14.5 The Process of Normalization

Normalization is a formal technique for analyzing relations based on their primary key (or candidate keys) and functional dependencies (Codd, 1972b). 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.

Three normal forms were initially proposed called First Normal Form (1NF), Second Normal Form (2NF), and Third Normal Form (3NF). Subsequently, R. Boyce and E. F. Codd introduced a stronger definition of third normal form called Boyce–Codd Normal Form (BCNF) (Codd, 1974). With the exception of 1NF, all these normal forms are based on functional dependencies among the attributes of a relation (Maier, 1983). Higher normal forms that go beyond BCNF were introduced later such as Fourth Normal Form (4NF) and Fifth Normal Form (5NF) (Fagin, 1977, 1979). However, these later normal forms deal with situations that are very rare. In this chapter we describe only the first three normal forms and leave discussion of BCNF, 4NF, and 5NF to the next chapter.

Normalization is often executed as a series of steps. Each step corresponds to a specific normal form that has known properties. As normalization proceeds, the relations become progressively more restricted (stronger) in format and also less vulnerable to update anomalies. 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. However, to avoid the update anomalies discussed in Section 14.3, it is generally recommended that we proceed to at least Third Normal Form (3NF). Figure 14.7 illustrates the relationship between the various normal forms. It shows that some 1NF relations are also in 2NF, and that some 2NF relations are also in 3NF, and so on.

In the following sections we describe the process of normalization in detail. Figure 14.8 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 this figure.

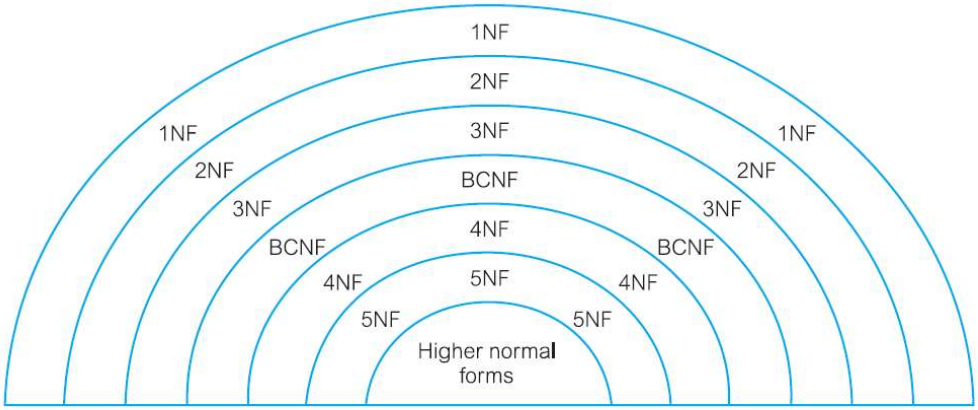
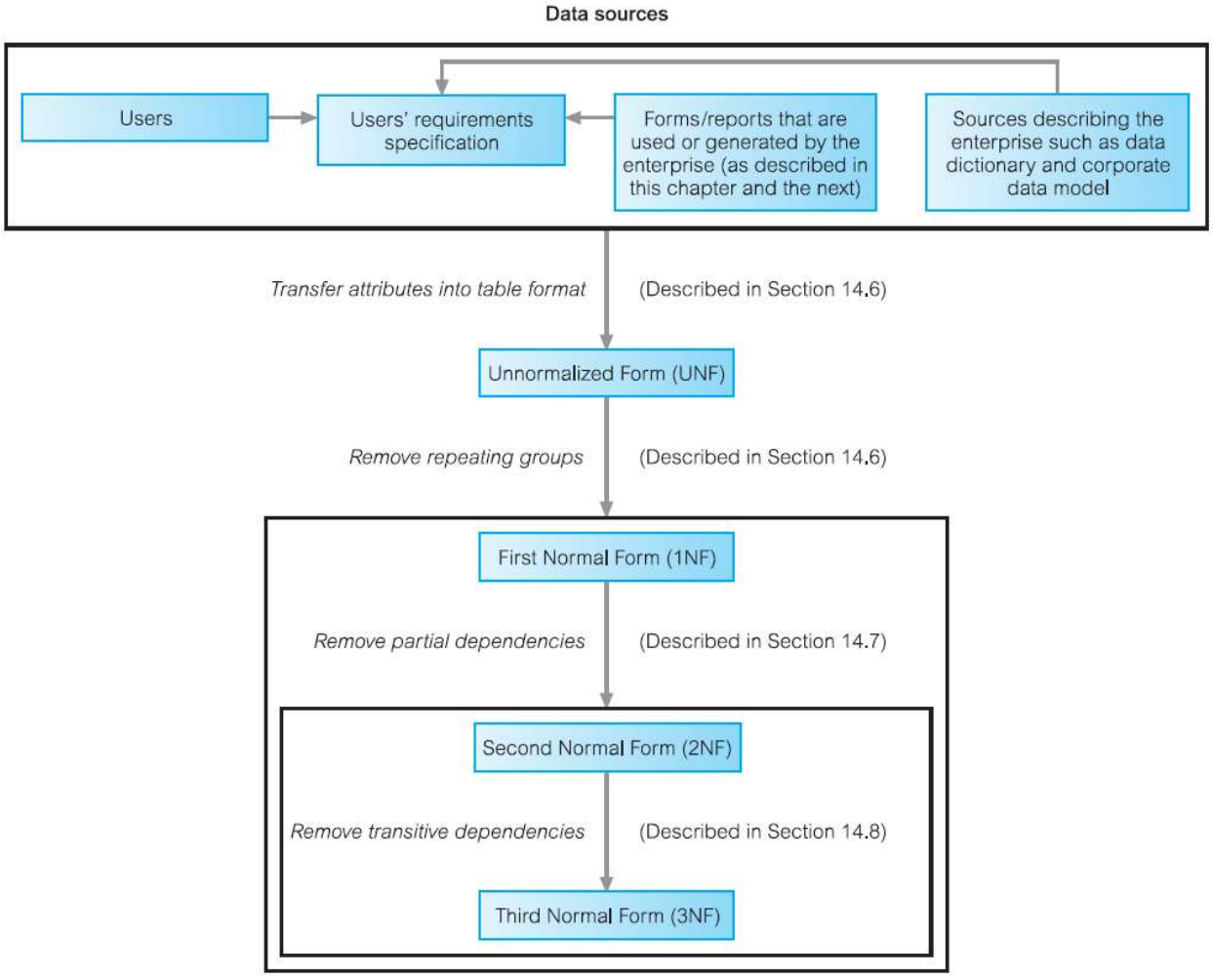


