Lecture 11

Functional Dependencies and Schema Normalization
Database Design and Normalization

- Database design consists in grouping attributes that form “good” relation schemas.
- Normalization: a design process to reduce redundancies and update anomalies in a relational schema.
  - Result: a set of decomposed relations that meet certain normal form tests
  - Four most commonly used normal forms are first (1NF), second (2NF) and third (3NF) normal forms, and Boyce–Codd normal form (BCNF)
  - A process that is based on keys and functional dependencies among the attributes of a relation
Data Redundancy

- Grouping attributes into relation schemas has an effect on storage space.
- A bad grouping may produce data redundancy.
- Problems associated with data redundancy are illustrated by comparing the following Staff and Branch relations with the StaffBranch relation.
  - StaffBranch is the Natural Join of Staff and Branch.
  - This relation has redundant data (Baddress).
## Data Redundancy

### Staff

<table>
<thead>
<tr>
<th>staffNo</th>
<th>sName</th>
<th>position</th>
<th>salary</th>
<th>branchNo</th>
</tr>
</thead>
<tbody>
<tr>
<td>SL21</td>
<td>John White</td>
<td>Manager</td>
<td>30000</td>
<td>B005</td>
</tr>
<tr>
<td>SG37</td>
<td>Ann Beech</td>
<td>Assistant</td>
<td>12000</td>
<td>B003</td>
</tr>
<tr>
<td>SG14</td>
<td>David Ford</td>
<td>Supervisor</td>
<td>18000</td>
<td>B003</td>
</tr>
<tr>
<td>SA9</td>
<td>Mary Howe</td>
<td>Assistant</td>
<td>9000</td>
<td>B007</td>
</tr>
<tr>
<td>SG5</td>
<td>Susan Brand</td>
<td>Manager</td>
<td>24000</td>
<td>B003</td>
</tr>
<tr>
<td>SL41</td>
<td>Julie Lee</td>
<td>Assistant</td>
<td>9000</td>
<td>B005</td>
</tr>
</tbody>
</table>

### Branch

<table>
<thead>
<tr>
<th>branchNo</th>
<th>bAddress</th>
</tr>
</thead>
<tbody>
<tr>
<td>B005</td>
<td>22 Deer Rd, London</td>
</tr>
<tr>
<td>B007</td>
<td>16 Argyll St, Aberdeen</td>
</tr>
<tr>
<td>B003</td>
<td>163 Main St, Glasgow</td>
</tr>
</tbody>
</table>

### StaffBranch

<table>
<thead>
<tr>
<th>staffNo</th>
<th>sName</th>
<th>position</th>
<th>salary</th>
<th>branchNo</th>
<th>bAddress</th>
</tr>
</thead>
<tbody>
<tr>
<td>SL21</td>
<td>John White</td>
<td>Manager</td>
<td>30000</td>
<td>B005</td>
<td>22 Deer Rd, London</td>
</tr>
<tr>
<td>SG37</td>
<td>Ann Beech</td>
<td>Assistant</td>
<td>12000</td>
<td>B003</td>
<td>163 Main St, Glasgow</td>
</tr>
<tr>
<td>SG14</td>
<td>David Ford</td>
<td>Supervisor</td>
<td>18000</td>
<td>B003</td>
<td>163 Main St, Glasgow</td>
</tr>
<tr>
<td>SA9</td>
<td>Mary Howe</td>
<td>Assistant</td>
<td>9000</td>
<td>B007</td>
<td>16 Argyll St, Aberdeen</td>
</tr>
<tr>
<td>SG5</td>
<td>Susan Brand</td>
<td>Manager</td>
<td>24000</td>
<td>B003</td>
<td>163 Main St, Glasgow</td>
</tr>
<tr>
<td>SL41</td>
<td>Julie Lee</td>
<td>Assistant</td>
<td>9000</td>
<td>B005</td>
<td>22 Deer Rd, London</td>
</tr>
</tbody>
</table>
Update Anomalies

