1.
Introduction
SQL and NoSQL Databases

Reading: [Me] chapter 7

Course content

1. Introduction SQL and NoSQL Databases

2. Relational Databases:
    1. Transactions
    2. Concurrency Control and Consistency

3. Data and Storage Models
    1. Relational (Reference)
    2. Key-Value
    3. Document
    4. Wide-Column / Graph (briefly)

4. Distributed Databases
    1. Replication
    2. Sharding
    3. Distributed Transactions / Consistency in Distributed Databases

knirsch@htw-berlin.de

Relational Databases

Name 3 key concepts of a relational database:

Relational Databases

What is the purpose of **foreign keys**?

A) They help to address tuples uniquely.

B) They support relations between data.

C) They support consistency of the data.

D) They help to identify duplicate records.

eva.knirsch@kiu.edu.ge

Relational Databases

What is the purpose of schema **normalization**?

A) It enforces the concept that one table stores attributes of one entity

B) It increases redundancy

C) It prevents write access anomalies

D) It helps repairing poor database designs

E) It enforces constraints

eva.knirsch@kiu.edu.ge

Relational Databases

Which of the following answers is NOT true?

A **secondary index** in the context of a relational database is

A) a redundant data structure, stored separately from the data

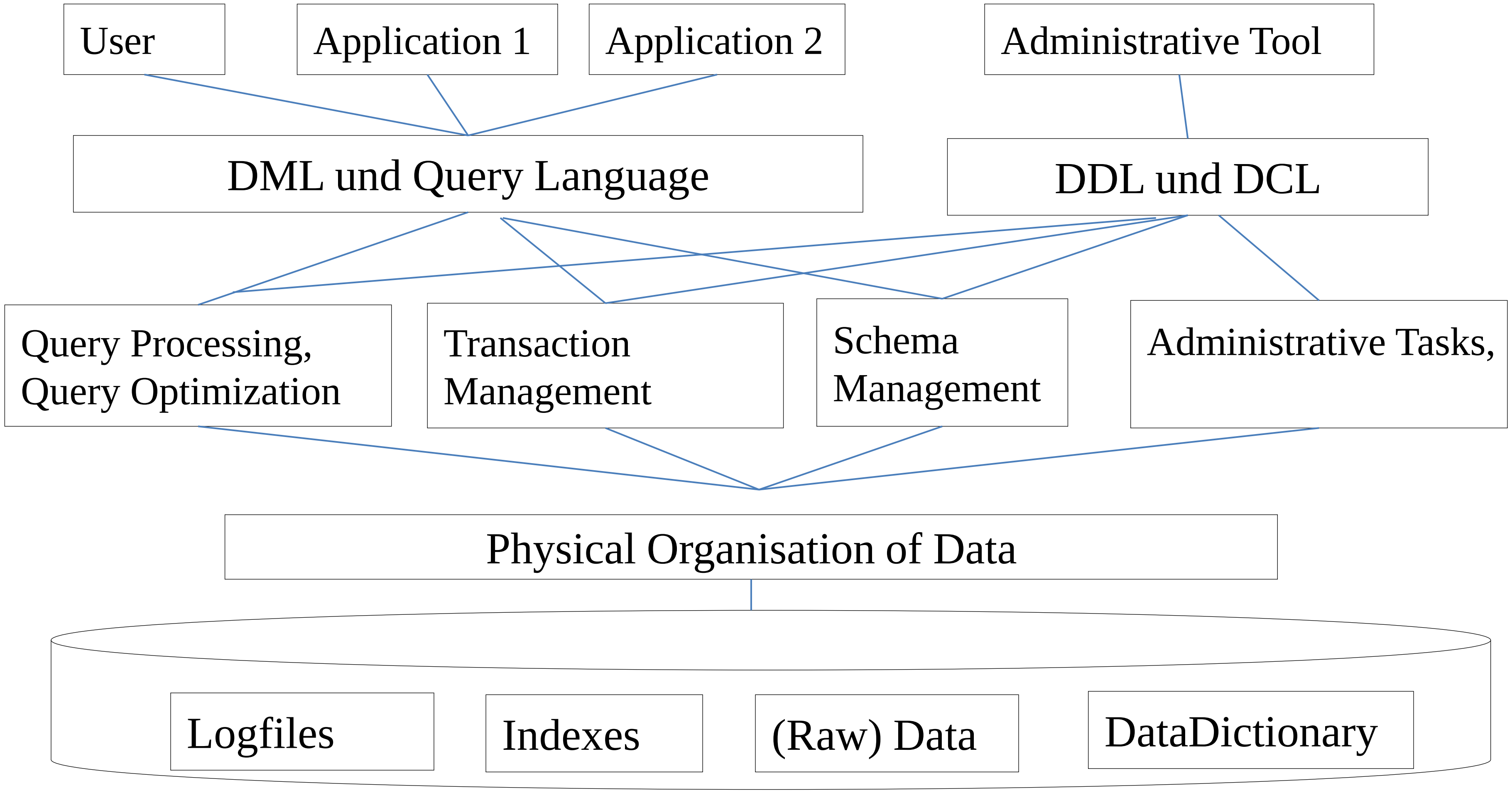
B) invisible to the application

C) a sorting order according to which the data is physically stored

D) designed to speed up data selection

eva.knirsch@kiu.edu.ge

DB / DBMS Architecture



Attempt to Categorize NoSQL Databases

NoSQL CATEGORY	EXAMPLE DATABASES	DEVELOPER
Key-value database	Dynamo Riak Redis Voldemort	Amazon Basho Redis Labs LinkedIn
Document databases	MongoDB CouchDB OrientDB RavenDB	MongoDB, Inc. Apache OrientDB Ltd. Hibernating Rhinos
Column-oriented databases	HBase Cassandra Hypertable	Apache Apache (originally Facebook) Hypertable, Inc.
Graph databases	Neo4J ArangoDB GraphBase	Neo4j ArangoDB, LLC FactNexus

Which relational databases do you know?

Source: Coronel,Morris, Data Base Systems, ch. 14

SQL and NoSQL Databases

What databases are important / popular today?

DB Engines Ranking:

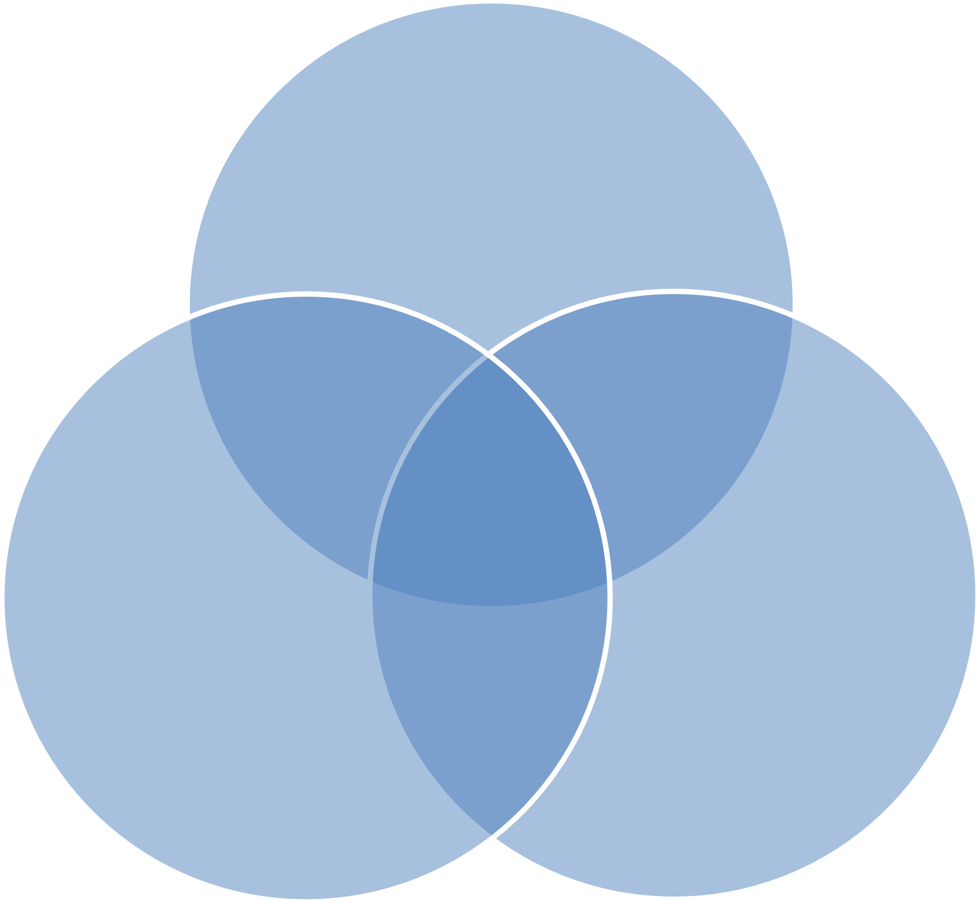
https://db-engines.com/en/ranking

eva.knirsch@kiu.edu.ge

Producers of Data

Source: University of Leipzig, Introductory Seminar: New Trends in Big Data

Big Data – as One Specific Use Case



Data Volume:
Huge amount of data →

Data Velocity / Speed:
High (Incoming) speed of write operations to create or update or delete
→

Data Variety / Data Complexity:
Variety: different data formats and data structures, complexity: relationships and / or interdependence across data
→

Volume - Scalability

”*Scalability* is the term we use to describe a system’s ability to cope with increased load.” (Kl, p.10ff)

- Scaling up (scaling vertically) –
- Scaling out (scaling horizontally) -

eva.knirsch@kiu.edu.ge

Example: Volume and Velocity

Traditional small online stores record only the items in the shopping cart and starts the corresponding ACID transactions after the "go to checkout" click.

