

# Relational Model 1: Tables and Keys

Gary KL Tam

Department of Computer Science  
Swansea University

The **relational model** is the *de facto* standard implemented in all the major database systems. It defines:

- 1 the format by which data should be stored;
- 2 the operations for querying the data.

We will focus on the first aspect in this lecture, leaving the second aspect to the next lecture.

A database conforming to the relational model is called a **relational database**.

# Table, a.k.a. Relation

In a relational database, data are stored in **tables**.

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000
		...		

- Each row is also called a **tuple**.
- Each column is also called an **attribute**.
- The **relation schema** of a table is the set of its attribute names.
  - E.g., the schema of the above table is {pid, name, dept, rank, sal}.

# Schema Examples

PROF

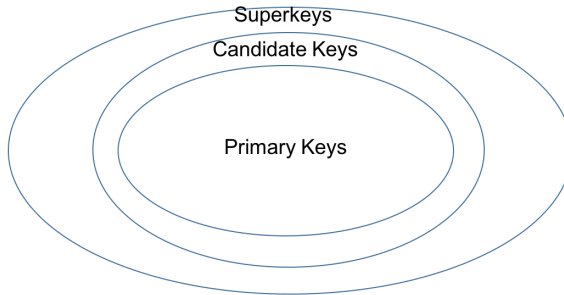
pid	name	dept	rank	sal
<i>p1</i>	Adam	CS	asst	6000
<i>p2</i>	Bob	EE	asso	8000
<i>p3</i>	Calvin	CS	full	10000
<i>p4</i>	Dorothy	EE	asst	5000
<i>p5</i>	Emily	EE	asso	8500
<i>p6</i>	Frank	CS	full	9000

CLASS

cid	title	dept	year
<i>c1</i>	database	CS	2011
<i>c2</i>	signal processing	EE	2012
<i>c1</i>	database	CS	2012

- optionally, attributes have domains; like types in a programming language
- A relation is a set of tuples, which means:
  - there can be no duplicate tuples (but in practise, commercial DBMSs allow duplicate rows)
  - order of the tuples doesn't matter
  - order of the attributes doesn't matter

- key: a set of attributes for which no two rows can have the same values
- Superkey
- Candidate key
- Primary key
- Foreign key



# Superkey

## Definition

**Superkey:** a set of one or more attributes whose combined values are unique. I.e., no two tuples can have the same values on these attributes.

- There may be more than one superkey.
- "Superkey" because it is a superset of some key.

## Example

- In the PROF table (pid, name, dept, rank, sal) in Slide 3, {pid} is a key. Hence, all the following are super keys:
- {pid}, {pid, name}, {pid, dept}, {pid, rank, sal}, ...

# Example

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

Consider these superkeys and a new tuple to add to PROF

- {pid} - p1, Adam, CS, asso, 7000 (conflict with p1)
- {name, dept} - p6, Dorothy, EE, asst, 7000 (conflict with p4)
- {pid, sal} - p5, Emily, EE, asst, 8500 (conflict with p5)
- {pid, sal} - p1, Markus, EE, asst, 8500 (OKAY!)



# Candidate Key

## Definition

In a table, a **candidate key** is a minimal set  $K$  of attributes such that no two tuples are allowed to be equivalent on all the attributes in  $K$ .

E.g., in the PROF table of the previous slide, if we set  $\{\text{pid}\}$  as a candidate key, then no two tuples can have the same pid.

- A candidate key is designated when the table is created.
- There can be multiple candidate keys.
  - E.g., if you want, you can specify  $\{\text{name}\}$  as another candidate key, but do you think it makes sense?
  - How about  $\{\text{dept}, \text{rank}\}$ ?

CLASS

<b>cid</b>	<b>title</b>	<b>dept</b>	<b>year</b>
c1	database	CS	2011
c2	signal processing	EE	2012
	...		

How would you set a candidate key?

- {cid}?
- {dept}?
- {title}?
- {year}?
- {title, year}?
- {cid, dept}?
- {cid, year}?

## CLASS

<b>cid</b>	<b>title</b>	<b>dept</b>	<b>year</b>
c1	database	CS	2011
c2	signal processing	EE	2012
	...		

### Note

A university stores the whole history of its modules in the CLASS database/table.

Example cid(s): CS-250, EE-301, MA-101

# Why important?

Can I set the candidate key  $\{ \text{cid}, \text{dept}, \text{year} \}$  ?

Class			
cid	title	dept	year
CS250	database	CS	2011
EE101	signal processing	EE	2012
	...		

# Why important?

Can I set the candidate key  $\{ \text{cid}, \text{dept}, \text{year} \}$  ?

Class			
cid	title	dept	year
CS250	database	CS	2011
EE101	signal processing	EE	2012
CS300	Project Management	CS	2013
CS300	Project Management	EE	2013

Consider these? What do you see?

