CSC7072: Databases, fall 2015

Dr. Kim Bauters

Converting ER to Relation Schemas

Converting ER to Relation Models

adding tables using SQL

don't forget:

```
CREATE TABLE name (
    {attribute_name  attribute_type  attribute_constraint}
    [{constraints}]
)
```

typical constraints:

- PRIMARY KEY({attribute})
- FOREIGN KEY({attribute}) REFERENCES (table)
- NOT NULL

where {argument} denotes you need to have at least one, and
where [argument] denotes a part that is optional and can be omitted

Converting ER to Relation Models

creating databases

what good are ER models?

they can easily be converted into relation schemas!
corresponds to a *reasonably* well-designed database

basic idea:

for each entity set/relationship set there is a unique schema
assigned the name of the corresponding entity/relationship set

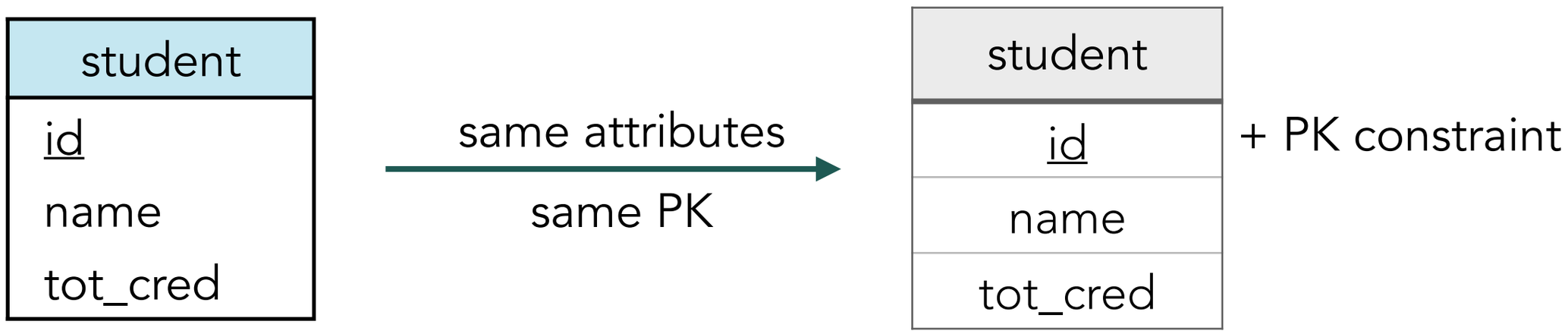
each schema has a number of columns (typically
corresponding to attributes) which each have a unique name

Converting ER to Relation Models

strong entity, simple attributes

strong entity set with simple attributes

trivial conversion!

