Database Management Systems

Madhavan Mukund

https://www.cmi.ac.in/~madhavan

Sai University

Lecture 10, 20 September 2023

Relational database design

- Set of attributes that one needs to keep track of

Relational database design

- Set of attributes that one needs to keep track of
- Why not combine into a single table?

Relational database design

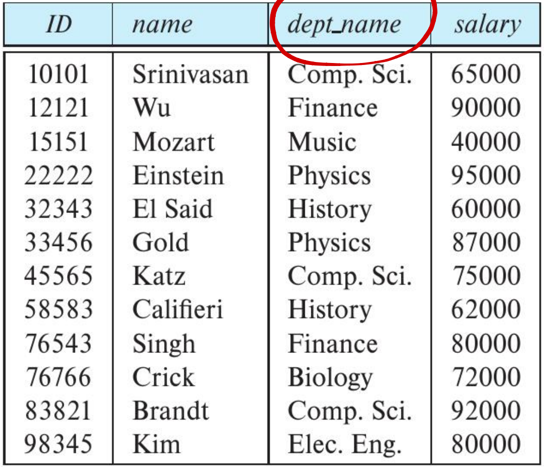
ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

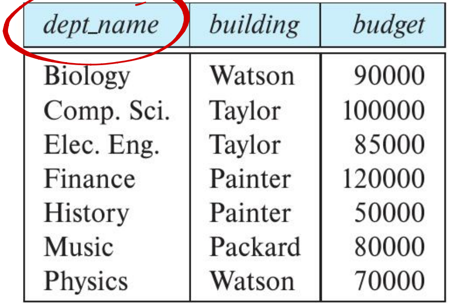
ID	Name	dept

ID	salary

Relational database design



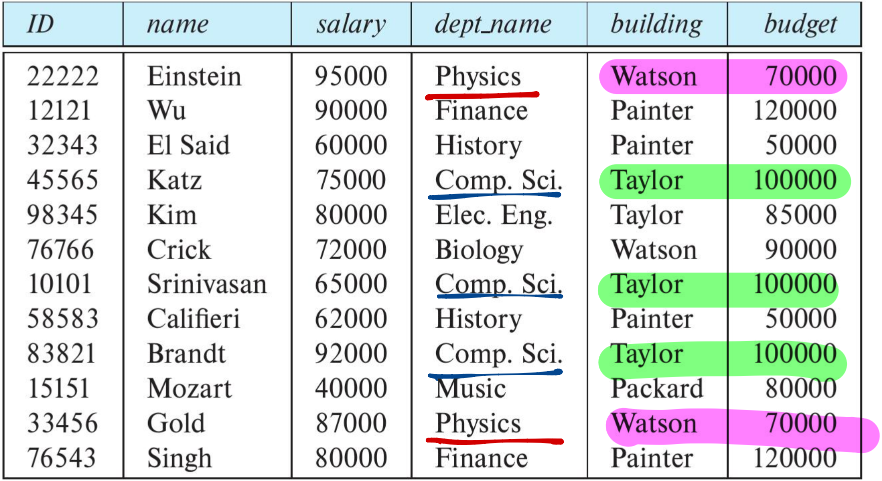
ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000



dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

- Combine these into a single table?

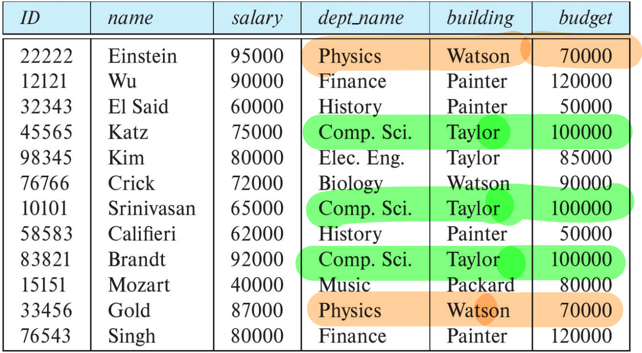
Relational database design



ID	*name*	*salary*	*dept_name*	*building*	*budget*
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

Relational database design

- Redundant storage



ID	*name*	*salary*	*dept_name*	*building*	*budget*
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

