UNSW SYDNEY | Australia's Global University

COMP9311: Database Systems

ER to Relational Mapping

(textbook: chapters 9)

Term 3 2022

Week 2 ER to Relational Mapping

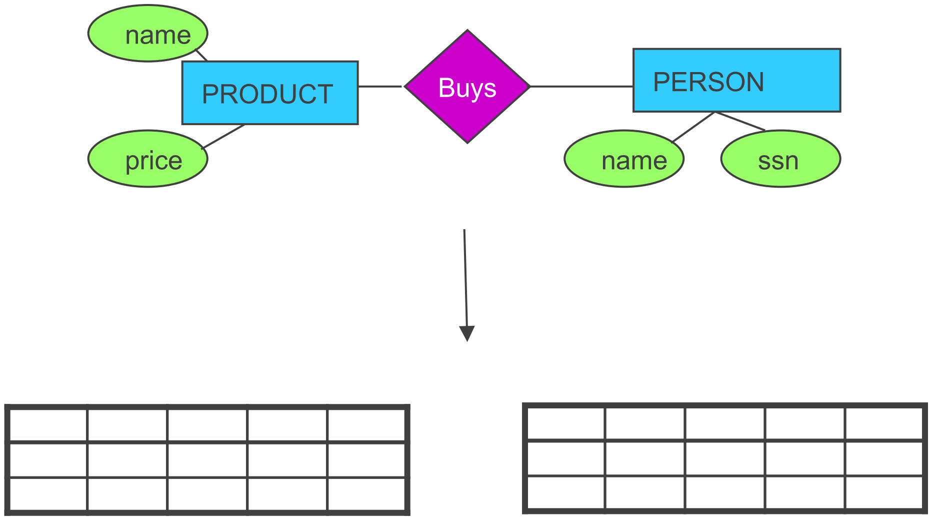
By Helen Paik, CSE UNSW

Disclaimer: the course materials are sourced from

- previous offerings of COMP9311 and COMP3311
- Prof. Werner Nutt on Introduction to Database Systems (http://www.inf.unibz.it/~nutt/Teaching/IDBs1011/)

Mapping ER Diagram to Relational Schema

Conceptual Model:



Relational Model:

We cannot store data in an ER model

➔ We translate our ER model into a relation schema so that a relational database can store the data accordingly

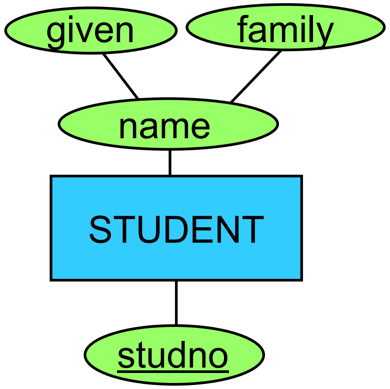
➔ What does “translation” mean?
➔ We have a set of “rules” applied to map ER to relations

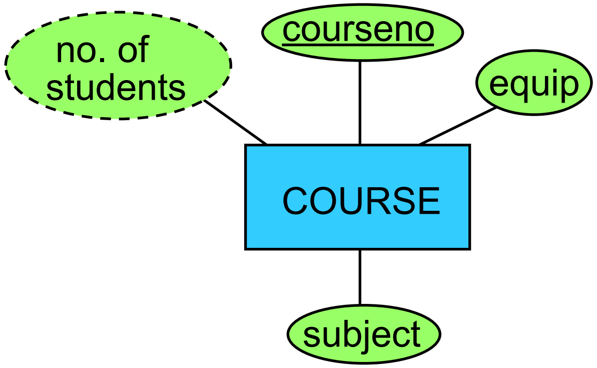
Ideally, the mapping between the models will not lose any information

Mapping Entity Types to Relations

General rules:

- for every entity type create a relation
- every atomic attribute of the entity type becomes a relation attribute
  - composite attributes: include all the atomic attributes
  - derived attributes are not included (but remember their derivation rules)
- Attributes of the entity key make up the primary key of the relation (if many, choose)





```
STUDENT (studno, given_name, family_name)
```

```
COURSE (courseno, subject, equip)
```

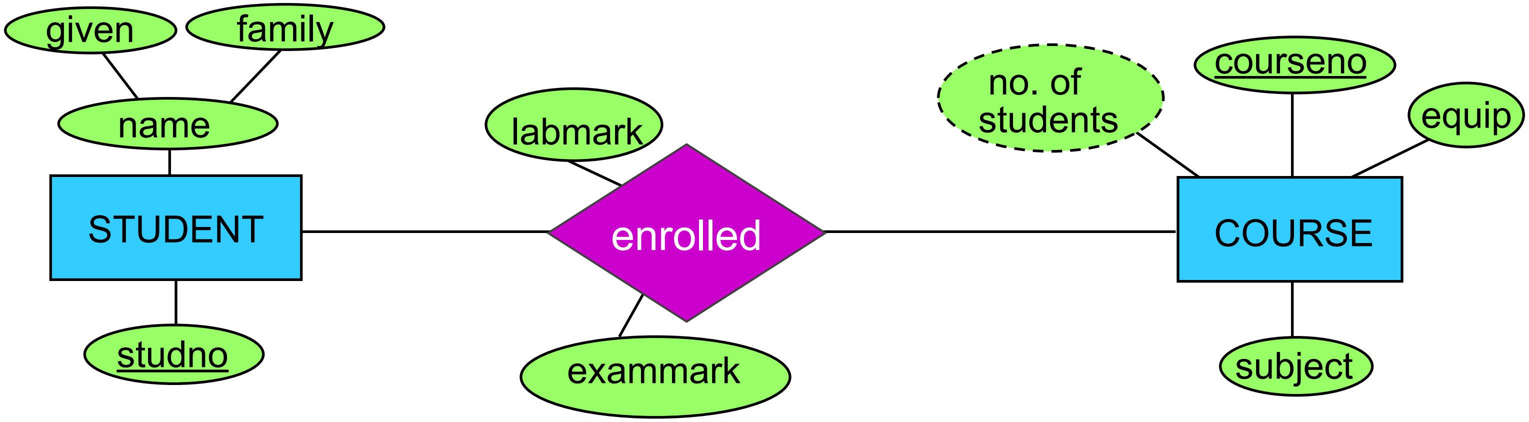
Mapping many:many Relationship Types

Rule: Create a relation with the following set of attributes:

$$\bigcup_{i=1}^{N\ \textit{(degree of relationship)}} \text{primary\_key}(E_i) \;\cup\; \{a_1,\ldots,a_M\}$$

primary keys of each entity type participating in the relationship

attributes of the relationship type (if any)

