OOAD: Normalization

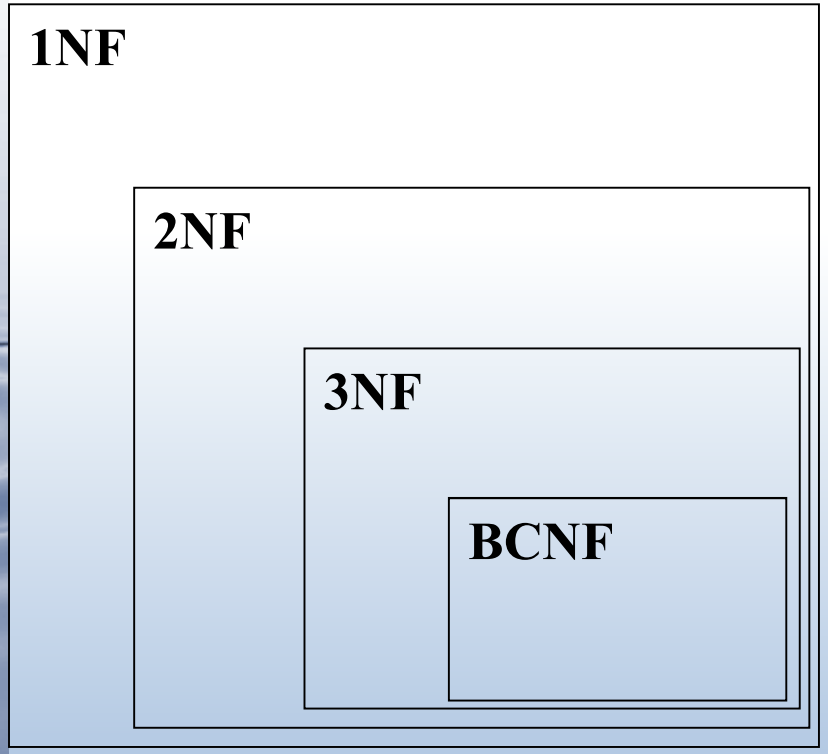
Presenter: Dr. Ha Viet Uyen Synh.

Normalization

Normalization is a process that “improves” a database design by generating relations that are of higher normal forms.

We discuss four normal forms: first, second, third, and Boyce-Codd normal forms _ 1NF, 2NF, 3NF, and BCNF

Normalization



a relation in BCNF, is also in 3NF

a relation in 3NF is also in 2NF

a relation in 2NF is also in 1NF

Normalization

We consider a relation in BCNF to be fully normalized.

The benefit of higher normal forms is that update semantics for the affected data are simplified.

This means that applications required to maintain the database are simpler.

A design that has a lower normal form than another design has more redundancy. Uncontrolled redundancy can lead to data integrity problems.

Functional Dependencies

We say an attribute, B, has a *functional dependency* on another attribute, A, if for any two records, which have the same value for A, then the values for B in these two records must be the same. We illustrate this as:

A → B

Example: Suppose we keep track of employee email addresses, and we only track one email address for each employee. Suppose each employee is identified by their unique employee number. We say there is a functional dependency of email address on employee number:

employee number → email address

Keys

Primary Key: a minimal set of attributes that form a candidate key

Any attribute or collection of attributes that functionally determine all attributes in a record is a Candidate Key.

A key consisting of more than one attribute is called a "composite key."

Foreign Key: A value in the "child" table that matches with the related value in the "parent" table.

Ex:

SalesRep(SalesRepNumber, Name)

Customer(CustomerNumber, SalesRepNumber)

Functional Dependencies

EmpNum	EmpEmail	EmpFname	EmpLname
123	jdoe@abc.com	John	Doe
456	psmith@abc.com	Peter	Smith
555	alee1@abc.com	Alan	Lee
633	pdoe@abc.com	Peter	Doe
787	alee2@abc.com	Alan	Lee

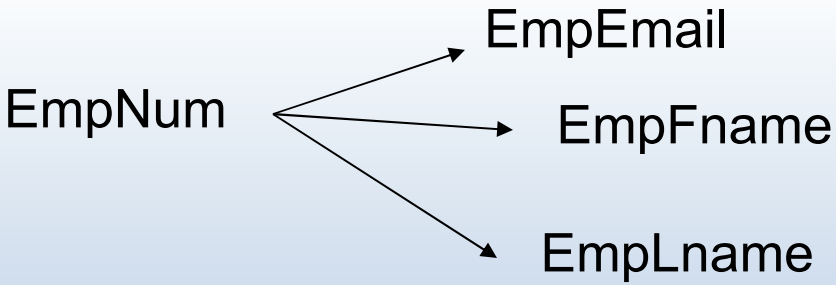
If EmpNum is the PK then the FDs:

