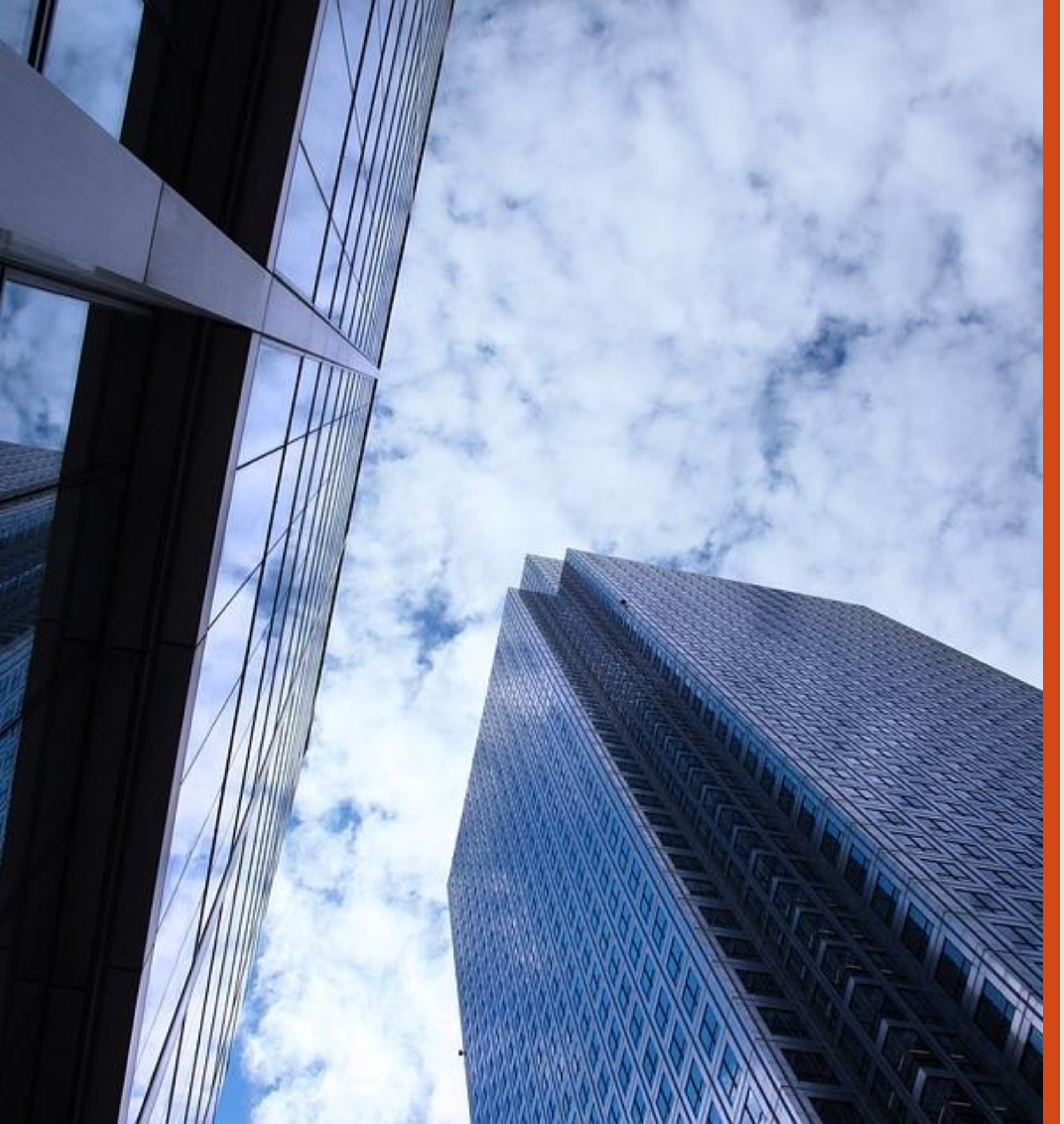


NORMALIZATION PART 03

Jayathma Chathurangani

ejc@ucsc.cmb.ac.lk

OUTLINE

- ✓ UNF
- ✓ INF
- ✓ 2NF
- ✓ 3NF

1

1 NF

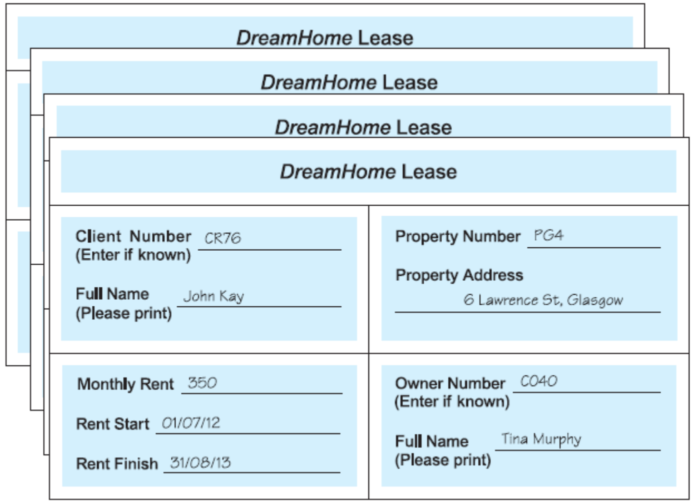
1.1 WHAT IS UNNORMALIZED FORM (UNF)?

- **A table that contains one or more repeating groups.**
- 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.
- Also note that you can come up with any meaningful name for a given relation.

