CHAPTER 14

NORMALIZATION

CHAPTER 14 - OBJECTIVES

- The potential problems associated with redundant data in base relations.
- The concept of functional dependency, which describes the relationship between attributes.
- How to undertake the process of normalization.
- How to identify the most commonly used normal forms, namely First Normal Form (1NF), Second Normal Form (2NF), and Third Normal Form (3NF).

NORMALIZATION

Normalization is a bottom up approach to database design that begins by examining the relationships between attributes.

Normalization is often performed as a series of test on a relation to determine whether it satisfy or violates the requirements of a given normal form .

The process of normalization is a formal method that identifies relations base on their primary or candidate keys and the functional dependencies among their attributes.

DATA REDUNDANCY AND UPDATE ANOMALIES

Major aim of relational database design is to group attributes into relations to *minimize* data redundancy.

Relations that contain redundant information may potentially suffer from update anomalies.

Types of update anomalies include:

- Insertion
- Deletion
- Modification

FUNCTIONAL DEPENDENCIES

- **Important concept associated with normalization.**

- **Functional dependency describes relationship between attributes.**
- **For example**:

If A and B are attributes of relation R, B is functionally dependent on A (denoted A $\rightarrow$ B), if each value of A in R is associated with exactly one value of B in R (A and B may each consist of one or more attributes)

SAMPLE DATA

Staff

staffNo	sName	position	salary	branchNo
SL21	John White	Manager	30000	B005
SG37	Ann Beech	Assistant	12000	B003
SG14	David Ford	Supervisor	18000	B003
SA9	Mary Howe	Assistant	9000	B007
SG5	Susan Brand	Manager	24000	B003
SL41	Julie Lee	Assistant	9000	B005

Branch

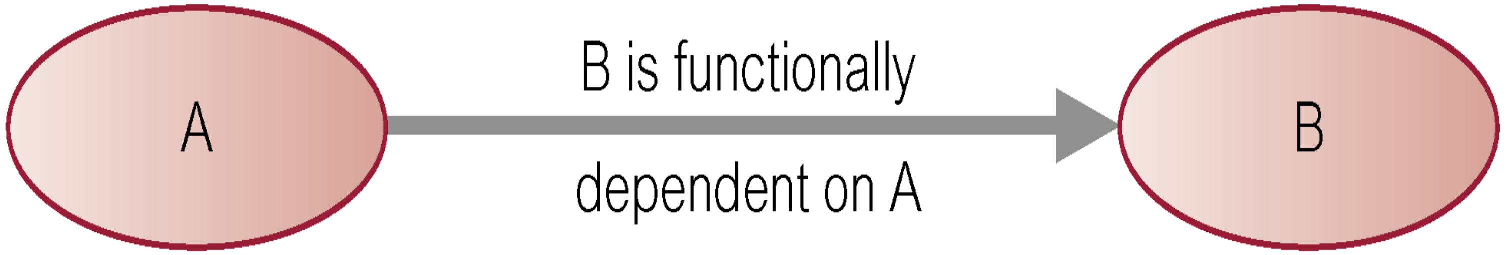
branchNo	bAddress
B005	22 Deer Rd, London
B007	16 Argyll St, Aberdeen
B003	163 Main St, Glasgow

