COMS W4111: Introduction to Databases
Spring 2023, Sections 002

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

</span> </center></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 [ ]:
```
# Print your name, uni, and track below

name = "Yixuan Wang"
uni = "yw3928"
track = "Programming Track"

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

```
Yixuan Wang
yw3928
Programming Track
```

Testing Environment

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

You may need to change passwords.

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

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

In [ ]:
```
%sql mysql+pymysql://root:12345678@localhost
```

Out[ ]:
```
'Connected: root@None'
```

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

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

Out[ ]:

ID	name	dept_name	tot_cred
12345	Shankar	Comp. Sci.	32

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

In [ ]:
```
engine = create_engine("mysql+pymysql://root:12345678@localhost")
```

In [ ]:
```
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", "12345678")
```

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

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

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

Out[ ]:
```
[{'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 [ ]:
```
import pandas
```

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

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

In [ ]:
```
df
```

Out[ ]:

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

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

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

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

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

Out[ ]:
```
3
```

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

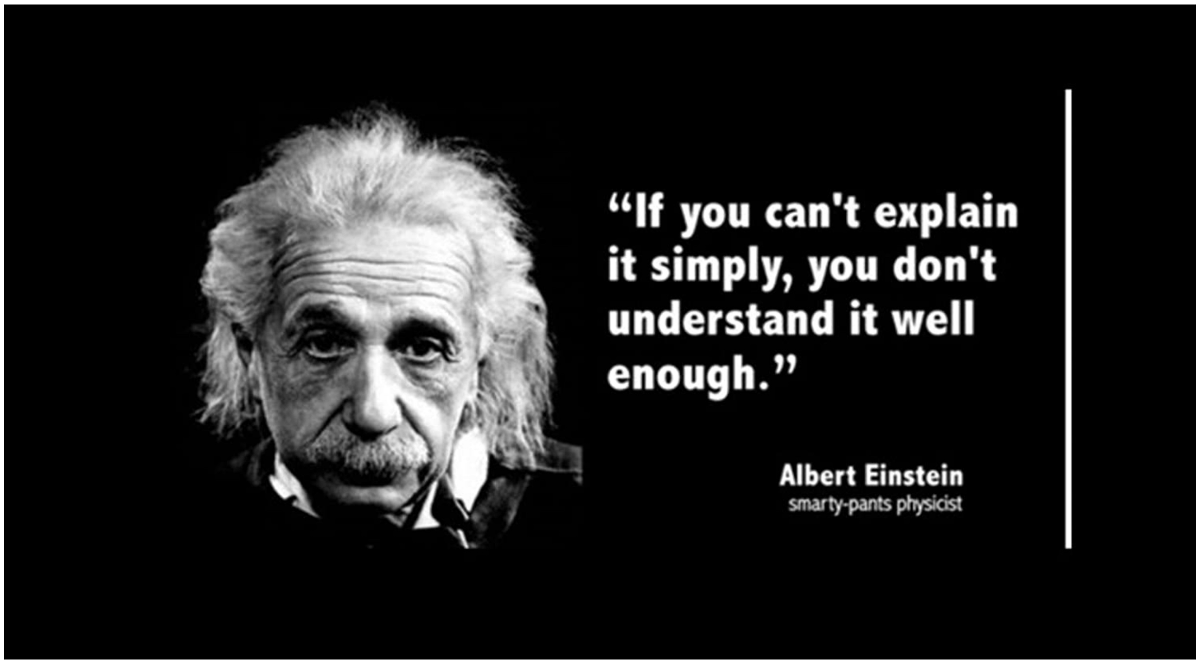
```
 * mysql+pymysql://root:***@localhost
3 rows affected.
```

Out[ ]:

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.


