COMS W4111: Introduction to Databases
Spring 2023, Sections 002

Homework 1, Part 1
Introduction to Core Concepts, ER Modeling, Relational Algebra, SQL

* </i>*

Introduction and Overview

HW Objectives

- *HW 1 will have two parts. This is the first part.*
- *Part 1 tests and reinforces the basic elements of:*
  - *Database concepts.*
  - *Relational model and relational algebra.*
  - *Data modeling.*
  - *SQL.*
- *Part 2 will have a set of practical exercises that have students implement simple but realistic tasks.*
- *Part 1 applies to both programming and non-programming tracks.*

Submission Instructions

Complete all the tests in this notebook and submit only this notebook as a PDF to GradeScope. To convert the jupyter notebook into a pdf you can use either of the following methods:

- *File --> Print Preview --> Print --> Save to PDF*
- *File --> Download As HTML --> Print --> Save to PDF*

Due date: February 12, 11:59 PM EDT on GradeScope

It is recommended that you put the screenshots into the same folder as this notebook so you do not have to alter the path to include your images.

Please read all the instructions thoroughly!

Guidelines

*You may not work with or collaborate with anyone in any way to complete the homework. You may speak with the professor and TAs. You may ask **private** questions on Ed if you need clarification.*

You may use lecture slides, the textbook slides, the textbook or public information on the web to help you answer your questions. You may not "cut and past" information. Your answer must be in your own words and demonstrate the you understand the concept. If you use information for sources other than lectures, lecture slides, textbook slides or the textbook, you MUST provide a URL to the source you used.

Read the Columbia University academic integrity information.. In the answer section, state that you take the pledge.

Answer:

Add Student Information

1. *Replace my name with your full name.*
2. *Replace my UNI with your UNI.*
3. *Replace "Cool Track" with either "Programming" or "Non-programming."*

In [1]:
```
# Print your name, uni, and track below

name = "Haoqing Wang"
uni = "hw2888"
track = "Programming"

print(name)
print(uni)
print(track)
```

```
Haoqing Wang
hw2888
Programming
```

Testing Environment

Run the following cells to ensure that your environment is set up.

You may need to change passwords.

In [2]:
```
import pymysql
```

In [3]:
```
%load_ext sql
```

In [4]:
```
%sql mysql+pymysql://root:WHQ21cd1c689742@localhost
```

In [5]:
```
%sql select * from db_book.student where ID=12345
```

```
 * mysql+pymysql://root:***@localhost
1 rows affected.
```

Out[5]:

ID	name	dept_name	tot_cred
12345	Shankar	Comp. Sci.	32

In [6]:
```
from sqlalchemy import create_engine
```

In [7]:
```
engine = create_engine("mysql+pymysql://root:WHQ21cd1c689742@localhost")
```

In [8]:
```
def get_connection(userid, pw):
    conn = pymysql.connect(
        host="localhost",
        user=userid,
        passwd=pw,
        autocommit=True,
        cursorclass=pymysql.cursors.DictCursor
    )
    return conn

pymysql_conn = get_connection("root", "WHQ21cd1c689742")
```

In [9]:
```
dept_name="Comp. Sci."
total_cred=50

cur = pymysql_conn.cursor()

sql = "select * from db_book.student where dept_name=%s and tot_cred >= %s"

res = cur.execute(sql, args=(dept_name, total_cred))
```

In [10]:
```
print("The number of rows in the result is", res)
```

```
The number of rows in the result is 3
```

In [11]:
```
students = cur.fetchall()
students
```

Out[11]:
```
[{'ID': '00128',
  'name': 'Zhang',
  'dept_name': 'Comp. Sci.',
  'tot_cred': Decimal('102')},
 {'ID': '54321',
  'name': 'Williams',
  'dept_name': 'Comp. Sci.',
  'tot_cred': Decimal('54')},
 {'ID': '76543',
  'name': 'Brown',
  'dept_name': 'Comp. Sci.',
  'tot_cred': Decimal('58')}]
```

In [12]:
```
import pandas
```

In [13]:
```
data_dir = "./data"
```

In [14]:
```
df = pandas.read_csv(data_dir + "/" + "departments.csv")
```

In [15]:
```
df
```

Out[15]:

	COMS	Computer Science
0	MATH	Mathematics
1	IEOR	Industrial Engineering/Operations Research
2	ECON	Economics

In [16]:
```
#%sql drop database W4111_HW1
%sql create database if not exists W4111_HW1
```

```
 * mysql+pymysql://root:***@localhost
1 rows affected.
```