◆ Types of update anomalies include:

– Insertion
  » Different address for Branch with BranchNo ‘B003’

– Deletion
  » Delete tuple for StaffNo ‘SA9’ in StaffBranch
  » Loose information about Branch with BranchNo ‘B007’

– Modification.
  » Update Baddress for StaffNo ‘SG14’
  » Different address for Branch with BranchNo ‘B003’
Functional Dependency (FD)

- **Main concept associated with normalization.**
- **Functional Dependency (FD)**
  - Are *constraints* that are derived from the *meaning* and *interrelationships* of the data attributes
  - If A and B are attributes of relation R, B is functionally dependent on A (denoted $A \rightarrow B$), or A functionally determines B, if each value of A in R is associated with exactly one value of B in R.
  - Each value of A maps to a single value of B
  - A and B can be sets of attributes.
Functional Dependencies (contd)

- X \rightarrow Y holds if whenever two tuples have the same value for X, they must have the same value for Y
  - For any two tuples t1 and t2 in any relation instance r(R):
    If \ t1[X]=t2[X], then t1[Y]=t2[Y]

- X \rightarrow Y in R specifies a constraint on all relation instances r(R)

- Written as X \rightarrow Y; can be displayed graphically on a relation schema as in Figures. (denoted by the arrow: \rightarrow).

- FDs are derived from the real-world constraints on the attributes
Example - Functional Dependencies

- Social security number determines employee name
  - SSN $\rightarrow$ ENAME

- Project number determines project name and location
  - PNUMBER $\rightarrow$ \{PNAME, PLOCATION\}

- Employee ssn and project number determines the hours per week that the employee works on the project
  - \{SSN, PNUMBER\} $\rightarrow$ HOURS
Functional Dependencies (contd)

- An FD is a property of the attributes in the schema R.
- The constraint must hold on every relation instance.
- If K is a key of R, then K functionally determines all attributes in R
  - (since we never have two distinct tuples with t1[K]=t2[K])
- FDs are nontrivial. (StaffNo→StaffNo is trivial)
Possible FDs

Given TEXT we know the COURSE.

TEXT $\rightarrow$ COURSE

TEXT maps to a single value of COURSE

<table>
<thead>
<tr>
<th>Teacher</th>
<th>Course</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
<td>Data Structures</td>
<td>Bartram</td>
</tr>
<tr>
<td>Smith</td>
<td>Data Management</td>
<td>Martin</td>
</tr>
<tr>
<td>Hall</td>
<td>Compilers</td>
<td>Hoffman</td>
</tr>
<tr>
<td>Brown</td>
<td>Data Structures</td>
<td>Horowitz</td>
</tr>
</tbody>
</table>

Figure 10.7

A relation state of TEACH with a possible functional dependency TEXT $\rightarrow$ COURSE. However, TEACHER $\rightarrow$ COURSE is ruled out.
Functional Dependencies (contd)

- Complete set of functional dependencies for a given relation can be very large.

- Important to find an approach that can reduce set to a manageable size.

- Need to identify a smaller set of functional dependencies (W) for a relation
  - A bigger set of FDs, Z can be found, so that every FD in Z can be implied by W.
  - Infer new FDs using Armstrong’s inference rules.
Armstrong’s axioms

Let $X$, $Y$, and $Z$ be subsets of the attributes of relation $R$. Armstrong’s axioms are as follows:

**IR1. Reflexivity**
If $Y$ is a subset of $X$, then $X \rightarrow Y$

**IR2. Augmentation**
If $X \rightarrow Y$, then $XZ \rightarrow YZ$
(Notation $XZ$ or $X,Z$ stands for $X \cup Z$)

**IR3. Transitivity**
If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$
Inference Rules

- **Closure** of a set F of FDs is the set $F^+$ of all FDs that can be inferred from F.

