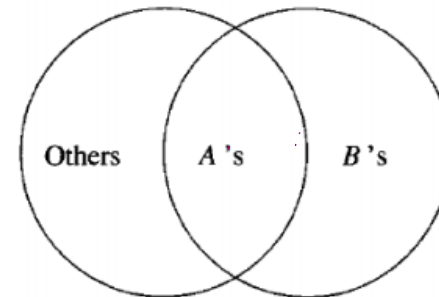


Figure 3.8: Relation schema decomposition based on a BCNF violation

Decomposition into BCNF

Example 5: Suppose $R = \{A, B, C, D, E\}$ and $F = \{A \rightarrow BC, C \rightarrow DE\}$, Decompose R to BCNF?

➤ Pick every FD $X \rightarrow A$ that violates the BCNF requirement: $(C \rightarrow DE)$

1. Decompose R into two relations: XA and $R - A$

$$R1 = \{C, D, E\} \quad R2 = \{A, B, C\}$$

2. project the FDs onto the new relations

$$F1 = \{C \rightarrow DE\} \quad F2 = \{A \rightarrow BC\}$$

3. If either $R-A$ or XA is not in BCNF decompose it further (recursion)

$$R1 = \{C, D, E\}, F1 = \{C \rightarrow DE\}$$

$C^+ = \{C, D, E\}$ $\{C\}$ is a superkey

$$R2 = \{A, B, C\}, F2 = \{A \rightarrow BC\}$$

$A^+ = \{A, B, C\}$ $\{A\}$ is a superkey

Decomposition into BCNF

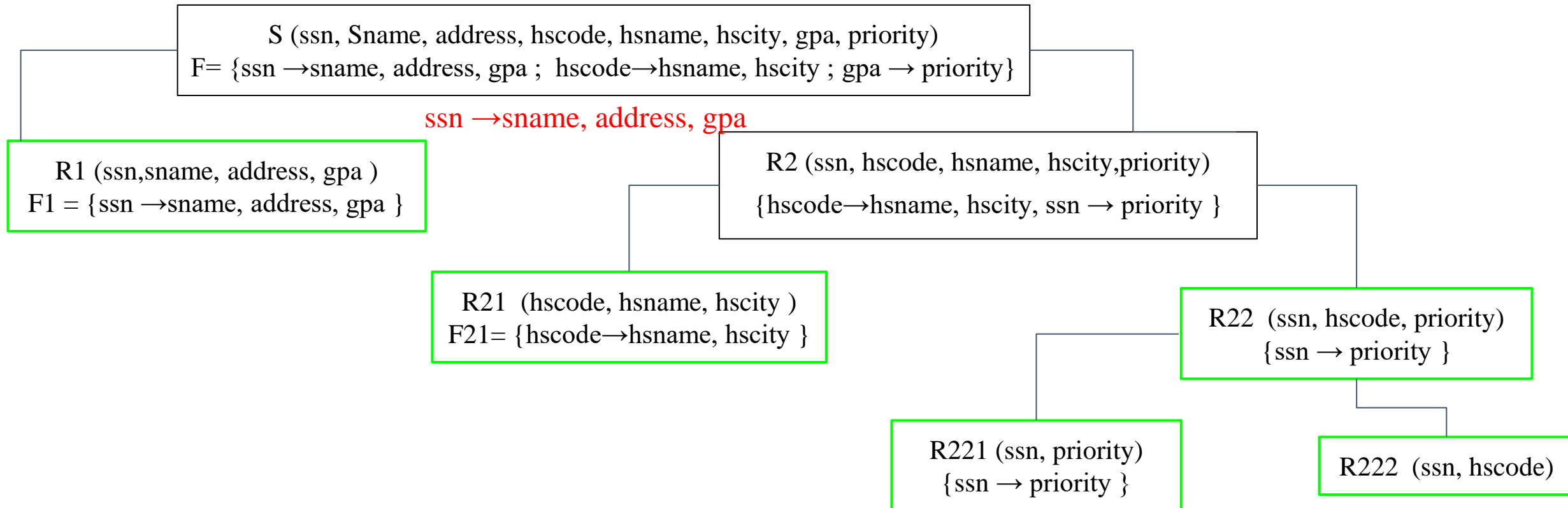
Example 6: If the following relation is not in BCNF w.r.t. F , decompose it into BCNF relations.

Student (ssn, Sname, address, hscore, hname, hscity, gpa, priority)

$F = \{ \text{ssn} \rightarrow \text{sname, address, gpa} ; \text{hscore} \rightarrow \text{hname, hscity} ; \text{gpa} \rightarrow \text{priority} \}$

Decomposition into BCNF

Example 6: Is the following relation in BCNF? **NO** , The key for the relation is {ssn, hscore}