Out[16]:
```
[]
```

In [17]:
```
df.to_sql("departments", schema="W4111_HW1", index=False, if_exists="replace", con=engine)
```

Out[17]:
```
3
```

In [18]:
```
%sql select * from w4111_hw1.departments
```

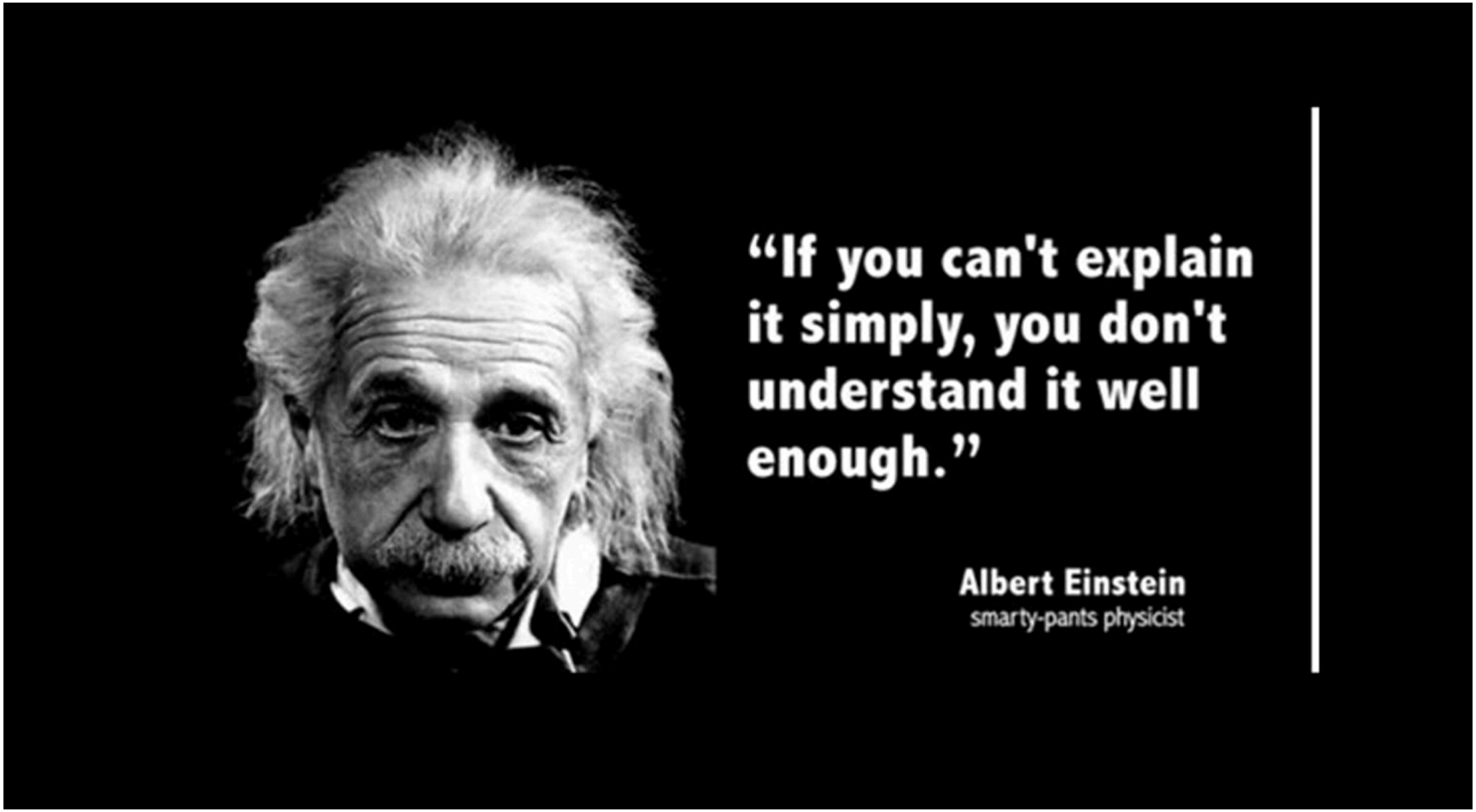
```
 * mysql+pymysql://root:***@localhost
3 rows affected.
```

Out[18]:

COMS	Computer Science
MATH	Mathematics
IEOR	Industrial Engineering/Operations Research
ECON	Economics

Written Questions

Do not bloviate. Students cnfuse quantity of words with quality of the answer. We will deduct points if you are not succinct.



