CS281 Lecture Notes

Week 4 (23.02.2015-27.02.2015)

Note Takers
Mustafa Özer
Mustafa Kağan Öztürk
Burcu Özmen
Merve Berber

24.02.2015

Relational Model

In relational model, we have relations/tables.

A relation schema is the basic information describing a table or relation.

Students (sid:string, name:string, age:integer, gpa:real) is an example of relation schema.
Sid, name, age and gpa are the attributes.

Degree of a relation: 4 (number of columns)

instance of a relation:
"10", "Zeynep", 21, 2.99
"20", "Ali", 20, 2.85

row = tuple = record

cardinality of relation: 2 (number of rows)

sid: primary key (unique identifier) (Primary key leads an unique tuple.)

(sid, gpa): superkey (primary key + attribute)

ssn and tcno are examples of candidate keys. Both can be a primary key since they are unique for every tuple.

Database is a set of relation schemas.

```
CREATE TABLE Students (sid, CHAR(20),
                       name, CHAR(30),
                       age, INTEGER,
                       gpa, REAL,
                       CONTRAINT StudentKeys,
                       PRIMARY KEY (sid))
```

If we try to insert a tuple with a sid which already appears in the relation, an error will be generated since it is a primary key. On the message StudentKeys will be displayed.

To create the table, insert the tuples one by one.
After creating the table, we can start to populate the table.

```
INSERT
INTO Stedents(sid, name, age, gpa)
VALUES (“10”, “Zeynep”, 21, 2.99)
```

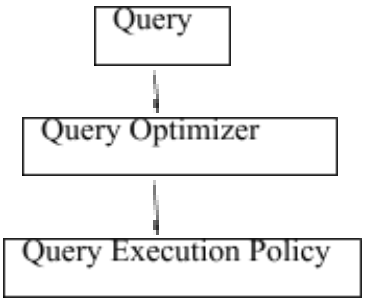
Activities on a Database (SQL)
Create, populate relation, delete, update/modify, insert

```
DELETE
FROM Stedents s
WHERE s.name = “Zeynep”
```

It deletes all tuples with “Zeynep”

```
UPDATE Students s
SET s.age = s.age+1, s.gpa = s.gpa-1
WHERE s.sid = “10”
```

QBE (Query By Example)



Query optimizer does the same thing more efficiently by correcting the logical mistakes.

Example:

```
for (i=0; i?100; i++){
        a = 10;
        k = a+i;
}
```

a = 10 is a redundant in the loop because it is initialized for 100 times. Hence, it should be written just before the loop. Query optimizer takes the statement out of the loop to work more efficiently.

```
SELECT name
FROM Students
WHERE s.age = 20
```

```
SELECT BookTitle
FROM Books
WHERE Author = “Orhan Pamuk”
```

How to get results from several tables?

Students (sid:string, name:string, age:integer, gpa:real)
Enrolled (sid:string, cid: string, grade: string)

SELECT s.name, e.cid
FROM Stedents s, Enrolled e
WHERE s.sid = e.sid AND e.grade = "A" AND e.semester = "Fall 2014"

It displays student names with grade A.

Students(sid, name)

"10" "Zeynep"
"20" "Ali"
"30" "Alper"
"40" "Zeynep"

Enrolled(sid, cid, semester, grade)