- **Closure** of a set of attributes $X$ with respect to $F$ is the set $X^+$ of all attributes that are functionally determined by $X$.

- $X^+$ can be calculated by repeatedly applying IR1, IR2, IR3 using the FDs in F.
Inference Rules (contd)

- Given some FDs, we can infer new FDs
  \[ F = \{ \text{StaffNo} \rightarrow \text{BranchNo}, \text{BranchNo} \rightarrow \text{Baddress} \} \]
  \[ \text{Implies StaffNo} \rightarrow \text{Baddress} \]

- \( F^+ \) is the set of all FDs that are implied by \( F \)

- Additional rules that follow from IR1-3.
  - Union: IF \( X \rightarrow Y \) and \( X \rightarrow Z \), then \( X \rightarrow YZ \)
  - Decomposition: If \( X \rightarrow YZ \), then \( X \rightarrow Y \) and \( X \rightarrow Z \)
  - Pseudo-transitivity: If \( X \rightarrow Y \) and \( WY \rightarrow Z \), then \( WX \rightarrow Z \)
Implied FDs using Inference rules

◆ **Relation P(EmpID, ProjID, Budget, TimeSpent)**

Each project has a single budget: \( \text{ProjID} \rightarrow \text{Budget} \) [FD1]

The relation keeps the time spent for an employee on each project: \( \text{EmpID, ProjID} \rightarrow \text{TimeSpent} \) [FD2]

◆ **Implied FDs**

- By augmentation on FD1:
  » \( \text{EmpID,ProjID} \rightarrow \text{EmpID,Budget} \) [FD3]

- By decomposing FD3:
  » \( \text{EmpID,ProjID} \rightarrow \text{EmpID} \) [FD4]
  » \( \text{EmpID,ProjID} \rightarrow \text{Budget} \) [FD5]
  » \( \text{EmpID,ProjID} \) determines all attribute of P (PK of P).
Does a set of FDs $F$ imply $X \rightarrow Y$?

- Computing $F+$ is expensive. Instead,
- Compute $X+$ (the closure of $X$) and check whether $X+$ includes all the attributes in $Y$.
- Algorithm for $X+$

  $$X+ = X$$

  repeat

  $$oldX+ = X+;$$

  for each FD $T \rightarrow Z$ in $F$ do

  if $X+ \supseteq T$ then $X+ = X+ \cup Z$;

  until ($X+ = oldX+$);

  If $X+$ includes all attributes, then $X$ is a key of the relation
Example of Implied FDs using X+

- Relation Contracts(EmpID, ProjID, Budget, TimeSpent) with set $F = \{FD1: \text{ProjID} \rightarrow \text{Budget}; \text{FD2: EmpID, ProjID} \rightarrow \text{TimeSpent}\}$
- Does $F$ imply $FD3: \text{EmpID, ProjID} \rightarrow \text{Budget}$?
- Using the algorithm for $X+$
  
  Initial $X+ = \text{EmpID, ProjID}$
  
  $X+ = \text{EmpID, ProjID} + \text{Budget}$ (using $FD1$)
  
  $X+ = \text{EmpID, ProjID, Budget} + \text{TimeSpent}$ (using $FD2$)
  
  $X+$ includes $Y$ therefore $FD3$ is implied by $F$

- Is $\text{EmpID} \rightarrow \text{Budget}$ implied by $F$?
Equivalence of sets of FDs

Two sets of FDs $F$ and $G$ are equivalent if:
- Every FD in $F$ can be inferred from $G$, and
- Every FD in $G$ can be inferred from $F$
- Hence, $F$ and $G$ are equivalent if $F^+ = G^+$

Definition (Covers):
- $F$ covers $G$ if every FD in $G$ can be inferred from $F$
  » (i.e., if $G^+ \subseteq F^+$)

$F$ and $G$ are equivalent if $F$ covers $G$ and $G$ covers $F$

There is an algorithm for checking equivalence of sets of FDs.
Minimal set of FDs

A set of FDs is minimal if it satisfies the following conditions:

1. Every dependency in F has a single attribute for its RHS.
2. We cannot remove any dependency from F and have a set of dependencies that is equivalent to F.
3. We cannot replace any dependency \( X \rightarrow A \) in F with a dependency \( Y \rightarrow A \), where Y is a proper-subset-of X (Y subset-of X) and still have a set of dependencies that is equivalent to F.