1.1 WHAT IS UNNORMALIZED FORM (UNF)? (CONTD.)

Example

- A collection of (simplified) *DreamHome* leases is shown in Figure. 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.
- 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,



DreamHome Lease

DreamHome Lease

DreamHome Lease

DreamHome Lease

Client Number CR76
(Enter if known)

Full Name John Kay
(Please print)

Property Number PG4

Property Address
6 Lawrence St, Glasgow

Monthly Rent 350

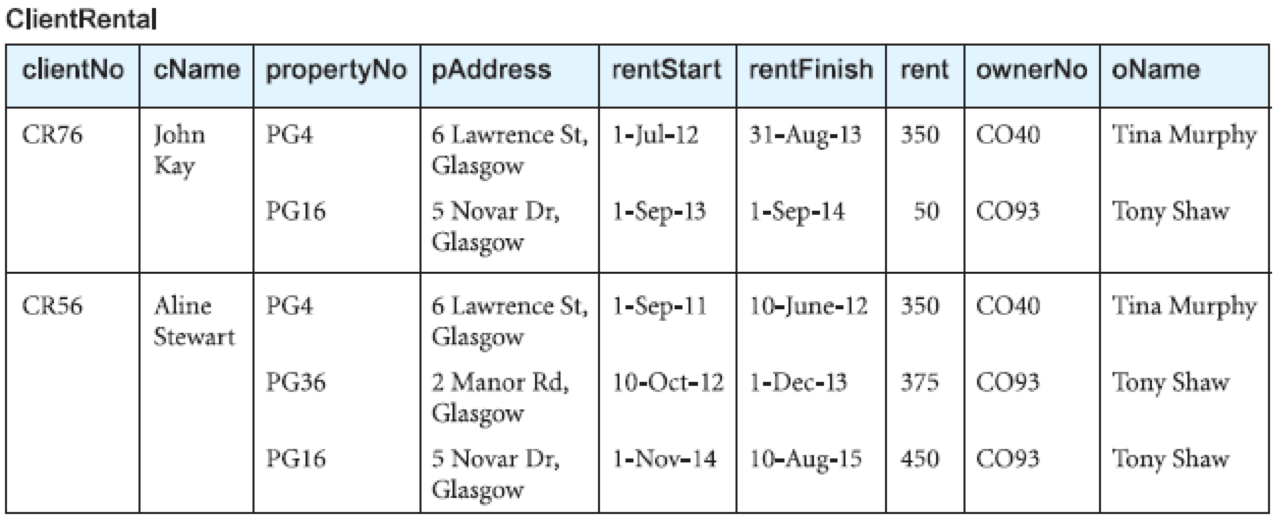
Rent Start 01/07/12

Rent Finish 31/08/13

Owner Number C040
(Enter if known)

Full Name Tina Murphy
(Please print)

Collection of (Simplified) DreamHome Leases



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

ClientRental Unnormalized table

1.1 WHAT IS UNNORMALIZED FORM (UNF)? (CONTD.)

Example (Contd.)

- 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.

1.2 WHAT IS 1NF?

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

- **To transform the unnormalized table to First Normal Form, we identify and remove repeating groups within the 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.

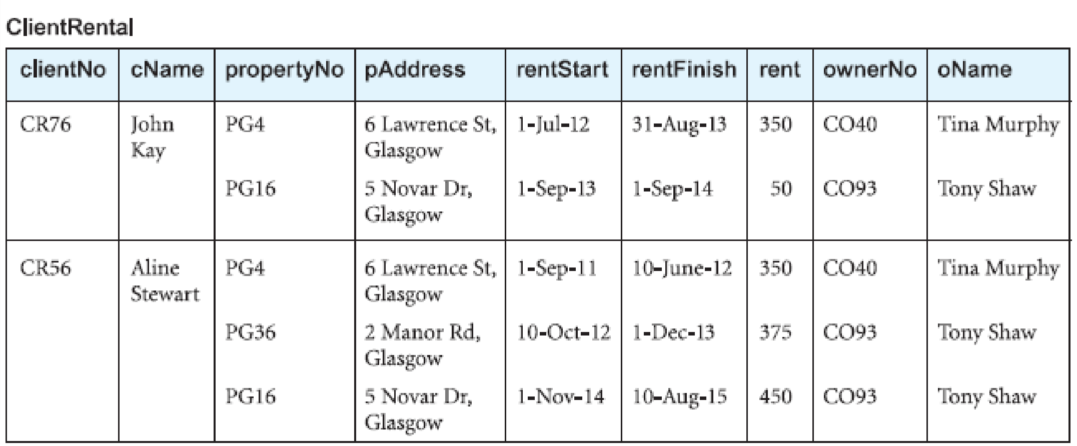
1.2 WHAT IS 1NF? (CONTD.)

- 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.