StaffBranch

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

FUNCTIONAL DEPENDENCIES

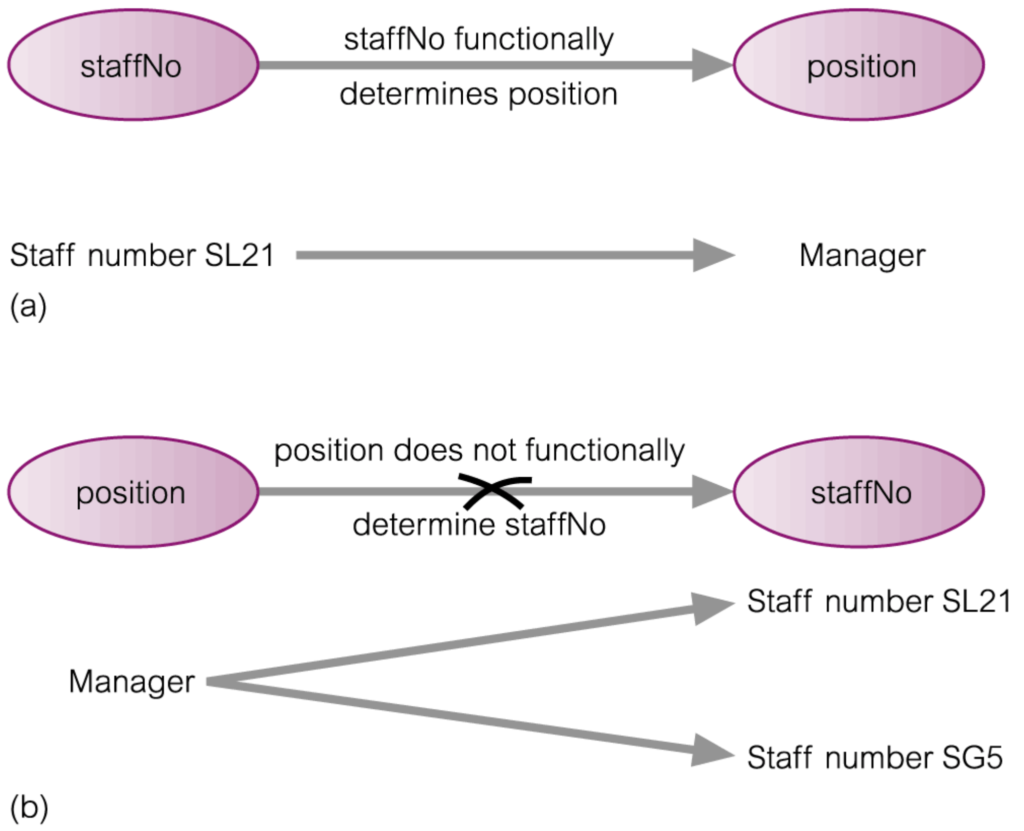
Diagrammatic representation.



The *determinant* of a functional dependency refers to the attribute or group of attributes on the left-hand side of the arrow.

AN EXAMPLE FUNCTIONAL DEPENDENCY

Figure 13.5
(a) staffNo functionally determines position (staffNo → position); (b) position does *not* functionally determine staffNo (position $\nrightarrow$ staffNo).



CHARACTERISTICS OF FUNCTIONAL DEPENDENCIES

Determinants should have the minimal number of attributes necessary to maintain the functional dependency with the attribute(s) on the right hand-side.

This requirement is called *full functional dependency.*

Full functional dependency indicates that if A and B are attributes of a relation, B is fully functionally dependent on A, if B is functionally dependent on A, but *not on any proper subset* of A.

EXAMPLE FULL FUNCTIONAL DEPENDENCY

- **Exists in the Staff relation.**

  staffNo, sName → branchNo

- **True -** each value of (staffNo, sName) is associated with a ***single*** value of branchNo.

- **However**, branchNo is also functionally dependent on a subset of (staffNo, sName), namely staffNo. Example above is **a *partial dependency.***

CHARACTERISTICS OF FUNCTIONAL DEPENDENCIES

Main characteristics of functional dependencies used in normalization:

- There is a *one-to-one* relationship between the attribute(s) on the left-hand side (determinant) and those on the right-hand side of a functional dependency.
- Holds for *all* time.
- The determinant has the *minimal* number of attributes necessary to maintain the dependency with the attribute(s) on the right hand-side.
- All attributes that are not part of the primary key should be functionally dependent on the key.

TRANSITIVE DEPENDENCIES

Important to recognize a transitive dependency because its existence in a relation can potentially cause update anomalies.

Transitive dependency describes a condition where A, B, and C are attributes of a relation such that ***if A → B and B → C, then C is transitively dependent on A via B*** (provided that A is not functionally dependent on B or C).

EXAMPLE TRANSITIVE DEPENDENCY

Consider functional dependencies in the StaffBranch relation (see Slide 5).

staffNo → sName, position, salary, branchNo, bAddress

branchNo → bAddress

Transitive dependency, branchNo → bAddress exists on staffNo via branchNo.