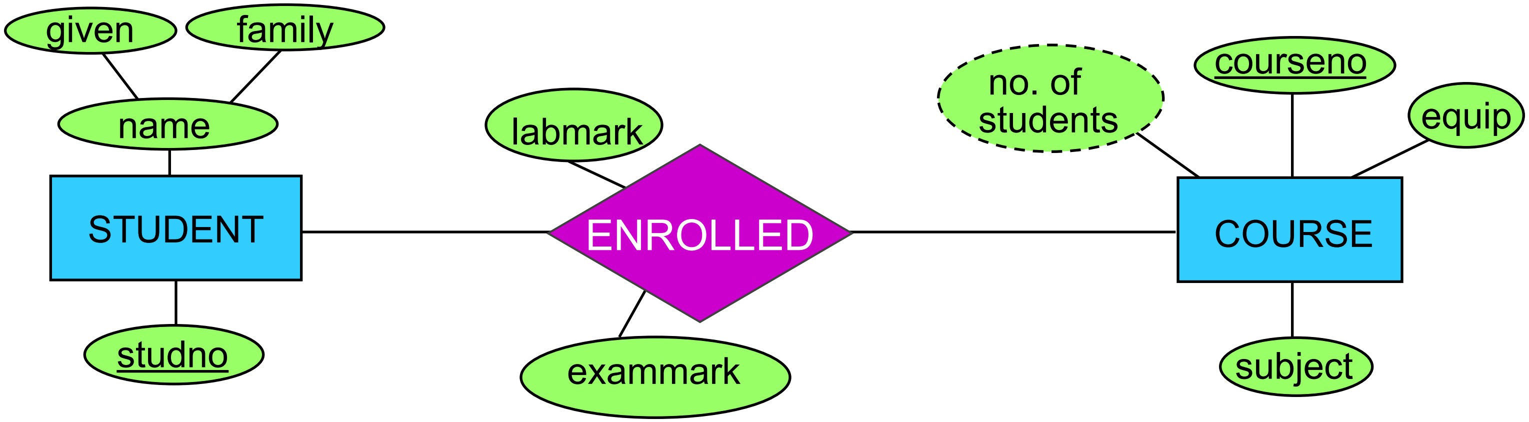


ENROL (studno, courseno, lab_mark, exam_mark)

STUDENT (studno, given_name, family_name)

COURSE (courseno, subject, equip)

Mapping many:many Relationship Types



To complete the mapping, let's remember the referential integrity as well …

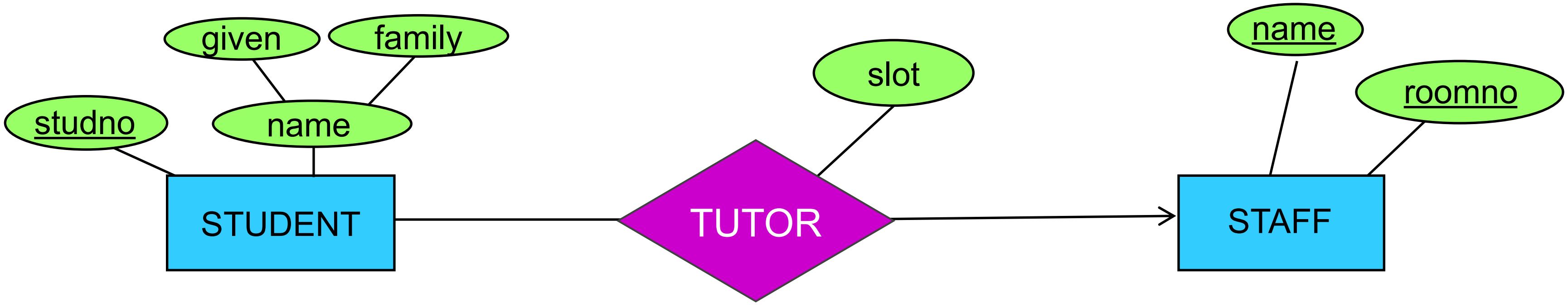
```
ENROL(studno, courseno, lab_mark, exam_mark)
    Foreign Key ENROL(studno) references STUDENT(studno)
    Foreign Key ENROL(courseno) references COURSE(courseno)
```

Mapping Many:One Relationship Types

Idea: "Post the primary key to your many-side partner"

Rule: given E1 at the 'many' end of relationship and E2 at the 'one' end of the relationship, add information of E2 to the relation for E1

The primary key of the entity at the 'one' end (the determined entity) becomes a foreign key in the entity at the 'many' end (the determining entity). Include any relationship attributes with the foreign key entity



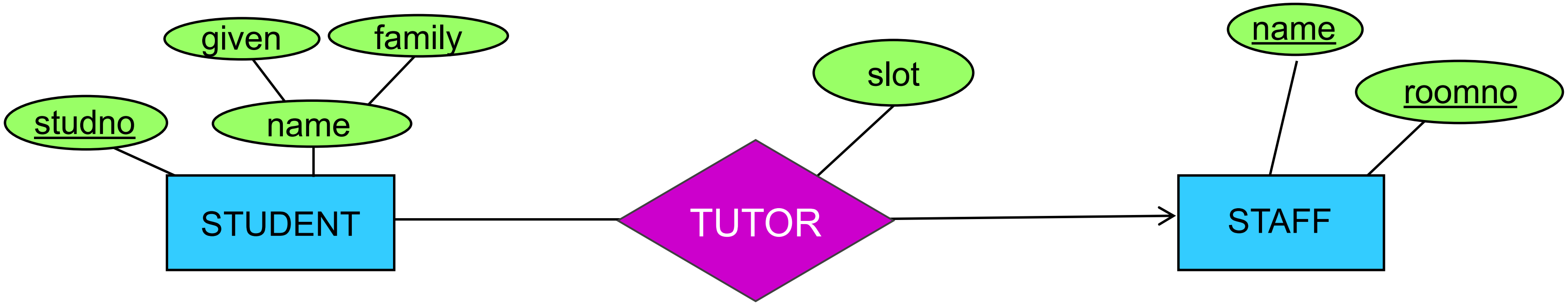
$$E1\ \ U\ \ \text{primary\_key}(E2)\ \ U\ \ \{a_1,\ldots,a_n\}$$

relation for entity E1

primary key for E2, *is now a* foreign key *to* E2

attributes on the relationship (if any)