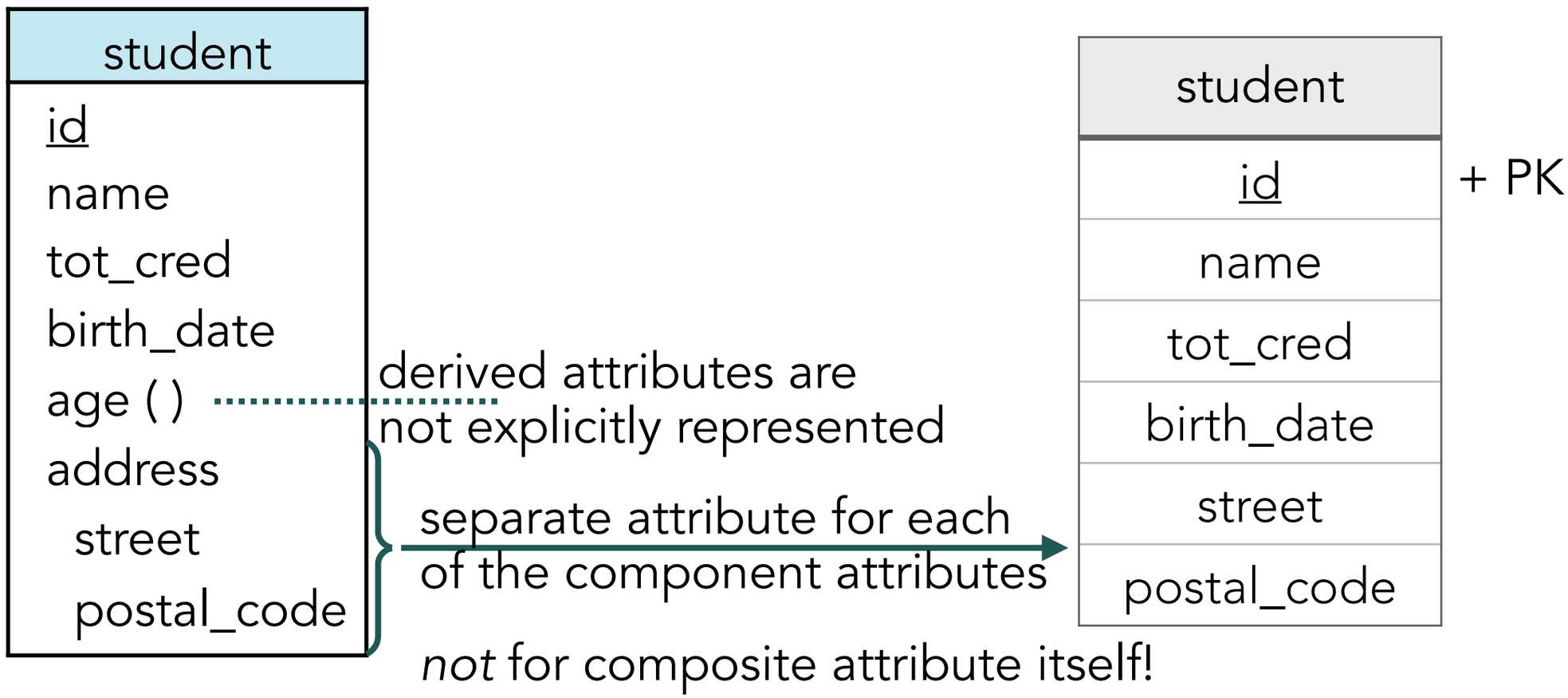


Converting ER to Relation Models

strong entity, composite and derived attributes

strong entity set with non-simple attributes

composite and derived attributes:

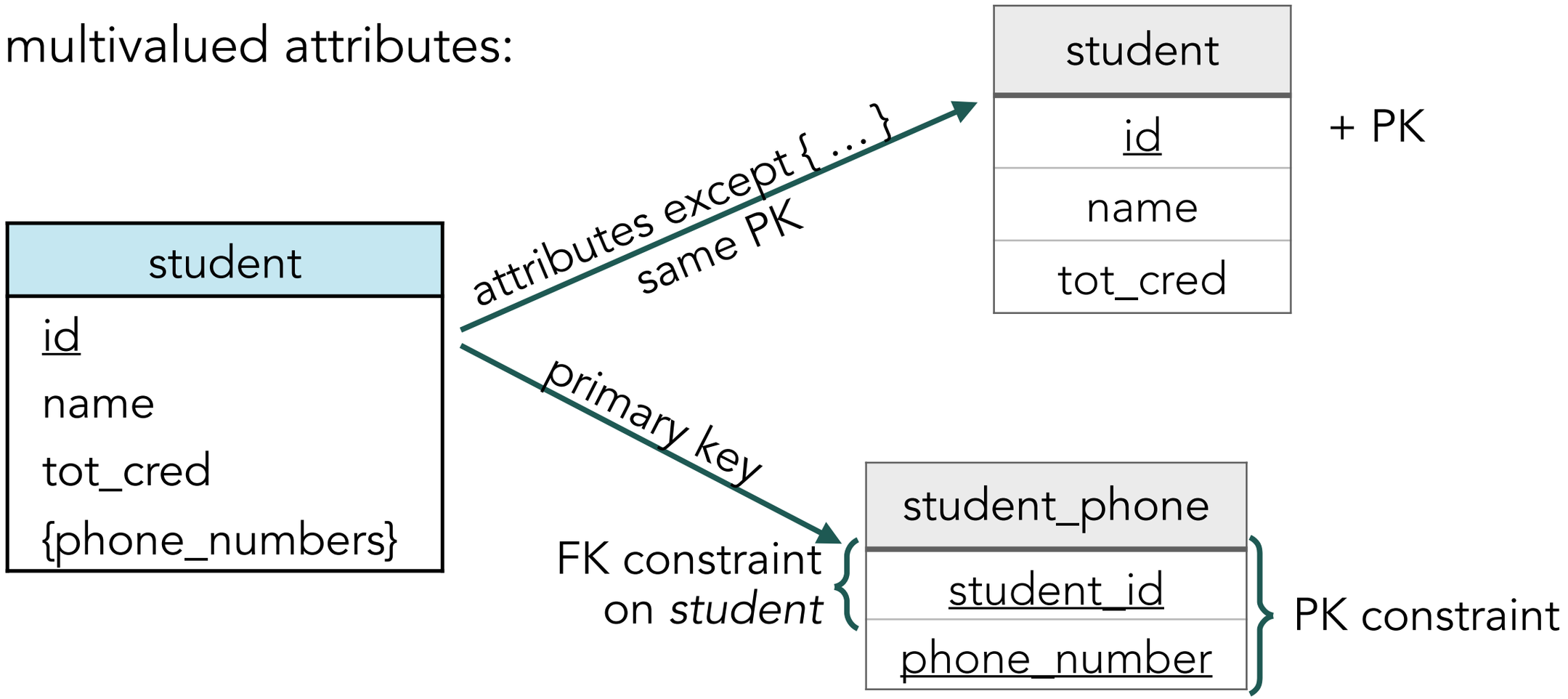


Converting ER to Relation Models

strong entity, multi-valued attributes

strong entity set with non-simple attributes

multivalued attributes:



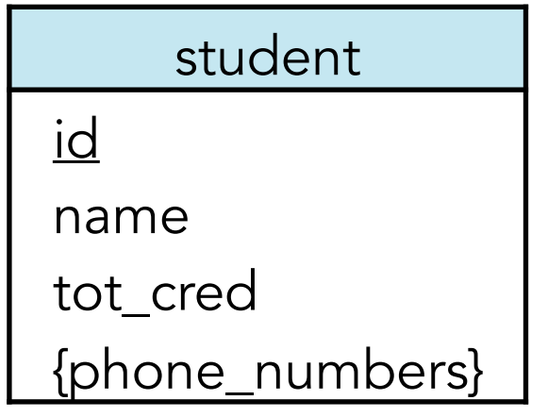
careful: PK may consist of multiple attributes, then all are adopted!

Converting ER to Relation Models

strong entity, multi-valued attributes

strong entity set with non-simple attributes

multivalued attributes:




```
CREATE TABLE student (
    id              VARCHAR(5),
    name            VARCHAR(20),
    tot_cred        NUMERIC(3,0),
   PRIMARY KEY (id)
)
```

