

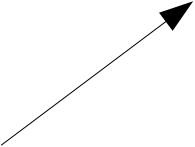
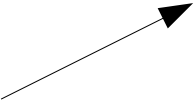
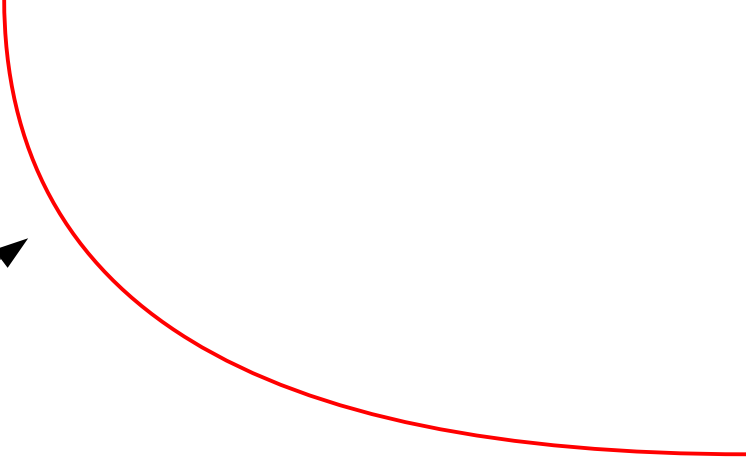
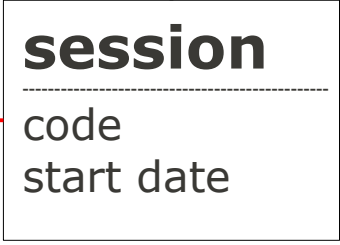
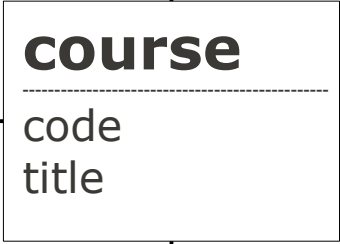
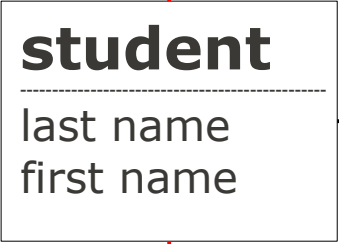
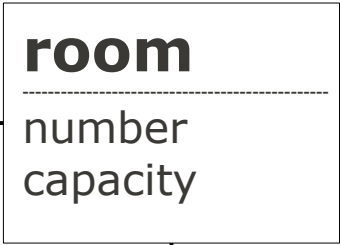
# INF1343, Week 5

## Translating ER into Relations; Normalization

Yuri Takhteyev  
University of Toronto  
January 31, 2011

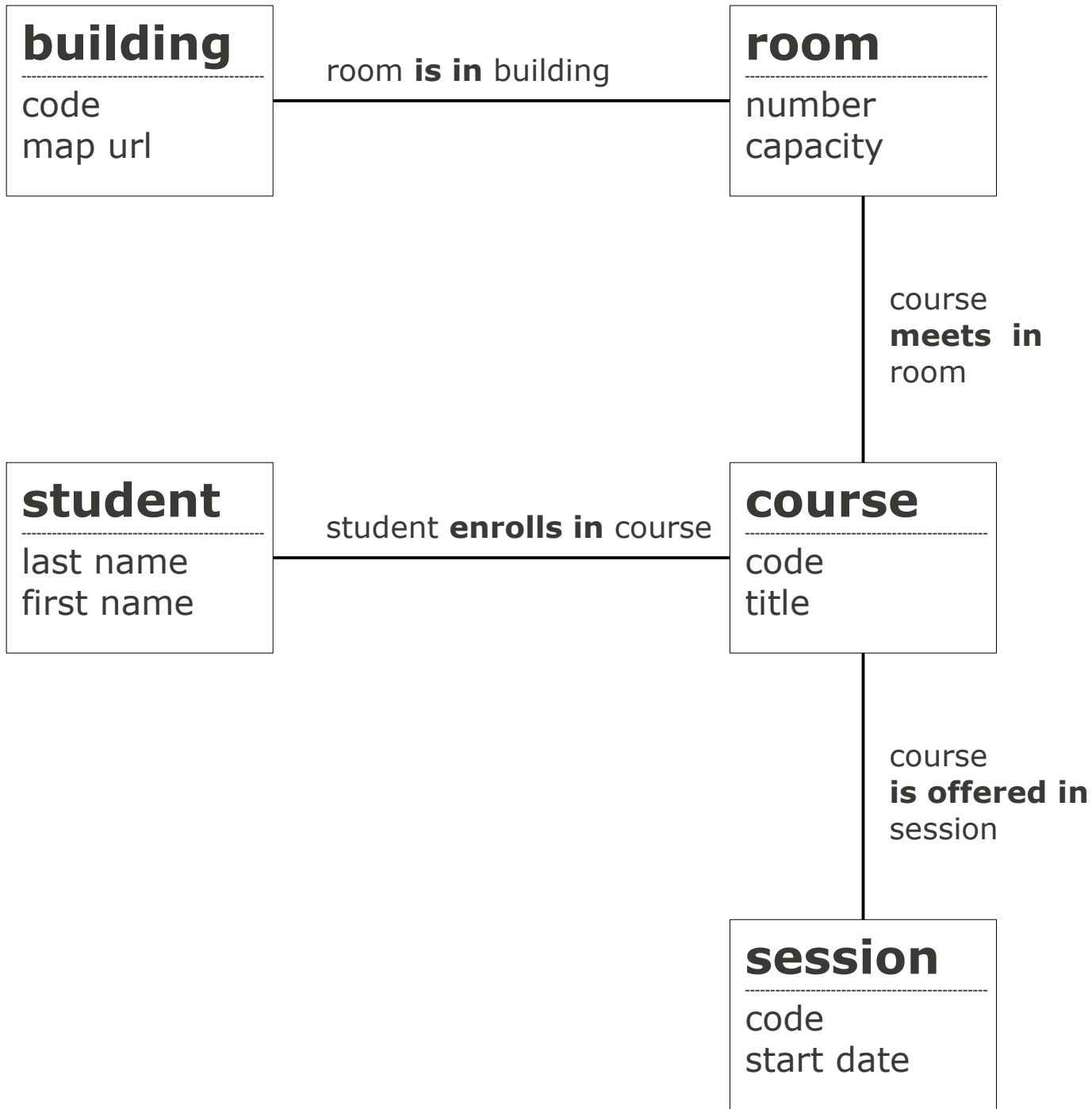


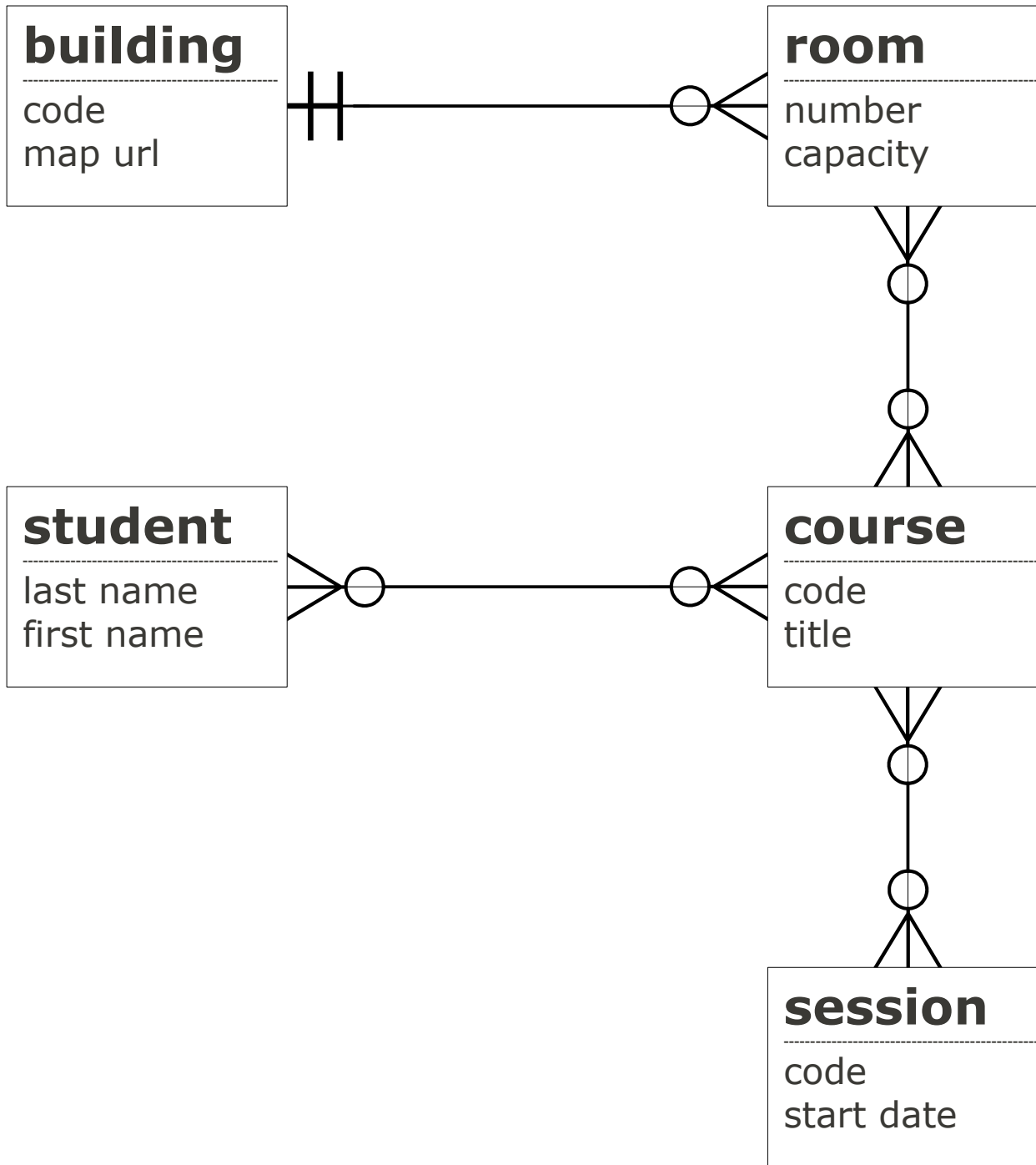
This presentation is licensed under Creative Commons Attribution License, v. 3.0. To view a copy of this license, visit <http://creativecommons.org/licenses/by/3.0/>. This presentation incorporates images from the Crystal Clear icon collection by Everaldo Coelho, available under LGPL from <http://everaldo.com/crystal/>.