THE PROCESS OF NORMALIZATION

Formal technique for analysing a relation based on its primary key and the functional dependencies between the attributes of that relation.

The technique involves a series of rules that can be used to test individual relations so that a database can be normalized

to any degree.

When a requirement is not met, the relation violating the requirement must be decomposed into relations that individually meet the requirements of normalization.

Often executed as a series of steps. Each step corresponds to a specific normal form..

EXAMPLE

Examine semantics of attributes in StaffBranch relation. Assume that position held and branch determine a member of staff's salary. With sufficient information available, identify the functional dependencies for the StaffBranch relation as:

staffNo → sName, position, salary, branchNo, bAddress

branchNo → bAddress

bAddress → branchNo

branchNo, position → salary

bAddress, position → salary

THE PROCESS OF NORMALIZATION

- **Normalization** is a bottom-up approach to database design that begins by examining the relationships between attributes. It is performed as a serious of tests on a relation to determine whether it satisfies or violates the requirements of a given normal form.

- **Purpose:** Guarantees no redundancy due to FDs
- **Normal Forms:**
  - First Normal Form (1NF)
  - Second Normal Form (2NF)
  - Third Normal Form (3NF)

THE PROCESS OF NORMALIZATION

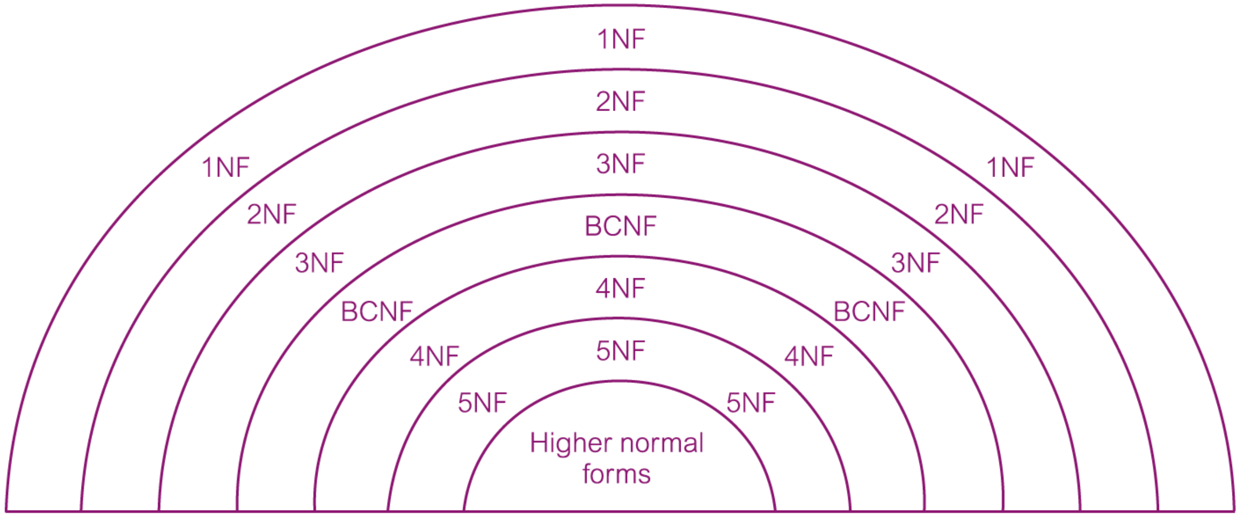


Figure 13.7 Diagrammatic illustration of the relationship between the normal forms.

PROCESS OF NORMALIZATION

In the following sections describe the process of normalization in detail. Figure 13.8 page 402 provides an overview of the process and highlights the main actions taken in each step of the process.

The number of the section that covers each step of the process is also shown in that figure.

PROCESS OF NORMALIZATION

- With the exception of 1NF, all normal forms are based on functional dependencies among the attributes of a relation.
- For the relational data model, it is important to recognize that it is only First Normal Form (1NF) that is critical in creating relations; all subsequent normal forms are optional.
- to avoid the update anomalies discussed before, it is generally recommended that we proceed to at least Third Normal Form (3NF).

UNNORMALIZED FORM (UNF)

- **A table that contains *one or more repeating groups.***
- **To create an unnormalized table**
  - Transform the data from the information source (e.g. form) into table format with columns and rows.