1.3 TRANSFORMING UNF TO 1NF?

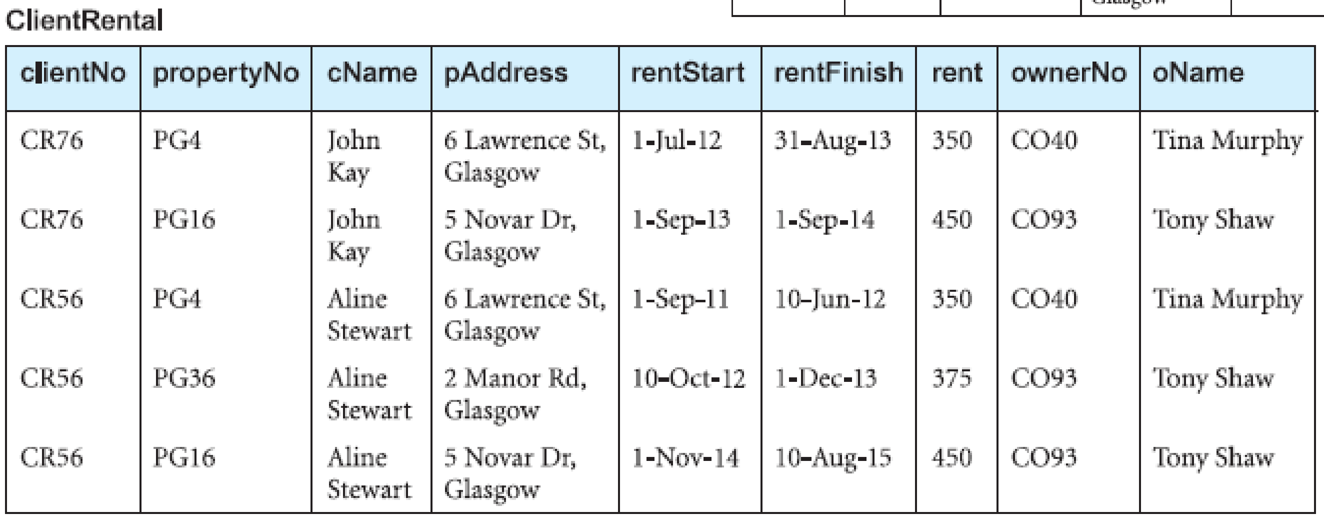
Example (Approach 01)

- 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.



ClientRental

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



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

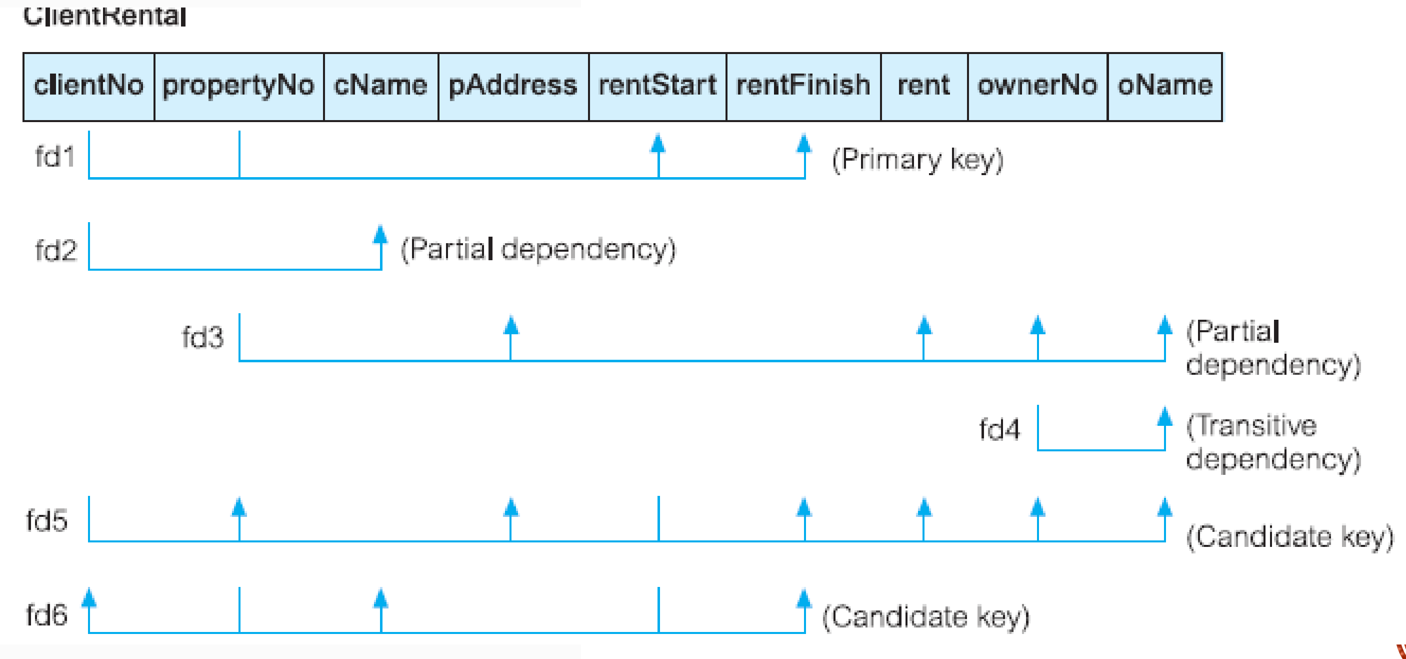
First Normal form ClientRental relation

Update Anomaly

1.3 TRANSFORMING UNF TO 1NF? (CONTD.)

Example (Approach 01) - Continued

- Functional dependencies (fd1 to fd6) for the **ClientRental** relation are given below.
- We use the functional dependencies (as discussed earlier) 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.