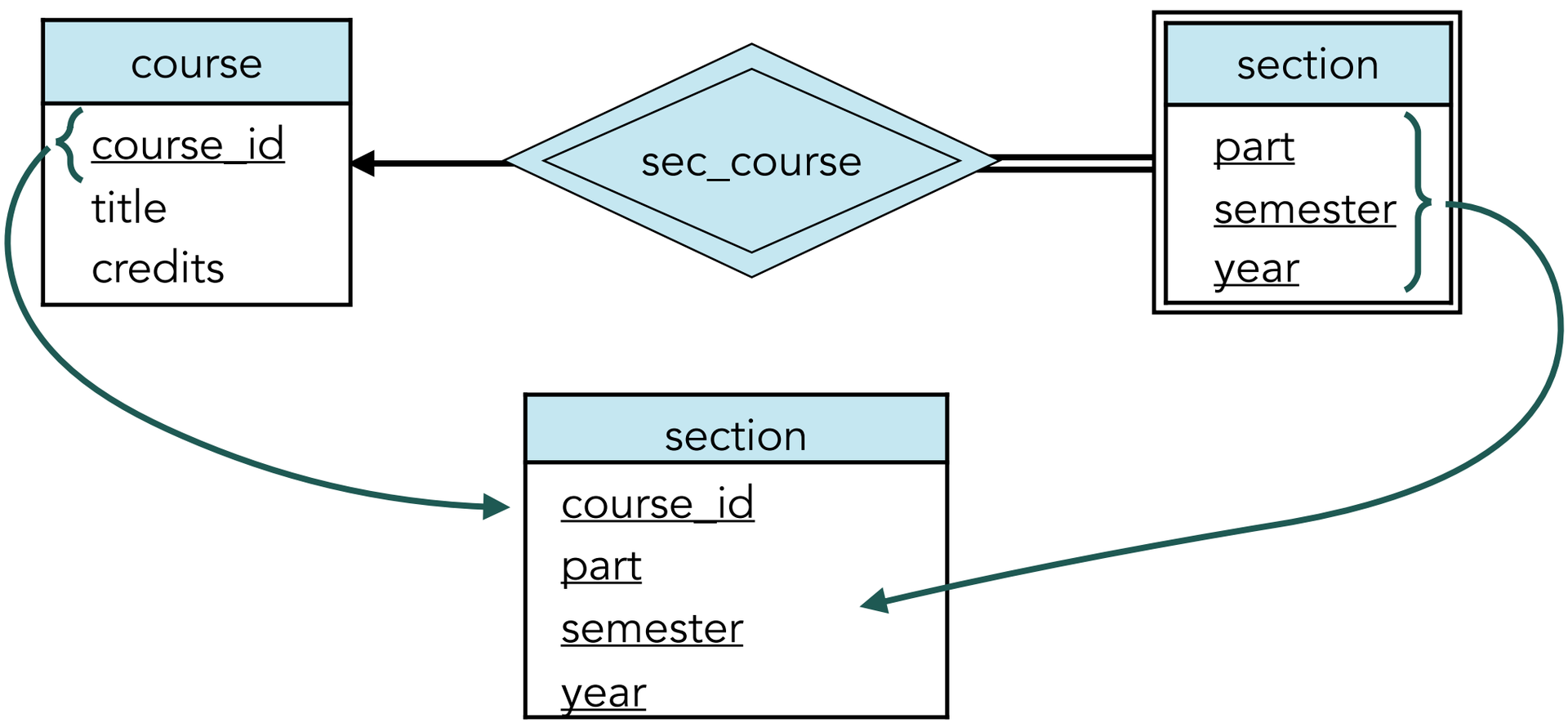
```
CREATE TABLE student_phone (
    student_id          VARCHAR(5),
    phone_number        VARCHAR(15),
   PRIMARY KEY (student_id, phone_number),
   FOREIGN KEY (student_id)
      REFERENCES student (id)
      ON DELETE SET NULL
)
```

Converting ER to Relation Models

weak entity sets

weak entity sets

becomes a table that includes (a) column(s) for the primary key of the identifying strong entity set



Converting ER to Relation Models

relationships: basics

relationships

follows a basic idea irrespective of cardinality:

- add attributes for the primary keys of both relations
- add any descriptive attributes of the relationship set