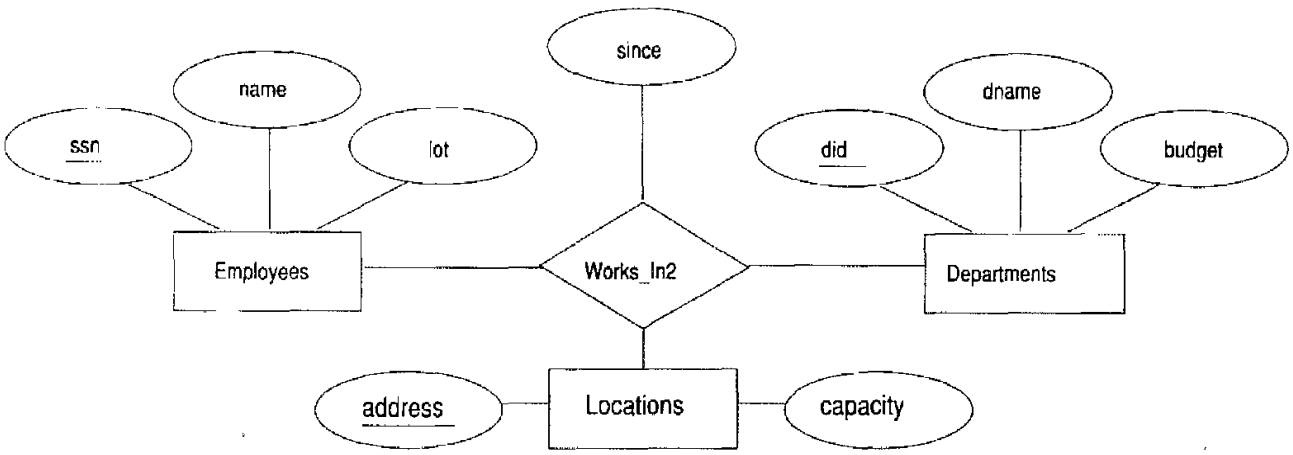
"10" "CS281" "A"
"20" "CS101" "B"
"30" "CS102" "A"
"40" "CS281" "A"

The output is the following:
Zeynep CS281
Alper CS102
~~Zeynep CS281~~ ⟶ There is a problem

27.02.2015

ER Model ⟶ Relational Model

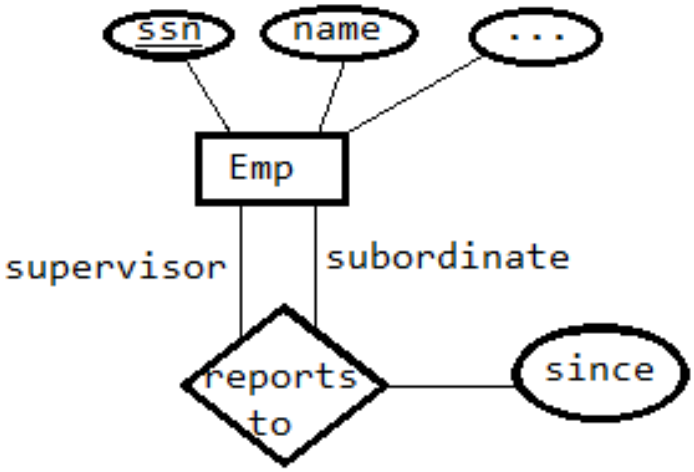
1. Entity set to tables
2. Relationship sets (without constraints) to tables



WorksIn (ssn, did, address, since)
ssn, did and address are foreign keys.

Referential Integrity: If a ssn appears in this relation, there must be an employee with the same ssn in Employee relation. A condition like this is referred to as "referential integrity."

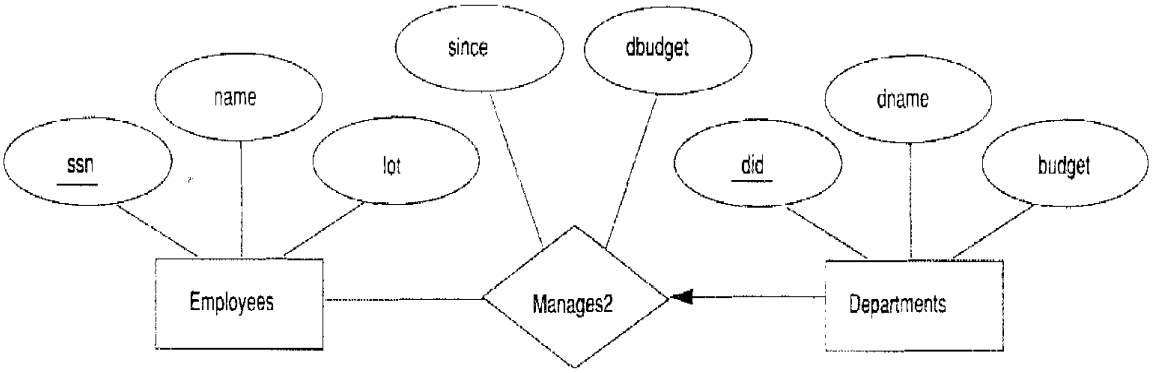
```
CREATE TABLE WorksIn(ssn CHAR(11),
                     did INTEGER,
                     address CHAR(20),
                     since DATE,
                     PRIMARY KEY (ssn, did, address),
                     FOREIGN KEY (ssn) REFERENCES Employee,
                     FOREIGN KEY (address) REFERENCES Location,
                     FOREIGN KEY (did) REFERENCES Department)
```