Decomposition into 3NF

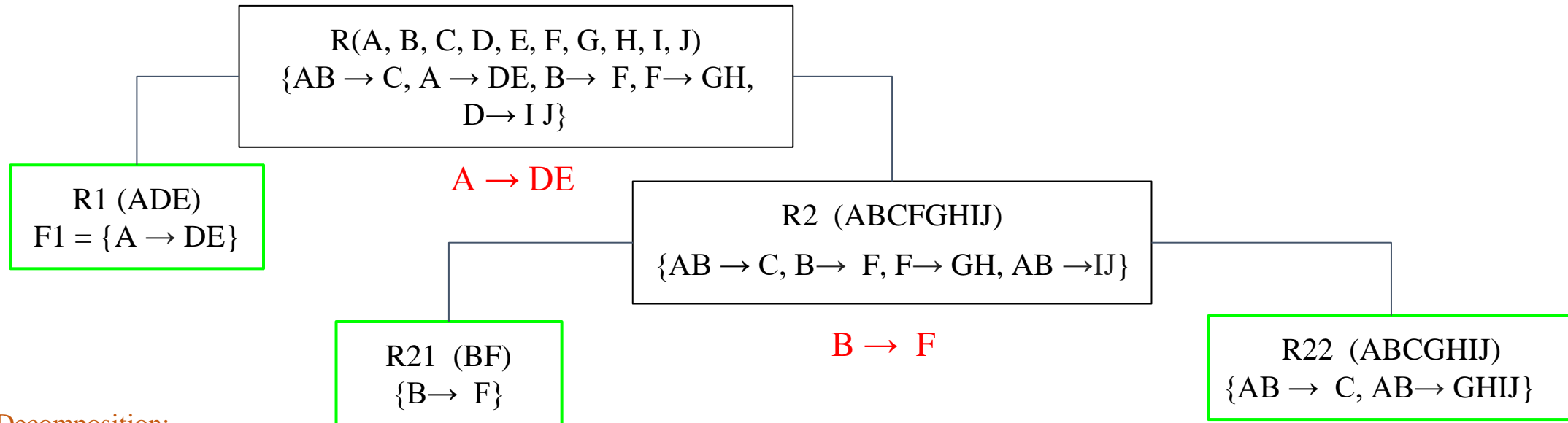
Example 7: Decompose the following relation into 3NF (if not already in 3NF!)?

$R = (A, B, C, D, E, F, G, H, I, J)$, $F = \{A B \rightarrow C, A \rightarrow DE, B \rightarrow F, F \rightarrow G, H, D \rightarrow I J\}$

Decomposition into 3NF

Example 7: Decompose the following relation to 3NF?

$R = (A, B, C, D, E, F, G, H, I, J)$, $F = \{AB \rightarrow C, A \rightarrow DE, B \rightarrow F, F \rightarrow GH, D \rightarrow IJ\}$ $\{A B\}$ is the only key.



Decomposition:

R1 (ADE)

R21 (BF)

R22 (ABCGHIJ)

R4(FGH)

R5(DIJ)

Chase Test

Example 8: Using the chase test, determine if the following decomposition is lossless or not?

$R(A, B, C, D, E)$

$FD = \{AB \rightarrow C, C \rightarrow E, B \rightarrow D, E \rightarrow A\}$

$R_1(B, C, D)$

$R_2(A, C, E)$

The Chase Test

Using the chase test method (below), we can determine if a decomposition is lossless or not.

$R(A, B, C, D, E)$ with $FD = \{AB \rightarrow C, C \rightarrow E, B \rightarrow D, E \rightarrow A\}$

$R_1(B, C, D)$

$R_2(A, C, E)$

Firstly, we need to initialize the table by putting alphas in the right places according to the resulting relations.

	A	B	C	D	E
R1		α	α	α	
R2	α		α		α

Chase Test

Using the chase test, determine if the following decomposition is lossless or lossy?

$R(A, B, C, D, E)$ with $FD = \{AB \rightarrow C, C \rightarrow E, B \rightarrow D, E \rightarrow A\}$

$R_1(B, C, D)$

$R_2(A, C, E)$

Then we need to go through the FDs.

➤ $AB \rightarrow C$ does not change the table.

	A	B	C	D	E
R1		α	α	α	
R2	α		α		α

Chase Test

Using the chase test, determine if the following decomposition is lossless or lossy?

R (A, B, C, D, E) with FD = {AB \rightarrow C, C \rightarrow E, B \rightarrow D, E \rightarrow A}

R1 (B, C, D)

R2(A, C, E)

- Then we check **C \rightarrow E**. To satisfy this FD, we add the corresponding alpha (red) in the table.

	A	B	C	D	E
R1		α	α	α	α
R2	α		α		α

Chase Test

Using the chase test, determine if the following decomposition is lossless or lossy?

$R(A, B, C, D, E)$ with $FD = \{AB \rightarrow C, C \rightarrow E, B \rightarrow D, E \rightarrow A\}$

$R_1(B, C, D)$

$R_2(A, C, E)$

➤ Then we check $B \rightarrow D$. This FD satisfied the table and hence does not change the table.

	A	B	C	D	E
R1		α	α	α	α
R2	α		α		α

Chase Test

Using the chase test, determine if the following decomposition is lossless or lossy?

$R(A, B, C, D, E)$ with $FD = \{AB \rightarrow C, C \rightarrow E, B \rightarrow D, E \rightarrow A\}$

$R_1(B, C, D)$

$R_2(A, C, E)$

Then check $E \rightarrow A$. To satisfy $E \rightarrow A$, we add the corresponding alpha (red) under A.