Figure 14.7 Diagrammatic illustration of the relationship between the normal forms.



In this chapter, we describe normalization as a bottom-up technique extracting information about attributes from sample forms that are first transformed into table format, which is described as being in Unnormalized Form (UNF). This table is then subjected progressively to the different requirements associated with each normal form until ultimately the attributes shown in the original sample forms are represented as a set of 3NF relations. Although the example used in this chapter proceeds from a given normal form to the one above, this is not necessarily the case with other examples. As shown in Figure 13.8, the resolution of a particular problem with, say, a 1NF relation may result in the relation being transformed to 2NF relations, or in some cases directly into 3NF relations in one step.

To simplify the description of normalization we assume that a set of functional dependencies is given for each relation in the worked examples and that each relation has a designated primary key. In other words, it is essential that the meaning of the attributes and their relationships is well understood before beginning the process of normalization. This information is fundamental to normalization and is used to test whether a relation is in a particular normal form. In Section 14.6 we begin by describing First Normal Form (1NF). In Sections 14.7 and 14.8 we describe Second Normal Form (2NF) and Third Normal Forms (3NF) based on the *primary key* of a relation and then present a more general definition of each in Section 14.9. The more general definitions of 2NF and 3NF take into account all *candidate keys* of a relation, rather than just the primary key.

14.6 First Normal Form (1NF)

Before discussing First Normal Form, we provide a definition of the state prior to First Normal Form.

Unnormalized Form (UNF) — A table that contains one or more repeating groups.

First Normal Form (1NF) — A relation in which the intersection of each row and column contains one and only one value.

In this chapter, we begin the process of normalization by first transferring the data from the source (for example, a standard data entry form) into table format with rows and columns. In this format, the table is in unnormalized Form and is referred to as an **unnormalized table**. To transform the unnormalized table to First Normal Form, we identify and remove repeating groups within the table. 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. Note that in this context, the term "key" refers to the attribute(s) that uniquely identify each row within the Unnormalized table. There are two common approaches to removing repeating groups from unnormalized tables:

(1) *By entering appropriate data in the empty columns of rows containing the repeating data.* In other words, we fill in the blanks by duplicating the nonrepeating data, where required. This approach is commonly referred to as "flattening" the table.