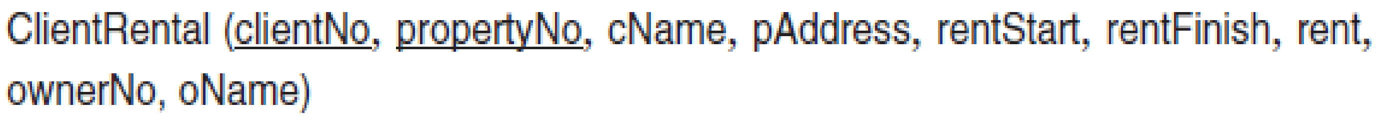
ClientRental

clientNo	propertyNo	cName	pAddress	rentStart	rentFinish	rent	ownerNo	oName

1.3 TRANSFORMING UNF TO 1NF? (CONTD.)

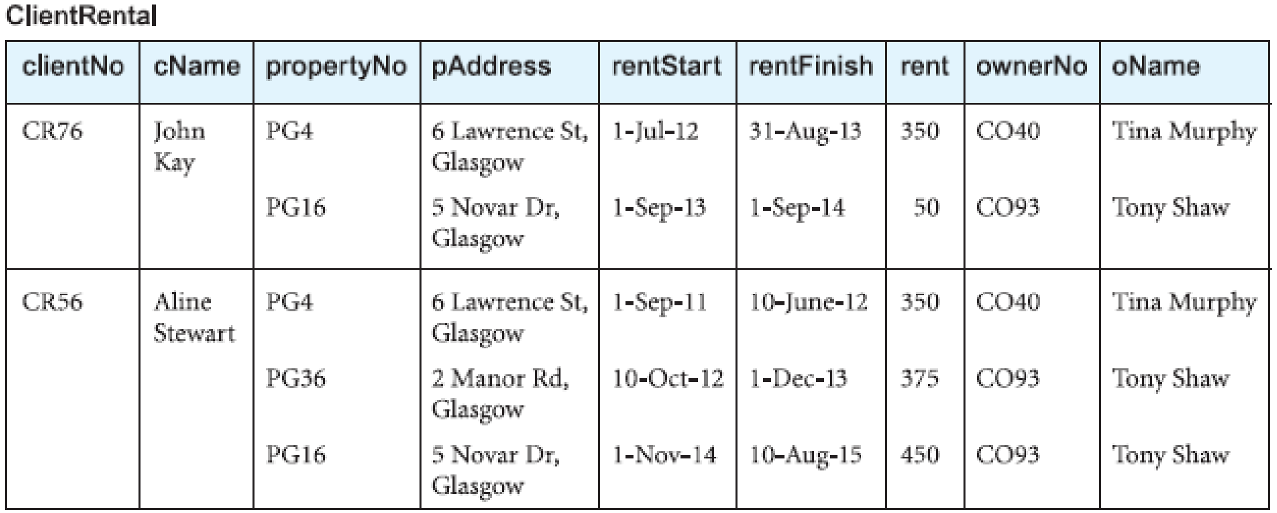
Example (Approach 01) - Continued

- 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 earlier.



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

- **To remove some of these, we have to transform the relation into second normal form.**



ClientRental

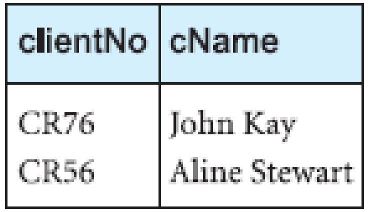
clientNo	cName	propertyNo	pAddress	rentStart	rentFinish	rent	ownerNo	oName
CR76	John Kay	PG4	6 Lawrence St, Glasgow	1-Jul-12	31-Aug-13	350	CO40	Tina Murphy
		PG16	5 Novar Dr, Glasgow	1-Sep-13	1-Sep-14	50	CO93	Tony Shaw
CR56	Aline Stewart	PG4	6 Lawrence St, Glasgow	1-Sep-11	10-June-12	350	CO40	Tina Murphy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

1.3 TRANSFORMING UNF TO 1NF? (CONTD.)

Example (Approach 02) - Alternative 1NF Client and PropertyRental-Owner relations.

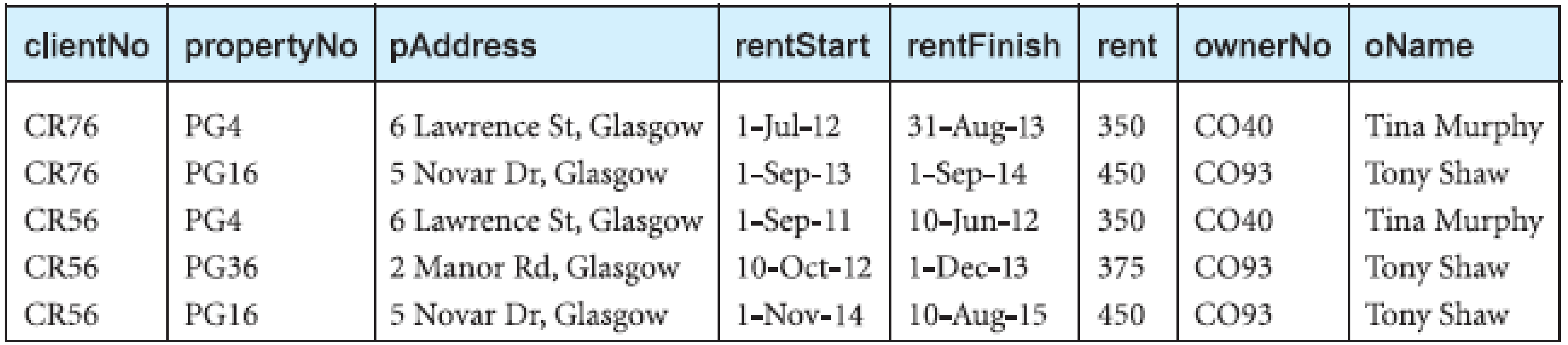
- We remove the repeating group (property rented details) by placing the repeating data along with a copy of the original key attribute (**clientNo**) in a separate relation, as shown.
- The **Client** and **PropertyRentalOwner** relations are both in 1NF, as there is a single value at the intersection 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.