Mapping Many:one Relationship Types



E1 U primary_key(E2) U $\{a_1,\ldots,a_n\}$

The relation

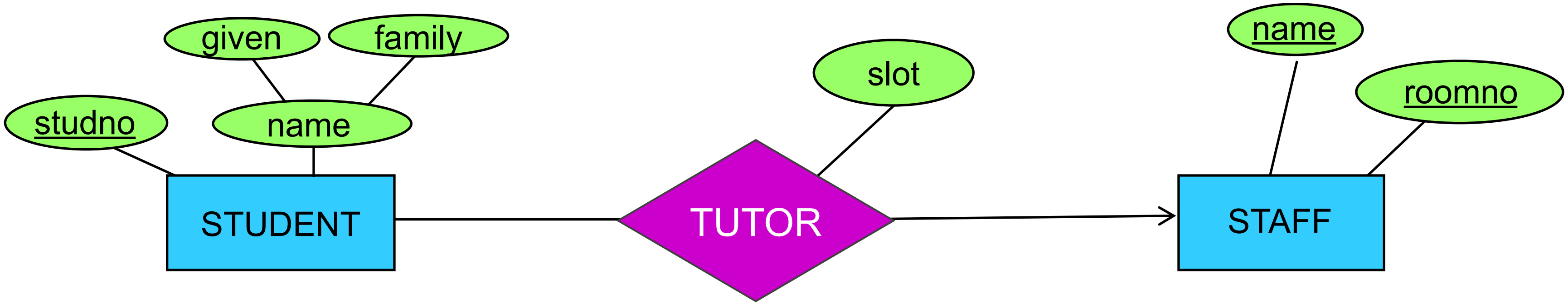
```
   STUDENT(studno, givenname, familyname)
```

is extended to

```
   STUDENT(studno, givenname, familyname, tutor, roomno, slot)
     Foreign Key STUDENT(tutor,roomno) references STAFF(name,roomno)
```

(don’t forget the constraint)

Mapping many:one Relationship Types



STUDENT

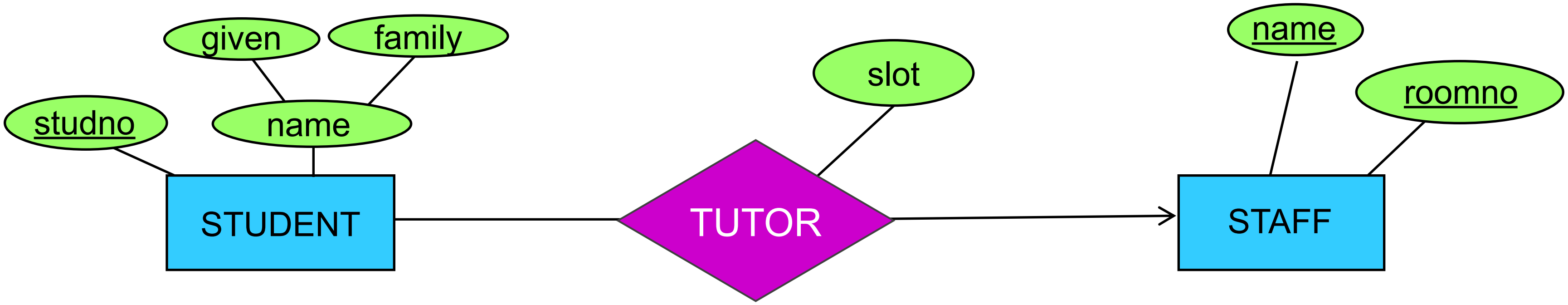
studno	given	family	tutor	roomno	slot
s1	fred	jones	bush	2.26	12B
s2	mary	brown	kahn	IT206	12B
s3	sue	smith	goble	2.82	10A
s4	fred	bloggs	goble	2.82	11A
s5	peter	jones	zobel	2.34	13B
s6	jill	peters	kahn	IT206	12A

STAFF

name	roomno
kahn	IT206
bush	2.26
goble	2.82
zobel	2.34
watson	IT212
woods	IT204
capon	A14
lindsey	2.10
barringer	2.125

The relation STUDENT captures that there is one tutor for a student

Mapping Many:one Relationship Types



Another Idea: If

- the relationship type is *optional* to both entity types, and
- an instance of the relationship is *rare*, and
- there are *many attributes* on the relationship then…

… create a new relation with the following set of attributes:

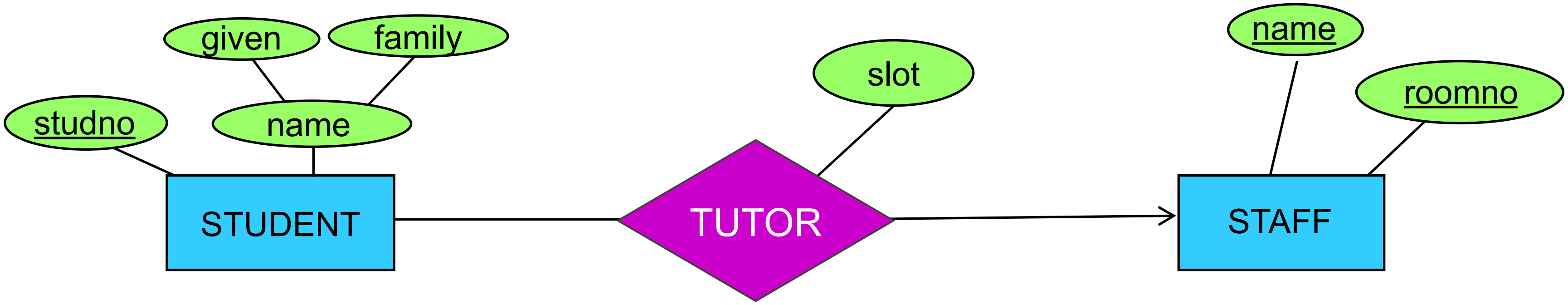
$$\text{primary\_key}(E_1) \;\cup\; \text{primary\_key}(E_2) \;\cup\; \{a_1,\ldots,a_m\}$$

primary key for E1, *is now a* foreign key *to* E1; *also the PK for this relation*

primary key for E2, *is now a* foreign key *to* E2

Any attributes on the relationship type

Mapping M:1 (alternative option)