Minimal set of FDs represents a canonical form.

No redundant FDs
Minimal set of FDs (contd)

- Every set of FDs has an equivalent minimal set.
- A minimal cover of a set of FDs $E$ is a minimal set of dependencies that is equivalent to $E$.

- Algorithm to find minimal cover $F$ of a set of FDs $E$.

1. Set $F = E$
2. Replace each $X \rightarrow \{A_1, A_2, \ldots A_n\}$ in $F$ by $n$ FDs $X \rightarrow A_1$, $X \rightarrow A_2$, $\ldots$
3. For each FD $X \rightarrow A$ in $F$
   
   For each attribute $B$ in $X$
   
   If $\{\{F-\{X \rightarrow A\}\} U \{(X-\{B\}) \rightarrow A\}\} \rightarrow A$ is equivalent to $F$
   
   then replace $X \rightarrow A$ with $(X-\{B\}) \rightarrow A$ in $F$.
4. For each remaining FD $X \rightarrow A$ in $F$
   
   If $\{F-\{X \rightarrow A\}\}$ is equivalent to $F$,
   
   Then remove $X \rightarrow A$ from $F$. 
Computing a minimal set of FDs

We illustrate the above algorithm with the following:
Let the given set of FDs be \( E : \{ B \rightarrow A, D \rightarrow A, AB \rightarrow D \} \). We have to find the minimum cover of \( E \).

All above dependencies are in canonical form; so we have completed step 1 of Algorithm 10.2 and can proceed to step 2. In step 2 we need to determine if \( AB \rightarrow D \) has any redundant attribute on the left-hand side; that is, can it be replaced by \( B \rightarrow D \) or \( A \rightarrow D \)?

Since \( B \rightarrow A \), by augmenting with \( B \) on both sides (IR2), we have \( BB \rightarrow AB \), or \( B \rightarrow AB \) (i). However, \( AB \rightarrow D \) as given (ii).

Hence by the transitive rule (IR3), we get from (i) and (ii), \( B \rightarrow D \). Hence \( AB \rightarrow D \) may be replaced by \( B \rightarrow D \).

We now have a set equivalent to original \( E \), say \( E' : \{ B \rightarrow A, D \rightarrow A, B \rightarrow D \} \). No further reduction is possible in step 2 since all FDs have a single attribute on the left-hand side.

In step 3 we look for a redundant FD in \( E' \). By using the transitive rule on \( B \rightarrow D \) and \( D \rightarrow A \), we derive \( B \rightarrow A \). Hence \( B \rightarrow A \) is redundant in \( E' \) and can be eliminated.

Hence the minimum cover of \( E \) is \( \{ B \rightarrow D, D \rightarrow A \} \).
The Process of Normalization

- Formal technique for designing a relation based on its primary key and functional dependencies between its attributes.

- Often executed as a series of steps. Each step corresponds to a specific normal form, which has known properties.

- As normalization proceeds, relations become progressively more restricted (stronger) in format and also less vulnerable to update anomalies.
Properties of a good design

- Two important properties of decomposition:
  - **Lossless-join property**: No spurious tuples should be generated by doing a natural-join of any relations.
    - *This property cannot be sacrificed.*
  - **Dependency preservation property**: enforce a constraint on original relation by enforcing some constraint on each of the smaller relations.
    - *This property may be sacrificed.*
Unnormalized Form (UNF)

- A table that contains one or more repeating groups.