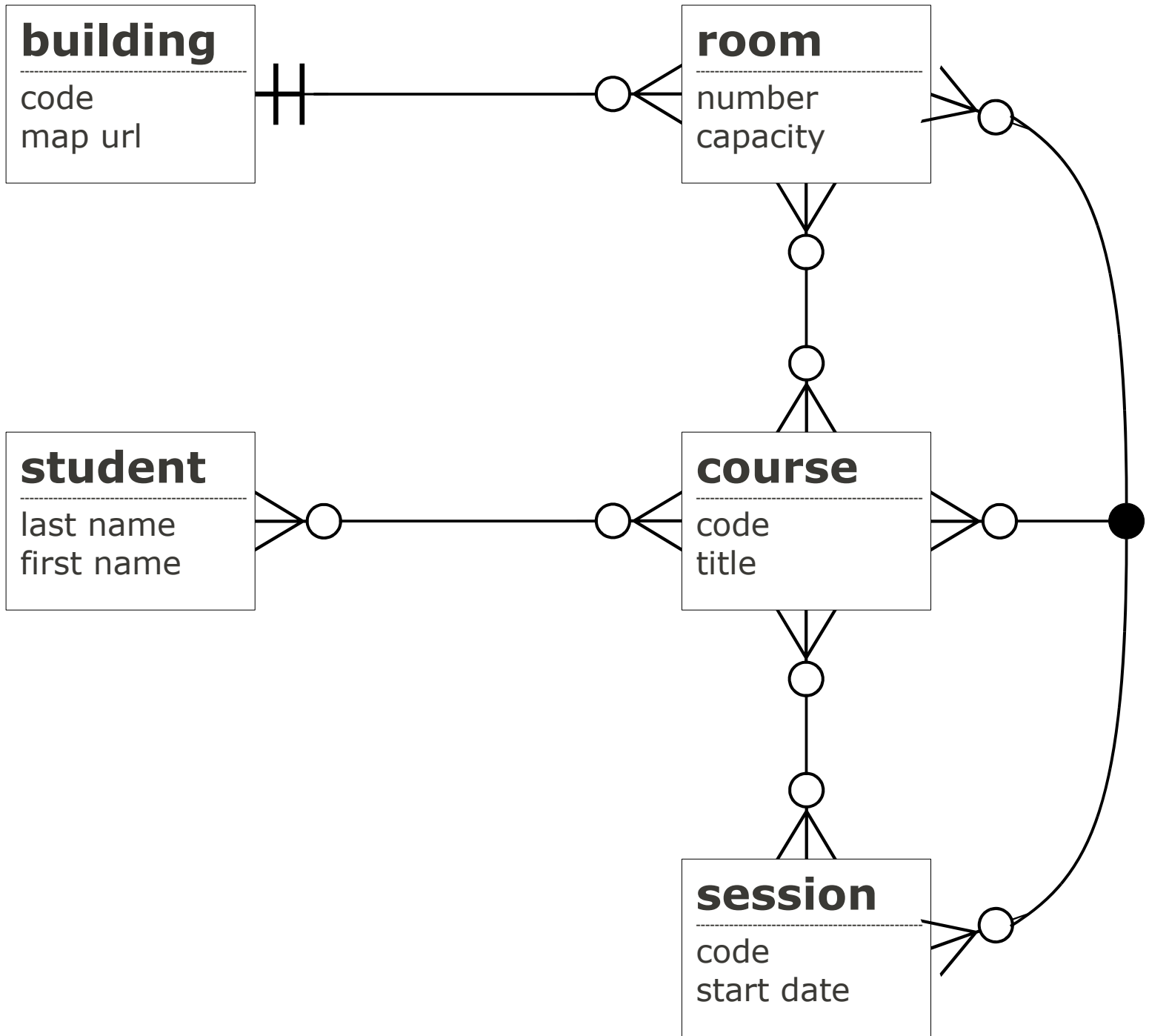


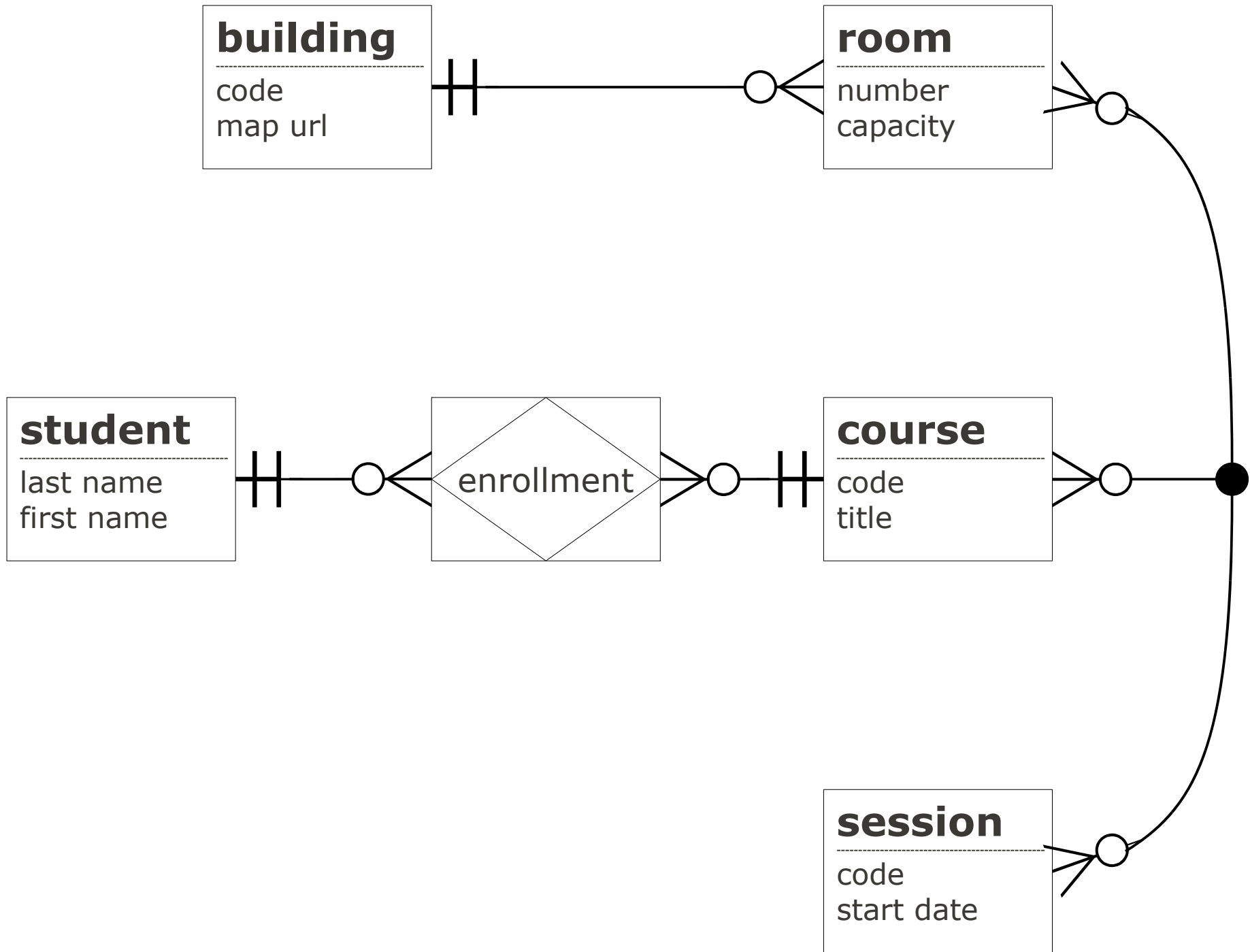
probably  
redundant

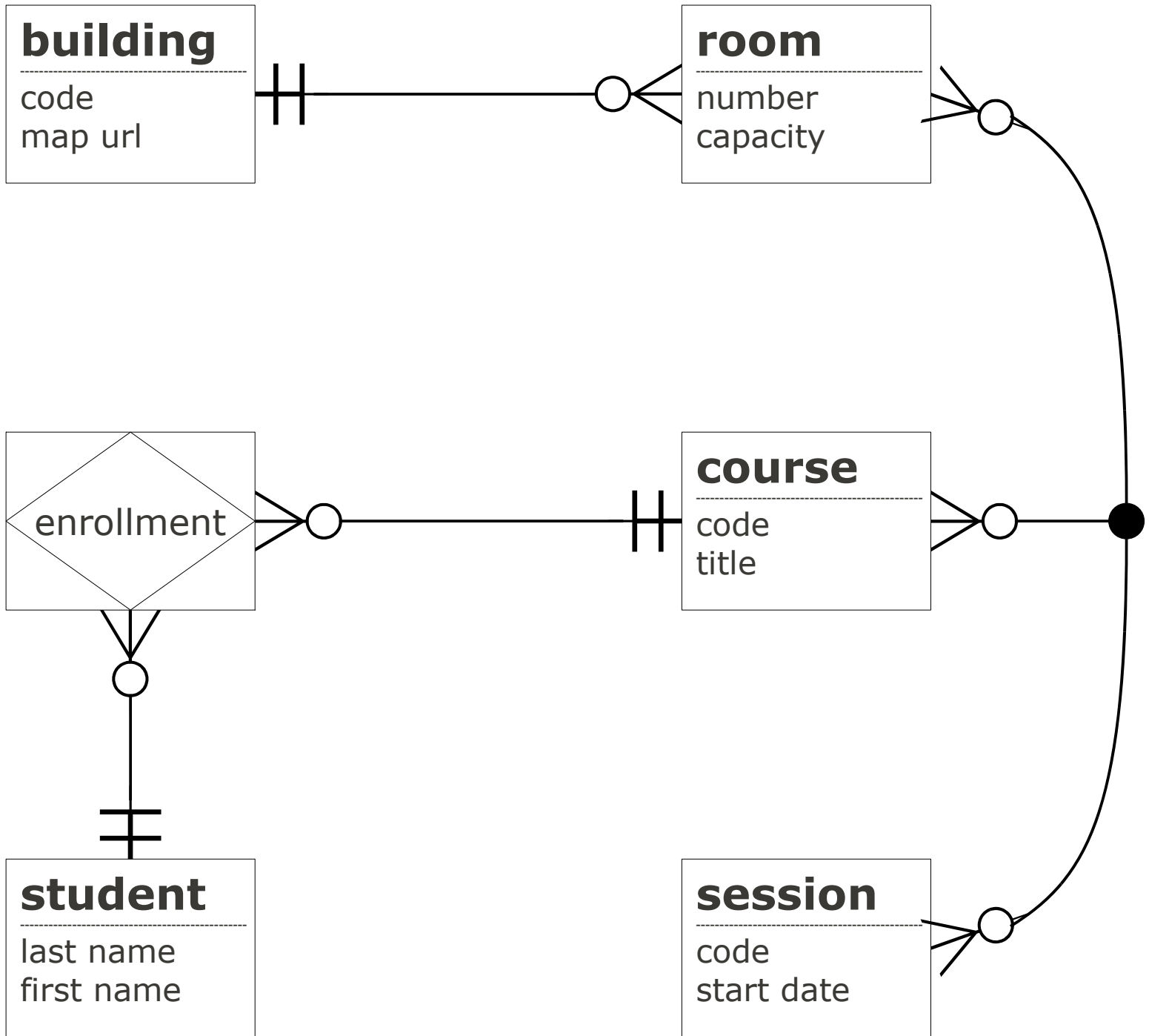
might be  
redundant

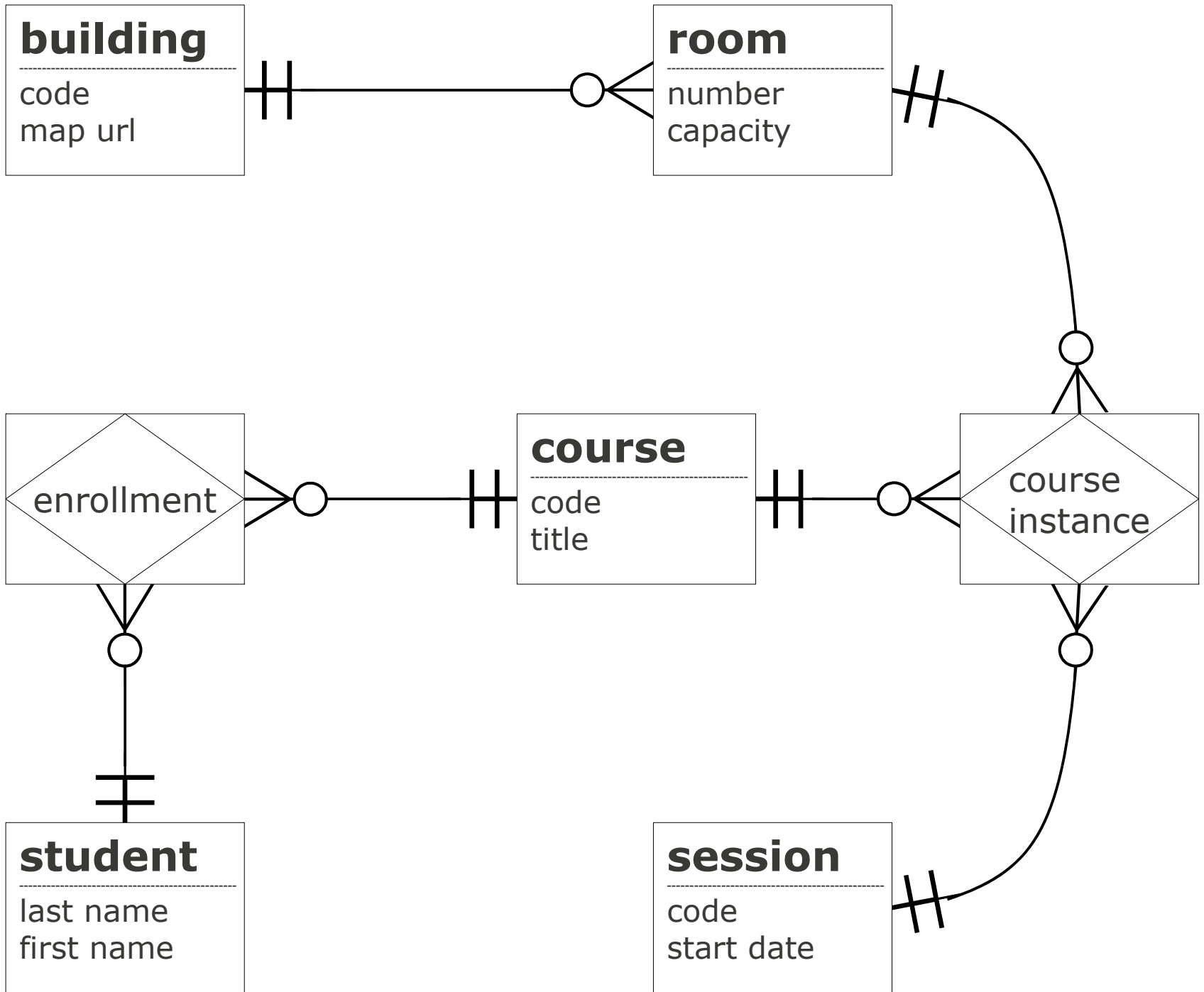




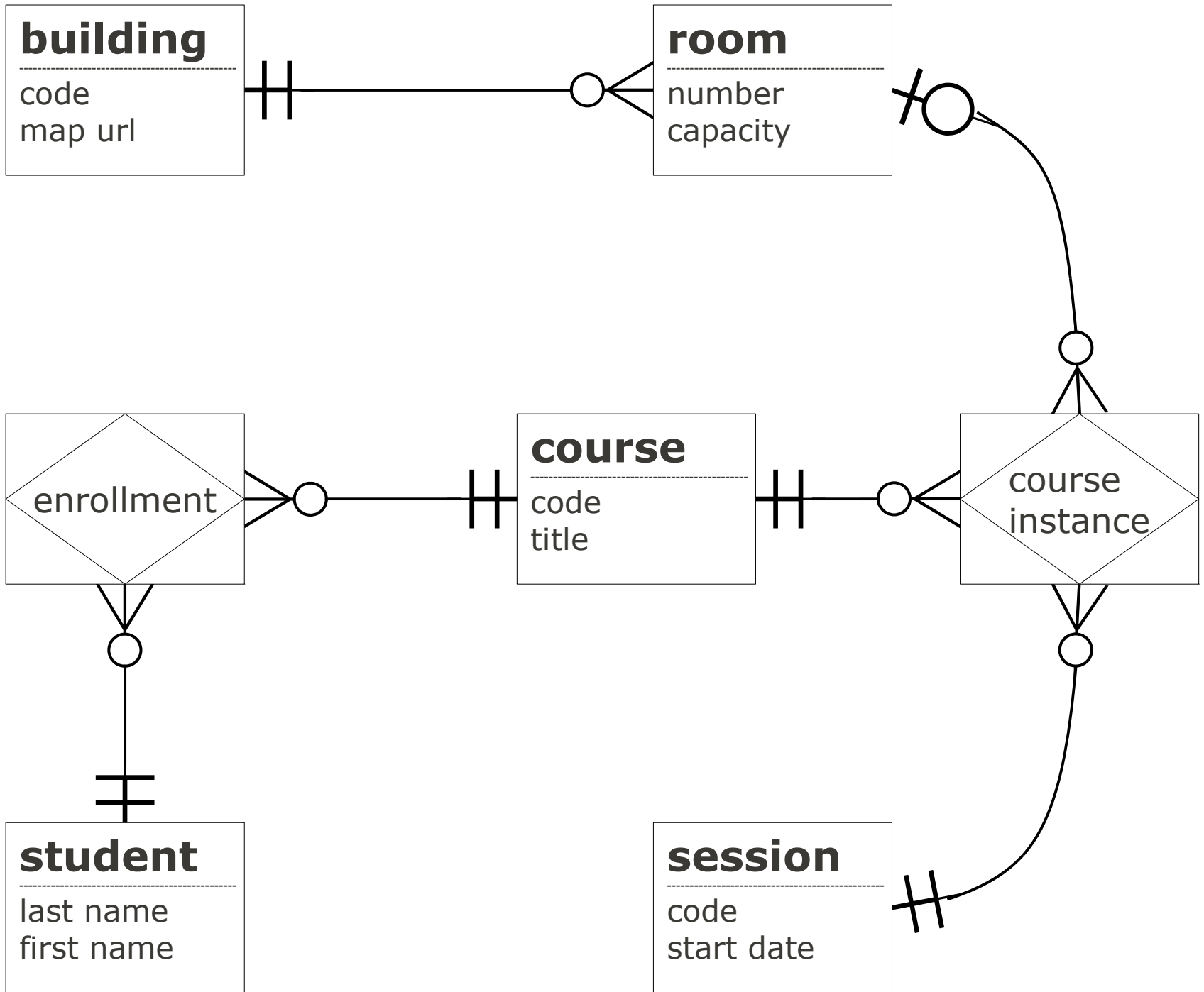


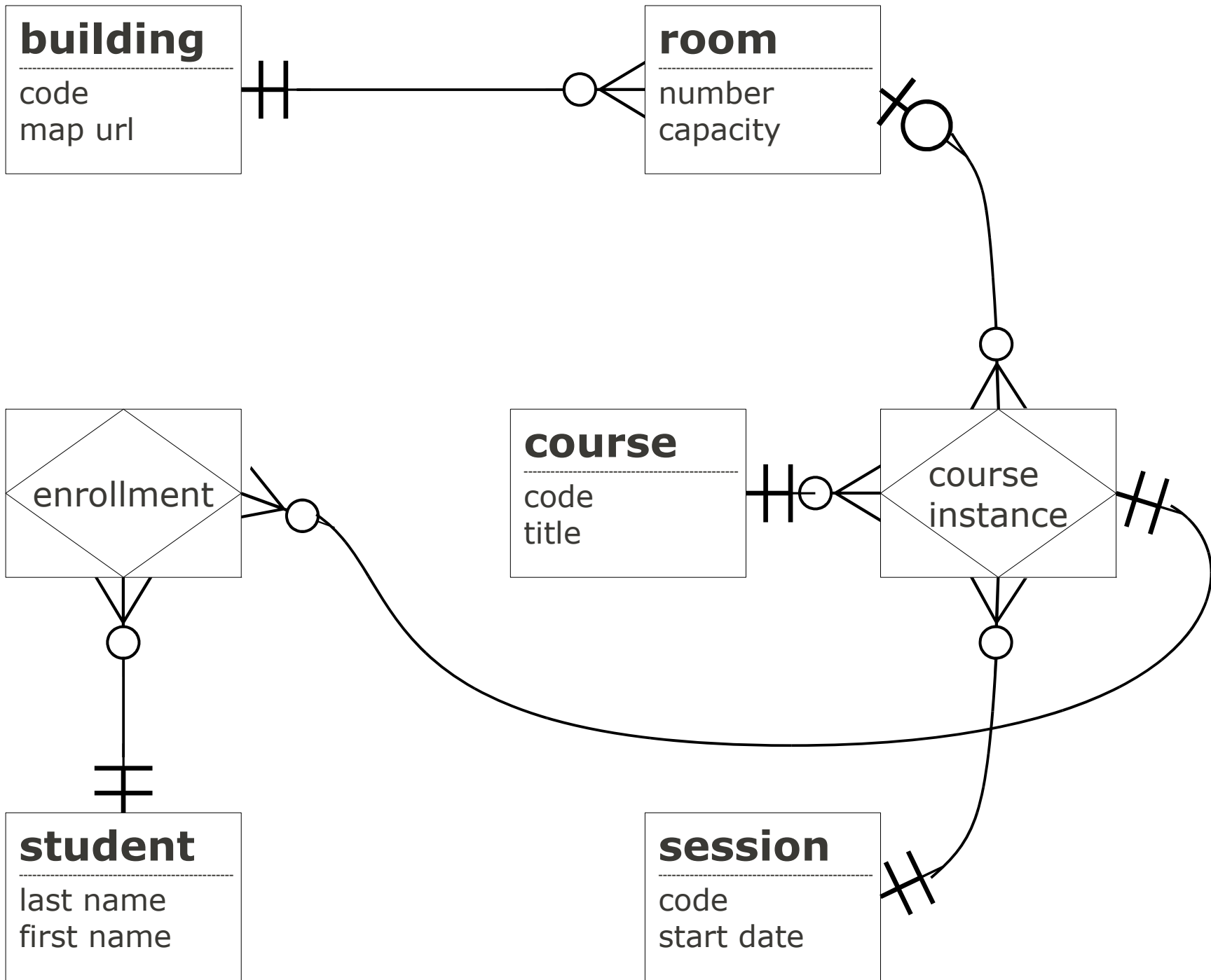


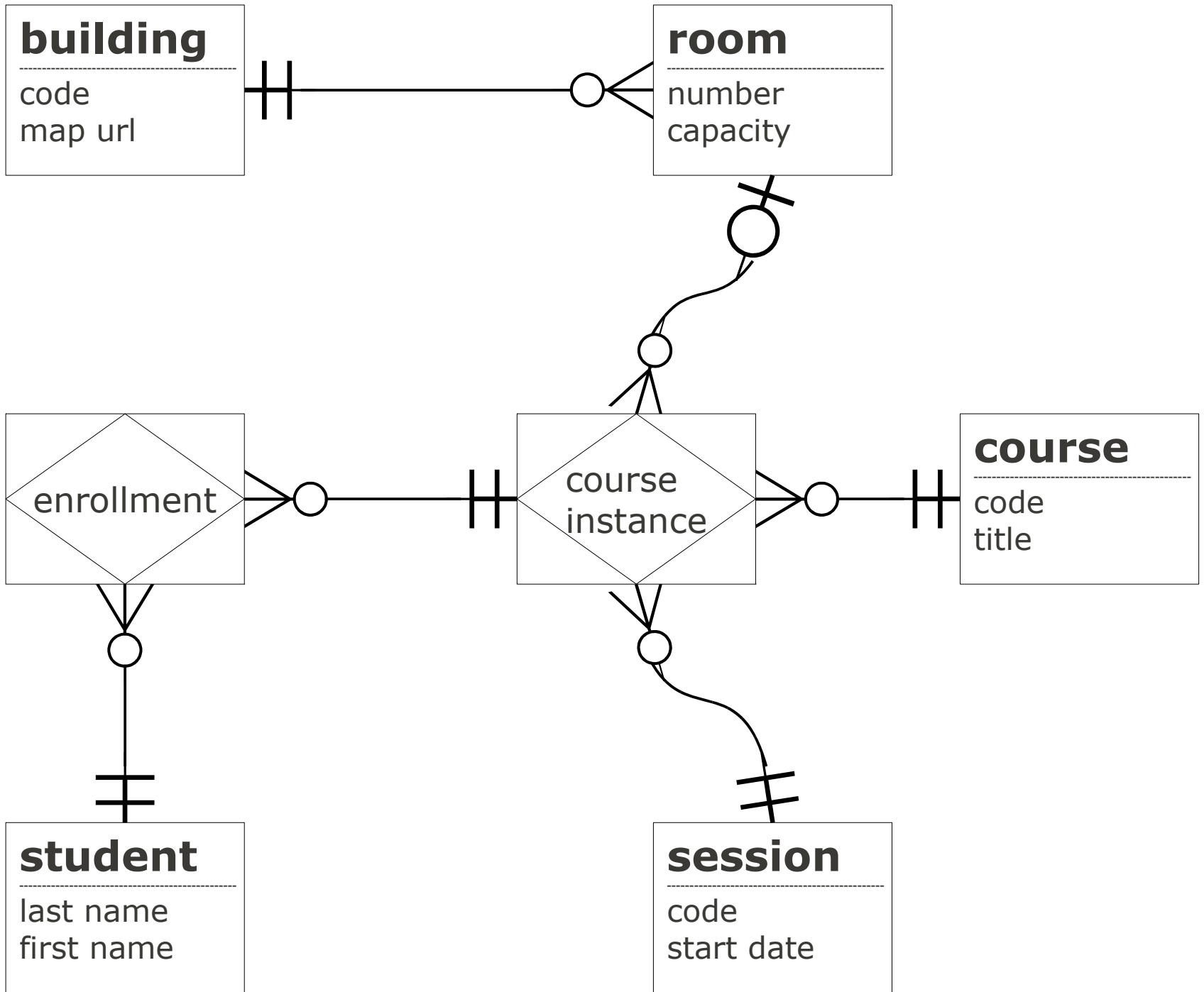


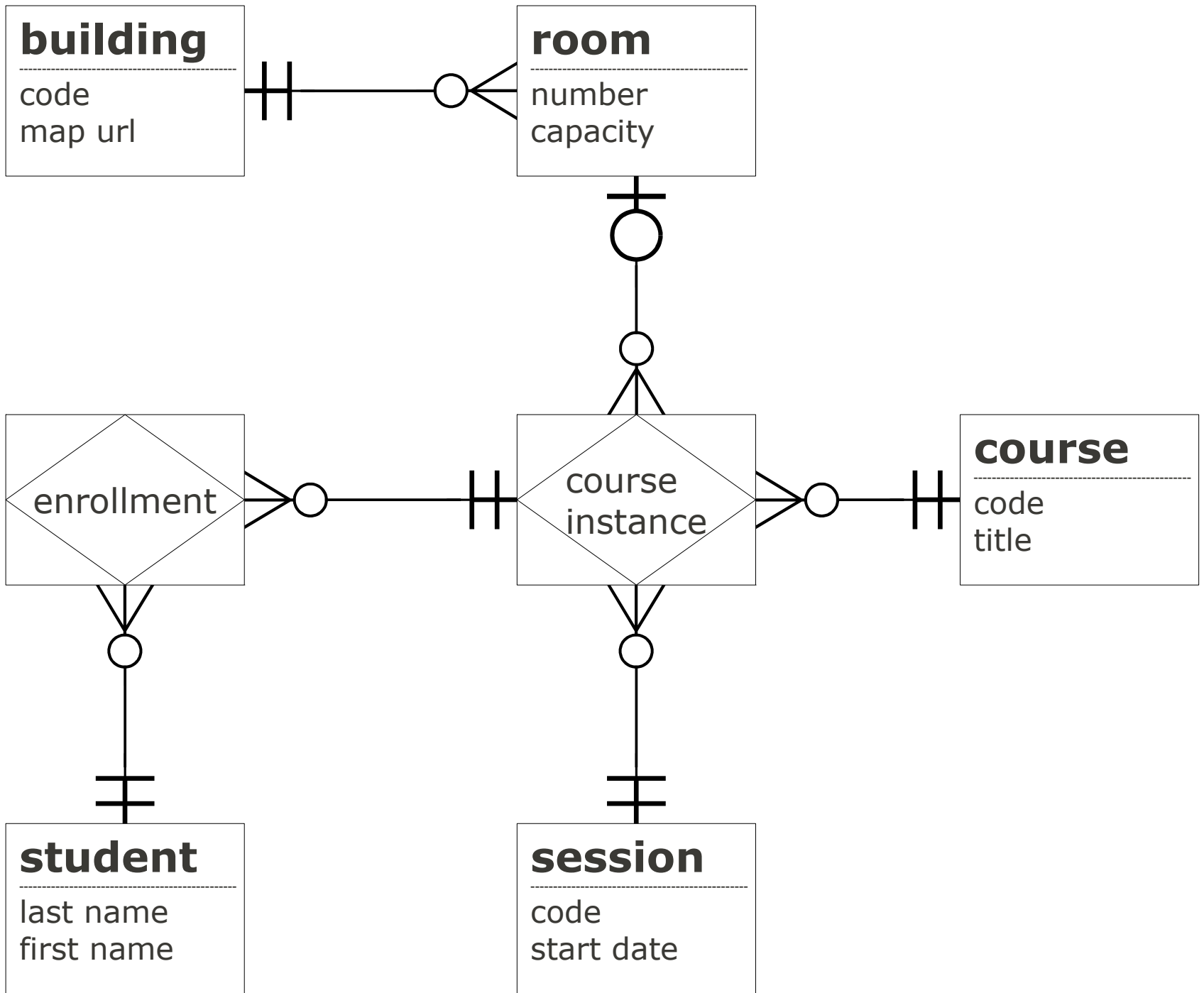


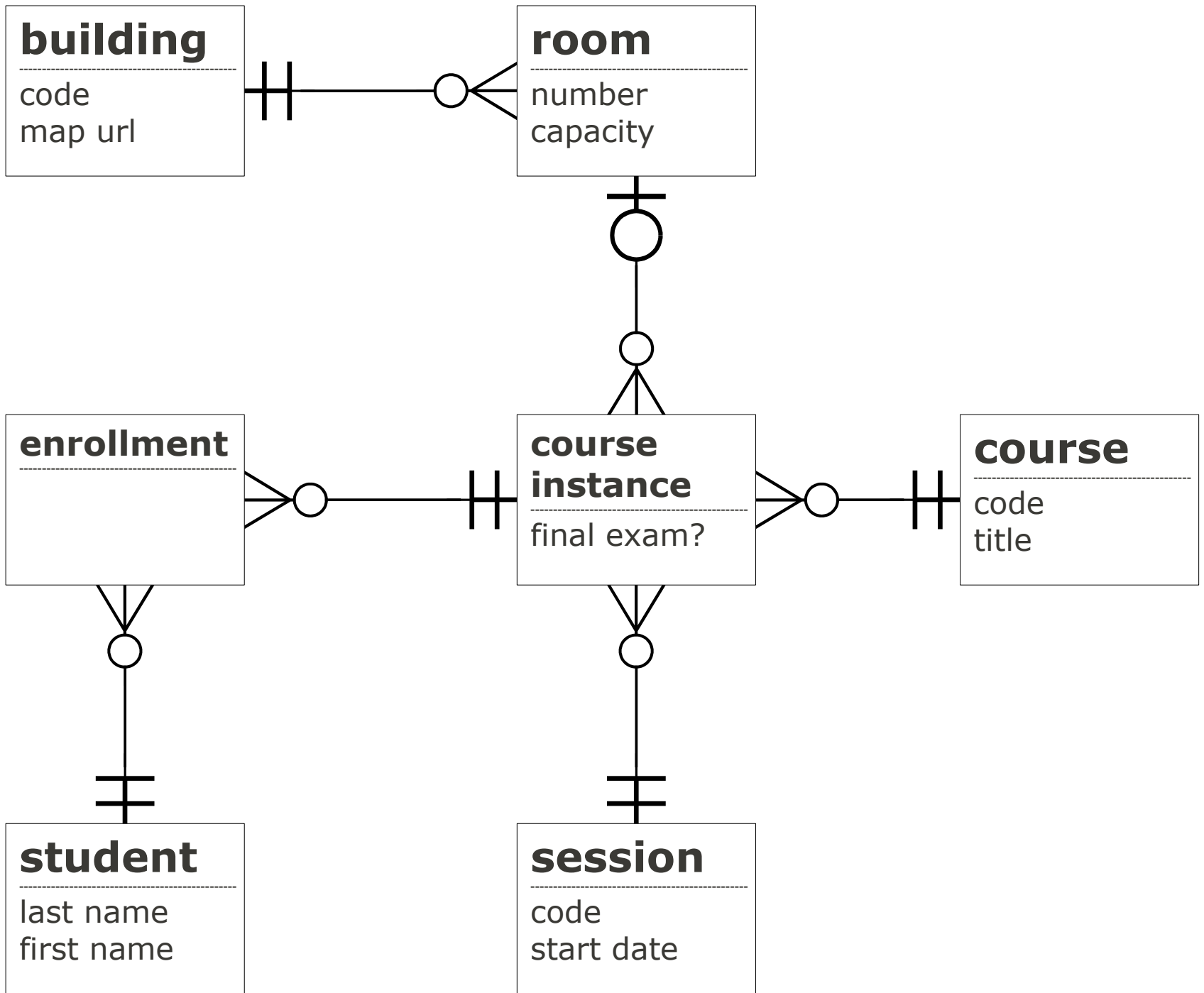


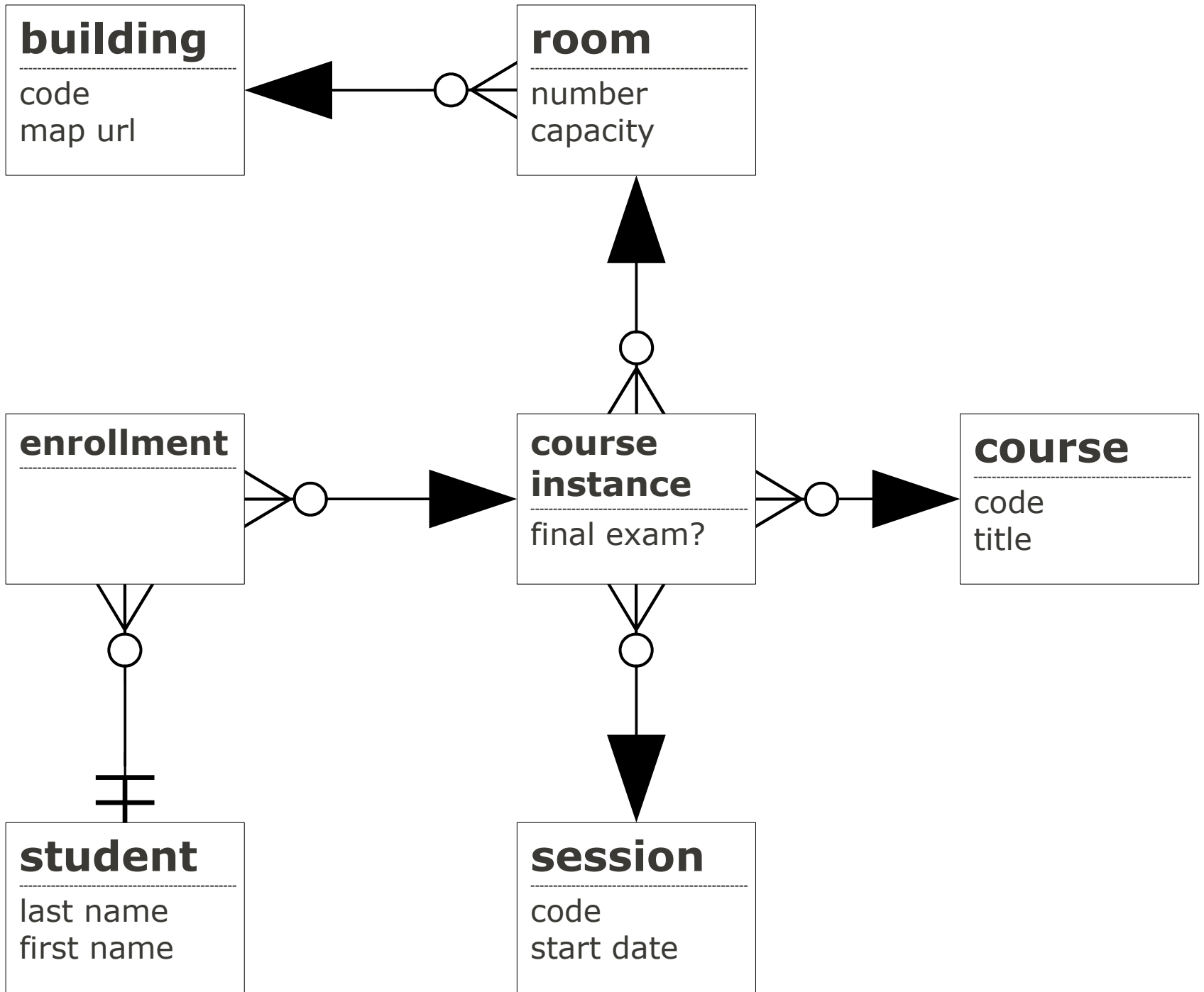












ER for M

# ER to Relations

**Entities:**

→ tables

**1:M relationships:**

→ primary and foreign keys



# Step 1

“Create one table for each entity”  
(after breaking up M:M)