	A	B	C	D	E
R1	α	α	α	α	α
R2	α		α		α

Chase Test

Using the chase test, determine if the following decomposition is lossless or lossy?

$R(A, B, C, D, E)$ with $FD = \{AB \rightarrow C, C \rightarrow E, B \rightarrow D, E \rightarrow A\}$

$R_1(B, C, D)$

$R_2(A, C, E)$

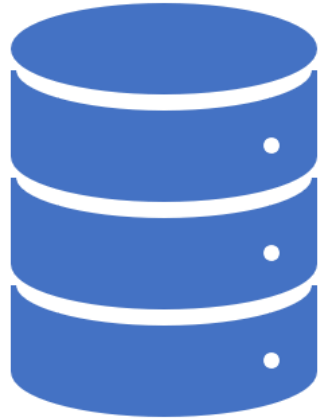
Now, we see that **we got a row of alphas**, based on which we conclude that this **decomposition is lossless**.

	A	B	C	D	E
R1	α	α	α	α	α
R2	α		α		α

Note: For a **binary** decomposition, we have an easier test. That is, a decomposition of R into R_1 and R_2 is lossless if

$R_1 \cap R_2 \rightarrow R_1$ or $R_1 \cap R_2 \rightarrow R_2$. Chase is a general method that works for binary or otherwise.

Welcome!



Comp 353 – Databases

Tutorial 10

Summer 2021

Sections CC, CD

Concordia University - Department of Computer Science & Software Engineering

Today's Plan

- ❖ Decomposition of relations into BCNF or 3NF
- ❖ Integrity Constraints
 - ❖ Assertions
 - ❖ Triggers

Decomposition into 3NF

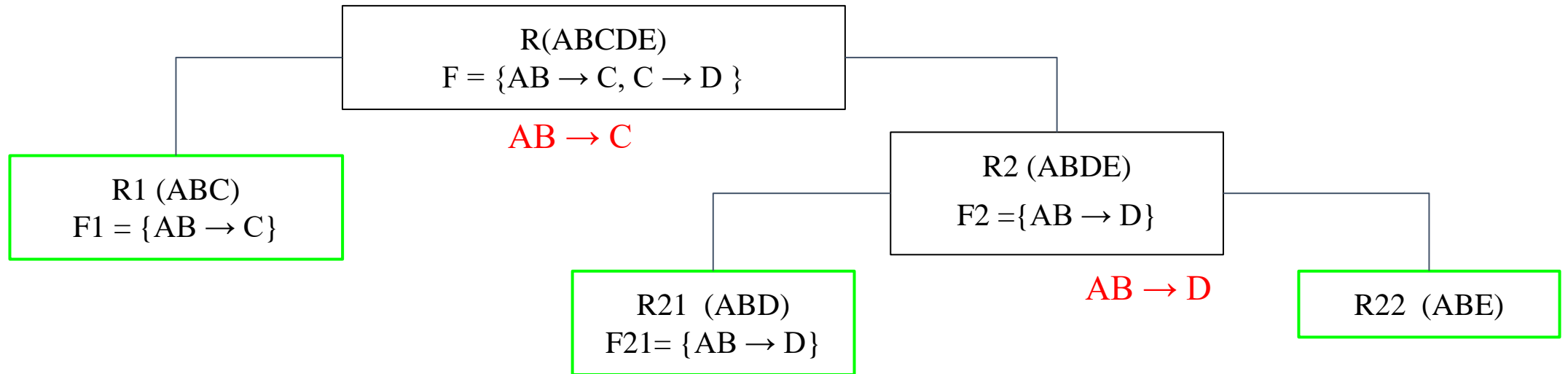
Example 1: If the following relation is not in 3NF w.r.t. F, decompose it into 3NF relations.

R(ABCDE) $F = \{AB \rightarrow C, C \rightarrow D\}$

The key for the relation is {A,B,E}

Decomposition into 3NF

Example 1: Is the following relation in 3NF? **NO**



Are all FDs preserved?

Integrity Constraints (IC's)

- An IC describes conditions that every legal/valid instance of a relation must satisfy
- To disallow inserts/deletes/updates that violate IC's
- Types of Constraints
 - Primary keys (entity integrity)
 - Foreign keys (referential integrity)
 - Attribute-based constraints
 - Restrictions on a single attribute
 - Tuple-based (row-based) Constraints
 - A constraint that include multiple columns
 - Assertions

Integrity Constraints

Example 2: Check constraints are checked when tuples are inserted or modified

```
CREATE TABLE Student (  
    sid INTEGER,  
    sname CHAR(10),  
    rating INTEGER,  
    age REAL,  
    PRIMARY KEY (sid),  
    CHECK (rating >= 1 AND rating <= 10)) ;
```

```
CREATE TABLE Student (  
    sid INTEGER,  
    sname CHAR(10),  
    rating INTEGER,  
    age REAL,  
    PRIMARY KEY (sid),  
    CONSTRAINT checkRating  
    CHECK (rating >= 1 AND rating <= 10));
```

Integrity Constraints

Example 3: Are the following statements the same?