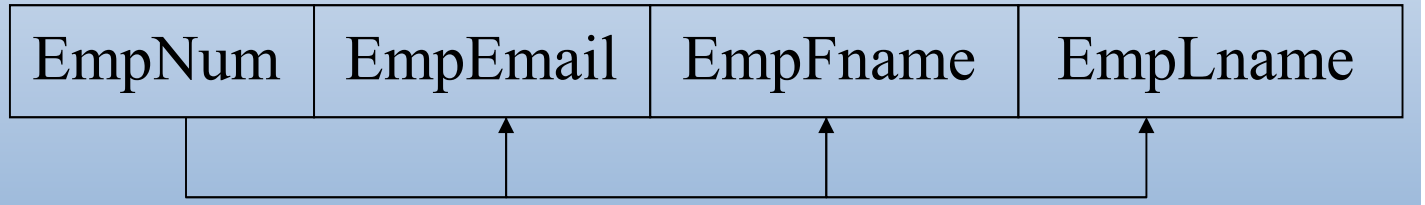
EmpNum → EmpEmail
EmpNum → EmpFname
EmpNum → EmpLname

must exist.

Functional Dependencies

EmpNum → EmpEmail
EmpNum → EmpFname
EmpNum → EmpLname





Determinant

Functional Dependency

EmpNum → EmpEmail

Attribute on the LHS is known as the *determinant*

- EmpNum is a determinant of EmpEmail

Transitive dependency

Consider attributes A, B, and C, and where

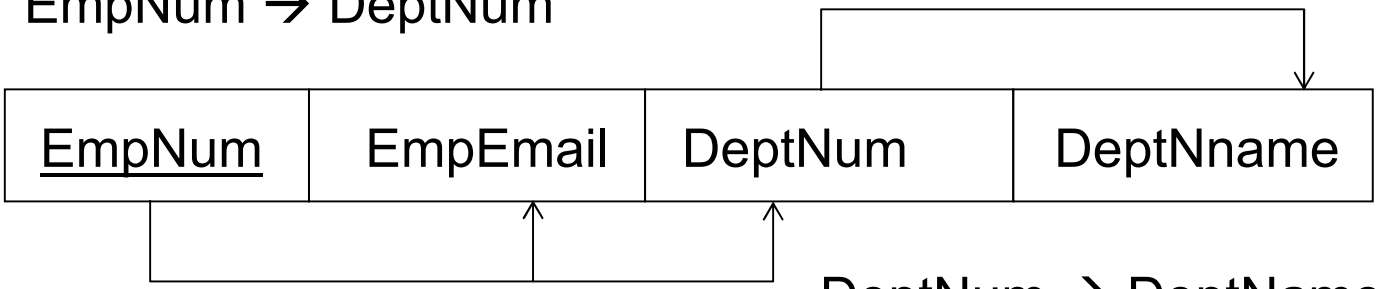
A → B and B → C.

Functional dependencies are transitive, which means that we also have the functional dependency A → C

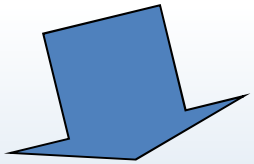
We say that C is transitively dependent on A through B.