**student**

first_name	last_name



**student**

name	domain
last_name	
first_name	



**student**

name	domain
last_name	varchar(100)
first_name	varchar(100)

## **student**

last name  
first name



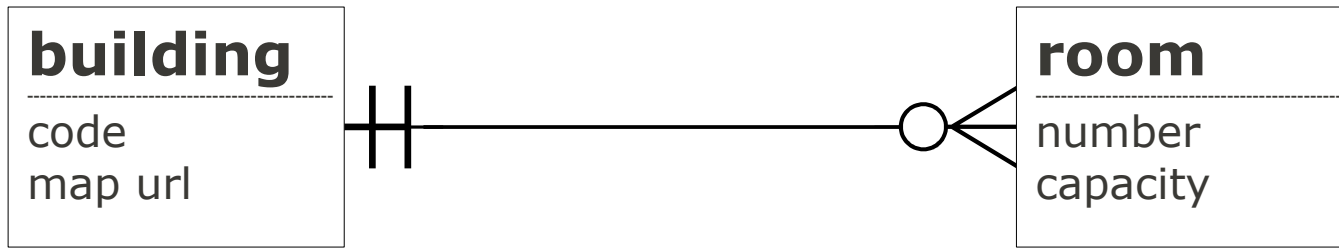
## **student**

<b>name</b>	<b>domain</b>	<b>NULL</b>
last_name	varchar(100)	YES
first_name	varchar(100)	NO

(do this for every entity)

# Steps 2-4

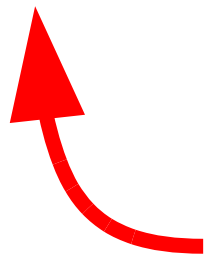
Creating primary and foreign keys  
to represent relationships



## building

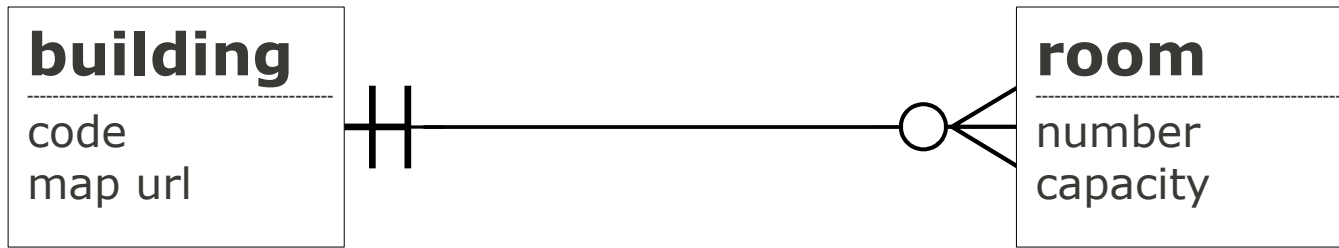
name	domain	NULL
code	varchar(100)	NO
map_url	varchar(100)	YES

## room



name	domain	NULL
number	integer	NO
capacity	integer	YES

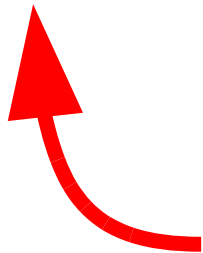




## building

name	domain	NULL
code	varchar(100)	NO
map_url	varchar(100)	YES

## room



name	domain	NULL
number	integer	NO
capacity	integer	YES
building_???		

# Keys

## **student**

name
last_name
first_name

no key!

# Keys

## **student**

name
last_name
first_name
student_no

← a candidate key

# Keys

## student

name
last_name
first_name
student_no
utorid

another  
candidate key



# Keys

## student

name
last_name
first_name
student_no
utorid



candidate #1



candidate #2

# Keys

## student

name
last_name
first_name
student_no*
utorid

primary key

candidate #1

candidate #2



# Multi-Column Keys

**city**

name
name
province
population



a key!

# Multi-Column Keys

**city**

name
name
province
country
population



a three column  
key



# Multi-Column Keys

**city**

name
name
province
country
population
city_id

candidate #1

candidate #2

# “Natural” Keys

## building

name
code e.g., “BL”
map_url

← a “natural” key

# “Natural” Keys

## building

name
code e.g., “BL”
map_url
building_id e.g., 56

← a “natural” key

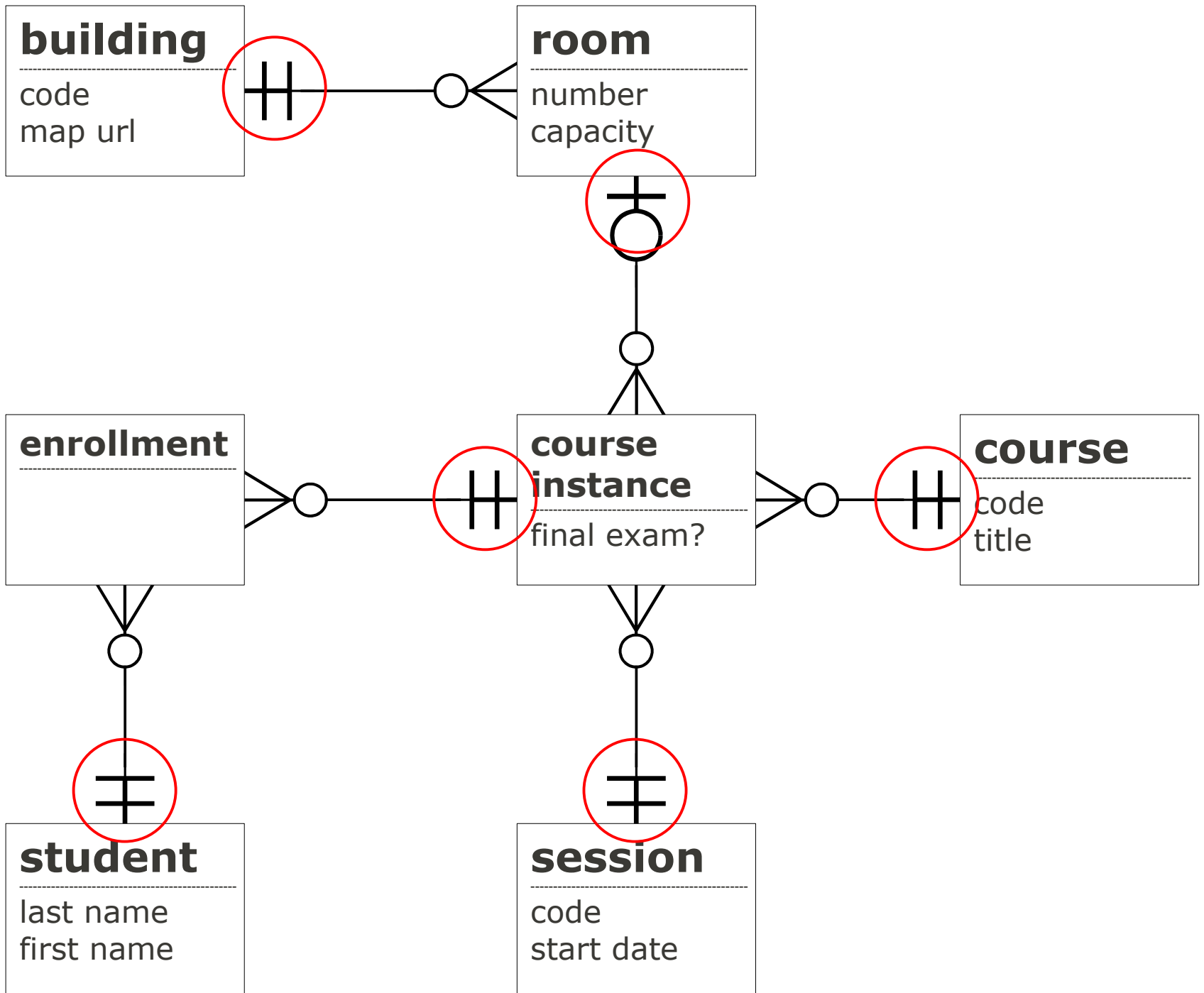
← a “surrogate” key

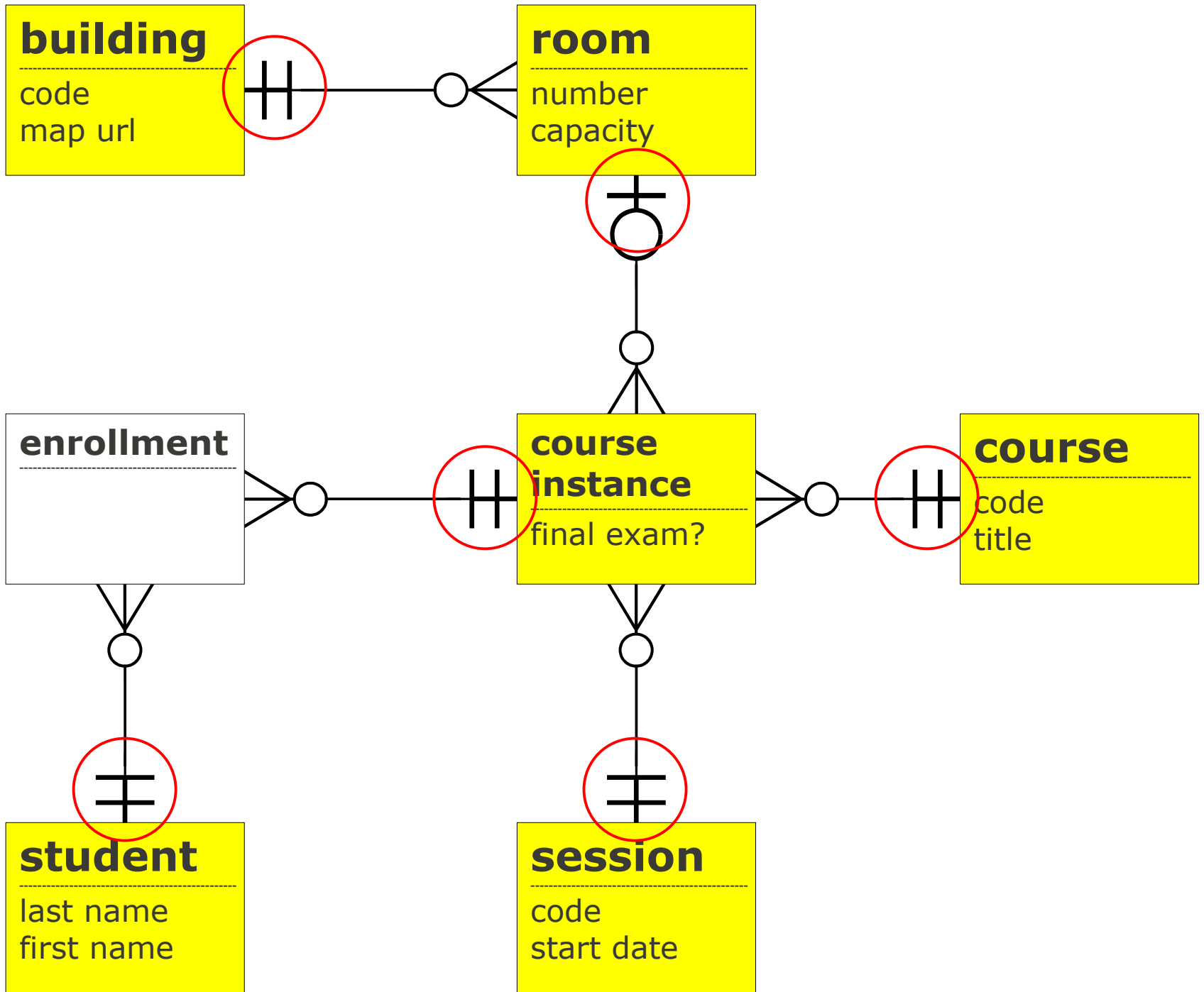
# Step 2

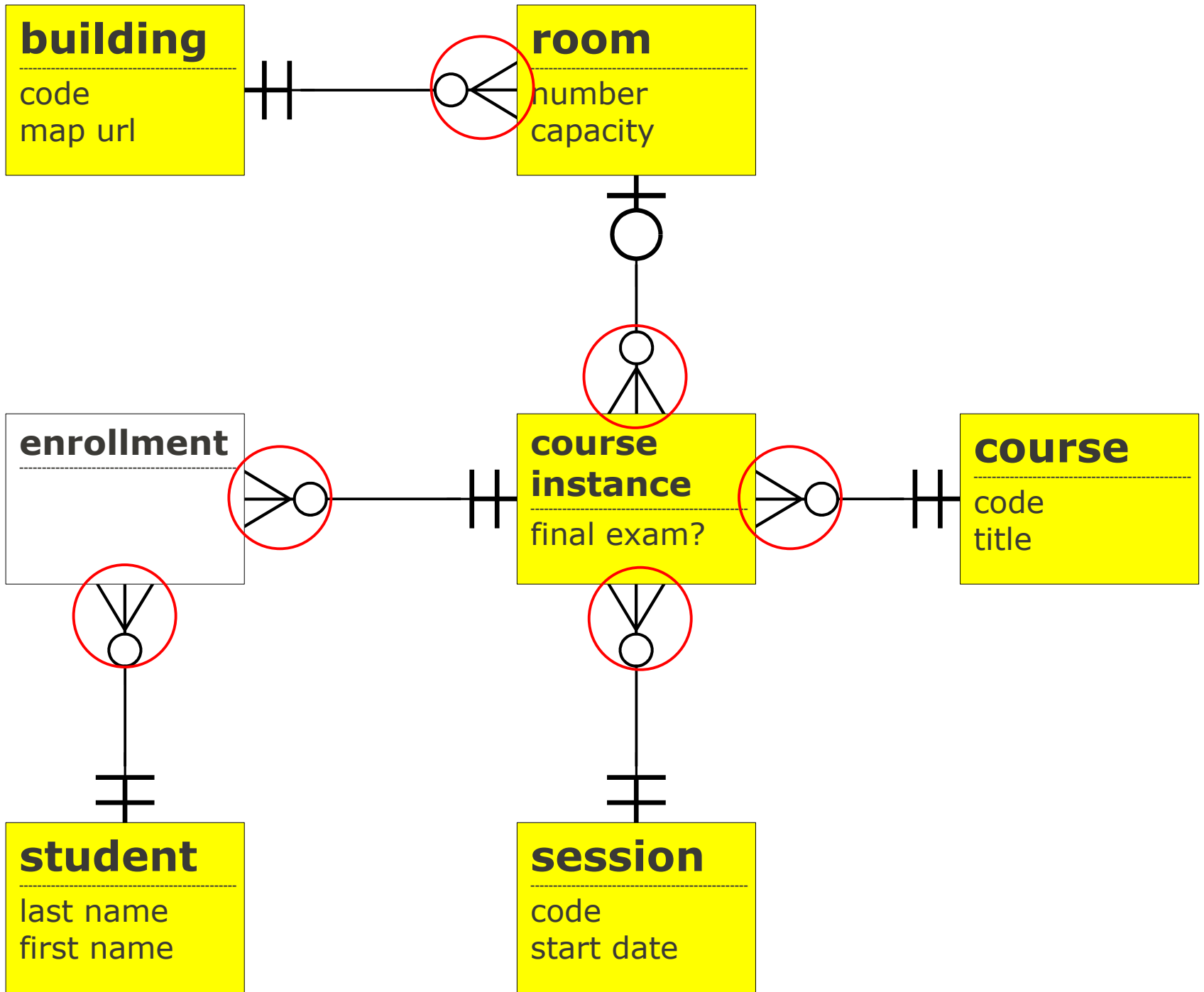
“For each entity that is only on the ‘one’ end of one or more relationships and not at the ‘many’ end of any relationship, create a single-column primary key... if no natural primary key is available.”

# Step 2

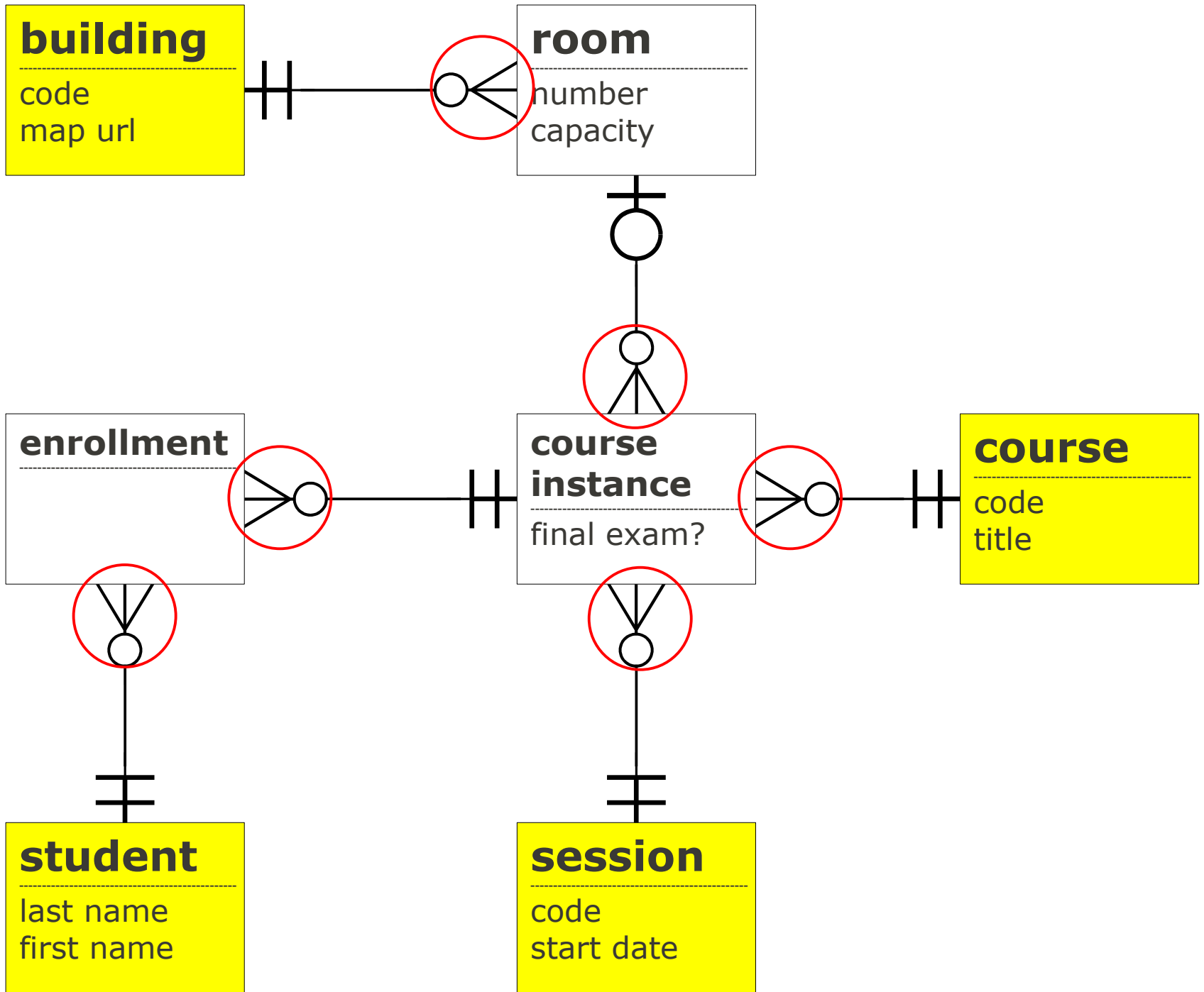
“For each entity that is only on the ‘one’ end of one or more relationships and not at the ‘many’ end of any relationship, create a single-column primary key... if no natural primary key is available.”