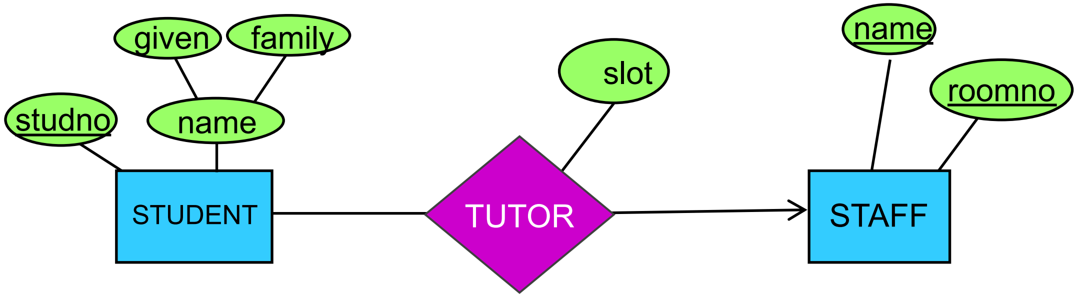
```
TUTOR(studno, staffname, rommno, slot)

Foreign key TUTOR(studno) references STUDENT(studno)

Foreign key TUTOR(staffname, roomno) references STAFF(name, roomno)
```

Note: primary key for E1, *is now a* foreign key *to* E1; *also the PK for this relation (i.e., A student has one tutor, so only single tuple of a particular studno value should appear in this relation)*

Mapping M:1 (alternative option)

STUDENT

studno	given	family
s1	fred	jones
s2	mary	brown
s3	sue	smith
s4	fred	bloggs
s5	peter	jones
s6	jill	peters



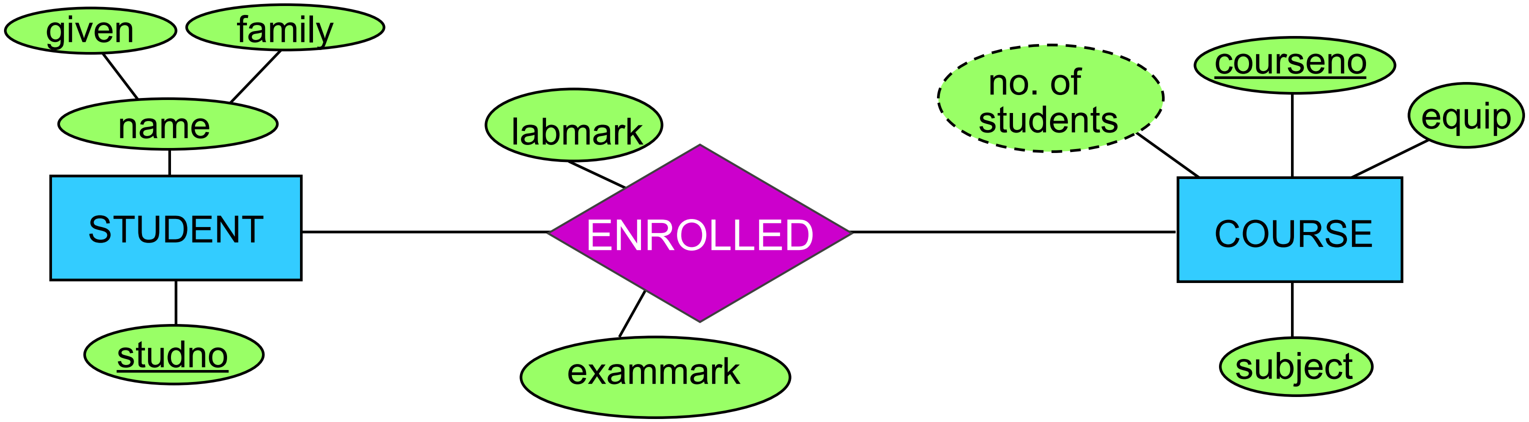
TUTOR

studno	tutor	roomno	slot
s1	bush	2.26	12B
s2	kahn	IT206	12B
s3	goble	2.82	10A
s4	goble	2.82	11A
s5	zobel	2.34	13B
s6	kahn	IT206	12A

STAFF

name	roomno
kahn	IT206
bush	2.26
goble	2.82
zobel	2.34
watson	IT212
woods	IT204
capon	A14
lindsey	2.10
barringer	2.125

Quick comparison to M:M mapping



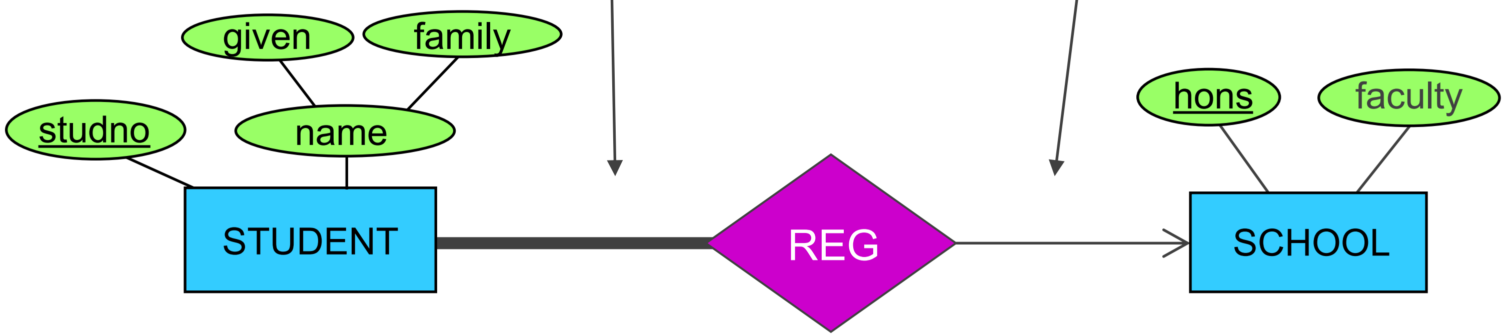
```
ENROL(studno, courseno, lab_mark, exam_mark)
    Foreign Key ENROL(studno) references STUDENT(studno)
    Foreign Key ENROL(courseno) references COURSE(courseno)
```

Note: ENROL takes the PK from each relation and makes a combined PK for itself - i.e., many instances of a particular studno, and many instances of a particular courseno would appear, so only a combination of the two would make a tuple unique in ENROL.