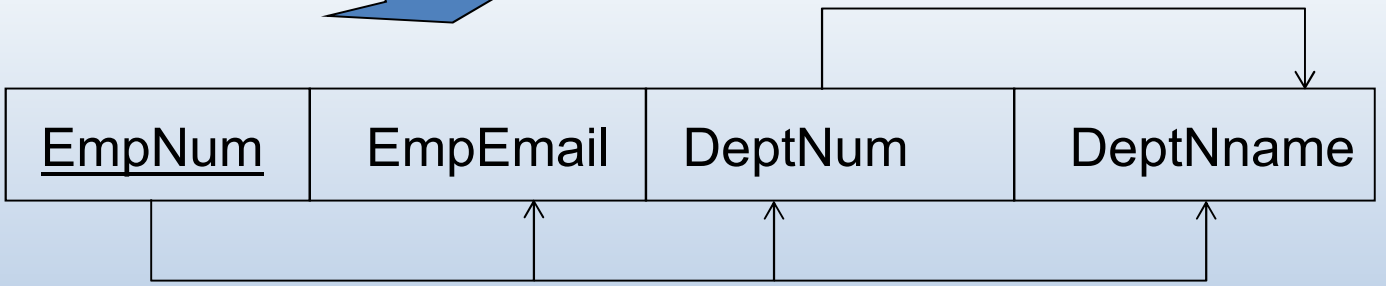
Transitive dependency

EmpNum → DeptNum



DeptNum → DeptName



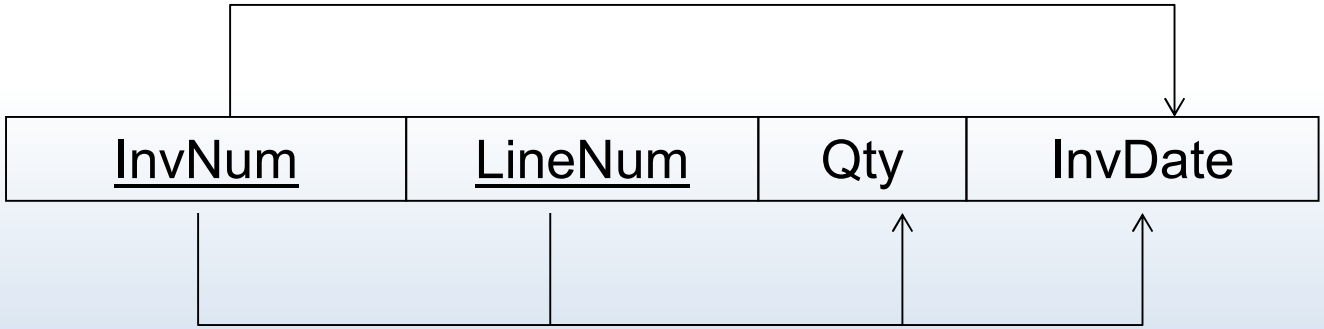


DeptName is *transitively dependent* on EmpNum via DeptNum

EmpNum → DeptName

Partial dependency

A partial dependency exists when an attribute B is functionally dependent on an attribute A, and A is a component of a multipart candidate key.



Candidate keys: {InvNum, LineNum} .
InvDate is *partially dependent* on {InvNum, LineNum} as InvNum is a determinant of InvDate and InvNum is part of a candidate key

Normal Forms

All attributes depend on the key, the whole key and nothing but the key.

1NF	Keys and no repeating groups
2NF	No partial dependencies
3NF	All determinants are candidate keys
BCNF	No multivalued dependencies

First Normal Form

We say a relation is in 1NF if all values stored in the relation are single-valued and atomic.

1NF places restrictions on the structure of relations.

Values must be simple.

- Table has a primary key
- Table has no repeating groups

A multivalued attribute is an attribute that may have several values for one record

A repeating group is a set of one or more multivalued attributes that are related

Example

Multivalued attribute:

Orders(OrderNumber, OrderDate, {PartNumber})

[12491 | 9/02/2001 | BT04, BZ66]

Repeating group:

Orders(OrderNumber, OrderDate, {PartNumber, NumberOrdered})

[12491 | 9/02/2001 | (BT04, 1), (BZ66, 1)]

First Normal Form

The following in not in 1NF

EmpNum	EmpPhone	EmpDegrees
123	233-9876	
333	233-1231	BA, BSc, PhD
679	233-1231	BSc, MSc

EmpDegrees is a multi-valued field:

employee 679 has two degrees: *BSc* and *MSc*

employee 333 has three degrees: *BA, BSc, PhD*

First Normal Form

EmpNum	EmpPhone	EmpDegrees
123	233-9876	
333	233-1231	BA, BSc, PhD
679	233-1231	BSc, MSc

To obtain 1NF relations we must, without loss of information, replace the above with two relations

Employee

EmpNum	EmpPhone
123	233-9876
333	233-1231
679	233-1231

EmployeeDegree

EmpNum	EmpDegree
333	BA
333	BSc
333	PhD
679	BSc
679	MSc

Second Normal Form

A relation is in 2NF if it is in 1NF, and every non-key attribute is fully dependent on each candidate key.

- 2NF (and 3NF) both involve the concepts of key and non-key attributes.
- A *key attribute* is any attribute that is part of a key; any attribute that is not a key attribute, is a *non-key attribute*.
- Relations that are not in BCNF have data redundancies
- A relation in 2NF will *not have any partial dependencies*

Second Normal Form

Consider this InvLine table (in 1NF):

InvNum	LineNum	ProdNum	Qty	InvDate

InvNum, LineNum → ProdNum, Qty

InvNum → InvDate

There are two candidate keys.

Qty is the only non-key attribute, and it is dependent on InvNum

Since there is a determinant that is not a candidate key, InvLine is not BCNF

InvLine is not 2NF since there is a partial dependency of InvDate on InvNum

=> InvLine is only in 1NF

Second Normal Form

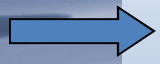
InvLine

InvNum	LineNum	ProdNum	Qty	InvDate

The above relation has redundancies: the invoice date is repeated on each invoice line.