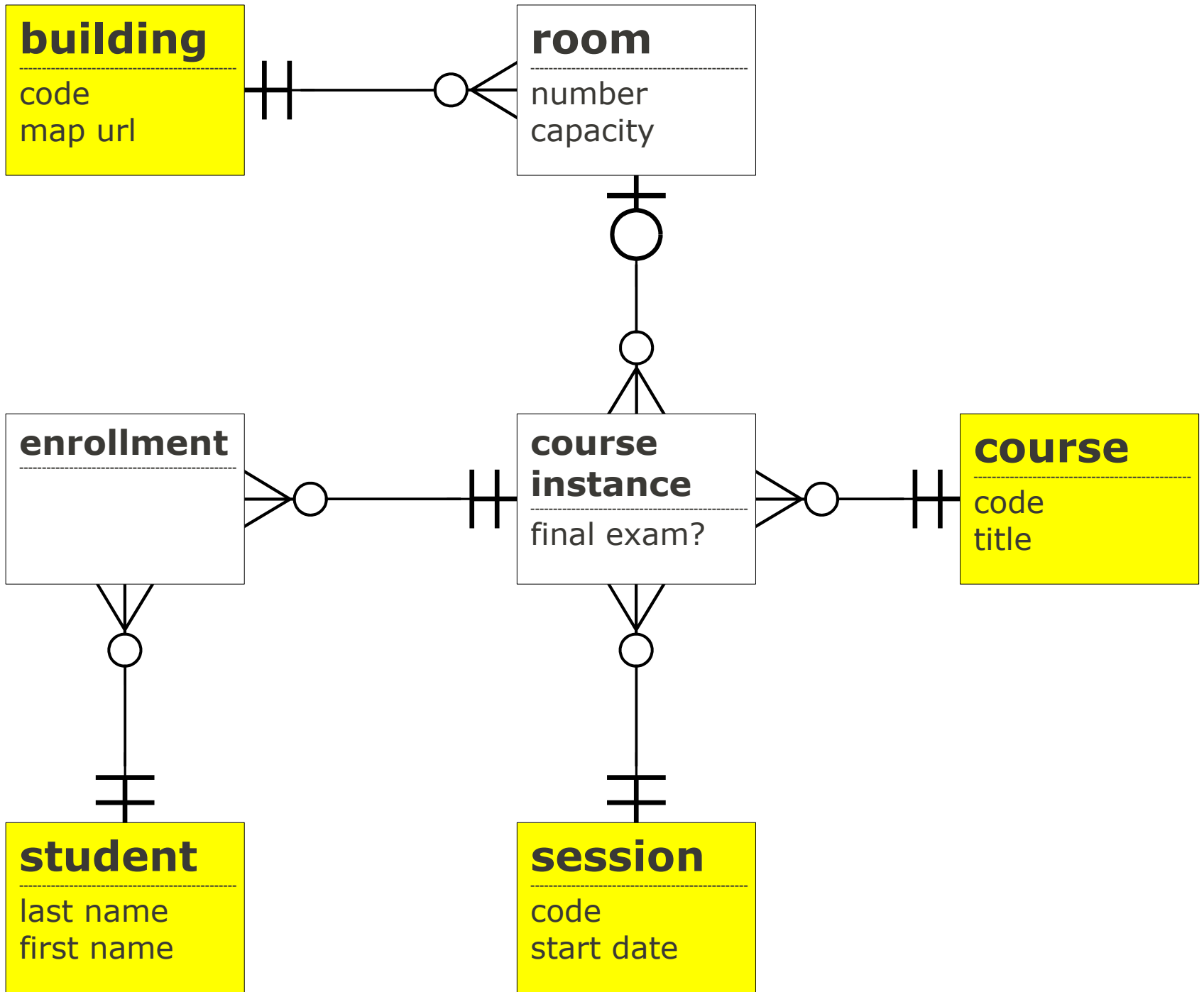












# Step 2

“For each entity that is only on the ‘one’ end of one or more relationships and not at the ‘many’ end of any relationship, **create a single-column primary key... if no natural primary key is available.”**

## **student**

<b>name</b>
last_name
first_name

## **student**

name

last\_name

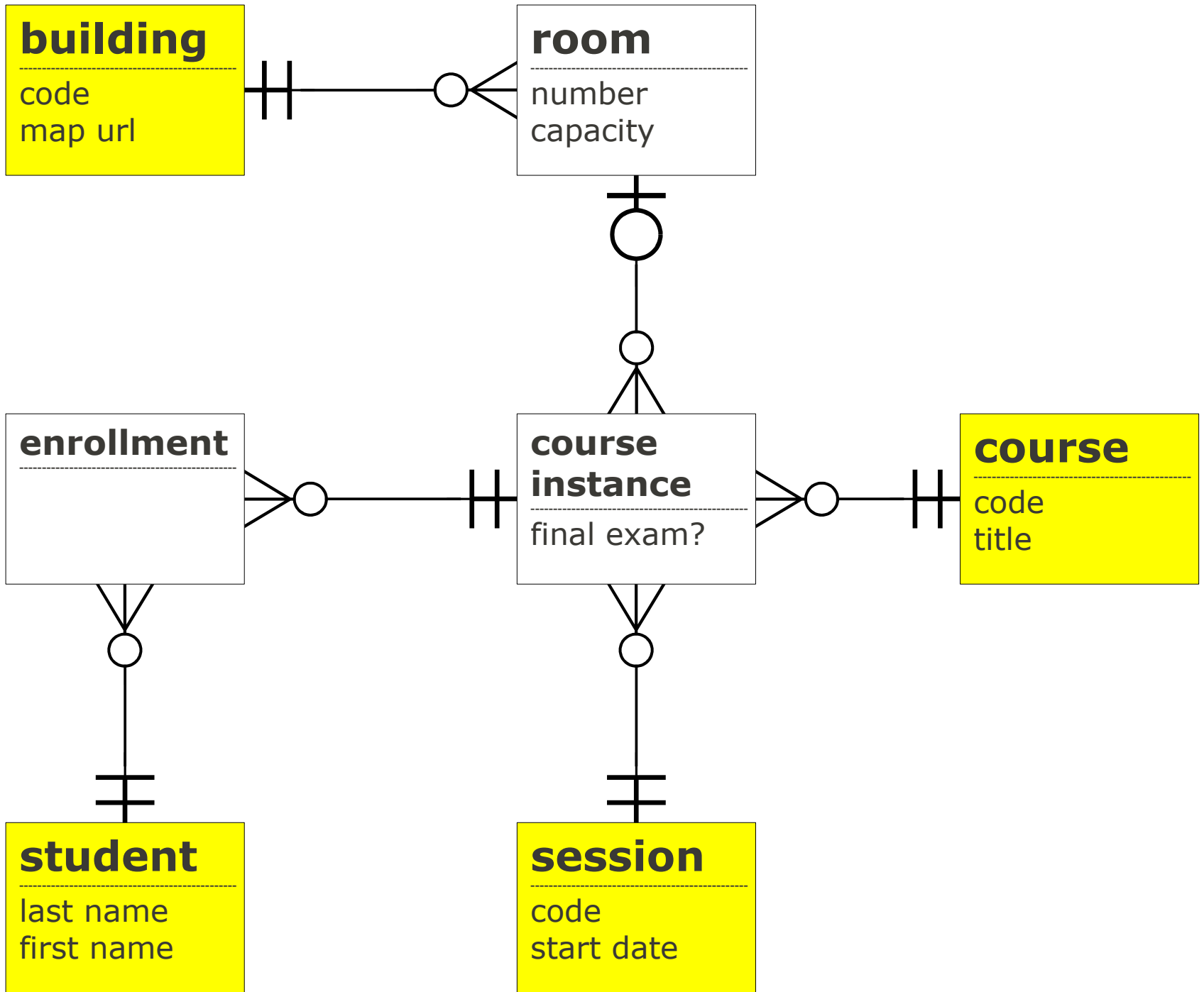
first\_name

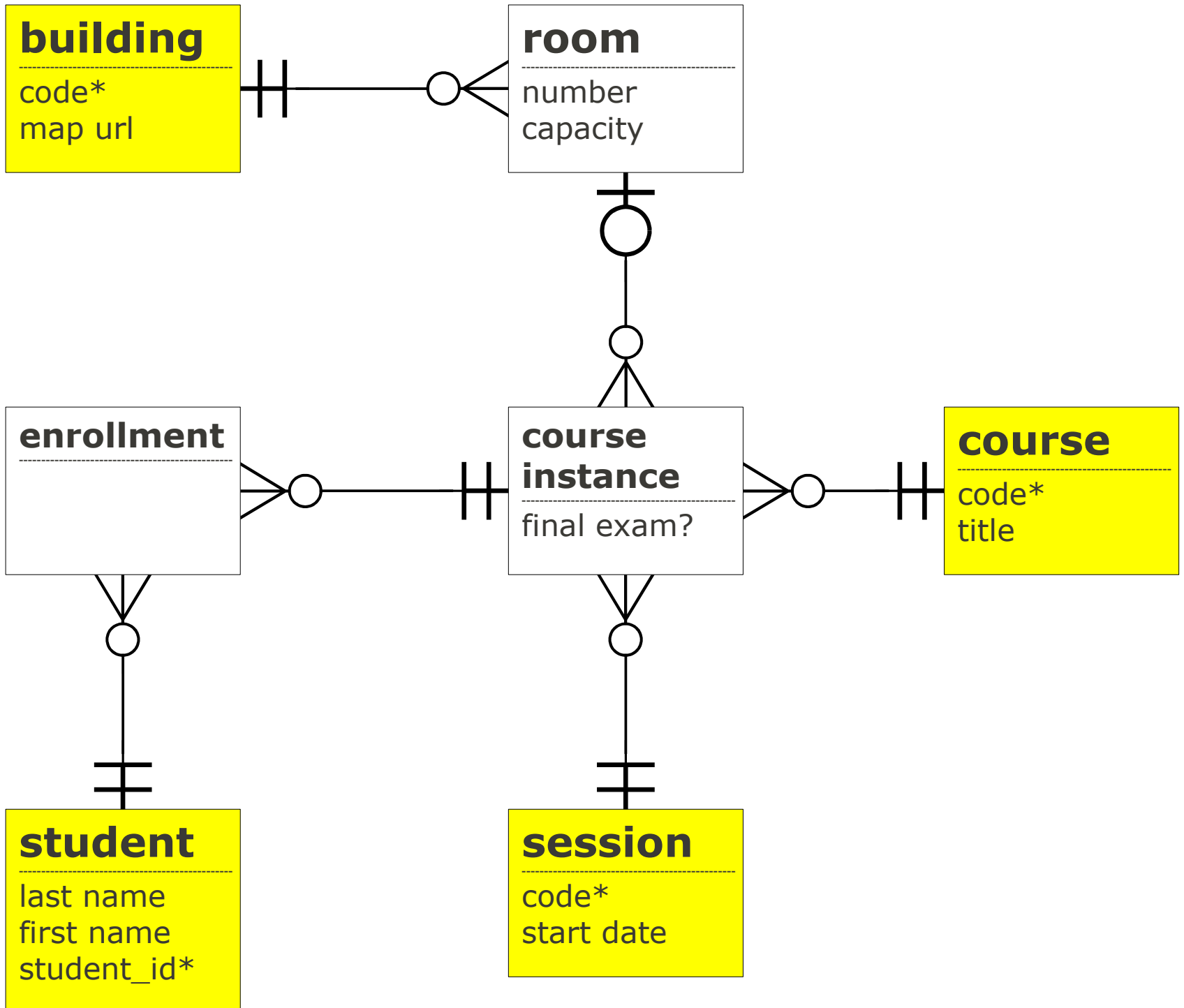
student\_id\*

← let's add it

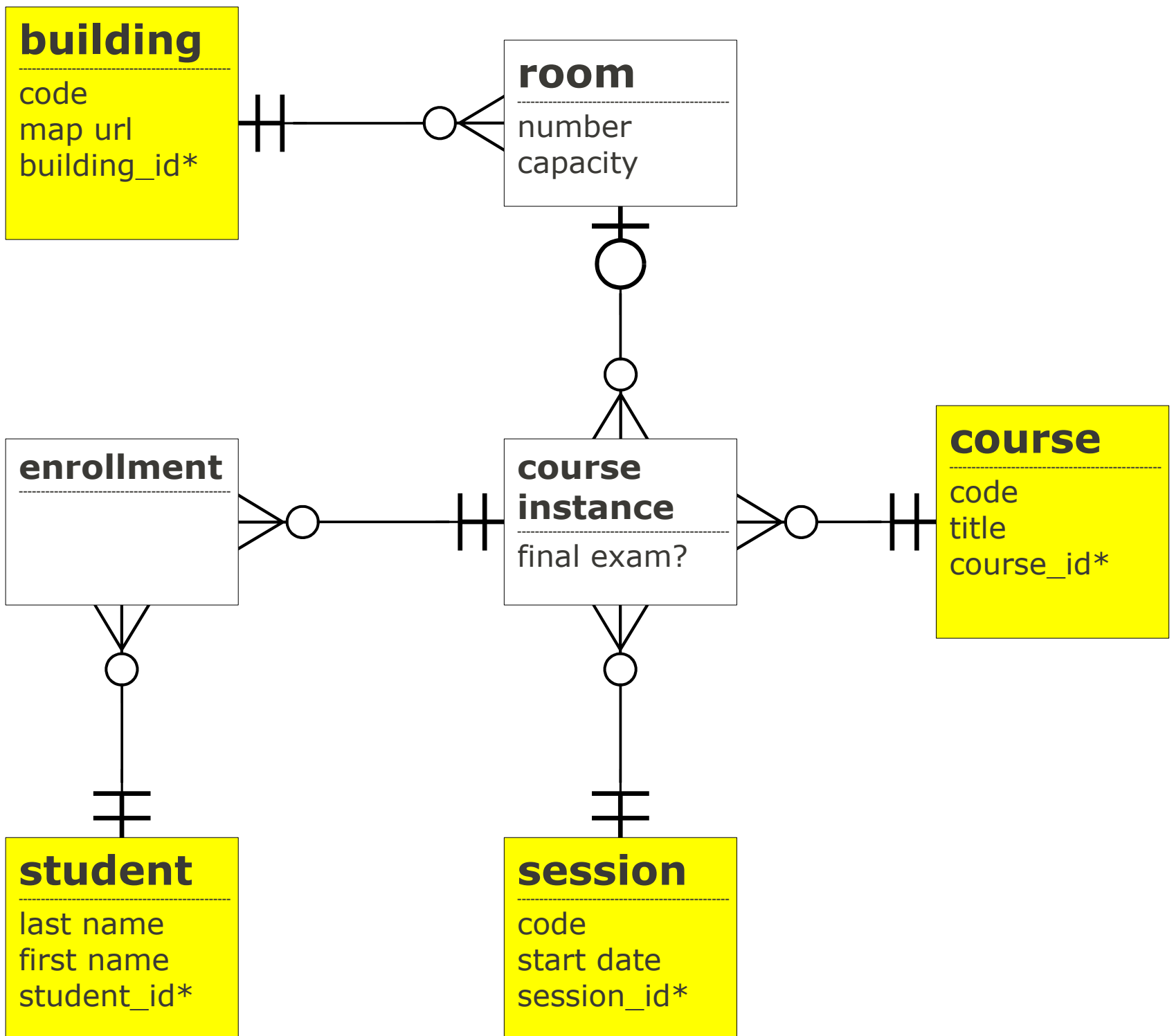
## student

name	type	null	key
last_name	varchar(100)	YES	
first_name	varchar(100)	NO	
student_id	int	NO	PRI







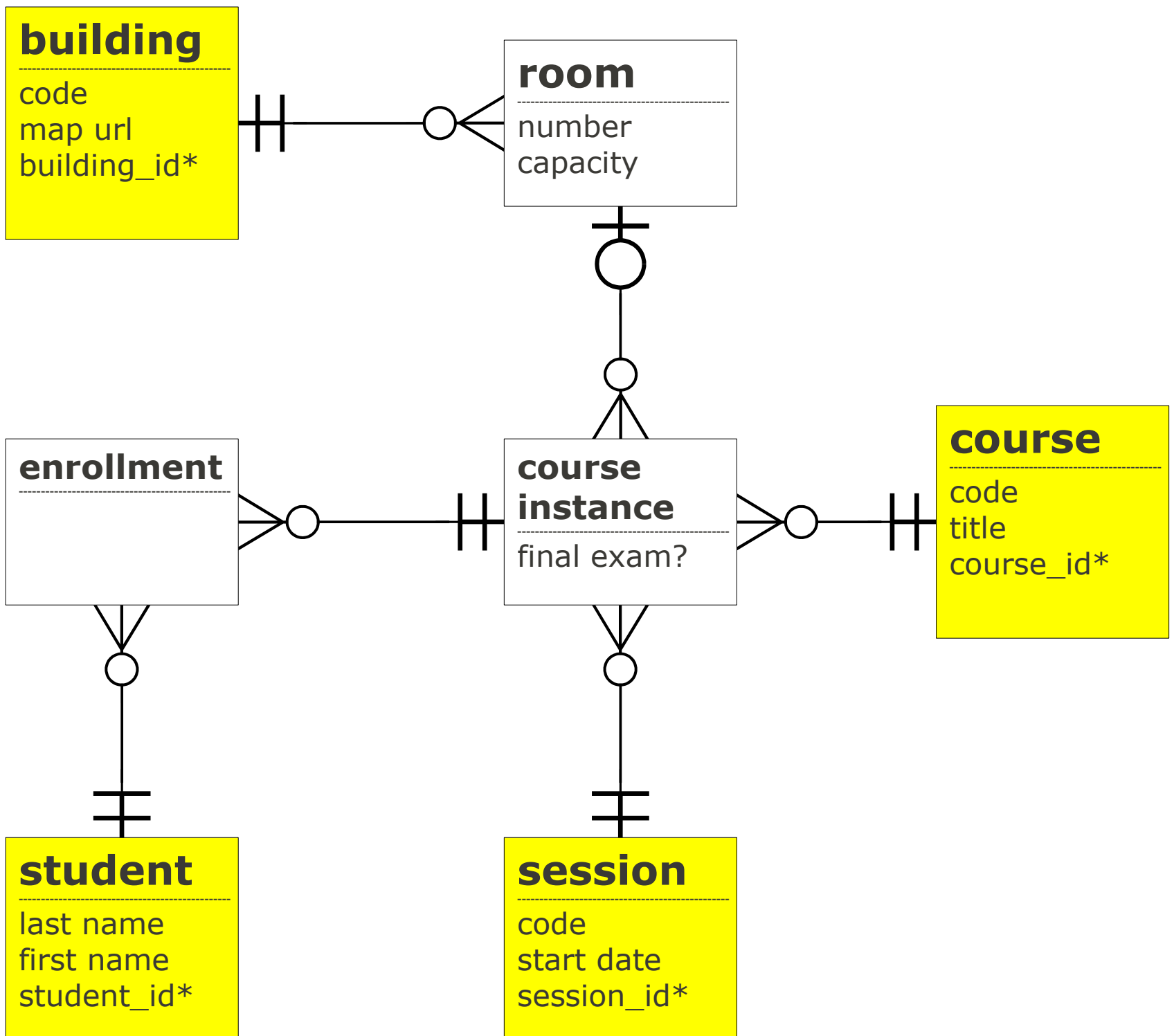


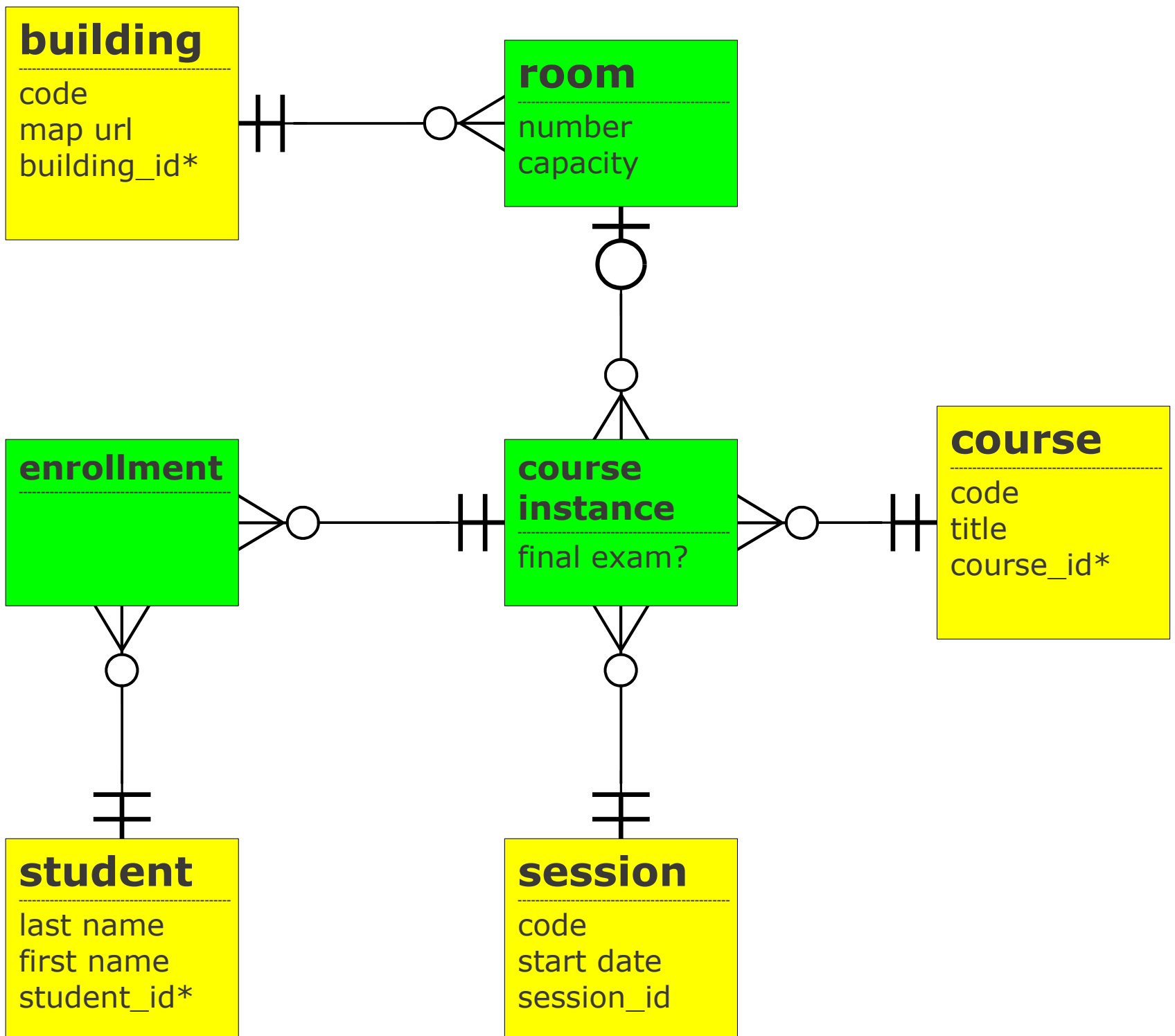
# Step 3

“For each entity that is at the ‘many’ end of one or more relationships, include the primary key of each parent entity in the table as foreign keys.”