Optional Participation of the Determined Entity ('one end')

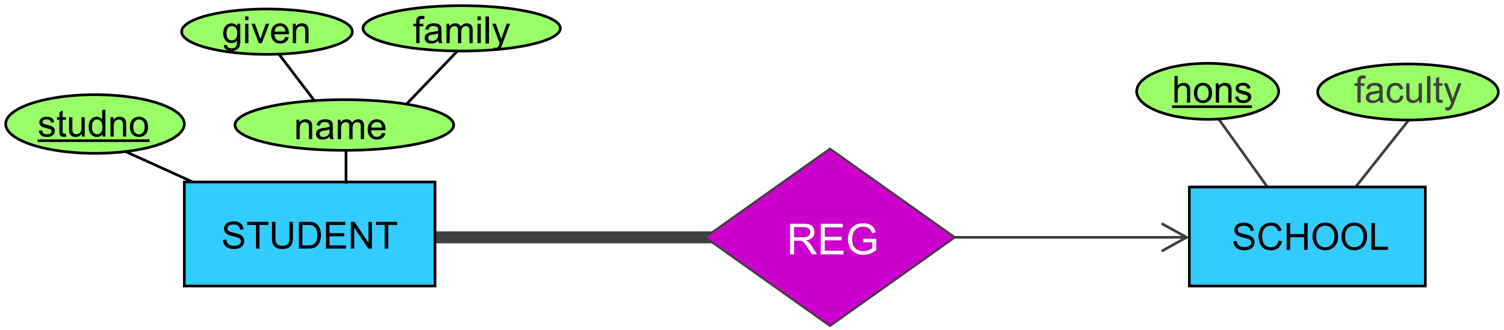
A student entity instance must *participate in a relationship instance of* REG

A school entity instance need not *participate in a relationship instance of* REG



SCHOOL (hons, faculty)

STUDENT (studno, givenname, familyname, hons(??))



STUDENT

studno	given	family	hons
s1	fred	jones	ca
s2	mary	brown	cis
s3	sue	smith	cs
s4	fred	bloggs	ca
s5	peter	jones	cs
s6	jill	peters	ca

"hons" cannot be NULL because it is mandatory for a student to be registered for a school

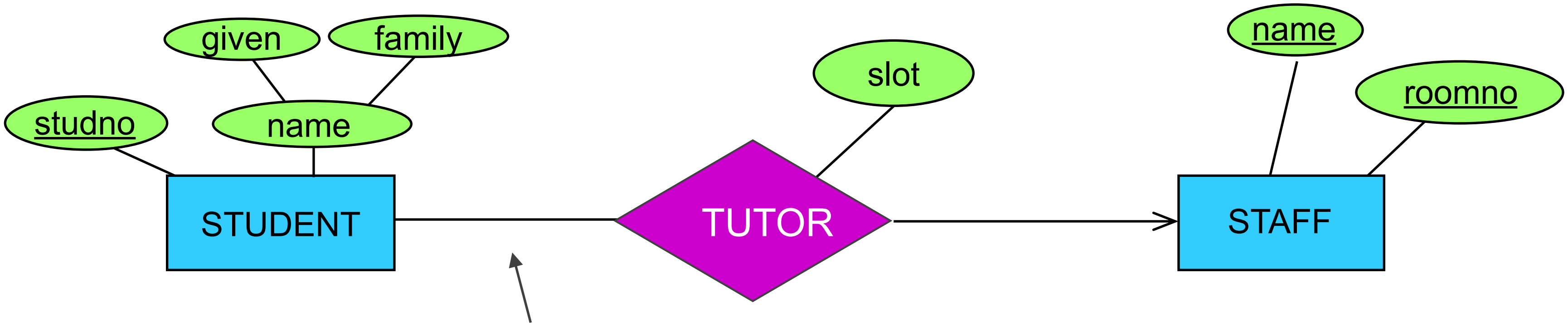
➔ "not null" constraint

SCHOOL

hons	faculty
ac	accountancy
is	information systems
cs	computer science
ce	computer science
mi	medicine
ma	mathematics

No student is registered for "mi", so "mi" doesn't occur as a foreign key value in STUDENT → This is no problem, i.e., the participation from SCHOOL is optional!

Optional Participation of the Determinant Entity (‘many end’)



A student entity instance
need not *participate in a relationship instance of* TUTOR

OPTION 1:

```
STUDENT (studno, givenname, familyname, tutor, roomno, slot)
STAFF(name, roomno)
 add FK constraint … and they can be null
```

OPTION 2:

```
STUDENT(studno, givenname, familyname)
STAFF(name, roomno)
TUTOR(studno, tutor, roomno, slot)
```

Optional Participation of the Determinant Entity (‘Many end’)

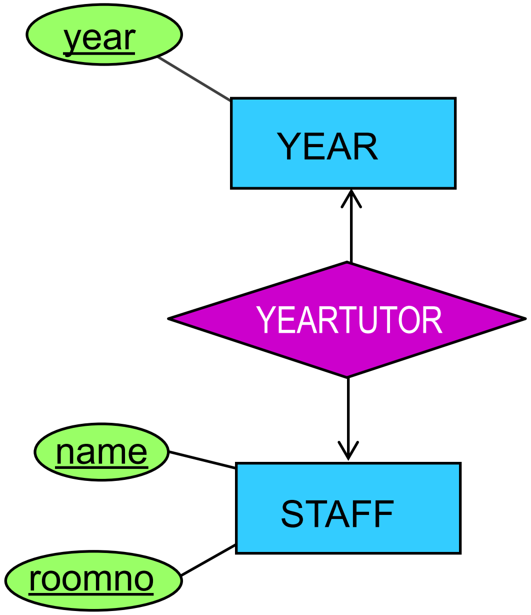
STUDENT

studno	given	family	tutor	roomno	slot
s1	fred	jones	bush	2.26	12B
s2	mary	brown	kahn	IT206	12B
s3	sue	smith	goble	2.82	10A
s4	fred	bloggs	goble	2.82	11A
s5	peter	jones	NULL	NULL	NULL
s6	jill	peters	kahn	IT206	12A

STAFF

name	roomno
kahn	IT206
bush	2.26
goble	2.82
zobel	2.34
watson	IT212
woods	IT204
capon	A14
lindsey	2.10
barringer	2.125

Mapping One:one Relationship Types



OPTION 1: Post the primary key of one of the entity types into the other entity type as a foreign key, including any relationship attributes with it (i.e., as shown in YEAR relation)

YEAR

year	yeartutor	roomno
1	zobel	2.34
2	bush	2.26
3	capon	A14

OPTION 2: Merge the entity types together (but only when the participation from both sides are total, otherwise many NULLs) , as shown in STAFF relation here …

