

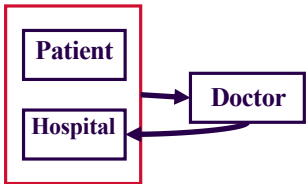
Types of Functional Dependencies...

- Transitive dependency: $X \rightarrow Y$ is transitive in R if there is a set of attributes Z that is not a subset of any key of R and both $X \rightarrow Z$ and $Z \rightarrow Y$ hold
 - E.g., EMP (SSN, EName, DeptID, MGRSSN)
 - (fd.1) $SSN \rightarrow DeptID$
 - (fd.2) $DeptID \rightarrow MGRSSN$
 - (from fd.1 & fd.2) $SSN \rightarrow MGRSSN$
- Multivalued dependency: $X \twoheadrightarrow Y$ is multivalued dependency in R if X is a key, Z in R and $Z \twoheadrightarrow Y$
 - E.g., DJP (DeptID, ProjectID, part)
 - (fd.1) $DeptID \rightarrow part$
 - (fd.2) $ProjectID \rightarrow part$

Normal Forms

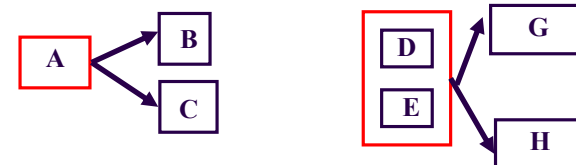
- 1NF: First Normal Form
Every attribute has a single atomic value.
- 2NF: Second Normal Form
It is in 1NF and does not have partial dependencies.
 - Counter Example: 1NF but not 2NF
SUPPLY (SID, PID, DID, SCity, DCity, Qty)
- 3NF: Third Normal Form
It is in 2NF and does not have transitive dependencies to attributes that are not part of a key.
 - If $X \rightarrow A$ is an FD then (a) it is trivial, or (b) X is a superKey, or (c) A is a subset of a **candidate** Key.
 - Counter Example: 2NF but not 3NF
EMP (SSN, EName, Bdate, Salary, DID, DName, MGRSSN)

Normal Forms - 3NF

- Example:
PATIENT-VISIT (PATIENT, HOSPITAL, DOCTOR)
 - fd.1 (Patient, Hospital) \rightarrow Doctor
 - fd.2 Doctor \rightarrow Hospital
- Pictorially:
 
- *3NF may not be adequate. Why??*
- May force NULL values to attributes of the primary key.

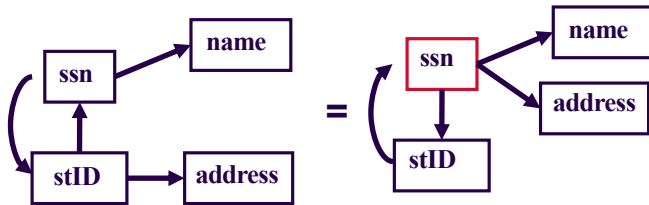
Normal Forms - BCNF

- Boyce-Codd Normal Form: A relation is in 3NF and has not transitive dependencies
 - if $X \rightarrow A$ is an FD, then
 - (a) it is trivial, or (b) X is a superKey.
 - Informally: everything depends on the **full key**, and nothing but the key
- Pictorially: we want a 'star' shape



Normal Forms - BCNF

- or a star-like: (e.g., 2 cand. keys):
STUDENT(ssn, stID, name, address)



Normal Forms - BCNF

- Theorem:
given a schema R and a set of FD 'F', we can always decompose it to schemas R1, ... Rn, so that
 - R1, ..., Rn are in BCNF and
 - the decompositions is lossless
- But, some decompositions might lose dependencies \Rightarrow use 3NF
 - 3NF always loseless
 - 3NF always preserves dependencies

BCNF & Dependency Preservation

- BCNF is not always dependency preserving
- Example: 3NF but not BCNF
PATIENT-VISIT (PATIENT, HOSPITAL, DOCTOR)
fd.1 (Patient, Hospital) \rightarrow Doctor
fd.2 Doctor \rightarrow Hospital
- Possible Decomposition 1:
 - Doctor-Hospital (Doctor, Hospital) {Doctor \rightarrow Hospital}
 - Patient-Doctor (Patient, Doctor) {Patient \rightarrow Doctor}
- Possible Decomposition 2:
 - Doctor-Hospital (Doctor, Hospital) {Doctor \rightarrow Hospital}
 - Patient-Doctor (Patient, Hospital) {Patient \rightarrow Hospital}
- BUT these decompositions lose fd.1

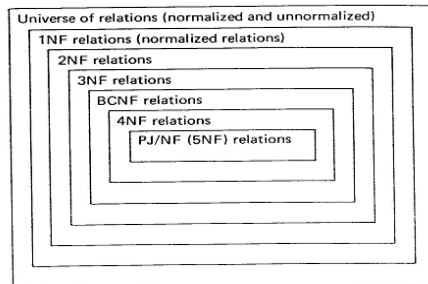
Normal Forms - 4NF

- Fourth Normal Form:** A relation is in BCNF and has no Multivalued Dependencies
- Example: (**FACULTY, Dept, Committee**)
 - A faculty member can belong to more than one dept.
 - A faculty can be on several college-wide committees.
 - There is no relation between dept. and committee.
- Anomalies? Change F101 from Budget to Admissions

FacultyID	Dept	Committee
F101	CS	Budget
F101	CoE	Budget
F101	CS	Curriculum
F101	CoE	Curriculum
F221	Bio	Library
F330	Math	Budget
F330	Math	Admissions

More Normal Forms...