# Step 3

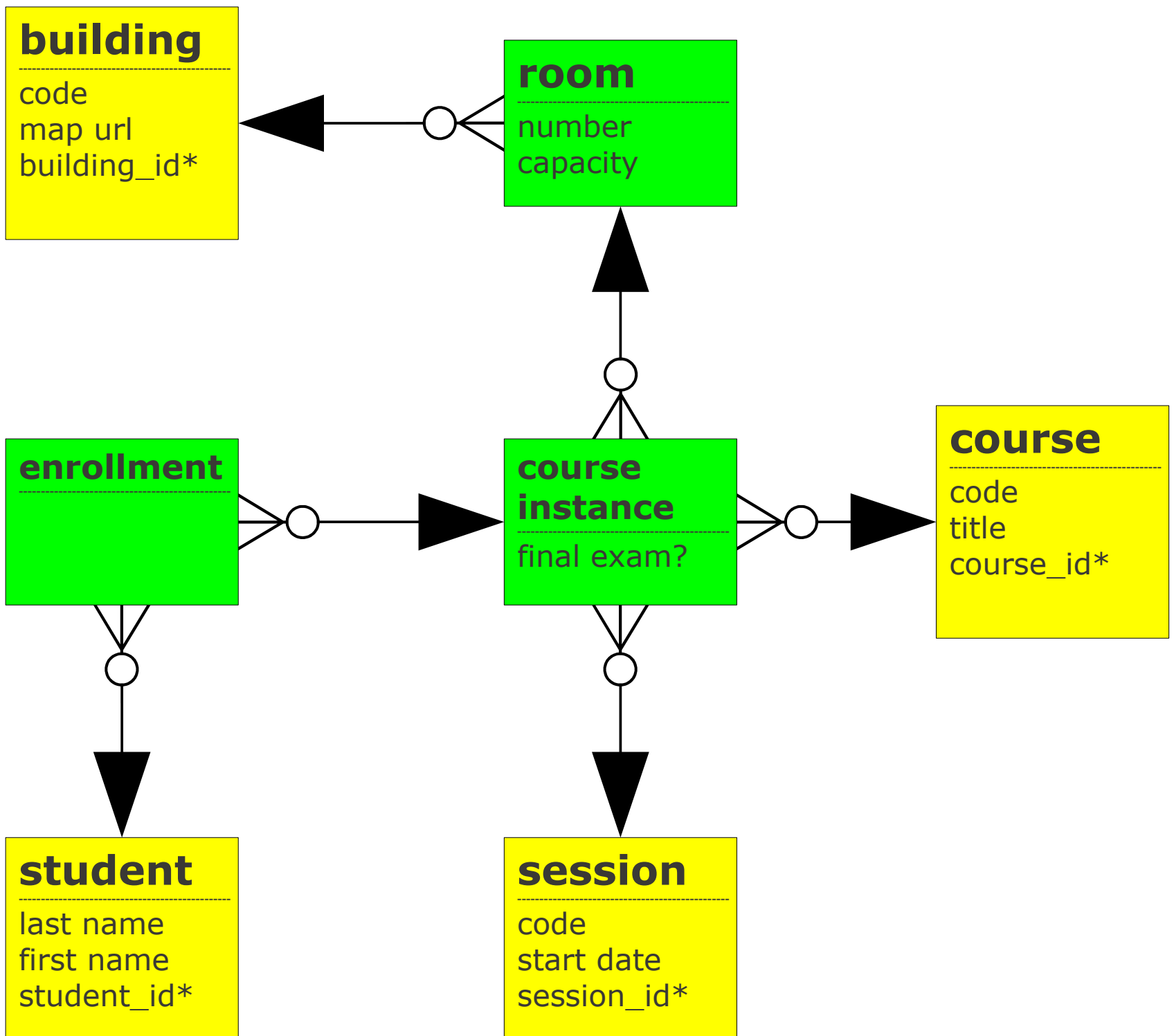
“For each entity that is at the ‘many’ end of one or more relationships, include the primary key of each parent entity in the table as foreign keys.”

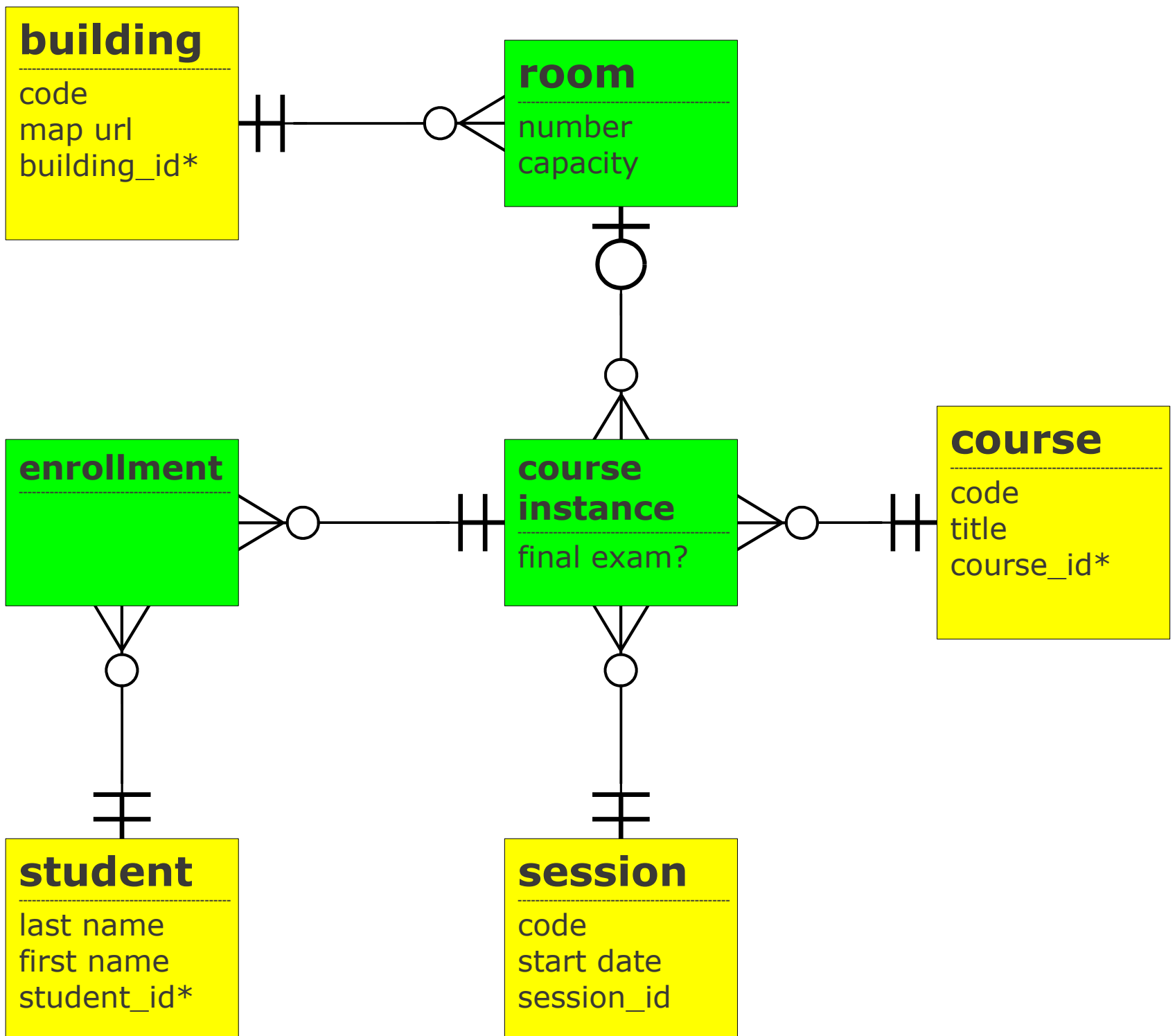




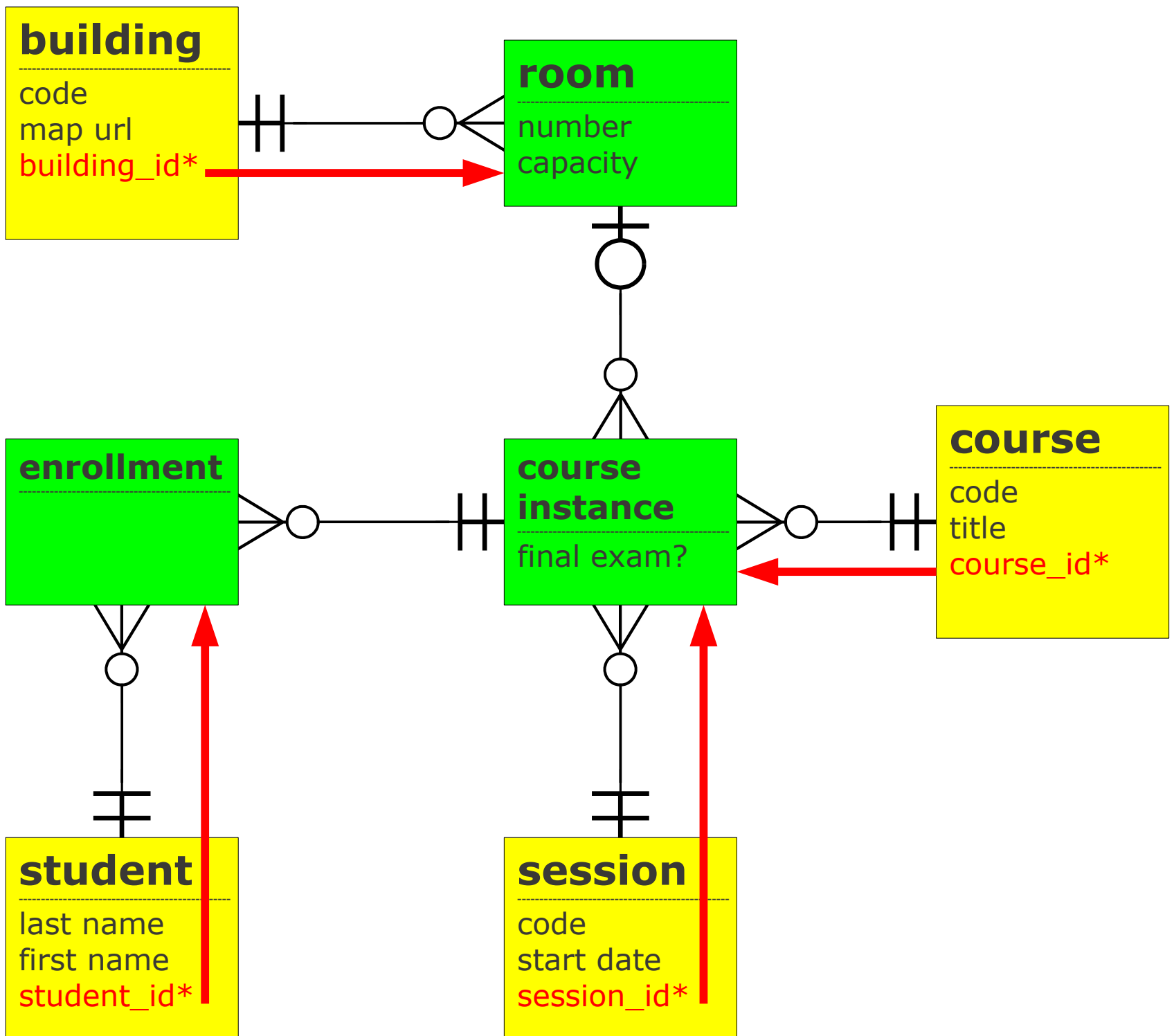
# Step 3

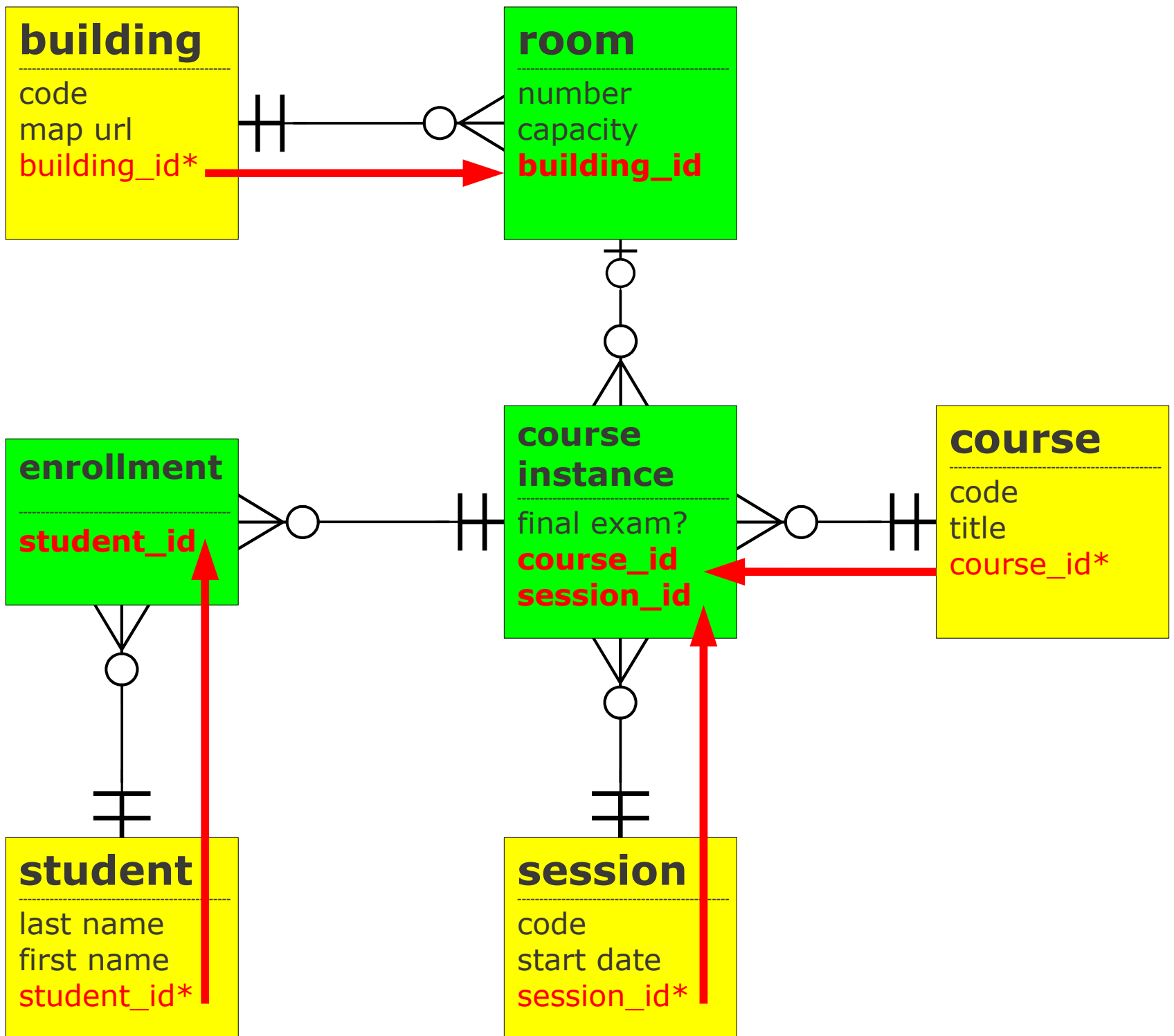
“For each entity that is at the ‘many’ end of one or more relationships, include the primary key of each parent entity in the table as foreign keys.”

