NORMALIZATION

Engr. Laraib Siddiqui

Normalization

- The goal is to generate a set of relation schemas that allows us to store information without unnecessary redundancy, yet also allows us to retrieve information easily. The approach is:
- Decide if a given relation schema is in "good form." There are a number of different forms (called *normal forms*)
- If a given relation schema is not in "good form," then we decompose it into a number of smaller relation schemas, each of which is in an appropriate normal form. The decomposition must be a lossless decomposition.

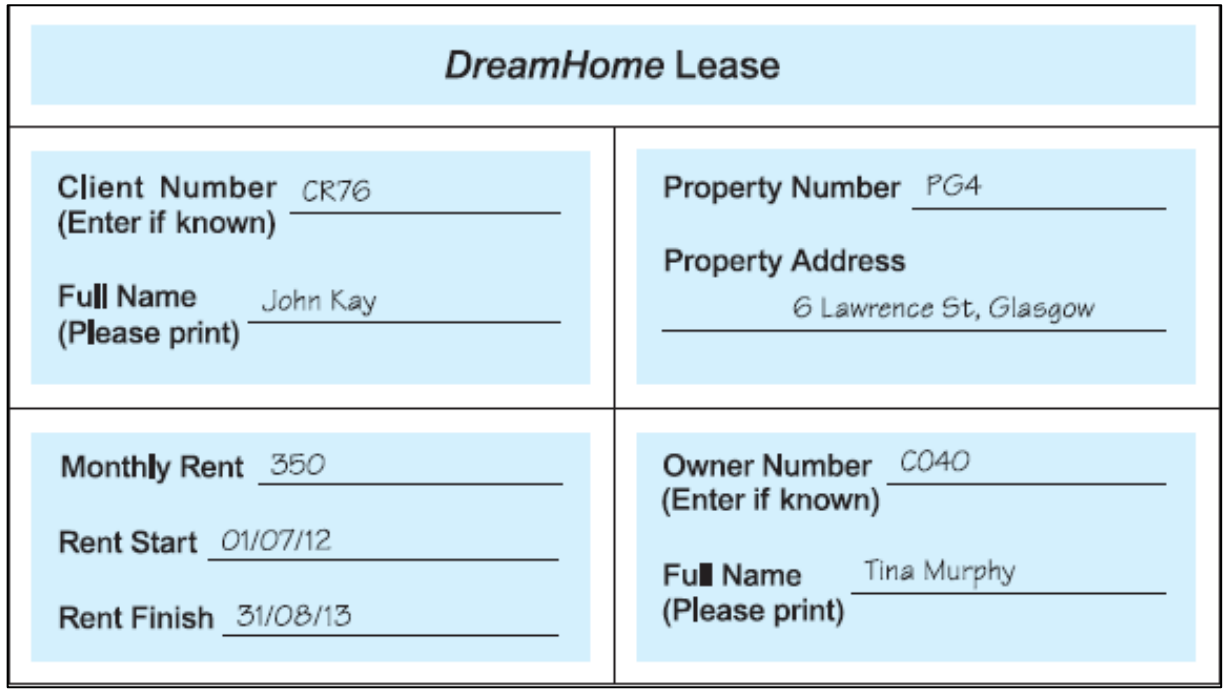
Anomalies

- **Insertion Anomaly** – We tried to insert data in a record that does not exist at all.
- **Deletion Anomaly** – We tried to delete a record, but parts of it was left undeleted because of unawareness, the data is also saved somewhere else.
- **Updation Anomaly** - If data items are scattered and are not linked to each other properly, then it could lead to strange situations. For example, when we try to update one data item having its copies scattered over several places, a few instances get updated properly while a few others are left with old values. Such instances leave the database in an inconsistent state.