I. CREATE TABLE Enrolled

(sid INTEGER **FOREIGN KEY REFERENCES** Student (sid),

cname CHAR(32),

PRIMARY KEY (sid, cname));

II. CREATE TABLE Enrolled

(sid INTEGER **CHECK** (sid IS NULL OR sid IN (SELECT sid FROM Student)),

cname CHAR(32),

PRIMARY KEY (sid, cname));

ASSERTIONs (Constraints Over Multiple Relations)

Example 3: Write an SQL assertion so that requires every student to be registered in at least one course.

```
CREATE ASSERTION Checkregistry
```

```
CHECK
```

```
( NOT EXISTS ((SELECT sid FROM student) EXCEPT (SELECT sid FROM enrolled)));
```

ASSERTIONs (Constraints Over Multiple Relations)

Example 4: Consider a toy example school, where the number of students and professors should be less than 500. Does the following integrity test suffice?

```
CREATE TABLE Student (  
  
    sid INTEGER PRIMARY KEY ,  
  
    sname CHAR(10),  
  
    rating INTEGER,  
  
    age REAL,  
  
    CHECK (  
  
        (SELECT COUNT (S.sid) FROM Student S) +  
  
        (SELECT COUNT (P.pid) FROM Prof P) < 500 ));
```

This integrity test is INADEQUATE as it focuses at individual relation (table Stdent could be empty and thus the integrity rule is never checked!)

ASSERTION (Constraints Over Multiple Relations)

Example 5: Consider the same example of a small school, where the number of students and professors should be less than 500.

```
CREATE ASSERTION smallSchool
```

```
CHECK ( (SELECT COUNT (S.sid) FROM Student S)
```

```
+
```

```
(SELECT COUNT (P.pid) FROM Prof P) < 500)
```

```
);
```

Triggers

- ❖ Event-Condition-Action (ECA) rules: procedures that start automatically if a change occurs to the DBMS
- ❖ When **event** occurs, check **condition**; if true do **action(s)**
 - ❖ Triggers can be triggered **BEFORE** or **AFTER** an event
 - ❖ The events can be **INSERT**, **UPDATE**, and **DELETE**
- ❖ Each trigger is given a name.
- ❖ In some versions of MySQL, you may not be able to add same triggers on a table. This means having two AFTER INSERT triggers for a table might not be applicable.
- ❖ Please note that to alter a trigger, you must first drop it and then rerun the create script.

Trigger

Example 6: Add triggers to control the price of the books to avoid having a book tuple with a negative price or greater than \$1000.

Trigger

Example 6: Add a trigger to control the price of the inserted books.

```
DELIMITER $$ #What does this line do?

CREATE TRIGGER CHECK_PRICE_ON_INSERT

    BEFORE INSERT ON COMP353.BOOKS

    FOR EACH ROW

    BEGIN

        # TO AVOID ENTERING INVALID PRICES

        IF (NEW.price < 0 OR NEW.price > 1000) THEN

            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'THE PRICE IS NOT IN RANGE 0 AND 1000!';

        END IF;

    END;$$

DELIMITER ;
```

Trigger

Example 6 (cont'd): Add a trigger to control the price of the books.

Question: Does the previous trigger run when the price is changed?

No, it would be activated when inserting a new tuple?

So, what should be done?

Define a new trigger with a different name but similar structure.

Trigger

Example 6 (cont'd): Add a trigger to control the price of the updated tuples in books table.

```
DROP TRIGGER IF EXISTS CHECK_PRICE_ON_UPDATE;

DELIMITER $$

CREATE TRIGGER CHECK_PRICE_ON_UPDATE
    BEFORE UPDATE ON COMP353.BOOKS
    FOR EACH ROW
    BEGIN
        IF (NEW.price < 0 OR NEW.price > 1000) THEN
            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'THE PRICE IS NOT IN RANGE 0 AND 1000!';
        END IF;
    END;$$

DELIMITER ;
```


Trigger

Example 7: How can we prevent users from inserting a book title twice?

(This is just to show how you can add the conditions, but normally you can use the UNIQUE constraint).

- ✓ First, we drop the trigger!
- ✓ Then, add the condition to the previous trigger to check if the title exists
- ✓ Finally, returns an error to the user if exists or accepts the new data.

Trigger

Example 7: How can we prevent users from inserting a book title twice?

```
DELIMITER $$

CREATE TRIGGER CHECK_PRICE_ON_INSERT
    BEFORE INSERT ON COMP353.BOOKS
    FOR EACH ROW
    BEGIN
        DECLARE SAME_NAME_COUNT INT DEFAULT 0;
        IF (NEW.price < 0 OR NEW.price > 1000) THEN
            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'THE PRICE IS NOT IN RANGE 0 AND 1000!';
        END IF;
        SET SAME_NAME_COUNT = (SELECT COUNT(*) FROM COMP353.BOOKS WHERE BOOK_TITLE = NEW.BOOK_TITLE);
        IF SAME_NAME_COUNT > 0 THEN
            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'THE TITLE HAS BEEN ALREADY INSERTED!';
        END IF;
    END;$$

DELIMITER ;
```

Trigger

Example 7 (cont'd): preventing users from inserting a book title twice?