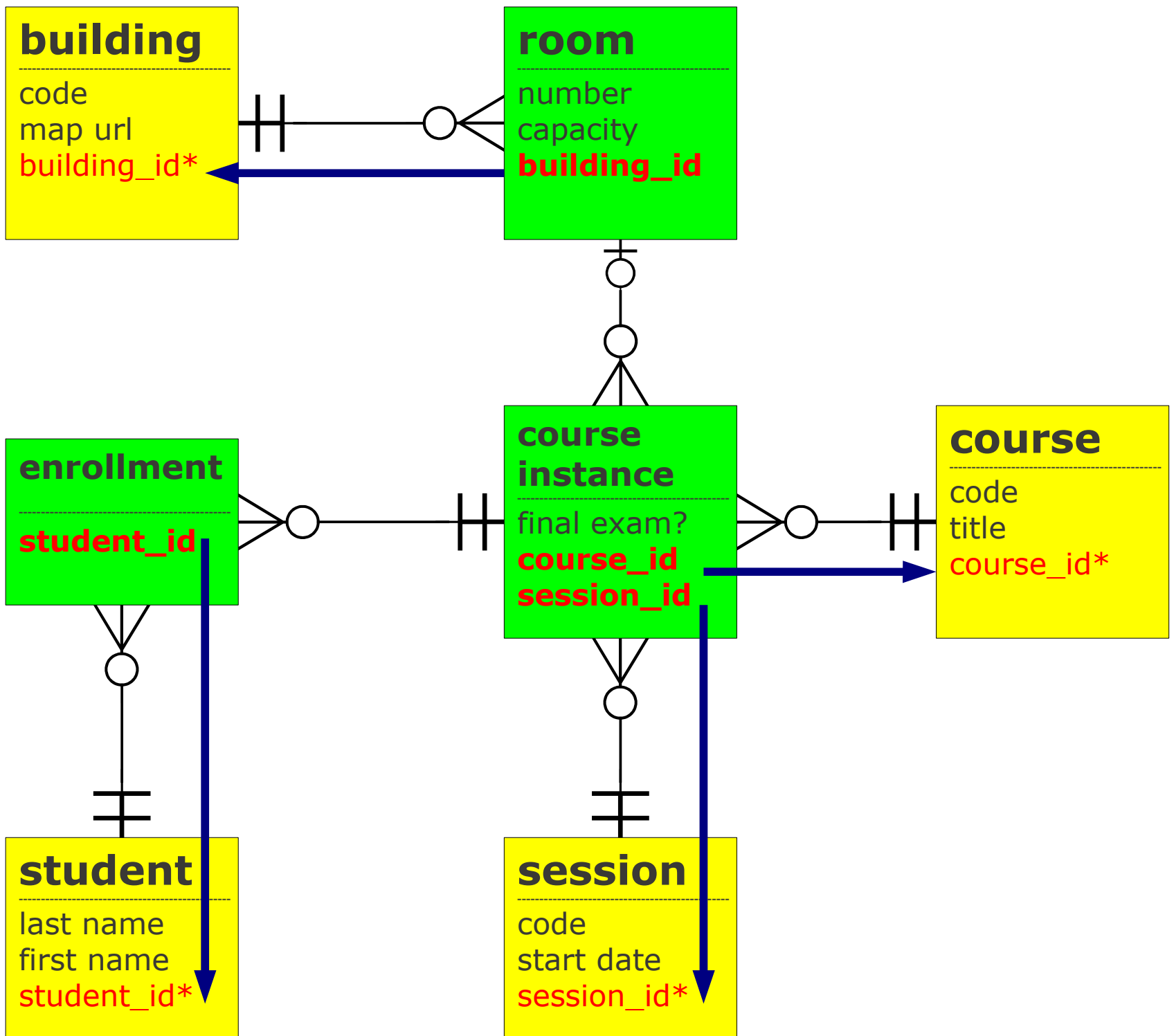






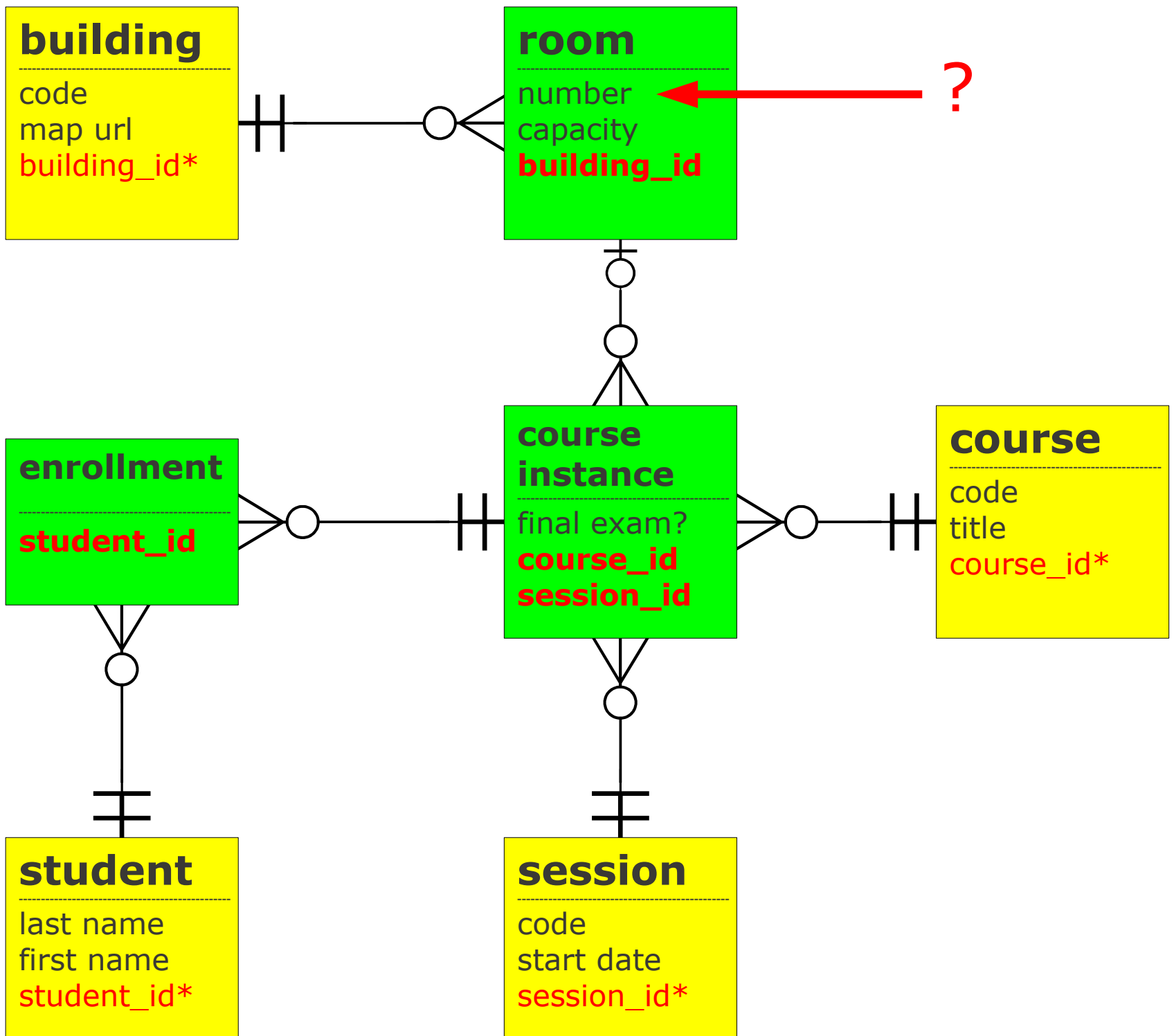






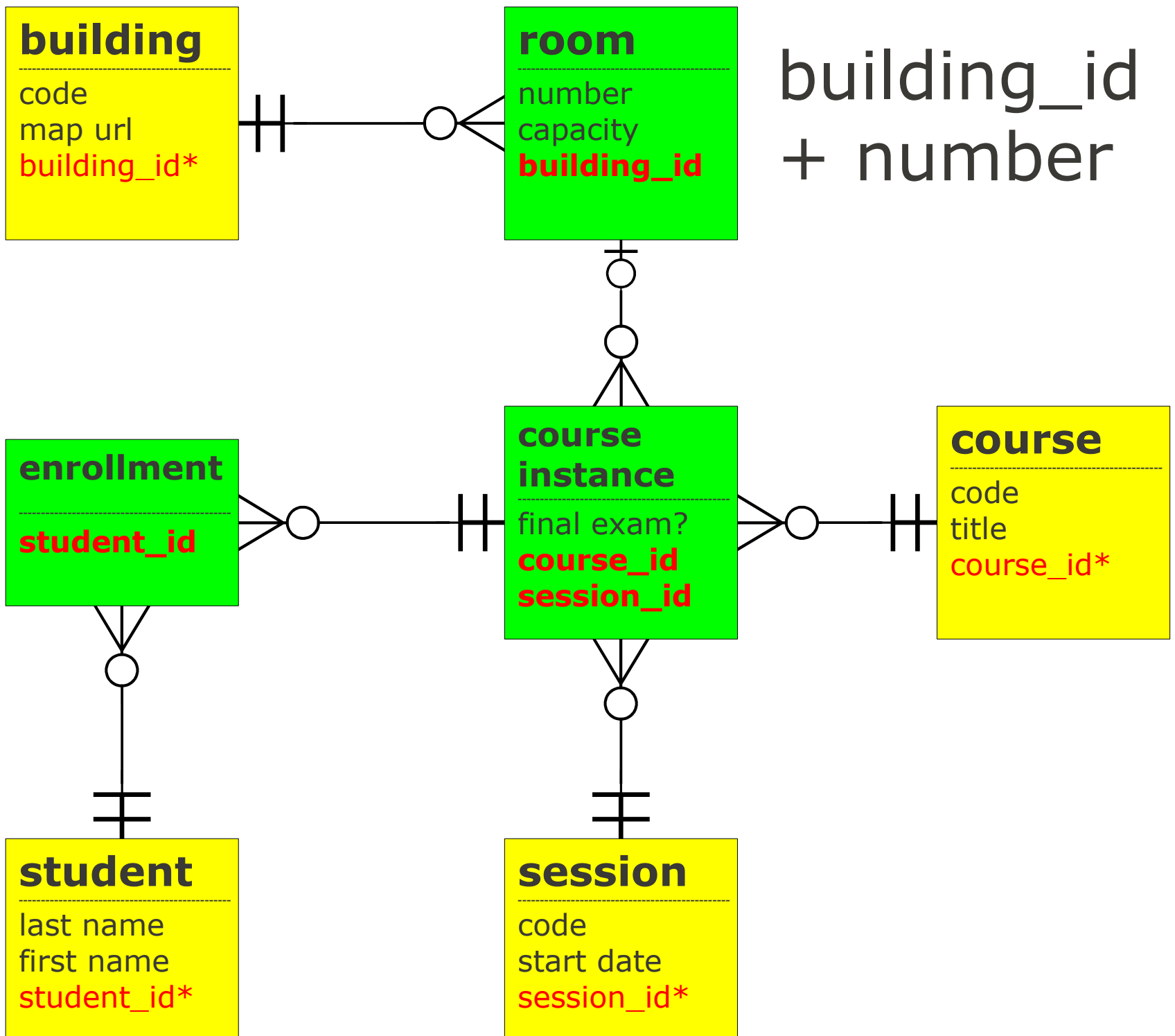
# Step 4a

“If an entity at the ‘many’ end of one or more relationships, has a natural primary key, use that single key as a the primary key.”



# Step 4b

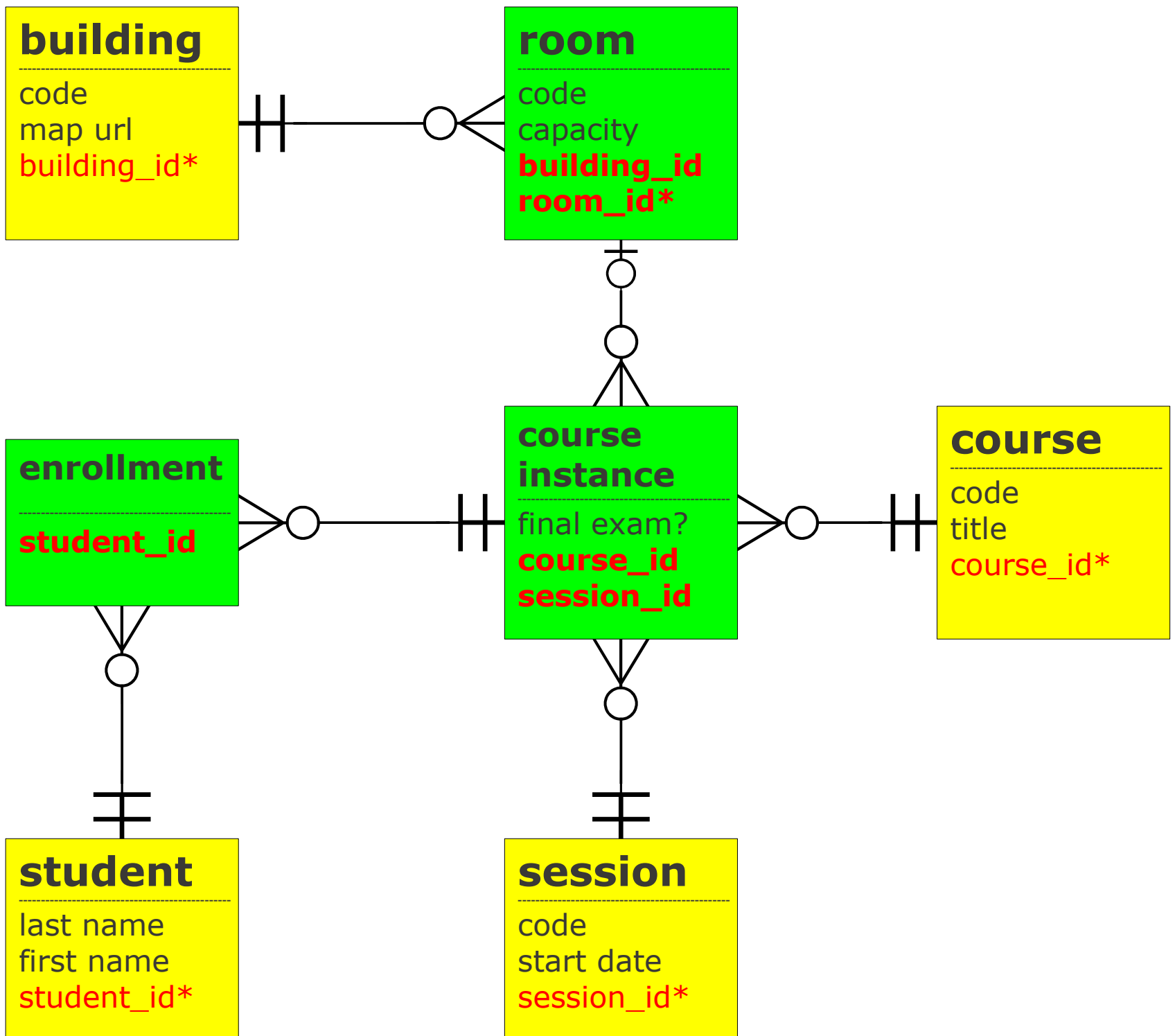
“Otherwise, concatenate the primary key of its parent with any other column or columns needed for uniqueness to form the table’s primary key.”

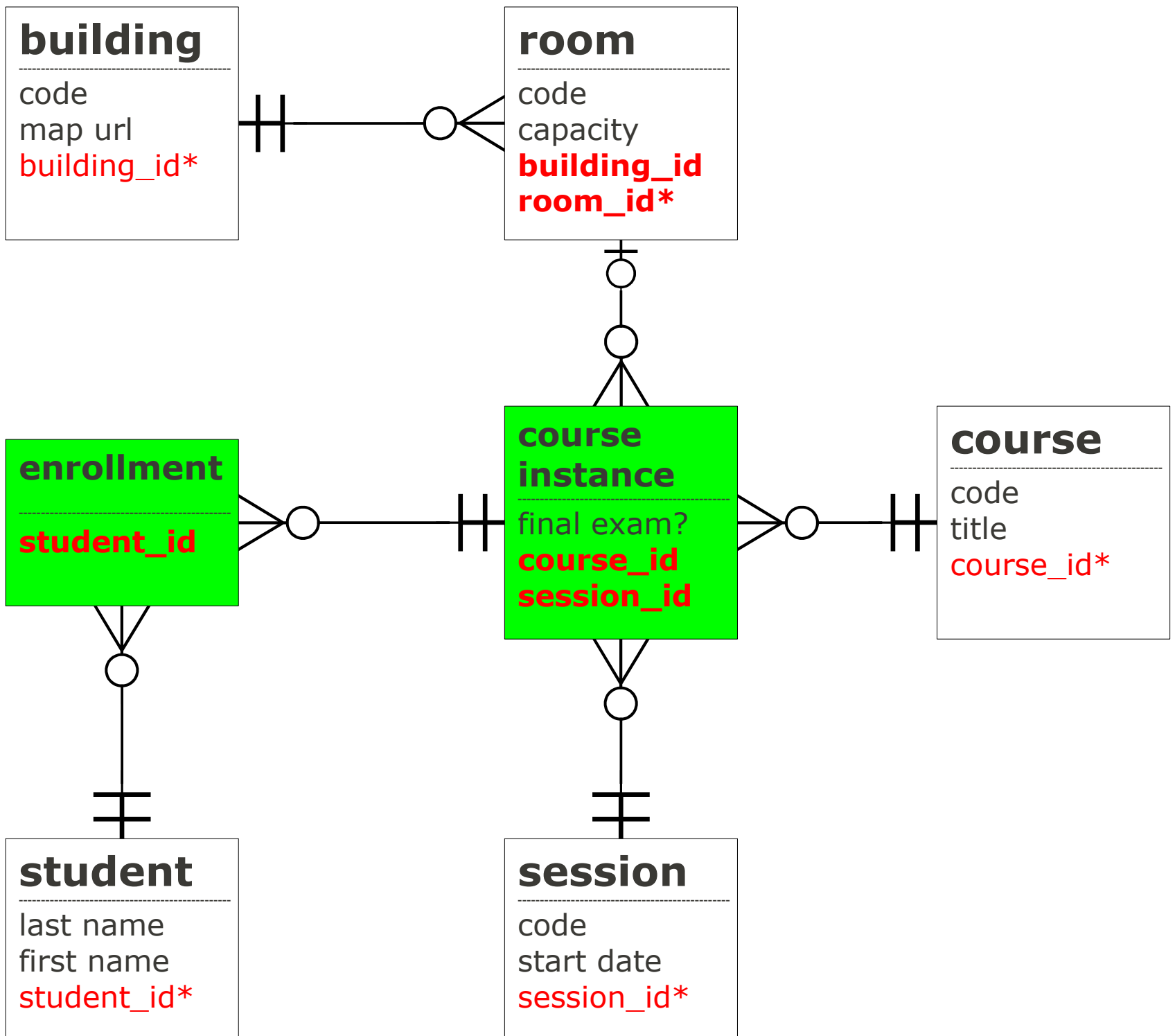


# Step 4c

If necessary, add a field!