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

Update Anomaly

1.3 TRANSFORMING UNF TO 1NF? (CONTD.)

Example (Approach 02) - Continued

- However, relations in both approaches contains some redundancy and as a result may suffer from similar update anomalies
- With the help of the functional dependencies identified in previously (mentioned in Approach 1) we identify a primary key for the relations. The format of the resulting 1NF relations are as follows:
  - Client (<u>clientNo</u>, cName)
  - PropertyRentalOwner (<u>clientNo</u>, <u>propertyNo</u>, pAddress, rentStart, rentFinish, rent,ownerNo, oName)
- For example, suppose we wish to change the rent of property number PG4. We have to update two tuples in the **ClientRental** relation. If only one tuple is updated with the new rent, this results in an inconsistency in the database.

2

2 NF

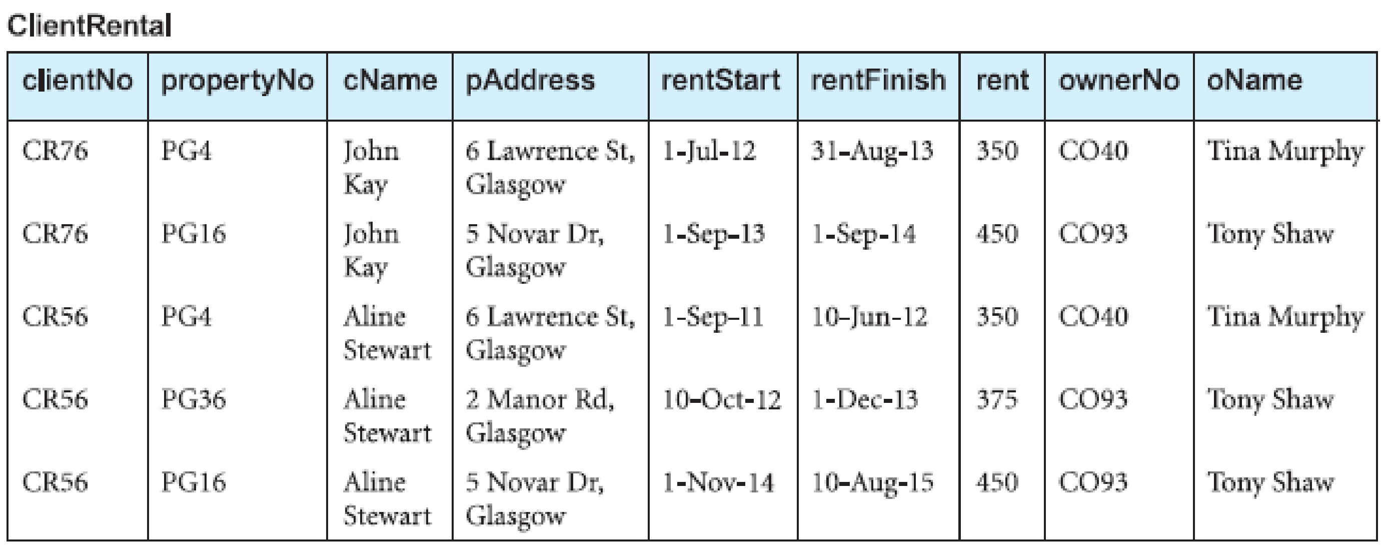
2.1 WHAT IS 2NF?

- **Second Normal Form (2NF) is A relation that is in first normal form and every non-primary-key attribute is fully functionally dependent on the primary key.**
- Second Normal Form (2NF) is 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 discussed in previous section.
- **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.
- *Some says, Here we remove redundancies (Microsoft)*

2.2 TRANSFORMING 1NF TO 2NF?

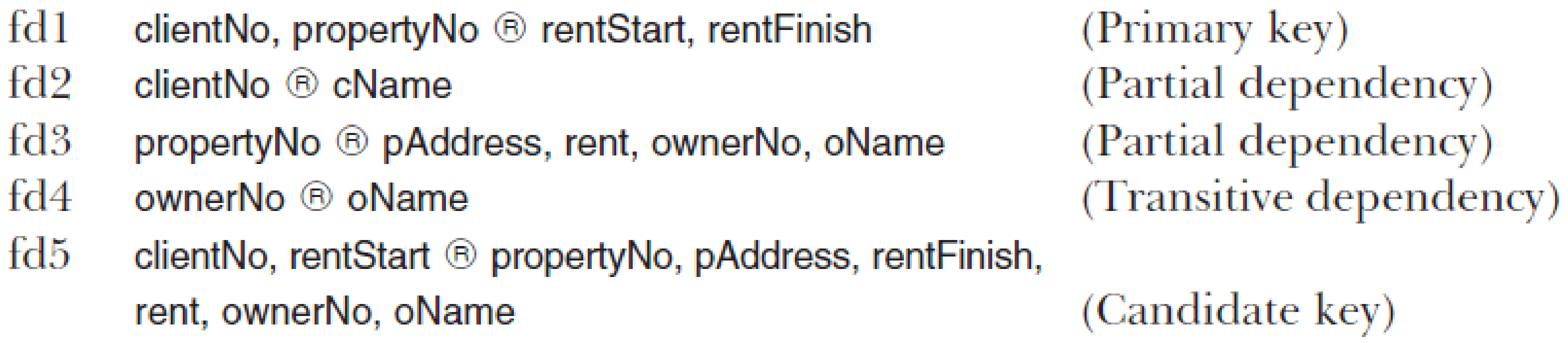
Example

- Consider **ClientRental** relation identified in 1NF using **approach 01**.
- It has below FDs as identified over there.
- Assuming PK as clientNo and propertyNo,



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



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)