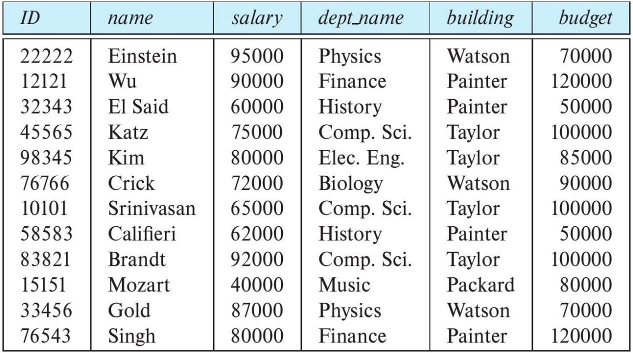
Relational database design

- Redundant storage
- Maintaining consistency
  - Updates
  - Inserts and deletes

Add a Biotech dept

Yet to recruit faculty

Add a row with null value for faculty data



ID	*name*	*salary*	*dept_name*	*building*	*budget*
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

Decomposition and information

- `(customer_name,regd_phone,regd_email)`

Decomposition and information

- (customer_name,regd_phone,regd_email)
- Decompose as (customer_name,regd_phone) and (customer_name,regd_email)

Natural join

Name is not unique

Ajay	N1	E1
Ajay	N2	E2

Ajay	N1
Ajay	N2

Ajay	E1
Ajay	E2

	N1	E2
	N2	E1

Decomposition and information

- (customer_name,regd_phone,regd_email)
- Decompose as (customer_name,regd_phone) and (customer_name,regd_email)
- Name is not unique — loss of information

Decomposition and information

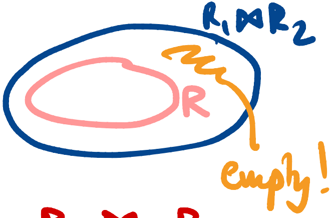
- (customer_name,regd_phone,regd_email)
- Decompose as (customer_name,regd_phone) and (customer_name,regd_email)
- Name is not unique — loss of information
- Recombining decomposed relation should not add tuples

|

Natural join

Decomposition and information

- `(customer_name,regd_phone,regd_email)`
- Decompose as `(customer_name,regd_phone)` and `(customer_name,regd_email)`
- Name is not unique — loss of information
- Recombining decomposed relation should not add tuples
- Lossless decomposition
  - Decompose R as R_1 and R_2
  - Want $R = R_1 \bowtie R_2$



<u>Clearly</u> $R \subseteq R_1 \bowtie R_2$

Problem is if $R_1 \bowtie R_2$ has rows not in R

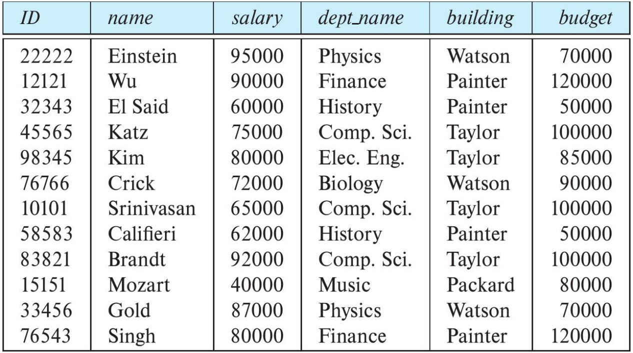
Functional dependencies

- $A_1, A_2, \ldots, A_k \rightarrow B_1, B_2, \ldots B_m$
  - LHS atributes uniquely fix RHS attributes
  - Must hold for every instance — semantic property of attributes

ID	*name*	*salary*	*dept_name*	*building*	*budget*
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

Functional dependencies

- $A_1, A_2, \ldots, A_k \rightarrow B_1, B_2, \ldots B_m$
  - LHS atributes uniquely fix RHS attributes
  - Must hold for every instance — semantic property of attributes
- Need not correspond to superkeys
  - dept_name → building
  - dept_name → budget

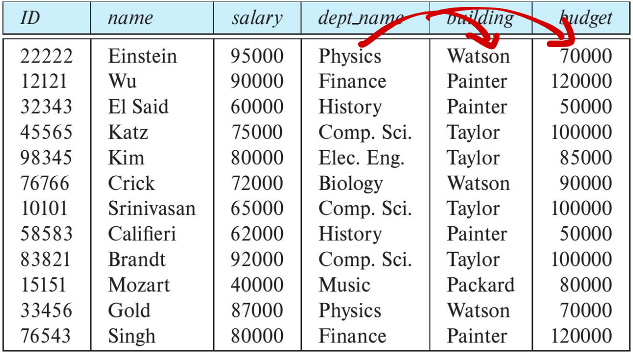


ID	*name*	*salary*	*dept_name*	*building*	*budget*
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