Large online traders track and capture the entire "buying experience."

- Every click is captured
- Every search is recorded
- Record which pages the shoppers visit and how long they stay on each page
- It is recorded which products are compared

The database system not only has to generate a lot of data very quickly. It must also quickly decide which of these data to store and which not. (E.g. only pages on which the buyer was more than x seconds).

„Big Data“ Examples

- Sensor Data, e.g. collecting and analyzing vehicle sensor data
- Large Hadron Collider / Particle Physics (CERN)
- Social Media
- GPS Data
- Meteorology / Climate Monitoring
- Research, e.g. Healthcare
- Personalized Marketing
- Fraud Detection

SQL and NoSQL Databases

Key Concept	SQL Database	NoSQL Database
Standardization	SQL standard	No standards
Schema	predefined and relatively fixed	no need to be predefined, flexible
Storage (logical)	2-dimensional, in columns and rows	different storage structures: JSON (document), key-value pairs, graphs, and other
Storage (physical)	data stored in B-trees or heap files	BSON heap file + indexes (MongoDB), LSM storage (Cassandra, HBASE, LevelDB, RocksDB, ElasticSearch, Apache Lucene), other storage models
Relations between data	supports relations between data very well: PK –FK structure	key-value: not supported document: partly supported graph: excellently supported
Normalization	database goes through a normalization process to ensure that database is able to maintain consistency	No normalization process

eva.knirsch@kiu.edu.ge

SQL and NoSQL Databases

Key Concept	SQL Database	NoSQL Database
Transaction support	Yes	No, not all all or only somewhat
Scaling	predominantly vertical	Predominantly horizontal
Query language	SQL allowing for complex querying of database	Varying querying possibilities
Use cases	general purpose databases, universal, with focus on business data processing and transactional data processing	specific use cases for specific databases

eva.knirsch@kiu.edu.ge

Relational Database Course Example

Business Idea: Private Lessons Agency

- Teachers and students can register on the web application. They can access their profile, read it and update it.
- The agency specifies the subjects for which it offers teachers and lessons, e.g. Math, Computer Science, Chemistry, Physics, …
- Teachers can register to offer lessons in multiple of these subjects. They can present themselves and their experience.
- Students can search for suitable teachers and buy lessons. A counter stores how many lessons a student has bought. Students can only buy lessons in increments of 5.
- Teachers also carry a counter with them. It stores how many lessons they taught and will get paid for.
- The agency, students or teachers can schedule a lesson. When a lesson is scheduled, the student's counter is decremented and the teacher's counter is incremented.
- The price for a lesson depends on the age of the student. There are 4 priceGroups: child, teen1, teen2 and adult.

- Note: **We will not be programming** this web application. We will just design, implement and use the underlying database for comparison with other databases. The database design will **NOT** represent the complete business case as we focus on areas where we can get new insights. So, please, only model and implement the given requirements.

Possible Minimal Frontend

Logo

Login

Menue

You are a student looking for a teacher?
Register here:

You are a teacher and would like to share your knowledge?
Register here:

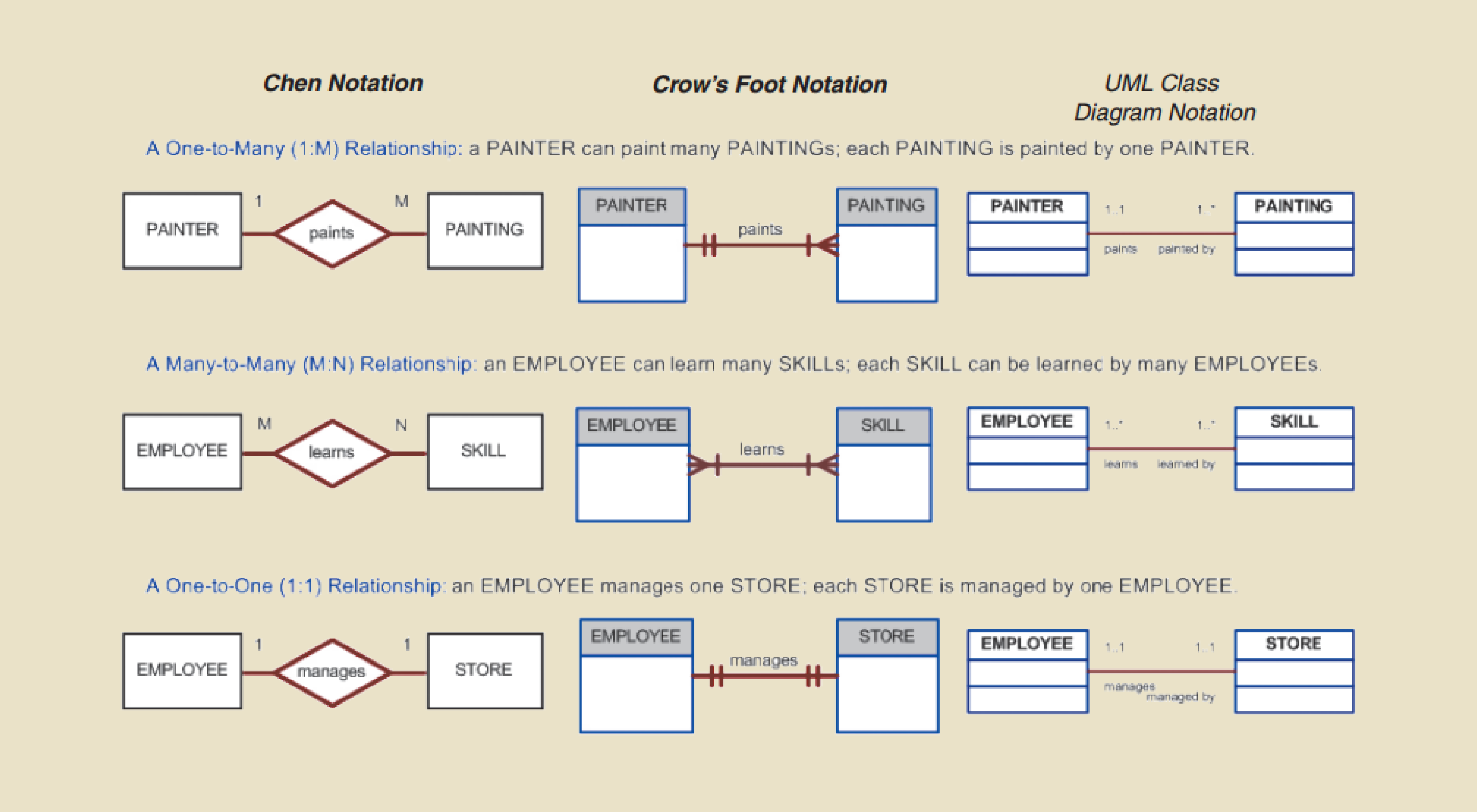
Lesson Database

Create the **ER model** for our course database, using Chen notation.

- You will need the following entity tables: subject, teacher, student, priceType, lessonPurchase
- What relationships (relationship tables) do you need?
- Model relationships and cardinalities as demanded by the requirement
- Model the attributes as needed

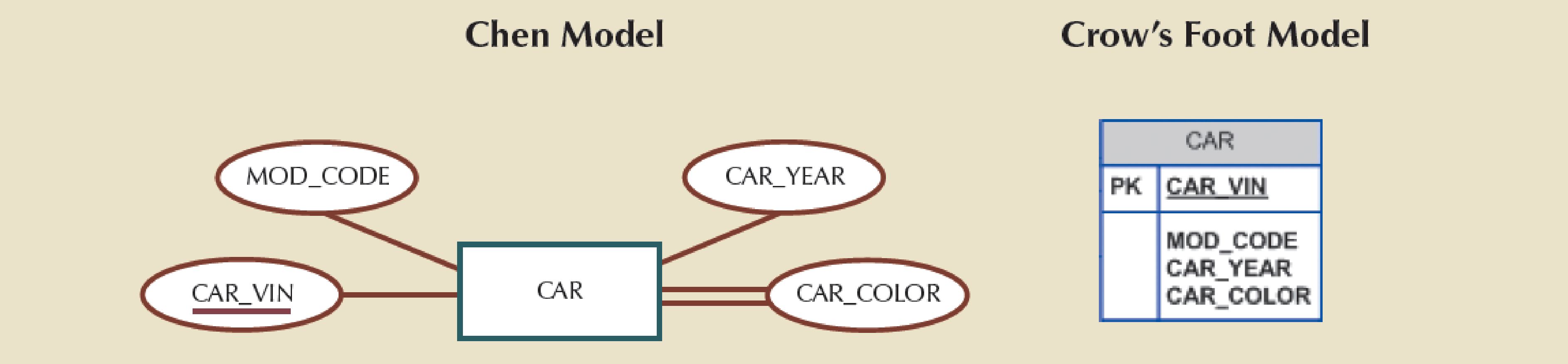
The Entity Relationship Model (2 of 2)