(2) *By placing the repeating data, along with a copy of the original key attribute(s), in a separate relation.* Sometimes the unnormalized table may contain more than one repeating group, or repeating groups within repeating groups. In such cases, this approach is applied repeatedly until no repeating groups remain. A set of relations is in 1NF if it contains no repeating groups.

For both approaches, the resulting tables are now referred to as 1NF relations containing atomic (or single) values at the intersection of each row and column. Although both approaches are correct, approach 1 introduces more redundancy into the original UNF table as part of the "flattening" process, whereas approach 2 creates two or more relations with less redundancy than in the original UNF table. In other words, approach 2 moves the original UNF table further along the normalization process than approach 1. However, no matter which initial approach is taken, the original UNF table will be normalized into the same set of 3NF relations.

We demonstrate both approaches in the following worked example using the *DreamHome* case study.

EXAMPLE 14.9 First Normal Form (1NF)

A collection of (simplified) *DreamHome* leases is shown in Figure 14.9. The lease on top is for a client called John Kay who is leasing a property in Glasgow, which is owned by Tina Murphy. For this worked example, we assume that a client rents a given property only once and cannot rent more than one property at any one time.

Figure 14.9 Collection of (simplified) *DreamHome* leases.

Sample data is taken from two leases for two different clients called John Kay and Aline Stewart and is transformed into table format with rows and columns, as shown in Figure 14.10. This is an example of an unnormalized table.

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

Figure 14.10 ClientRental unnormalized table.

We identify the key attribute for the ClientRental unnormalized table as clientNo. Next, we identify the repeating group in the unnormalized table as the property rented details, which repeats for each client. The structure of the repeating group is:

Repeating Group = (propertyNo, pAddress, rentStart, rentFinish, rent, ownerNo, oName)

As a consequence, there are multiple values at the intersection of certain rows and columns. For example, there are two values for propertyNo (PG4 and PG16) for the client named John Kay. To transform an unnormalized table into 1NF, we ensure that there is a single value at the intersection of each row and column. This is achieved by removing the repeating group.

With the first approach, we remove the repeating group (property rented details) by entering the appropriate client data into each row. The resulting first normal form ClientRental relation is shown in Figure 14.11.

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

Figure 14.11 First Normal Form ClientRental relation.

In Figure 14.12, we present the functional dependencies (fdl to fd6) for the ClientRental relation. We use the functional dependencies (as discussed in Section 14.4.3) to identify candidate keys for the ClientRental relation as being composite keys comprising (clientNo, propertyNo), (clientNo, rentStart), and (propertyNo, rentStart). We select (clientNo, propertyNo) as the primary key for the relation, and for clarity we place the attributes that make up the primary key together at the left-hand side of the relation. In this example, we assume that the rentFinish attribute is not appropriate as a component of a candidate key as it may contain nulls (see Section 4.3.1).

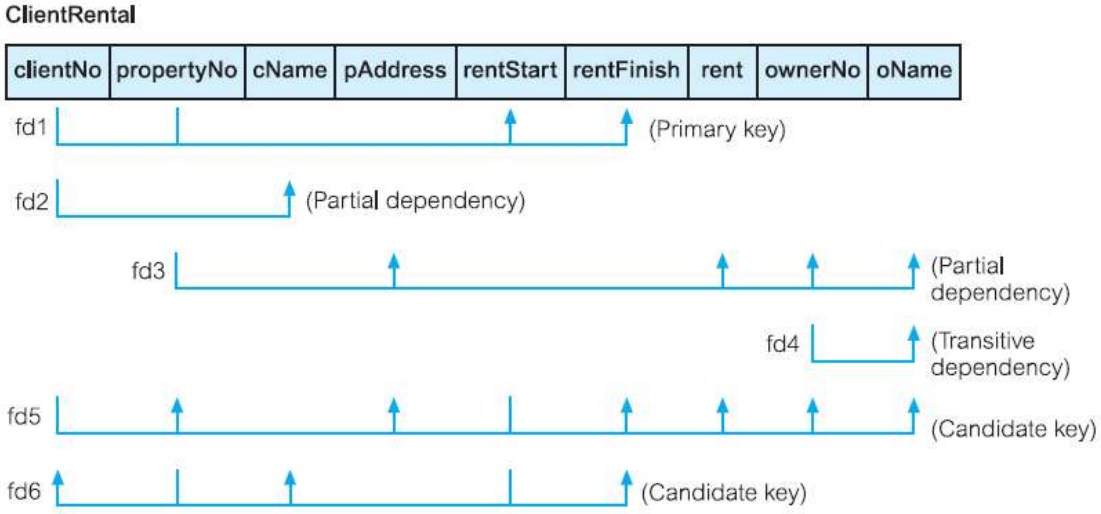


Figure 14.12 Functional dependencies of the ClientRental relation.