W1: *Briefly explain 4 types of database user. Briefly explain a database administrator.*

Answer:

***Naive Users**: Unsophisticated users interacting with the database with the predefined user interfaces.*

***Application Programmers**: Professionals who write application programs.*

***Sophisticated Users**: Interact with the database system without writing programs, rather, directly send request to a database using query languages or tools to explore data matching the query sentence.*

***Database Administrator**: Person who have central control over both the data and programs acessing the data.*

1. *DBA creates the original database schema through executing a series of data definition in DDL.*
2. *DBA defines storage and acess method.*
3. *DBA update the schema and physical organization for performance improvement.*
4. *DBA is able to give different level of acess of the data to different types of user.*
5. *DBA should keeps routine maintenance to keep everything updated and checked.*

***W2:** Briefly explain structured data and unstructured data. Give an example of each.*

Answer:

***Structured Data**: Data that have a well-defined structure, and have fixed number of attributes and fields. Example would be Excel.*

***Semi-structured Data**: Data which inidividual items of the same type may have different sets of attributes. It is structured but does not have a fixed schema. Example would be JSON and XML.*

***Unstructured Data**: Data that shows no structure, it is a collection of varies data types that stored in native formats. Hard to manage and interpret. Example would be Videos.*

W3: *Briefly define the following terms. Give an example of each using the sample database associated with the recommended textbook.*

1. *Super Key*

   Set of one or more attribuates that allow us to identify a unique tuple from the relation. For example for the instructor relation, {ID, Name} is a superkey because every ID and name together is unique and name is not super key because instructor can have dupplicated names.

2. *Candidate Key*

   Candidate Key is a minimal Super Key. This means any one or combination of attributes forms an unique identification of data, and removing any attribute from combination forms duplicated entrys. An example of instructor relation would be, suppose name and department is sufficient to identify an unique instructor, then {name, department} will be a candidate key, ID is also a candidate key. If we seperate name and department, both name and department have duplicated entries.

3. *Primary Key*

   Primary Key is a candidate key that is chosen by the designer as a primary means of identifying unique tuples from the relation. An example would be the ID of instructor, as it is underlined, unique, and identified the instructor.

4. *Foreign Key*

   Foreign key is the a field referencing the primary key from other relation, so that value in one table appears on the other table referencing each other. An example would be the dept_name of instructor referencing

department relation.

W4: *Columbia University uses several applications that use databases to run the university. Examples are SSOL and CourseWorks. An alternate approach could be letting students, faculty, admimistrators, etc. use shared Google Sheets to create, retrieve, update and delete information. What are some problems with the shared sheet approach and what functions do DMBS implement to solve the problems.*

Answer:

***Data Redundancy**: It is possible that for both student and faculty to record the student's information which will lead to dupliated data.*

***Format Inconsistency**: There is no constrained format for google sheets, which entering information by different people lead to inconsistent data format. For example,someone will enter their date of birth as 1/1/2022, while others uses the format 2022 Jan 1st, or someone will miss important information such as entering their major.*

***Security**: Everyone is able to add and delete data, which if someone changed the information of other students without consent, this is troublesome.*

***Data Query**: Google sheets does not allow complicated query accessing, for example if I want to select all students who have same major or same instructor, it is impossible to process automatically and slow to query.*

***Simutaneous BroadCast**: When multiple people send updating request simutaneously, it will reflect to everyone at the same time.*

***Network congestion**: When too many people sending request at the same time, it may cause network crash because google sheet may not be able to handle that many request at same time.*

DBMS allows defining data structures which constrains the format of the data, and remove any data redundancy. DBMS also allows database access language allowing for query data. Security management of DBMS allows database administrator to define who can acess the data and what data can be acessed. In addition, many modern DBMS allows multi user control, which allows multipe users acessing data at the same time without congestion.

***W5:** The relational algebra is is closed under the operators. Explain what this means and give an example.*

Answer:

Relation algebra consists of a set of operations that take one or two relations as input and produce a new relation as the result. Since the output is a relation, this allows relational algebra to do nested expressions, meaning closed under operators. An example would be the following:

$$\Pi_{ID}(\delta_{depart\_name="biology"}(instructor))$$

Instead of putting a name in projection, we can directly put an relational algebra expression inside.