FIGURE 2.3 THE ER MODEL NOTATIONS



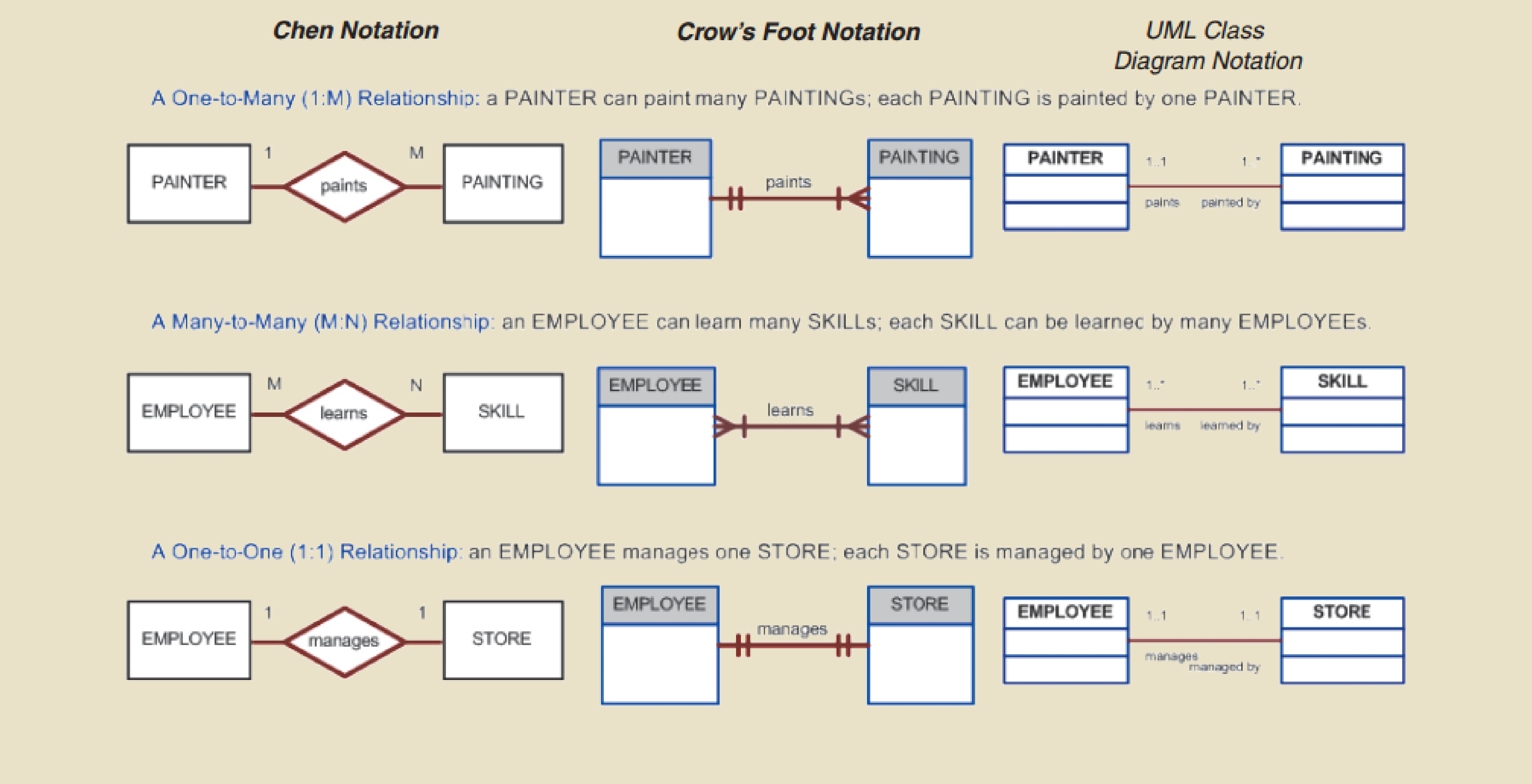
Attributes (3 of 7)

FIGURE 4.3 A MULTIVALUED ATTRIBUTE IN AN ENTITY



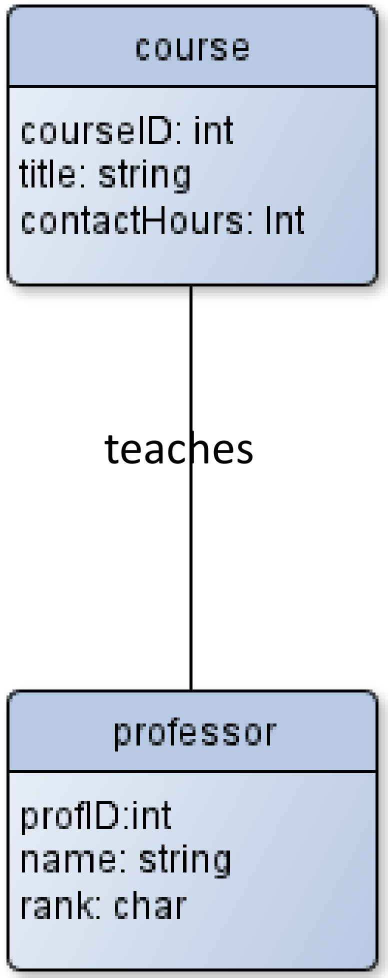
The Entity Relationship Model (2 of 2)

FIGURE 2.3 THE ER MODEL NOTATIONS

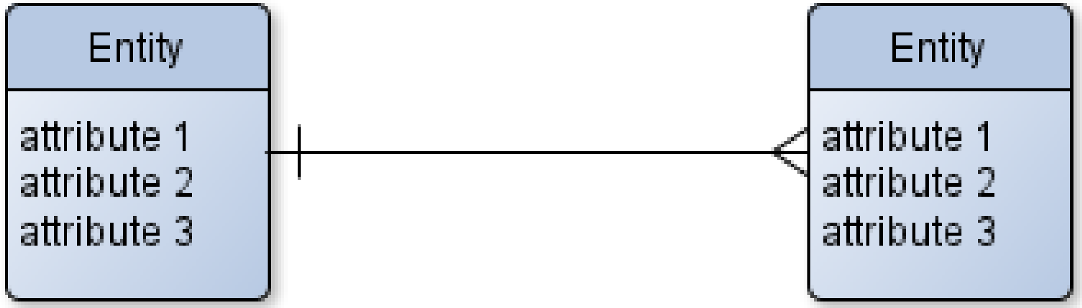


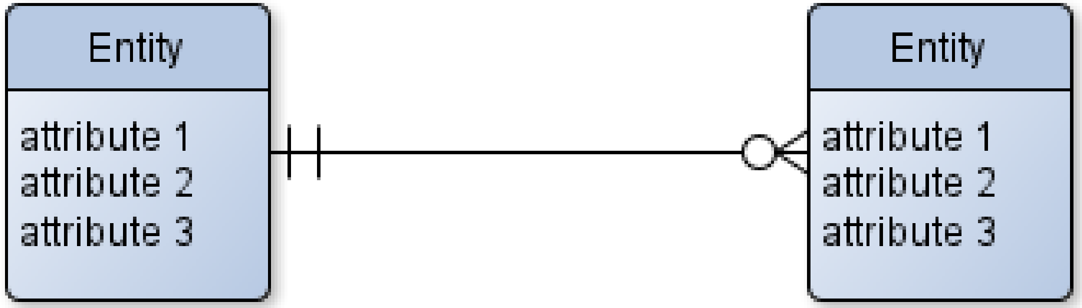
ER Model Crow-Foot Notation

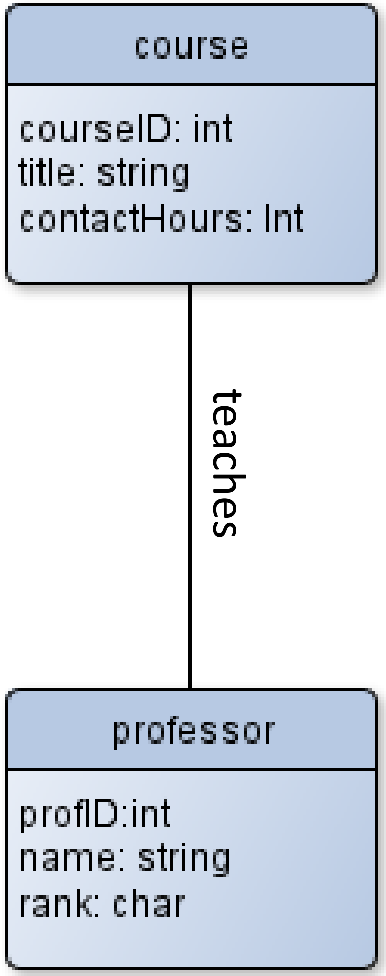
- In crow-foot notation, an entity is represented by a rectangle with the name of the entity in the top.
- Attributs (and possibly properties of attributes) are given in the body of the rectangle. In addition to the datatype, sometimes PK or NULL / not NULL is shown.
- Relationships do NOT have a specific shape. They are simply represented by a line between the entities. The line usually carries a label defining the relationship.
- Severe limitations of crow-foot notation:
    - attributes of relationships cannot be represented. They have to be described verbally.
    - Representation of n-ary relations (e.g.ternary relations) does not exist.
- Advantage: ER model more clearly arranged
- Disadvantage: More difficult to map properly to a database schema.