Reports_To (supervisor_ssn, subordinate_ssn, since)

```
CREATE TABLE Reports_To (
                     supervisor_ssn CHAR (11),
                     subordinate_ssn CHAR (11) ,
                     PRIMARY KEY (supervisor_ssn, subordinate_ssn, since DATE),
                     FOREIGN KEY (supervisor_ssn) REFERENCES Employees(ssn),
                     FOREIGN KEY (subordinate_ssn) REFERENCES Employees(ssn) )
```

3. Translating Relationship Sets with Key Constraints



Key Constraint: For a department there can be only one manager.

Manages (ssn, did, since)

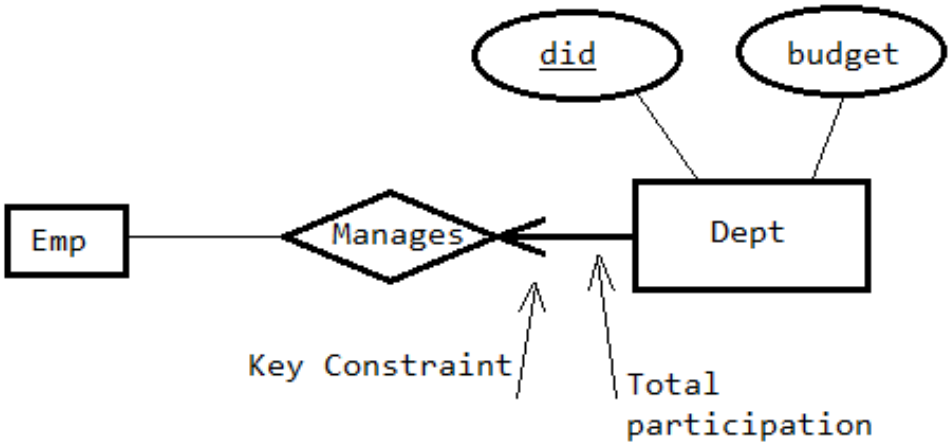
↓

Manages (<u>did</u>, ssn, since)

	d10	s1	2000
X	d10	s2	2010

Other way of implementing the same relationship is to move all attributes of Manages to Departments. Hence, there will be no table for Manages.

Dept (<u>did</u>, dname, budget, supervisor_ssn, since)



Dept (<u>did</u>, supervisor_ssn, budget)

d20 ~~NULL~~

Manages (<u>did</u>, supervisor_ssn, …)

d20 s20

Emp (<u>ssn</u>, …)

s20 …

```
CREATE TABLE Manages
        ( did INTEGER,
        dname CHAR(20) ,
        budget REAL,
        supervisor_ssn CHAR(11) NOT NULL,
        since DATE,
        PRIMARY KEY (did),
        FOREIGN KEY (supervisor_ssn) REFERENCES Employees (ssn)
        ON DELETE NO ACTION)
```

It includes the constraint that every department must have a manager: Because ssn cannot take on null values. The NO ACTION specification ensures that an Employees tuple cannot be deleted while it is pointed to by a Manages tuple.

Since we use NOT NULL we prefer to write the following:

ON DELETE SET DEFAULT (9999)

When an employee which is the manager of a department is deleted the supervisor_ssn for the department he/she manages will be set to 9999.

CASCATE
Used for implementing weak entities.

CHAPTER 3
THE RELATİONAL MODEL
(Slide is available on the web)

Attributes get their values from a domain.
Domain: Set of all possible values for an attribute.