CLIENT_PROGRAM

ClientNo	Name	PrgramNo
CR76	John Key	PG4 PG16
CR56	Aline Stewart	PG4 PG36 PG16

Repeating group

Unnormalized table

FIRST NORMAL FORM (1NF)

- **A relation in which the intersection of each row and column contains *one and only one value.***
- **A repeating group is an attribute, or group of attributes, within a table that occurs with multiple values for a single occurrence of the nominated key attribute(s) for that table.**

clientNo	name	programNo
CR76	John Key	PG4 PG6
CR56	Aline Stewart	PG4 PG16 PG36

Unnormalized table

UNF TO 1NF

There are three common approaches to removing repeating groups from unnormalized tables:

- By entering appropriate data in the empty columns of rows containing the repeating data. (fill in the blanks by duplicating the nonrepeating data, where required.) This approach is commonly referred to as ‘flattening’ the table.
- If the maximum number of values is known for the attribute, replace repeated attribute with a number of atomic attributes.
- By placing the repeating data, along with a copy of the original key attribute(s), in a separate relation.

For all approaches, the resulting tables are now referred to as 1NF relations containing atomic (or single) values at the intersection of each row and column

UNF TO 1NF – APPROACH 1

- **Expand the key so that there will be a separate tuple in the original relation for each repeated attribute(s).**
- **Primary key becomes the combination of primary key and redundant value**

CLIENT_PROGRAM

ClientNo	Name	propertyNo
CR76	John Key	PG4
CR76	John Key	PG16
CR56	Aline Stewart	PG4
CR56	Aline Stewart	PG36
CR56	Aline Stewart	PG16

- **Disadvantage:** introduce redundancy in the relation

UNF TO 1NF – APPROACH 2

- **If the maximum number of values is known for the attribute, replace repeated attribute (ProgramNo) with a number of atomic attributes (ProgramNo1, ProgramNo2, ProgramNo3)**

CLIENT_property

<u>ClientNo</u>	Name	PropertyNo1	propertyNo2	propertyNo3
CR76	John Key	PG4	PG16	NULL
CR56	Aline Stewart	PG4	PG36	PG16

- **Disadvantage:** introduce NULL values in the relation

UNF TO 1NF – APPROACH 3

- **Remove the attribute that violates the 1NF and place it in a separate relation.**
- **Primary key of the new relation becomes the combination of primary key and redundant value**

CLIENT

<u>ClientNo</u>	Name
CR76	John Key
CR56	Aline Stewart

1NF relation

PROGRAM

<u>ClientNo</u>	<u>propertyNo</u>
CR76	PG4
CR76	PG16
CR56	PG4
CR56	PG36
CR56	PG16

1NF relation

SECOND NORMAL FORM (2NF)

- **Based on the concept of *full functional dependency*.**
- **Second Normal Form applies to relations with composite**
- **keys, that is, relations with a primary key composed of two or more attributes.**
- **A relation with a single-attribute primary key is automatically in at least 2NF.**
- **A relation that is not in 2NF may suffer from the update anomalies**
- **Full functional dependency indicates that if:**
    - 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.

SECOND NORMAL FORM (2NF)

A relation that is in 1NF and every non-primary-key attribute is fully functionally dependent on the primary key.

CLIENT_RENTAL

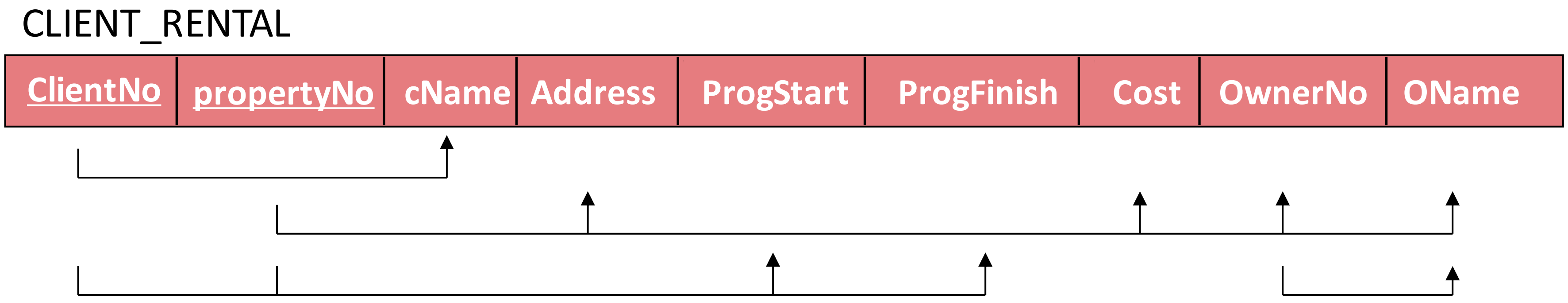
<u>ClientNo</u>	<u>propertyNo</u>	cName	Address	ProgStart	ProgFinish	Cost	OwnerNo	OName