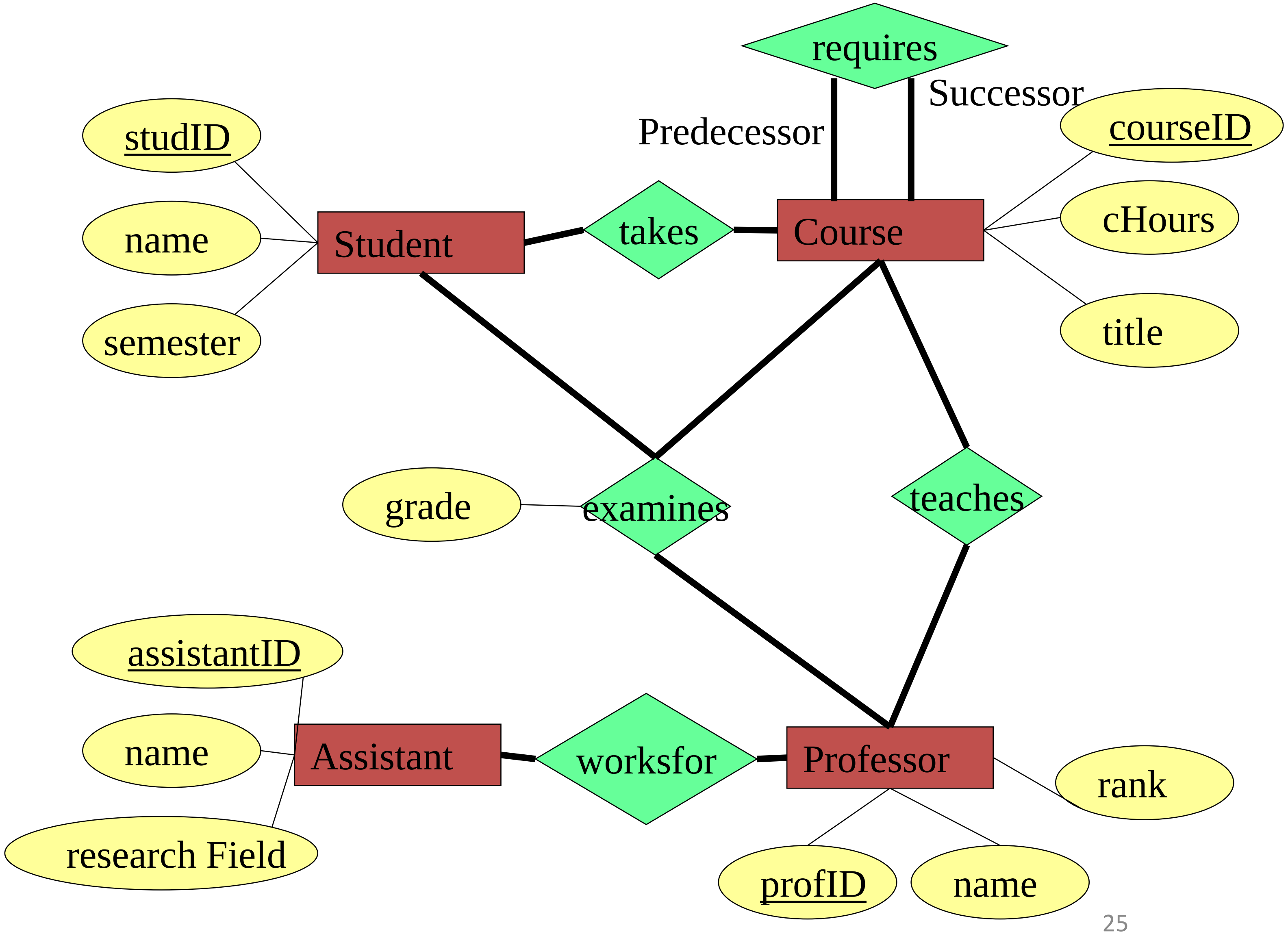
ER Model Crow-Foot Cardinalities





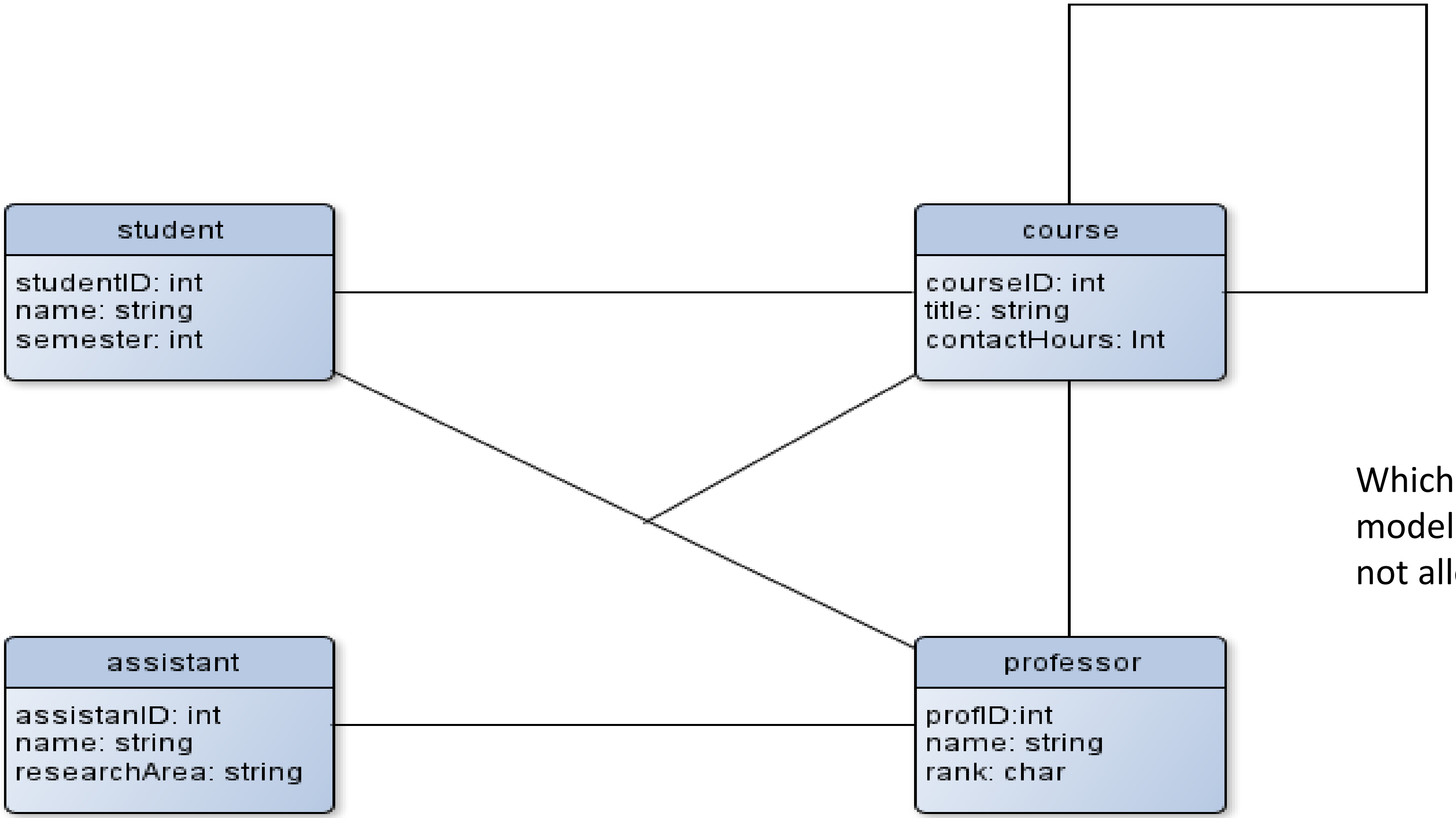


University ER Model Chen Notation



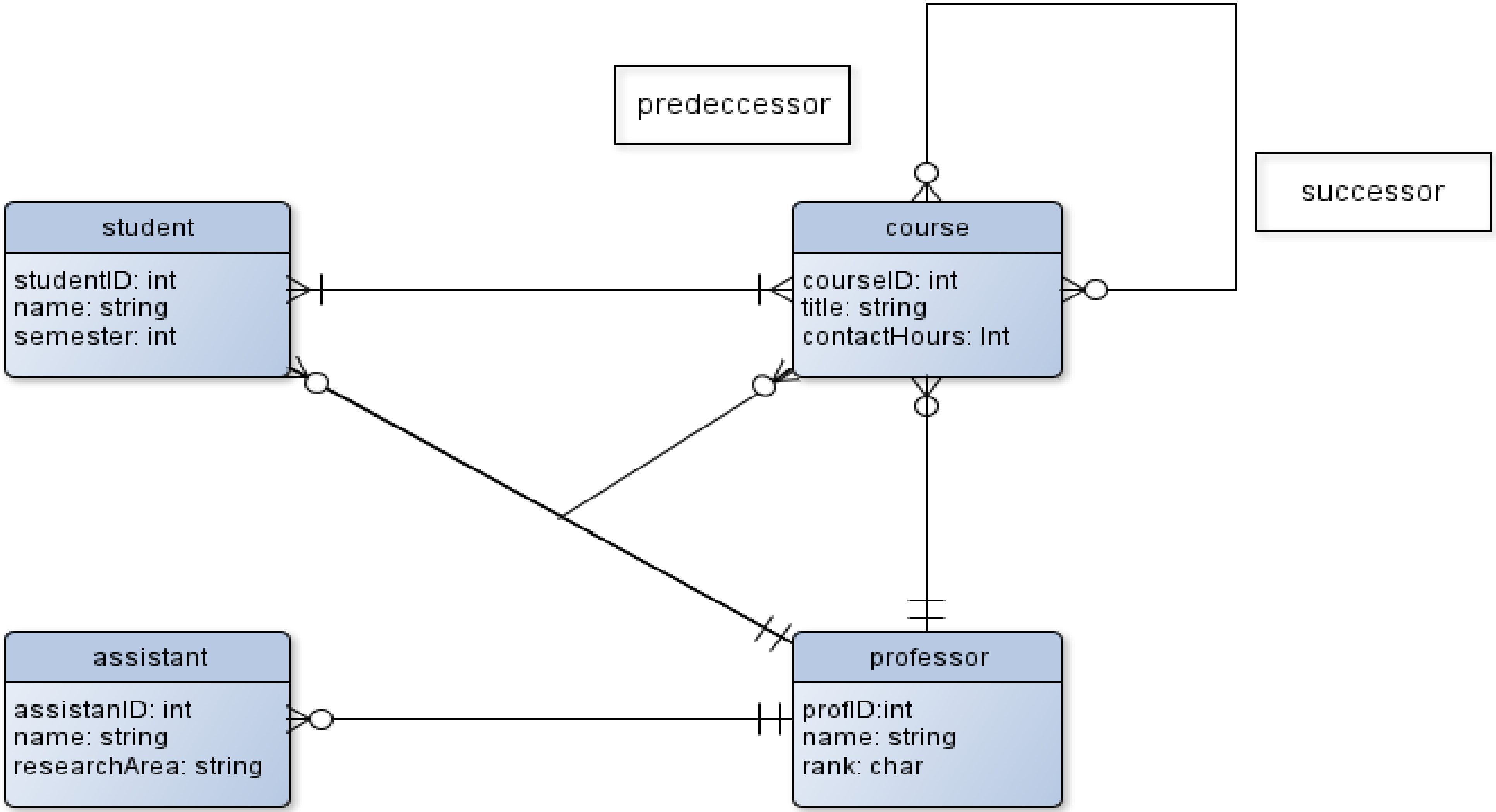
KUTAISI INTERNATIONAL UNIVERSITY

University ER Model Crow-Foot Notation



Which part of the model is officially not allowed?