- To create an unnormalized table:
  - Transform data from information source (e.g. form) into table format with columns and rows.

- Denormalization:
  - The process of storing the join of higher normal form relations as a base relation—which is in a lower normal form.
First Normal Form (1NF)

- A relation in which intersection of each row and column contains one and only one value.
  - It does not allow non-atomic attributes.

- An unnormalized relation must be converted to a 1NF relation
  - First identify a primary key, then
  - Either
    » place repeating data along with copy of the original key attribute(s) into a separate relation
    » Place each value of a repeating group on a tuple with duplicate values of the non-repeating data (called “flattening” the table)
Normalization into 1NF

(a) DEPARTMENT

<table>
<thead>
<tr>
<th>Dname</th>
<th>Dnumber</th>
<th>Dmgr_ssn</th>
<th>Dlocations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b) DEPARTMENT

<table>
<thead>
<tr>
<th>Dname</th>
<th>Dnumber</th>
<th>Dmgr_ssn</th>
<th>Dlocations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Research</td>
<td>5</td>
<td>333445555</td>
<td>{Bellaire, Sugarland, Houston}</td>
</tr>
<tr>
<td>Administration</td>
<td>4</td>
<td>987654321</td>
<td>{Stafford}</td>
</tr>
<tr>
<td>Headquarters</td>
<td>1</td>
<td>888665555</td>
<td>{Houston}</td>
</tr>
</tbody>
</table>

(c) DEPARTMENT

<table>
<thead>
<tr>
<th>Dname</th>
<th>Dnumber</th>
<th>Dmgr_ssn</th>
<th>Dlocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Research</td>
<td>5</td>
<td>333445555</td>
<td>Bellaire</td>
</tr>
<tr>
<td>Research</td>
<td>5</td>
<td>333445555</td>
<td>Sugarland</td>
</tr>
<tr>
<td>Research</td>
<td>5</td>
<td>333445555</td>
<td>Houston</td>
</tr>
<tr>
<td>Admin</td>
<td>4</td>
<td>987654321</td>
<td>Stafford</td>
</tr>
<tr>
<td>Headquarters</td>
<td>1</td>
<td>888665555</td>
<td>Houston</td>
</tr>
</tbody>
</table>

Figure 10.8
Normalization into 1NF.
(a) A relation schema that is not in 1NF. (b) Example state of relation DEPARTMENT. (c) 1NF version of the same relation with redundancy.
Second Normal Form (2NF)

- Based on concept of full functional dependency:
  - A and B are attributes of a relation,
  - B is fully dependent on A if B is functionally dependent on A but not on any proper subset of A.

- 2NF - A relation that is in 1NF and every non-prime attribute is fully functionally dependent on the primary key.
  - Prime attribute: An attribute that is member of the primary key K
  - Example: EMP_PROJ(SSN, Pnum, Hours, Ename, Pname, Ploc)
    \{SSN, Pnum\} \rightarrow \text{Hours}, \text{SSN} \rightarrow \text{Ename}, \text{Pnum} \rightarrow \{\text{Pname, Ploc}\}

* Relation is not in 2NF
1NF to 2NF

- Identify FD’s in the relation.
- If partial dependencies exist on the primary key remove them by placing them in a new relation along with copy of their determinant.

\[ \text{EMP\_PROJ(} \text{SSN, Pnum, Hours, Ename, Pname, Ploc)} \]
\[ \{\text{SSN, Pnum} \rightarrow \text{Hours, SSN} \rightarrow \text{Ename, Pnum} \rightarrow \{\text{Pname, Ploc}\} \]

\text{EMP\_PROJ decomposed into:}
- \text{EMPLOYEE(} \text{SSN, Pnum, Hours)} , \{\text{SSN, Pnum} \rightarrow \text{Hours}\}
- \text{WORKS\_ON(} \text{SSN, Ename)} , \text{SSN} \rightarrow \text{Ename}
- \text{PROJECT(} \text{Pnum, Pname, Ploc)} , \text{Pnum} \rightarrow \{\text{Pname, Ploc}\}
Third Normal Form (3NF)

- Based on concept of transitive dependency:
  - A, B and C are attributes of a relation such that if \( A \rightarrow B \) and \( B \rightarrow C \),
  - then C is transitively dependent on A through B. (Provided that A is not functionally dependent on B or C).