As a good practice, every table should have at least a candidate key, a convention that will be enforced in the rest of the course. This implies that no two tuples in the table can be entirely equivalent to each other (think: why?).

## Choosing a key

- Need to know not just that there are no duplicates now, but that there in principle cannot be any.
- Is every relation guaranteed to have a key?
- Often we have to invent an artificial new attribute to ensure all tuples will be unique. Eg, NI number, ISBN number.
- A key is a kind of integrity constraint.

### Primary Key

Primary key - one of the candidate keys chosen by DB designer.

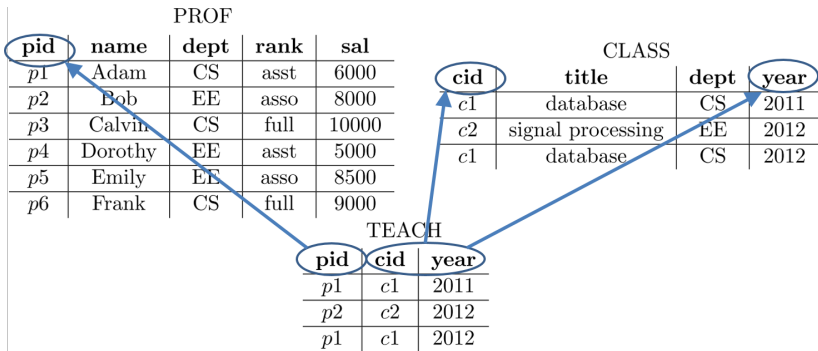
# Foreign Key

## Definition

Let  $T$  and  $T'$  be two tables, and  $K$  a candidate key in  $T$ . If  $T'$  also contains  $K$ , then  $K$  is a **foreign key** of  $T'$  **referencing**  $T$ .

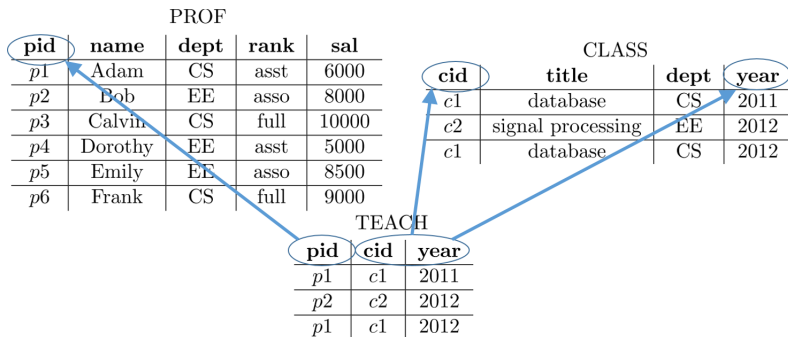
See the next slide for an example.





Suppose that PROF has a candidate key pid, and CLASS has a candidate key cid, year. Then

- {pid}: is a foreign key of TEACH referencing PROF.
- {cid, year}: is a foreign key of TEACH referencing CLASS.



These violate referential constraint (foreign key) in TEACH:

- <p10, c1, 2012> - p10 does not exist in PROF
- <p2, c3, 2016> - c3, 2016 does not exist in CLASS

# Discussion

PROF

<b>pid</b>	<b>name</b>	<b>dept</b>	<b>rank</b>	<b>sal</b>
<i>p1</i>	Adam	CS	asst	6000
<i>p2</i>	Bob	EE	asso	8000
<i>p3</i>	Calvin	CS	full	10000
<i>p4</i>	Dorothy	EE	asst	5000
<i>p5</i>	Emily	EE	asso	8500
<i>p6</i>	Frank	CS	full	9000
		...		

CLASS

<b>cid</b>	<b>title</b>	<b>dept</b>	<b>year</b>
<i>c1</i>	database	CS	2011
<i>c2</i>	signal processing	EE	2012
<i>c1</i>	database	CS	2012
	...		

TEACH

<b>pid</b>	<b>cid</b>	<b>year</b>
<i>p1</i>	<i>c1</i>	2011
<i>p2</i>	<i>c2</i>	2012
<i>p1</i>	<i>c1</i>	2012
	...	

How would you designate a candidate key for TEACH?

# Roadmap - next couple of lectures

- We will learn how to define a database's structure and write queries on it.
- We will learn these first in the relational model, then in SQL because:
  - the relational model and algebra are the foundation for SQL
  - other important concepts, like query optimization, are defined in terms of relational algebra

## FAQ

So far, we seem to pick the keys in a very ad-hoc way (by guessing/try-and-error). We will cover more theories (esp. functional dependencies in week7). After that, we will formally introduce "HOW" to define candidate keys properly. Stay tuned.

For today, make sure you understand:

- Definitions of: superkeys, candidate keys, primary keys, foreign keys
- The meaning of keys
- The use of keys