University ER Model Crow-Foot Notation

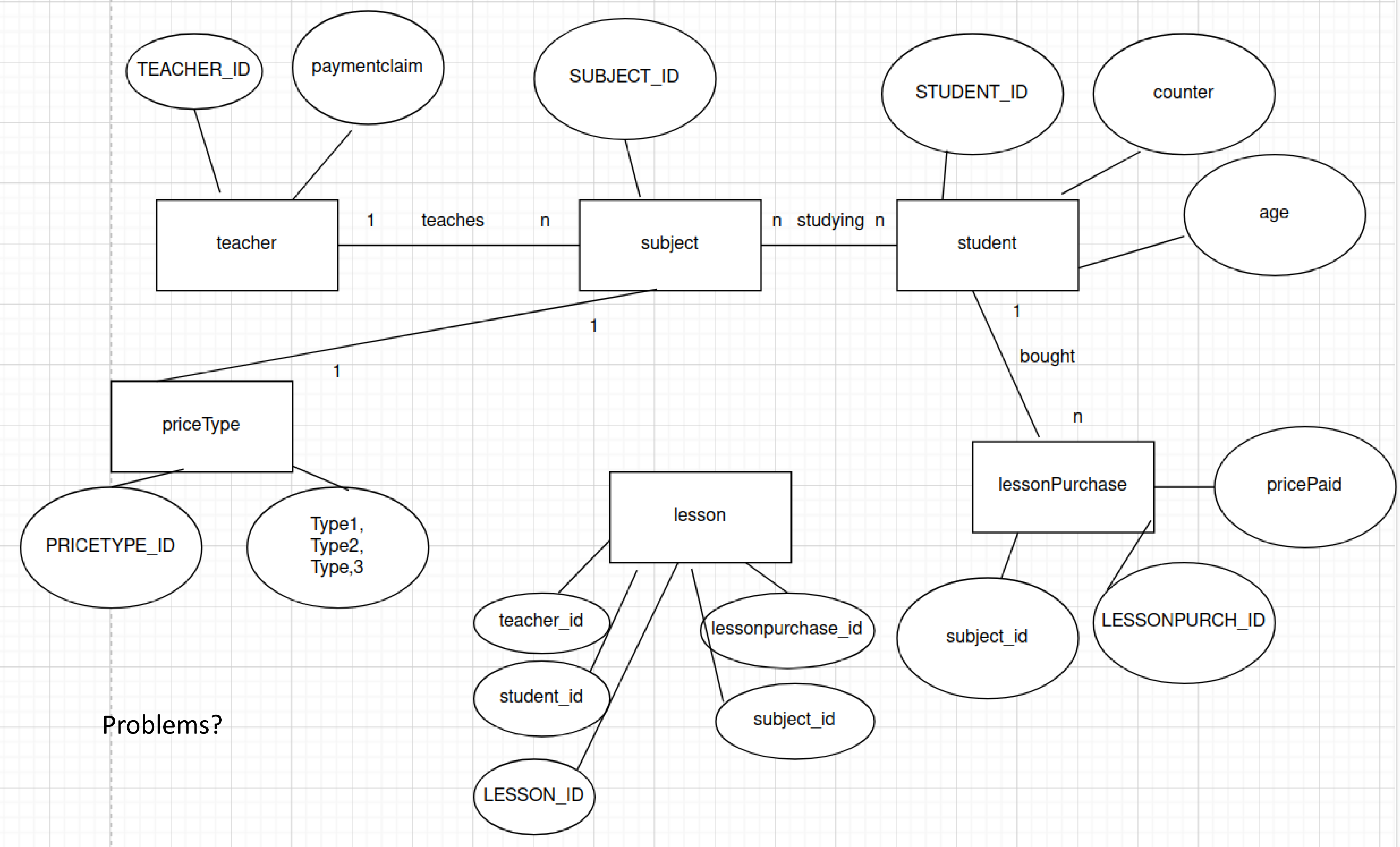


Relational Reference Course Example

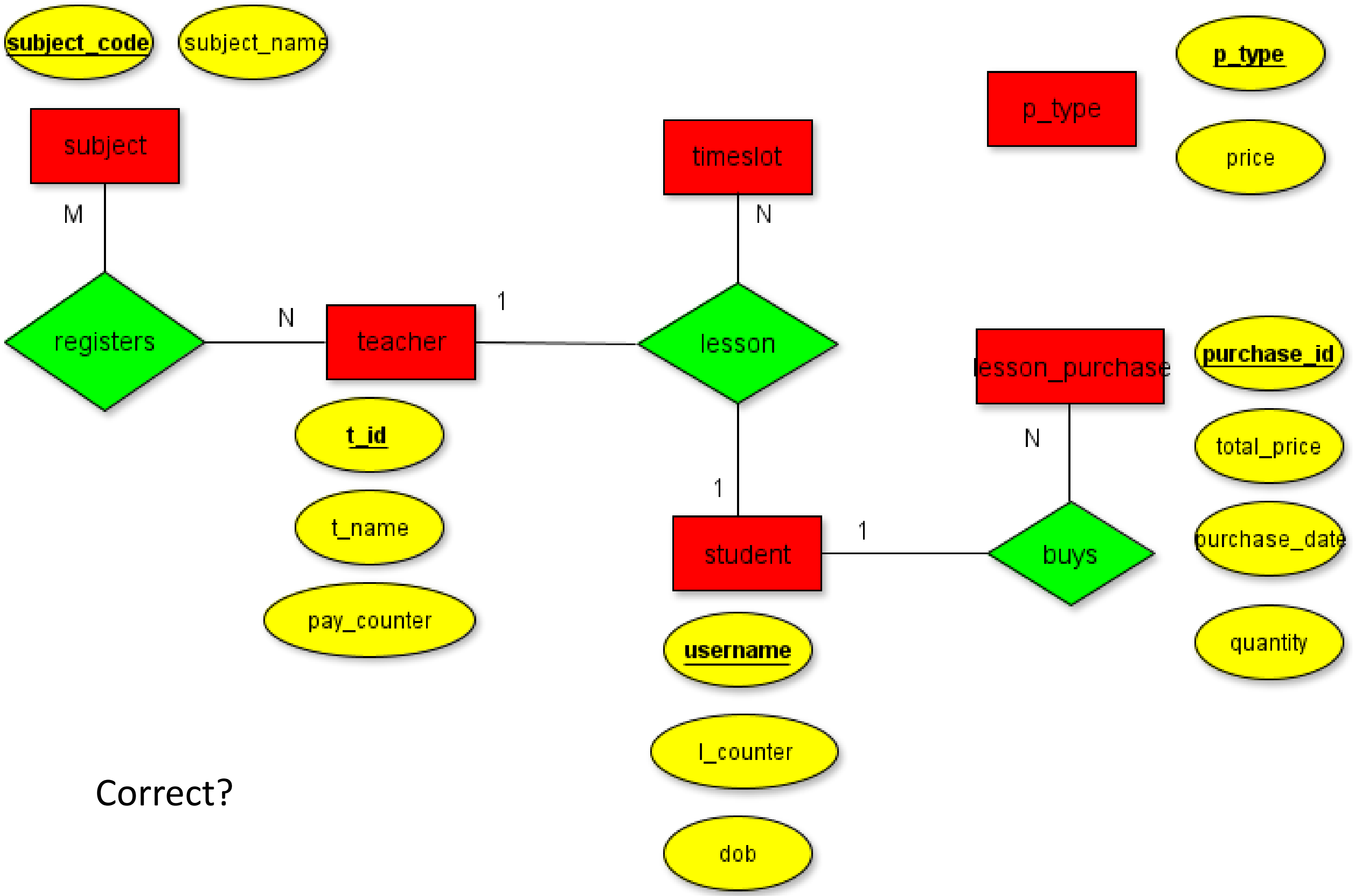
Which of the integrity requirements can be directly designed in the ER model and directly implmenented in the database scheme? Which of the integrity requirements need additional methods to be implemented? How would you implement these?

Referential Integrity Requirements:

1. Only registered teachers and students can schedule a lesson.
2. Teachers can register for several subjects but only for subjects that the agency offers. They can only teach the subjects that they registered for.
3. Only registered students can buy lessons. Students need to buy lessons in advance in increments of 5. A counter per student contains the number of prepaid lessons.
4. The price for a lesson depends on the student age, with 4 priceTypes: CHILD (age <10), TEEN1 (age <15), TEEN2 (age < 20) and ADULT (age 20+). This means that a lesson price for an individual student changes with his / her age.
5. Students can only schedule / take lessons that they already paid for.
6. With each lesson, the student counter is decremented by one. Teachers also have a counter. The teacher counter is incremented by one with each lesson. The teacher counter contains the teacher's current payment claim.
7. Lessons can only be scheduled in such a way that at a certain timeslot
   - a specific teacher can only teach one lesson with one student
   - a specific student can only take one lesson with one teacher
   - the teacher can only teach a lesson in a subject he registered for.