2.2 TRANSFORMING 1NF TO 2NF? (CONTD.)

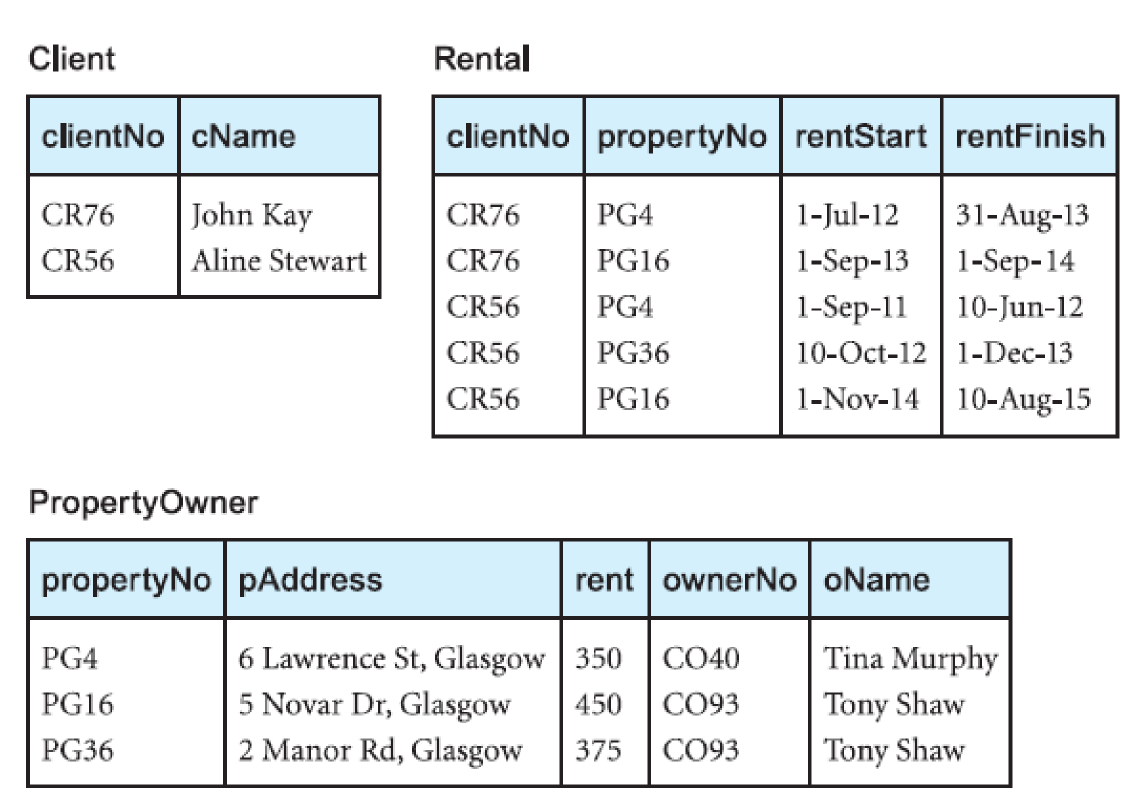
Example - Continued

- 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).

2.2 TRANSFORMING 1NF TO 2NF? (CONTD.)

Example - Continued

- 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.
- 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.



Client

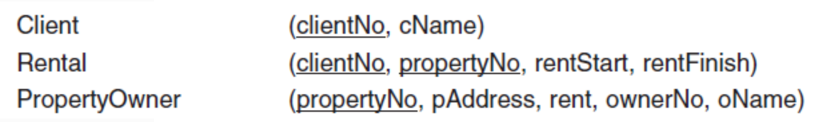
clientNo	cName
CR76	John Kay
CR56	Aline Stewart

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)

Update Anomaly

2.2 TRANSFORMING 1NF TO 2NF? (CONTD.)

Example - Continued

- 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.
- We need to remove such dependencies by progressing to third normal form.

3

3 NF

3.1 WHAT IS 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.

3.2 TRANSFORMING 2NF TO 3NF? (CONTD.)

Example

- The functional dependencies derived for the **Client**, **Rental**, and **PropertyOwner** relations are as follows.
- **Client, Rental are already in 3NF**

Client

fd2 clientNo ® cName (Primary key)

Rental

fd1 clientNo, propertyNo ® rentStart, rentFinish (Primary key)

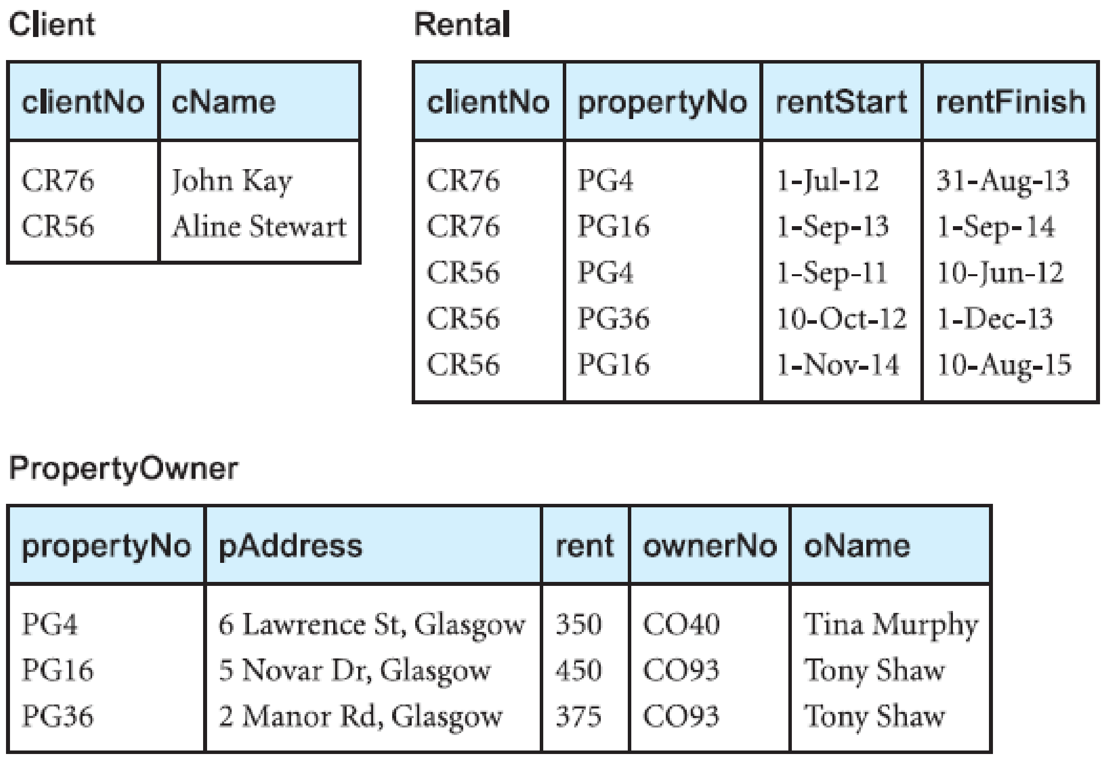
fd5′ clientNo, rentStart ® propertyNo, rentFinish (Candidate key)