The ClientRental relation is defined as follows:

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

The ClientRental relation is in 1NF, as there is a single value at the intersection of each row and column. The relation contains data describing clients, property rented, and property owners, which is repeated several times. As a result, the ClientRental relation contains significant data redundancy. If implemented, the 1NF relation would be subject to the update anomalies described in Section 14.3. To remove some of these, we must transform the relation into second normal form, which we discuss shortly.

With the second approach, we 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, as shown in Figure 14.13.

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

Figure 14.13 Alternative 1NF Client and PropertyRental-Owner relations.

With the help of the functional dependencies identified in Figure 14.12 we identify a primary key for the relations. The format of the resulting 1NF relations are as follows:

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

The Client and PropertyRentalOwner relations are both in 1NF, as there is a single value at the intersection of each row and column. The Client relation contains data describing clients and the PropertyRentalOwner relation contains data describing property rented by clients and property owners. However, as we see from Figure 14.13, this relation also contains some redundancy and as a result may suffer from similar update anomalies to those described in Section 14.3.

To demonstrate the process of normalizing relations from 1NF to 2NF, we use only the ClientRental relation shown in Figure 14.11. However, recall that both approaches are correct, and will ultimately result in the production of the same relations as we continue the process of normalization. We leave the process of completing the normalization of the Client and PropertyRentalOwner relations as an exercise for the reader, which is given at the end of this chapter.

14.7 Second Normal Form (2NF)

Second Normal Form (2NF) is based on the concept of full functional dependency, which we described in Section 14.4. 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 discussed in Section 14.3. For example, suppose we wish to change the rent of property number PG4. We have to update two tuples in the ClientRental relation in Figure 14.11. If only one tuple is updated with the new rent, this results in an inconsistency in the database.

Second Normal Form (2NF) A relation that is in first normal form and every non-primary-key attribute is fully functionally dependent on the primary key.

The normalization of 1NF relations to 2NF involves the removal of partial dependencies. 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. We demonstrate the process of converting 1NF relations to 2NF relations in the following example.

EXAMPLE 14.10 Second Normal Form (2NF)

As shown in Figure 14.12, the ClientRental relation has the following functional dependencies:

fd1	clientNo, propertyNo ® rentStart, rentFinish	(Primary key)
fd2	clientNo ® cName	(Partial dependency)
fd3	propertyNo ® pAddress, rent, ownerNo, oName	(Partial dependency)
fd4	ownerNo ® oName	(Transitive dependency)
fd5	clientNo, rentStart ® propertyNo, pAddress, rentFinish, rent, ownerNo, oName	(Candidate key)
fd6	propertyNo, rentStart ® clientNo, cName, rentFinish	(Candidate key)

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

Figure 14.14 Second normal form relations derived from the ClientRental relation.

Using these functional dependencies, we continue the process of normalizing the ClientRental relation. We begin by testing whether the ClientRental relation is in 2NF by identifying the presence of any partial dependencies on the primary key. We note that the client attribute (cName) is partially dependent on the primary key, in other words, on only the clientNo attribute (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). The property rented attributes (rentStart and rentFinish) are fully dependent on the whole primary key; that is the clientNo and propertyNo attributes (represented as fd1).

The identification of partial dependencies within the ClientRental relation indicates that the relation is not in 2NF. To transform the ClientRental relation into 2NF requires the creation of new relations so that the non-primary-key attributes are removed along with a copy of the part of the primary key on which they are fully functionally dependent. This results in the creation of three new relations called Client, Rental, and PropertyOwner, as shown in Figure 14.14. These three relations are in second normal form, as every non-primary-key attribute is fully functionally dependent on the primary key of the relation. The relations have the following form:

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

14.8 Third Normal Form (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 of Figure 14.14. 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, which we described in Section 14.4. We need to remove such dependencies by progressing to third normal form.