Note that the same procedure should be followed for an UPDATE action.

You have to drop the action in older versions and recreate the trigger.

We would like in general to be able to define multiple triggers for a table.

Trigger

Example 8: Add a logger to your application which logs the information any time you add a publisher.

First, we need to create a table called logger.

Then, we add a trigger for the publisher.

The trigger should add a row to the log table when a new publisher is added.

We can also add some constraints to the table such as avoiding duplicate names.

Trigger Syntax

Example 8: Add a logger to your application which logs the information any time you add a publisher.

```
CREATE TABLE LOGS(  
    log_id INT PRIMARY KEY AUTO_INCREMENT,  
    log_text VARCHAR(255) NOT NULL  
);
```

Trigger Syntax

Example 8 (cont'd): Add a logger to your application which logs the information any time you add a publisher?

```
DELIMITER $$

CREATE TRIGGER AFTER_INSERT_PUBLISHER
    BEFORE INSERT ON COMP353.PUBLISHERS
    FOR EACH ROW
    BEGIN

        DECLARE PUBLISHER_NAME_COUNT INT DEFAULT 0;

        DECLARE LOG_TEXT VARCHAR(100);

        SET PUBLISHER_NAME_COUNT = (SELECT COUNT(*) FROM COMP353.PUBLISHERS WHERE PUBLISHER_NAME = NEW.PUBLISHER_NAME);

        IF PUBLISHER_NAME_COUNT > 0 THEN

            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'THE PUBLISHER HAS BEEN ALREADY INSERTED!';

        END IF;

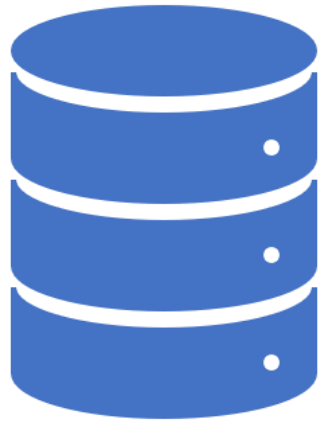
        SET LOG_TEXT = CONCAT(NEW.PUBLISHER_NAME, " IS ADDED AS A PUBLIHSER.");

        INSERT INTO logs (log_text) VALUES(LOG_TEXT);

    END;$$

DELIMITER ;
```

Welcome!



Comp 353 – Databases

Tutorial 11

Summer 2021

Sections CC, CD

Concordia University - Department of Computer Science & Software Engineering

Today's Plan > Relational Algebra

- Select $\sigma_{\langle \text{selection condition} \rangle}(\mathbf{R})$
- Project $\pi_{\langle \text{attribute list} \rangle}(\mathbf{R})$
- Rename $\rho_{\langle \text{new schema} \rangle}(\mathbf{R})$
- Union $\mathbf{A} \cup \mathbf{B}$
- Intersection $\mathbf{A} \cap \mathbf{B}$
- Difference $\mathbf{A} - \mathbf{B}$
- Cross product $\mathbf{A} \times \mathbf{B}$
- Join $\mathbf{A} \bowtie_{\langle \text{join condition} \rangle} \mathbf{B}$
- Natural Join $\mathbf{A} \bowtie \mathbf{B}$
- Division $\mathbf{A} \div \mathbf{B}$

A Simple RA Expression

Example 1: Consider the following instance of relation $R(A,B,C)$.

Which of the following tuples will be present in the output of the query $\pi_{C,B}(R)$?

- A. (2,3)
- B. (4,2,3)
- C. (6,4)
- D. (6,5)
- E. None

A	B	C
1	2	3
4	2	3
4	5	6
2	5	3
1	2	6

Relational Algebra

Question? We wish to evaluate query $R \bowtie_{R.A < S.C \text{ and } R.B < S.D} S$ over the following database instance.

Which of the following tuples will be in the result? The schema of the result is (A, R.B, S.B, C, D).

R(A, B)

A	B
1	2
3	4
5	6

S(B, C, D)

B	C	D
2	4	6
4	6	8
4	7	9

- A. (1,2,2,6,8) B. (1,2,4,4,6) C. (5,6,2,4,6) D. All E. None

Relational Algebra

Answer: Compute $R \bowtie_{R.A < S.C \text{ and } R.B < S.D} S$, where:

R(A, B)

A	B
1	2
3	4
5	6

S(B, C, D)

B	C	D
2	4	6
4	6	8
4	7	9

A. (1,2,2,6,8) B. (1,2,4,4,6) C. (5,6,2,4,6) D. All E. None

$A \Rightarrow (2,6,8) \notin S$

$B \Rightarrow (4,4,6) \notin S$

$C \Rightarrow$ Violates the condition ($5 > 2$ and $6 < 6$)

Natural Join

Example 2: Consider the following database instance:

MovieStar

StarId	Name	Gender
1	Harrison Ford	Male
2	Vivian Leigh	Female
3	Judy Garland	Female

StarsIn

MoviedId	StarId	Character
1	1	Han Solo
4	1	Indiana Jones
2	2	Scarlett O'Hara
3	3	Dorothy Gale

Natural Join:

StarId	Name	Gender	MoviedId	Character
1	Harrison Ford	Male	1	Han Solo
1	Harrison Ford	Male	4	Indiana Jones
3	Judy Garland	Female	3	Dorothy Gale
2	Vivian Leigh	Female	2	Scarlett O'Hara

Sample Queries in RA

Example 3: Find the actors of the movie “Indiana Jones”.