First Normal Form

- A relation is in first normal form if every attribute in every tuple contains an atomic value
- There is no multivalued (repeating group) in the relation

Collection of (simplified) DreamHome leases



***DreamHome* Lease**

Client Number (Enter if known) CR76

Full Name (Please print) John Kay

Property Number PG4

Property Address 6 Lawrence St, Glasgow

Monthly Rent 350

Rent Start 01/07/12

Rent Finish 31/08/13

Owner Number (Enter if known) CO40

Full Name (Please print) Tina Murphy

ClientRental 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

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

Converting to 1NF

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.

First Normal Form ClientRental relation.

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

Redundancy

Now The ClientRental relation is in 1NF but 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.

To remove some of these, we must transform the relation into second normal form.

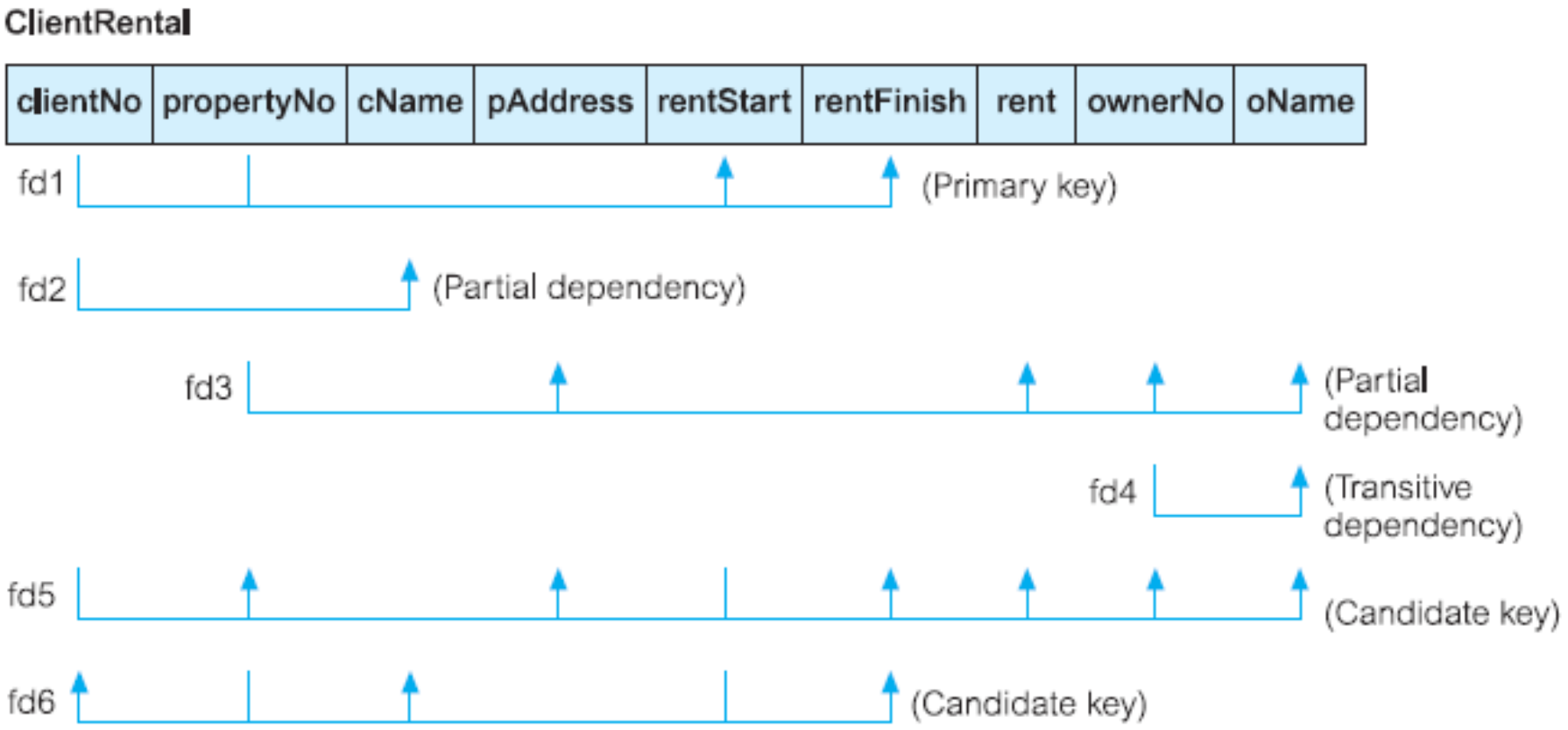
Second Normal Form

A relation that is in first normal form and every non-primary-key attribute is fully functionally dependent on the primary key.

Converting to 2NF

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.

Functional dependencies of the ClientRental relation



2NF relations derived from the ClientRental

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)

Third Normal Form

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.

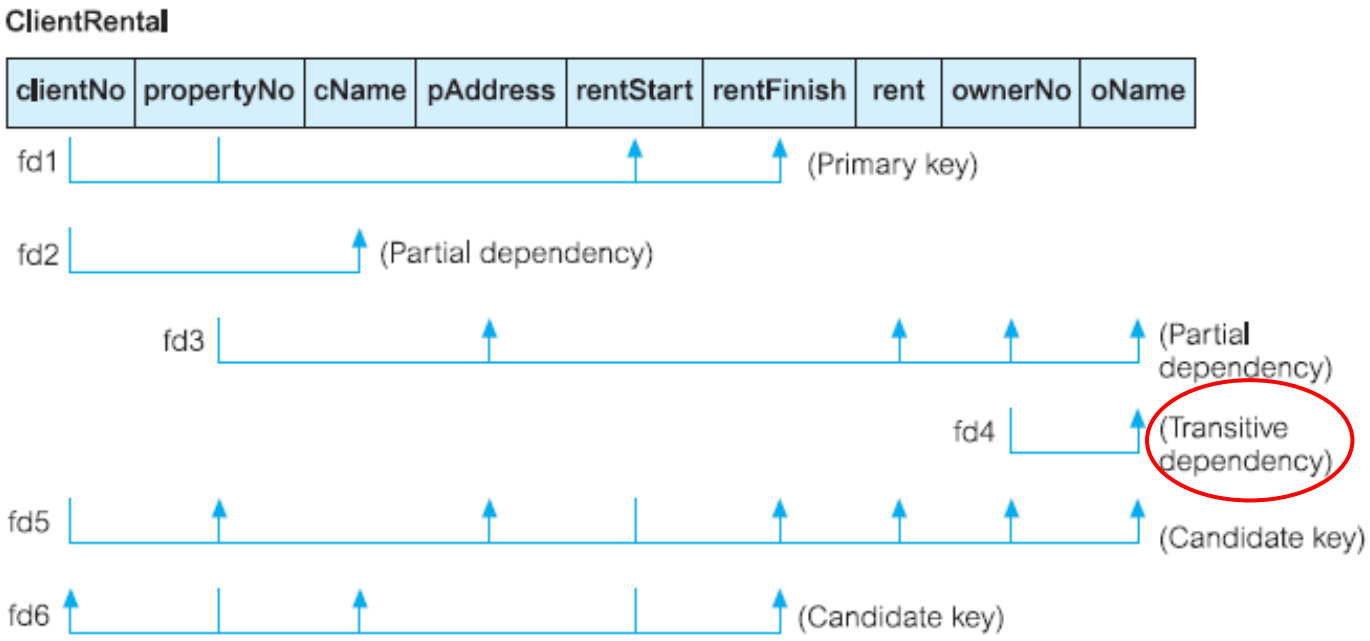
Third Normal Form

A table is in third normal form (3NF) if it is in 2NF and there is no transitive dependency, that is, no non-key attribute is dependent on another non-key attribute

Converting to 3NF

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 .

To transform the PropertyOwner relation into 3NF, we must first remove this transitive dependency by creating two new relations called PropertyForRent and Owner.