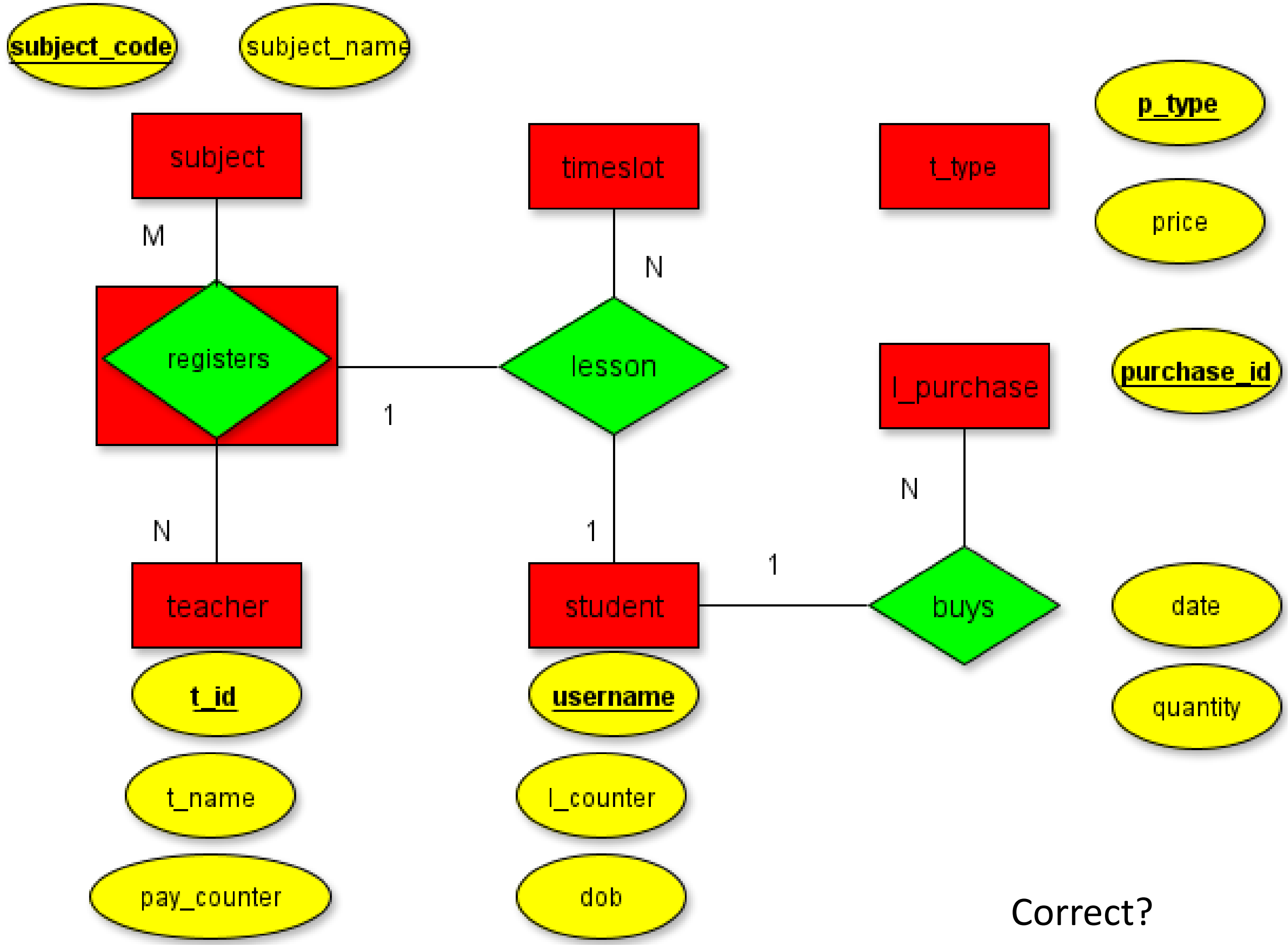

TEACHER_ID
paymentclaim
SUBJECT_ID
STUDENT_ID
counter
age
teacher
1
teaches
n
subject
n
studying
n
student
1
1
1
bought
n
priceType
lessonPurchase
pricePaid
PRICETYPE_ID
Type1,
Type2,
Type,3
lesson
teacher_id
student_id
LESSON_ID
lessonpurchase_id
subject_id
subject_id
LESSONPURCH_ID

Problems?

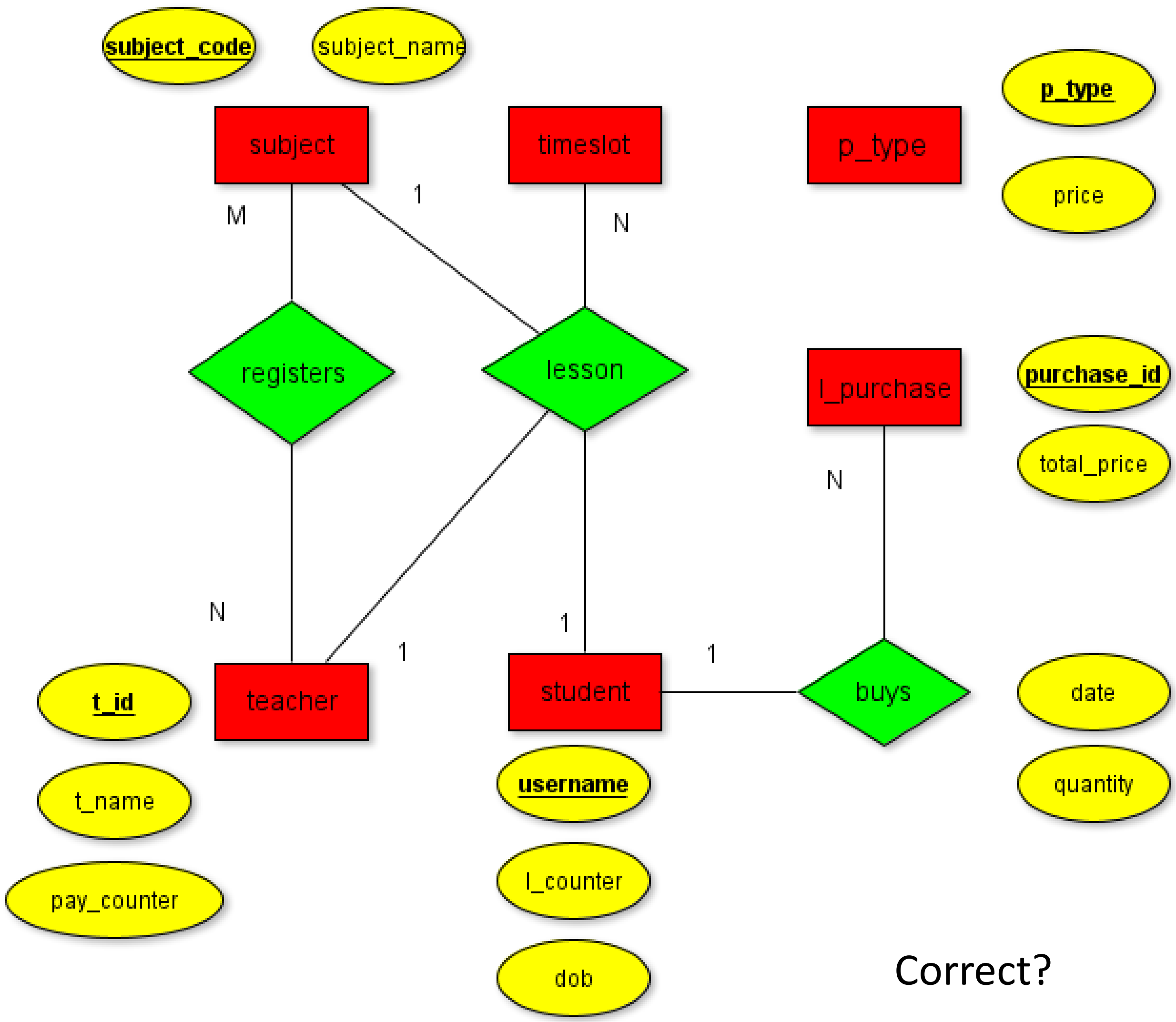


Correct?