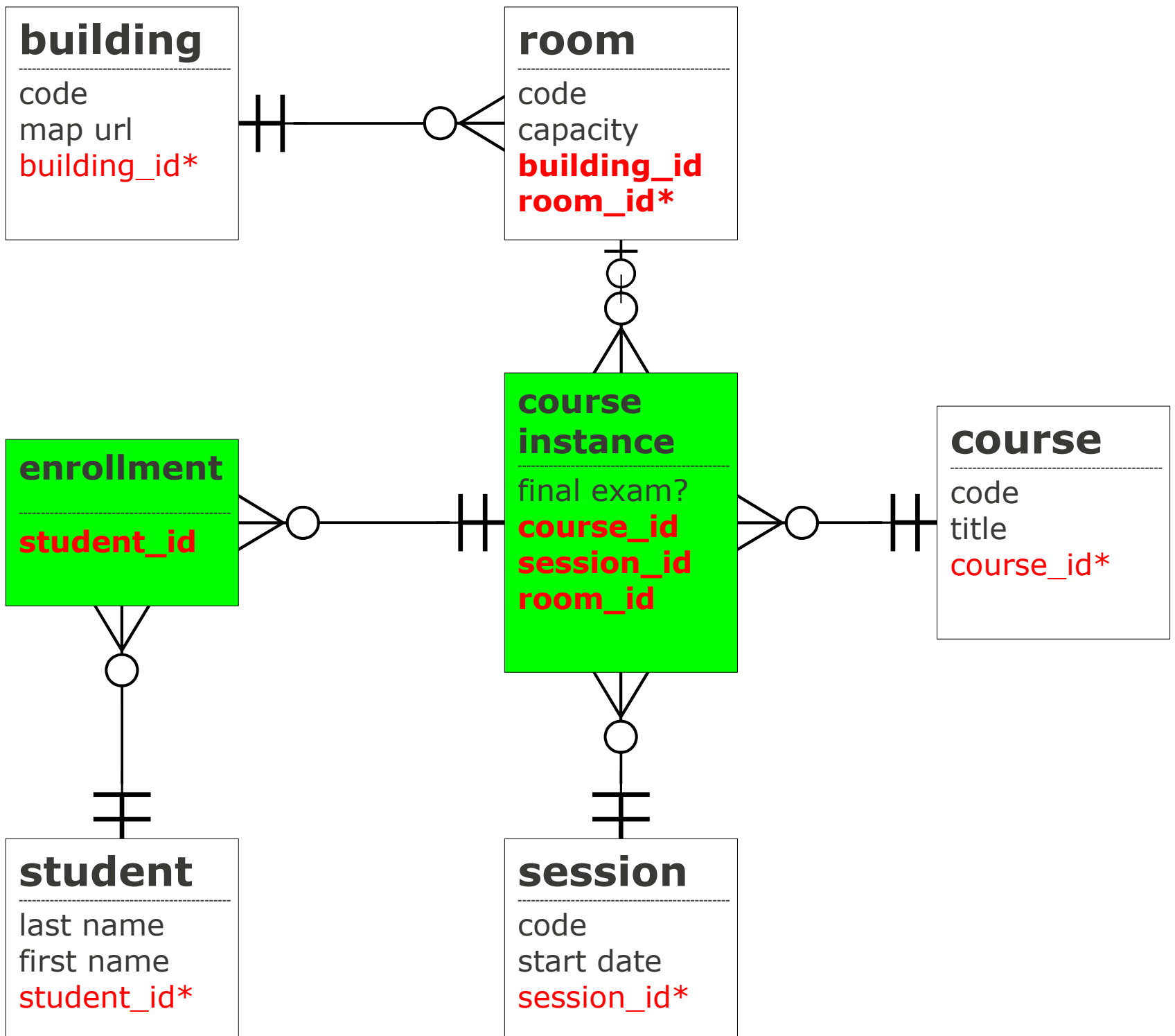


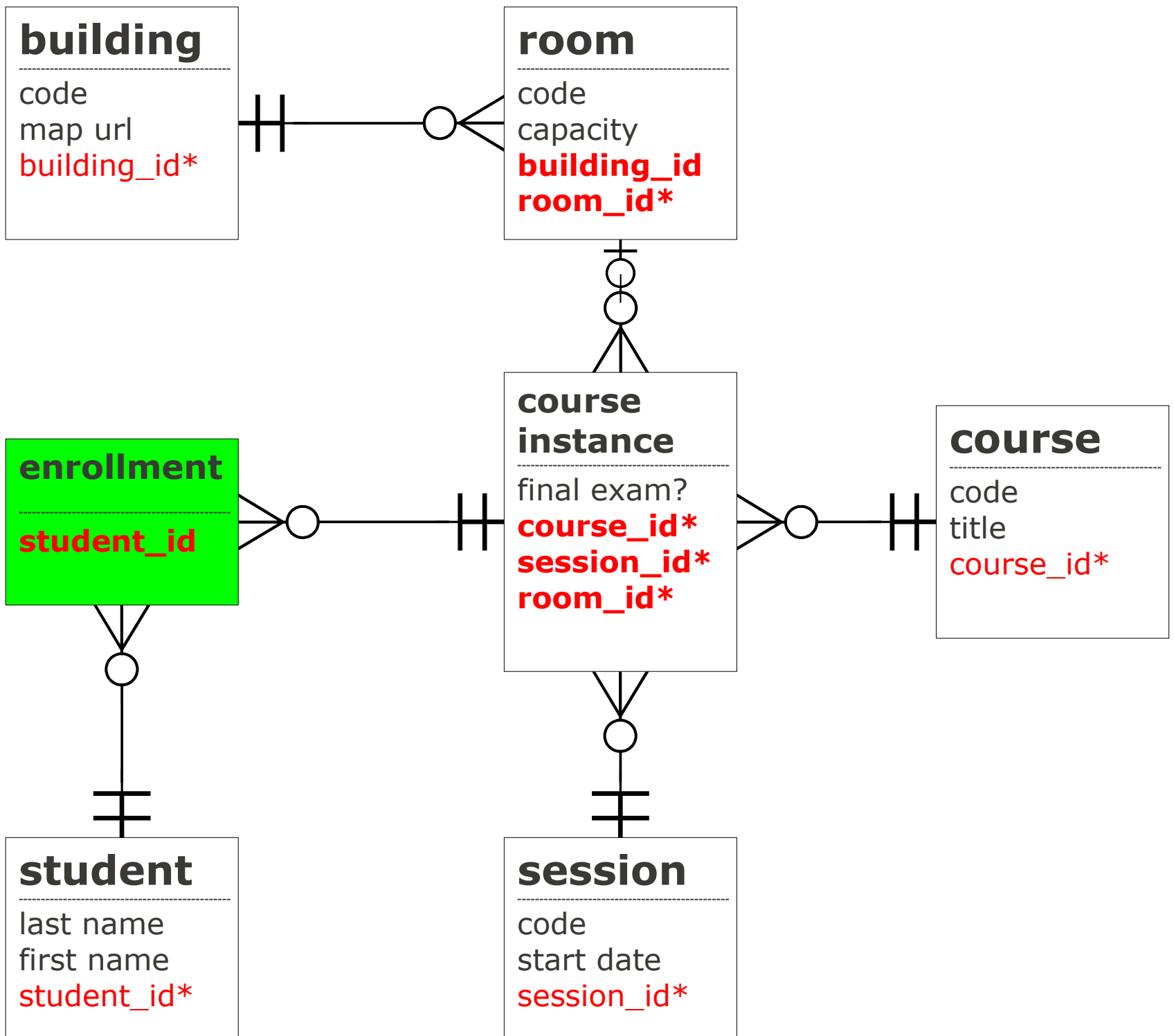


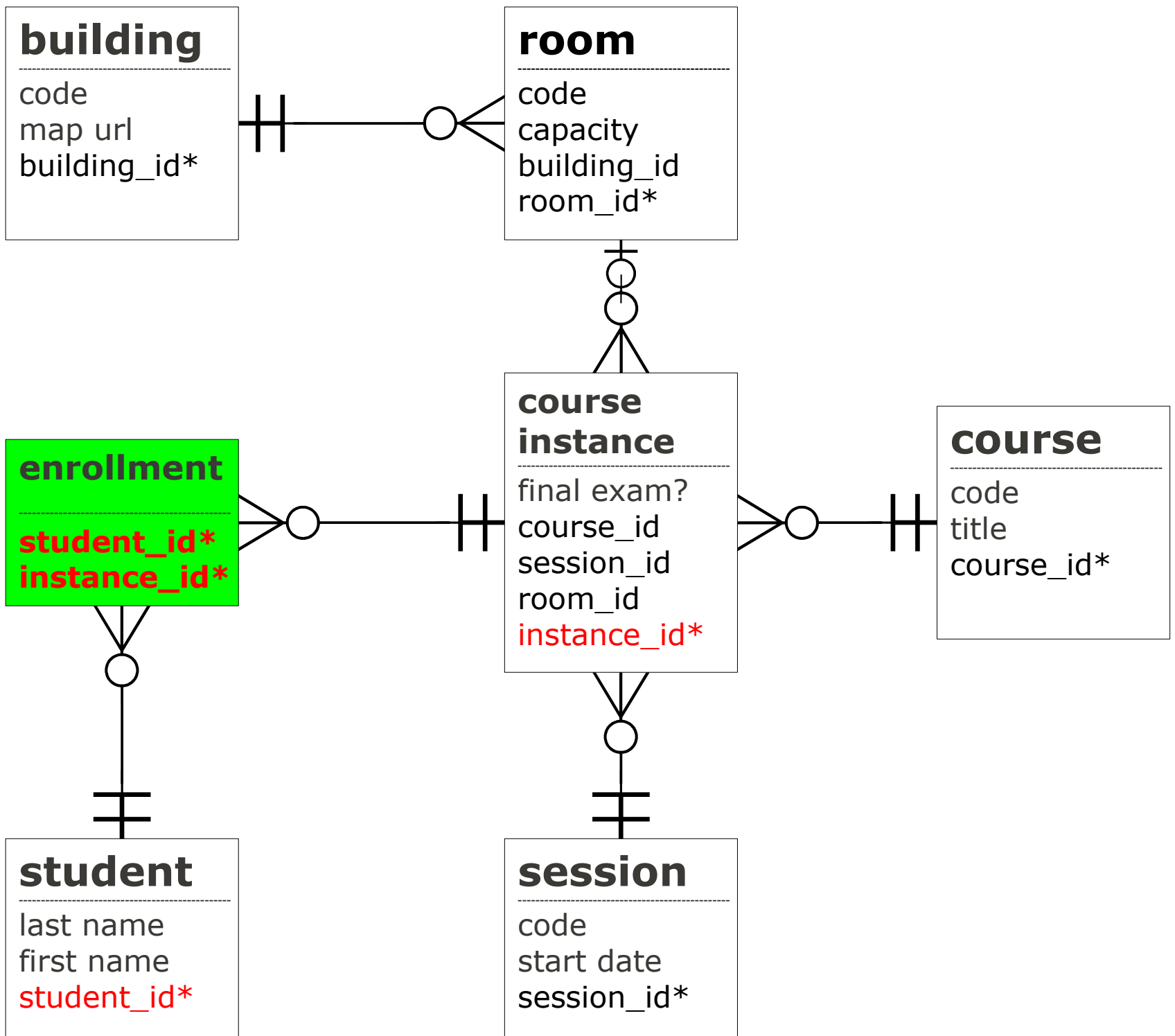


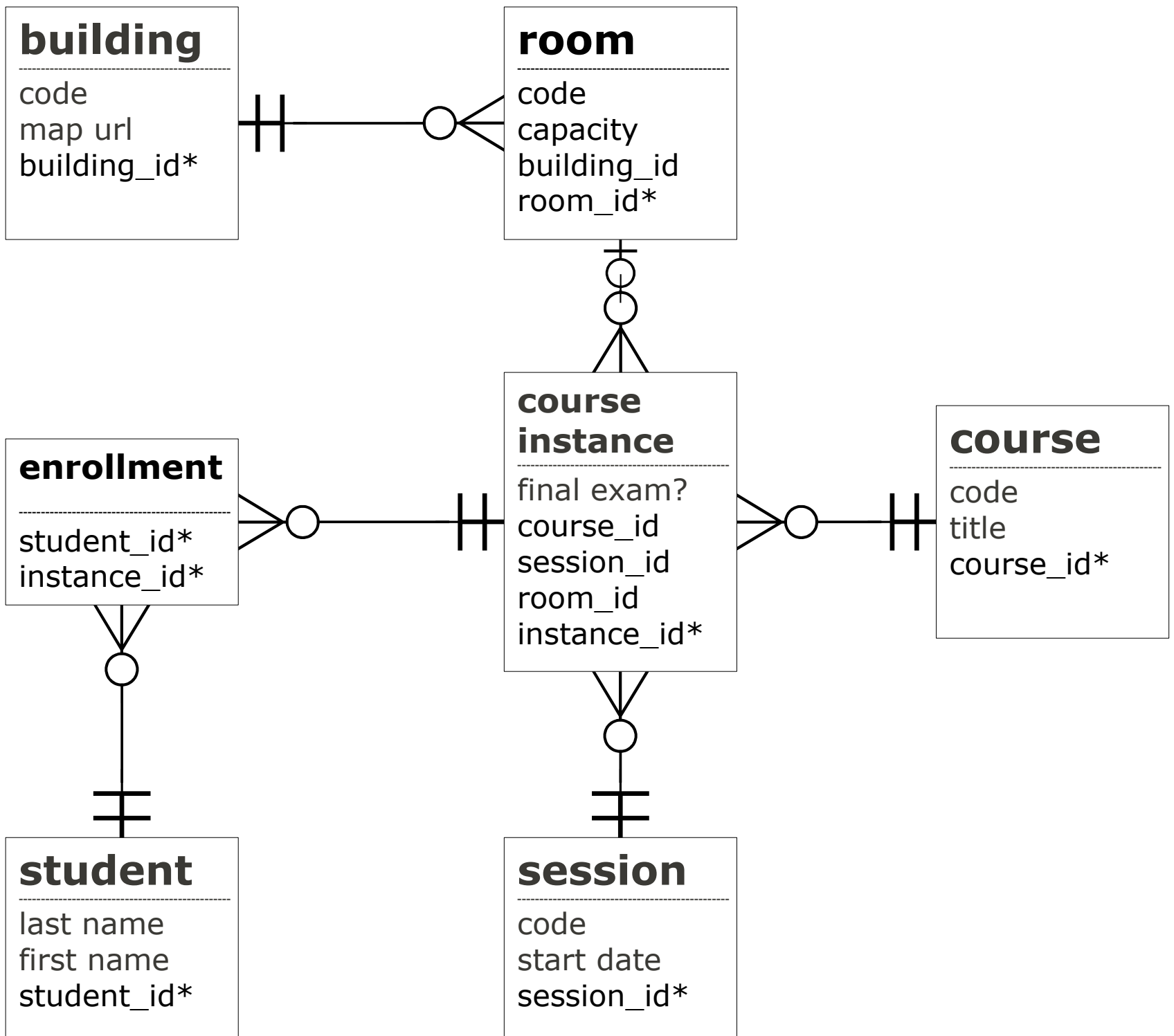
# Step 4d

Go back to step 3!





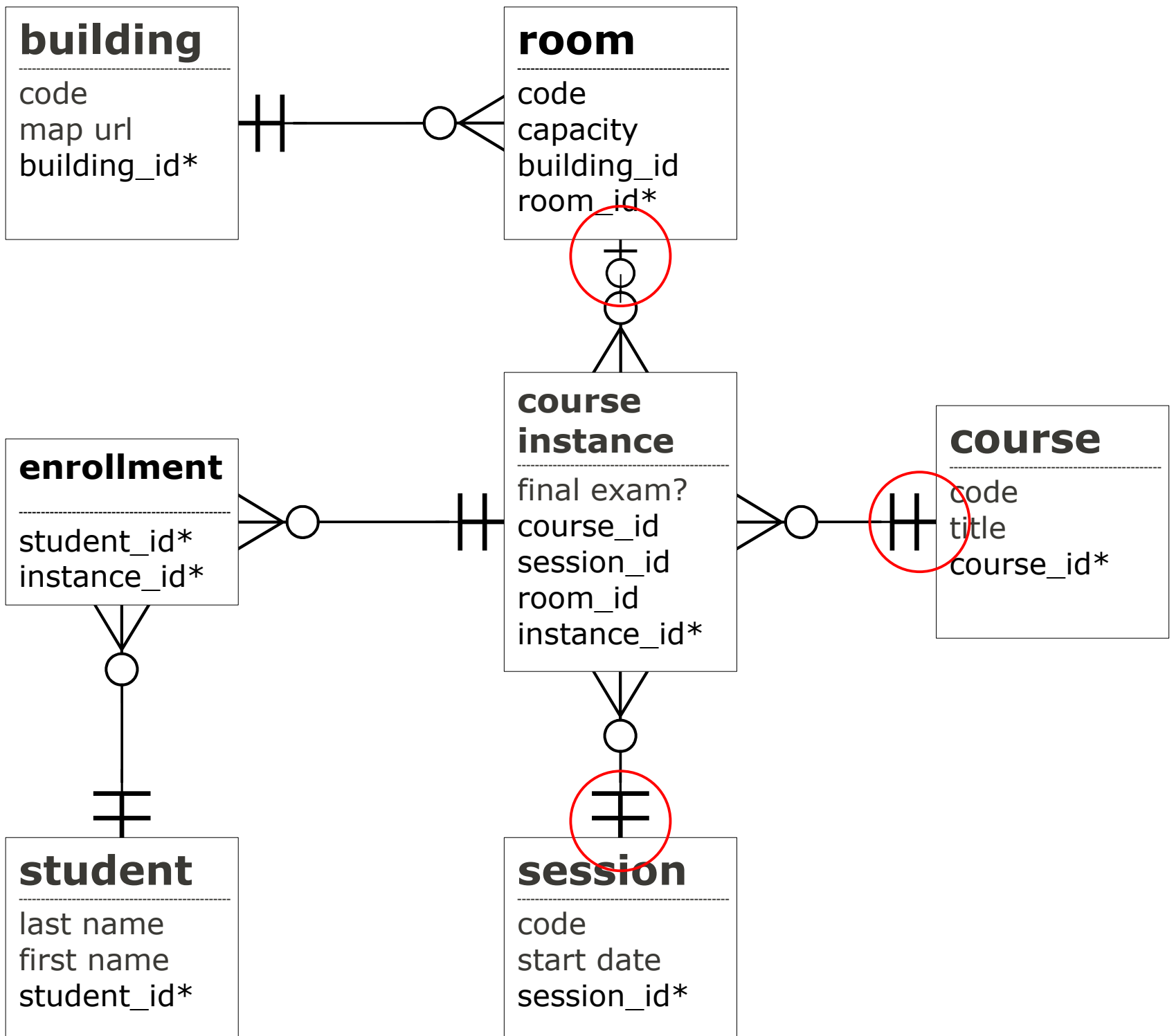




## course\_instance

name	type	null	key
instance_id	int	NO	PRI
course_id	int		
session_id	int		
room_id	int		
final_exam	bool		





## course\_instance

name	type	null	key
instance_id	int	NO	PRI
course_id	int	NO	
session_id	int	NO	
room_id	int	YES	
final_exam	bool	YES	

# Step 5

Normalize!

# Normalization

Checking that the table does not have any well-known problems

# What Problems?

problem type	cause	solution
update	redundancy	decomposition
insertion	redundancy	decomposition
deletion	redundancy	decomposition

# Normal Forms

5<sup>th</sup> Normal Form

4<sup>th</sup> Normal Form

BC Normal Form

3<sup>rd</sup> Normal Form

2<sup>nd</sup> Normal Form

1<sup>st</sup> Normal Form

fixing  
weird  
issues

you can do it!

trivial!

# 1NF

No multi-valued attributes

pet_id	owner	names
1	Bob	“Slim”, “the Serpent”, “Ribbon”
2	Gwen	“Fluffy”, “Big Dog”



# The Wrong Solution

pet_id	owner	name1	name2	name3
1	Bob	"Slim"	"the Serpent"	"Ribbon"
2	Gwen	"Fluffy"	"Big Dog"	

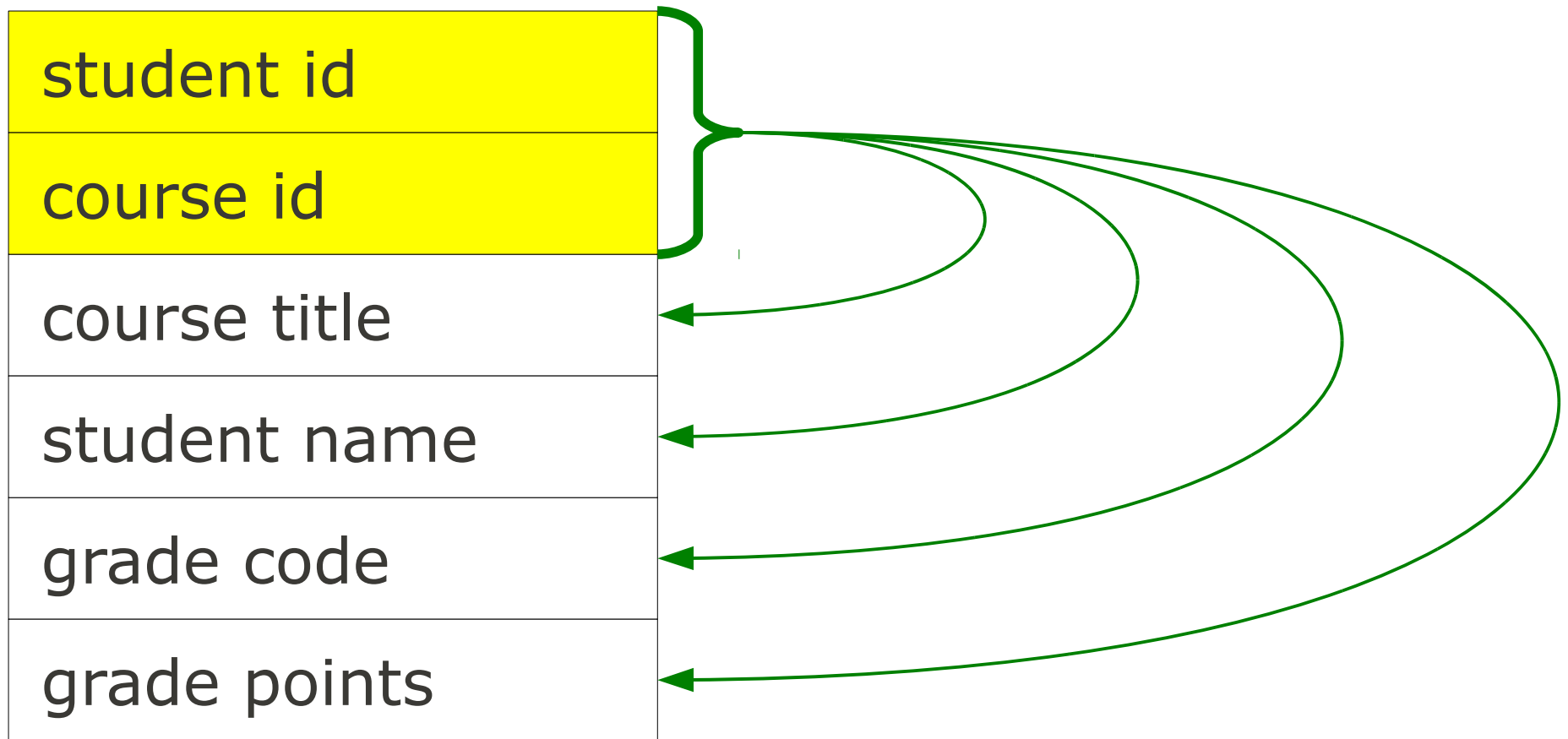
# The Right Solution

pet_id	owner
1	Bob
2	Gwen

pet_id	names
1	Slim
1	the Serpent
1	Ribbon
2	Fluffy
2	Big Dog

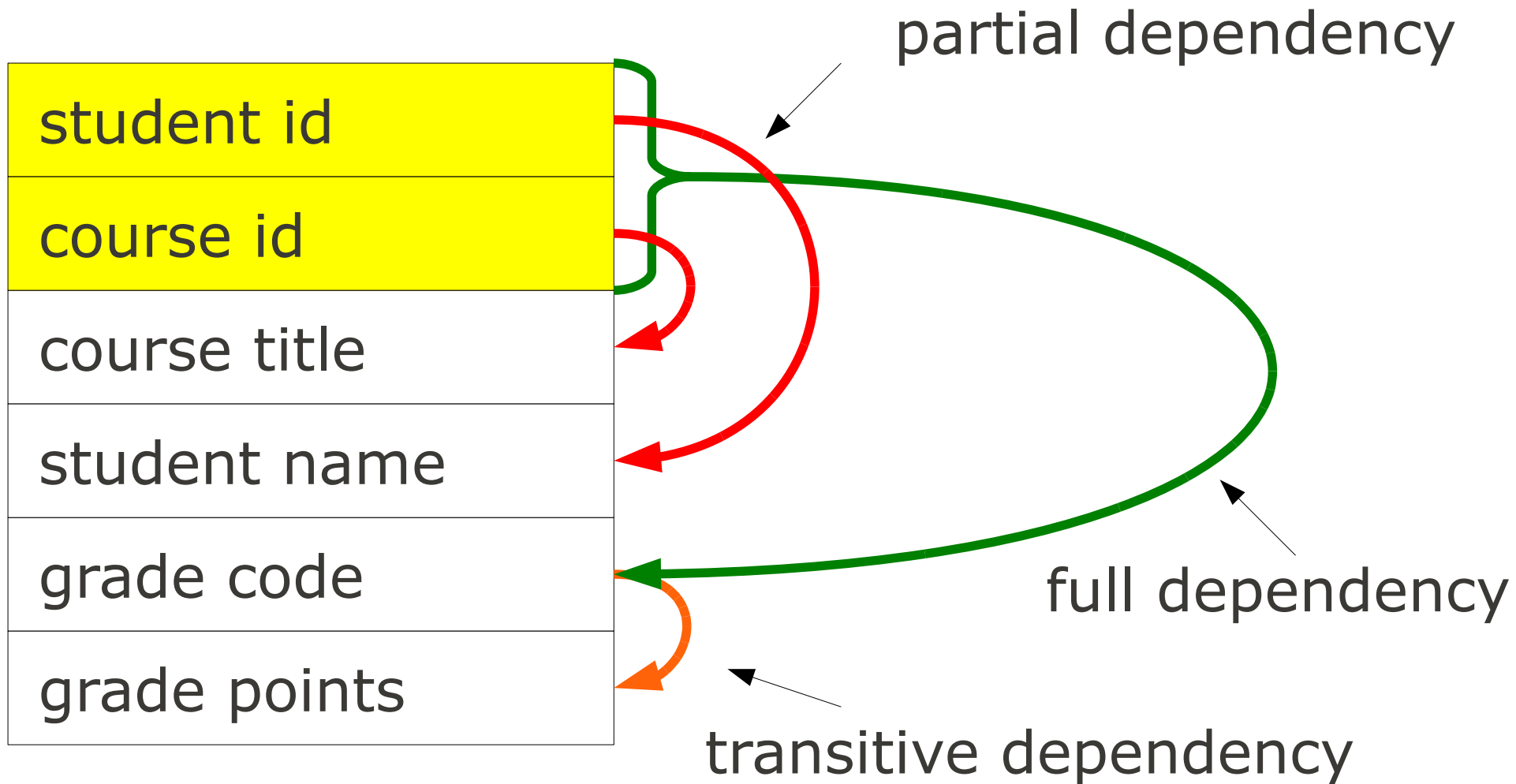
# 2NF and 3NF

Getting rid of  
“functional dependencies”



“Each attribute must describe the key,  
the whole key, and nothing but the key.”

“So help me Codd!”



“Each attribute must describe the key,  
the whole key, and nothing but the key.”

“So help me Codd!”

# 2NF and 3NF

## **2NF:**

no partial dependencies  
("the whole key")

## **3NF:**

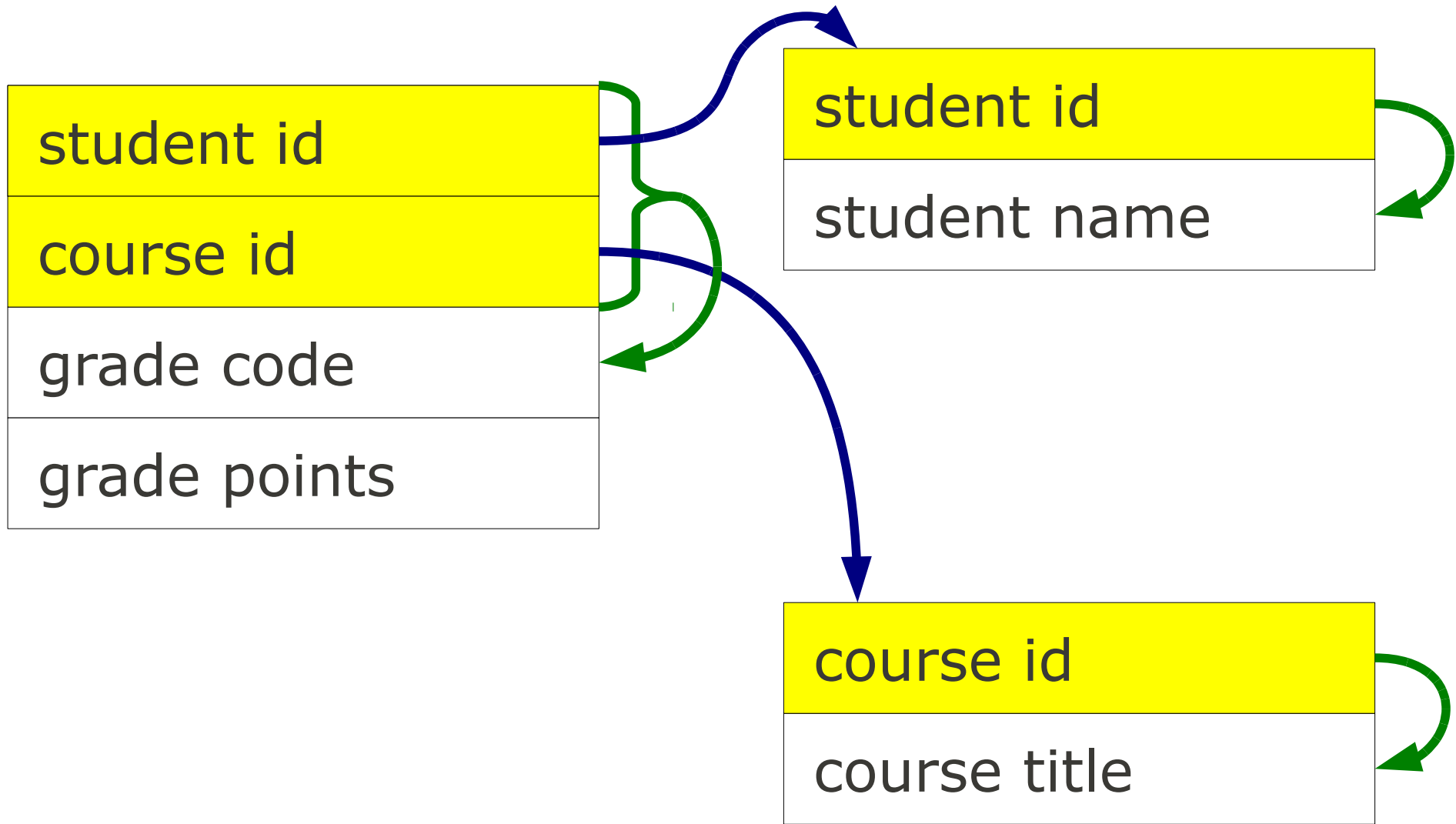
2NF + no transitive dependencies  
("nothing but the key")