⇒ dept_name → building, budget

Functional dependencies

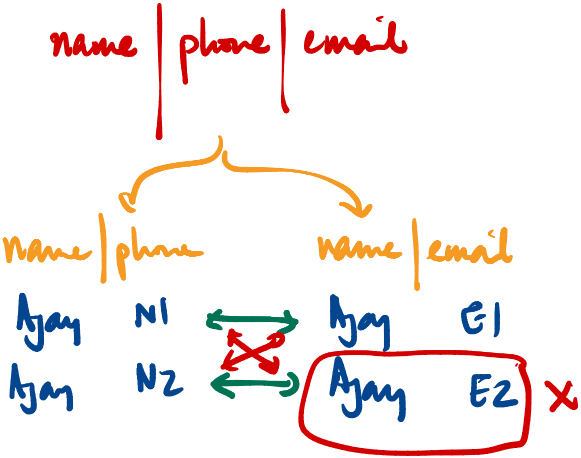
- $A_1, A_2, \ldots, A_k \rightarrow B_1, B_2, \ldots B_m$
  - LHS atributes uniquely fix RHS attributes
  - Must hold for every instance — semantic property of attributes
- Need not correspond to superkeys
  - `dept_name` → `building`
  - `dept_name` → `budget`
- Use to identify sources of redundancy, guide decomposition



ID	*name*	*salary*	*dept_name*	*building*	*budget*
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

Lossless decomposition and functional dependencies

- Decompose R as R_1 and R_2

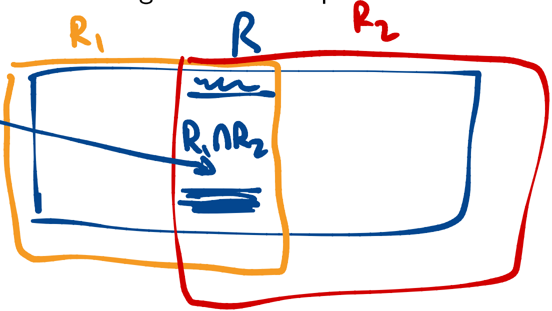


Lossless decomposition and functional dependencies

- Decompose R as R_1 and R_2

- Decomposition is lossless if at least one of the following functional dependencies hold
  - $R_1 \cap R_2 \rightarrow R_1$
  - $R_1 \cap R_2 \rightarrow R_2$

property of data as a whole



Lossless decomposition and functional dependencies

- Decompose R as R_1 and R_2
- Decomposition is lossless if at least one of the following functional dependencies hold
  - $R_1 \cap R_2 \rightarrow R_1$
  - $R_1 \cap R_2 \rightarrow R_2$
- Decompose Instructor-Department as Instructor and Department
  - Instructor ∩ Department is dept_name
  - dept_name is primary key for Department

ID Name dept Salary × $R_1 \cap R_2 \rightarrow R_1$

Dept Bldg Budget ✓

$R_1 \cap R_2 \rightarrow R_2$

Lossless decomposition and functional dependencies

- Decompose R as R_1 and R_2
- Decomposition is lossless if at least one of the following functional dependencies hold
  - $R_1 \cap R_2 \rightarrow R_1$
  - $R_1 \cap R_2 \rightarrow R_2$
- Decompose `Instructor-Department` as `Instructor` and `Department`
  - `Instructor` ∩ `Department` is `dept_name`
  - `dept_name` is primary key for `Department`
- In general need to compute all implied dependencies
  - From $A \rightarrow B$ and $B \rightarrow C$, conclude that $A \rightarrow C$
- Closure of a set of dependencies F — denoted F^+

Computing the closure of a set of attributes

- Given $A_1, A_2, \ldots, A_2$ k and B, does $A_1, A_2, \ldots, A_k \to B$?

F^+

Does $R_1 \cap R_2 \to R_1$?