W6: *SQL is a declarative data manipulation language. What are some pros and cons of declarative DMLs relative to procedural DMLs?*

Answer:

Pros:

1. *Declarative DMLs requires user to specify what data are needed without specifying how to get the data while Procedural DMLs requires user to specify how to retrieve the data.*
2. *Declarative DMLs are easier to use than Procedual DMLs, since user do not need to know how to get the data and the system can figure an efficient way of acessing data.*
3. *The code is more concise and readable.*

Cons:

1. *Declarative DML is not expressive enough for more complex functions.*
2. *Since declaritive DML let system figure out an way of execution, the performance may improve when procedural DML have more control.*
3. *Declarative DML have no control for excution process.*

W7: *Briefly explain the concepts of database schema and instance. Give an example from the sample database associated with the recommended textbook.*

Answer:

Schema*: predefined-structure of database. An instructor table defining the ID, name, depart_name and salary will be the schema.*

Instance*: collection of information stored in database at a particular moment. An exmaple would be suppose an instructor table have 50 rows today, this will be an instance, tomorrow, we may have 100 rows, which is also an instance.*

W8: *What is the semi-structured data model and how is it different from the relational data model?*

Answer:

Semi-structured Data: Data which inidividual items of the same type may have different sets of attributes. It is structured but does not have a fixed schema. Example would be JSON and XML.Semi-structured data model permits individual data to have different sets of attributes with same type, while relational data model must have the same set of attributes of a particular type. Semi-structured data model does not have a predefined schema, while relational data model have tables with fixed number of attributes and each contains an atomic value.

W9: *Some of the Columbia University databases/applications represent the year/semester attribute of a section in the form "2023_2." Where the first for characters are the academic year, and the last character is the semester (1, 2, or 3). The data type for this attribute might be* `char(6)` *or* `str.` *Explain the concepts of domain and atomic domain and the difference from type using this example.*

Answer:

Domain is the data type definition for the attribute. A domain is atomic if elements of the domain are considered to be indivisible units. If we see year/semster as one single element, it is considered as atomic domain. While if we see year and semester as 2 seperate element, it is not considered as atomic. char(6) is a length specified data type while string is used to represent variable-length strings, which can have any length and can change dynamically,

---

Relational Algebra

R1: *Defining relations: The following is the SQL DDL for the* `db_book.classroom` *table.*

```
create table if not exists db_book.classroom
(
    building    varchar(15) not null,
    room_number varchar(7)  not null,
    capacity    decimal(4)  null,
    primary key (building, room_number)
);
```

Using the notation/format from the lecture slides, provide the corresponding relational model/algebra definition. You do not need to worry about data types, null/not null, ...

Answer:

classroom(<u>building</u>, <u>room number</u>, capacity)

S1: *This is a sample of the format for answering the relational algebra questions. Your answer will contain:*

1. *A markdown cell with the relational algebra statement.*
2. *A screen capture of the execution.*

You will use the RelaX calculator with the schema associated with the book.

Write a relational algebra statement that produces a relation with the columns:

- *section.course_id*
- *section.sec_id*
- *section.semester*
- *section.year*
- *section.building*
- *section.room_number*
- *classroom.capacity*

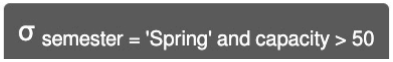
And only contains tuples from the `Spring` *semester and a* `classroom.capacity > 50.`

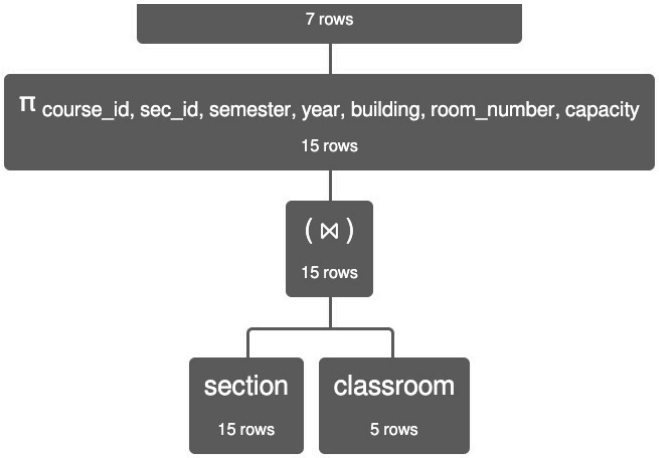
Answer:

Algebra statement.

```
σ semester='Spring'∧capacity>50
    (π course_id, sec_id, semester, year, building, room_number, capacity
        (section ⋈ classroom)
        )
```