- 3NF - A relation that is in 1NF and 2NF and in which no non-prime attribute is transitively dependent on the primary key.
2NF to 3NF

- Identify FD’s in the relation.
- If transitive dependencies exist on the primary key remove them by placing them in a new relation along with copy of their determinant.

```latex
\text{EMP\_DEPT}(\text{Enum}, \text{Ename}, \text{Sal}, \text{Dnum}, \text{Dname}, \text{Mgr})
\text{Enum} \rightarrow \{\text{Ename}, \text{Sal}, \text{Dnum}\}, \text{Dnum} \rightarrow \{\text{Dname}, \text{Mgr}\}
```

second FD is transitive through Dnum

Decompose EMP\_DEPT into

```latex
\text{EMP}(\text{Enum}, \text{Ename}, \text{Sal}, \text{Dnum}) \text{ and }
\text{DEPT}(\text{Dnum}, \text{Dname}, \text{Mgr})
```
## Summary of NFs based on primary keys

**Table 10.1**
Summary of Normal Forms Based on Primary Keys and Corresponding Normalization

<table>
<thead>
<tr>
<th>Normal Form</th>
<th>Test</th>
<th>Remedy (Normalization)</th>
</tr>
</thead>
<tbody>
<tr>
<td>First (1NF)</td>
<td>Relation should have no multivalued attributes or nested relations.</td>
<td>Form new relations for each multivalued attribute or nested relation.</td>
</tr>
<tr>
<td>Second (2NF)</td>
<td>For relations where primary key contains multiple attributes, no nonkey attribute should be functionally dependent on a part of the primary key.</td>
<td>Decompose and set up a new relation for each partial key with its dependent attribute(s). Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it.</td>
</tr>
<tr>
<td>Third (3NF)</td>
<td>Relation should not have a nonkey attribute functionally determined by another nonkey attribute (or by a set of nonkey attributes). That is, there should be no transitive dependency of a nonkey attribute on the primary key.</td>
<td>Decompose and set up a relation that includes the nonkey attribute(s) that functionally determine(s) other nonkey attribute(s).</td>
</tr>
</tbody>
</table>
General Definitions of 2NF and 3NF

- **Based on candidate keys (not only primary keys)**

- **Second normal form (2NF)**
  - A relation that is in 1NF and every non-prime attribute is fully functionally dependent on any candidate key.

- **Third normal form (3NF)**
  - A relation that is in 1NF and 2NF and in which no non-prime attribute is transitively dependent on any candidate key.

- **These definitions may find hidden redundancies.**
Boyce–Codd Normal Form (BCNF)

- Based on functional dependencies that take into account all candidate keys in a relation, however BCNF also has additional constraints compared with general definition of 3NF.

- BCNF - A relation is in BCNF if and only if every determinant is a candidate key.
Boyce–Codd normal form (BCNF)

- Difference between 3NF and BCNF is that for a functional dependency $A \rightarrow B$, 3NF allows this dependency in a relation if $B$ is a prime attribute and $A$ is not a candidate key.

- Whereas, BCNF insists that for this dependency to remain in a relation, $A$ must be a candidate key.