$A_1 \text{--} A_k \to B_1 \text{--} B_m$

suff to show

$A_1 - A_2 \; k \to B_1$

$A_1 \ldots A_k \to B_2$

$\vdots$

$A_1 — A_k \to B_m$

Computing the closure of a set of attributes

- Given $A_1, A_2, \ldots, A_2$ and B, does $A_1, A_2, \ldots, A_k \rightarrow B$?
- Iterative algorithm

Compute $\mathcal{C}$, the set of attributes "fixed" by $A_1 \ldots, A_k$

Check if B is in $\mathcal{C}$

$A_1 .. A_k$ fix $A_1 -- A_k$ Start with $\mathcal{C} = \{A_1, \ldots, A_k\}$

Check a rule $D_1 .., D_m \rightarrow E_1 .. E_n$ s.t $D_1 .., D_m \subseteq \mathcal{C}$

$\Rightarrow$ Add $E_1 .. E_n$ to $\mathcal{C}$

Stop when nothing new is a

Normal forms

- Criteria to determine if the collection of tables is “good”

Normal forms

- Criteria to determine if the collection of tables is “good”
- Normalization — decompose tables till they achieve a normal form

Normal forms

- Criteria to determine if the collection of tables is “good”
- Normalization — decompose tables till they achieve a normal form
- Guided by functional dependencies

Boyce-Codd Normal Form (BCNF)

- Relational schema R, set of functional dependencies F

Boyce-Codd Normal Form (BCNF)

- Relational schema R, set of functional dependencies F
- Write α, β to represent sequences of attributes $A_1, A_2, \ldots, A_k$, $B_1, B_2, \ldots, B_m$

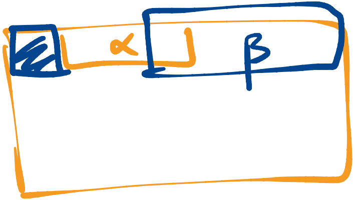
Boyce-Codd Normal Form (BCNF)

- Relational schema R, set of functional dependencies F
- Write α, β to represent sequences of attributes $A_1, A_2, \ldots, A_k$, $B_1, B_2, \ldots, B_m$
- R is in BCNF if, for every $\alpha \rightarrow \beta \in F^+$, one of the following holds
  - $\alpha \rightarrow \beta$ is trivial (i.e., $\beta \subseteq \alpha$)
  - α is a superkey for R

α, β are sets of attributes in R

$A_1, A_2 \ldots A_k \rightarrow A_i$

always



Boyce-Codd Normal Form (BCNF)

- Relational schema R, set of functional dependencies F
- Write α, β to represent sequences of attributes $A_1, A_2, \ldots, A_k$, $B_1, B_2, \ldots, B_m$
- R is in BCNF if, for every $\alpha \rightarrow \beta \in F^+$, one of the following holds
  - $\alpha \rightarrow \beta$ is trivial (i.e., $\beta \subseteq \alpha$
  - α is a superkey for R
- `InstructorDepartment(ID,name,salary,dept_name,building,budget)` not in BCNF

not fixed

Boyce-Codd Normal Form (BCNF)

- Relational schema R, set of functional dependencies F
- Write α, β to represent sequences of attributes $A_1, A_2, \ldots, A_k$, $B_1, B_2, \ldots, B_m$
- R is in BCNF if, for every $\alpha \rightarrow \beta \in F^+$, one of the following holds
  - $\alpha \rightarrow \beta$ is trivial (i.e., $\beta \subseteq \alpha$
  - α is a superkey for R
- InstructorDepartment(ID,name,salary,dept_name,building,budget not in BCNF
- Instructor(ID,name,dept_name,salary) and Department(dept_name,building,budget) are in BCNF

$R_1 \cap R_2 = \text{dept\_name}$

Achieving BCNF

- $\alpha \rightarrow \beta \in F^{+}$ is a BCNF violation for R if neither of the following holds
  - $\alpha \rightarrow \beta$ is trivial (i.e., $\beta \subseteq \alpha$)
  - α is a superkey for R

Achieving BCNF

- $\alpha \rightarrow \beta \in F^+$ is a BCNF violation for R if neither of the following holds
  - $\alpha \rightarrow \beta$ is trivial (i.e., $\beta \subseteq \alpha$)
  - α is a superkey for R
- To fix this, decompose R as
  - $\alpha \cup \beta$
  - $R \setminus (\beta \setminus \alpha)$