Screen capture:





$$\sigma_{\text{semester = 'Spring' and capacity > 50}} (\pi_{\text{course\_id, sec\_id, semester, year, building, room\_number, capacity}} (\text{section} \bowtie \text{classroom}))$$

Execution time: 2 ms

section.course_id	section.sec_id	section.semester	section.year	section.building	section.room_number	classroom.capacity
'CS-101'	1	'Spring'	2010	'Packard'	101	500
'CS-190'	1	'Spring'	2009	'Taylor'	3128	70
'CS-190'	2	'Spring'	2009	'Taylor'	3128	70
'CS-319'	2	'Spring'	2010	'Taylor'	3128	70
'EE-181'	1	'Spring'	2009	'Taylor'	3128	70
'FIN-201'	1	'Spring'	2010	'Packard'	101	500
'MU-199'	1	'Spring'	2010	'Packard'	101	500

R2: *Write a relational algebra expression that returns a relation of the form:*

- *section.course_id*
- *section.sec_id*
- *section.semester*
- *section.year*
- *teaches.ID*
- *instructor.name*
- *course.credits*

The relation contains courses that earn at least 4 credits.

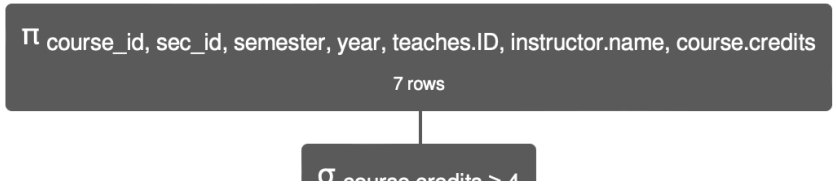
Answer:

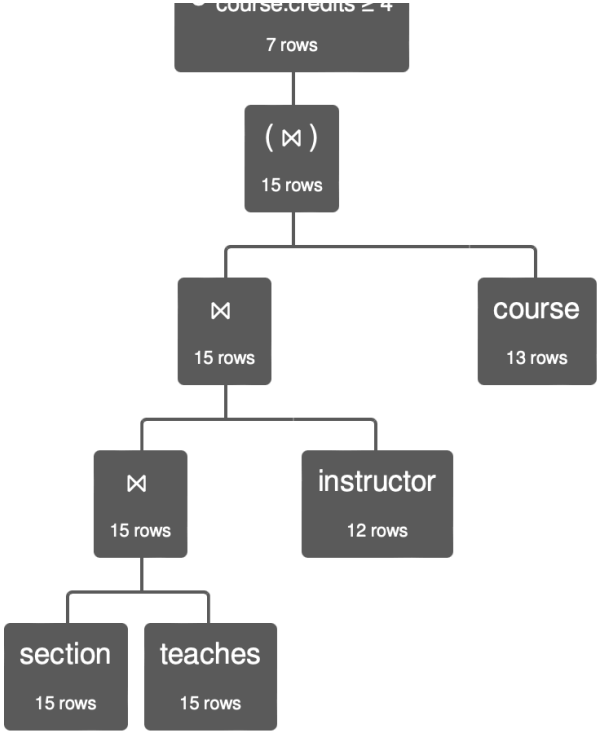
Relational algebra:

π course_id, sec_id, semester, year, teaches.ID, instructor.name, course.credit

```
(σ course.credits >= 4
    (section ⋈ teaches ⋈ instructor ⋈ course))
```

Screen capture:





$$\pi \text{ course\_id, sec\_id, semester, year, teaches.ID, instructor.name, course.credits } (\sigma \text{ course.credits} \geq 4 (((\text{section} \bowtie \text{teaches}) \bowtie \text{instructor}) \bowtie \text{course}))$$