- Every relation in BCNF is also in 3NF. However, relation in 3NF may not be in BCNF.
Example of BCNF decomposition

(a) LOTS1A

<table>
<thead>
<tr>
<th>Property_id#</th>
<th>County_name</th>
<th>Lot#</th>
<th>Area</th>
</tr>
</thead>
</table>

FD1
FD2
FD5

BCNF Normalization

LOTS1AX

| Property_id# | Area | Lot# |

LOTS1AY

| Area | County_name |

(b) R

| A | B | C |

FD1
FD2

Figure 10.12

Boyce-Codd normal form. (a) BCNF normalization of LOTS1A with the functional dependency FD2 being lost in the decomposition. (b) A schematic relation with FDs; it is in 3NF, but not in BCNF.
## BCNF (contd)

### TEACH

<table>
<thead>
<tr>
<th>Student</th>
<th>Course</th>
<th>Instructor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Narayan</td>
<td>Database</td>
<td>Mark</td>
</tr>
<tr>
<td>Smith</td>
<td>Database</td>
<td>Navathe</td>
</tr>
<tr>
<td>Smith</td>
<td>Operating Systems</td>
<td>Ammar</td>
</tr>
<tr>
<td>Smith</td>
<td>Theory</td>
<td>Schulman</td>
</tr>
<tr>
<td>Wallace</td>
<td>Database</td>
<td>Mark</td>
</tr>
<tr>
<td>Wallace</td>
<td>Operating Systems</td>
<td>Ahamad</td>
</tr>
<tr>
<td>Wong</td>
<td>Database</td>
<td>Omiecinski</td>
</tr>
<tr>
<td>Zelaya</td>
<td>Database</td>
<td>Navathe</td>
</tr>
<tr>
<td>Narayan</td>
<td>Operating Systems</td>
<td>Ammar</td>
</tr>
</tbody>
</table>

### Figure 10.13

A relation `TEACH` that is in 3NF but not BCNF.
BCNF (contd)

- Two FDs exist in the relation TEACH:
  - fd1: \{ student, course \} \rightarrow instructor
  - fd2: instructor \rightarrow course

- \{ student, course \} is a candidate key for this relation and that the dependencies shown follow the pattern in Figure 10.12 (b).
  - So this relation is in 3NF but not in BCNF

- A relation NOT in BCNF should be decomposed so as to meet this property, while possibly forgoing the preservation of all functional dependencies in the decomposed relations.
  - (See Algorithm 11.3)
BCNF (contd)

- Main steps of algorithm 11.3:
  - Find the FD $X \rightarrow Y$ in $Q$ that violates BCNF
  - Replace $Q$ by two relations with schemas $(Q-Y)$ and $(X,Y)$.
  - TEACH relation is decomposed into:
    - $T1(\text{instructor, course})$
    - $T2(\text{instructor, student})$
  - FD fd1 is lost after decomposition.
  - A test for lossless decomposition is discussed in 11.1.4
Review of Normalization (UNF to BCNF)

DreamHome
Property Inspection Report

Property Number  PG4

Property Address  6 Lawrence St, Glasgow

<table>
<thead>
<tr>
<th>Inspection Date</th>
<th>Inspection Time</th>
<th>Comments</th>
<th>Staff no</th>
<th>Staff Name</th>
<th>Car Registration</th>
</tr>
</thead>
<tbody>
<tr>
<td>18-Oct-00</td>
<td>10.00</td>
<td>Need to replace crockery</td>
<td>SG37</td>
<td>Ann Beech</td>
<td>M231 JGR</td>
</tr>
<tr>
<td>22-Apr-01</td>
<td>09.00</td>
<td>In good order</td>
<td>SG14</td>
<td>David Ford</td>
<td>M533 HDR</td>
</tr>
<tr>
<td>1-Oct-01</td>
<td>12.00</td>
<td>Damp rot in bathroom</td>
<td>SG14</td>
<td>David Ford</td>
<td>N721 HFR</td>
</tr>
</tbody>
</table>
### Review of Normalization (UNF to BCNF)