Third Normal Form (3NF)

A relation that is in first and second normal form and in which no non-primary-key attribute is transitively dependent on the primary key.

The normalization of 2NF relations to 3NF involves the removal of transitive dependencies. If a transitive dependency exists, we remove the transitively dependent attribute(s) from the relation by placing the attribute(s) in a new relation along with a copy of the determinant. We demonstrate the process of converting 2NF relations to 3NF relations in the following example.

EXAMPLE 14.11 Third Normal Form (3NF)

The functional dependencies for the Client, Rental, and PropertyOwner relations, derived in Example 14.10, are as follows:

Client

fd2	clientNo ® cName	(Primary key)

Rental

fd1	clientNo, propertyNo ® rentStart, rentFinish	(Primary key)
fd5′	clientNo, rentStart ® propertyNo, rentFinish	(Candidate key)
fd6′	propertyNo, rentStart ® clientNo, rentFinish	(Candidate key)

PropertyOwner

fd3	propertyNo ® pAddress, rent, ownerNo, oName	(Primary key)
fd4	ownerNo ® oName	(Transitive dependency)

All the non-primary-key attributes within the Client and Rental relations are functionally dependent on only their primary keys. The Client and Rental relations have no transitive dependencies and are therefore already in 3NF. Note that where a functional dependency (fd) is labeled with a prime (such as fd5'), this indicates that the dependency has altered compared with the original functional dependency shown in Figure 14.12.

All the non-primary-key attributes within the PropertyOwner relation are functionally dependent on the primary key, with the exception of oName, which is transitively dependent on ownerNo (represented as fd4). This transitive dependency was previously identified in Figure 14.12. To transform the PropertyOwner relation into 3NF, we must first remove this transitive dependency by creating two new relations called PropertyForRent and Owner, as shown in Figure 14.15. The new relations have the following form:

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

The PropertyForRent and Owner relations are in 3NF, as there are no further transitive dependencies on the primary key.

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

Figure 14.15 Third normal form relations derived from the PropertyOwner relation.

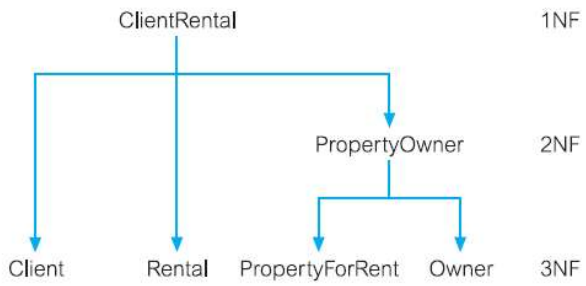


Figure 14.16
The decomposition of the ClientRental 1NF relation into 3NF relations.

The ClientRental relation shown in Figure 14.11 has been transformed by the process of normalization into four relations in 3NF. Figure 14.16 illustrates the process by which the original 1NF relation is decomposed into the 3NF relations. The resulting 3NF relations have the form:

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

The original ClientRental relation shown in Figure 14.11 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. The ownerNo attribute acting as a primary key/foreign key allows the association of the PropertyForRent and Owner relations to identify the name of property owners.

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.

In other words, the normalization process has decomposed the original ClientRental relation using a series of relational algebra projections (see Section 5.1). This results in a **lossless-join** (also called *nonloss-* or *nonadditive-*join) decomposition, which is reversible using the natural join operation. The Client, Rental, PropertyForRent, and Owner relations are shown in Figure 14.17.

14.9 General Definitions of 2NF and 3NF

The definitions for 2NF and 3NF given in Sections 14.7 and 14.8 disallow partial or transitive dependencies on the *primary key* of relations to avoid the update anomalies described in Section 14.3. However, these definitions do not take into account other candidate keys of a relation, if any exist. In this section, we present more general definitions for 2NF and 3NF that take into account candidate keys of a relation. Note that this requirement does not alter the definition for 1NF as this normal form is independent of keys and functional dependencies. For the general definitions, we state that a candidate-key attribute is part of any candidate key and

Figure 14.17
A summary of the 3NF relations derived from the ClientRental relation.

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

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

that partial, full, and transitive dependencies are with respect to all candidate keys of a relation.

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

Third Normal Form (3NF)
A relation that is in first and second normal form and in which no non-candidate-key attribute is transitively dependent on *any candidate key*.