We can *improve* the database by decomposing the relation into two relations:

InvNum	LineNum	ProdNum	Qty

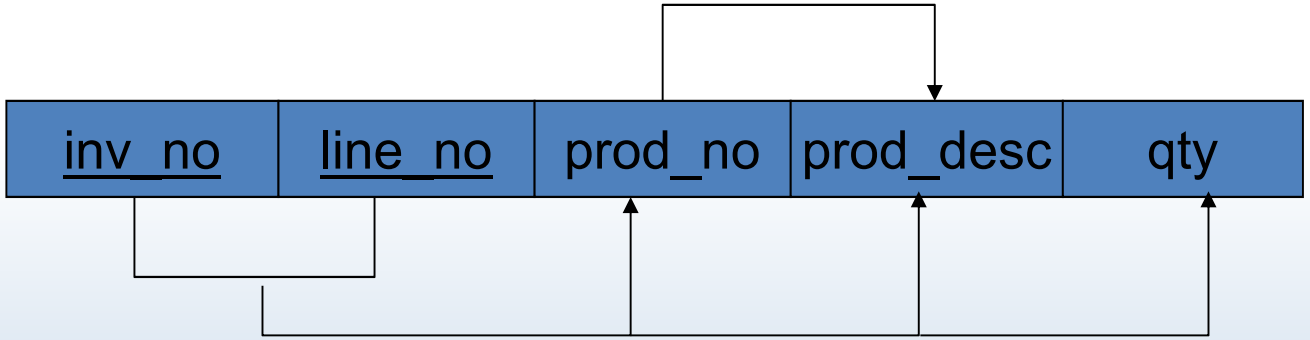


InvNum	InvDate

Question: What is the highest normal form for these relations? 2NF? 3NF? BCNF?

Exercise #1

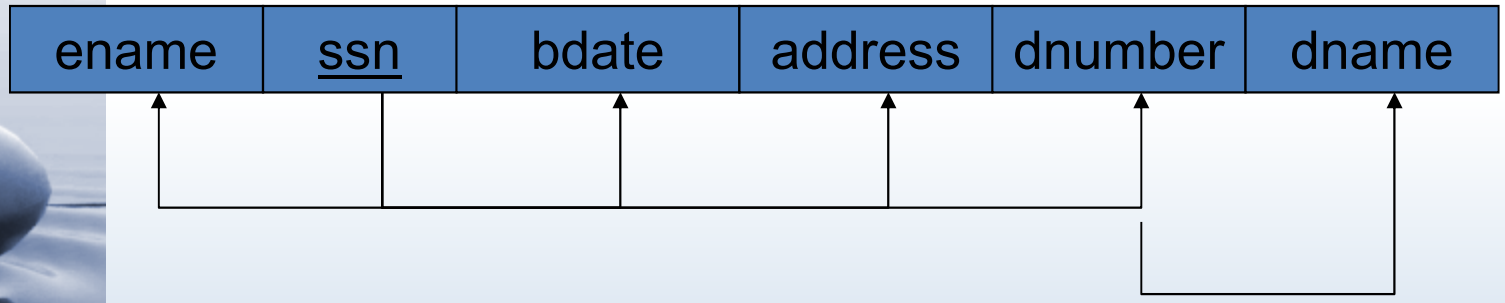
Is the following relation in 2NF?



Exercise #2

What is the highest normal form for these relations?

EmployeeDept



Third Normal Form

- A relation is in 3NF if the relation is in 1NF and all determinants of *non-key* attributes are candidate keys. That is, for any functional dependency:

  $X \rightarrow Y$,

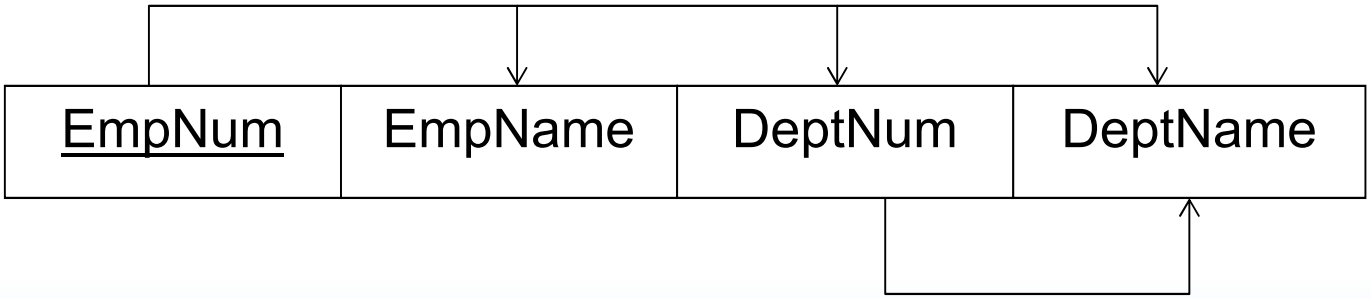
  where Y is a non-key attribute (or a set of non-key attributes), X is a candidate key.

- This definition of 3NF differs from BCNF only in the specification of non-key attributes - 3NF is weaker than BCNF. (BCNF requires all determinants to be candidate keys.)

- A relation in 3NF will not have any transitive dependencies of non-key attribute on a candidate key through another non-key attribute.

Third Normal Form

Consider this Employee relation



EmpName, DeptNum, and DeptName are non-key attributes.