<table>
<thead>
<tr>
<th>propertyNo</th>
<th>pAddress</th>
<th>iDate</th>
<th>iTime</th>
<th>comments</th>
<th>staffNo</th>
<th>sName</th>
<th>carReg</th>
</tr>
</thead>
<tbody>
<tr>
<td>PG4</td>
<td>6 Lawrence St, Glasgow</td>
<td>18-Oct-00</td>
<td>10.00</td>
<td>Need to replace crockery</td>
<td>SG37</td>
<td>Ann Beech</td>
<td>M231 JGR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>22-Apr-01</td>
<td>09.00</td>
<td>In good order</td>
<td>SG14</td>
<td>David Ford</td>
<td>M533 HDR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1-Oct-01</td>
<td>12.00</td>
<td>Damp rot in bathroom</td>
<td>SG14</td>
<td>David Ford</td>
<td>N721 HFR</td>
</tr>
<tr>
<td>PG16</td>
<td>5 Novar Dr, Glasgow</td>
<td>22-Apr-01</td>
<td>13.00</td>
<td>Replace living room carpet</td>
<td>SG14</td>
<td>David Ford</td>
<td>M533 HDR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>24-Oct-01</td>
<td>14.00</td>
<td>Good condition</td>
<td>SG37</td>
<td>Ann Beech</td>
<td>N721 HFR</td>
</tr>
</tbody>
</table>

### StaffPropertyInspection

<table>
<thead>
<tr>
<th>propertyNo</th>
<th>iDate</th>
<th>iTime</th>
<th>pAddress</th>
<th>comments</th>
<th>staffNo</th>
<th>sName</th>
<th>carReg</th>
</tr>
</thead>
<tbody>
<tr>
<td>PG4</td>
<td>18-Oct-00</td>
<td>10.00</td>
<td>6 Lawrence St, Glasgow</td>
<td>Need to replace crockery</td>
<td>SG37</td>
<td>Ann Beech</td>
<td>M231 JGR</td>
</tr>
<tr>
<td></td>
<td>22-Apr-01</td>
<td>09.00</td>
<td>6 Lawrence St, Glasgow</td>
<td>In good order</td>
<td>SG14</td>
<td>David Ford</td>
<td>M533 HDR</td>
</tr>
<tr>
<td></td>
<td>1-Oct-01</td>
<td>12.00</td>
<td>6 Lawrence St, Glasgow</td>
<td>Damp rot in bathroom</td>
<td>SG14</td>
<td>David Ford</td>
<td>N721 HFR</td>
</tr>
<tr>
<td>PG16</td>
<td>22-Apr-01</td>
<td>13.00</td>
<td>5 Novar Dr, Glasgow</td>
<td>Replace living room carpet</td>
<td>SG14</td>
<td>David Ford</td>
<td>M533 HDR</td>
</tr>
<tr>
<td></td>
<td>24-Oct-01</td>
<td>14.00</td>
<td>5 Novar Dr, Glasgow</td>
<td>Good condition</td>
<td>SG37</td>
<td>Ann Beech</td>
<td>N721 HFR</td>
</tr>
</tbody>
</table>
Review of Normalization (UNF to BCNF)

**StaffPropertyInspection**

<table>
<thead>
<tr>
<th>propertyNo</th>
<th>iDate</th>
<th>iTIme</th>
<th>pAddress</th>
<th>comments</th>
<th>staffNo</th>
<th>sName</th>
<th>carReg</th>
</tr>
</thead>
</table>

- **fd1**: (Primary key)
- **fd2**: (Partial dependency)
- **fd3**: (Transitive dependency)
- **fd4**:  
- **fd5**: (Candidate key)
- **fd6**: (Candidate key)
Review of Normalization (UNF to BCNF)

- StaffPropertyInspection (1NF)
  - PropertyInspection (2NF)
    - PropertyInspect (3NF)
      - Staff (fd3)
      - StaffCar (fd4)
      - Inspection (fd1', fd6')
      - Property (fd2)
  - Use fd2
  - (lost fd5)