MovieStar

StarId	Name	Gender
1	Harrison Ford	Male
2	Vivian Leigh	Female
3	Judy Garland	Female

StarsIn

Movied	StarId	Character
1	1	Han Solo
4	1	Indiana Jones
2	2	Scarlett O'Hara
3	3	Dorothy Gale

Movie

Movied	Title	Year
1	Star Wars	1977
1	Gone with the Wind	1939
3	The Wizard of Oz	1939
4	Indiana Jones	1981

Sample Queries in RA

Example 3: Find the actors of the movie “Indiana Jones”.

- Step 1:

- $\sigma_{\text{title} = \text{“Indiana Jones”}}(\text{Movie})$

MovieId	Title	Year
4	Indiana Jones	1981

- Step 2:

- $((\sigma_{\text{title} = \text{“Indiana Jones”}}(\text{Movie})) \bowtie \text{StarsIn})$

MovieId	Title	Year	StarId	Character
4	Indiana Jones	1981	1	Indiana Jones

- Step 3:

- $\pi_{\text{name}}((\sigma_{\text{title} = \text{“Indiana Jones”}}(\text{Movie})) \bowtie \text{StarsIn} \bowtie \text{MovieStar})$

Name
Harrison Ford

Sample Queries in RA

Example 4: Find the actors (names) in “Indiana Jones” or “Star Wars”:

MovieStar

StarId	Name	Gender
1	Harrison Ford	Male
2	Vivian Leigh	Female
3	Judy Garland	Female

StarsIn

Movied	StarId	Character
1	1	Han Solo
4	1	Indiana Jones
2	2	Scarlett O'Hara
3	3	Dorothy Gale

Movie

Movied	Title	Character
1	Star Wars	1977
1	Gone with the Wind	1939
3	The Wizard of Oz	1939
4	Indiana Jones	1981

Sample Queries in RA

Example 4: Find the actors (names) in “Indiana Jones” or “Star Wars”:

- Step 1:
 - $\sigma_{\text{title} = \text{“Indiana Jones” OR title} = \text{“Star Wars”}}(\text{Movie})$

Movied	Title	Year
4	Indiana Jones	1981
1	Star Wars	1977

- Step 2:
 - $\pi_{\text{name}}((\sigma_{\text{title} = \text{“Indiana Jones” OR title} = \text{“Star Wars”}}(\text{Movie})) \bowtie \text{StarsIn} \bowtie \text{MovieStar})$

Name
Harrison Ford

Sample Queries in RA

Example 5: Find the actors in “Indiana Jones” and “Star Wars”.

MovieStar

StarId	Name	Gender
1	Harrison Ford	Male
2	Vivian Leigh	Female
3	Judy Garland	Female

StarsIn

Movied	StarId	Character
1	1	Han Solo
4	1	Indiana Jones
2	2	Scarlett O'Hara
3	3	Dorothy Gale

Movie

Movied	Title	Character
1	Star Wars	1977
1	Gone with the Wind	1939
3	The Wizard of Oz	1939
4	Indiana Jones	1981

Sample Queries in RA

Example 5: Find the actors in “Indiana Jones” and “Star Wars”.

- Step 1:
 - $\text{IndianaJones} := \pi_{\text{StarId}}(\sigma_{\text{title} = \text{“Indiana Jones”}}(\text{Movie}) \bowtie \text{StarsIn})$
 - $\text{StarWars} := \pi_{\text{StarId}}(\sigma_{\text{title} = \text{“Star Wars”}}(\text{Movie}) \bowtie \text{StarsIn})$
- Step 2:
 - $\text{Actors} := \text{IndianaJones} \cap \text{StarWars}$
- Step 3:
 - $\pi_{\text{name}}(\text{Actors} \bowtie \text{MovieStar})$

Examples on Customer Database

Example : Let Customer be a database schema defined as follows:

Customer (cid, cname, rating, salary)

Item (iid, iname, type)

Order (cid, iid, date, qty)

Example : Customer Database

Example 6: Find names of customers who've ordered item with id 100

Customer (cid, cname, rating, salary)

Item (iid, iname, type)

Order (cid, iid, date, qty)

Example : Customer Database

Example 6: Find names of customers who've ordered item with id 100

Solution 1

$$\pi_{cname}(\sigma_{iid=100}(Order \bowtie Customer))$$

Solution 2

$$\pi_{cname}(\sigma_{iid=100}(Order) \bowtie Customer)$$

Example : Customer Database

Example 7: Find names of Customers who've ordered a laptop (i.e., an item of type "laptop")

Customer (cid, cname, rating, salary)

Item (iid, iname, type)