STAFF

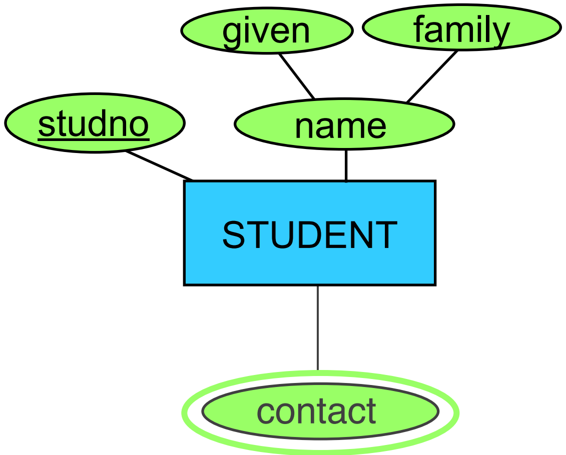
name	roomno	year
kahn	IT206	NULL
bush	2.26	2
goble	2.82	NULL
zobel	2.34	1
watson	IT212	NULL
woods	IT204	NULL
capon	A14	3
lindsey	2.10	NULL
barringer	2.125	NULL

Multi-Valued Attributes

For each multi-valued attribute of E_i, create a relation with the attributes

primary_key(E_i) U multi-valued attribute

The new relation’s primary key comprises all attributes



STUDENT

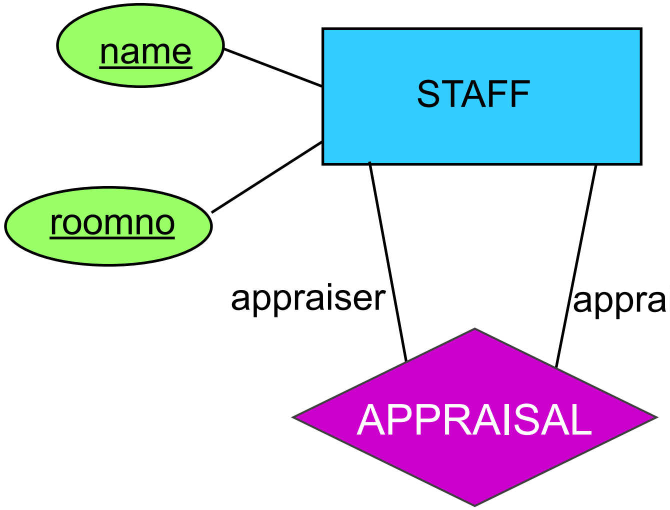
studno	given	family
s1	fred	jones
s2	mary	brown

STUDENT_CONTACT

studno	contact
s1	Mr. Jones
s1	Mrs Jones
s2	Bill Brown
s2	Mrs Jones
s2	Billy-Jo Woods

Mapping Roles and Recursive Relationships

How can the entity STAFF appear in both of its roles ?

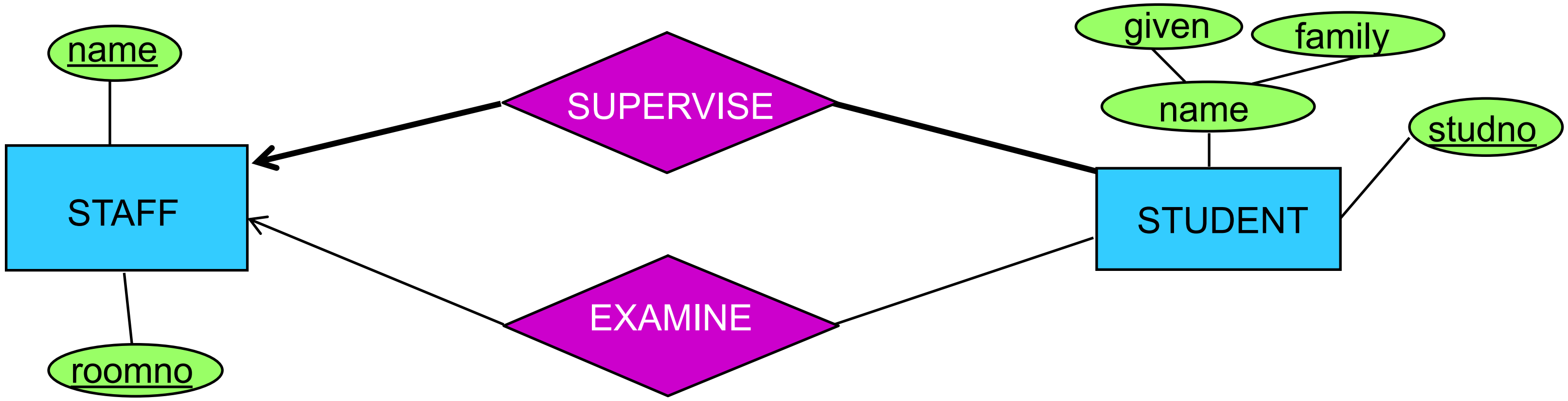


`APPRAISAL (name, roomno, appraiser, app_roomno)`

Multiple Relationships between Entity Types

Treat each relationship type separately

Represent distinct relationships by different foreign keys drawing on the same relation

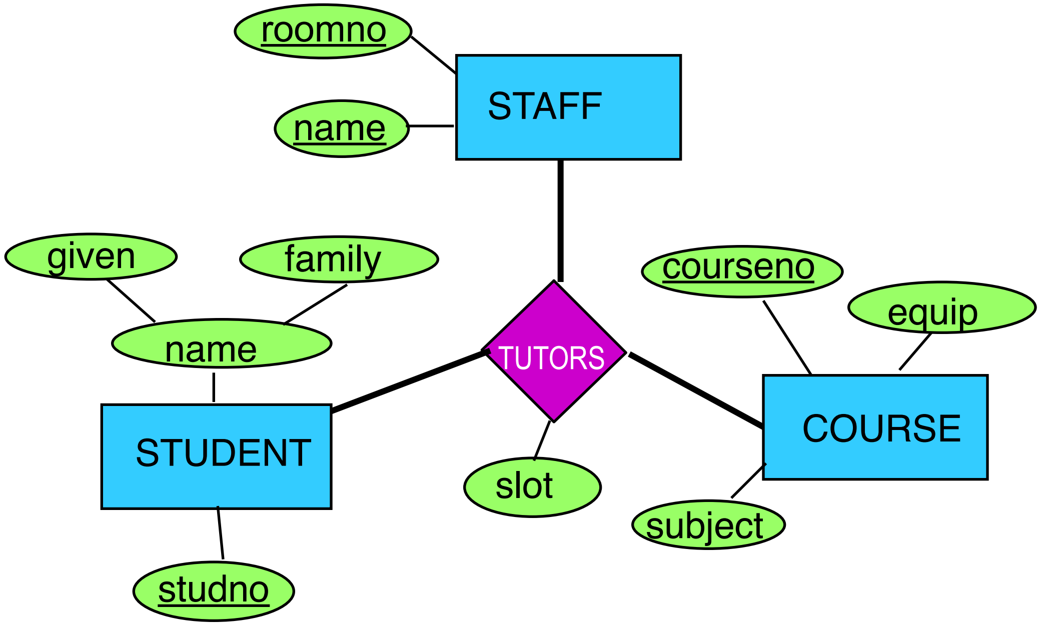


So starting with the entities … Decide if you want to add foreign keys or new relations for each relationship type.