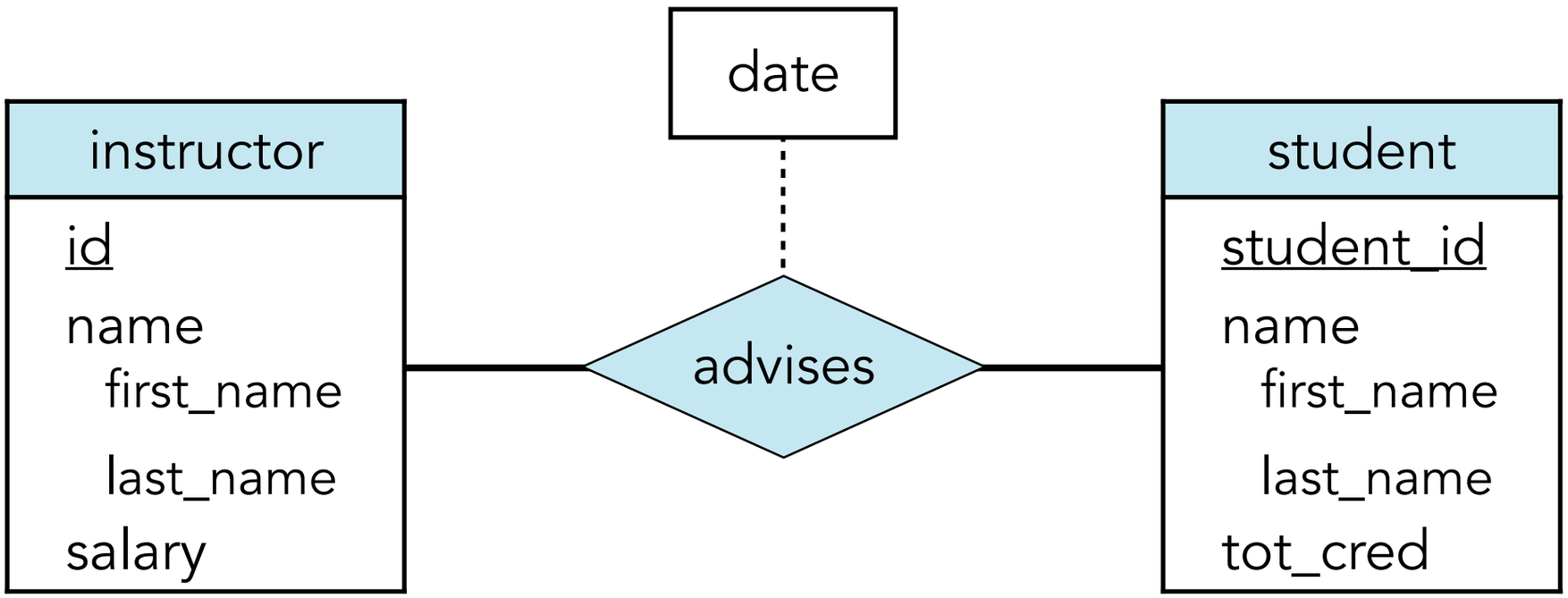
+ special rules depending on cardinality

Converting ER to Relation Models

relationships: many-to-many

relationships: many to many

the combination of the two PKs is the PK of the new table



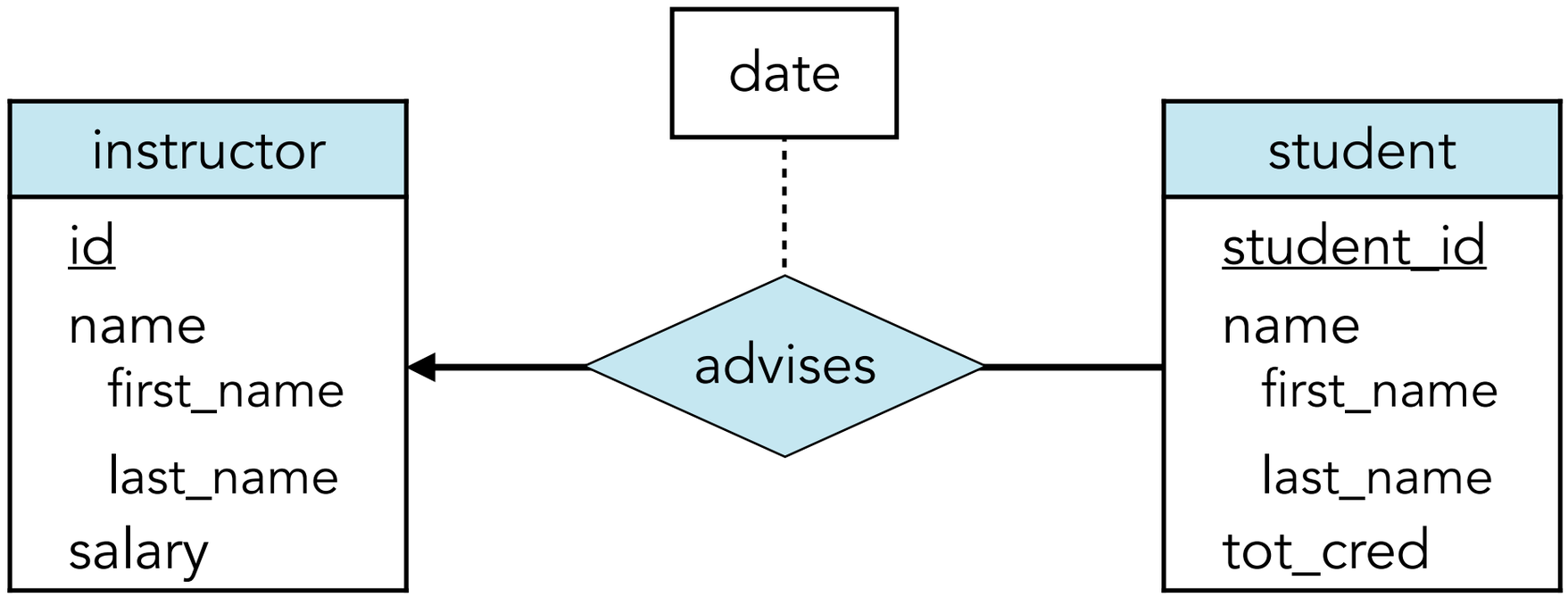
new table: advisor(instructor_id, student_id, date)
don't forget foreign key constraints!