eva.knirsch@kiu.edu.ge

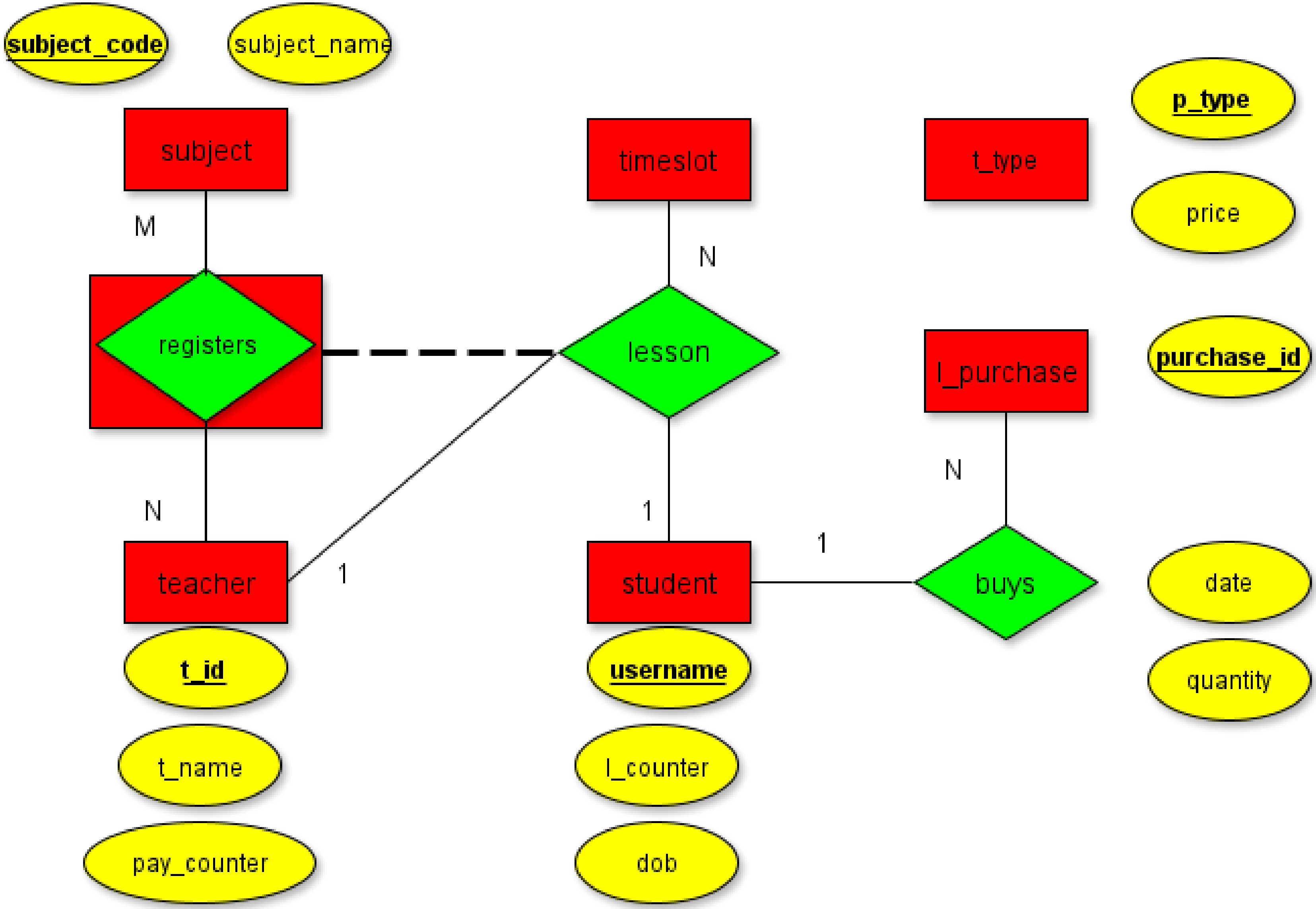

subject_code
subject_name
subject
timeslot
t_type
p_type
price
M
N
registers
lesson
1
l_purchase
purchase_id
N
N
1
1
teacher
student
buys
date
quantity
t_id
username
t_name
l_counter
pay_counter
dob

Correct?

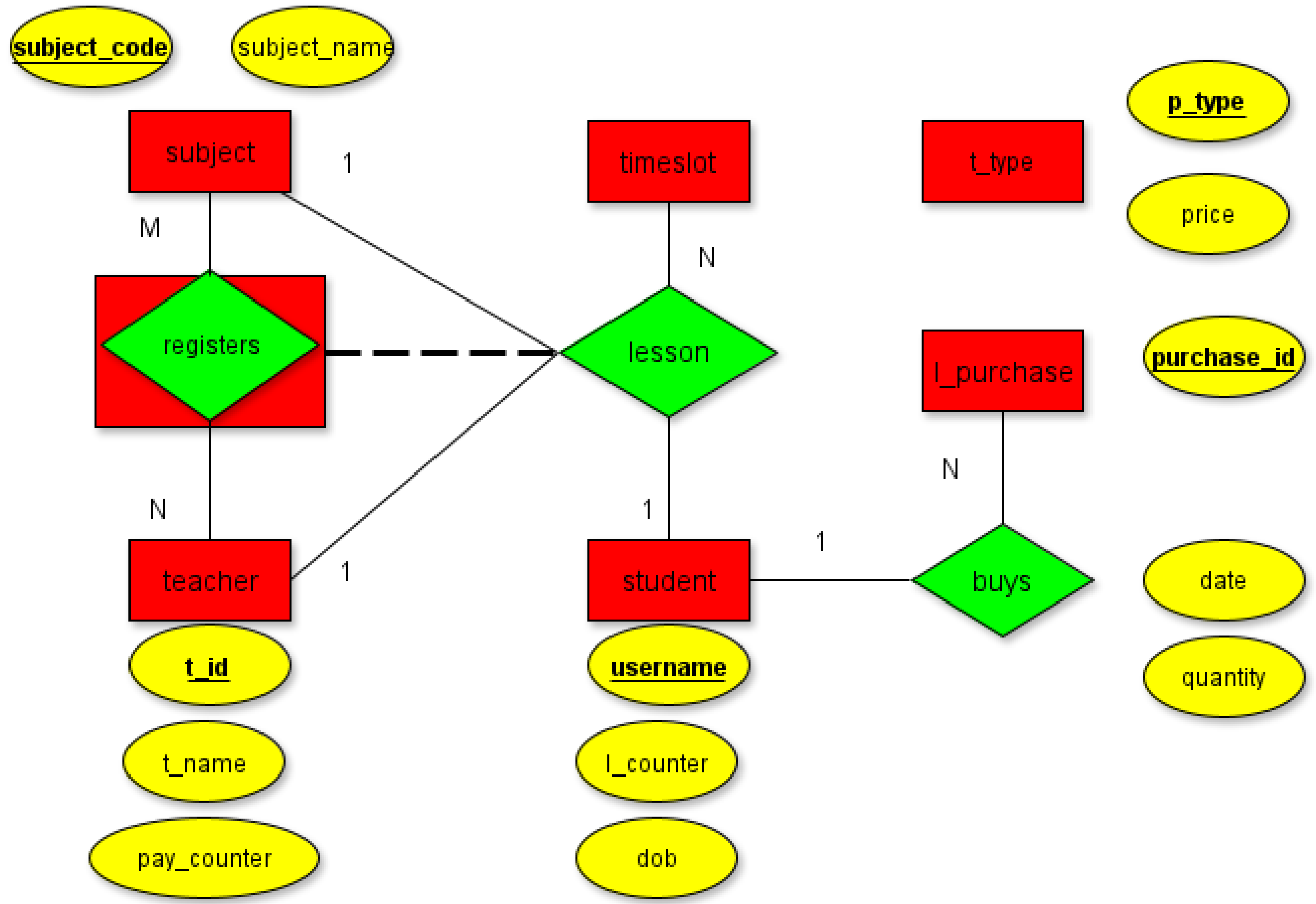


Correct?