- ❑ 5NF: Fifth Normal Form
 - No Join Dependencies
- ❑ 6NF:



Universal Relational Approach

- ❑ One single, large table
- ❑ Simple ?
- ❑ Good ? or Bad? Or just Ugly?
- ❑ Normalize it!

The Normalization Process

- ❑ The process of finding good (stable) set of relations that is a faithful model of the enterprise.
- ❑ **Decomposition (top-down process)**
 - start with a universal relation
 - identify functional dependencies
 - identify key(s)
 - If necessary use decomposition to split the universal relation into a set of relations

Deriving the Keys of R from its FDs

- ❑ Let $X \subseteq R$, X^+ is its closure
- ❑ Algorithm for finding the keys
 - ♦ $X^+ = X$;
 - ♦ If \exists an used $U \rightarrow V$ in FDs and $U \subseteq X^+$ then goto step 3; else STOP;
 - ♦ $X^+ = X^+ \cup V$;
 - ♦ If $X^+ = R$ then STOP; else goto step 2;
- ❑ A primary key is one of the minimal keys

Universal Relation: Example of BCNF Decomposition

- $R = (\text{branch_name}, \text{branch_city}, \text{assets}, \text{customer}, \text{loan_number}, \text{amount})$
 $F = \{\text{branch_name} \rightarrow \text{assets branch_city}, \text{loan_number} \rightarrow \text{amount branch_name}\}$
Key = $\{\text{loan_number}, \text{customer}\}$
- Decomposition
 - $R_1 = (\text{branch_name}, \text{assets}, \text{branch_city})$
 - $R_2 = (\text{branch_name}, \text{customer}, \text{loan_number}, \text{amount})$
 - $R_3 = (\text{branch_name}, \text{loan_number}, \text{amount})$
 - $R_4 = (\text{customer}, \text{loan_number})$
- Final decomposition: R_1, R_3, R_4

BCNF Decomposition Algorithm

```
result := {R};
done := false;
compute F+;
while (not done) do
  if (there is a schema  $R_i$  in result that is not in BCNF)
    then begin
      let  $\alpha \rightarrow \beta$  be a nontrivial FD that holds on  $R_i$ 
      such that  $\alpha \rightarrow R_i$  is not in  $F^+$ , and  $\alpha \cap \beta = \emptyset$ ;
      result := (result -  $R_i$ )  $\cup$  ( $R_i - \beta$ )  $\cup$  ( $\alpha, \beta$ );
    end
  else done := true;
```

Note: each R_i is in BCNF, and decomposition is lossless-join.

Synthesis (bottom-up process)

- Begin with attributes and combine them into related group using functional dependencies to develop a set of normalized relations.
- A synthesis algorithm was developed by Bernstein.
- Basic steps:
 - make a list of all FDs
 - groups together those with the same determinant
 - construct a relation of each group.

Synthesis... Elaborated steps

1. Make a list of all FDs.
2. Eliminate extraneous attributes in each FD.
3. Remove any redundant FDs and find a non redundant covering of the input FDs.
 - Combine FD groups with equivalent key.
4. Group together those with the same determinant.
5. Construct a relation for each group.
6. If none of the resulting relations has the same key as one of the keys of the original universal relation, we add another relation containing one of the keys, with an empty FD set.

Example: Consider the following set of FDs.

f1: A \longrightarrow B
f2: A \longrightarrow C
f3: B \longrightarrow C
f4: B \longrightarrow D
f5: D \longrightarrow B
f6: ABC \longrightarrow F

Q: Using Synthesis construct a set 3NF relations.

Example: Consider the following set of FDs.

f1: A \longrightarrow BC
f2: A \longrightarrow D
f3: B \longrightarrow C
f4: C \longrightarrow D
f5: DE \longrightarrow C
f6: BC \longrightarrow D

Q: Using Synthesis construct a set 3NF relations.

Design Goals

- Goal:
 - BCNF
 - Lossless join
 - Dependency preservation
- If we cannot achieve this, we accept one of
 - Lack of dependency preservation
 - Redundancy due to use of 3NF
- Note, SQL does not provide a direct way of specifying FDs other than superkeys.
 - Can specify FDs using assertions, but they are expensive
 - Hard to test a FD whose left hand side is not a key

Conclusions

- Disadvantages of Normal Forms
 - It is not constructive. It does not provide a way to get a good design.
 - It can only be applied after we have a schema and tell if it is good or not.
 - It provides no conceptual design. It deals directly with relations and attributes.
 - It can be applied (more or less) only to relational schemas.
- Schema Design: Conceptual Design
 - Entity-Relationship (E-R) or UML
- In practice, E-R diagrams usually lead to tables in BCNF