Converting ER to Relation Models

relationships: many-to-one or vice versa

relationships: many to one, or, one to many

the PK of the new table is PK on the many side



new table: advisor(<u>student_id</u>, instructor_id, date)

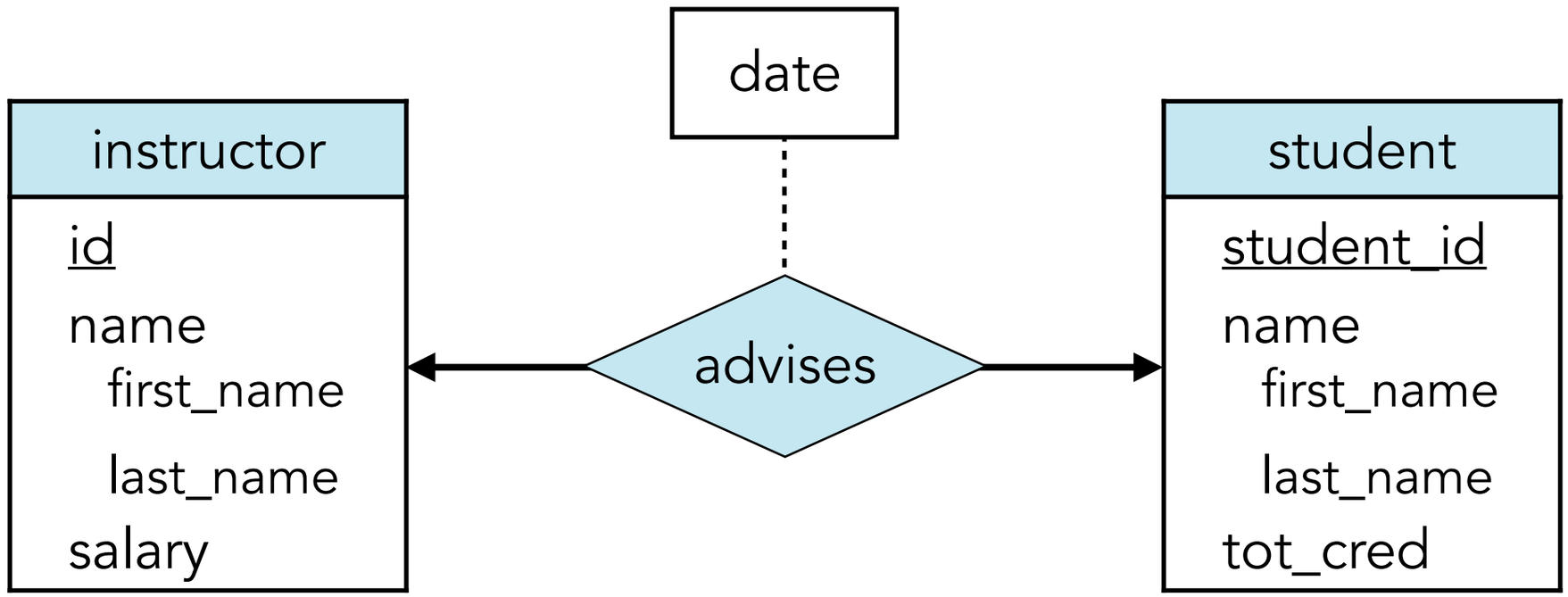
don't forget foreign key constraints!