1NF relation

1NF TO 2NF

- Identify the primary key for the 1NF relation.
- Identify the functional dependencies in the relation.
- If partial dependencies exist on the primary key remove them by placing then in a new relation along with a copy of their determinant.

1NF TO 2NF



(ClientNo, propertyNo)   PK

ClientNo, ProgramNo → ProgStart, ProgFinish		**Full Dependency**
ClientNo → CName		**Partial Dependency**
propertyNo → Paddress, Cost, OwnerNo, Oname		**Partial Dependency**

1NF TO 2NF

- **Remove partial dependencies by placing the functionally dependent attributes in a new relation along with a copy of their determinants**

CLIENT

ClientNo	cName

2NF relation

RENTAL

ClientNo	propertyNo	ProgStart	ProgFinish

2NF relation

property_OWNER

ProgramNo	pAddress	Cost	OwnerNo	OName

2NF relation

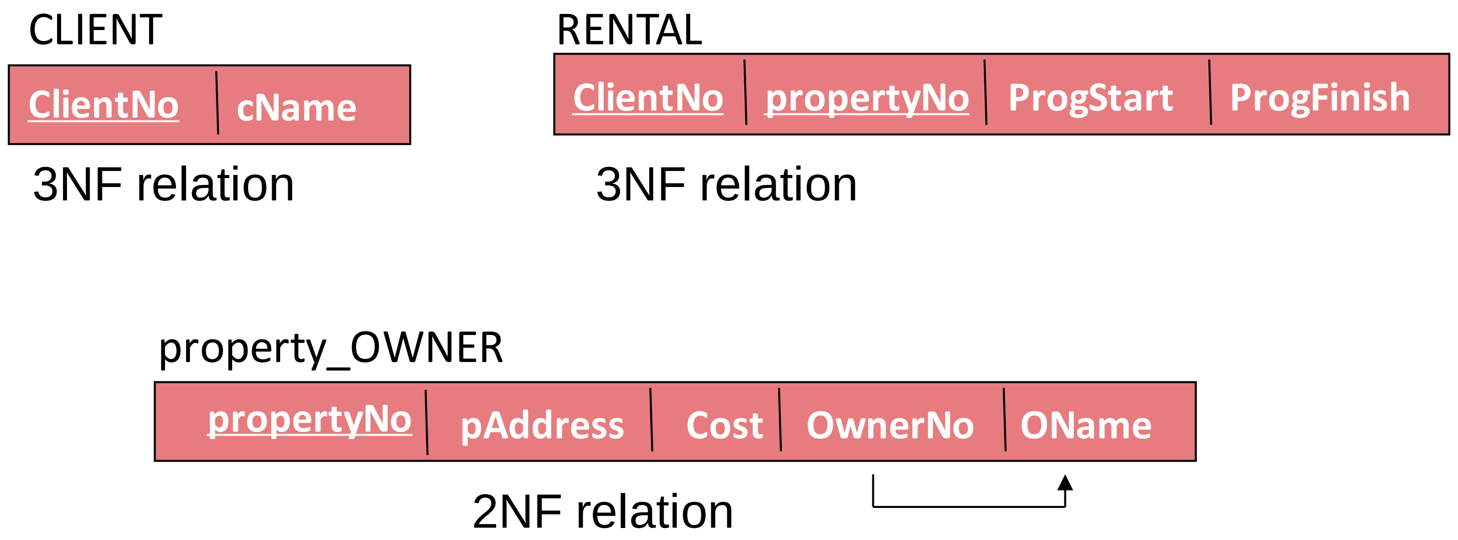
THIRD NORMAL FORM (3NF)

- **Based on the concept of *transitive dependency.***

- **Transitive Dependency is a condition where**
  - 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).