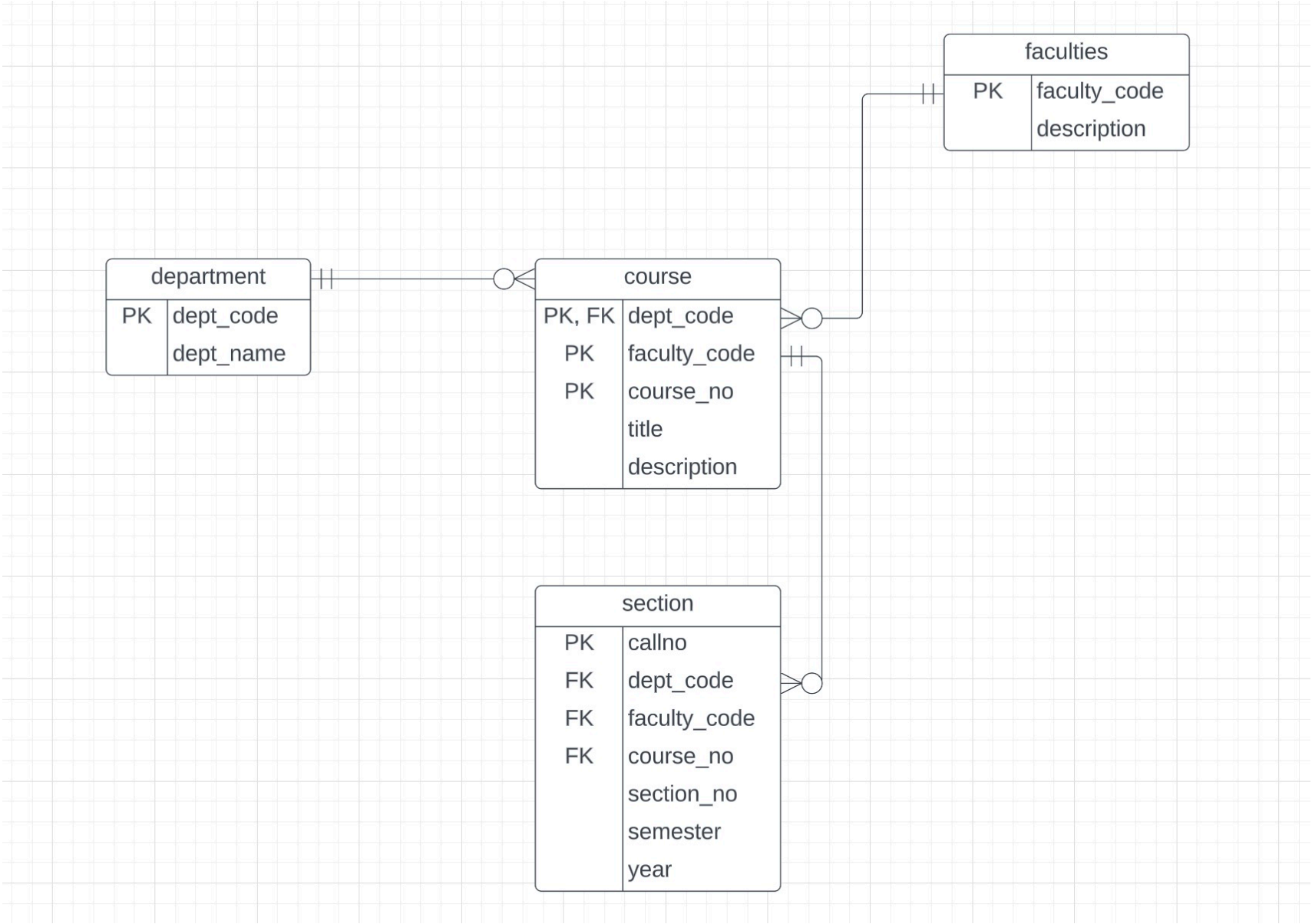
Execution time: 5 ms

section.course_id	section.sec_id	section.semester	section.year	teaches.ID	instructor.name	course.credits
'BIO-101'	1	'Summer'	2009	76766	'Crick'	4
'BIO-301'	1	'Summer'	2010	76766	'Crick'	4
'CS-101'	1	'Fall'	2009	10101	'Srinivasan'	4

'CS-101'	1	'Spring'	2010	45565	'Katz'	4
'CS-190'	1	'Spring'	2009	83821	'Brandt'	4
'CS-190'	2	'Spring'	2009	83821	'Brandt'	4
'PHY-101'	1	'Fall'	2009	22222	'Einstein'	4

Data Modeling

ER Diagram to SQL DDL


faculties
PK
faculty_code
description
department
PK
dept_code
dept_name
course
PK, FK
dept_code
PK
faculty_code
PK
course_no
title
description
section
PK
callno
FK
dept_code
FK
faculty_code
FK
course_no
section_no
semester
year

- *We covered the preceding ER diagram in class on Friday, 27-JAN.*
- *In the cells below, write and execute the* `create table` *statements to produce an SQL schema that realizes the diagram.*
- *The primary focus is on correctly implementing keys. You should make reasonable assumptions about column data types,* `not null` *, etc.*
- *The next cell provides one example to help you get started.*