STAFF(<u>name</u>, <u>roomno</u>)

STUDENT(<u>studno</u>, given, family)

Non-binary Relationship



```
COURSE(courseno, subject, equip)
STUDENT(studno, givenname, familyname)
STAFF(staffname, roomno)
TUTORS(courseno, studno, staffname, roomno, slot)
```

Mapping Weak Entities to Relations

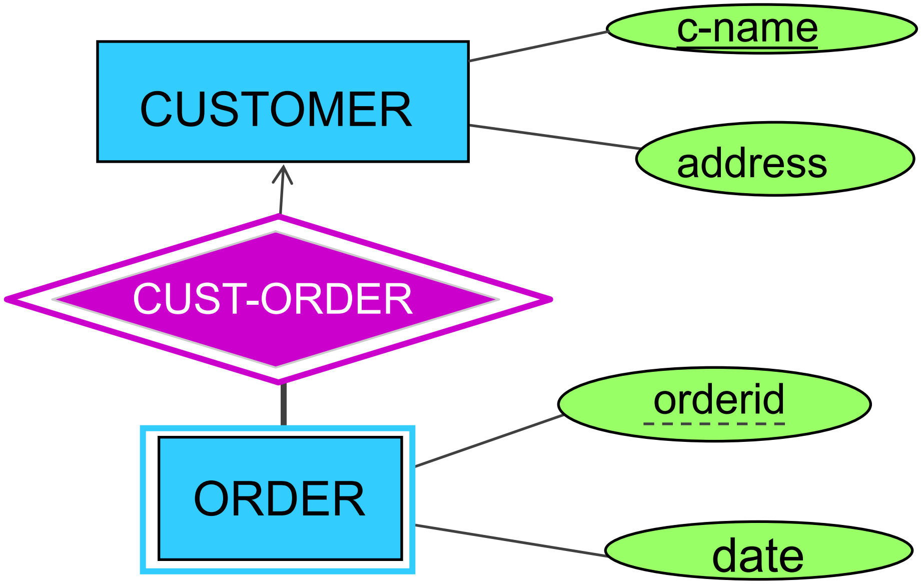
Create a relation with the following attributes:

$$\text{primary\_key}(E_0) \;\cup\; \bigcup_{i=1}^{n} \text{discriminator}(E_i) \;\cup\; \{a_1,\ldots,a_n\}$$

Primary key of identifying strong entity type

Discriminators of identifying weak entity types

Attributes of the weak entity type

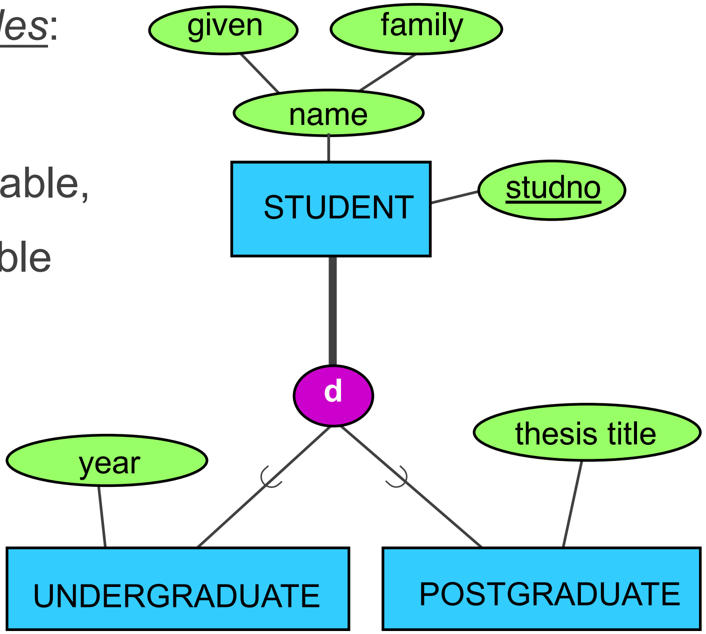


CUST_ORDER (c_name, order_id, date)

The discriminator and primary key from the strong entity become the primary key of this new relation.

Translating of Hierarchies: Options

Three different approaches to *mapping subclasses to tables*:



ER style

- superclass and subclasses entity become a separate table,
- containing attributes of subclass + FK to superclass table

object-oriented

- only subclasses entity become a separate table,
- inheriting all attributes from all superclasses

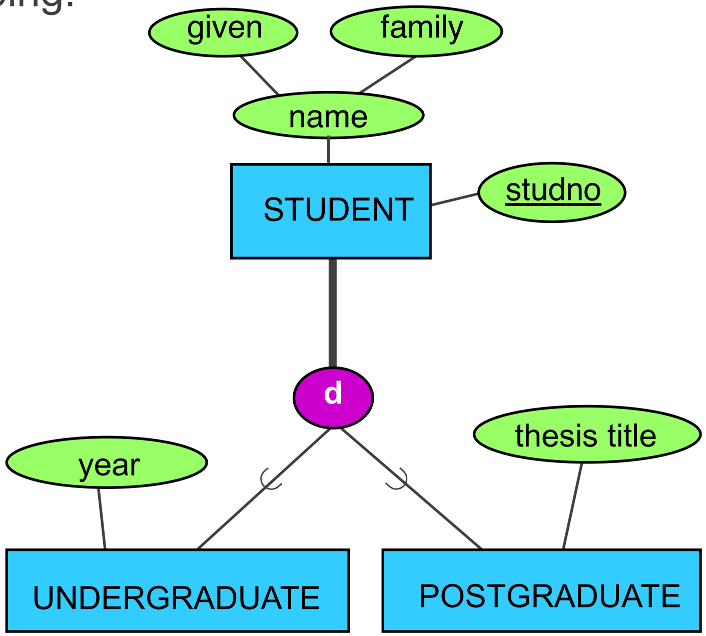
single table with nulls (all-in-one)