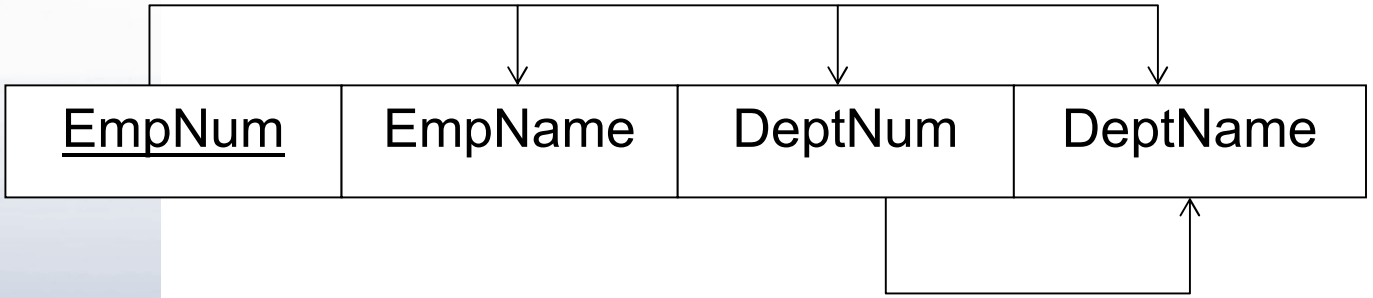
DeptNum determines DeptName, a non-key attribute, and DeptNum is not a candidate key.

Is the relation in BCNF? …

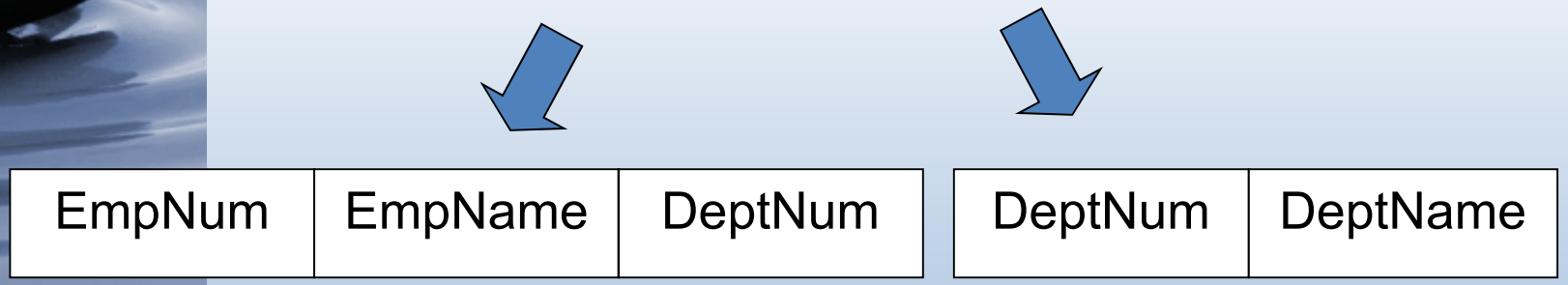
Is the relation in 3NF? …

Is the relation in 2NF? …

Third Normal Form

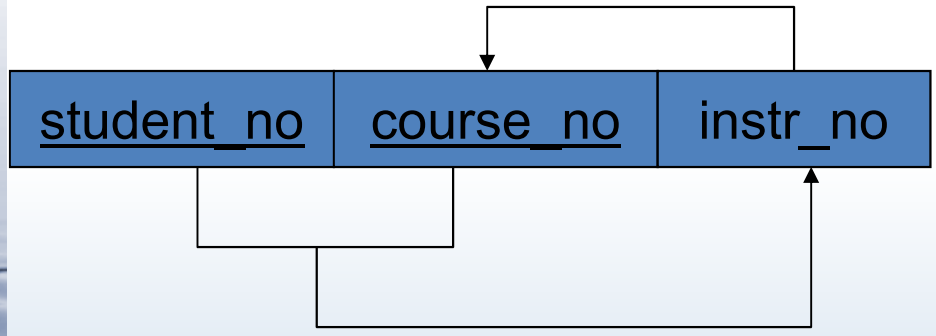


We correct the situation by decomposing the original relation into two 3NF relations. Note the decomposition is *lossless*.



Verify these two relations are in 3NF.