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

PropertyOwner

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

fd4 ownerNo ® oName (Transitive dependency)



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)

3.2 TRANSFORMING 2NF TO 3NF? (CONTD.)

Example (Continued)

- 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 (Figure in slide 10).
- 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 in slide 10.
- To transform the **PropertyOwner** relation into 3NF, we must first remove this transitive dependency by creating two new relations called **PropertyForRent** and **Owner.**

3.2 TRANSFORMING 2NF TO 3NF? (CONTD.)

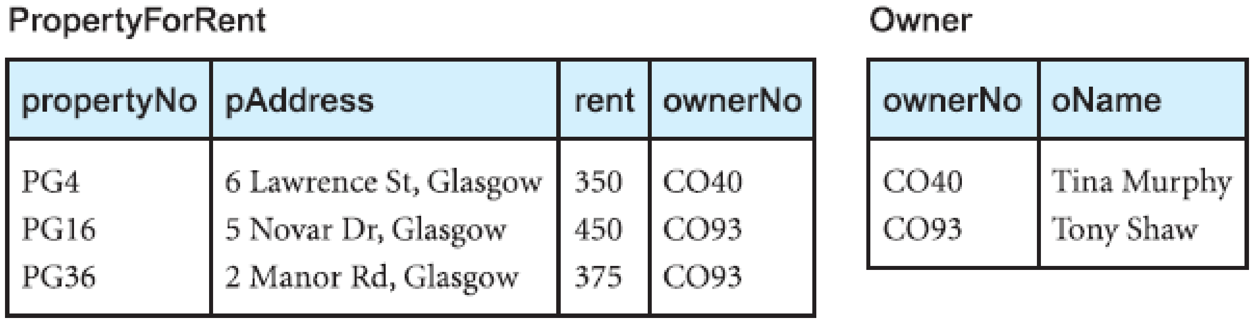
Example (Continued)

- The new relations have the following form:

  PropertyForRent (<u>propertyNo</u>, pAddress, rent, ownerNo)

  Owner (<u>ownerNo</u>, 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

3.2 TRANSFORMING 2NF TO 3NF? (CONTD.)

Example (Continued)

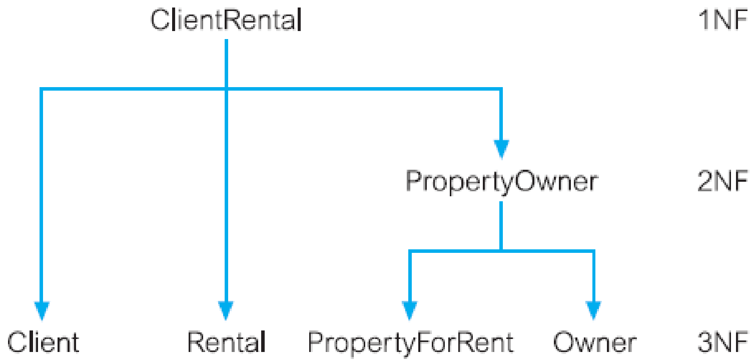
- The **ClientRental** relation has been transformed by the process of normalization i

  Client (<u>clientNo</u>, cName)

  Rental (<u>clientNo</u>, <u>propertyNo</u>, rentStart, rentFinish)

  PropertyForRent (<u>propertyNo</u>, pAddress, rent, ownerNo)

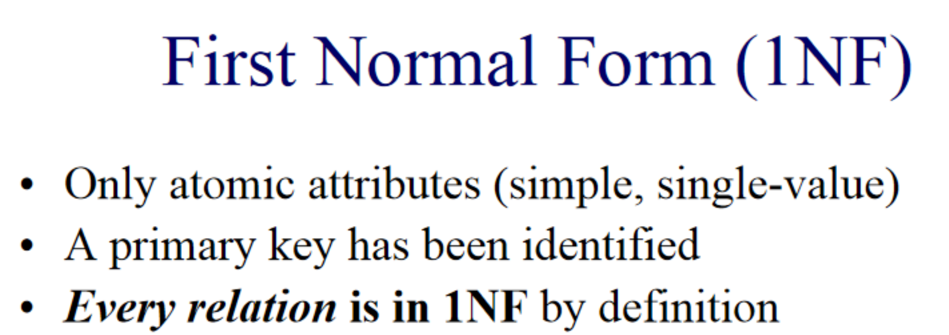
  Owner (<u>ownerNo</u>, oName)



- The original **ClientRental** relation be recreated by joining the **Client, Rental, PropertyForRent, and Owner** relations through the primary key/ foreign key mechanism.