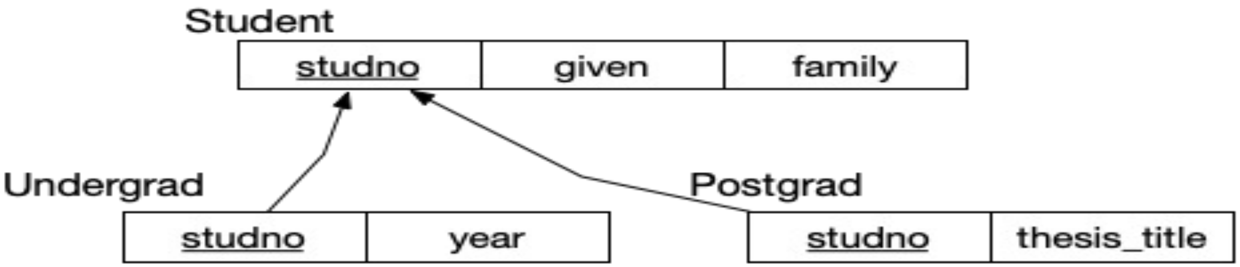
- whole class hierarchy becomes one table,
- containing all attributes of all subclasses (null, if unused)
- a special attribute “type/class” can be used to indicate which subclass

Which mapping is best depends on how you intend to use the data (i.e., your requirements)

Translating of Hierarchies: Options

An example of ER Style mapping:





Student

studno	given	family

Undergrad

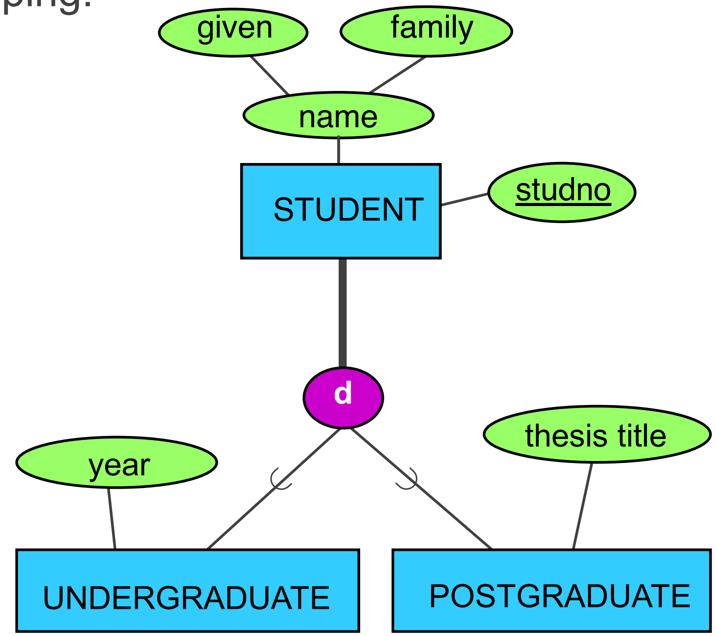
studno	year

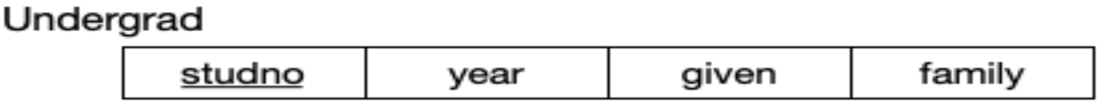
Postgrad

studno	thesis_title

Translating of Hierarchies: Options

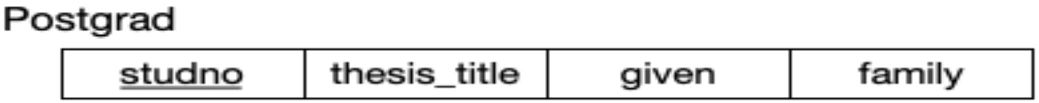
An example of OO Style mapping:





Undergrad

studno	year	given	family

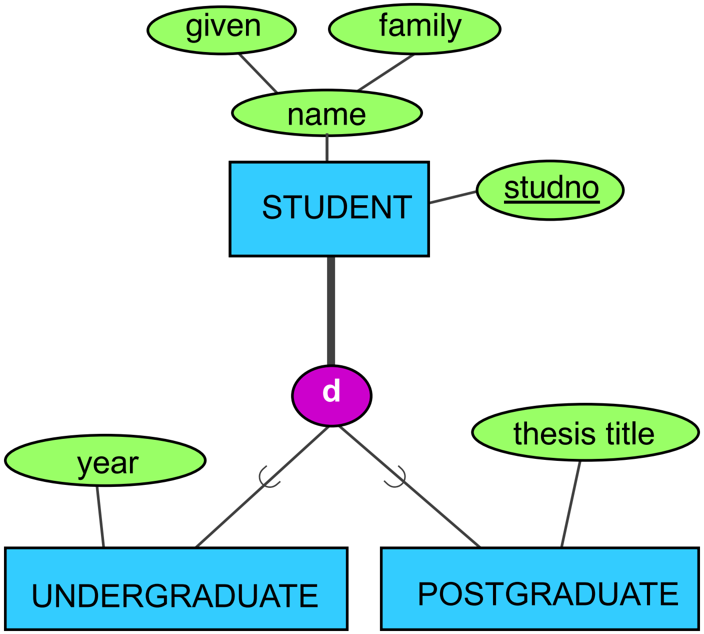


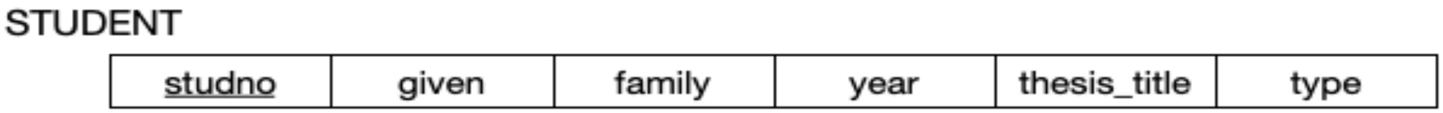
Postgrad

studno	thesis_title	given	family

Translating of Hierarchies: Options

An example of One table Style mapping:





STUDENT

studno	given	family	year	thesis_title	type