In 3NF, but not in BCNF:

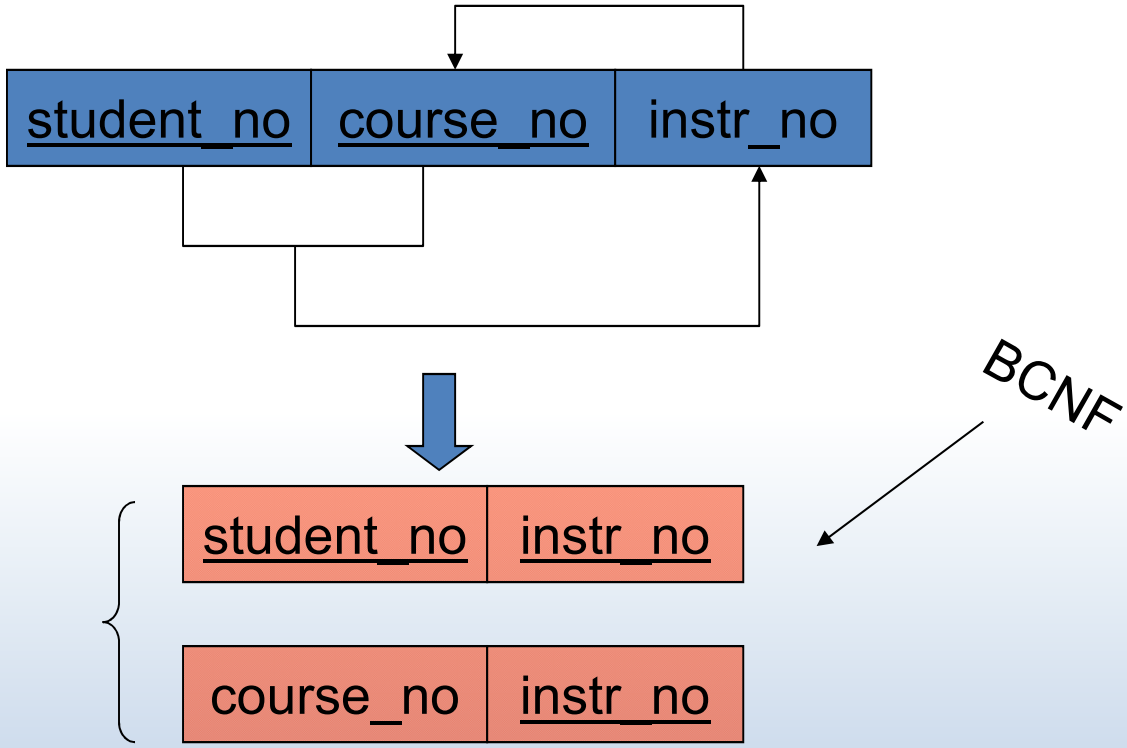


Instructor teaches one course only.

Student takes a course and has one instructor.

{student_no, course_no} → instr_no
instr_no → course_no

since we have instr_no → course-no, but instr_no is not a Candidate key.



{student_no, instr_no} → student_no
{student_no, instr_no} → instr_no
instr_no → course_no

Boyce-Codd Normal Form

Boyce-Codd Normal Form

BCNF is defined very simply:

> a relation is in BCNF if only if every determinant is a candidate key.

If our database will be used for OLTP (on line transaction processing), then BCNF is our target. Usually, we meet this objective. However, we might denormalize (3NF, 2NF, or 1NF) for performance reasons.

Exercise #3

Order(OrderNumber, OrderDate, {PartNumber, {Supplier}})

Exercise #4

(supplier_no, status, city, part_no, quantity)

Functional Dependencies:

(supplier_no, part_no) → quantity

(supplier_no) → status

(supplier_no) → city

city → status (Supplier's status is determined by location)

Exercise #5

SUPPLIER_PART (supplier_no, supplier_name, part_no, quantity)

Functional Dependencies:

We assume that supplier_name's are always unique to each supplier. Thus we have two candidate keys:

(supplier_no, part_no) and (supplier_name, part_no)

Thus we have the following dependencies:

(supplier_no, part_no) → quantity

(supplier_no, part_no) → supplier_name

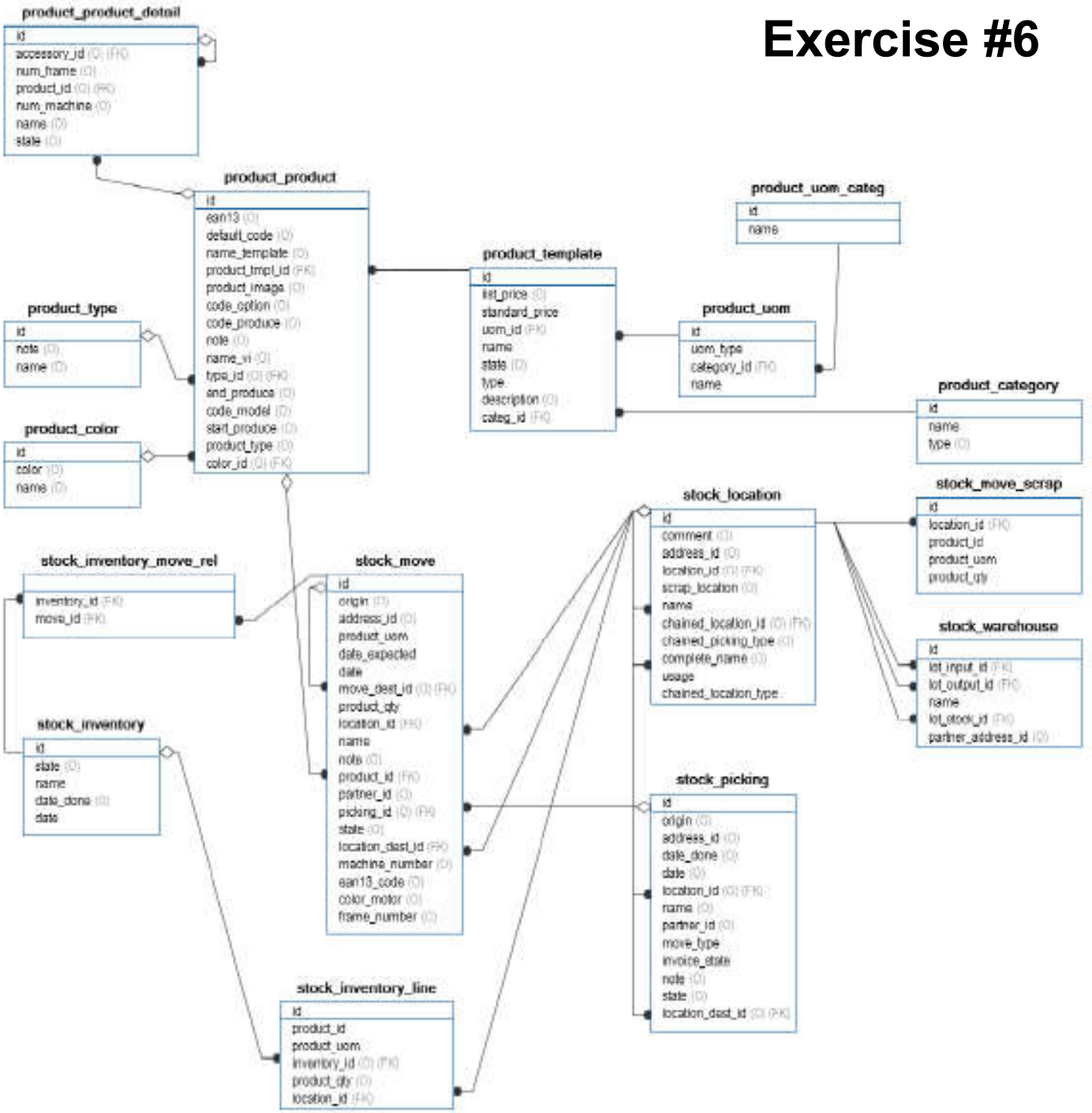
(supplier_name, part_no) → quantity

(supplier_name, part_no) → supplier_no

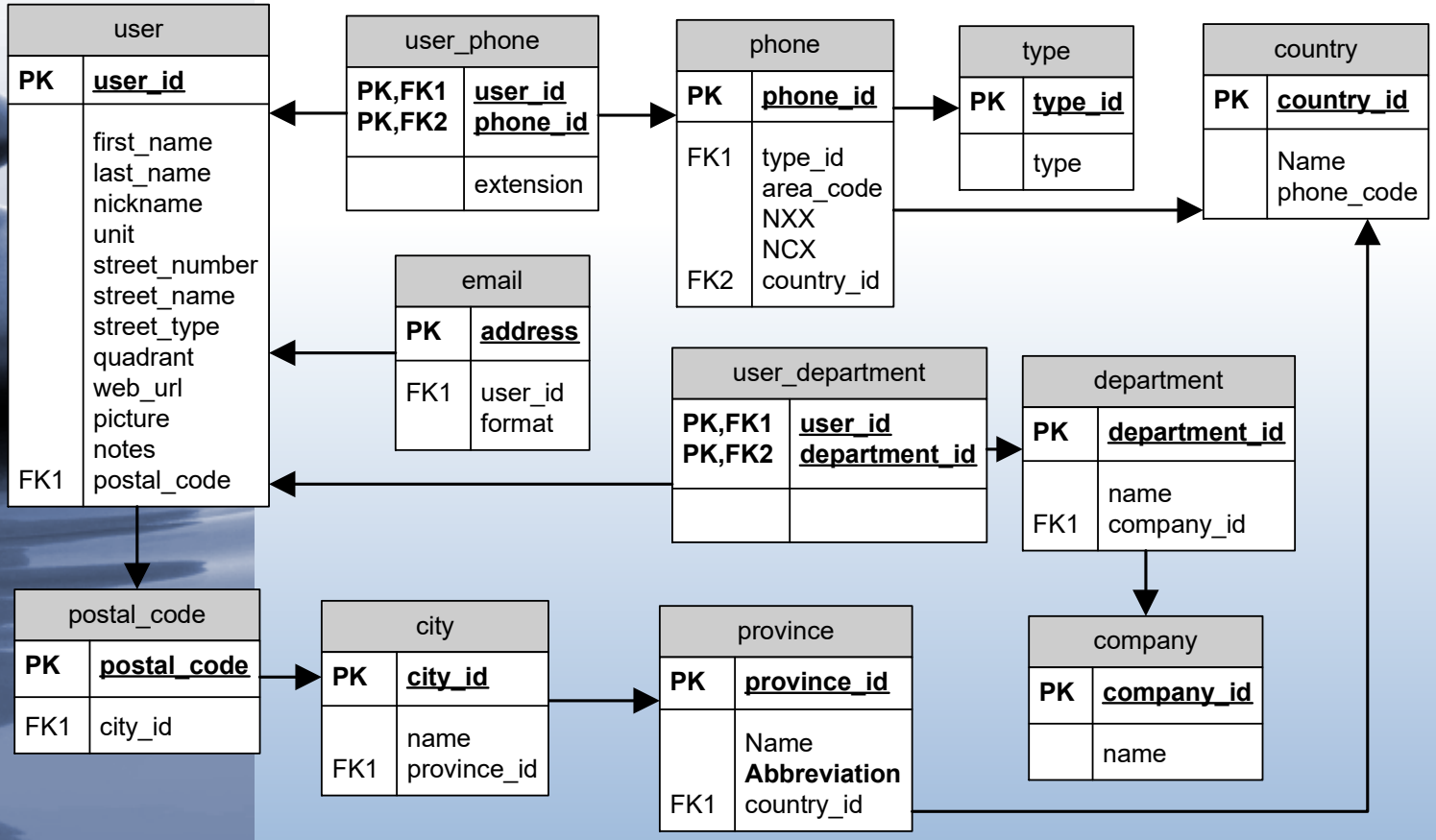
supplier_name → supplier_no

supplier_no → supplier_name

Exercise #6


product_product_detail
id
accessory_id (O) (FK)
num_frame (O)
product_id (O) (FK)
num_machine (O)
name (O)
state (O)
product_product
id
ean13 (O)
default_code (O)
name_template (O)
product_tmpl_id (FK)
product_image (O)
code_option (O)
code_produce (O)
note (O)
name_vi (O)
type_id (O) (FK)
and_produce (O)
code_model (O)
start_produce (O)
product_type (O)
color_id (O) (FK)
product_template
id
list_price (O)
standard_price
uom_id (FK)
name
state (O)
type
description (O)
categ_id (FK)
product_uom_categ
id
name
product_uom
id
uom_type
category_id (FK)
name
product_type
id
note (O)
name (O)
product_color
id
color (O)
name (O)
product_category
id
name
type (O)
stock_location
id
comment (O)
address_id (O)
location_id (O) (FK)
scrap_location (O)
name
chained_location_id (O) (FK)
chained_picking_type (O)
complete_name (O)
usage
chained_location_type
stock_move_scrap
id
location_id (FK)
product_id
product_uom
product_qty
stock_warehouse
id
lot_input_id (FK)
lot_output_id (FK)
name
lot_stock_id (FK)