Converting ER to Relation Models

relationships: one-to-one

relationships: one to one

the PK of the new table is PK of either one of the sides



new table: advisor(student_id, instructor_id, date)

or: advisor(instructor_id, student_id, date)

FK!

Converting ER to Relation Models

relationships: problems

relationships

follows a basic idea irrespective of cardinality:

- add attributes for the primary keys of both relations
- add any descriptive attributes of the relationship set

+ special rules depending on cardinality

but not perfect … we may have introduced redundancy!

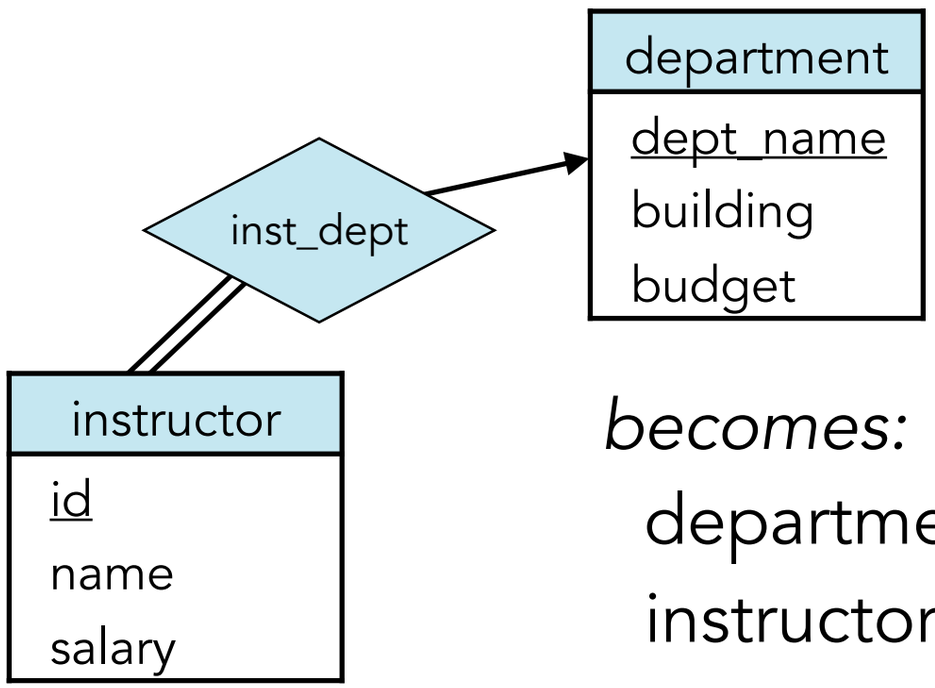
Converting ER to Relation Models

relationships: resolving redundancy

reduction of relationships: many to one, or, one to many

if the participation is total on the *many* side, then redundancy

instead: simply add attribute to the *many* side consisting of PK on the *one* side



becomes:

department(dept_name, building, budget)

instructor(id, name, salary, dept_name)

Converting ER to Relation Models

relationships: resolving redundancy cont.

reduction of relationships: many to one, or, one to many

if the participation is total on the *many* side, then redundancy

hence: if participation of one side is total, then redundancy

either new table can be chosen to
add PK attribute of other table

finally, note how relationship of weak entity set is also redundant,
so it is never explicitly added