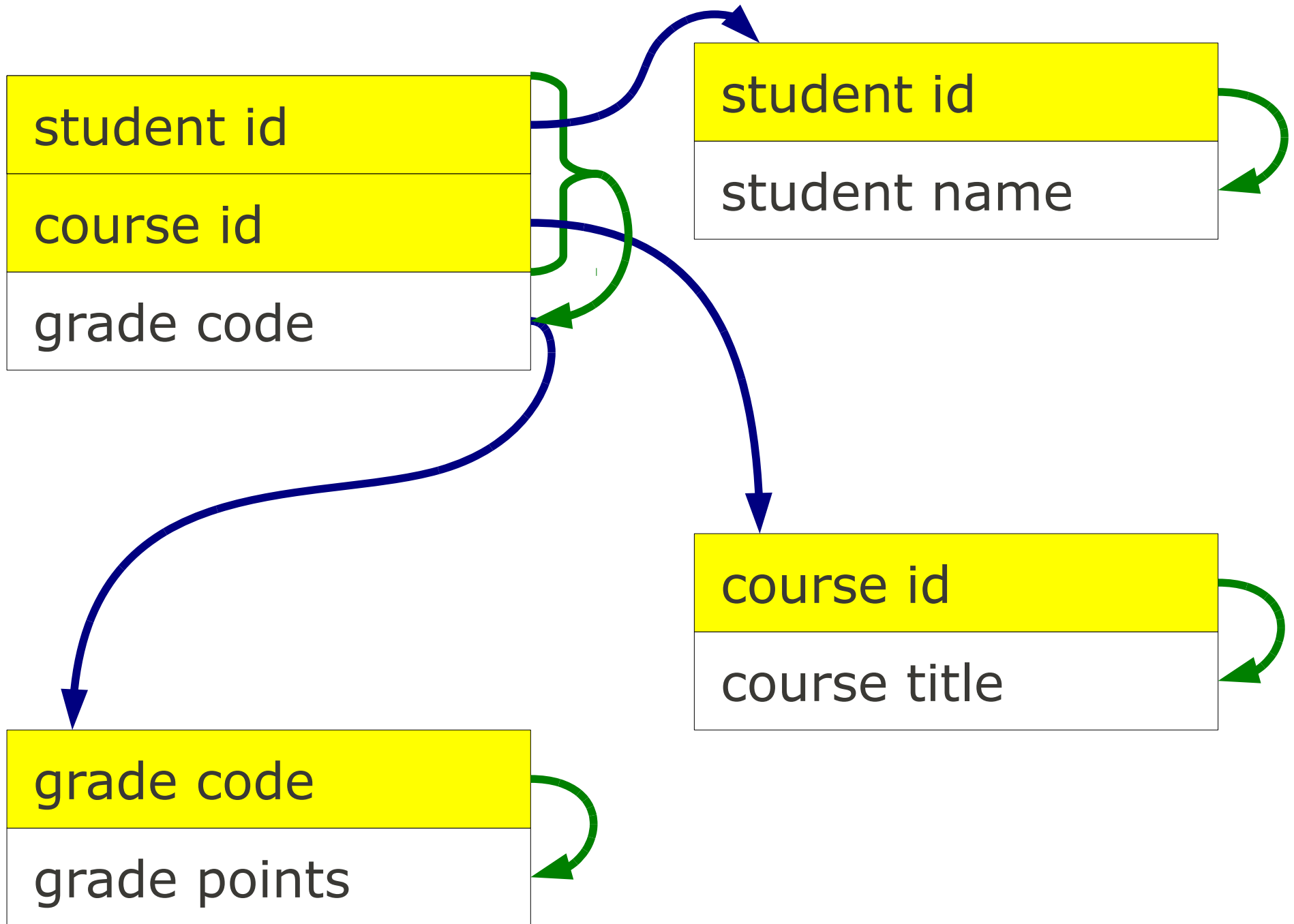
# Reaching 2NF and 3NF

Decomposition  
(break up tables into several)

student id
course id
course title
student name
grade code
grade points

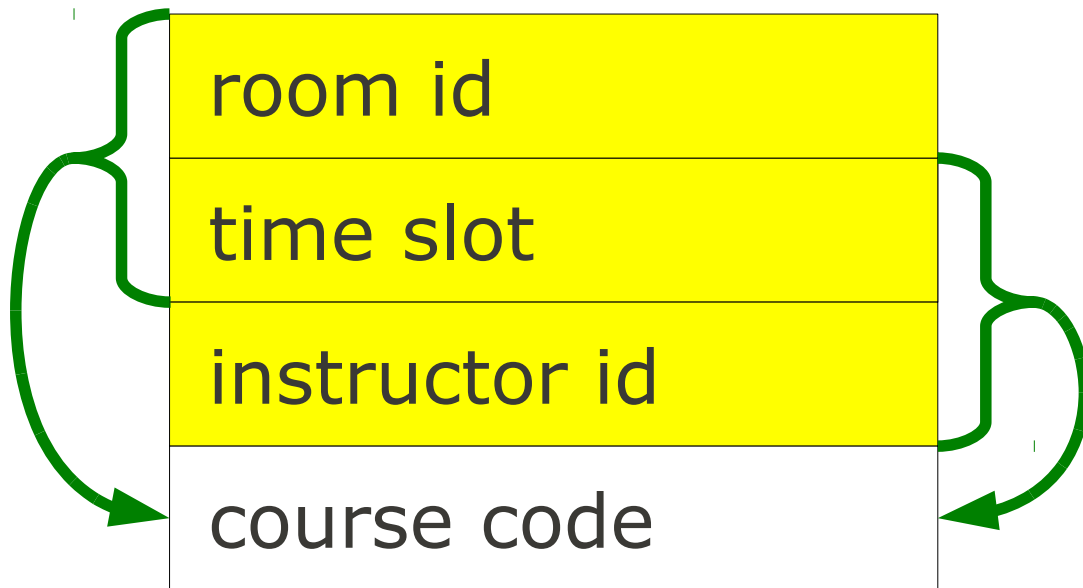




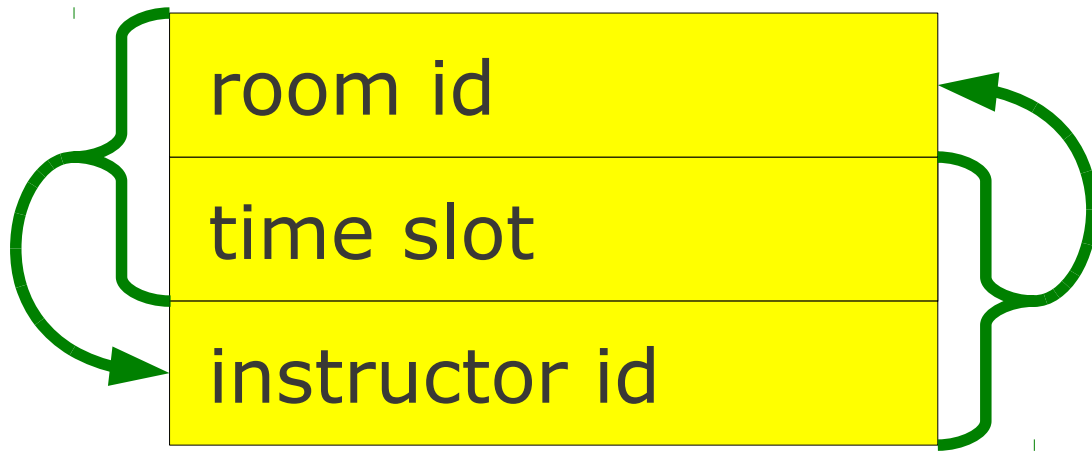


# BCNF

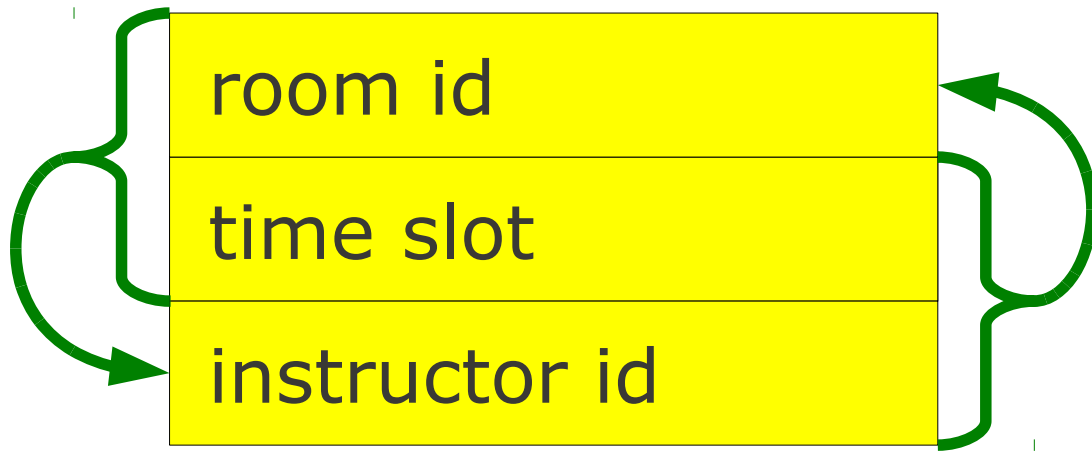
“all determinants must be candidate keys”



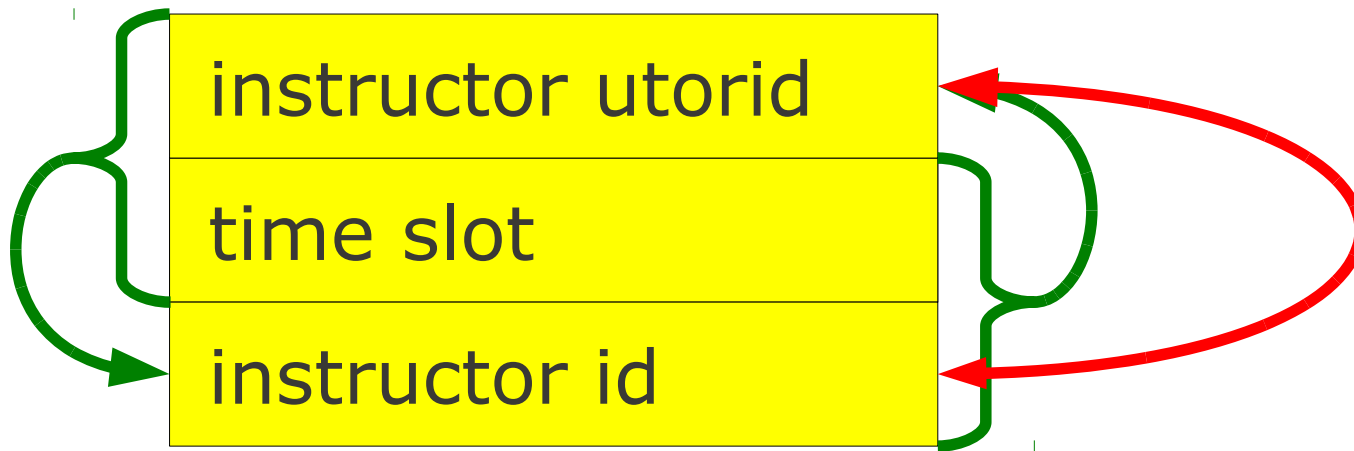
overlapping candidate keys



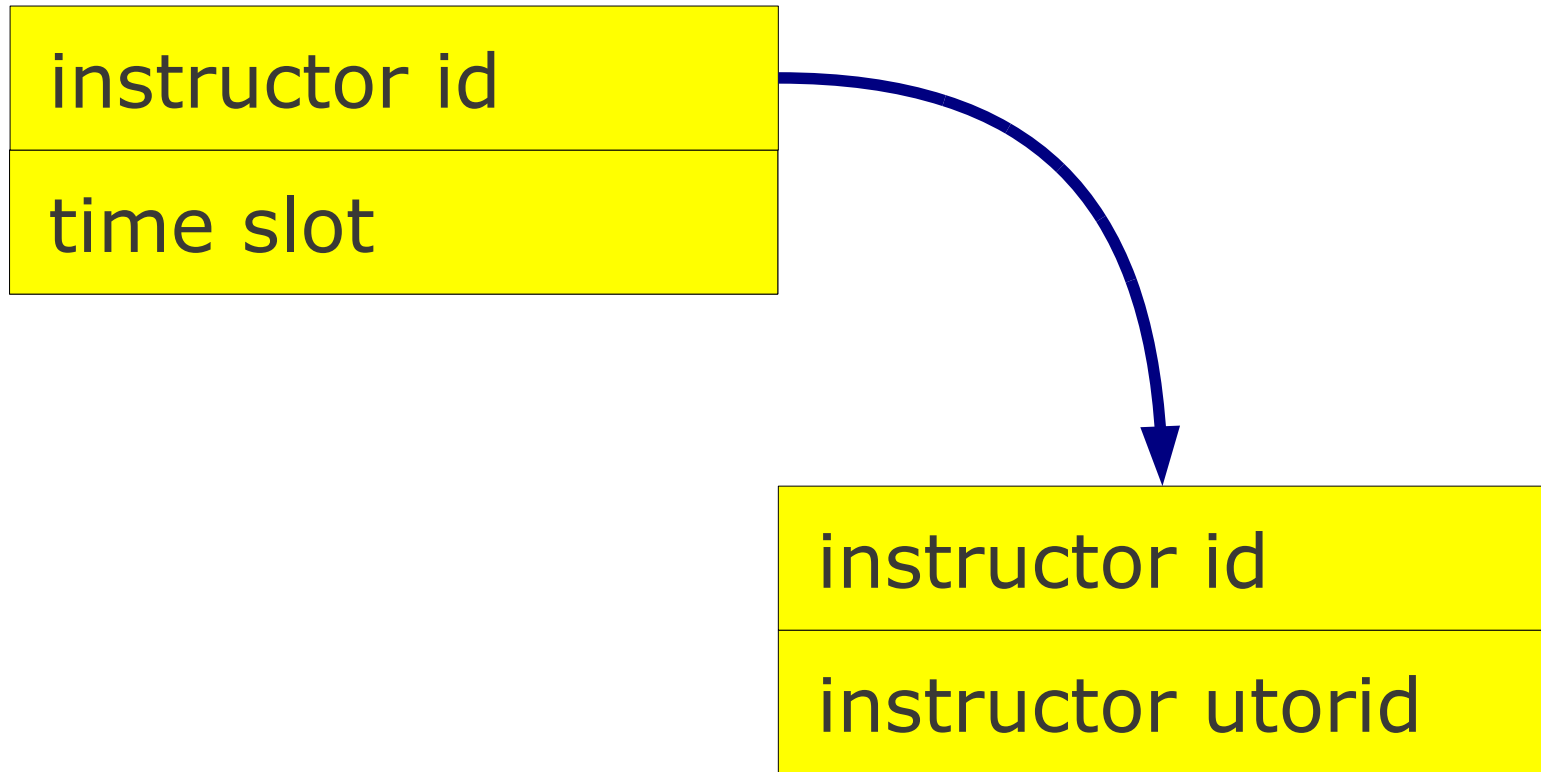
dependencies between keys



dependencies between keys

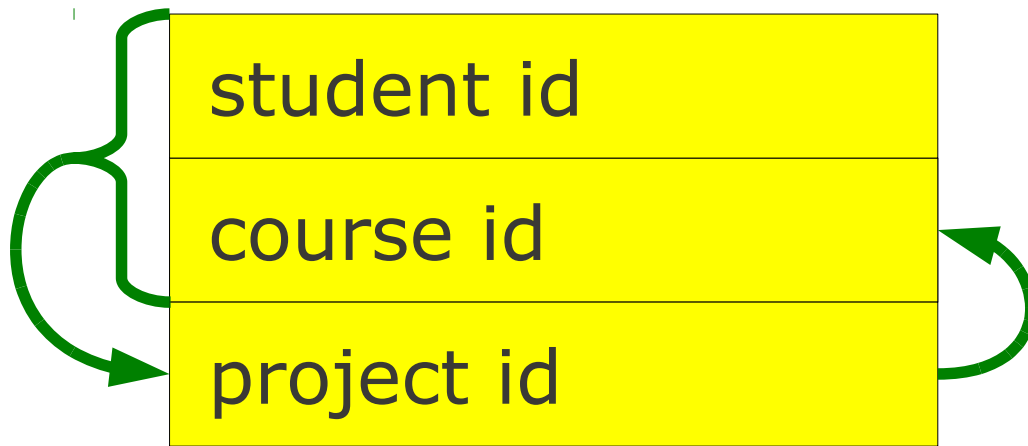


this is 3NF, but not BCNF  
("so help me Codd")



decompose it





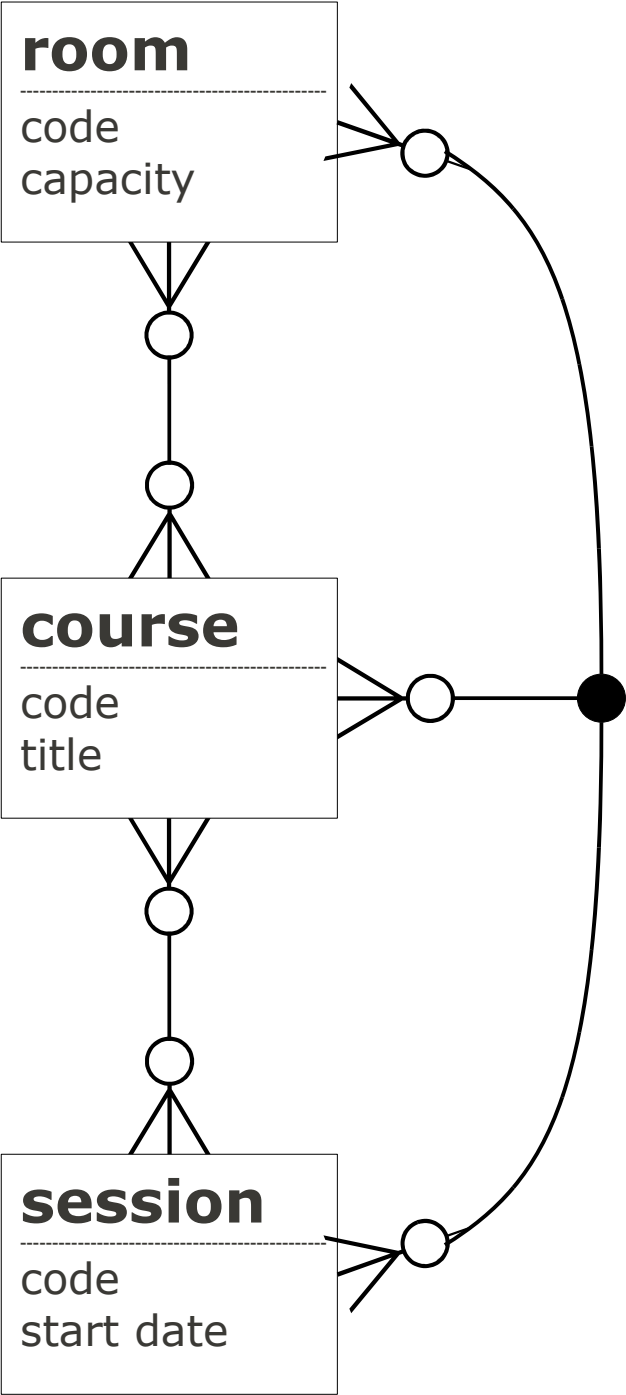
student, course  $\rightarrow$  project  
project  $\rightarrow$  course

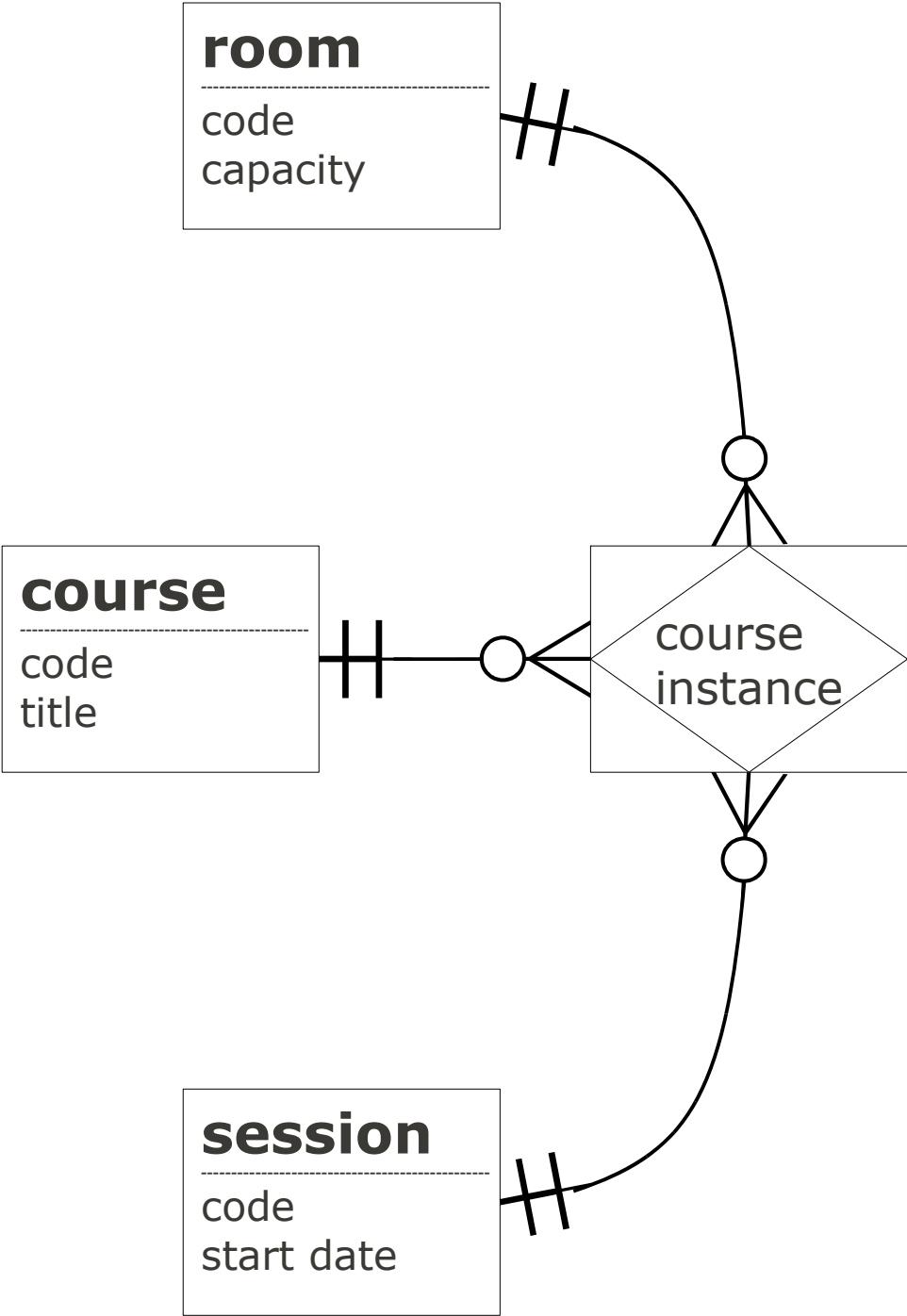
some relations cannot be  
decomposed into BCNF

("so help me Codd")

# 4NF

Mistaking binary relationships  
for ternary





<b>name</b>
student_id
course_id
internship_id

# 5NF

“We could break it up yet more”

Questions?