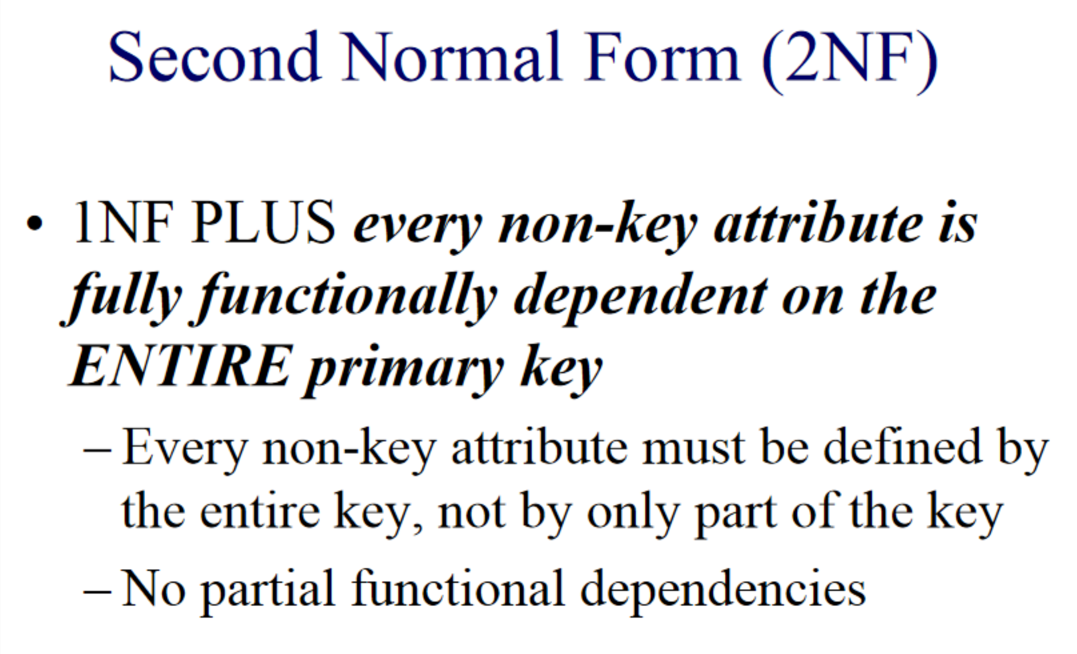
- For example, the **ownerNo** attribute is a primary key within the **Owner** relation and is also present within the **PropertyForRent** relation as a foreign key. The **ownerNo** attribute acting as a primary key/foreign key allows the association of the **PropertyForRent and Owner** relations to identify the name of property owners.

WRAP UP AND THANK YOU



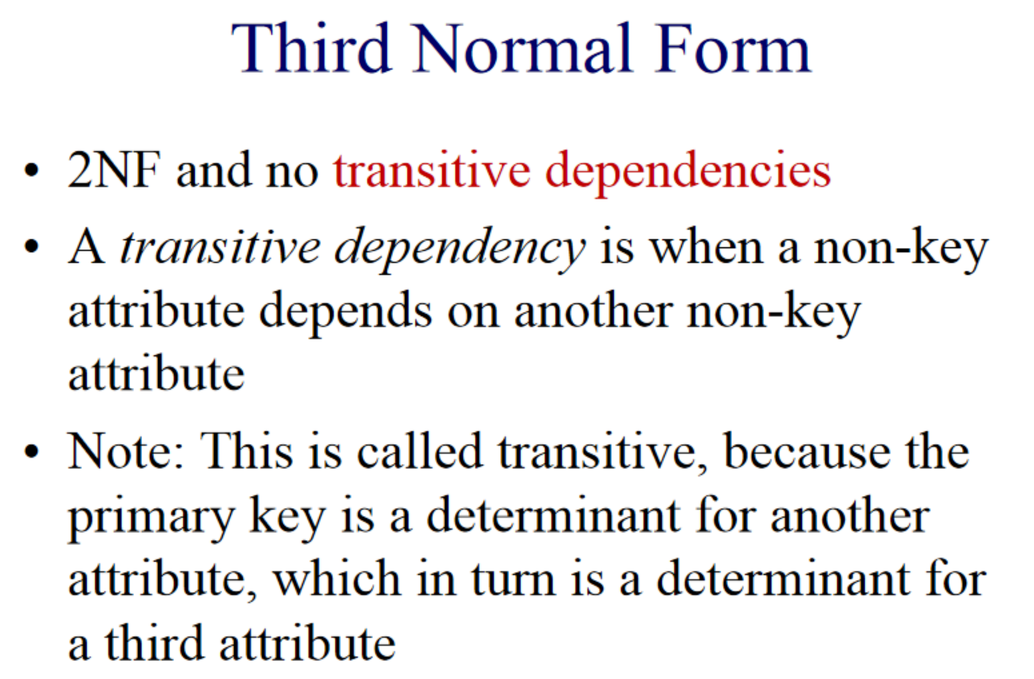
First Normal Form (1NF)

- Only atomic attributes (simple, single-value)
- A primary key has been identified
- ***Every relation*** **is in 1NF** by definition



Second Normal Form (2NF)

- 1NF PLUS ***every non-key attribute is fully functionally dependent on the ENTIRE primary key***
  - Every non-key attribute must be defined by the entire key, not by only part of the key
  - No partial functional dependencies



Third Normal Form

- 2NF and no transitive dependencies
- A *transitive dependency* is when a non-key attribute depends on another non-key attribute
- Note: This is called transitive, because the primary key is a determinant for another attribute, which in turn is a determinant for a third attribute