In [20]:
```
%%sql

use w4111_hw1;

drop table if exists departments;

create table W4111_HW1.departments
(
    department_code char(4)     not null,
    department_name varchar(64) not null,
    constraint departments_pk
        primary key (department_code)
);
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
0 rows affected.
```

Out[20]:
```
[]
```

In [21]:
```
%%sql

drop table if exists W4111_HW1.simple_course;

create table if not exists W4111_HW1.simple_course
(
    dept_code    varchar(4)   not null,
    faculty_code char(2)      not null,
    course_no    char(4)      not null,
    title        varchar(64)  null,
    description  varchar(512) null,
    primary key (dept_code, faculty_code, course_no),
    constraint simple_course_departments_null_fk
        foreign key (dept_code) references W4111_HW1.departments (department_code)
);
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
```

Out[21]:
```
[]
```

In [22]:
```
%%sql
drop table if exists faculties;
drop table if exists course;
drop table if exists section;

create table W4111_HW1.faculties
(
    faculty_code     char(8)      not null,
    description      varchar(256) not null,
    primary key (faculty_code)
);


create table W4111_HW1.course
(
    department_code  varchar(4)      not null,
    faculty_code     char(4)      not null,
    course_no        char(8)          not null,
    title            varchar(64)  not null,
    description      varchar(512) not null,
    foreign key(department_code) references departments(department_code),
    foreign key(faculty_code) references faculties(faculty_code),
    primary key (department_code, faculty_code, course_no)
);

create table W4111_HW1.section
(
    callno           char(8)      not null,
    department_code  varchar(4)   not null,
    faculty_code     char(8)      not null,
    course_no        char(8)      not null,
    section_no       varchar(8)   not null,
    semester         varchar(8)   not null,
    year             int(4)       not null,
    foreign key(department_code, faculty_code, course_no)
        references course(department_code, faculty_code, course_no),
    primary key(callno)
);
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
```

Out[22]: []

ER Modeling

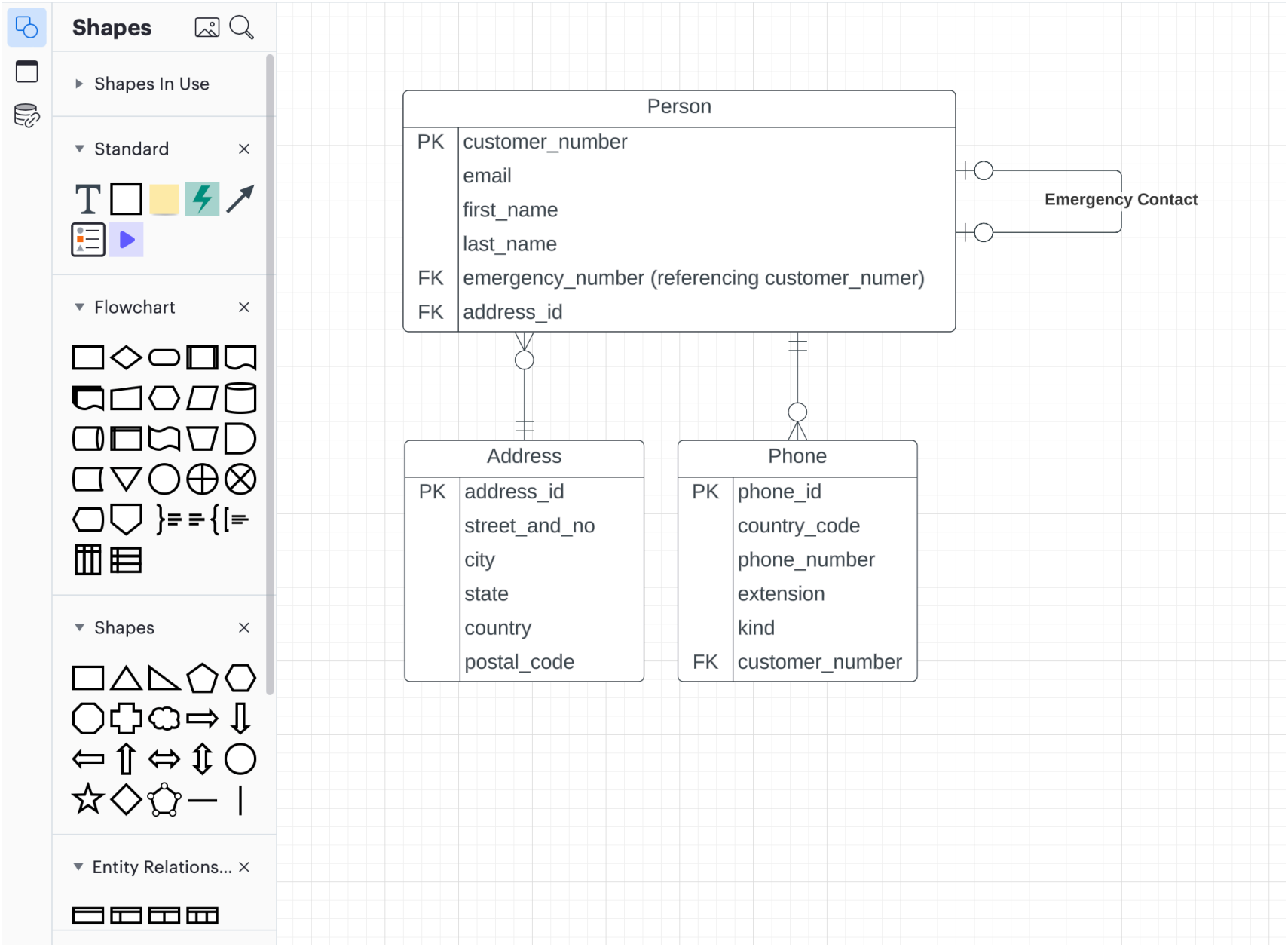
- *Consider a personal profile for a customer using an application.*
- *There are three entity types.*
  1. `Person` *with attributes:*
     - `customer_number` *(Uniquely identifies a Person))*
     - `email`
     - `first_name`
     - `last_name`
  2. `Address` *with attributes:*
     - `address_id` *(Uniquely identifies an address)*
     - `street_and_no`
     - `city`
     - `state`
     - `country`
     - `postal_code`
  3. `Phone` *with attributes:*
     - `phone_id`
     - `country_code`

