Normalization 3: Third Normal Form

Review

- Redundancy in tables causes problems
- Most redundancy can be traced to the presence of functional dependencies
- A schema in Boyce Codd Normal Form (BCNF) has no such redundancy
- Splitting a table can rid it of redundancy – but what is the general method?

When we "split" a table, we will always use the projection operation of relational algebra.

Rule for splitting to achieve BCNF

ID	name	office
gill1992	Harman	DH282
gill1992	Harman	MH999
benr9431	Ben	MH213

We are not in BCNF because:

- ID → name is a non-trivial functional dependency
- ID is not a superkey

Split table into two, using these attribute sets:

1. {ID, name} (attributes in the FD)
2. {office, ID} (the "other attributes", plus LHS of FD)

(LHS = "left hand side")

Applying the rule

ID	name	office
gill1992	Harman	DH282
gill1992	Harman	MH999
benr9431	Ben	MH213

FD is
ID → name

ID	name
gill1992	Harman
benr9431	Ben

ID	office
gill1992	DH282
gill1992	MH999
benr9431	MH213

The decomposition is "lossless"! ☺

The rule for BCNF decomposition

Let R be a relational schema that is not in BCNF. Then there exists a non-trivial FD $X \rightarrow Y$ such that X is not a superkey for R.

Replace R with two schemas, having these attributes:

1. $X \cup Y$ // the attributes of the FD
2. R – $(Y - X)$ // the "other" attributes, plus X

- The rule guarantees a lossless decomposition
- The two new schemas share only the attributes in X
- The new schemas may not be in BCNF!

  … so may need to apply the rule repeatedly

A drawback of BCNF

ID	name	office
gill1992	Harman	DH282
gill1992	Harman	MH999
benr9431	Ben	MH213

FDs:
ID, office → name
ID → name

{ID, name}

ID	name
gill1992	Harman
benr9431	Ben

{office, ID}

ID	office
gill1992	DH282
gill1992	MH999
benr9431	MH213

- The constraint ID, office → name is now a constraint **across** tables.
- So this decomposition is not "dependency preserving".

Third Normal Form (3NF)

A relation schema R is in **Third Normal Form** if:

- whenever there's a (non-trivial) functional dependency $X \rightarrow Y$ for R
- then X is a superkey for R, **or**
- every attribute in $Y - X$ is contained in a candidate key for R

3NF Example

dept_advisor

student_ID	dept_name	inst_ID
1	Biology	10
1	Chemistry	20
2	Biology	10

FD: inst_ID → dept_name

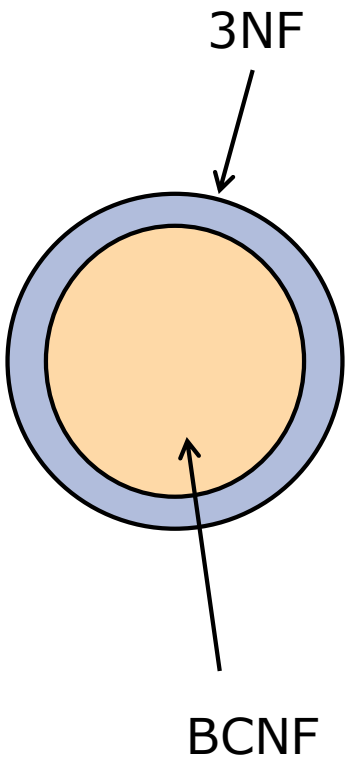
This schema is not in BCNF: why?

It is in 3NF, because dept_name is a prime attribute
(it belongs to a candidate key of R)

A delete anomaly: if we want to modify the table to show that student 1 is no longer a Chemistry student, we lose that instructor 20 is a Chemistry adviser.

BCNF vs. 3NF

	BCNF decomposition	3NF decomposition
eliminates all FD redundancy?	✓	
lossless decomposition?	✓	✓
dependency preservation?		✓



So sometimes 3NF is preferred to BCNF

Summary

Normalization

- reduces redundancy
- but means queries may need joins (performance hit)

Schemas derived from ER diagrams are often already in BCNF.

FDs capture a common kind of redundancy