Essentially $R \setminus \beta$

dept → building, budget

$\alpha \rightarrow \beta$

↓

separate table

Achieving BCNF

- $\alpha \to \beta \in F^+$ is a BCNF violation for R if neither of the following holds
  - $\alpha \to \beta$ is trivial (i.e., $\beta \subseteq \alpha$
  - α is a superkey for R
- To fix this, decompose R as
  - R_1 $\alpha \cup \beta$ → guarantees $R_1 \cap R_2 \to R_1$ — lossless
  - R_2 $R \setminus (\beta \setminus \alpha)$
- Example: `dept_name` → `building,budget` is a BCNF violation for

$$\texttt{InstructorDepartment}(\underbrace{\texttt{ID,name,salary,}\underbrace{\texttt{dept\_name}}_{\alpha}}_{R-(\overset{\beta}{\overset{\shortparallel}{\beta-\alpha}})}\texttt{,}\underbrace{\texttt{building,budget}}_{\beta})$$

$\alpha \cup \beta$

Achieving BCNF

- $\alpha \to \beta \in F^+$ is a BCNF violation for R if neither of the following holds
  - $\alpha \to \beta$ is trivial (i.e., $\beta \subseteq \alpha$
  - α is a superkey for R
- To fix this, decompose R as
  - $\alpha \cup \beta$
  - $R \setminus (\beta \setminus \alpha)$
- Example: `dept_name` → `building,budget` is a BCNF violation for `InstructorDepartment(ID,name,salary,dept_name,building,budget`
- Decompose as

  α β
  - `Department(dept_name,building,budget)`
  - `Instructor(ID,name,dept_name,salary)`

  R - β

Dependency preservation

- `Advisor(student_id,faculty_id,dept_name)`
- Each faculty member is in only one department
- Students can be across multiple departments
- Each student has at most one advisor in each department

Dependency preservation

- `Advisor(student_id,faculty_id,dept_name)`
- Each faculty member is in only one department
- Students can be across multiple departments
- Each student has at most one advisor in each department
- BCNF decomposition is `(student_id,faculty_id)`, `(faculty_id,dept_name)`

F: fac_id → dept

st_id, dept → fac_id

violation, fac_id is not a key

$R - \beta$

$\alpha \cup \beta$

Dependency preservation

- Advisor(student_id,faculty_id,dept_name)
- Each faculty member is in only one department
- Students can be across multiple departments
- Each student has at most one advisor in each department
- BCNF decomposition is (student_id,faculty_id), (faculty_id,dept_name)
- Functional dependencies
  - faculty_id → dept_name
  - student_id,dept_name → faculty_id

Dependency preservation

- `Advisor(student_id,faculty_id,dept_name)`
- Each faculty member is in only one department
- Students can be across multiple departments
- Each student has at most one advisor in each department
- BCNF decomposition is `(student_id,faculty_id)`, `(faculty_id,dept_name)`
- Functional dependencies
  - `faculty_id` → `dept_name`
  - `student_id,dept_name` → `faculty_id`
- Need join to check second dependency

Third normal form (3NF)

- R is in 3NF if, for every $\alpha \rightarrow \beta \in F^+$, one of the following holds

  BCNF
  - ✓ $\alpha \rightarrow \beta$ is trivial (i.e., $\beta \subseteq \alpha$)
  - ✓ α is a superkey for R
  - Each attribute A in $\beta \setminus \alpha$ is contained in some candidate key for R Mysterious!

Third normal form (3NF)

- R is in 3NF if, for every $\alpha \to \beta \in F^+$, one of the following holds
  - $\alpha \to \beta$ is trivial (i.e., $\beta \subseteq \alpha$
  - α is a superkey for R
  - Each attribute A in $\beta \setminus \alpha$ is contained in some candidate key for R
- BCNF is a stricter condition than 3NF

BCNF

$\alpha \to \beta$

Adv(stud, fac, dept)

stud, dept is a key

Third normal form (3NF)

- R is in 3NF if, for every $\alpha \rightarrow \beta \in F^+$, one of the following holds
  - $\alpha \rightarrow \beta$ is trivial (i.e., $\beta \subseteq \alpha$
  - α is a superkey for R
  - Each attribute A in $\beta \setminus \alpha$ is contained in some candidate key for R
- BCNF is a stricter condition than 3NF
- Priorities
  - Lossless decomposition Not negotiable
  - BCNF
  - Dependency preservation