    - *`phone_number`*
    - *`extension`*
    - *`kind` (e.g. 'Home', 'Mobile', 'Work')*

- *A `Person` has the following relationships:*
    - *Exactly one `Address`. There may be addresses not associated with a `Person`.*
    - *0 or 1 relationships to another person who is the `emergency contact.`*
    - *0, 1 or many phone numbers. A `Phone` is associated with exactly one `Person`.*

Use Lucidhart to draw a logical model diagram and include a screen capture below. We used Lucidchart in lecture on Friday. You can register for a free account.

- *You can replace the following diagram with your screen capture. Note the instructions for how to enable ER shapes.*
- *You may add explanatory notes. I did an example in lecture.*


Shapes
Shapes In Use
Standard
Flowchart
Shapes
Entity Relations...
Person
PK customer_number
email
first_name
last_name
FK emergency_number (referencing customer_numer)
FK address_id
Emergency Contact
Address
PK address_id
street_and_no
city
state
country
postal_code
Phone
PK phone_id
country_code
phone_number
extension
kind
FK customer_number

SQL

- *Use the `db_book` database/schema that you created in HW 1 for these questions.*

SQL1:

Write and execute SQL to produce the table below. The query uses `student` and `advisor` tables.

Out[7]:

ID	name	dept_name	tot_cred	s_ID	i_ID
12345	Shankar	Comp. Sci.	32	12345	10101
00128	Zhang	Comp. Sci.	102	00128	45565
76543	Brown	Comp. Sci.	58	76543	45565

In [39]:
```
%%sql

select *
from db_book.student
inner join db_book.advisor
on db_book.student.ID = db_book.advisor.s_ID
where db_book.student.dept_name = 'Comp. Sci.';
```

```
 * mysql+pymysql://root:***@localhost
3 rows affected.
```

Out[39]:

ID	name	dept_name	tot_cred	s_ID	i_ID
12345	Shankar	Comp. Sci.	32	12345	10101
00128	Zhang	Comp. Sci.	102	00128	45565
76543	Brown	Comp. Sci.	58	76543	45565

SQL2: *Produce the table below. The query uses* `student` *and contains tuples for students with less than 50* `tot_cred`

Out[9]:

ID	name	tot_cred
12345	Shankar	32
45678	Levy	46
55739	Sanchez	38
70557	Snow	0

In [40]:

```
%%sql

select ID, name, tot_cred
from db_book.student
where tot_cred < 50;
```

```
 * mysql+pymysql://root:***@localhost
4 rows affected.
```

Out[40]:

ID	name	tot_cred
12345	Shankar	32
45678	Levy	46
55739	Sanchez	38
70557	Snow	0

In [ ]: