Chapter 14
Normalization

Content

Informal Design Guideline for Relation Schemas

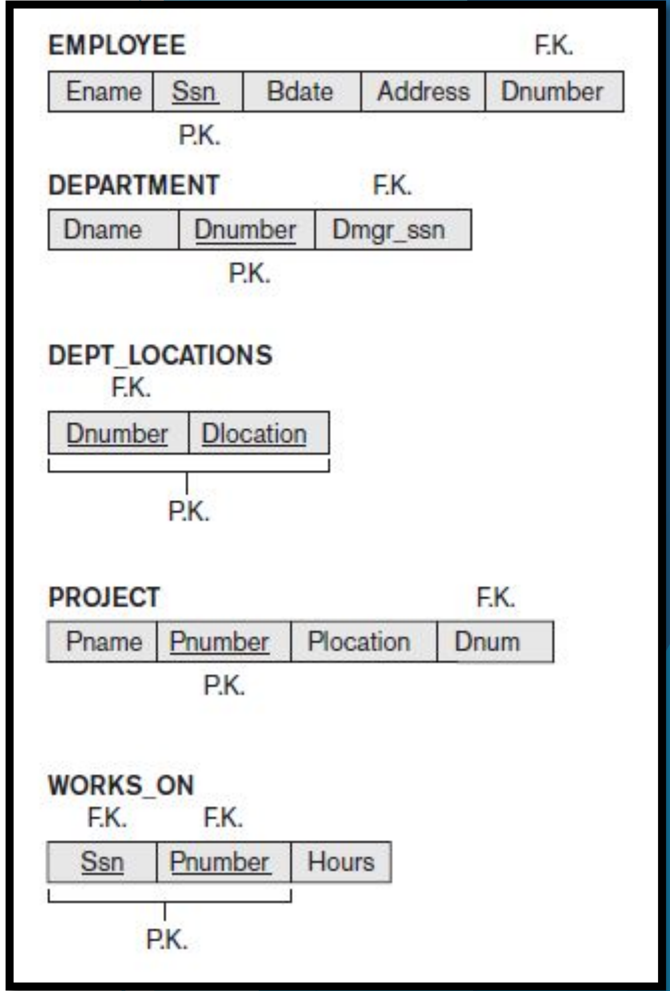
- We will discuss four informal guidelines that may be used as measures to determine the quality of relation schema design:
  - Making sure that the semantics of the attributes is clear in the schema
  - Reducing the redundant information in tuples
  - Reducing the NULL values in tuples
  - Disallowing the possibility of generating spurious tuples

Informal Design Guideline for Relation Schemas

- ***Imparting Clear Semantics to Attributes in Relations***
- semantics of a relation:
  - **refers to its meaning** resulting from the interpretation of attribute values in a tuple.
  - i.e., what a relation exactly means and stands for

Informal Design Guideline for Relation Schemas

- ***Imparting Clear Semantics to Attributes in Relations***
- The meaning of the EMPLOYEE relation schema is simple:
  - Each tuple represents an employee, with values for:
    - the employee's name (Ename),
    - Social Security number (Ssn),
    - birth date (Bdate),
    - address (Address),
    - number of the department that the employee works for (Dnumber).
- **Dnumber attribute:** A FK that represents a relationship between EMPLOYEE and DEPARTMENT.
- The semantics of the DEPARTMENT and PROJECT schemas are also straightforward:
  - Each DEPARTMENT tuple represents a department entity,
  - Each PROJECT tuple represents a project entity.
  - The attribute Dmgr_ssn of DEPARTMENT relates a department to the employee who is its manager,
  - whereas Dnum of PROJECT relates a project to its controlling department; both are foreign key attributes.



Informal Design Guideline for Relation Schemas

- ***Imparting Clear Semantics to Attributes in Relations***
- Each tuple in DEPT_LOCATIONS gives
  - a department number (Dnumber)
  - and one of the locations of the department (Dlocation).
- Each tuple in WORKS_ON gives
  - an employee Social Security number (Ssn),
  - the project number of one of the projects that the employee works on (Pnumber),
  - and the number of hours per week that the employee works on that project (Hours).

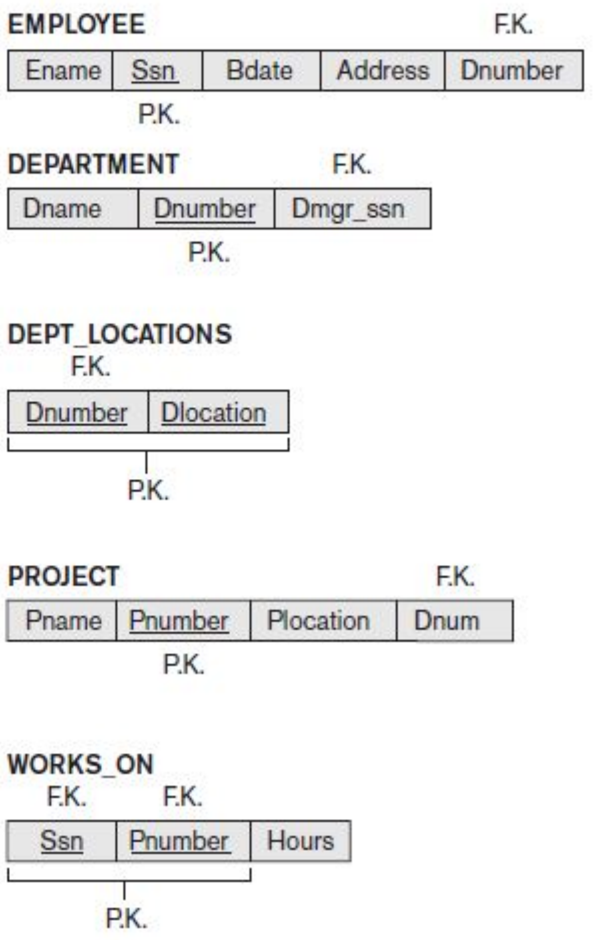


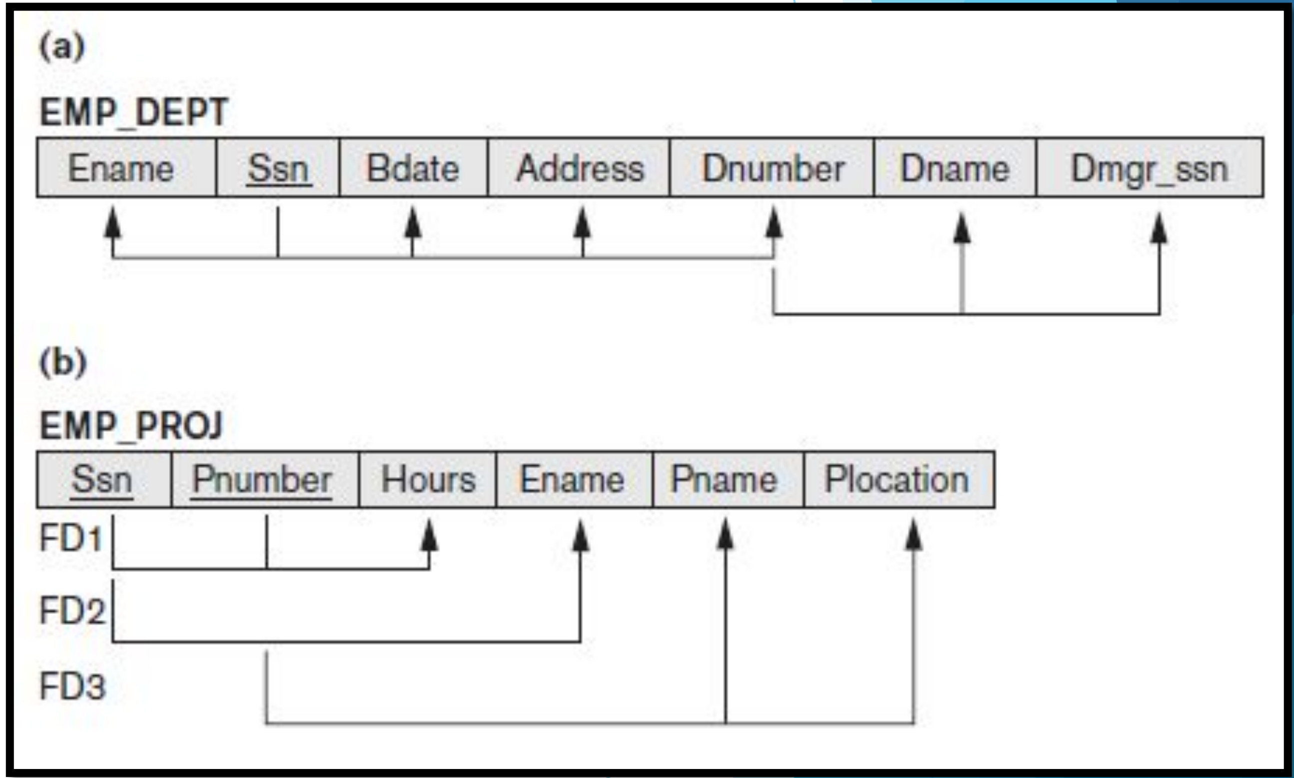
Figure 14.1
A simplified COMPANY relational database schema.

Informal Design Guideline for Relation Schemas

- *Imparting Clear Semantics to Attributes in Relations*
- *Guideline 1.*
- **Design** a relation schema so that it is **easy to explain its meaning**.
- **Do not combine attributes from multiple entity types and relationship types into a single relation.**
- If a relation schema corresponds to one **entity type or one relationship type**, it is straightforward to explain its meaning.
- Otherwise, if the relation corresponds to a **mixture of multiple entities and relationships**, semantic ambiguities will result, and the relation cannot be easily explained.

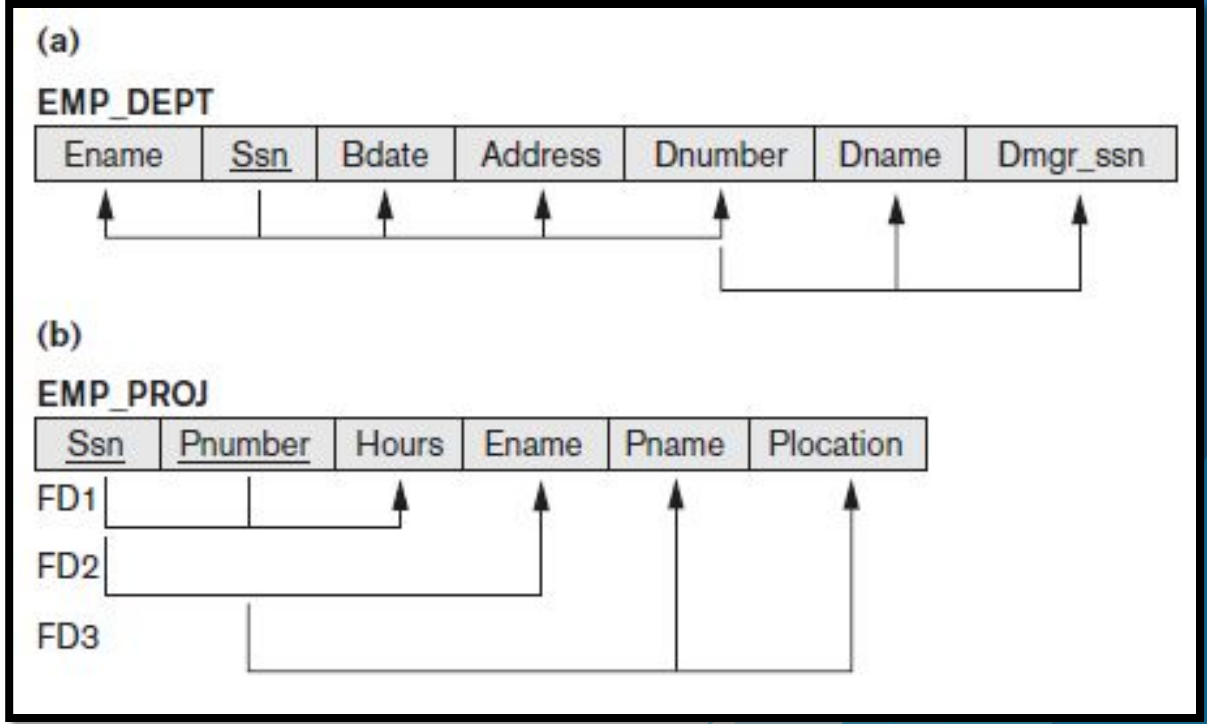
Informal Design Guideline for Relation Schemas

- ***Imparting Clear Semantics to Attributes in Relations***
- ***Guideline 1.***
- ***Examples of Violating Guideline 1***
- The relation schemas in Figures (a) and (b) also have clear semantics.
- A **tuple in the EMP_DEPT relation schema in Figure (a) represents a single employee but includes, along with the Dnumber (the identifier for the department he/she works for), additional information**—namely, the name (Dname) of the department for which the employee works and the Social Security number (Dmgr_ssn) of the department manager.



Informal Design Guideline for Relation Schemas

- ***Imparting Clear Semantics to Attributes in Relations***
- ***Guideline 1.***
- ***Examples of Violating Guideline 1***
- For the EMP_PROJ relation in Figure 14.3(b), each tuple relates an employee to a project but also includes the employee name (Ename), project name (Pname), and project location (Plocation).
- **Although there is nothing wrong logically with these two relations, they violate Guideline 1 by mixing attributes from distinct real-world entities:**
  - **EMP_DEPT mixes attributes of employees and departments, and EMP_PROJ mixes attributes of employees and projects and the WORKS_ON relationship.**



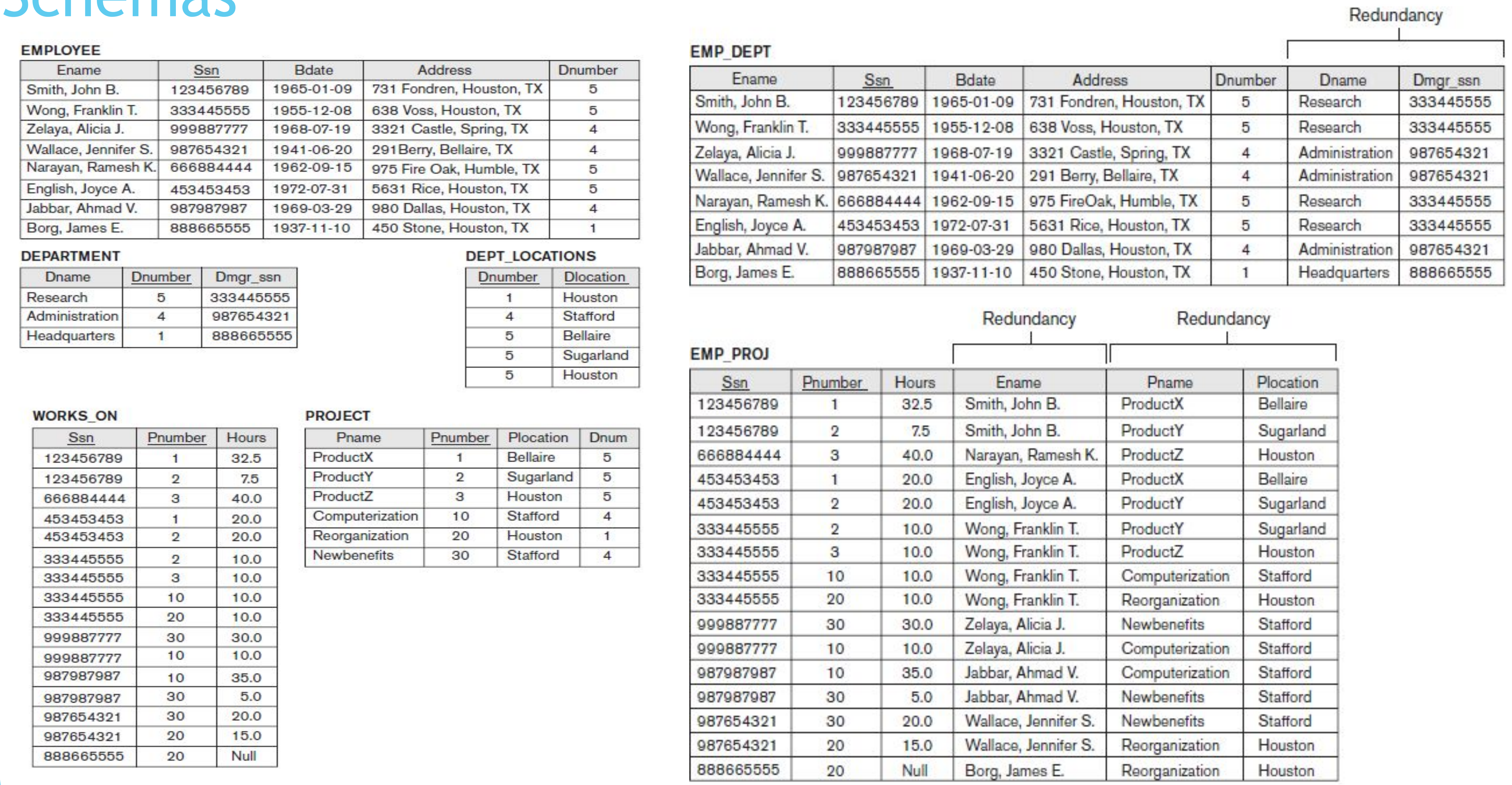
Informal Design Guideline for Relation Schemas

- ***Redundant Information in Tuples and Update Anomalies***
- One goal of schema design is to **minimize the storage space used by the base relations.**
- For example,
  - **compare the space used by the two base relations EMPLOYEE** and **DEPARTMENT** with that for an **EMP_DEPT** base relation, (which is the result of applying the **NATURAL JOIN** operation to EMPLOYEE and DEPARTMENT).
- In EMP_DEPT, the attribute **values** pertaining to a particular department (Dnumber, Dname, Dmgr_ssn) are **repeated** for every employee who works for that department.

Informal Design Guideline for Relation Schemas

- ***Redundant Information in Tuples and Update Anomalies***
- In contrast, **each department's information appears only once in the DEPARTMENT relation.**
- Only the department number **(Dnumber)** is **repeated** in the EMPLOYEE relation for each employee who works in that department as a **foreign key**.
- **Similar comments apply to the EMP_PROJ relation** (see Figure 14.4), which augments the WORKS_ON relation with additional attributes from EMPLOYEE and PROJECT.

Informal Design Guideline for Relation Schemas



EMPLOYEE

Ename	Ssn	Bdate	Address	Dnumber
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4
Wallace, Jennifer S.	987654321	1941-06-20	291Berry, Bellaire, TX	4
Narayan, Ramesh K.	666884444	1962-09-15	975 Fire Oak, Humble, TX	5
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1

DEPARTMENT

Dname	Dnumber	Dmgr_ssn
Research	5	333445555
Administration	4	987654321
Headquarters	1	888665555

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Ssn	Pnumber	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	Null

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

Redundancy

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975 FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5	Research	333445555
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1	Headquarters	888665555

Redundancy

Redundancy

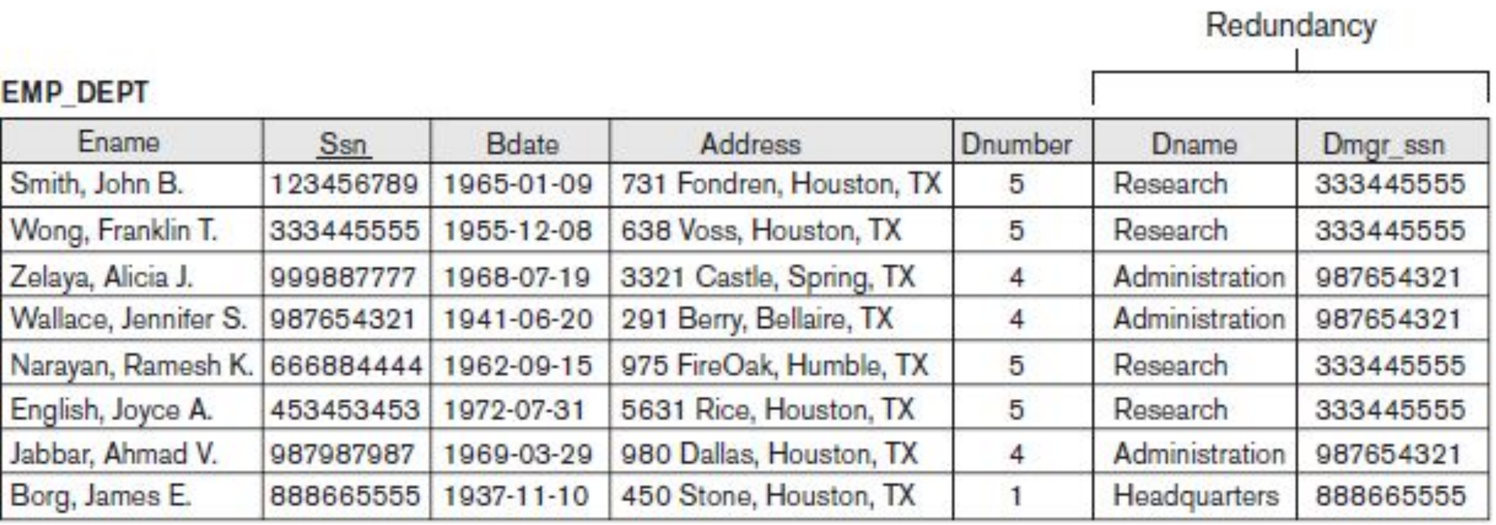
EMP_PROJ

Ssn	Pnumber	Hours	Ename	Pname	Plocation
123456789	1	32.5	Smith, John B.	ProductX	Bellaire
123456789	2	7.5	Smith, John B.	ProductY	Sugarland
666884444	3	40.0	Narayan, Ramesh K.	ProductZ	Houston
453453453	1	20.0	English, Joyce A.	ProductX	Bellaire
453453453	2	20.0	English, Joyce A.	ProductY	Sugarland
333445555	2	10.0	Wong, Franklin T.	ProductY	Sugarland
333445555	3	10.0	Wong, Franklin T.	ProductZ	Houston
333445555	10	10.0	Wong, Franklin T.	Computerization	Stafford
333445555	20	10.0	Wong, Franklin T.	Reorganization	Houston
999887777	30	30.0	Zelaya, Alicia J.	Newbenefits	Stafford
999887777	10	10.0	Zelaya, Alicia J.	Computerization	Stafford
987987987	10	35.0	Jabbar, Ahmad V.	Computerization	Stafford
987987987	30	5.0	Jabbar, Ahmad V.	Newbenefits	Stafford
987654321	30	20.0	Wallace, Jennifer S.	Newbenefits	Stafford
987654321	20	15.0	Wallace, Jennifer S.	Reorganization	Houston
888665555	20	Null	Borg, James E.	Reorganization	Houston

Informal Design Guideline for Relation Schemas

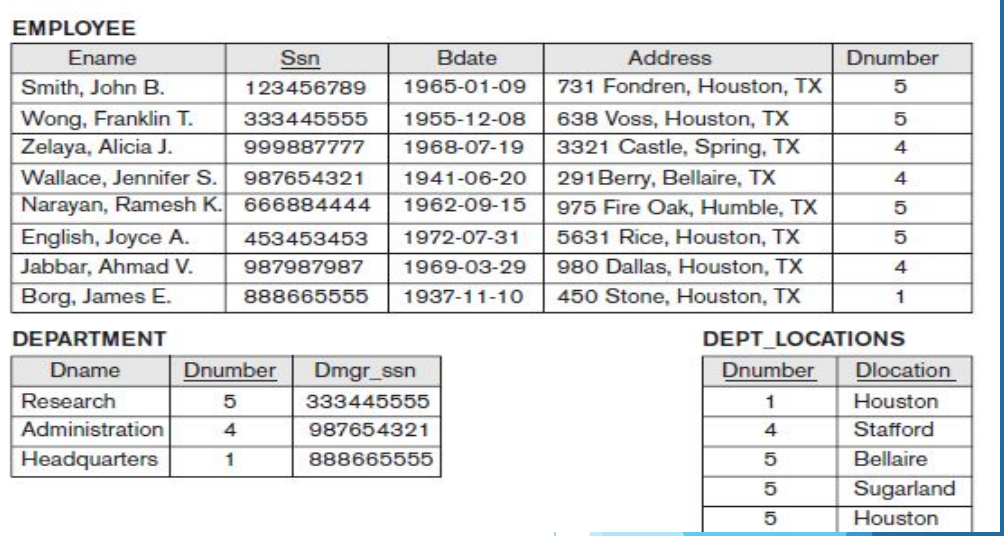
- ***Redundant Information in Tuples and Update Anomalies***
- **Storing natural joins of base relations leads to an additional problem referred to as update anomalies.**
- These can be classified into:
  - insertion anomalies,
  - deletion anomalies,
  - modification anomalies.
- **Insertion Anomalies**. Insertion anomalies can be differentiated into two types, illustrated by the following examples based on the **EMP_DEPT** relation.

Informal Design Guideline for Relation Schemas


Redundancy

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975 FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5	Research	333445555
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1	Headquarters	888665555



EMPLOYEE

Ename	Ssn	Bdate	Address	Dnumber
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4
Narayan, Ramesh K.	666884444	1962-09-15	975 Fire Oak, Humble, TX	5
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1

DEPARTMENT

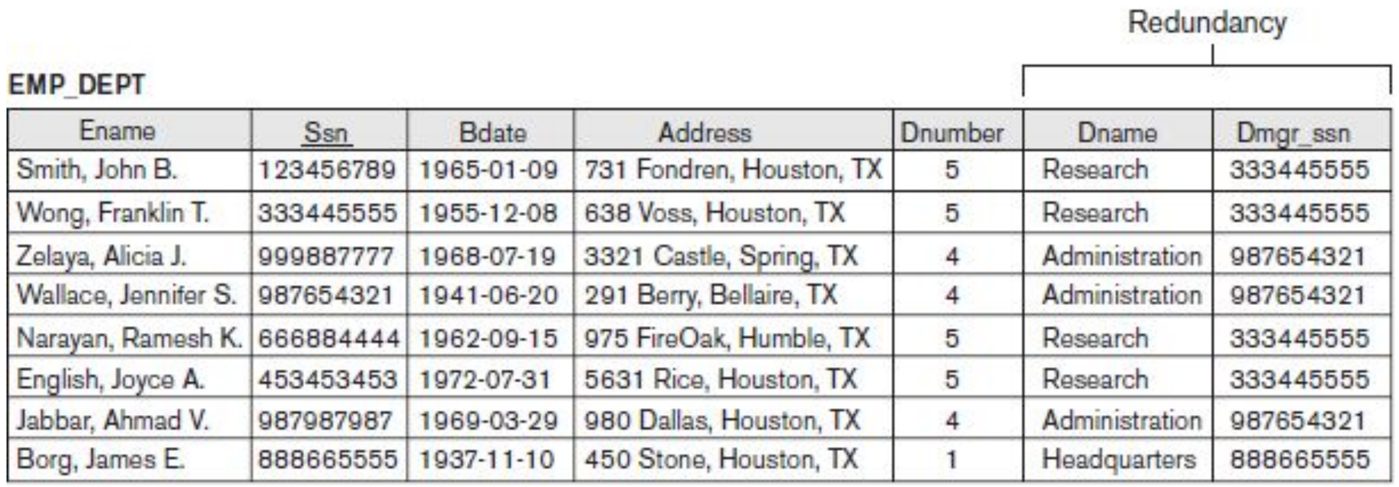
Dname	Dnumber	Dmgr_ssn
Research	5	333445555
Administration	4	987654321
Headquarters	1	888665555

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

- ***Redundant Information in Tuples and Update Anomalies***
- Insertion Anomalies
- To insert a new employee tuple into EMP_DEPT, **we must include either the attribute values for the department that the employee works for, or NULLs (if the employee does not work for a department as yet).**
  - For example, to insert a new tuple for an employee who works in department number 5, we must enter all the attribute values of department 5 correctly so that they are consistent with the corresponding values for department 5 in other tuples in EMP_DEPT.
    - In Figure 2, we do not have to worry about this consistency problem because we enter only the department number in the employee tuple; all other attribute values of department 5 are recorded only once in the database, as a single tuple in the DEPARTMENT relation.

Informal Design Guideline for Relation Schemas

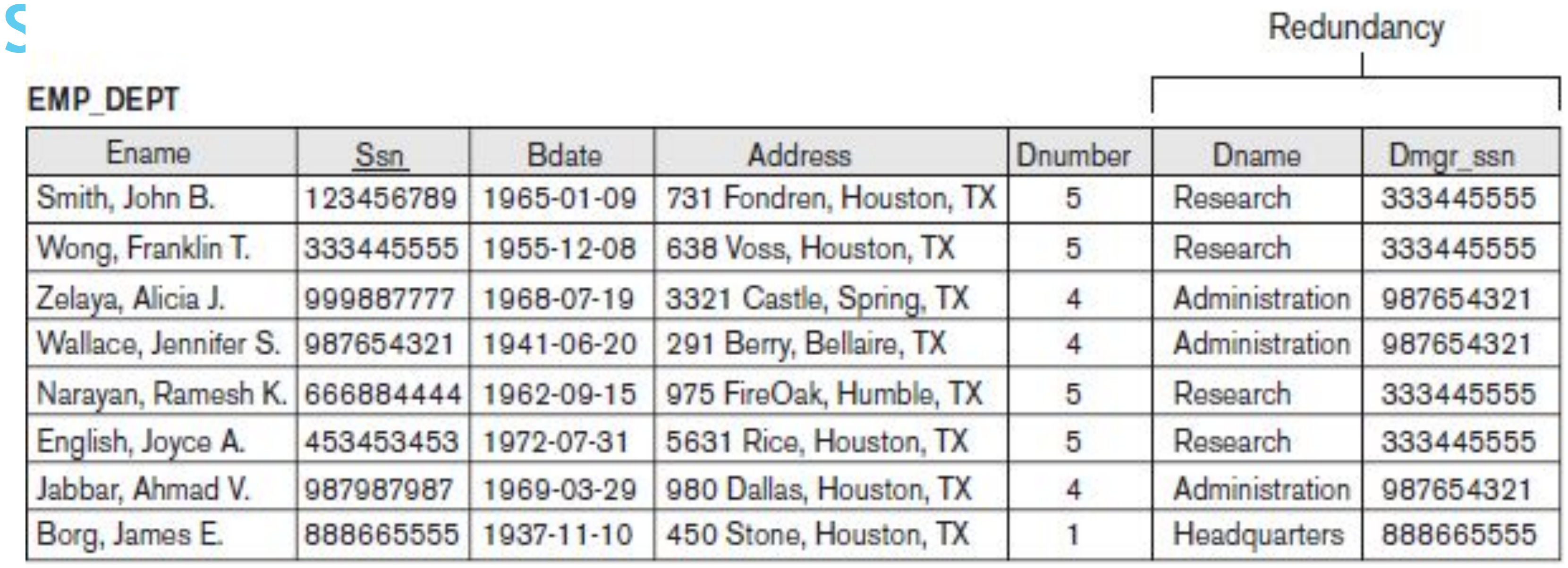

Redundancy

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975 FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5	Research	333445555
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1	Headquarters	888665555

- *Redundant Information in Tuples and Update Anomalies*
- **It is difficult to insert a new department that has no employees as yet in the EMP_DEPT relation.**
  - The **only way to do this is to place NULL values in the attributes for employee**.
  - This **violates the entity integrity for EMP_DEPT because its primary key SSN cannot be null.**
    - This problem does not occur in the design of Figure 14.2 because a department is entered in the DEPARTMENT relation whether any employees work for it or not, and whenever an employee is assigned to that department, a corresponding tuple is inserted in EMPLOYEE.

Informal Design Guideline for Relation Schemas



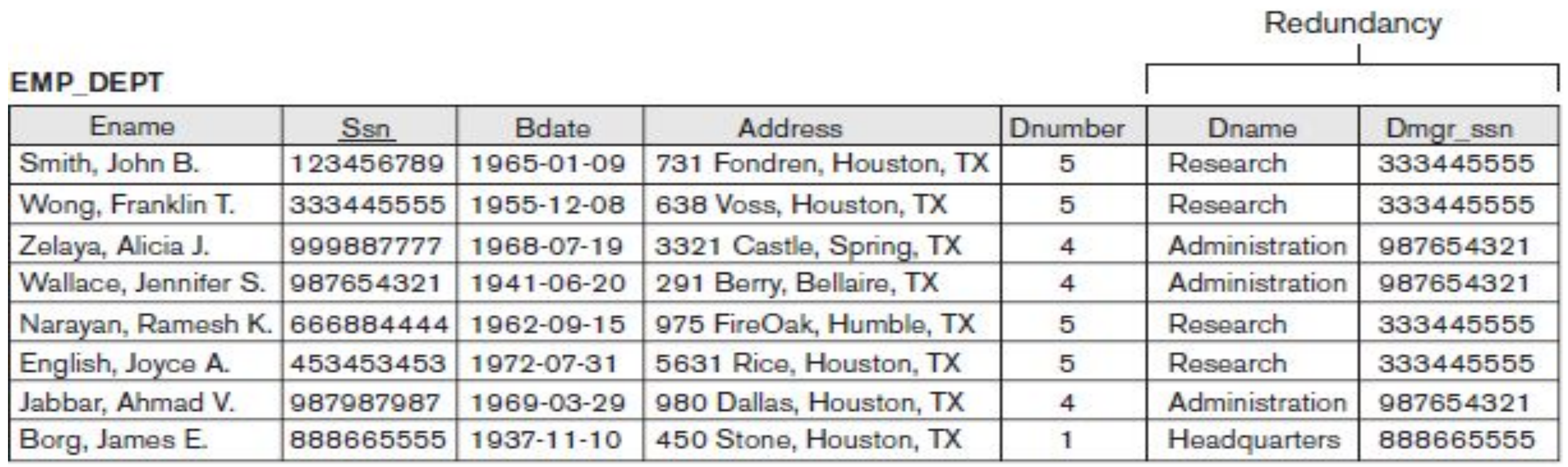
Redundancy

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975 FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5	Research	333445555
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1	Headquarters	888665555

- *Redundant Information in Tuples and Update Anomalies*
- **Deletion Anomalies.**
- If we delete from EMP_DEPT,
  - and that employee is the last employee working in a particular department,
  - then the information related to that department will be lost as well.
- This problem does not occur in the 14.2 designed database because DEPARTMENT tuples are stored separately.

Informal Design Guideline for Relation Schemas


Redundancy

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975 FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5	Research	333445555
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1	Headquarters	888665555

- *Redundant Information in Tuples and Update Anomalies*
- **Modification Anomalies.**
  - In EMP_DEPT, if we change the value of one of the attributes of a particular department
  - —say, the manager of department 5
  - —**we must update the tuples of all employees who work in that department**;
  - **otherwise, the database will become inconsistent**.

Informal Design Guideline for Relation Schemas

- ***Redundant Information in Tuples and Update Anomalies***
- ***Guideline 2.***
- **Design the base relation schemas so that no insertion, deletion, or modification anomalies are present in the relations.**
- These guidelines may sometimes have to be violated in order to improve the performance of certain queries.
  - For example,
    - If EMP_DEPT is stored as a materialized view in addition to the base relations of EMPLOYEE and DEPARTMENT, the anomalies in EMP_DEPT must be noted and accounted for
    - for example, by using triggers or stored procedures that would make **automatic updates**.
- This way, whenever the base relation is updated, we do not end up with inconsistencies.

Informal Design Guideline for Relation Schemas

- ***NULL Values in Tuples***
- Problem 1:
  - If **most of the attributes do not apply to all tuples** in the relation, **then we have many NULLs in the tuples.**
  - This can **waste space at the storage level.**
- Problem 2:
  - What to do with **NULLs when aggregate operations such as COUNT or SUM are applied.**
- Problem 3:
  - **SELECT and JOIN operations involve comparisons**; with NULL the **results may become unpredictable.**

Informal Design Guideline for Relation Schemas

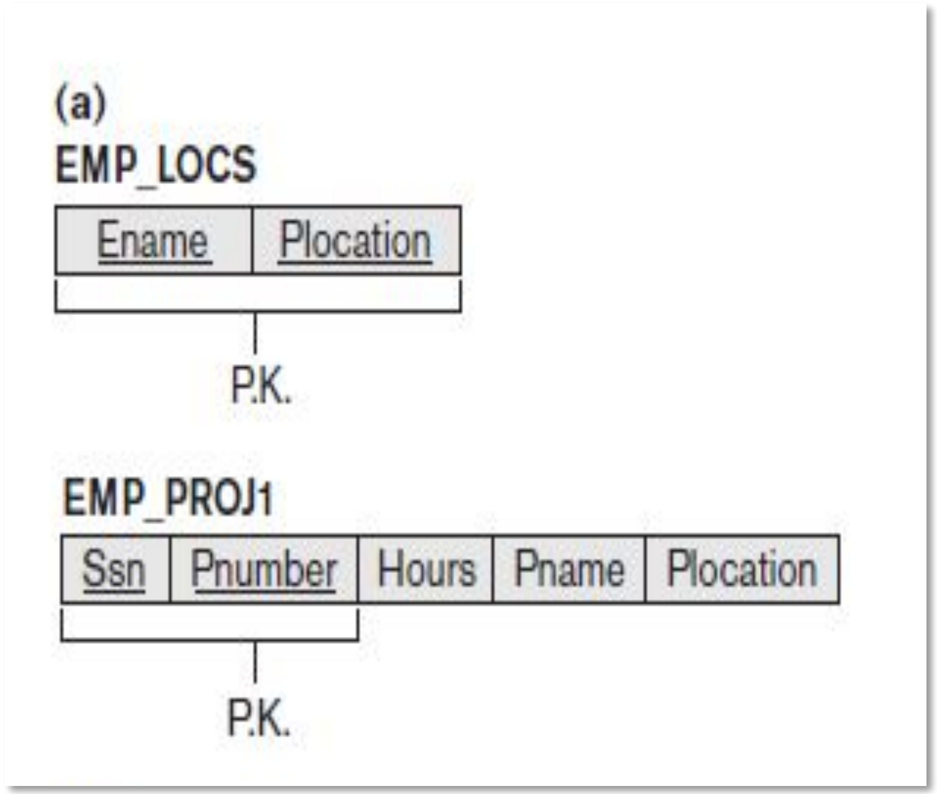
- ***NULL Values in Tuples***
- Moreover, **NULLs can have multiple interpretations**, such as the following:
  - The attribute **does not apply** to this tuple. For example, Visa_status may not apply to U.S. students.
  - The attribute **value for this tuple is unknown**. For example, the Date_of_birth may be unknown for an employee.
  - The value is **known but absent**; that is, it has not been recorded yet. For example, the Home_Phone_Number for an employee may exist, but may not be available and recorded yet.

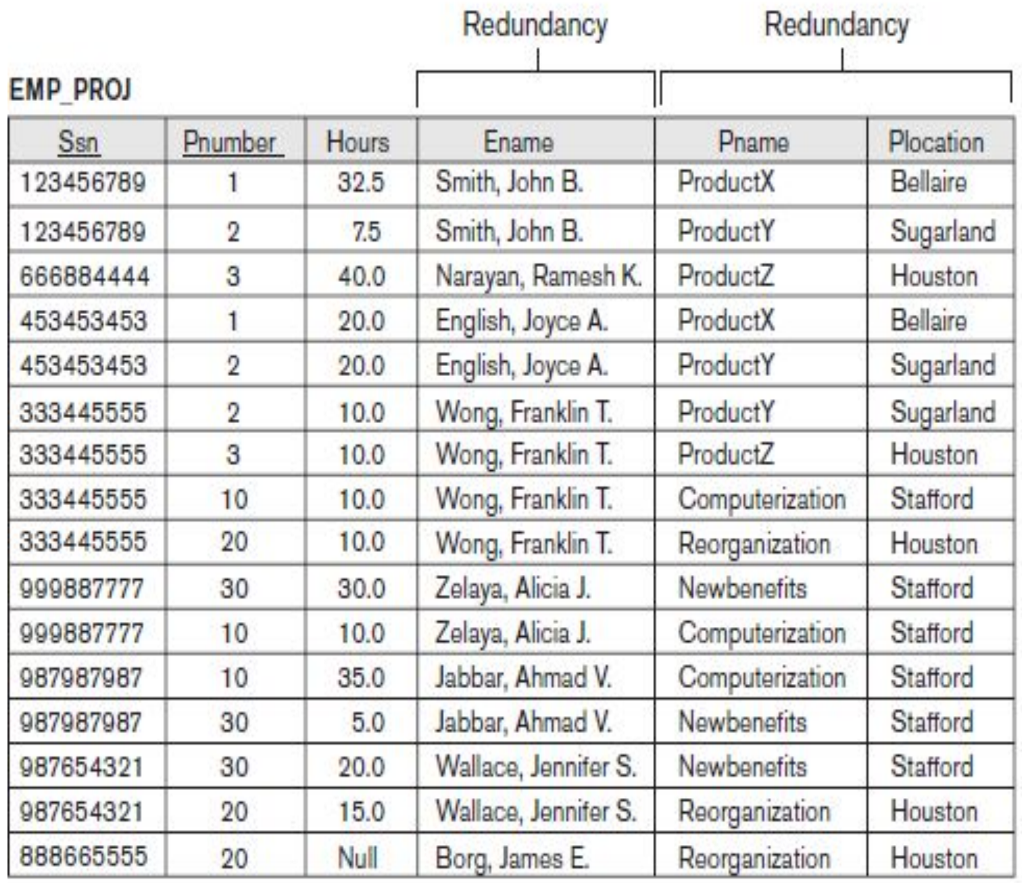
Informal Design Guideline for Relation Schemas

- ***NULL Values in Tuples***
- ***Guideline 3***.
- Avoid placing **attributes** in a base relation whose values may frequently be NULL.
- If NULLs are **unavoidable**,
  - apply in exceptional cases only
  - and do not apply to most tuples in the relation.
- For example,
  - if only 15% of employees have individual offices,
  - Then there is no need to include an attribute **Office_number** in the EMPLOYEE relation;
  - rather, a relation **EMP_OFFICES (Essn, Office_number)** can be created to include tuples for only the employees with individual offices.

Informal Design Guideline for Relation Schemas

Generation of Spurious Tuples





Redundancy Redundancy

EMP_PROJ

Ssn	Pnumber	Hours	Ename	Pname	Plocation
123456789	1	32.5	Smith, John B.	ProductX	Bellaire
123456789	2	7.5	Smith, John B.	ProductY	Sugarland
666884444	3	40.0	Narayan, Ramesh K.	ProductZ	Houston
453453453	1	20.0	English, Joyce A.	ProductX	Bellaire
453453453	2	20.0	English, Joyce A.	ProductY	Sugarland
333445555	2	10.0	Wong, Franklin T.	ProductY	Sugarland
333445555	3	10.0	Wong, Franklin T.	ProductZ	Houston
333445555	10	10.0	Wong, Franklin T.	Computerization	Stafford
333445555	20	10.0	Wong, Franklin T.	Reorganization	Houston
999887777	30	30.0	Zelaya, Alicia J.	Newbenefits	Stafford
999887777	10	10.0	Zelaya, Alicia J.	Computerization	Stafford
987987987	10	35.0	Jabbar, Ahmad V.	Computerization	Stafford
987987987	30	5.0	Jabbar, Ahmad V.	Newbenefits	Stafford
987654321	30	20.0	Wallace, Jennifer S.	Newbenefits	Stafford
987654321	20	15.0	Wallace, Jennifer S.	Reorganization	Houston
888665555	20	Null	Borg, James E.	Reorganization	Houston

Informal Design Guideline for Relation Schemas

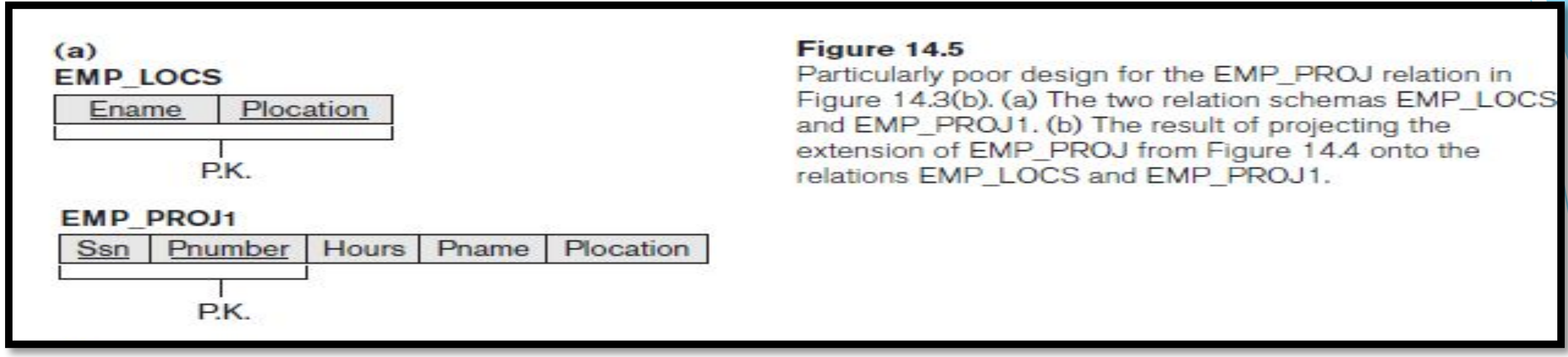
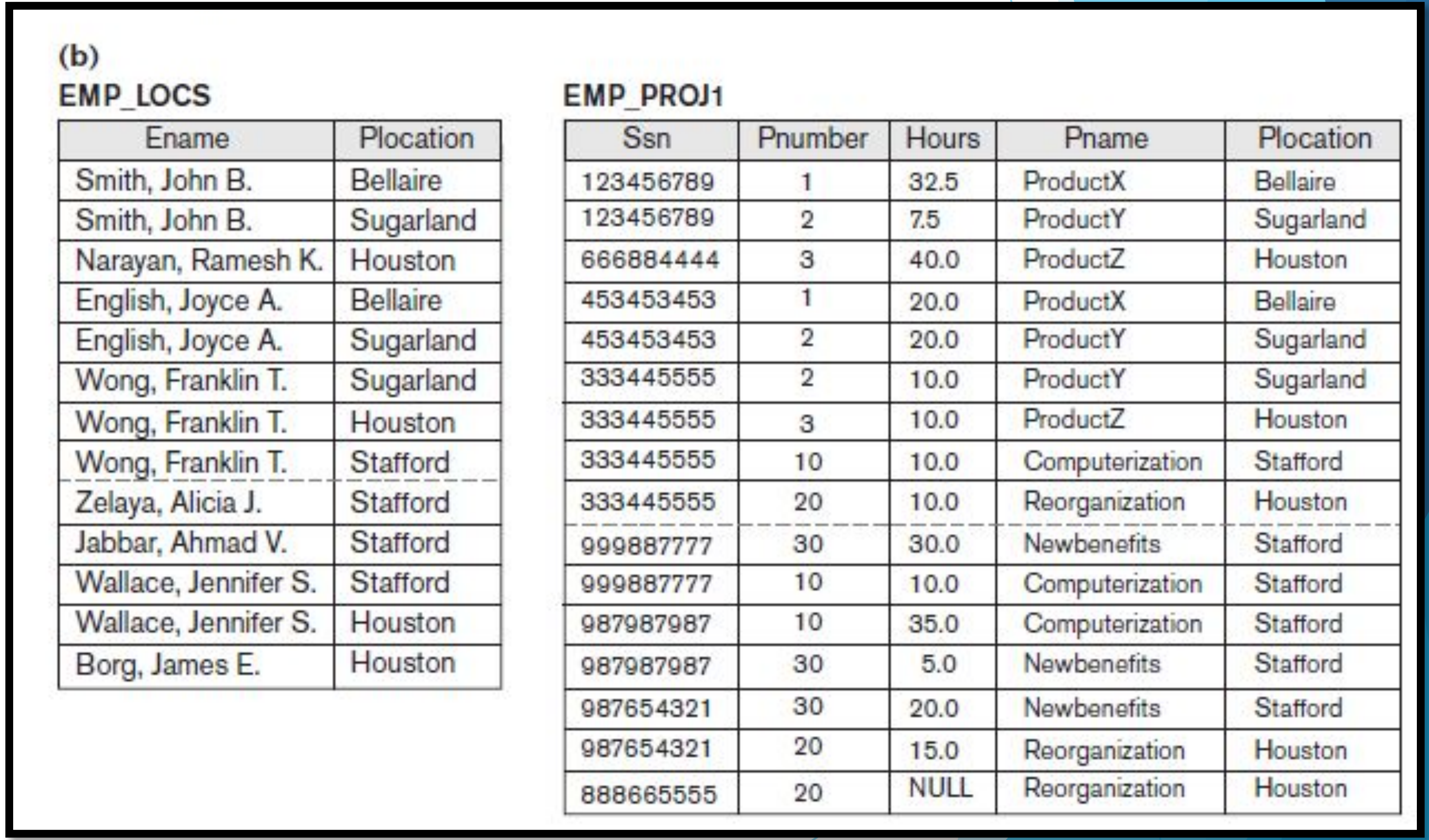


Figure 14.5
Particularly poor design for the EMP_PROJ relation in Figure 14.3(b). (a) The two relation schemas EMP_LOCS and EMP_PROJ1. (b) The result of projecting the extension of EMP_PROJ from Figure 14.4 onto the relations EMP_LOCS and EMP_PROJ1.

- ***Generation of Spurious Tuples***
- ***Spurious tuples: tuples that actually doesn't exists in a table.***
- Consider the two relation schemas EMP_LOCS and EMP_PROJ1.
- A tuple in EMP_LOCS means that the employee whose name is Ename works on at least one project located at Plocation.
- A tuple in EMP_PROJ1 refers to the fact that the employee whose Social Security number is Ssn works the given Hours per week on the project whose name, number, and location are Pname, Pnumber, and Plocation.

Informal Design Guideline for Relation Schemas

- ***Generation of Spurious Tuples***
- Figure 14.5(b) shows relation states of EMP_LOCS and EMP_PROJ1 corresponding to the EMP_PROJ relation in Figure 14.4.



(b)

EMP_LOCS

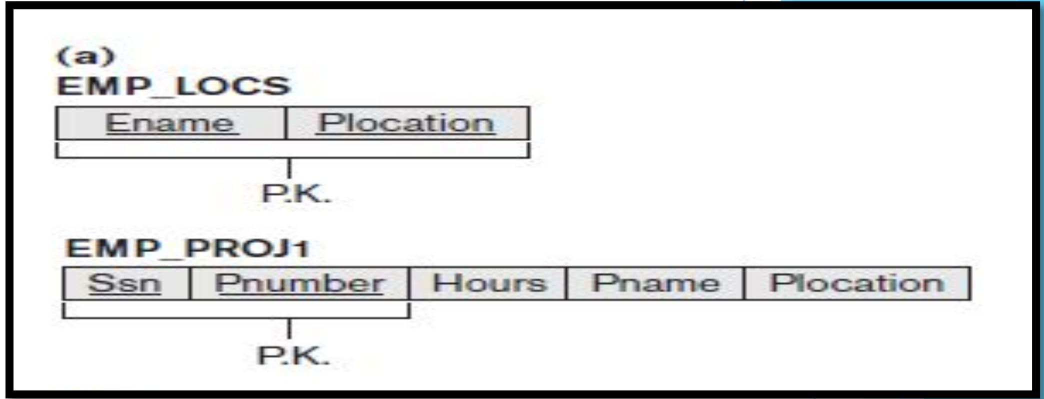
Ename	Plocation
Smith, John B.	Bellaire
Smith, John B.	Sugarland
Narayan, Ramesh K.	Houston
English, Joyce A.	Bellaire
English, Joyce A.	Sugarland
Wong, Franklin T.	Sugarland
Wong, Franklin T.	Houston
Wong, Franklin T.	Stafford
Zelaya, Alicia J.	Stafford
Jabbar, Ahmad V.	Stafford
Wallace, Jennifer S.	Stafford
Wallace, Jennifer S.	Houston
Borg, James E.	Houston

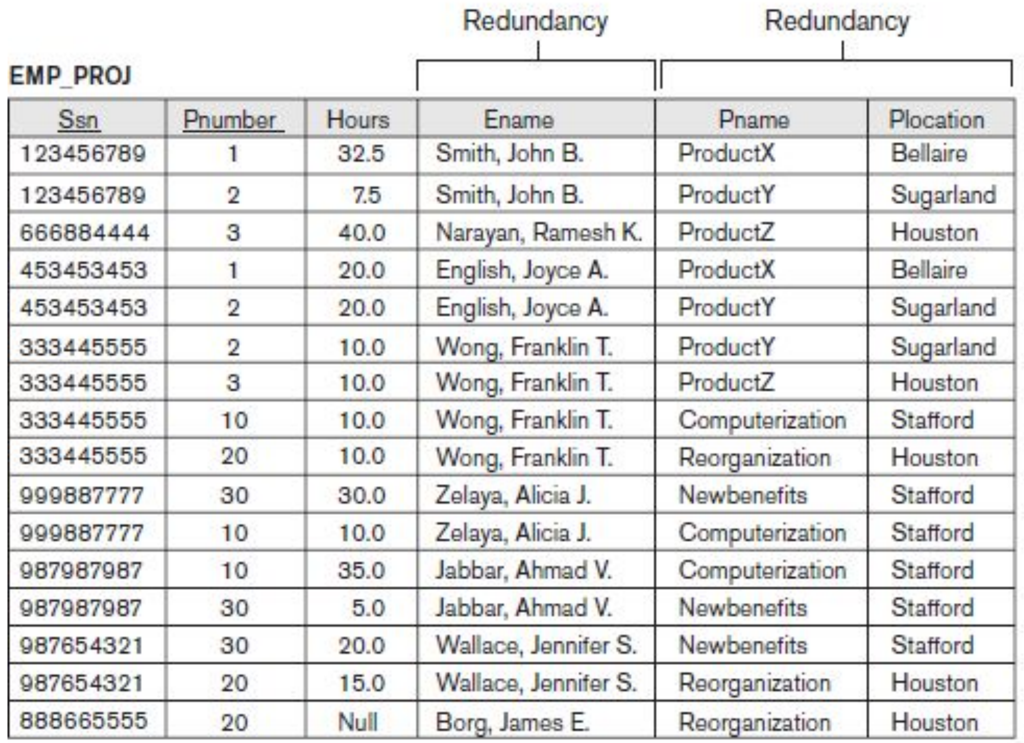
EMP_PROJ1

Ssn	Pnumber	Hours	Pname	Plocation
123456789	1	32.5	ProductX	Bellaire
123456789	2	7.5	ProductY	Sugarland
666884444	3	40.0	ProductZ	Houston
453453453	1	20.0	ProductX	Bellaire
453453453	2	20.0	ProductY	Sugarland
333445555	2	10.0	ProductY	Sugarland
333445555	3	10.0	ProductZ	Houston
333445555	10	10.0	Computerization	Stafford
333445555	20	10.0	Reorganization	Houston
999887777	30	30.0	Newbenefits	Stafford
999887777	10	10.0	Computerization	Stafford
987987987	10	35.0	Computerization	Stafford
987987987	30	5.0	Newbenefits	Stafford
987654321	30	20.0	Newbenefits	Stafford
987654321	20	15.0	Reorganization	Houston
888665555	20	NULL	Reorganization	Houston

Informal Design Guideline for Relation Schemas

- ***Generation of Spurious Tuples***
- Suppose that we used EMP_PROJ1 and EMP_LOCS as the base relations instead of EMP_PROJ.
- This produces a particularly **bad schema design** because we cannot recover the information that was originally in EMP_PROJ from EMP_PROJ1 and EMP_LOCS.
- **If we attempt a NATURAL JOIN operation on EMP_PROJ1 and EMP_LOCS, the result produces many more tuples than the original set of tuples in EMP_PROJ.**



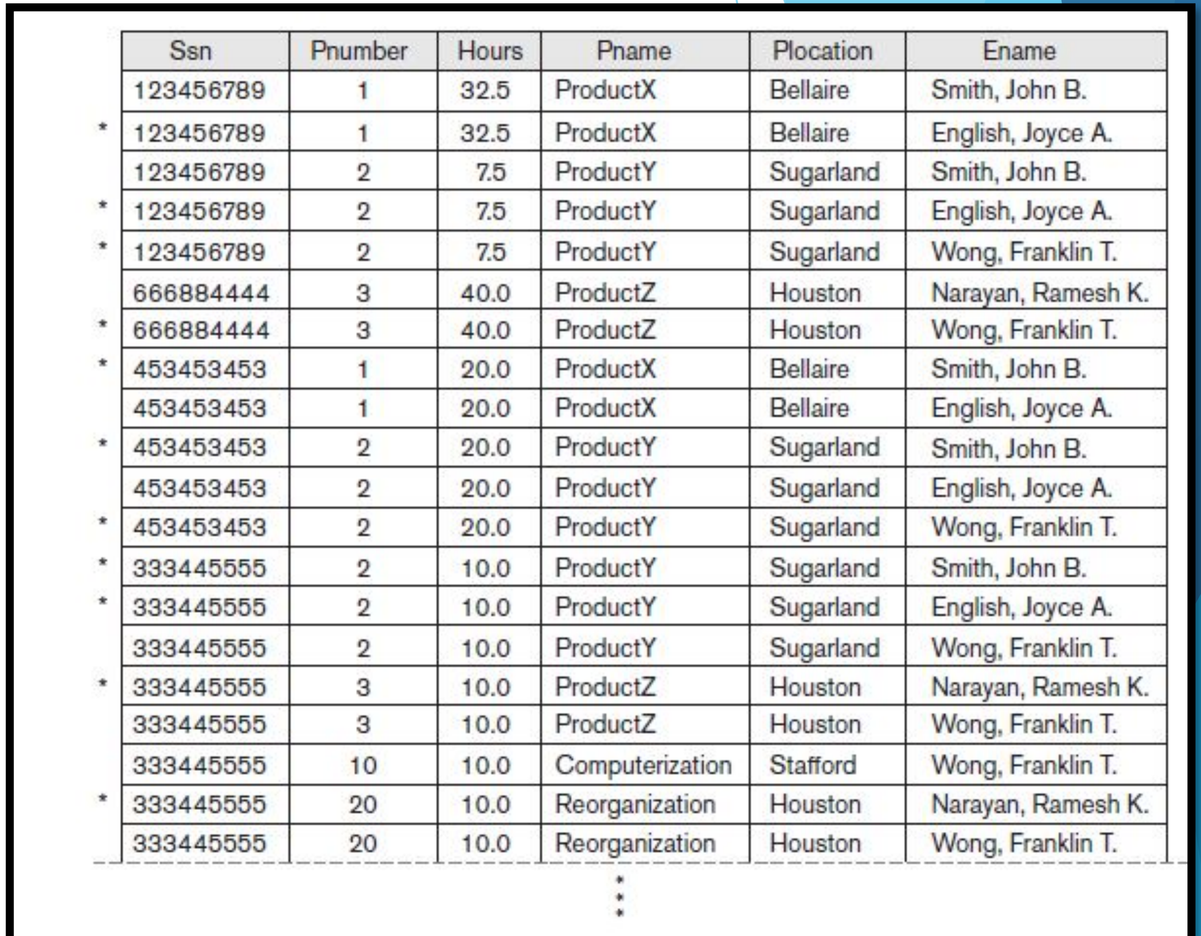

Redundancy
Redundancy

EMP_PROJ

Ssn	Pnumber	Hours	Ename	Pname	Plocation
123456789	1	32.5	Smith, John B.	ProductX	Bellaire
123456789	2	7.5	Smith, John B.	ProductY	Sugarland
666884444	3	40.0	Narayan, Ramesh K.	ProductZ	Houston
453453453	1	20.0	English, Joyce A.	ProductX	Bellaire
453453453	2	20.0	English, Joyce A.	ProductY	Sugarland
333445555	2	10.0	Wong, Franklin T.	ProductY	Sugarland
333445555	3	10.0	Wong, Franklin T.	ProductZ	Houston
333445555	10	10.0	Wong, Franklin T.	Computerization	Stafford
333445555	20	10.0	Wong, Franklin T.	Reorganization	Houston
999887777	30	30.0	Zelaya, Alicia J.	Newbenefits	Stafford
999887777	10	10.0	Zelaya, Alicia J.	Computerization	Stafford
987987987	10	35.0	Jabbar, Ahmad V.	Computerization	Stafford
987987987	30	5.0	Jabbar, Ahmad V.	Newbenefits	Stafford
987654321	30	20.0	Wallace, Jennifer S.	Newbenefits	Stafford
987654321	20	15.0	Wallace, Jennifer S.	Reorganization	Houston
888665555	20	Null	Borg, James E.	Reorganization	Houston

Informal Design Guideline for Relation Schemas

- ***Generation of Spurious Tuples***
- In Figure 14.6, the result of applying the join to only the tuples for employee with Ssn = "123456789" is shown (to reduce the size of the resulting relation).
- **Additional tuples that were not in EMP_PROJ are called spurious tuples because they represent spurious information that is not valid.**
- **The spurious tuples are marked by asterisks (*) in Figure 14.6.**



	Ssn	Pnumber	Hours	Pname	Plocation	Ename
	123456789	1	32.5	ProductX	Bellaire	Smith, John B.
*	123456789	1	32.5	ProductX	Bellaire	English, Joyce A.
	123456789	2	7.5	ProductY	Sugarland	Smith, John B.
*	123456789	2	7.5	ProductY	Sugarland	English, Joyce A.
*	123456789	2	7.5	ProductY	Sugarland	Wong, Franklin T.
	666884444	3	40.0	ProductZ	Houston	Narayan, Ramesh K.
*	666884444	3	40.0	ProductZ	Houston	Wong, Franklin T.
*	453453453	1	20.0	ProductX	Bellaire	Smith, John B.
	453453453	1	20.0	ProductX	Bellaire	English, Joyce A.
*	453453453	2	20.0	ProductY	Sugarland	Smith, John B.
	453453453	2	20.0	ProductY	Sugarland	English, Joyce A.
*	453453453	2	20.0	ProductY	Sugarland	Wong, Franklin T.
*	333445555	2	10.0	ProductY	Sugarland	Smith, John B.
*	333445555	2	10.0	ProductY	Sugarland	English, Joyce A.
	333445555	2	10.0	ProductY	Sugarland	Wong, Franklin T.
*	333445555	3	10.0	ProductZ	Houston	Narayan, Ramesh K.
	333445555	3	10.0	ProductZ	Houston	Wong, Franklin T.
	333445555	10	10.0	Computerization	Stafford	Wong, Franklin T.
*	333445555	20	10.0	Reorganization	Houston	Narayan, Ramesh K.
	333445555	20	10.0	Reorganization	Houston	Wong, Franklin T.

⋮

Figure 14.6
Result of applying NATURAL JOIN to the tuples in EMP_PROJ1 and EMP_LOCS of Figure 14.5 just for employee with Ssn = "123456789". Generated spurious tuples are marked by asterisks.

Informal Design Guideline for Relation Schemas

- ***Generation of Spurious Tuples***
- Decomposing EMP_PROJ into EMP_LOCS and EMP_PROJ1 is undesirable because when we JOIN them back using NATURAL JOIN, we do not get the correct original information.
- This is because in this case **Plocation** happens to be the attribute that relates **EMP_LOCS and EMP_PROJ1**, and Plocation is neither a primary key nor a foreign key in either EMP_LOCS or EMP_PROJ1.
- ***Guideline 4.***
- **Design relation schemas so that they can be joined with equality conditions on attributes that are appropriately related (primary key, foreign key) pairs in a way that guarantees that no spurious tuples are generated.**
- **Join should be done on whole PK projected as FK in another table, not a portion of that.**
- Avoid relations that contain matching attributes that are not (foreign key, primary key) combinations because joining on such attributes may produce spurious tuples.

Informal Design Guideline for Relation Schemas

- ***Summary and Discussion of Design Guidelines***
- **The problems are:**
- ■ Anomalies that cause redundant work to be done during insertion into and modification of a relation, and that may cause accidental loss of information during a deletion from a relation
- ■ Waste of storage space due to NULLs and the difficulty of performing selections, aggregation operations, and joins due to NULL values
- ■ Generation of invalid and spurious data during joins on base relations with matched attributes that may not represent a proper (foreign key, primary key) relationship
- **Strategy** for achieving a **good design:**
  - decompose a badly designed relation appropriately to achieve higher normal forms.

Functional Dependencies

- ***Definition of Functional Dependency***
- A functional dependency is a constraint between two sets of attributes from the database.
- **EXAMPLE:**
- Suppose that our relational database schema has n attributes A1, A2,… , An; let us think of the whole database as being described by a single universal relation schema

  R = {A1, A2, … , An}.
- **Definition:**
  - A functional dependency, denoted by X → Y, between two sets of attributes X and Y that are subsets of R specifies a constraint on the possible tuples that can form a relation state r of R.
  - The constraint is that, for any two tuples t1 and t2 in r that have t1[X] = t2[X], they must also have t1[Y] = t2[Y].

Functional Dependencies

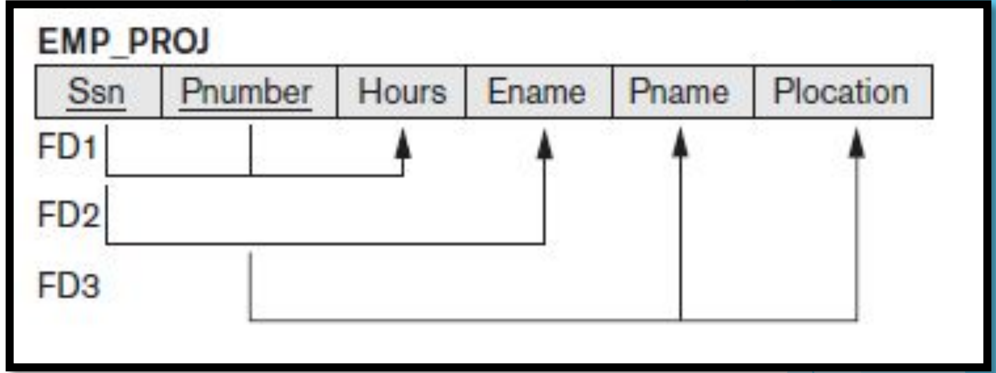
- ***Definition of Functional Dependency***
- **EXAMPLE:**
- This means that the values of the **Y component of a tuple in r depend on, or are determined by, the values of the X component;** alternatively, the values of the X component of a tuple uniquely (or functionally) determine the values of the Y component.
- We also say that there is a **functional dependency from X to Y, or that Y is functionally dependent on X.**
- The abbreviation for functional dependency is **FD or f.d**.
- The set of attributes **X is called the left-hand side** of the FD, and Y is called the **right-hand side**.

Functional Dependencies

- ***Definition of Functional Dependency***
- Note the following:
- If a constraint on R states that there cannot be more than one tuple with a given X-value in any relation instance r(R)—that is, X is a candidate key of **R—this implies that X → Y for any subset of attributes Y of R (because the key constraint implies that no two tuples in any legal state r(R) will have the same value of X).** If X is a candidate key of R, then X → R.
- If X → Y in R, this does not say whether or not Y → X in R.

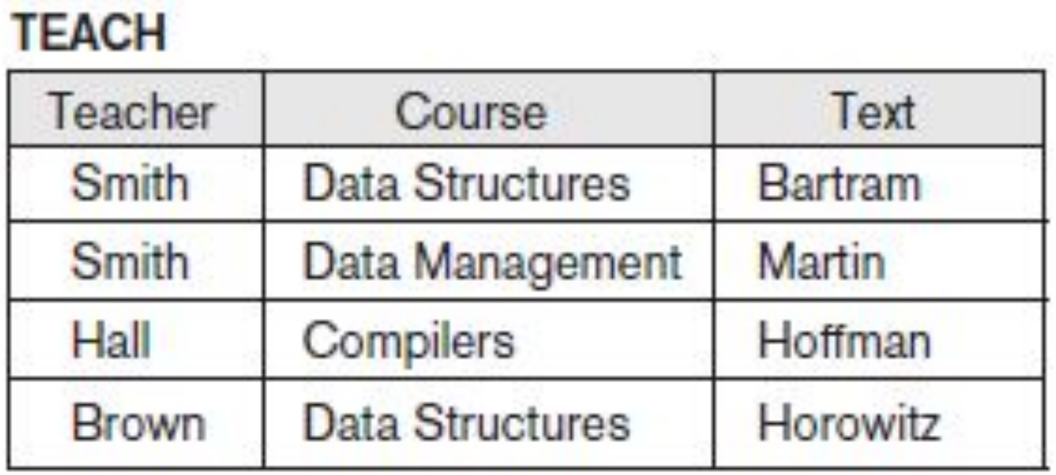
Functional Dependencies

- ***Definition of Functional Dependency***
- Consider the relation schema EMP_PROJ in Figure 14.3(b); from the semantics of the attributes and the relation, we know that the following functional dependencies should hold:
- **a. Ssn → Ename**
- **b. Pnumber → {Pname, Plocation}**
- **c. {Ssn, Pnumber} → Hours**



- These functional dependencies specify that:
  - (a) the value of an employee's Social Security number (Ssn) uniquely determines the employee name (Ename),
  - (b) the value of a project's number (Pnumber) uniquely determines the project name (Pname) and location (Plocation),
  - (c) a combination of Ssn and Pnumber values uniquely determines the number of hours the employee currently works on the project per week (Hours).

Functional Dependencies

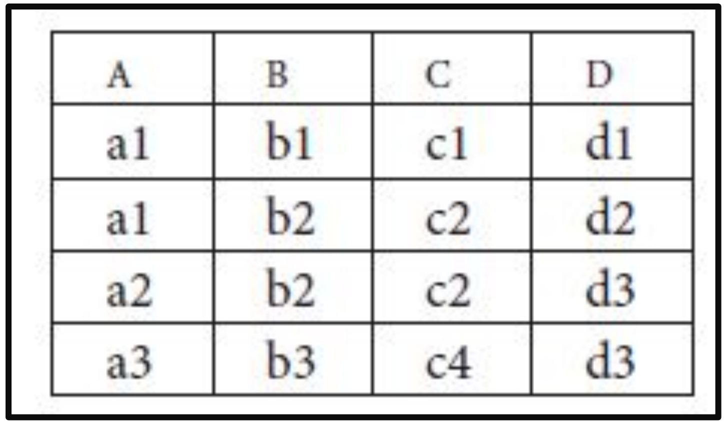

TEACH

Teacher	Course	Text
Smith	Data Structures	Bartram
Smith	Data Management	Martin
Hall	Compilers	Hoffman
Brown	Data Structures	Horowitz

- ***Definition of Functional Dependency***
- A functional dependency is a property of the relation schema R, not of a particular legal relation state r of R.
- For example, Figure 14.7 shows a particular state of the TEACH relation schema.
- We may think that **Text → Course**, we cannot confirm this unless we know that it is true for all possible legal states of TEACH.
- For example, because 'Smith' teaches both 'Data Structures' and 'Database Systems,' we can conclude that **Teacher does not functionally determine Course**.

Functional Dependencies

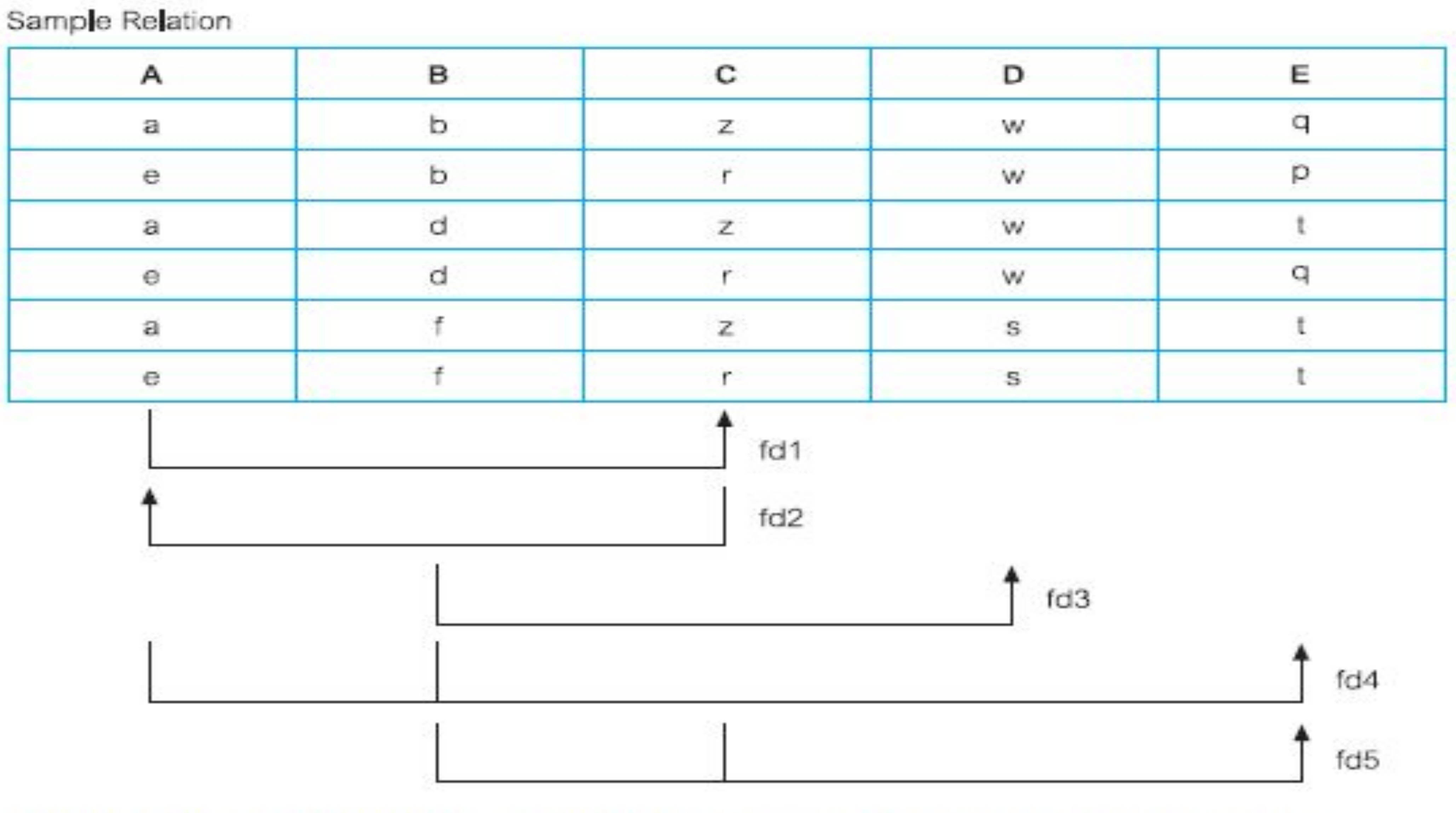
- ***Definition of Functional Dependency***
- Here, the following FDs may hold because the four tuples in the current extension have no violation of these constraints:
- B → C;
- C → B;
- {A, B} → C;
- {A, B} → D;
- {C, D} → B



A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d2
a2	b2	c2	d3
a3	b3	c4	d3

- However, the following do not hold because we already have violations of them in the given extension:
- A → B
- B → A
- D → C

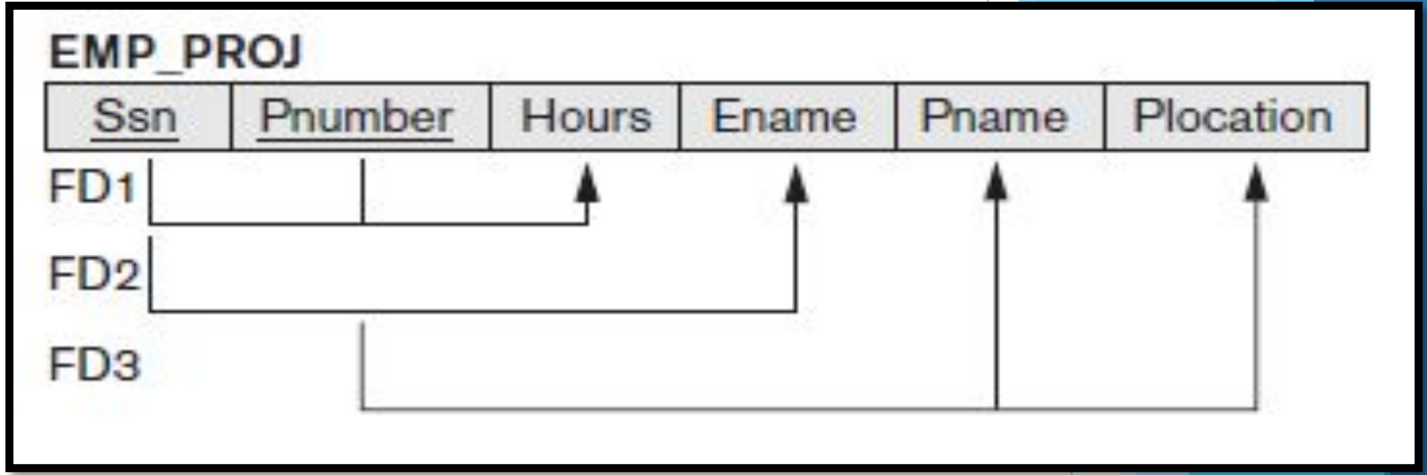
Functional Dependencies


Sample Relation

A	B	C	D	E
a	b	z	w	q
e	b	r	w	p
a	d	z	w	t
e	d	r	w	q
a	f	z	s	t
e	f	r	s	t

Figure 14.6 The Sample relation displaying data for attributes A, B, C, D, and E and the functional dependencies (fd1 to fd5) that exist between these attributes.

Functional Dependencies



- *[Definition of Functional Dependency]*
- Figure 14.3
  - diagrammatic notation for displaying FDs
  - Each FD is displayed as a horizontal line.
- The left-hand-side attributes of the FD are connected by vertical lines to the line representing the FD,
- The right-hand-side attributes are connected by the lines with arrows pointing toward the attributes.

Properties of Functional Dependencies

- ***Reflexivity***
  - *If* X → Y & y is the subset of X , then X → X
  - An attribute determines itself
  - Always valid
  - Trivial FD
- ***Transitivity***
  - *If* (X → Y & Y → Z), then X → Z (might or might not be valid if any one of the condition in if case is false)
- ***Augmentation***
  - *If* (X → Y), then XA → YA
- ***Union***
  - *If* (X → Y & X → Z), then X → YZ
- ***Decomposition***
  - *If* (X → YZ), then X → Y , X → Z
  - Converse is not true
- ***Pseudo Transitivity***
  - *If* (X → Y & YZ → A), then XZ → A
- ***Composition***
  - If (X → Y & A → B), then XA → YB

Roll No	Name	Marks	Department	Course
1	Maryam	78	CS	OOP
2	Maira	60	AI	OOP
3	Maryam	78	CS	DB
4	Maira	60	AI	ML
5	Mohammad	80	SE	DB

Normal Forms Based on Primary Keys

- ***Normalization of Relations***
- The normalization process, as first proposed by Codd (1972), **takes a relation schema through a series of tests to certify whether it satisfies a certain normal form.**
- The process, which proceeds in a **top-down** fashion by evaluating each relation against the criteria for normal forms and decomposing relations as necessary.
- Initially, Codd proposed three normal forms, which he called first, second, and third normal form.
- **A stronger definition of 3NF—called Boyce-Codd normal form (BCNF)**—was proposed later by Boyce and Codd.
- All these normal forms are based on a single analytical tool: the **functional dependencies among the attributes of a relation.**

Normal Forms Based on Primary Keys

- ***Normalization of Relations***
- **Normalization of data is a process of analyzing the given relation schemas based on their FDs and primary keys to achieve the desirable properties of**
  - **(1) minimizing redundancy and**
  - **(2) minimizing the insertion, deletion, and update anomalies.**
- An unsatisfactory relation schema that does not meet the condition for a normal form—the normal form test—is decomposed into smaller relation schemas that contain a subset of the attributes and meet the test that was otherwise not met by the original relation.

Normal Forms Based on Primary Keys

- ***Normalization of Relations***
- Definition:
  - The normal form of a relation is the highest normal form condition that a relation meets,
  - and hence indicates the degree to which it has been normalized.
- Existing designs are evaluated by applying the tests for normal forms, and normalization is carried out in practice so that the resulting designs are of high quality and meet the desirable properties stated previously.
- **Denormalization is a technique used by database administrators to optimize the efficiency of their database infrastructure. This method allows us to add redundant data into a normalized database to alleviate issues with database queries that merge data from several tables into a single table.**

Normal Forms Based on Primary Keys

- ***Definitions of Keys and Attributes Participating in Keys***
- Definition: A **superkey** of a relation schema R = {A1, A2, ... , An} is a set of attributes S ⊆ R with the property that no two tuples t1 and t2 in any legal relation state r of R will have t1[S] = t2[S].
- A **key K is a superkey** with the additional property that removal of any attribute from K will cause K not to be a super key anymore.
- The difference between a key and a super key is that a key has to be minimal.
- **{Ssn} is a key** for EMPLOYEE,
- whereas **{Ssn}, {Ssn, Ename}, {Ssn, Ename, Bdate}**, and any set of attributes that includes Ssn are all superkeys.

Normal Forms Based on Primary Keys

- ***Definitions of Keys and Attributes Participating in Keys***
- If a relation schema has more than one key, each is called a **candidate key**.
- One of the candidate keys is arbitrarily designated to be the primary key, and the others are called **secondary keys.**
- In a practical relational database, each relation schema must have a primary key.
- **If no candidate key is known for a relation, the entire relation can be treated as a default superkey.**
- {Ssn} is the only candidate key for EMPLOYEE, so it is also the primary key.

Normal Forms Based on Primary Keys

- ***Definitions of Keys and Attributes Participating in Keys***
- Definition. An attribute of relation schema R is called a **prime attribute** of R if it is a member of some candidate key of R.
  - For example, {SSn,Ename} is a CK in R, then prime attributes are SSn and Ename.
- An attribute is called **nonprime** if it is not a prime attribute—that is, if it is not a member of any candidate key.

Identifying SK and CK in a relation

- **Closure set / Attribute closure to find CK:**
  - Represented as X^+
  - X could be an attribute or a set of attributes
  - **SUPERKEY:** whose closure set contains all attributes of a relation R
    - Find closure set
    - If the closure set contains all attributes then it is a SK
  - **Candidate Key:** a key whose proper subset is not a SK
    - find proper subsets of a SK
    - and check if the closure sets of these are not SK then
    - It's a candidate key
  - Question:
    - R(A,B,C,D,E)
    - FD ($A \rightarrow B$, $B \rightarrow C$, $C \rightarrow D$, $D \rightarrow E$)

Identifying SK and CK in a relation

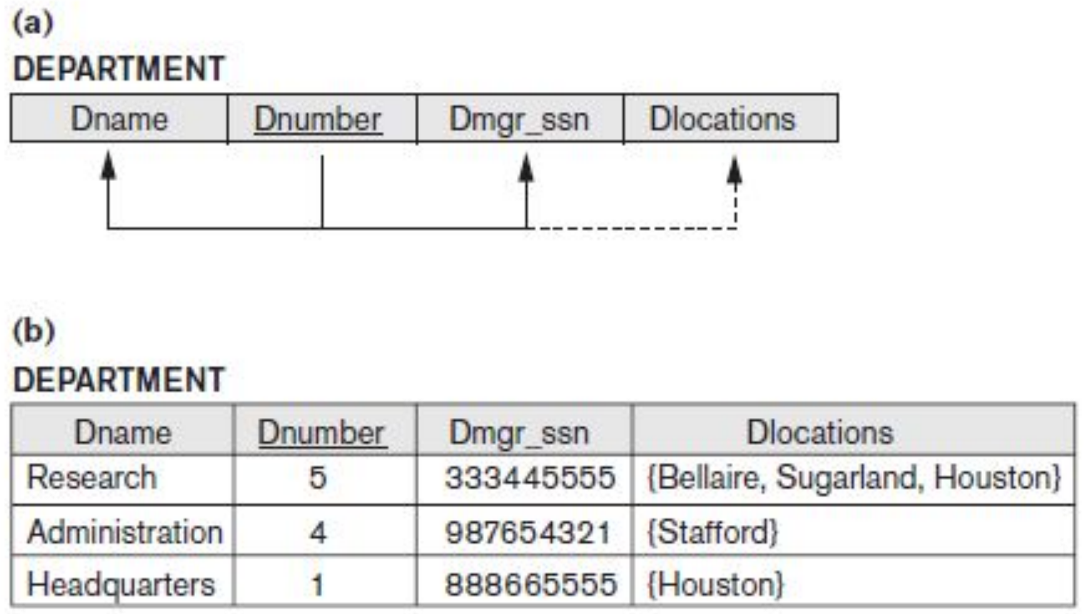
- **How to find SK and CK:**
  - Question:
    - R(A,B,C,D,E)
    - FD ($A \rightarrow B$, $D \rightarrow E$)
  - STEPS:
    - Find closures sets
    - Identify SK
    - Check if it's a CK
      - find proper subsets of a SK
      - and check if the closure sets of these are not SK then
      - It's a candidate key

Normal Forms Based on Primary Keys

- ***First Normal Form***
- It states that the **domain of an attribute must include only atomic (simple, indivisible) values** and that the value of any attribute in a tuple must be a single value from the domain of that attribute.
- Hence, **1NF disallows having a set of values**, a tuple of values, or a combination of both as an attribute value for a single tuple.
- The only attribute values permitted by 1NF are **single atomic (or indivisible) values**.

Normal Forms Based on Primary Keys

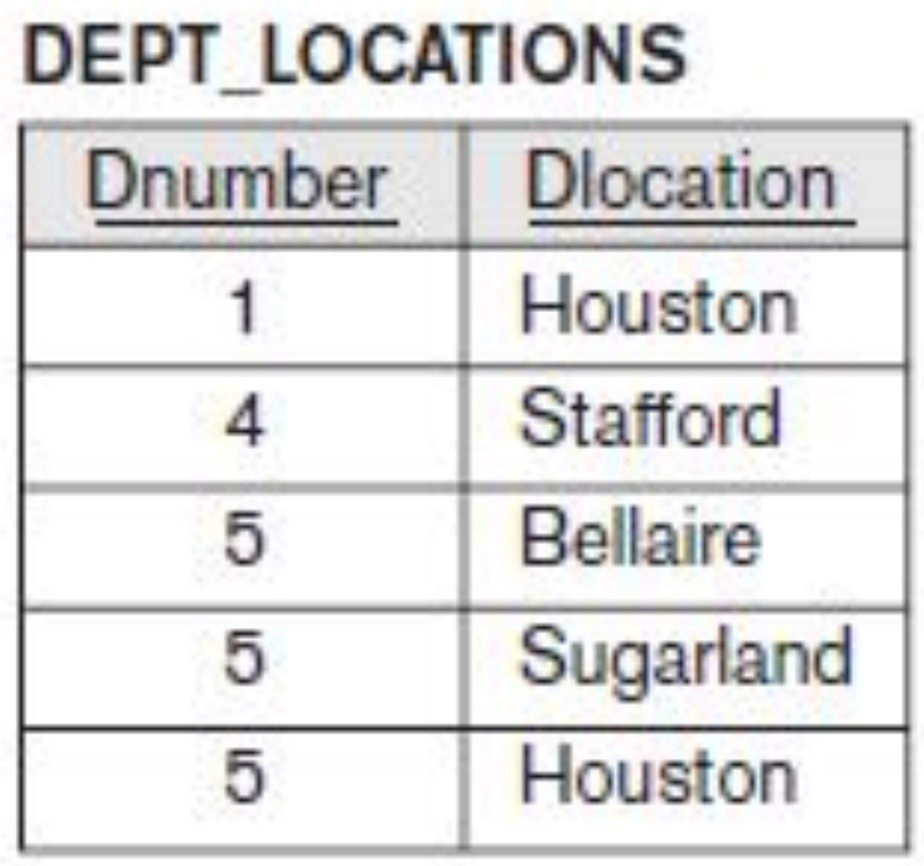
- ***First Normal Form***
- Consider the DEPARTMENT relation schema shown in Figure 14.1, whose primary key is Dnumber, and suppose that we extend it by including the Dlocations attribute as shown in Figure 14.9(a).
- We assume that each department can have a number of locations.
- As we can see, this is not in 1NF because Dlocations is not an atomic attribute, as illustrated by the first tuple in Figure 14.9(b).



Dname	Dnumber	Dmgr_ssn	Dlocations
Research	5	333445555	{Bellaire, Sugarland, Houston}
Administration	4	987654321	{Stafford}
Headquarters	1	888665555	{Houston}

Normal Forms Based on Primary Keys

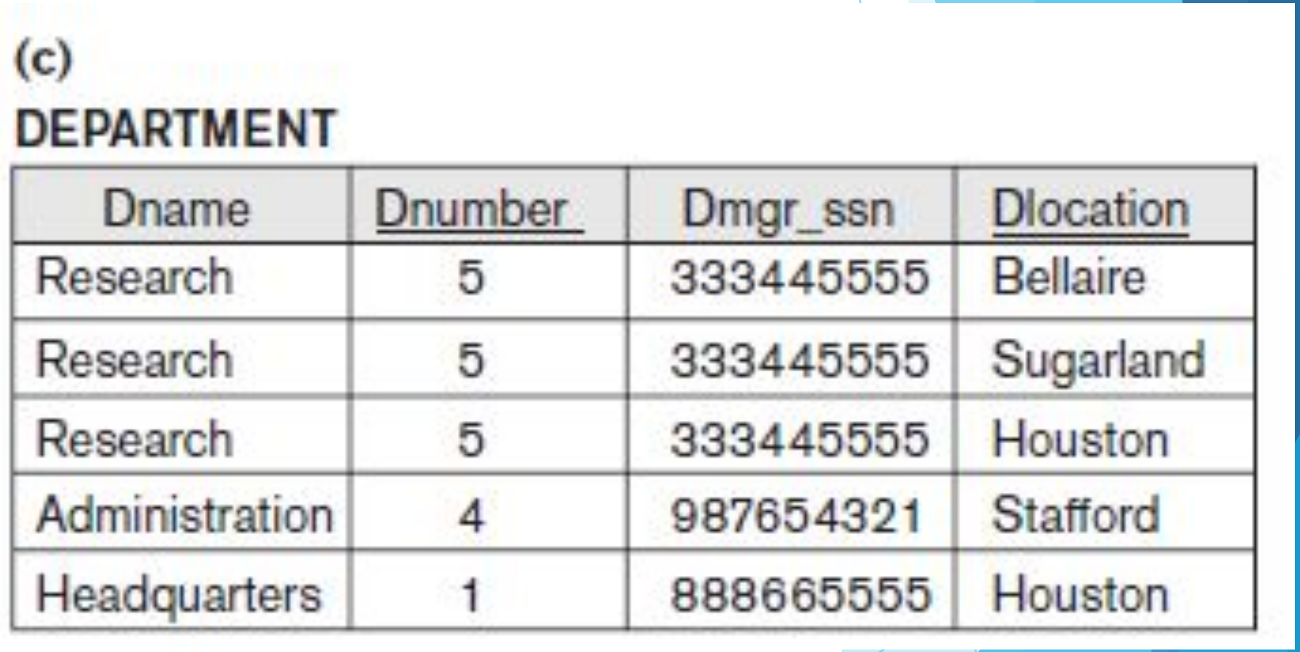
- ***First Normal Form***
- ***FIRST OPTION:***
- **Remove the attribute Dlocations that violates 1NF and place it in a separate relation DEPT_LOCATIONS along with the primary key Dnumber of DEPARTMENT.**
- The primary key of this newly formed relation is the combination {Dnumber, Dlocation}, as shown in Figure 14.2.
- A distinct tuple in DEPT_LOCATIONS exists for each location of a department.
- This decomposes the non-1NF relation into two 1NF relations.


DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

Normal Forms Based on Primary Keys

- ***First Normal Form***
- ***SECOND OPTION:***
- **Expand the key so that there will be a separate tuple in the original DEPARTMENT relation for each location of a DEPARTMENT.**
- In this case, the primary key becomes the combination {Dnumber, Dlocation}.
- This solution has the disadvantage of **introducing redundancy** in the relation which leads to updated anomalies and hence is rarely adopted.


(c)

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocation
Research	5	333445555	Bellaire
Research	5	333445555	Sugarland
Research	5	333445555	Houston
Administration	4	987654321	Stafford
Headquarters	1	888665555	Houston

Normal Forms Based on Primary Keys

- ***First Normal Form***
- ***THIRD OPTION:***
- **If a maximum number of values is known for the attribute—for example, if it is known that at most three locations can exist for a department—replace the Dlocations attribute by three atomic attributes: Dlocation1, Dlocation2, and Dlocation3.**
- **Disadvantage:**
  - **NULL values ~** if most departments have fewer than three locations.
  - **Querying on this attribute becomes more difficult;**
    - for example, consider how you would write the query: List the departments that have 'Bellaire' as one of their locations in this design.
- For all these reasons, it **is best to avoid this alternative.**

Normal Forms Based on Primary Keys

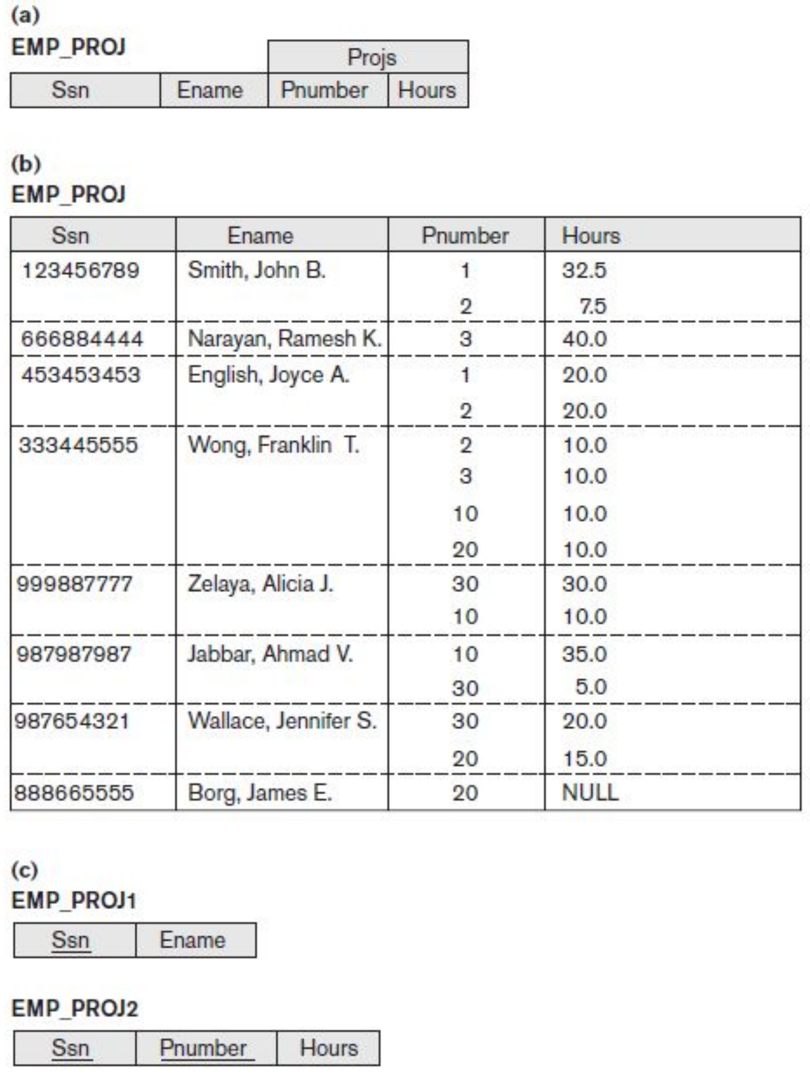
- ***First Normal Form***
- ***THIRD OPTION:***
- Of the three solutions above, **the first is generally considered best because it does not suffer from redundancy and it is completely general; it places no maximum limit on the number of values.**
- In fact, **if we choose the second solution, it will be decomposed further during subsequent normalization steps into the first solution**.

Normal Forms Based on Primary Keys

- ***First Normal Form***
- ***THIRD OPTION:***
- 1NF also disallows multivalued attributes that are themselves composite.
- These are called **nested relations** because each tuple can have a relation within it.
- Figure 14.10 shows how the EMP_PROJ relation could appear if nesting is allowed.
- In EMP_Proj -> shows each employee entity with the projects on which he/she is working.
- The schema of this EMP_PROJ relation can be represented as follows:
- EMP_PROJ(Ssn, Ename, {PROJS(Pnumber, Hours)})
- The set braces { } identify the attribute PROJS as multivalued, and we list the component attributes that form PROJS between parentheses ( ).

Normal Forms Based on Primary Keys

- ***First Normal Form***
- Ssn is the primary key of the EMP_PROJ relation in Figures 14.10(a) and (b),
- whereas Pnumber is the partial key of the nested relation; that is, within each tuple, the nested relation must have unique values of Pnumber.
- Figure 14.10 (c) shows the solution.
- To normalize this into 1NF, we remove the nested relation attributes into a new relation and propagate the primary key into it; the primary key of the new relation will combine the partial key with the primary key of the original relation.



(a)

EMP_PROJ

		Projs	
Ssn	Ename	Pnumber	Hours

(b)

EMP_PROJ

Ssn	Ename	Pnumber	Hours
123456789	Smith, John B.	1	32.5
		2	7.5
666884444	Narayan, Ramesh K.	3	40.0
453453453	English, Joyce A.	1	20.0
		2	20.0
333445555	Wong, Franklin T.	2	10.0
		3	10.0
		10	10.0
		20	10.0
999887777	Zelaya, Alicia J.	30	30.0
		10	10.0
987987987	Jabbar, Ahmad V.	10	35.0
		30	5.0
987654321	Wallace, Jennifer S.	30	20.0
		20	15.0
888665555	Borg, James E.	20	NULL

(c)

EMP_PROJ1

Ssn	Ename

EMP_PROJ2

Ssn	Pnumber	Hours

Figure 14.10
Normalizing nested relations into 1NF.
(a) Schema of the EMP_PROJ relation with a *nested relation* attribute PROJS. (b) Sample extension of the EMP_PROJ relation showing nested relations within each tuple.
(c) Decomposition of EMP_PROJ into relations EMP_PROJ1 and EMP_PROJ2 by propagating the primary key.

Normal Forms Based on Primary Keys

- ***First Normal Form***
- This procedure can be applied recursively to a relation with multiple-level nesting to un nest the relation into a set of 1NF relations.
- This is useful in converting an un normalized relation schema with many levels of nesting into 1NF relations.
- **As an example, consider the following:**
- **CANDIDATE (Ssn, Name, {JOB_HIST (Company, Highest_position, {SAL_HIST (Year, Max_sal)})})**
- **The first normalization using internal partial keys Company and Year,respectively, results in the following 1NF relations:**
- **CANDIDATE_1 (Ssn, Name)**
- **CANDIDATE_JOB_HIST (Ssn, Company, Highest_position)**
- **CANDIDATE_SAL_HIST (Ssn, Company, Year, Max-sal)**

Normal Forms Based on Primary Keys

- ***Second Normal Form***
- Second normal form (2NF) is based on the concept of **full functional dependency**.
- A functional dependency X → Y is a **full functional dependency** if removal of any attribute A from X means that the dependency does not hold anymore; that is, for any attribute A ε X, (X − {A}) does not functionally determine Y.
  - The dependency {Ssn, Pnumber} → Hours is a full dependency
    - (neither Ssn → Hours nor Pnumber → Hours holds).
- A functional dependency X → Y is a **partial dependency** if some attribute A ε X can be removed from X and the dependency still holds; that is, for some A ε X, (X − {A}) → Y.
  - The dependency {Ssn, Pnumber} → Ename is partial
    - because Ssn → Ename holds.
- **Definition:**
  - **A relation schema R is in 2NF if every nonprime attribute A in R is fully functionally dependent on the primary key of R. ~ NO PARTIAL DEPENDENCY**
  - **And table should be in 1NF.**

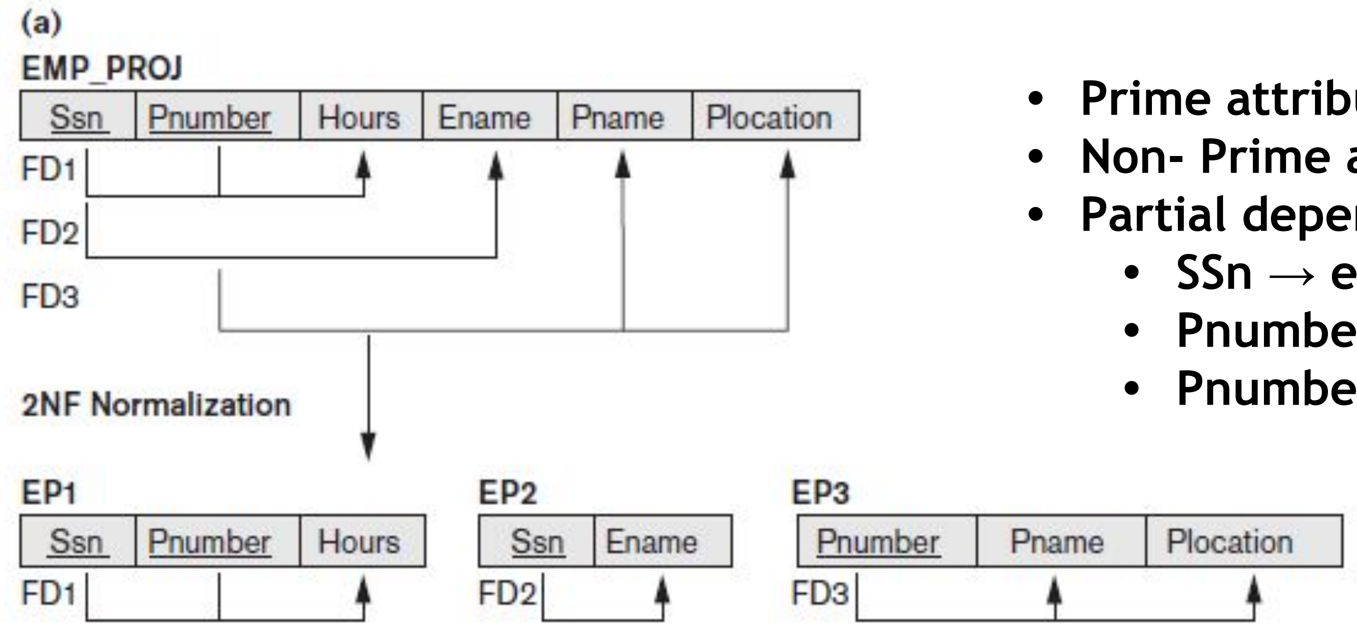
Normal Forms Based on Primary Keys

- ***Second Normal Form***
- Second normal form (2NF) is based on the concept of **full functional dependency**.
- A functional dependency X → Y is a **full functional dependency** if removal of any attribute A from X means that the dependency does not hold anymore; that is, for any attribute A ε X, (X − {A}) does not functionally determine Y.
  - The dependency {Ssn, Pnumber} → Hours is a full dependency
    - (neither Ssn → Hours nor Pnumber → Hours holds).
- A functional dependency X → Y is a **partial dependency** if some attribute A ε X can be removed from X and the dependency still holds; that is, for some A ε X, (X − {A}) → Y.
  - The dependency {Ssn, Pnumber} → Ename is partial
    - because Ssn → Ename holds.
- **Definition:**
  - **A relation schema R is in 2NF if every nonprime attribute A in R is fully functionally dependent on the primary key of R. ~ NO PARTIAL DEPENDENCY**
  - **And table should be in 1NF.**

CHECK for 2NF

- **Check for Partial depndencies**
  - Question:
    - R(A,B,C,D,E)
    - FD ($A \rightarrow B$, $D \rightarrow E$)
  - STEPS:
    - Find CK
      - Check if non prime attributes are determined by part of Ck
  - Solution:
    - Prime attributes:
    - Non-prime attributes:
    - CK = ACD
      - Check if B or E are determined by either A / C / D / AC / CD / AD
      - only then it's a partial dependency.
    - If Partial dependency exists, then decompose the relation

Normal Forms Based on Primary Keys



- **Prime attribute:** SSN,Pnumber
- **Non- Prime attribute:** hours, ename, pname, plocation
- **Partial dependencies:**
  - **SSn → ename ,**
  - **Pnumber → Pname,**
  - **Pnumber → Plocation**

- ***Second Normal Form***
- The test for 2NF involves
  - testing for FD whose left-hand side attributes are part of the primary key.
- The EMP_PROJ relation in Figure 14.3(b) is in 1NF but is not in 2NF.
- The nonprime attribute Ename violates 2NF because of FD2, as do the nonprime attributes Pname and Plocation because of FD3.

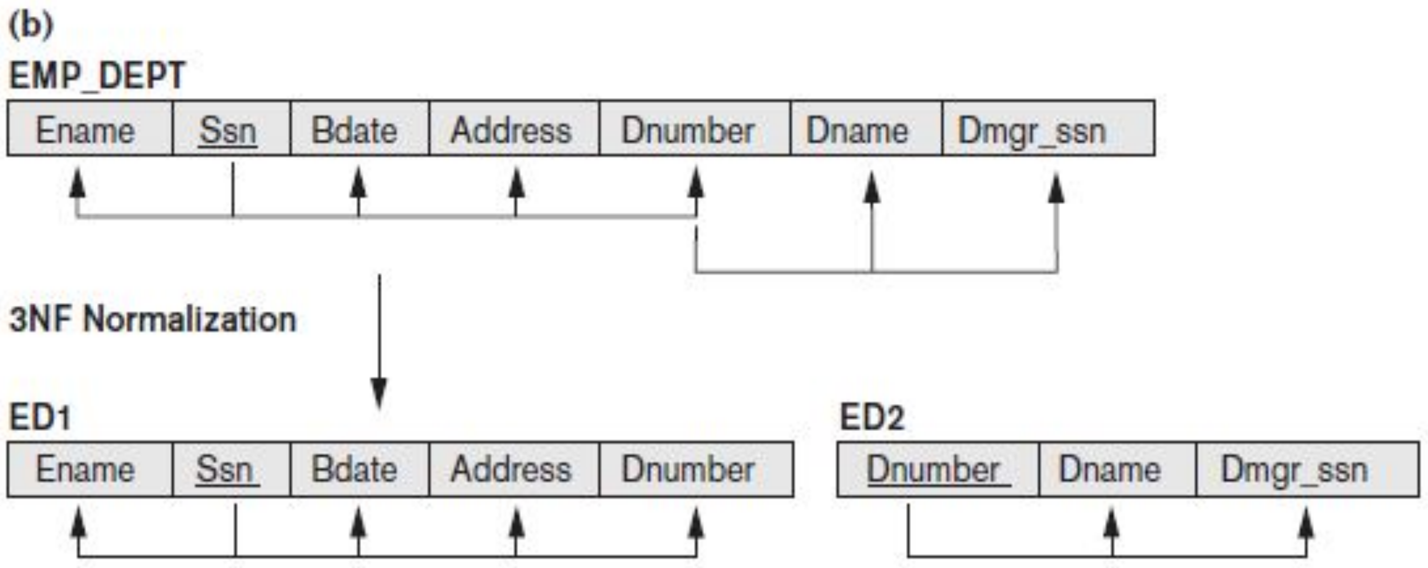
Normal Forms Based on Primary Keys

- ***Third Normal Form***
- Third normal form (3NF) is based on the concept of **transitive dependency**.
- A functional dependency X → Y in a relation schema R is a transitive dependency if there exists a set of attributes Z in R that is neither a candidate key nor a subset of any key of R, and both X → Z and Z → Y hold.
- Transitive dependencies in **EMP_DEPT(SSN, Ename, Dnumber, Dname, Dlocations, Dmgr_SSn)** could be :
  - Ssn → Dnumber and Dnumber → Dmgr_ssn hold and Dnumber is neither a key itself nor a subset of the key of EMP_DEPT.
  - The dependency Ssn → Dmgr_ssn is transitive through Dnumber in EMP_DEPT because both the dependencies, and Dnumber is not a prime attribute.
- **Definition: A relation schema R is in 3NF if**
  - **it satisfies 2NF and**
  - **no nonprime attribute of R is transitively dependent on the primary key. (or no non prime attribute is determined by a non prime attribute)**

CHECK for 3NF

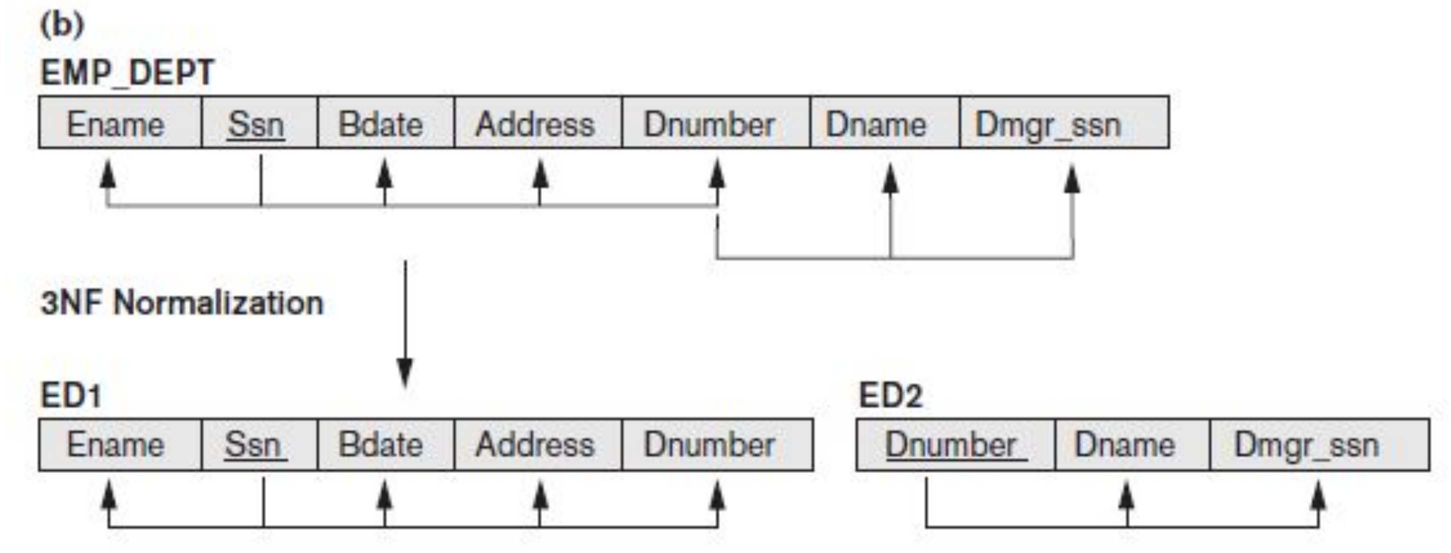
- **Check for Transitive dependencies**
  - Question:
    - R(A,B,C,D,E,F)
    - FD ($AB \rightarrow CDEF$, $BD \rightarrow F$)
  - STEPS:
    - Find CK
    - Then identify if any non prime attribute is determined by a non prime attribute.
  - Solution:
    - Prime attributes:
    - Non-prime attributes:
    - CK = AB
      - Check if non prime attributes are determined by any non-prime attribute except AB
      - only then it's a transitive dependency.
    - If transitive dependencies exist, then decompose the relation

Normal Forms Based on Primary Keys



- **Prime attribute:** SSN
- **Non- Prime attribute:** ename,bdate,address,dnumber,dname,Dmgr_SSN
- **dependencies:**
  - SSn → Ename, Bdate, Address, Dnumber
  - Dnumber → Dname, Dmgr_SSN

Normal Forms Based on Primary Keys



- ***Third Normal Form***
- A NATURAL JOIN operation on ED1 and ED2 will recover the original relation EMP_DEPT without generating spurious tuples.

General Definitions of Second and Third Normal Forms

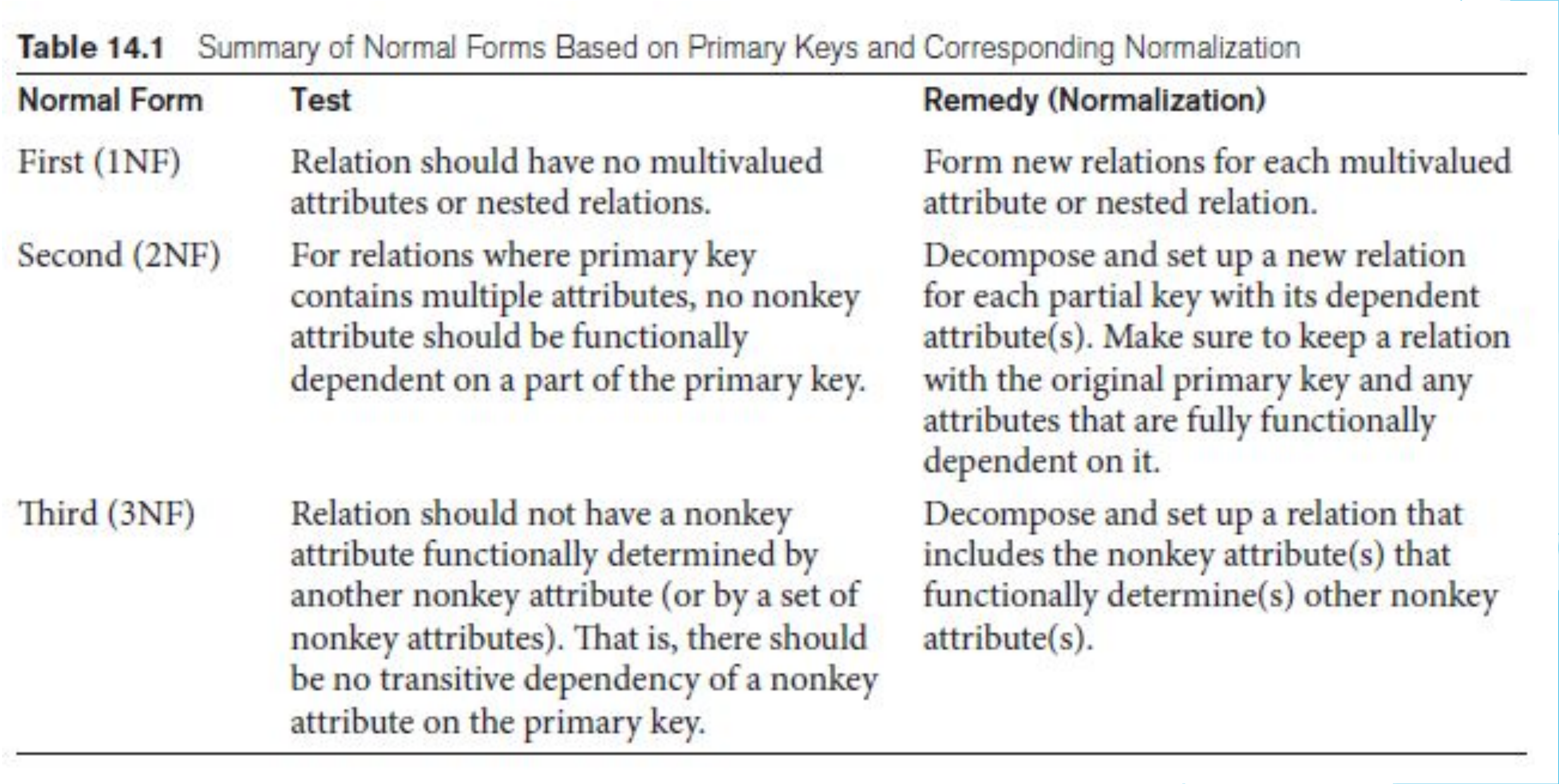
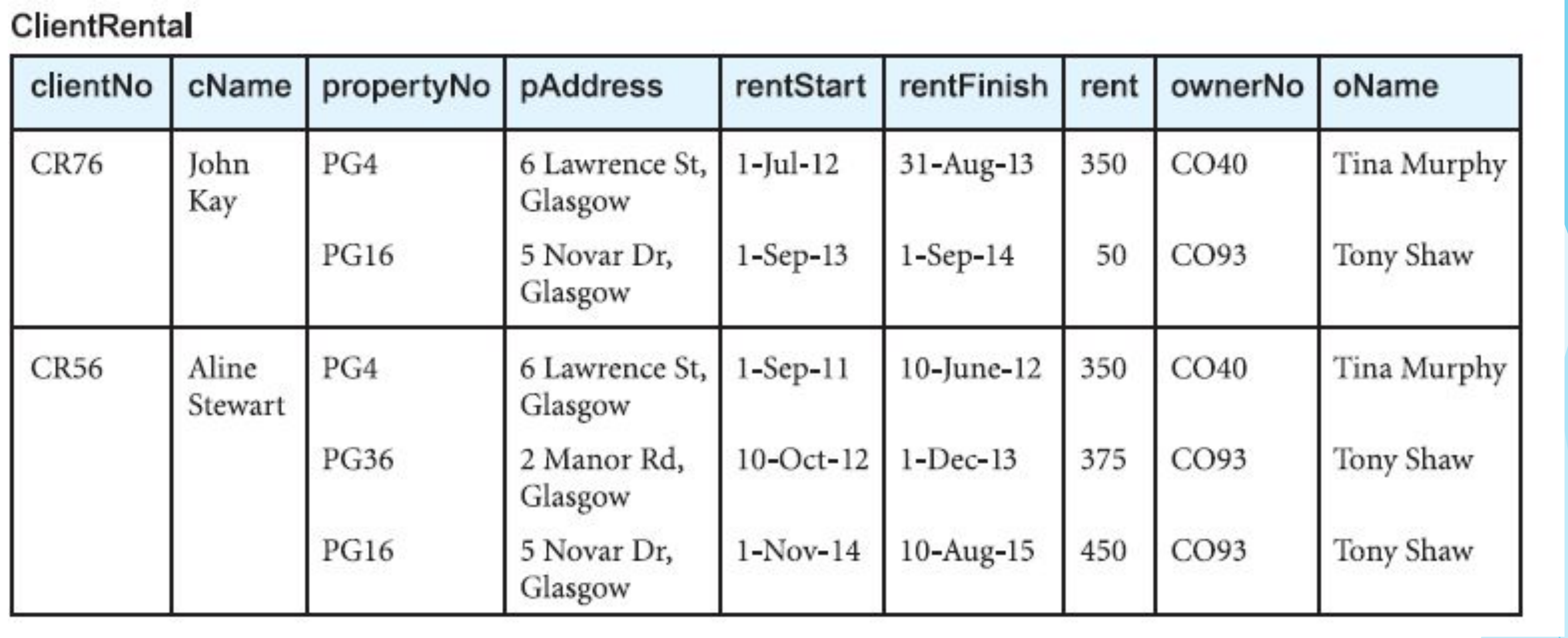


Table 14.1 Summary of Normal Forms Based on Primary Keys and Corresponding Normalization

Normal Form	Test	Remedy (Normalization)
First (1NF)	Relation should have no multivalued attributes or nested relations.	Form new relations for each multivalued attribute or nested relation.
Second (2NF)	For relations where primary key contains multiple attributes, no nonkey attribute should be functionally dependent on a part of the primary key.	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.
Third (3NF)	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.	Decompose and set up a relation that includes the nonkey attribute(s) that functionally determine(s) other nonkey attribute(s).

Normal Forms Based on Primary Keys

- ***Example:***



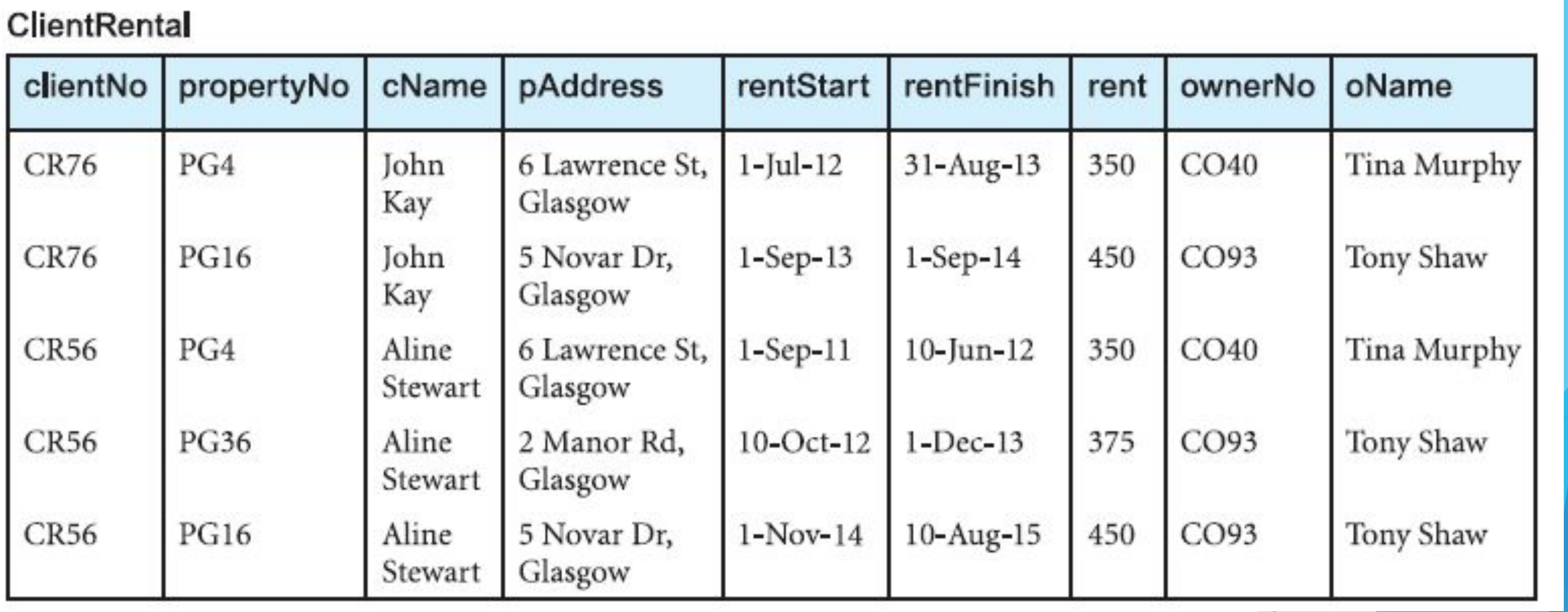
ClientRental

clientNo	cName	propertyNo	pAddress	rentStart	rentFinish	rent	ownerNo	oName
CR76	John Kay	PG4	6 Lawrence St, Glasgow	1-Jul-12	31-Aug-13	350	CO40	Tina Murphy
		PG16	5 Novar Dr, Glasgow	1-Sep-13	1-Sep-14	50	CO93	Tony Shaw
CR56	Aline Stewart	PG4	6 Lawrence St, Glasgow	1-Sep-11	10-June-12	350	CO40	Tina Murphy
		PG36	2 Manor Rd, Glasgow	10-Oct-12	1-Dec-13	375	CO93	Tony Shaw
		PG16	5 Novar Dr, Glasgow	1-Nov-14	10-Aug-15	450	CO93	Tony Shaw

- A client can rent multiple properties of different owners
- Key attribute is clientNo.
- **Repeating Group = propertyNo, pAddress, rentStart, rentFinish, rent, ownerNo, oName**

Normal Forms Based on Primary Keys

- ***Example:***
- **1NF - 2 APPROACHES**
- **FIRST APPROACH**



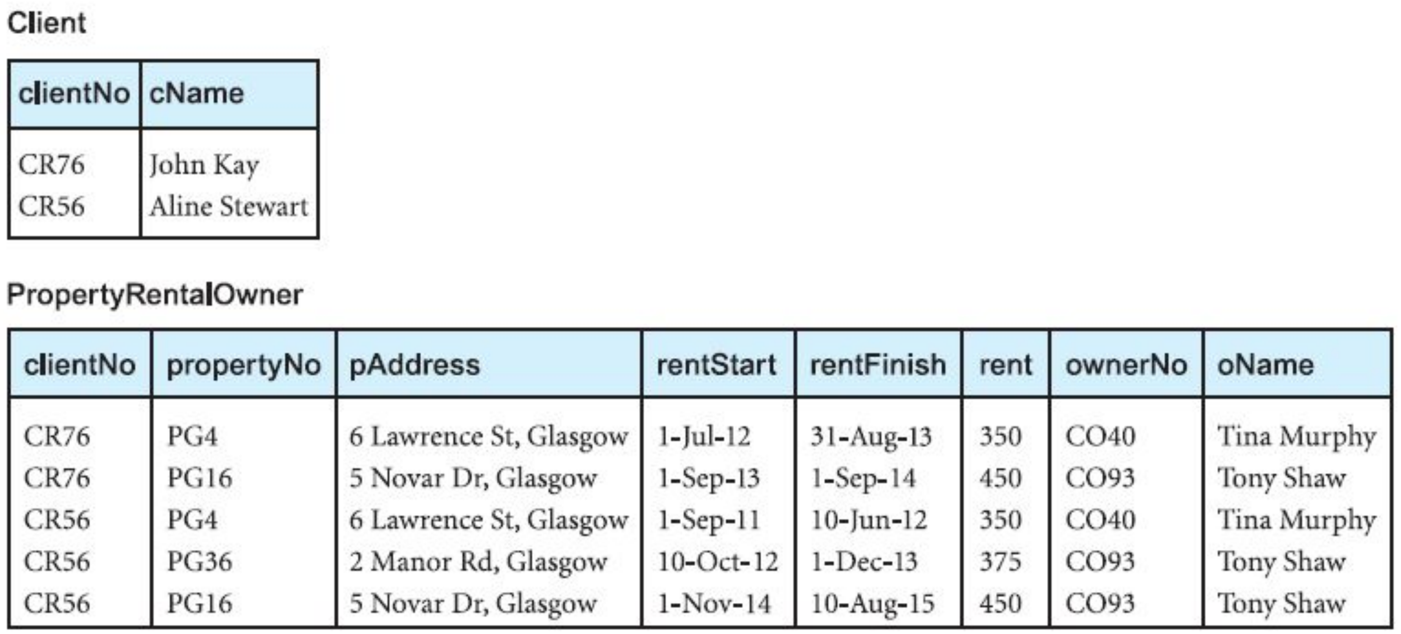
ClientRental

clientNo	propertyNo	cName	pAddress	rentStart	rentFinish	rent	ownerNo	oName
CR76	PG4	John Kay	6 Lawrence St, Glasgow	1-Jul-12	31-Aug-13	350	CO40	Tina Murphy
CR76	PG16	John Kay	5 Novar Dr, Glasgow	1-Sep-13	1-Sep-14	450	CO93	Tony Shaw
CR56	PG4	Aline Stewart	6 Lawrence St, Glasgow	1-Sep-11	10-Jun-12	350	CO40	Tina Murphy
CR56	PG36	Aline Stewart	2 Manor Rd, Glasgow	10-Oct-12	1-Dec-13	375	CO93	Tony Shaw
CR56	PG16	Aline Stewart	5 Novar Dr, Glasgow	1-Nov-14	10-Aug-15	450	CO93	Tony Shaw

- Client no and property no is a composite key here
- The relation contains data redundancy.
- If implemented, the 1NF relation would be subject to the update anomalies.

Normal Forms Based on Primary Keys

- ***Example:***
- **1NF - 2 APPROACHES**
- **SECOND APPROACH**



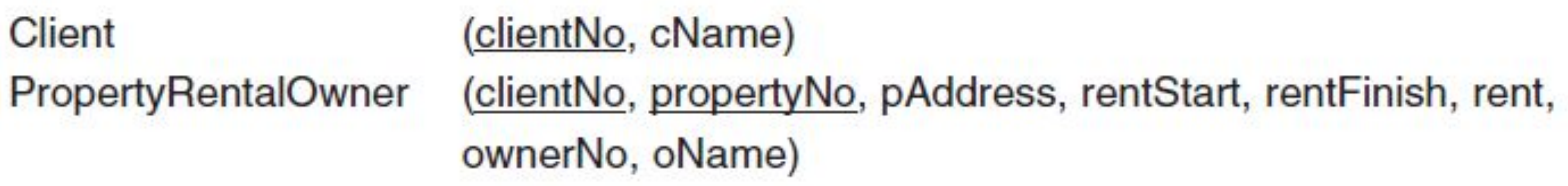
Client

clientNo	cName
CR76	John Kay
CR56	Aline Stewart

PropertyRentalOwner

clientNo	propertyNo	pAddress	rentStart	rentFinish	rent	ownerNo	oName
CR76	PG4	6 Lawrence St, Glasgow	1-Jul-12	31-Aug-13	350	CO40	Tina Murphy
CR76	PG16	5 Novar Dr, Glasgow	1-Sep-13	1-Sep-14	450	CO93	Tony Shaw
CR56	PG4	6 Lawrence St, Glasgow	1-Sep-11	10-Jun-12	350	CO40	Tina Murphy
CR56	PG36	2 Manor Rd, Glasgow	10-Oct-12	1-Dec-13	375	CO93	Tony Shaw
CR56	PG16	5 Novar Dr, Glasgow	1-Nov-14	10-Aug-15	450	CO93	Tony Shaw

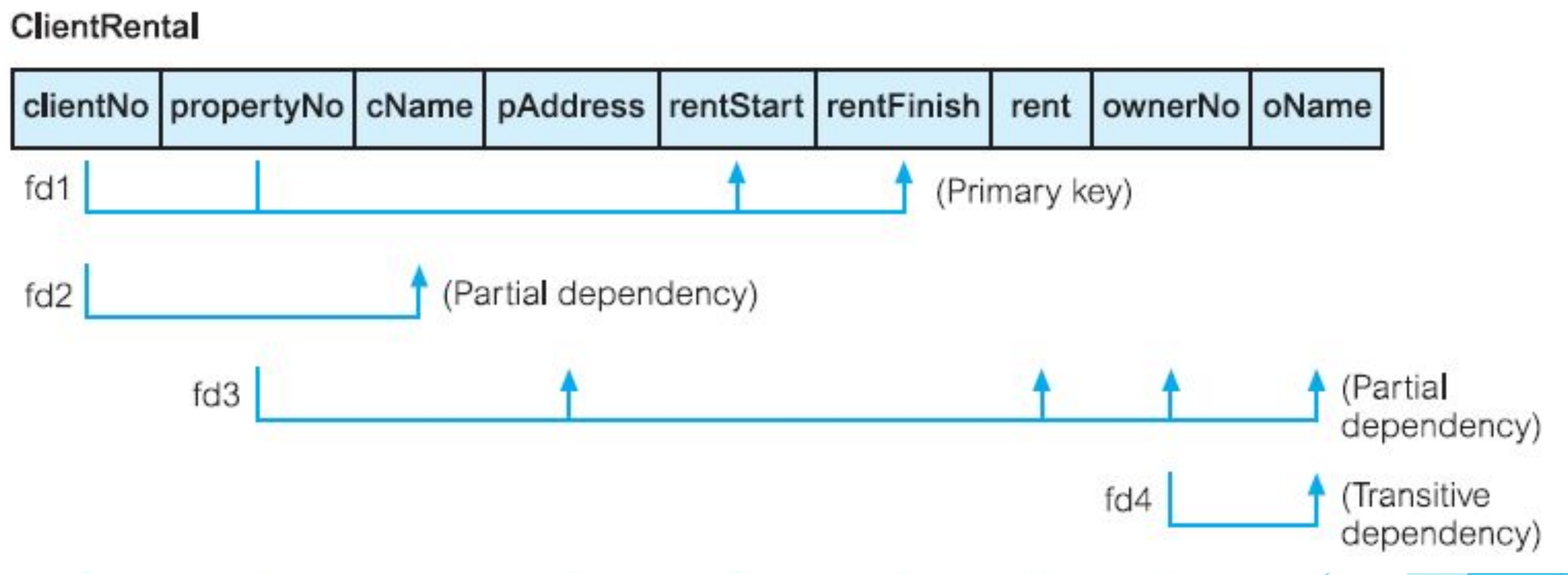
- Remove the repeating group (property rented details) by placing the repeating data along with a copy of the original key attribute (clientNo) in a separate relation.



Client	(clientNo, cName)
PropertyRentalOwner	(clientNo, propertyNo, pAddress, rentStart, rentFinish, rent, ownerNo, oName)

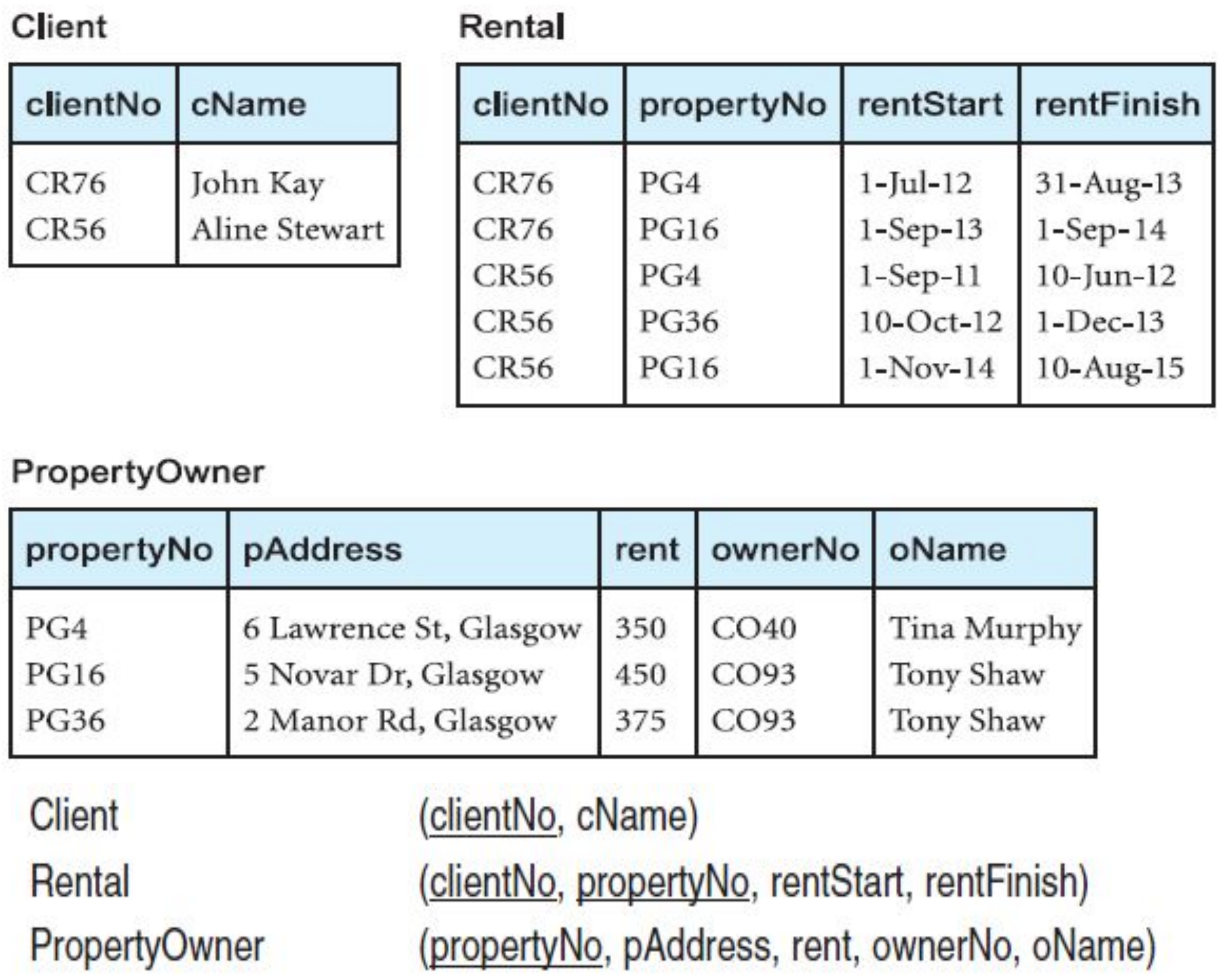
Normal Forms Based on Primary Keys

- ***Example:***
- **FD**



Normal Forms Based on Primary Keys

- ***Example:***
- **2NF**
- If a partial dependency exists, we remove the partially dependent attribute(s) from the relation by placing them in a new relation along with a copy of their determinant.
- Client attribute (cName) is partially dependent on the primary key(represented as fd2).
- The property attributes (pAddress, rent, ownerNo, oName) are partially dependent on the primary key, that is, on only the propertyNo attribute (represented as fd3).



Client

clientNo	cName
CR76	John Kay
CR56	Aline Stewart

Rental

clientNo	propertyNo	rentStart	rentFinish
CR76	PG4	1-Jul-12	31-Aug-13
CR76	PG16	1-Sep-13	1-Sep-14
CR56	PG4	1-Sep-11	10-Jun-12
CR56	PG36	10-Oct-12	1-Dec-13
CR56	PG16	1-Nov-14	10-Aug-15

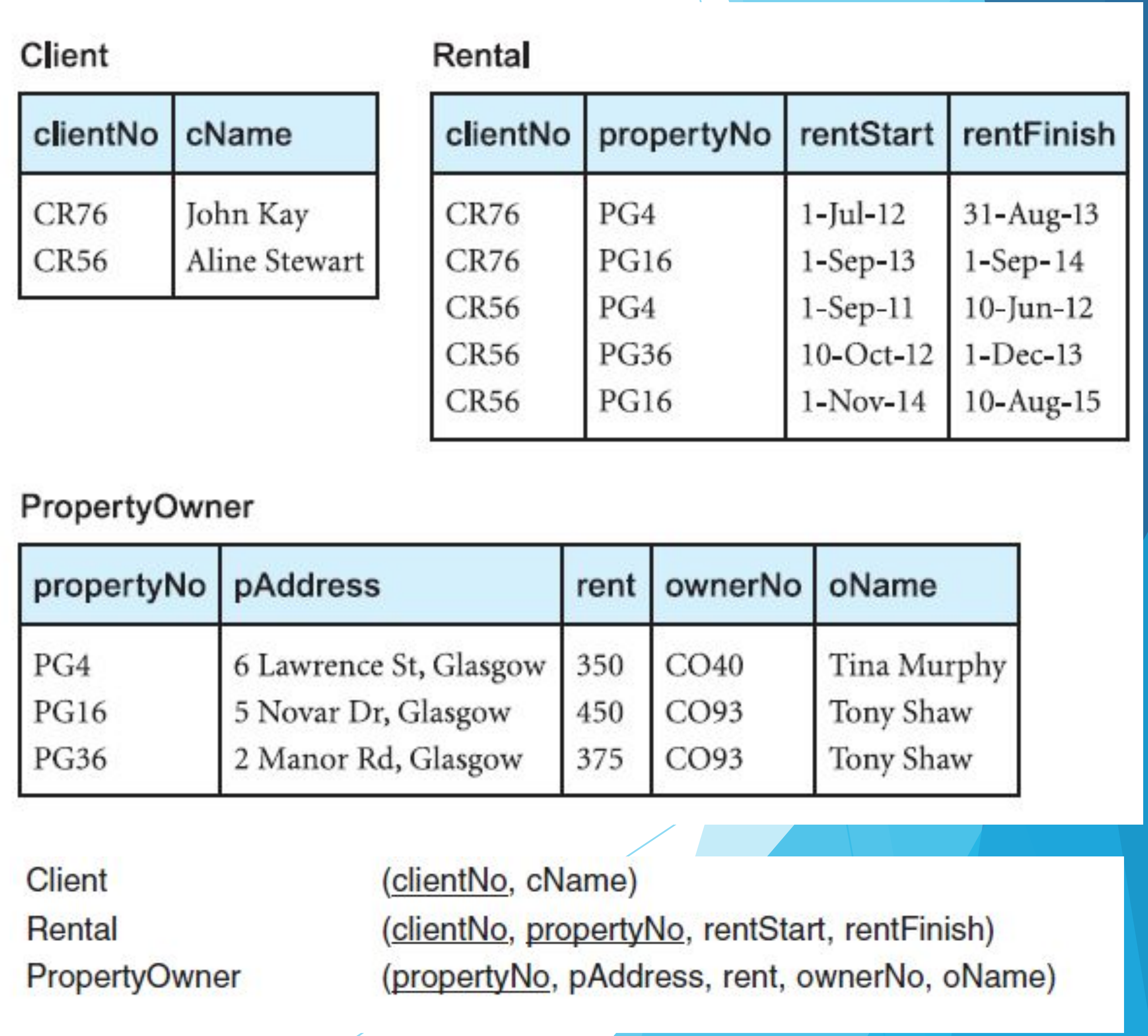
PropertyOwner

propertyNo	pAddress	rent	ownerNo	oName
PG4	6 Lawrence St, Glasgow	350	CO40	Tina Murphy
PG16	5 Novar Dr, Glasgow	450	CO93	Tony Shaw
PG36	2 Manor Rd, Glasgow	375	CO93	Tony Shaw

Client	(clientNo, cName)
Rental	(clientNo, propertyNo, rentStart, rentFinish)
PropertyOwner	(propertyNo, pAddress, rent, ownerNo, oName)

Normal Forms Based on Primary Keys

- ***Example:***
- **3NF**
- Although 2NF relations have less redundancy than those in 1NF, they may still suffer from update anomalies.
- For example,
  - if we want to update the name of an owner, such as Tony Shaw (ownerNo CO93),
  - we have to update two tuples in the PropertyOwner relation.
- If we update only one tuple and not the other, the database would be in an inconsistent state.
- **This update anomaly is caused by a transitive dependency.**



Client

clientNo	cName
CR76	John Kay
CR56	Aline Stewart

Rental

clientNo	propertyNo	rentStart	rentFinish
CR76	PG4	1-Jul-12	31-Aug-13
CR76	PG16	1-Sep-13	1-Sep-14
CR56	PG4	1-Sep-11	10-Jun-12
CR56	PG36	10-Oct-12	1-Dec-13
CR56	PG16	1-Nov-14	10-Aug-15

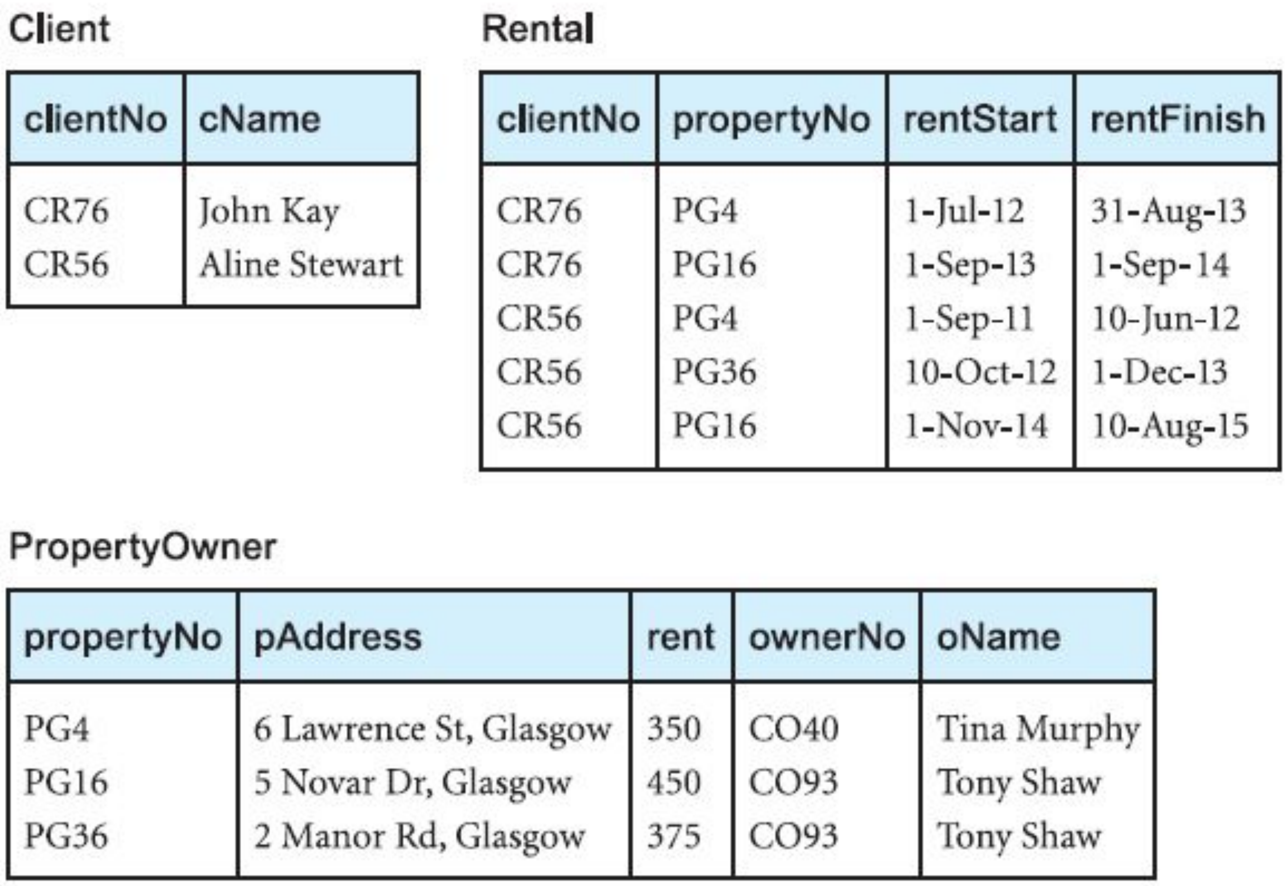
PropertyOwner

propertyNo	pAddress	rent	ownerNo	oName
PG4	6 Lawrence St, Glasgow	350	CO40	Tina Murphy
PG16	5 Novar Dr, Glasgow	450	CO93	Tony Shaw
PG36	2 Manor Rd, Glasgow	375	CO93	Tony Shaw

Client	(clientNo, cName)
Rental	(clientNo, propertyNo, rentStart, rentFinish)
PropertyOwner	(propertyNo, pAddress, rent, ownerNo, oName)

Normal Forms Based on Primary Keys

- ***Example:***
- **3NF**
- **SOLUTION:**
- If a transitive dependency exists, we remove the transitively dependendent attribute(s) from the relation by placing the attribute(s) in a new relation along with a copy of the determinant.



Client

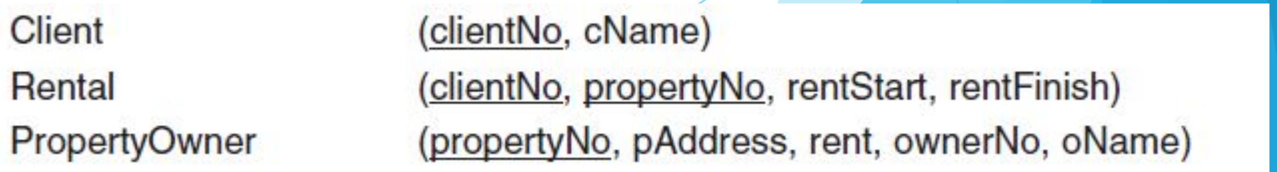
clientNo	cName
CR76	John Kay
CR56	Aline Stewart

Rental

clientNo	propertyNo	rentStart	rentFinish
CR76	PG4	1-Jul-12	31-Aug-13
CR76	PG16	1-Sep-13	1-Sep-14
CR56	PG4	1-Sep-11	10-Jun-12
CR56	PG36	10-Oct-12	1-Dec-13
CR56	PG16	1-Nov-14	10-Aug-15

PropertyOwner

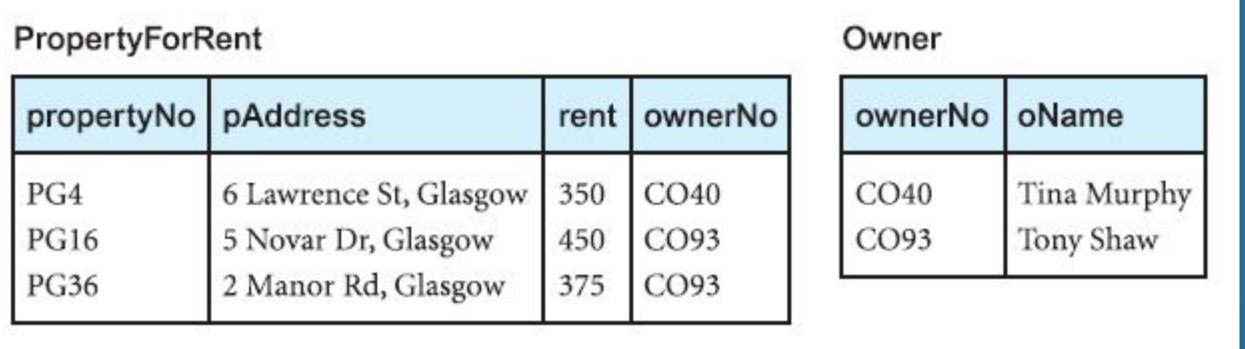
propertyNo	pAddress	rent	ownerNo	oName
PG4	6 Lawrence St, Glasgow	350	CO40	Tina Murphy
PG16	5 Novar Dr, Glasgow	450	CO93	Tony Shaw
PG36	2 Manor Rd, Glasgow	375	CO93	Tony Shaw



Client	(<u>clientNo</u>, cName)
Rental	(<u>clientNo</u>, <u>propertyNo</u>, rentStart, rentFinish)
PropertyOwner	(<u>propertyNo</u>, pAddress, rent, ownerNo, oName)

Normal Forms Based on Primary Keys

- ***Example:***
- **3NF**
- FD4 rectified here as:
  - By transforming the PropertyOwner relation into two new relations called PropertyForRent and Owner.

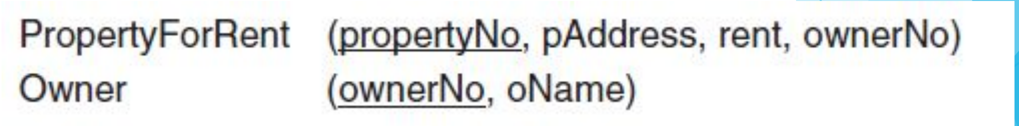


PropertyForRent

propertyNo	pAddress	rent	ownerNo
PG4	6 Lawrence St, Glasgow	350	CO40
PG16	5 Novar Dr, Glasgow	450	CO93
PG36	2 Manor Rd, Glasgow	375	CO93

Owner

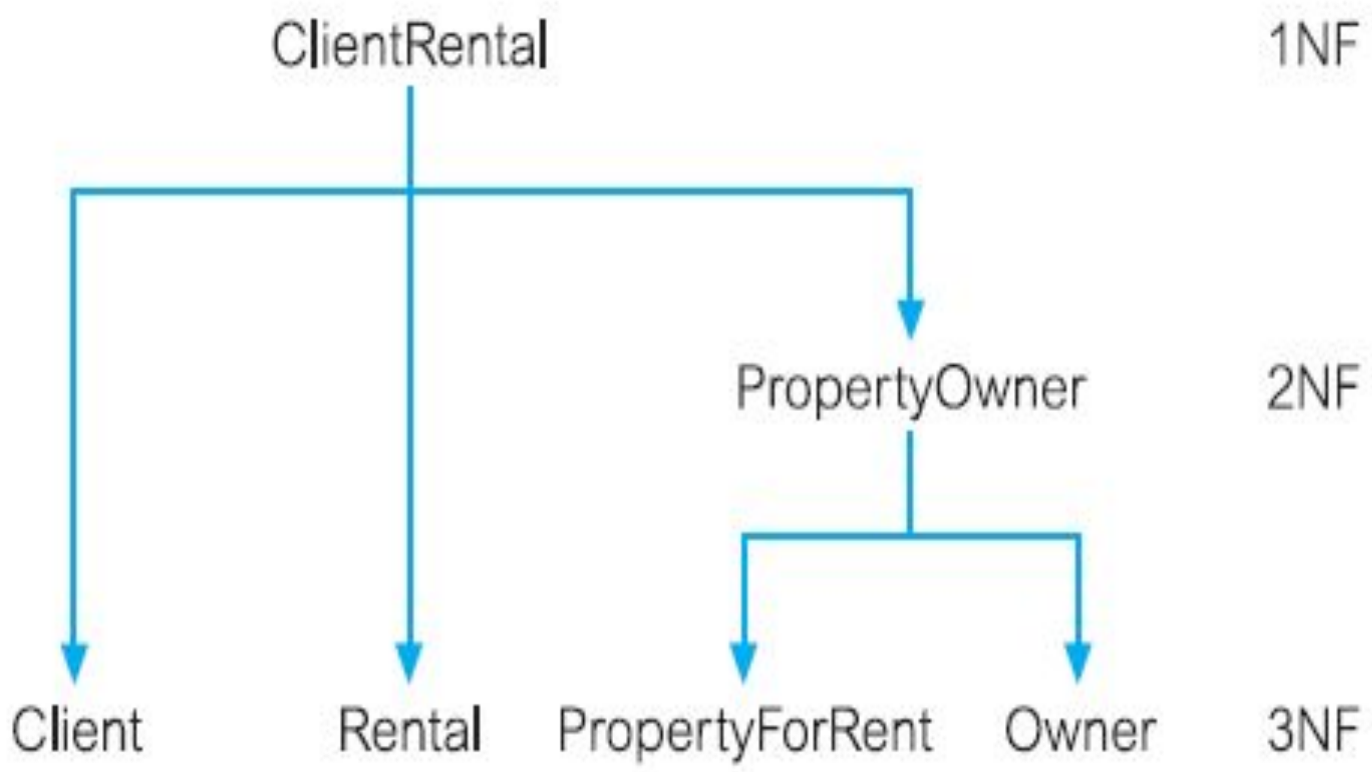
ownerNo	oName
CO40	Tina Murphy
CO93	Tony Shaw

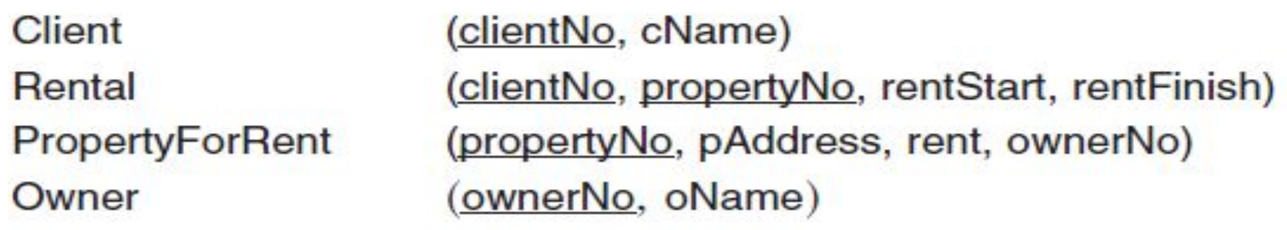


PropertyForRent	(propertyNo, pAddress, rent, ownerNo)
Owner	(ownerNo, oName)

Normal Forms Based on Primary Keys

- ***Example:***
- The original ClientRental relation can be recreated by joining the Client, Rental, PropertyForRent, and Owner relations through the primary key/ foreign key mechanism.
- For example, the ownerNo attribute is a primary key within the Owner relation and is also present within the PropertyForRent relation as a foreign key.

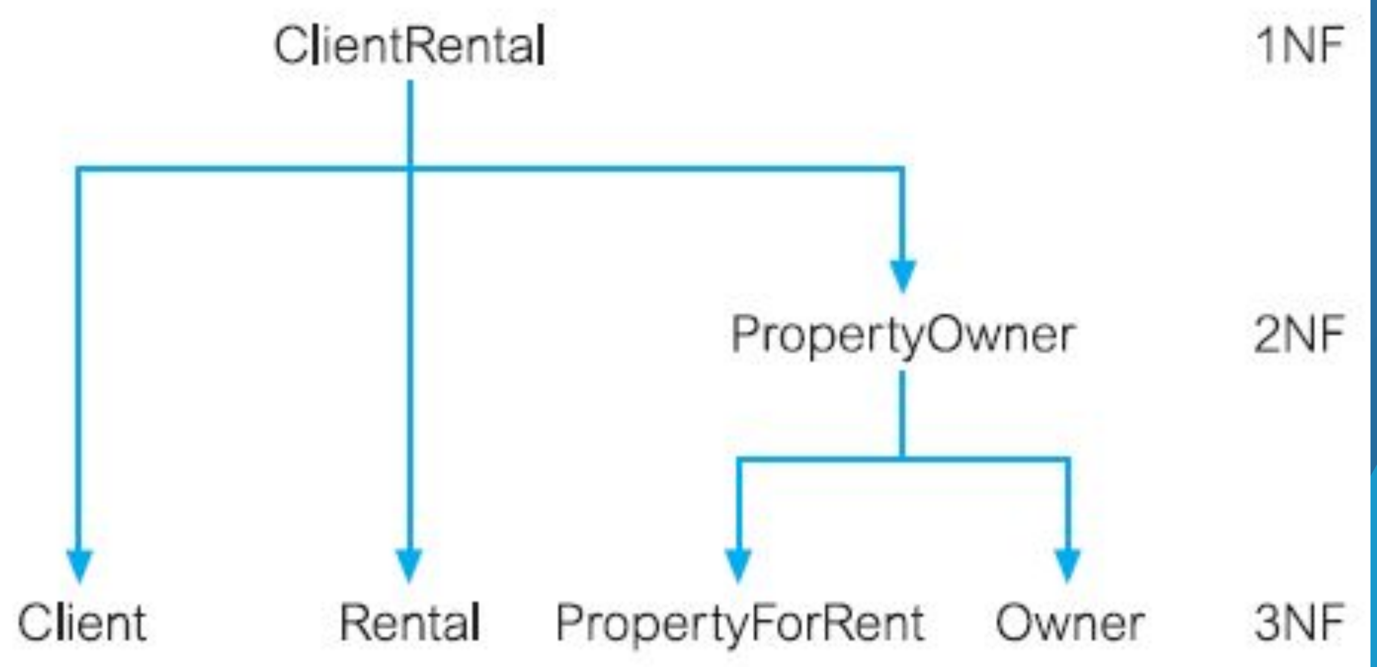




Client	(clientNo, cName)
Rental	(clientNo, propertyNo, rentStart, rentFinish)
PropertyForRent	(propertyNo, pAddress, rent, ownerNo)
Owner	(ownerNo, oName)

Normal Forms Based on Primary Keys

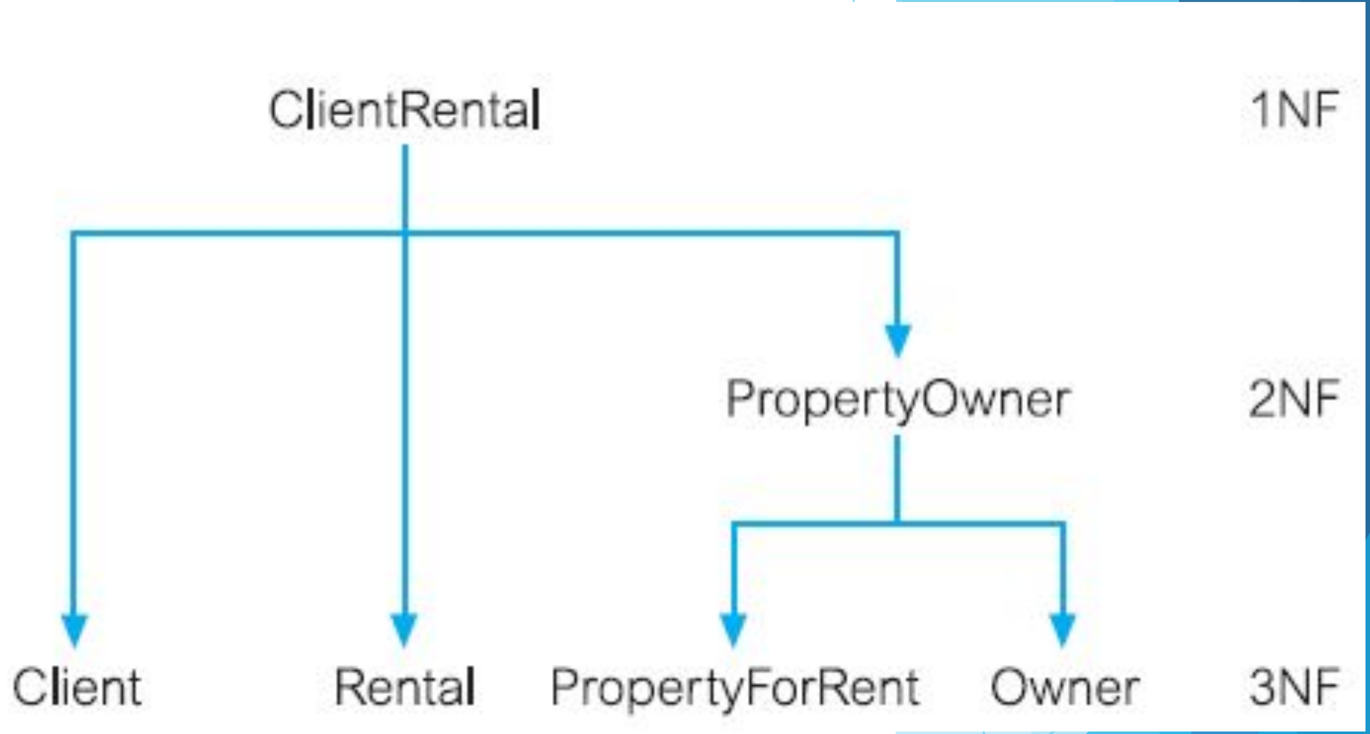
- ***Example:***
- The clientNo attribute is a primary key of the Client relation and is also present within the Rental relation as a foreign key.
- Note that in this case the clientNo attribute in the Rental relation acts both as a foreign key and as part of the primary key of this relation.
- Similarly, the propertyNo attribute is the primary key of the PropertyForRent relation and is also present within the Rental relation acting both as a foreign key and as part of the primary key for this relation.

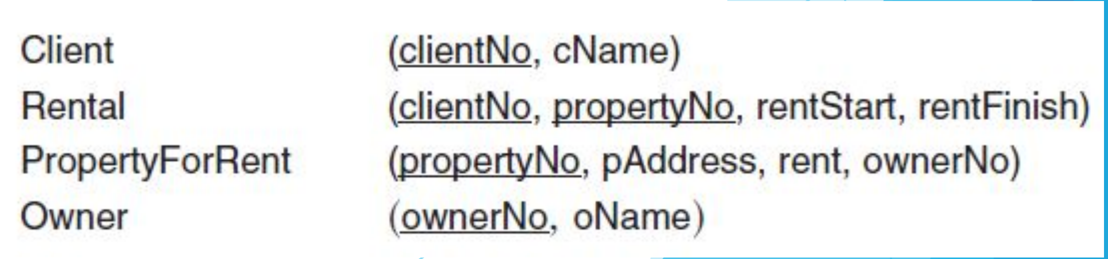


Client	(clientNo, cName)
Rental	(clientNo, propertyNo, rentStart, rentFinish)
PropertyForRent	(propertyNo, pAddress, rent, ownerNo)
Owner	(ownerNo, oName)

Normal Forms Based on Primary Keys

- ***Example:***
- In other words, the normalization process has decomposed the original ClientRental relation.
- This results in a lossless-join (also called non lossy or non additive-join) decomposition, which is reversible.
  - ***As natural join won't result in generation of spurious tuples***





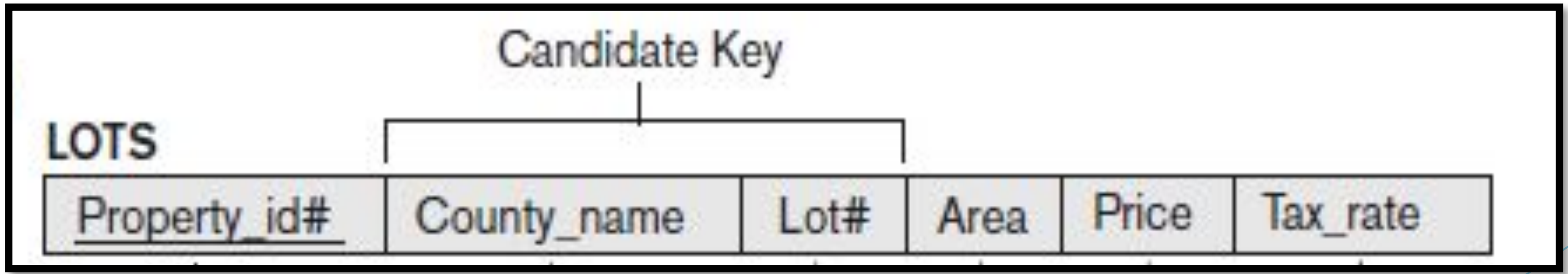
Client	(clientNo, cName)
Rental	(clientNo, propertyNo, rentStart, rentFinish)
PropertyForRent	(propertyNo, pAddress, rent, ownerNo)
Owner	(ownerNo, oName)

General Definitions of 2NF & 3NF

- ***General Definition of Second Normal Form***
- **Definition.**
  - A relation schema R is in 2NF if, **every nonprime attribute A in R is not partially dependent on ANY KEY of R.**
- **The test for 2NF**
  - check for functional dependencies whose left-hand side attributes are part of the primary key.
  - If the primary key contains a single attribute, the test need not be applied at all.

General Definitions of 2NF & 3NF

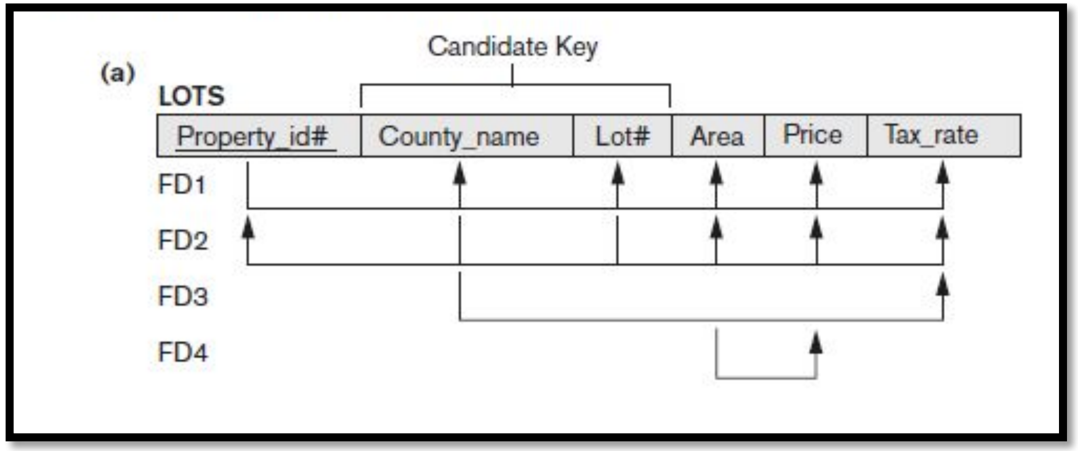
- ***General Definition of Second Normal Form***
- Consider the relation schema LOTS which describes parcels of land for sale in various counties of a state.
- Suppose that there are two candidate keys:
  - **Property_id#**
  - **{County_name, Lot#}**
- Lot# are unique only within each county,
- but Property_id# are unique across counties for the entire state.



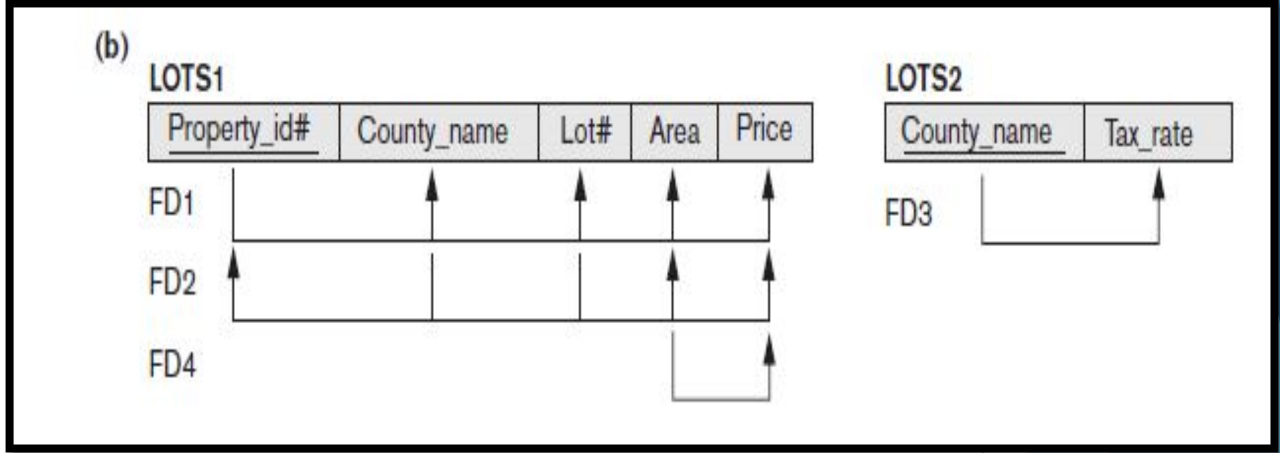
General Definitions of 2NF & 3NF

- ***General Definition of Second Normal Form***
- Following FDs hold in the given schema:
- **FD1: Property_id# → county_name, Lot#,Area, price,Tax_rate**
- **FD2: {County_name, Lot#} → property_Id#, Area, price,Tax_rate**
- **FD3: County_name → Tax_rate**
  - **Assume:**
    - the tax rate is fixed for a given county
    - does not vary lot by lot within the same county
- **FD4: Area → Price**
  - **Assume:**
    - The price of a lot is determined by its area regardless of the county it is in.

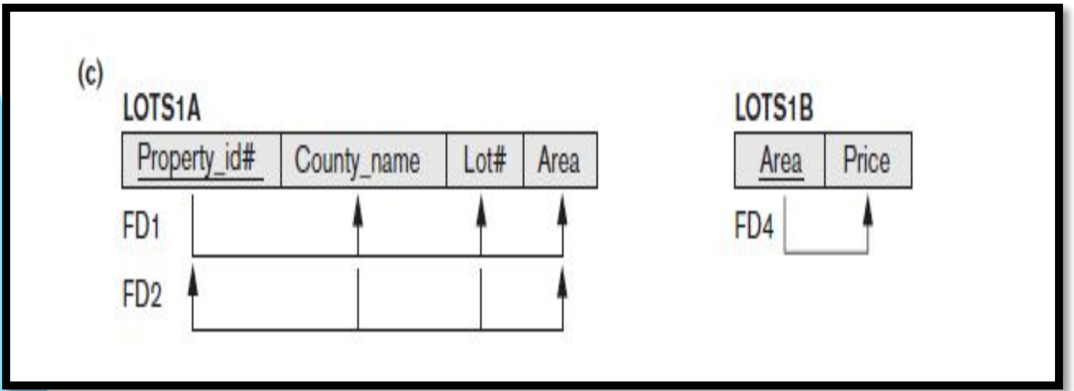
General Definitions of 2NF & 3NF



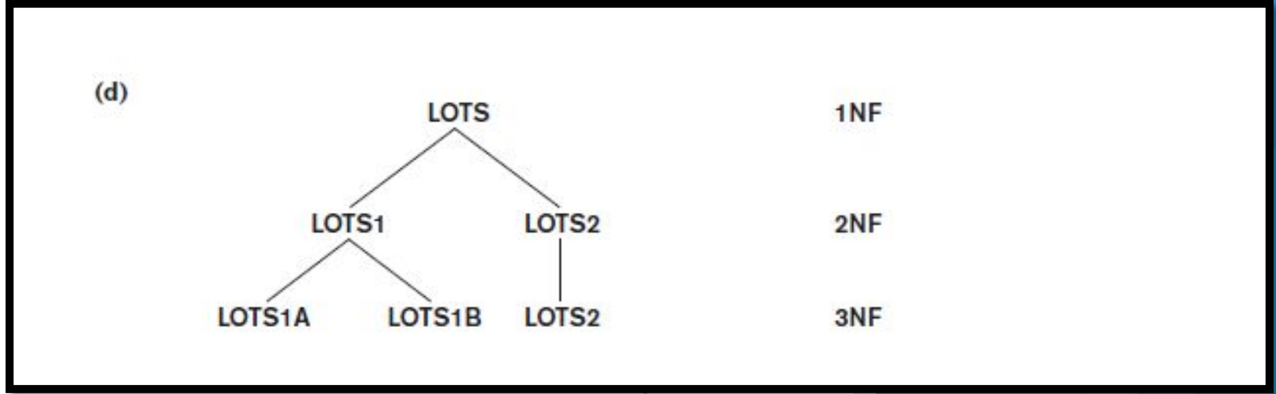
Identify FDs



Remove partial dependencies on candidate key



Remove transitive dependencies

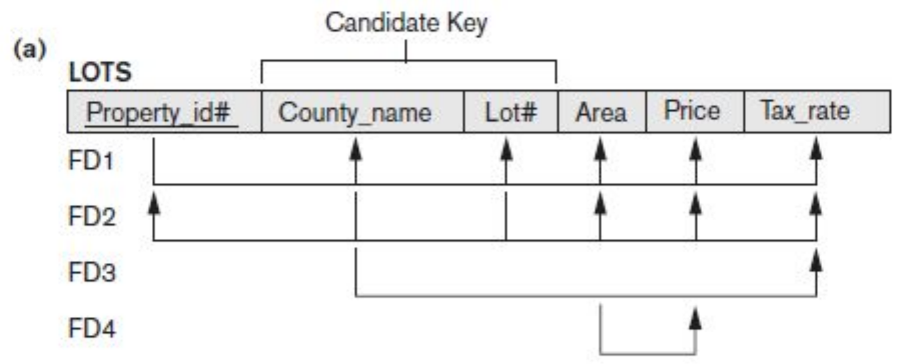


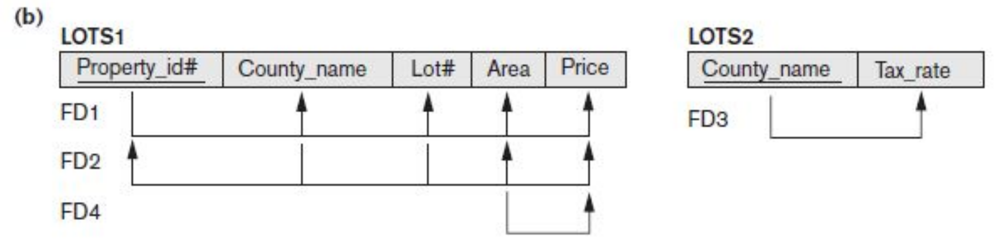
Finalized schema

General Definitions of 2NF & 3NF

- ***General Definition of Second Normal Form***
- **2NF violation:**
  - FD3: Tax_rate is **partially dependent on the candidate key {County_name, Lot#}.**
- **Solution:**
  - decompose it into the two relations LOTS1 and LOTS2.
  - We construct LOTS1 by removing the attribute Tax_rate that violates 2NF from LOTS and placing it with County_name (the left-hand side of FD3 that causes the partial dependency) into another relation LOTS2.
  - Both LOTS1 and LOTS2 are in 2NF.

Figure 14.12
Normalization into 2NF and 3NF. (a) The LOTS relation with its functional dependencies FD1 through FD4. (b) Decomposing into the 2NF relations LOTS1 and LOTS2. (c) Decomposing LOTS1 into the 3NF relations LOTS1A and LOTS1B. (d) Progressive normalization of LOTS into a 3NF design.



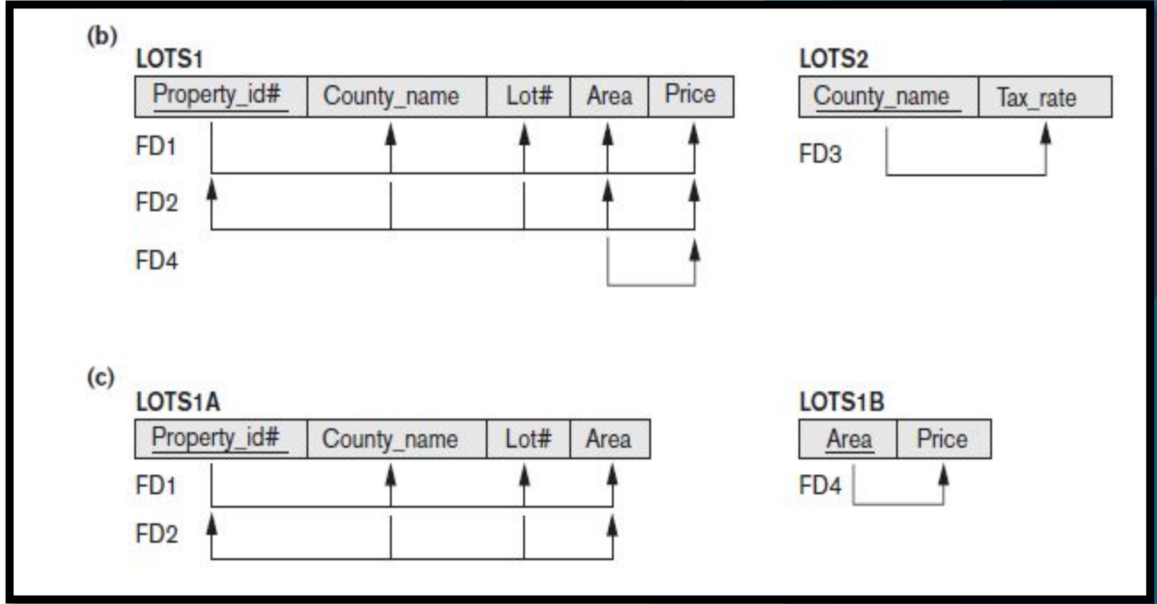


General Definitions of 2NF & 3NF

- ***General Definition of Third Normal Form***
- **Definition.**
  - A relation schema R is in third normal form (3NF) if,
    - whenever a nontrivial functional dependency $X \rightarrow A$ holds in R,
      - Either (a) X is a super key of R,
      - or (b) A is a prime attribute of R.

General Definitions of Second and Third Normal Forms

- *General Definition of Third Normal Form*
- **3NF violation:**
  - FD4: Area → Price
  - Here, Area is not a superkey and Price is not a prime attribute in LOTS1.
- **Solution:**
  - we decompose LOTS1 into the relation schemas LOTS1A and LOTS1B.
  - We construct LOTS1A by removing the attribute Price that violates 3NF from LOTS1 and placing it with Area (the left-hand side of FD4 that causes the transitive dependency) into another relation LOTS1B.



General Definitions of Second and Third Normal Forms

- ***General Definition of Third Normal Form***
- LOTS1 violates 3NF because Price is transitively dependent on each of the candidate keys of LOTS1 via the nonprime attribute Area.
- **This general definition can be applied directly to test whether a relation schema is in 3NF;**
  - it does not have to go through 2NF first.
- **In other words, if a relation passes the general 3NF test, then it automatically passes the 2NF test.**
- If we apply the GENERAL 3NF definition to LOTS with the dependencies FD1 through FD4,
  - we find that both FD3 and FD4 violate 3NF because the LHS County_name in FD3 is not a super key.

General Definitions of Second and Third Normal Forms

- ***Interpreting the General Definition of Third Normal Form***
- A relation schema R violates the general definition of 3NF if a functional dependency X → A holds in R that meets either of the two conditions, namely (a) and (b).
- The first condition "catches" two types of problematic dependencies:
  - **A nonprime attribute determines another nonprime attribute**. Here we typically have a transitive dependency that violates 3NF.
  - **A proper subset of a key of R functionally determines a nonprime attribute**. Here we have a partial dependency that violates 2NF.

General Definitions of Second and Third Normal Forms

- *Interpreting the General Definition of Third Normal Form*
- Therefore, we can state a **general alternative definition of 3NF** as follows:
- **A relation schema R is in 3NF** if every nonprime attribute of R meets both of the following conditions:
  - **It is fully functionally dependent on every key of R.**
  - **It is non transitively dependent on every key of R.**

General Definitions of Boyce Codd Normal Form

- *General Definition of BCNF*
- **Definition.**
- A relation is in BCNF if & only if,
  - Relation is in 3NF &
  - For each FD in R,
    - The left hand side of all FDs are a Super key (i.e., **X → A holds in R, then X is a super key of R**.)

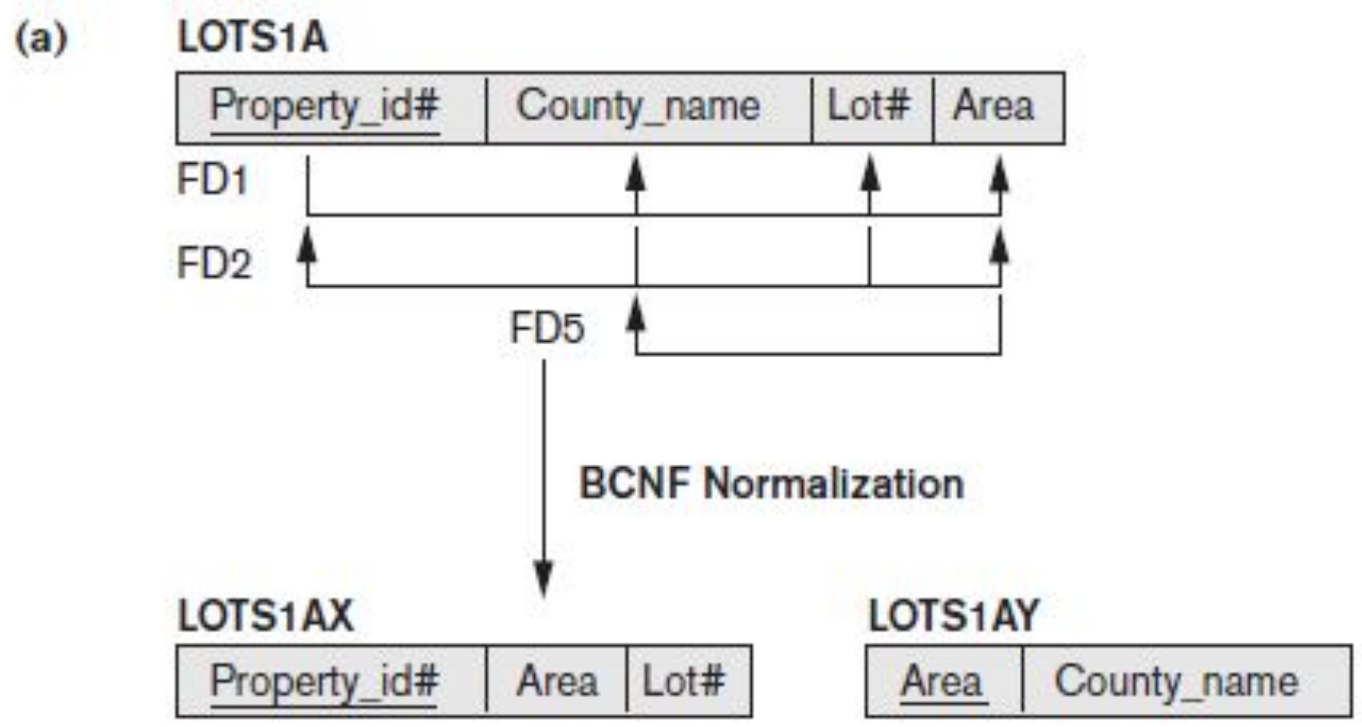
- **Scenario for BCNF violation:**
  - check if a FD : **X → A , here X is a non prime attribute but A is prime attribute.**
  - Now, relation is in 3NF but not in BCNF.

Boyce-Codd Normal Form

- ***Boyce-Codd Normal Form***
- Assume,
  - we have thousands of lots in the relation
  - but the lots are from only two counties: DeKalb and Fulton.
  - lot sizes 0.5, 0.6, 0.7, 0.8, 0.9, and 1.0 acres are only in DeKalb County
  - lot sizes 1.1, 1.2, … , 1.9, and 2.0 acres are only in Fulton County
    - In such a situation we would have the additional functional dependency **FD5: Area → County_name.**
  - **County_name is a prime attribute, but Area is a non -prime attribute ~ BCNF violation**
- **CKs were Property_id# and {County_name, Lot#}**

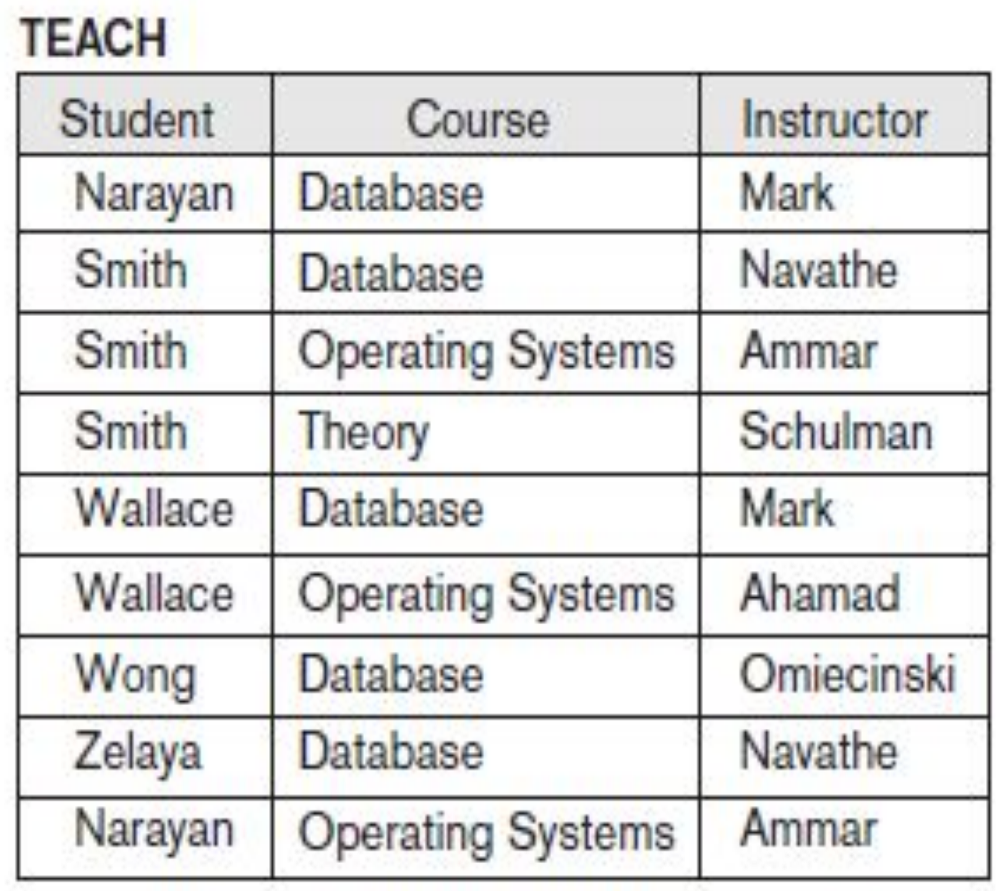
Boyce-Codd Normal Form

- ***Boyce-Codd Normal Form (Decomposition of tables)***



Boyce-Codd Normal Form

- ***Decomposition of Relations not in BCNF***
- A relation TEACH with the following dependencies:
- **FD1: {Student, Course} → Instructor**
- **FD2:Instructor → Course**
- Note that {Student, Course} is a candidate key for this relation.
- **Hence this relation is in 3NF but not BCNF.**
- Decomposition of this relation schema may be done in multiple ways.like:
  - R {instructor, course } and R {instructor, student}
  - R {instructor, course } and R {course, student}



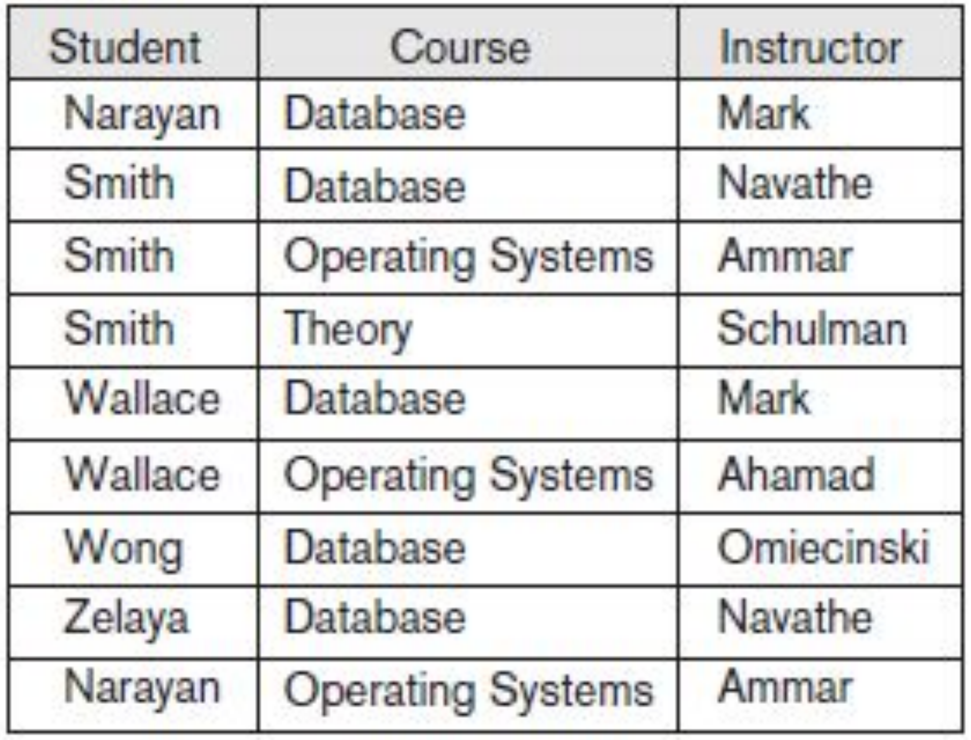
TEACH

Student	Course	Instructor
Narayan	Database	Mark
Smith	Database	Navathe
Smith	Operating Systems	Ammar
Smith	Theory	Schulman
Wallace	Database	Mark
Wallace	Operating Systems	Ahamad
Wong	Database	Omiecinski
Zelaya	Database	Navathe
Narayan	Operating Systems	Ammar

Boyce-Codd Normal Form

- The decomposition for relation TEACH
  - R {instructor, course } and R {instructor, student}
- This decomposition will lose fd1

TEACH



Student	Course	Instructor
Narayan	Database	Mark
Smith	Database	Navathe
Smith	Operating Systems	Ammar
Smith	Theory	Schulman
Wallace	Database	Mark
Wallace	Operating Systems	Ahamad
Wong	Database	Omiecinski
Zelaya	Database	Navathe
Narayan	Operating Systems	Ammar

Fourth Normal Form (4NF)

- ***Definition:***
  - A relation will be in 4NF if:
    - It is in Boyce Codd normal form and has no multi-valued dependency
    - AND For a dependency A → B, if for a single value of A, multiple values of B exists, then the relation will be a multi-valued dependency.

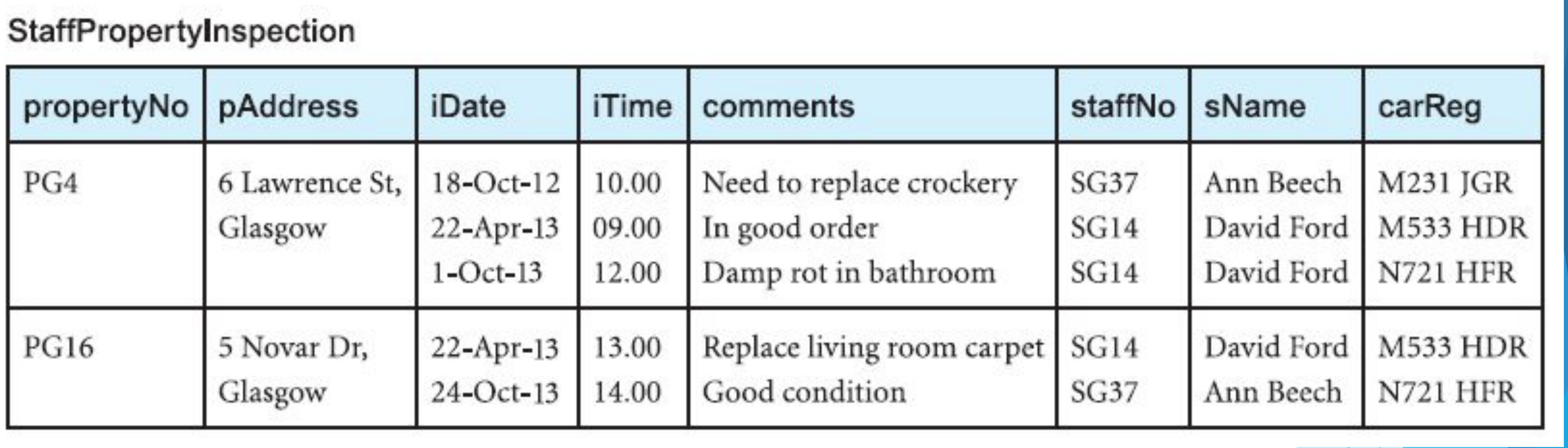
- ***Example:***
  - *R (Student, course, Hobbies)*
    - Student Course
    - Student Hobbies
      - Here course & hobbies should be independent
      - redundant data
  - *R (Employee#, Project#, DependentName)*
    - Employee Project
    - Employee DependenctName

Fifth Normal Form (5NF)

- ***Definition:***
  - A relation will be in 5NF if:
    - It is in 4th normal form and has no multi-valued dependency
    - AND no lossy join conditions are there. i.e., no join dependencies exists in the table.
      - Can be avoided when decomposition of tables are done in a way that there reconstruction leads to same data values & no spurious tuples are generated.

Boyce-Codd Normal Form

- ***Example:***

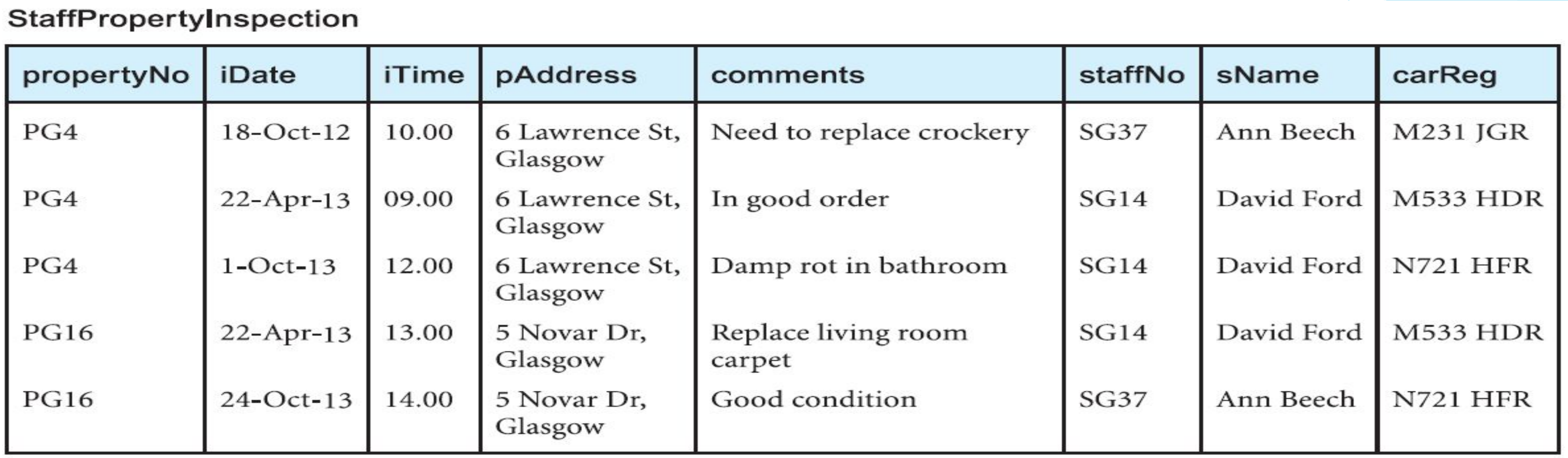


StaffPropertyInspection

propertyNo	pAddress	iDate	iTime	comments	staffNo	sName	carReg
PG4	6 Lawrence St, Glasgow	18-Oct-12	10.00	Need to replace crockery	SG37	Ann Beech	M231 JGR
		22-Apr-13	09.00	In good order	SG14	David Ford	M533 HDR
		1-Oct-13	12.00	Damp rot in bathroom	SG14	David Ford	N721 HFR
PG16	5 Novar Dr, Glasgow	22-Apr-13	13.00	Replace living room carpet	SG14	David Ford	M533 HDR
		24-Oct-13	14.00	Good condition	SG37	Ann Beech	N721 HFR

Boyce-Codd Normal Form

- ***Example:***



StaffPropertyInspection

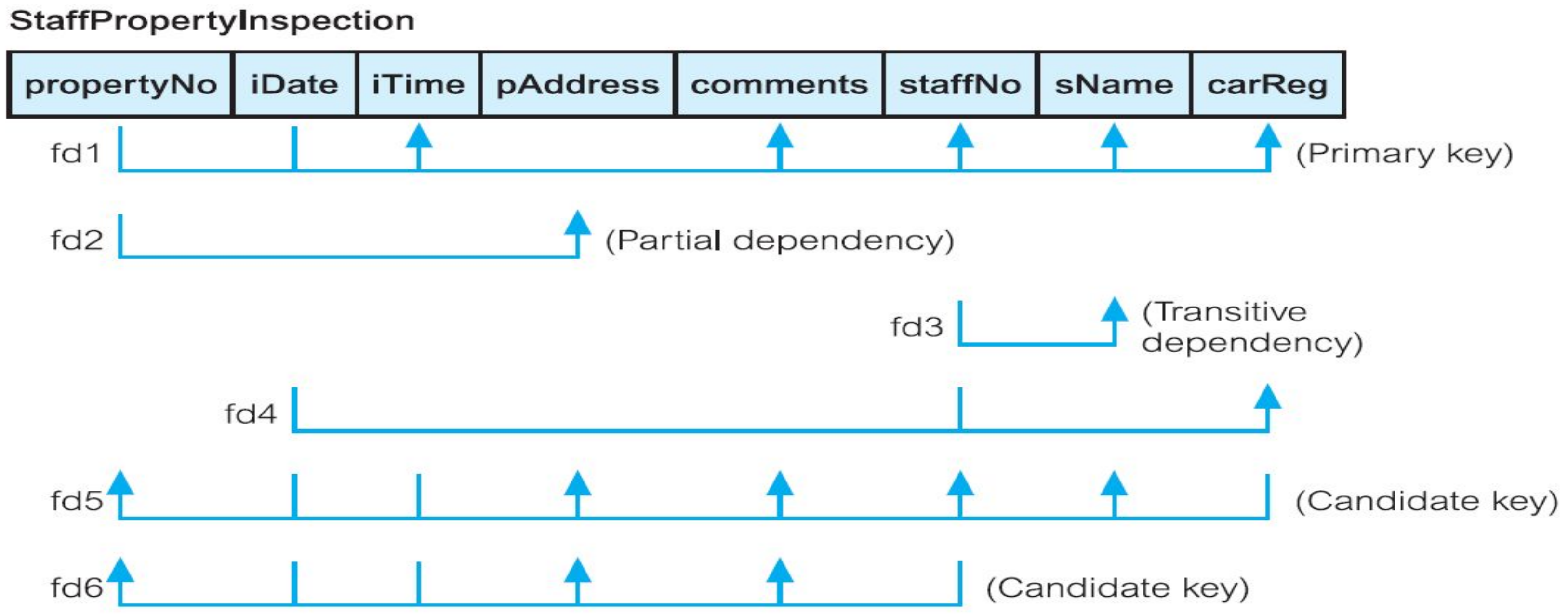
propertyNo	iDate	iTime	pAddress	comments	staffNo	sName	carReg
PG4	18-Oct-12	10.00	6 Lawrence St, Glasgow	Need to replace crockery	SG37	Ann Beech	M231 JGR
PG4	22-Apr-13	09.00	6 Lawrence St, Glasgow	In good order	SG14	David Ford	M533 HDR
PG4	1-Oct-13	12.00	6 Lawrence St, Glasgow	Damp rot in bathroom	SG14	David Ford	N721 HFR
PG16	22-Apr-13	13.00	5 Novar Dr, Glasgow	Replace living room carpet	SG14	David Ford	M533 HDR
PG16	24-Oct-13	14.00	5 Novar Dr, Glasgow	Good condition	SG37	Ann Beech	N721 HFR

The First Normal Form(1NF) StaffPropertyInspection relation.

StaffPropertyInspection (propertyNo, iDate, iTime, pAddress, comments, staffNo, sName, carReg)

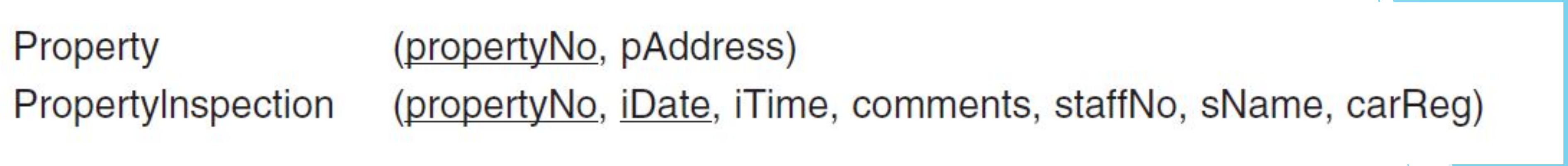
Boyce-Codd Normal Form

- ***Example:***



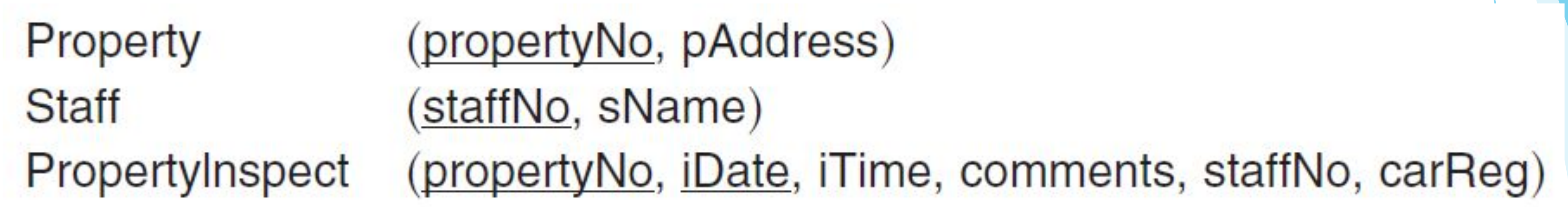
fd1	propertyNo, iDate → iTime, comments, staffNo, sName, carReg	(Primary key)
fd2	propertyNo → pAddress	(Partial dependency)
fd3	staffNo → sName	(Transitive dependency)
fd4	staffNo, iDate → carReg	
fd5	carReg, iDate, iTime → propertyNo, pAddress, comments, staffNo, sName	(Candidate key)
fd6	staffNo, iDate, iTime → propertyNo, pAddress, comments	(Candidate key)

Second Normal Form (2NF)



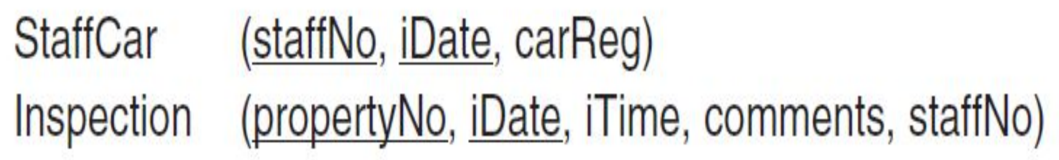
Property	(propertyNo, pAddress)
PropertyInspection	(propertyNo, iDate, iTime, comments, staffNo, sName, carReg)

Third Normal Form (3NF)

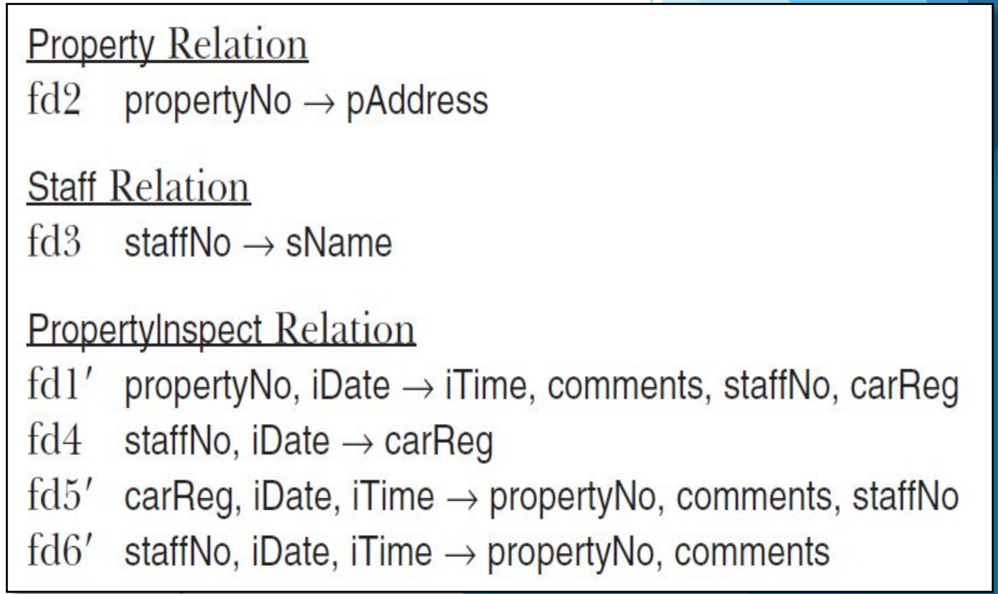


Property	(propertyNo, pAddress)
Staff	(staffNo, sName)
PropertyInspect	(propertyNo, iDate, iTime, comments, staffNo, carReg)

Boyce–Codd Normal Form (BCNF)



StaffCar	(staffNo, iDate, carReg)
Inspection	(propertyNo, iDate, iTime, comments, staffNo)



Property Relation

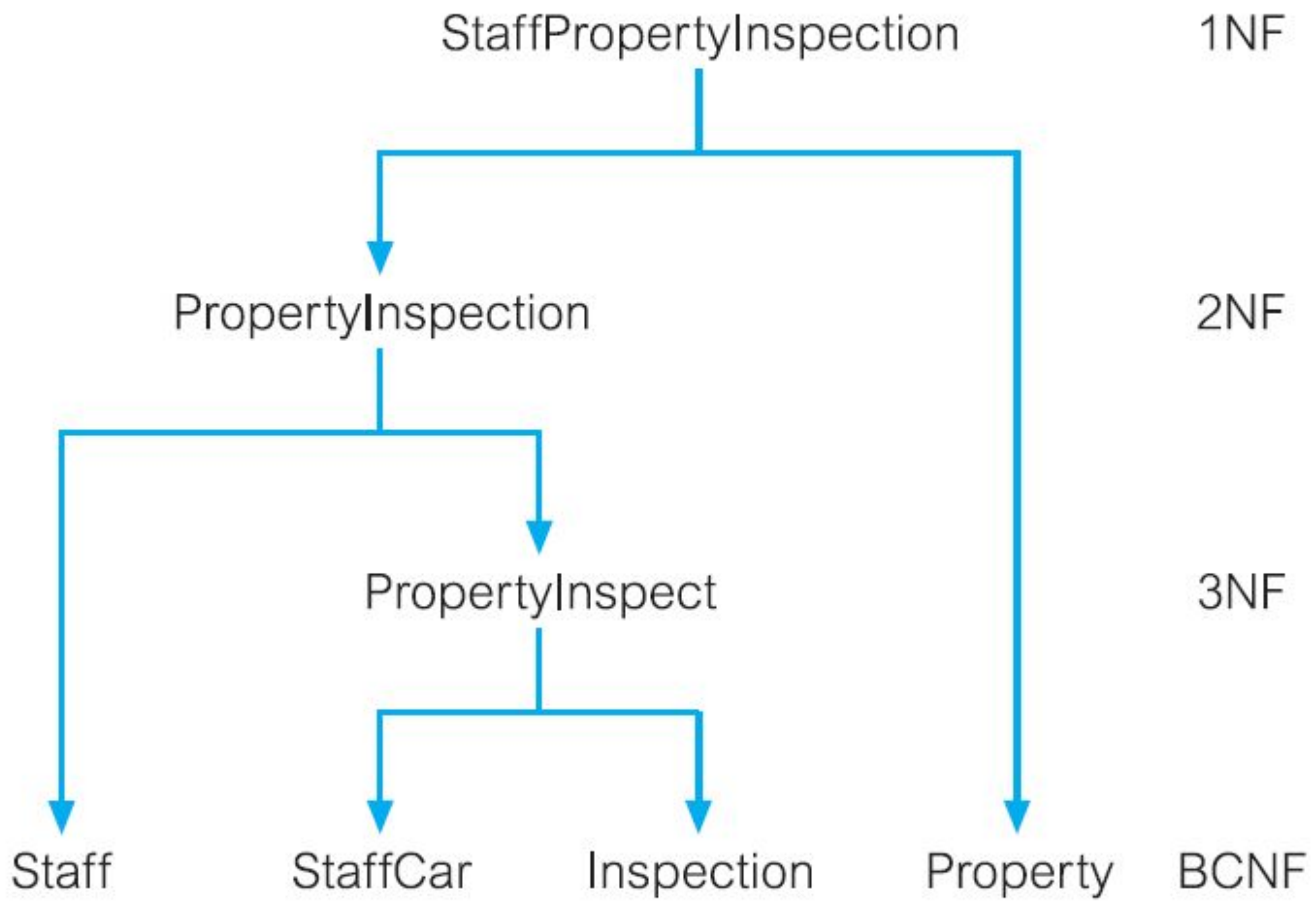
fd2	propertyNo → pAddress

Staff Relation

fd3	staffNo → sName

PropertyInspect Relation

fd1′	propertyNo, iDate → iTime, comments, staffNo, carReg
fd4	staffNo, iDate → carReg
fd5′	carReg, iDate, iTime → propertyNo, comments, staffNo
fd6′	staffNo, iDate, iTime → propertyNo, comments


StaffPropertyInspection
1NF
PropertyInspection
2NF
PropertyInspect
3NF
Staff
StaffCar
Inspection
Property
BCNF