Order (cid, iid, date, qty)

Example : Customer Database

Example 7: Find names of Customers who've ordered a laptop (i.e., an item of type “laptop”)

Solution 1

$$\pi_{cname}(\sigma_{type="laptop"}(Item \bowtie Order) \bowtie Customer))$$

Solution 2

$$\pi_{cname}((\sigma_{type="laptop"}(Item) \bowtie Order) \bowtie Customer)$$

Example : Customer Database

Example 8: Find customers who've ordered a laptop or a desktop computer

Customer (cid, cname, rating, salary)

Item (iid, iname, type)

Order (cid, iid, date, qty)

Example : Customer Database

Example 8: Find customers who've ordered a laptop or a desktop computer

Solution 1

$$\pi_{Cname}((\sigma_{type="desktop" \vee type="laptop"}(Item) \bowtie Order) \bowtie Customer)$$

Solution 2

$$claps := \pi_{cid}(\sigma_{type="laptop"}(Item) \bowtie Order)$$
$$cdesks := \pi_{cid}(\sigma_{type="desktop"}(Item) \bowtie Order)$$
$$\pi_{Cname}((claps \cup cdesks) \bowtie Customer)$$

Example : Customer Database

Example 9: Find name and type of those items that haven't been ordered at all.

Customer (cid, cname, rating, salary)

Item (iid, iname, type)

Order (cid, iid, date, qty)

$\pi_{iname,type}((\pi_{iid}(item) - \pi_{iid}(order)) \bowtie item)$

Division

Notation: $r \div s$

Let r and s be relations on schemas R and S , respectively, where

$r = (A_1, \dots, A_m, B_1, \dots, B_n)$ and $s = (B_1, \dots, B_n)$

Then $r \div s$ is a relation with schema (A_1, \dots, A_m) , with the tuples $\{ t \mid t \in \prod r-s(r) \wedge \forall u \in s (tu \in r) \}$

Useful for expressing FORALL queries

e.g., Find movie stars who acted in every/all movies.

i.e. $A \div B$ contains all x tuples (MovieStars) such that for every y tuple (movies) in B , there is an x,y tuple in

A

Examples of Division $A \div B$

A

sno	pno
s1	p1
s1	p2
s1	p3
s1	p4
s2	p1
s2	p2
s3	p2
s4	p2
s4	p4

B1

pno
p2

A/B1

sno
s1
s2
s3
s4

B2

pno
p2
p4

A/B2

sno
s1
s4

B3

pno
p1
p2
p4

A/B3

sno
s1

Division (\div)

Example 10: Find movie stars who acted in every/all movies.

$InAll := (\pi_{StarID, MovieID} StarsIn) \div (\pi_{MovieID} Movie)$

$\pi_{Name} (InAll \bowtie MovieStar)$

Division

Example 11: Compute $S(B,A) \div Y$ where $Y(A) := \pi_C(R)$

$R(C, D, E)$

C	D	E
1	2	1
2	2	1
3	2	1

$S(A,B)$

A	B
1	2
2	2
3	2

Thank you!



Comp 353 – Databases

Tutorial 12

Summer 2021

Sections CC & CD

Concordia University - Department of Computer Science & Software Engineering

Today's Plan

- ❖ SQL Queries
 - ❖ Set Operations
 - ❖ Nested Queries
 - ❖ Aggregation Functions
 - ❖ Distinct, Order By, etc.

SQL Queries

Example 1: Find the names of actors who've been in at least one movie?

DB:

Movie(MovieID, Title, Year)

StarsIn(MovieID, StarID, role)

MovieStar(StarID, Name, Gender)

SELECT DISTINCT Name

FROM StarsIn S, MovieStar M

WHERE S.StarID = M.StarID;

SQL Queries

Example2: List in alphabetic order the names of actors who were in a movie in 1939

DB:

Movie(MovieID, Title, Year)

StarsIn(MovieID, StarID, role)

MovieStar(StarID, Name, Gender)

SELECT DISTINCT Name

FROM Movie, StarsIn, MovieStar

WHERE Movie.MovieID = StarsIn.MovieID and StarsIn.StarID =

MovieStar.StarID and Year = 1939

ORDER BY Name;

SQL Queries

Example 3: Find IDs of stars who have been in movies in 1944 and in 1974.

DB:

Movie(MovieID, Title, Year)

StarsIn(MovieID, StarID, role)

MovieStar(StarID, Name, Gender)

SQL Queries

Example 3: Find IDs of stars who have been in movies in 1944 and in 1974.

SELECT StarID

FROM Movie M, StarsIn S

WHERE M.MovieID = S.MovieID AND Year = 1944

INTERSECT

SELECT StarID

FROM Movie M, StarsIn S

WHERE M.MovieID = S.MovieID AND Year = 1974;

DB:

Movie(MovieID, Title, Year)

StarsIn(MovieID, StarID, role)

MovieStar(StarID, Name, Gender)

SQL Queries

Example 4: Find IDs of stars who have been in movies in 1944 and in 1974 **without using INTERSECT**.