3NF

PropertyForRent

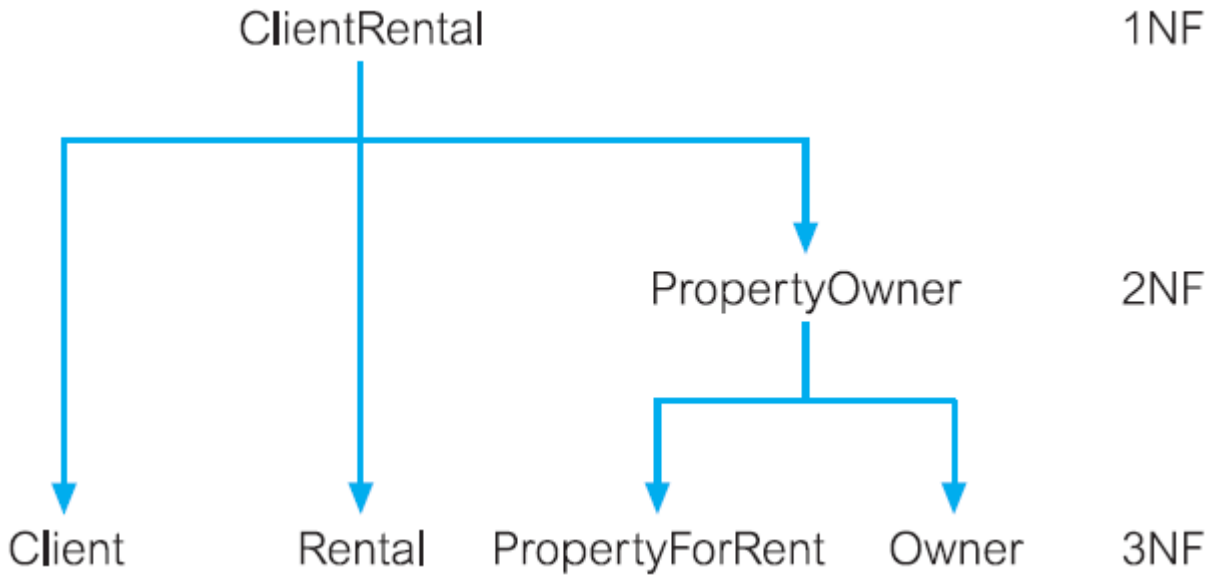
propertyNo	pAddress	rent	ownerNo
PG4	6 Lawrence St, Glasgow	350	CO40
PG16	5 Novar Dr, Glasgow	450	CO93
PG36	2 Manor Rd, Glasgow	375	CO93

Owner

ownerNo	oName
CO40	Tina Murphy
CO93	Tony Shaw

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

The decomposition of the ClientRental 1NF into 3NF



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)

Another Example

item	colors	price	tax
T-shirt	red, blue	12.00	0.60
polo	red, yellow	12.00	0.60
T-shirt	red, blue	12.00	0.60
sweatshirt	blue, black	25.00	1.25

Table is not in first normal form because:

- Multiple items in color field
- Duplicate records / no primary key

Example

item	color	price	tax
T-shirt	red	12.00	0.60
T-shirt	blue	12.00	0.60
polo	red	12.00	0.60
polo	yellow	12.00	0.60
sweatshirt	blue	25.00	1.25
sweatshirt	black	25.00	1.25

Table is now in first normal form.