"If you can't explain it simply, you don't understand it well enough."
Albert Einstein
smarty-pants physicist

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

Answer:

- ***Application programmers*** *write application programs for the use of naive users and sophisticated users, explained as below.*
- ***Naive users*** *interact with the system by programmer-defined user interfaces, typically a forms interface. Such an interface allows the naive user to fill in fields of the form.*
- ***Sophisticated users*** *send their requests with a database query language or through tools, including data analysis software.*
- ***Database Administrator*** *have control of the data and the programs that access the data. They are responsible for creating/modifying the original database schema definitions, editing parameters for storage structure, granting access authorization and regular database maintenance.*

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

Answer:

- ***Structured data*** *is highly organized data and easily managed by a database query language. Examples include data downloaded from Yahoo Finance satisfies the 'rows and columns' format and is easily modified by a database query language.*
- ***Unstructured data*** *can hardly be modified or analyzed through data tools and database query languages. Example include data mined from Linkedin to identify preferences and locations of job applicants.*

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

1. *Super Key*
2. *Candidate Key*
3. *Primary Key*
4. *Foreign Key*

Answer:

- *A* ***super key*** *is a set of one or more attributes that uniquelt defines a tuple in the relation. For example, the ID attribute in the instructor relation allows us to uniquely identify one instructor.*
- *A* ***candidate key*** *is the minimal set of super keys. For example, if ID and name is a super key for the instructor relation, then ID is the candidate key.*
- *A* ***primary key*** *is a candidate key that is chosen to be the main method of uniquely identifying tuples in a relation. For example in the classroom relation, the primary key attributes are building and room_number.*
- *A* ***foreign key*** *is the attribute set A of relation P to the primary key B of relation Q represents that the value of A of relation P is also the value of the primary key B of relation Q. For example, dept_name in relation instructor is a foreigh key from department.*

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:

Potential problem:

1. *Failed update. This problem may occur due to futher change of the email for students/staff, or the email of an alumni is overwritten by a new student. DMBS solve this problem by adding primary keys to each entity that ensures database administrators to uniquely define students, faculty, admimistrators, etc.*
2. *Data Accessibility. Shared sheet means every person affiliated with the university have the access to the database, which means everyone have the accessibility to modify the data. DMBS solve this problem by only a selection of personnels are granted access to the database.*

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

Answer:

This means the output from one operation can be the input of another operations - the operand and the outputs are relations. This is similar to integers are closed under addition and subtraction, but are not closed under division.

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 are usually easier to learn and use than are procedural DMLs.*
2. *Declarative DMLs can generate sorted lists from predefined features. Cons:*
3. *The machine efficiency is lower, compared to procedural DMLs.*

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

Answer:

- *An **instance** of the database is the entire set of information stored in the database in a specific time. For example, a collection of the information regarding required attributes* *of the instructor record at 2022-12-28 is an instance of the database.*

- *A **schema** is the structure of the design for the database. An example might be that instructor record is designed to include attributes such as instructor_id, courses, student_name.*

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

Answer:

The semi-structured data model specify the data where different sets of attributes are allowed to exist within an entity of the data of the same type.

This is different from the relational data model which requires that every dat item of the same type should have identical set of attributes.

***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:

*The **domain** of one attribute is the set of possible values. An **atomic domain** is a domain which contains elements that cannot be divided. In the example above, the value of the year/semest attribute is not indivisible - meaning the values, such as '2023-2' have subparts.*

Answer:

Relational Algebra

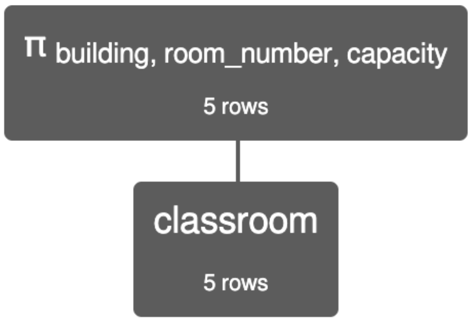
***R1:** Defining relations: The following is the SQL DDL for the `db_book.classroom` table.*