Converting ER to Relation Models

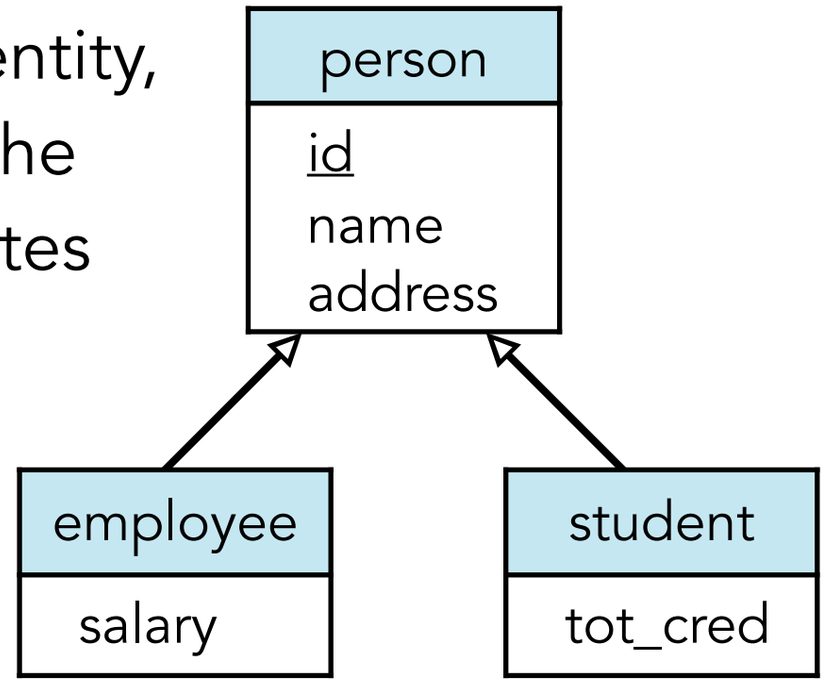
converting specialisations

converting specialisations

method 1:

- form a schema for the high-level entity (*e.g.* person)
- form a schema for every lower-level entity, including a FK to the primary key of the high-level entity and any local attributes

person(<u>id</u>, name, address)
employee(<u>person_id</u>, salary)
student(<u>person_id</u>, tot_cred)



drawback: getting information about lower-level entity requires accessing two relations (higher-level and lower-level)

Converting ER to Relation Models

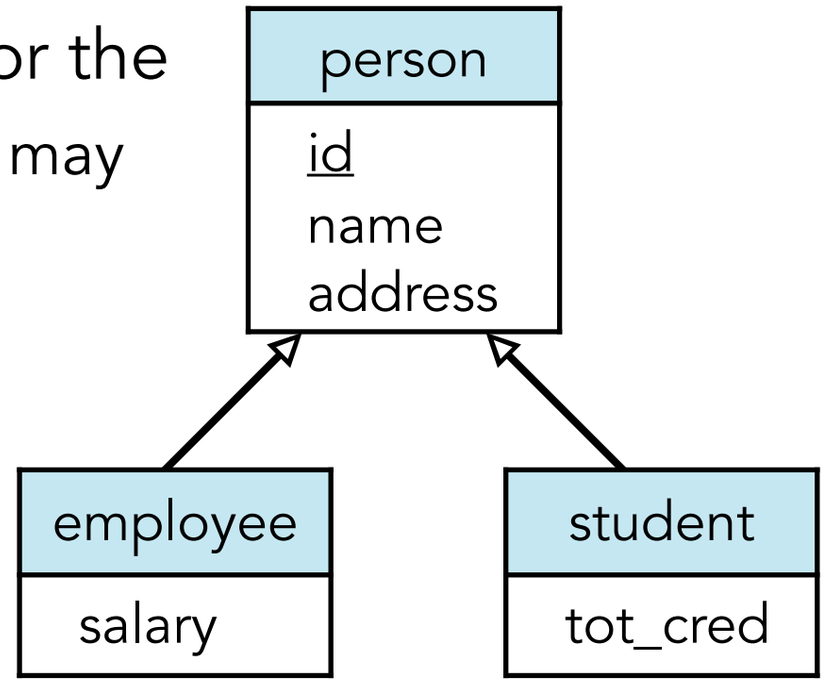
converting specialisations

converting specialisations

method 2:

- form a schema for each entity including all attributes
- if specialisation is total, the schema for the higher-level entity is not requires (but may still be needed for FK constraints)

person(<u>id</u>, name, address)
employee(<u>id</u>, name, address, salary)
student(<u>id</u>, name, address, tot_cred)



drawback: information may be stored redundantly if not disjoint,
for example: a person is both employee and student

Converting ER to Relation Models

general algorithm for converting an ER model

general approach, convert:

1. strong entity sets;
2. weak entity sets;
3. relationship sets;
4. reduction (of redundancy) of relationship sets;
5. composite and multivalued attributes;
6. specialisations.

do remember: the quality of the conversion depends (greatly!) on the quality of the original ER model

Converting ER to Relation Models

example: student ER model