DB:

Movie(MovieID, Title, Year)

StarsIn(MovieID, StarID, role)

MovieStar(StarID, Name, Gender)

SQL Queries

Example 4: Find IDs of stars who have been in movies in 1944 and in 1974 **without using INTERSECT**.

DB:

Movie(MovieID, Title, Year)

StarsIn(MovieID, StarID, role)

MovieStar(StarID, Name, Gender)

SELECT distinct S1.StarID

FROM Movie M1, StarsIn S1, Movie M2, StarsIn S2

WHERE M1.MovieID = S1.MovieID AND M1.Year = 1944

AND M2.MovieID = S2.MovieID AND M2.Year = 1974 AND

S2.StarID = S1.StarID

SQL Queries

Example 5: How many movies has Brad Pitt played in?

DB:

Movie(MovieID, Title, Year)

StarsIn(MovieID, StarID, role)

MovieStar(StarID, Name, Gender)

SELECT COUNT(*)

From StarsIn, MovieStar

WHERE StarsIn.StarID = MovieStar.StarID AND Name = 'Brad Pitt'

SQL Queries

Example 6: Find ids and names of stars who have not been in the movie with ID 28

```
SELECT M.StarID, M.Name
FROM MovieStar M
WHERE M.StarID NOT IN (SELECT S.StarID
                        FROM StarsIn S
                        WHERE MovieID=28);
```

DB:

Movie(MovieID, Title, Year)

StarsIn(MovieID, StarID, role)

MovieStar(StarID, Name, Gender)

SQL Queries

Example 7: Find the departments that have more than one faculty member?

DB:

Student(snum,sname,major,standing,age)

Enrolled(snum,cname)

Faculty(fid,fname,deptid)

```
SELECT DISTINCT f1.deptid
```

```
FROM faculty f1, faculty f2
```

```
WHERE f1.fid <>f2.fid AND F1.deptid = f2.deptid;
```

SQL Queries

Example 8: Find the ids of all students who took Operating Systems Design but did not take Database Systems

Solution 1:

```
SELECT snum
FROM enrolled e
WHERE cname = 'Operating Systems Design'
EXCEPT
SELECT snum
FROM enrolled e
WHERE cname = 'Database Systems';
```

DB:

Student(snum,sname,major,standing,age)

Enrolled(snum,cname)

Faculty(fid,fname,deptid)

SQL Queries

Example 8: Find the ids of all students who took Operating Systems Design but did not take Database Systems

DB:

Student(snum,sname,major,standing,age)

Enrolled(snum,cname)

Faculty(fid,fname,deptid)

Solution 2:

```
SELECT snum
FROM enrolled
WHERE cname = 'Operating Systems Design' AND
      snum NOT IN (SELECT snum
                    FROM enrolled
                    WHERE cname = 'Database Systems'
                    );
```

SQL Queries

Example 9: Find the name and age of the oldest student(s)

DB:

Student(snum,sname,major,standing,age)

Enrolled(snum,cname)

Faculty(fid,fname,deptid)

SQL Queries

Example 9: Find the name and age of the oldest student(s)

Solution 1:

```
SELECT sname, age
FROM student s2
WHERE NOT EXISTS (SELECT *
                   FROM student s1
                   WHERE s1.age > s2.age);
```

DB:

Student(snum,sname,major,standing,age)

Enrolled(snum,cname)

Faculty(fid,fname,deptid)

SQL Queries

Example 9: find the name and age of the oldest student(s)

Solution 2:

```
SELECT sname, age
FROM student s2
WHERE s2.age >= ALL (SELECT age
                     FROM student s1);
```

Solution 3:

```
SELECT sname, age
FROM student s2
WHERE NOT s2.age < ANY (SELECT age
                       FROM student s1);
```

DB:

Student(snum,sname,major,standing,age)

Enrolled(snum,cname)

Faculty(fid,fname,deptid)

SQL Queries

Example 9: Find the name and age of the oldest student(s)

Solution 4:

```
SELECT sname
FROM Student S
WHERE S.age= (SELECT MAX(S2.age)
              FROM Student S2);
```

DB:

Student(snum,sname,major,standing,age)

Enrolled(snum,cname)

Faculty(fid,fname,deptid)

SQL Queries

Example 10: How many students have taken a class with “Database” in the title? Note: want “distinct” for when students take different db classes .

```
SELECT COUNT(DISTINCT snum)
FROM enrolled
where cname LIKE '%Database%;
```

DB:

Student(snum,sname,major,standing,age)

Enrolled(snum,cname)

Faculty(fid,fname,deptid)

SQL Queries

Example 11: For each course with more than 1 enrollment, find the age of the youngest student who has taken the course:

DB:

Student(snum,sname,major,standing,age)

Enrolled(snum,cname)

Faculty(fid,fname,deptid)

SQL Queries

Example 11: For each course with more than 1 enrollment, find the age of the youngest student who has taken the course:

```
SELECT cname, MIN(age)
FROM Student S, Enrolled E
WHERE S.snum = E.snum
GROUP BY cname
HAVING COUNT(*) > 1;
```

DB:

Student(snum,sname,major,standing,age)

Enrolled(snum,cname)

Faculty(fid,fname,deptid)

Thank you!