But not in second normal form because:

price and tax depend on item, but not color

Example

item	color
T-shirt	red
T-shirt	blue
polo	red
polo	yellow
sweatshirt	blue
sweatshirt	black

item	price	tax
T-shirt	12.00	0.60
polo	12.00	0.60
sweatshirt	25.00	1.25

Now tables are in 2NF but not in third normal form because:

tax depends on price, not item

Example

item	color
T-shirt	red
T-shirt	blue
polo	red
polo	yellow
sweatshirt	blue
sweatshirt	black

item	price
T-shirt	12.00
polo	12.00
sweatshirt	25.00

price	tax
12.00	0.60
25.00	1.25

Tables are now in third normal form

Practice

staffNo	dentistName	patNo	patName	appointment date	time	surgeryNo
S1011	Tony Smith	P100	Gillian White	12-Sep-13	10.00	S15
S1011	Tony Smith	P105	Jill Bell	12-Sep-13	12.00	S15
S1024	Helen Pearson	P108	Ian MacKay	12-Sep-13	10.00	S10
S1024	Helen Pearson	P108	Ian MacKay	14-Sep-13	14.00	S10
S1032	Robin Plevin	P105	Jill Bell	14-Sep-13	16.30	S15
S1032	Robin Plevin	P110	John Walker	15-Sep-13	18.00	S13

1. Identify the functional dependencies represented by the attributes shown in the table.
2. Describe and illustrate the process of normalizing the attributes to produce a set of well-designed 3NF relations.
3. Identify the primary, alternate, and foreign keys in your 3NF relations.

Boyce-Codd Normal Form

- A general form of 3NF
- Every relation in BCNF is in 3NF vice-versa is not always true
- A table is in BCNF if every determinant is a candidate key

Boyce-Codd Normal Form

In the last example the Client, PropertyForRent, and Owner relations are all in BCNF, as each relation only has a single determinant, which is the candidate key. Except the following:

- fd1 clientNo, propertyNo -> rentStart, rentFinish
- fd5 clientNo, rentStart -> propertyNo, rentFinish
- fd6 propertyNo, rentStart -> clientNo, rentFinish

Rental relation is also already in BCNF.

The potential to violate BCNF may occur when:

- the relation contains two (or more) composite candidate keys or
- the candidate keys overlap, that is have at least one attribute in common.

Extended the DreamHome case study

Including a description of client interviews by members of staff.

The members of staff involved in interviewing clients are allocated to a specific room on the day of interview.

A room may be allocated to several members of staff as required throughout a working day.

A client is interviewed only once on a given date, but may be requested to attend further interviews at later dates.

Boyce-Codd Normal Form

ClientInterview

clientNo	interviewDate	interviewTime	staffNo	roomNo
CR76	13-May-14	10.30	SG5	G101
CR56	13-May-14	12.00	SG5	G101
CR74	13-May-14	12.00	SG37	G102
CR56	1-Jul-14	10.30	SG5	G102

ClientInterview (clientNo, interviewDate. interviewTime, staffNo, roomNo)

fd1 clientNo, interviewDate -> interviewTime, staffNo, roomNo (Primary key)
fd2 staffNo, interviewDate, interviewTime -> clientNo (Candidate key)
fd3 roomNo, interviewDate, interviewTime -> staffNo, clientNo (Candidate key)
fd4 staffNo, interviewDate -> roomNo

BCNF

- This relation is not in BCNF (a stronger normal form of 3NF) due to the presence of the (staffNo, interviewDate) determinant, which is not a candidate key for the relation.
- BCNF requires that all determinants in a relation must be a candidate key for the relation.
- ClientInterview relation may suffer from update anomalies. For example, to change the room number for staff number SG5 on the 13-May-14 we must update two tuples. If only one tuple is updated with the new room number, this results in an inconsistent state for the database.
- To transform the ClientInterview relation to BCNF, we must remove the violating functional dependency by creating two new relations called Interview and StaffRoom

BCNF

Interview

clientNo	interviewDate	interviewTime	staffNo
CR76	13-May-14	10.30	SG5
CR56	13-May-14	12.00	SG5
CR74	13-May-14	12.00	SG37
CR56	1-Jul-14	10.30	SG5

StaffRoom

staffNo	interviewDate	roomNo
SG5	13-May-14	G101
SG37	13-May-14	G102
SG5	1-Jul-14	G102