partner_address_id (O)
stock_inventory_move_rel
inventory_id (FK)
move_id (FK)
stock_move
id
origin (O)
address_id (O)
product_uom
date_expected
date
move_dest_id (O) (FK)
product_qty
location_id (FK)
name
note (O)
product_id (FK)
partner_id (O)
picking_id (O) (FK)
state (O)
location_dest_id (FK)
machine_number (O)
ean13_code (O)
color_motor (O)
frame_number (O)
stock_inventory
id
state (O)
name
date_done (O)
date
stock_picking
id
origin (O)
address_id (O)
date_done (O)
date (O)
location_id (O) (FK)
name (O)
partner_id (O)
move_type
invoice_state
note (O)
state (O)
location_dest_id (O) (FK)
stock_inventory_line
id
product_id
product_uom
inventory_id (O) (FK)
product_qty (O)
location_id (FK)

Exercise #7


user
PK user_id
first_name
last_name
nickname
unit
street_number
street_name
street_type
quadrant
web_url
picture
notes
FK1 postal_code
user_phone
PK,FK1 user_id
PK,FK2 phone_id
extension
phone
PK phone_id
FK1 type_id
area_code
NXX
NCX
FK2 country_id
type
PK type_id
type
country
PK country_id
Name
phone_code
email
PK address
FK1 user_id
format
user_department
PK,FK1 user_id
PK,FK2 department_id
department
PK department_id
name
FK1 company_id
postal_code
PK postal_code
FK1 city_id
city
PK city_id
name
FK1 province_id
province
PK province_id
Name
Abbreviation
FK1 country_id
company
PK company_id
name