THIRD NORMAL FORM (3NF)

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



2NF TO 3NF

- Identify the primary key in the 2NF relation.
- Identify functional dependencies in the relation.
- If transitive dependencies exist on the primary key remove them by placing them in a new relation along with a copy of their dominant.

OWNER

OwnerNo	OName

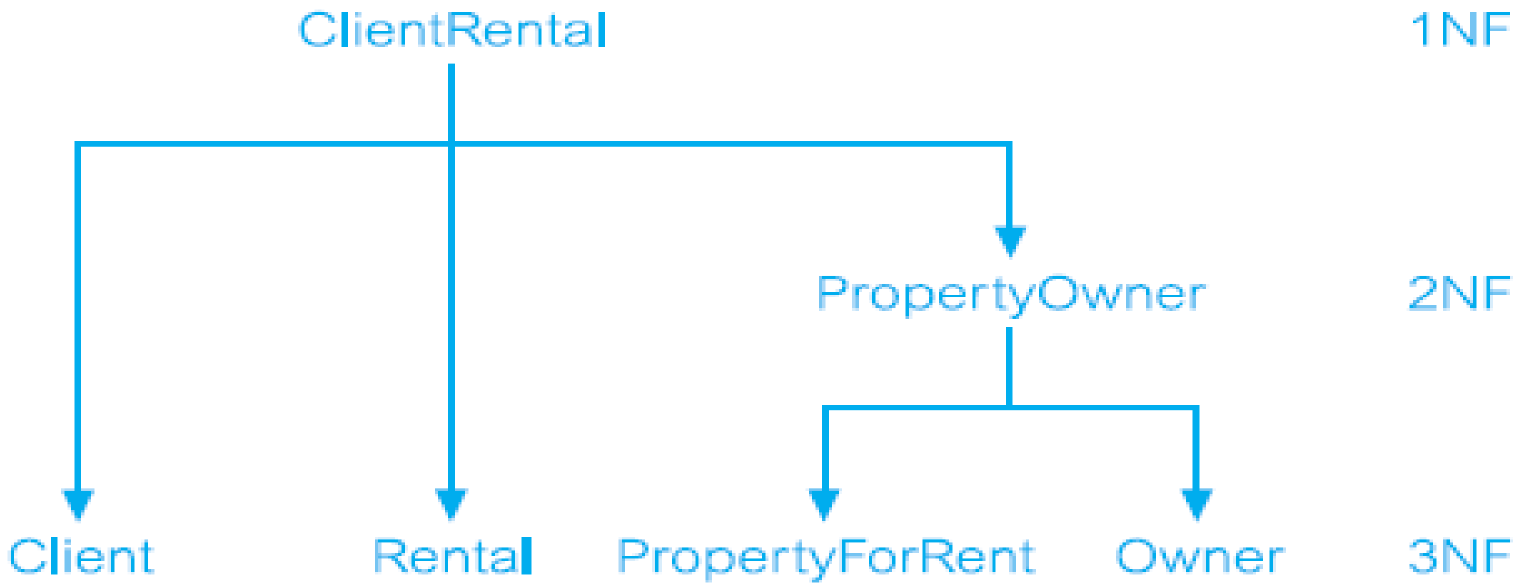
3NF relation

PROGRAM

ProgramNo	pAddress	Cost	OwnerNo

3NF relation

THE DECOMPOSITION OF THE CLIENTRENTAL



Client (clientNo, cName)

Rental (clientNo, propertyNo, rentStart, rentFinish)

PropertyForRent (propertyNo, pAddress, rent, ownerNo)

Owner (ownerNo, oName)

SUMMARY

- Redundant data problems in base relations.
- Functional dependency concept.
- Normalization process.
- First Normal Form (1NF), Second Normal Form (2NF), and Third Normal Form (3NF).