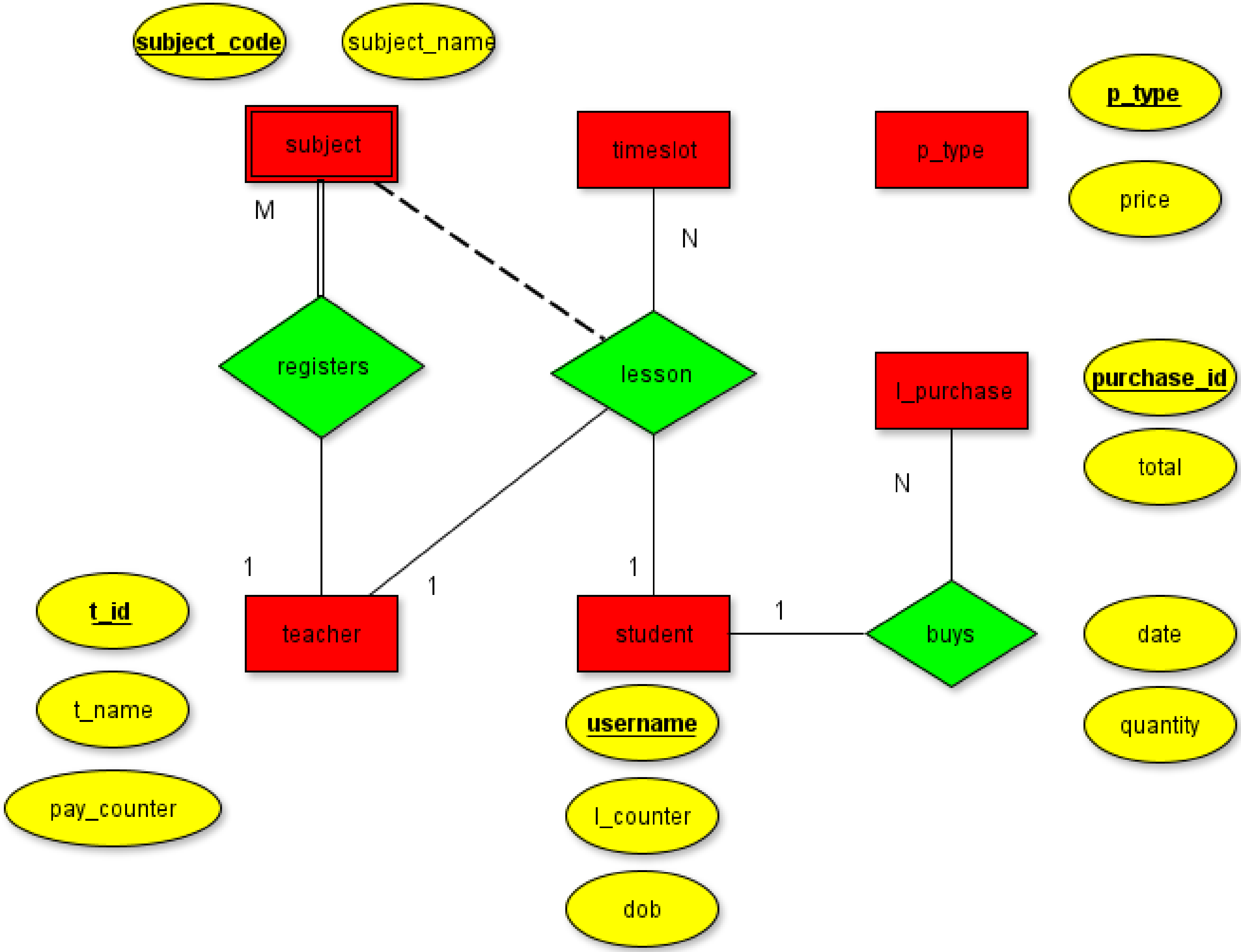
eva.knirsch@kiu.edu.ge



eva.knirsch@kiu.edu.ge



eva.knirsch@kiu.edu.ge


subject_code
subject_name
subject
timeslot
p_type
p_type
price
M
N
registers
lesson
l_purchase
purchase_id
total
N
1
1
1
1
1
teacher
student
buys
date
quantity
t_id
t_name
pay_counter
username
l_counter
dob

Implementation of Semantic Requirements

How can you implement the requirement that

Teachers can only register in subjects that the agency offers.

eva.knirsch@kiu.edu.ge

Implementation of Semantic Requirements

How can you implement the requirement that

Only registered students can buy lessons and they can only buy lessons in increments of 5?

eva.knirsch@kiu.edu.ge

Implementation of Semantic Requirements

How can you implement the requirement that

the price of a lesson depends on the age of the student?

Note that PostgresQL does NOT have generated virtual columns.

eva.knirsch@kiu.edu.ge

Implementation of Semantic Requirements

How can you implement the requirement that

the price of a lesson depends on the age of the student?

- If this is implemented with a FK-PK relation between student table and price table, then there is always a certain percentage of incorrect values stored in the FK in the student table.
Attention: This is not a violation of referential integrity. It is simply a wrong value according to the business rules.

- Better not to implement a (at times) wrong FK-PK link but to find out the correct price at purchase time, e.g. by running a trigger on the table l_purachse.

- Of course, you can have "standalone" tables in your database, like price_type in our example.

Implementation of Semantic Requirements

How can you implement the requirement that

at a certain timeslot one teacher can only teach one student in a subject that he is registered for?

eva.knirsch@kiu.edu.ge

Implementation of Semantic Requirements

How can you implement the requirement that **at a certain timeslot one teacher can only teach one student in a subject that he is registered for?**

- a ternary relation takes care of the requirement:

at a certain timeslot one teacher can only teach one student

- the requirement that the lesson is taught in a subject the teacher is registered for is more challenging to implement. It is not possible with a simple ternary relation (timeslot, student, teacher_subject) and also not possible with a simple quad relation (timeslot, student, teacher, subject)

Possible solution options:

- quad relation between timeslot, student, teacher and teacherSubject with the characteristic that the entity teacher_subject is NOT part of the PK but another FK relation.

- solution using a trigger: whenever a row is inserted in lesson table, the trigger checks that the subject is one the teacher is registered for.

- tolerate a possible violation at insert time and handle the issue via the application, e.g. checking ossacionally if the teachers only teach the subjects they are registered in.

eva.knirsch@kiu.edu.ge

DB2 Assignments 2

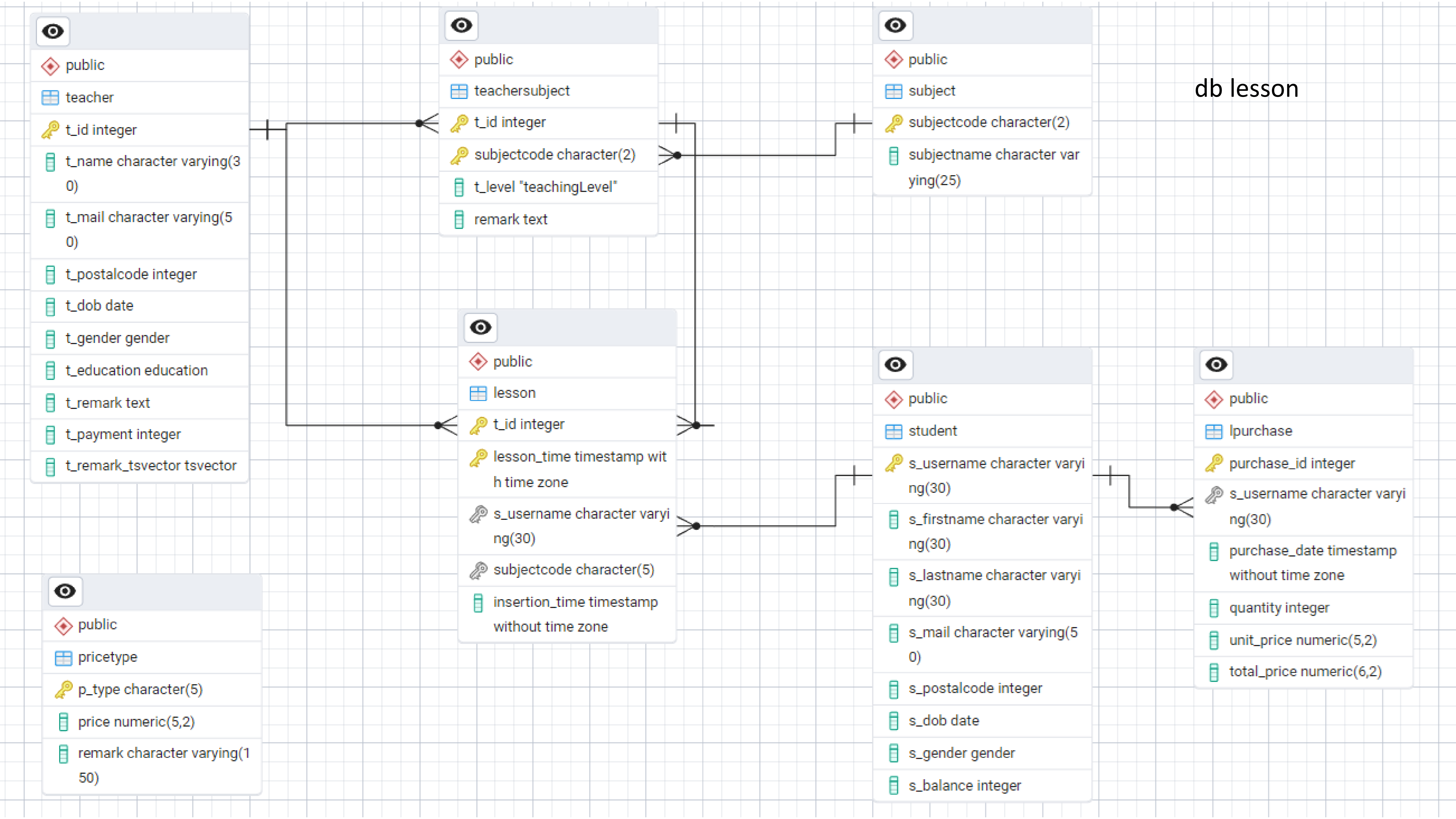
Lesson Database

Implement database lesson according to ERD on next slide.

ERD: ER-Diagram: visualized scheme of an existing database. An ERD is NOT a database design.

- add constraints (PKs, FKs) to fullfill integrity requirements,
- add check constraint to only allow purchases in increments of 5,
- optional: add trigger to calculate price at purchase time depending on age of students,
- insert values as provided in TEAMS.

eva.knirsch@kiu.edu.ge


db lesson
public
teacher
t_id integer
t_name character varying(30)
t_mail character varying(50)
t_postalcode integer
t_dob date
t_gender gender
t_education education
t_remark text
t_payment integer
t_remark_tsvector tsvector
public
teachersubject
t_id integer
subjectcode character(2)
t_level "teachingLevel"
remark text
public
subject
subjectcode character(2)
subjectname character varying(25)
public
lesson
t_id integer
lesson_time timestamp with time zone
s_username character varying(30)
subjectcode character(5)
insertion_time timestamp without time zone
public
student
s_username character varying(30)
s_firstname character varying(30)
s_lastname character varying(30)
s_mail character varying(50)
s_postalcode integer
s_dob date
s_gender gender
s_balance integer
public
lpurchase
purchase_id integer
s_username character varying(30)
purchase_date timestamp without time zone
quantity integer
unit_price numeric(5,2)
total_price numeric(6,2)
public
pricetype
p_type character(5)
price numeric(5,2)
remark character varying(150)

Referential Integrity

Insert the following row into table lesson:

```
INSERT INTO public.lesson(
	t_id, s_username, subjectcode)
	VALUES (1, 'Wolf', 'CH');
```

What error do you get an why?

eva.knirsch@kiu.edu.ge

Referential Integrity

We schedule a couple of lessons in our database.
What happens when we insert the following two rows into table lesson?

```
INSERT INTO lesson (t_id, lesson_time, s_username, subjectcode)
        VALUES (14, '2024-04-08 05:22:12.000000', 'Mickey', 'EN');

INSERT INTO lesson (t_id, lesson_time, s_username, subjectcode)
        VALUES (14, '2024-04-08 05:22:12.000000', 'Rose', 'EN');
```

Does the database react as we expect?