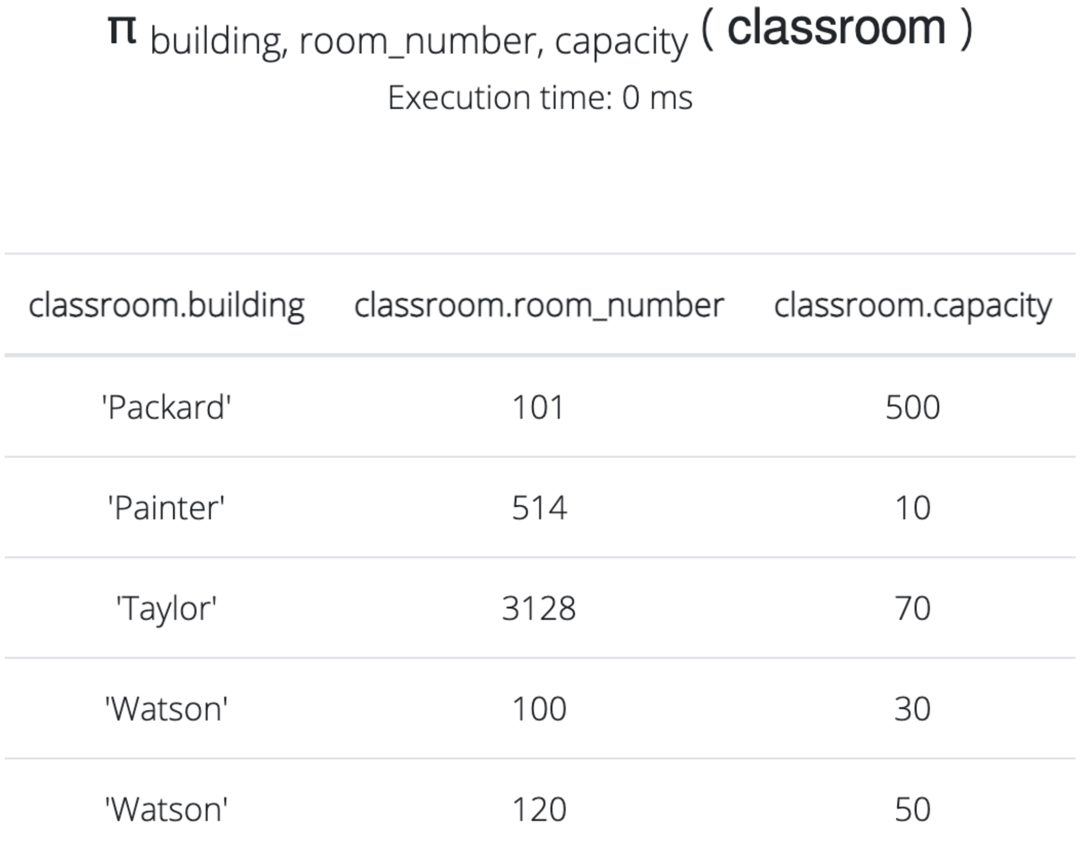
```
create table if not exists db_book.classroom
(
    building    varchar(15) not null,
    room_number varchar(7)  not null,
    capacity    decimal(4)  null,
    primary key (building, room_number)
);
```

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:

π building, room_number, capacity (classroom)





π building, room_number, capacity (classroom)
Execution time: 0 ms

classroom.building	classroom.room_number	classroom.capacity
'Packard'	101	500
'Painter'	514	10
'Taylor'	3128	70
'Watson'	100	30
'Watson'	120	50

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:

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

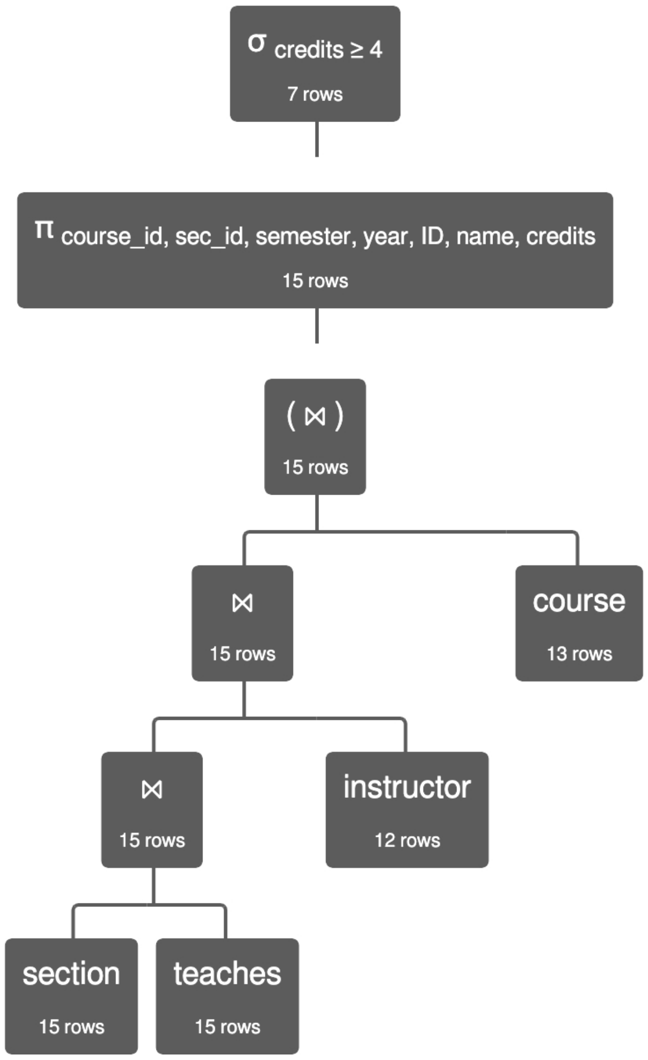
- *section.course_id*
- *section.sec_id*
- *section.semester*
- *section.year*
- *teaches.ID*
- *instructor.name*
- *course.credits*

The relation contains courses that earn at least 4 credits.

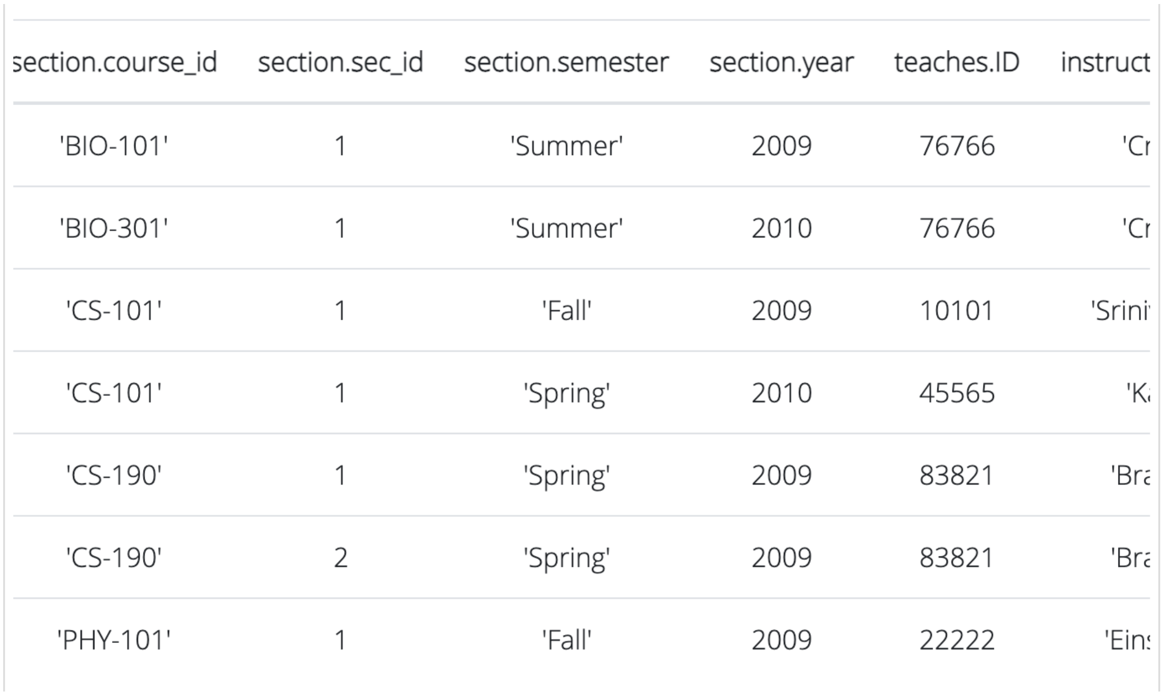
Answer:

Relational algebra: σ credits ≥ 4 (π course_id, sec_id, semester, year, ID, name, credits (section ⋈ teaches ⋈ instructor ⋈ course))

Screen capture:

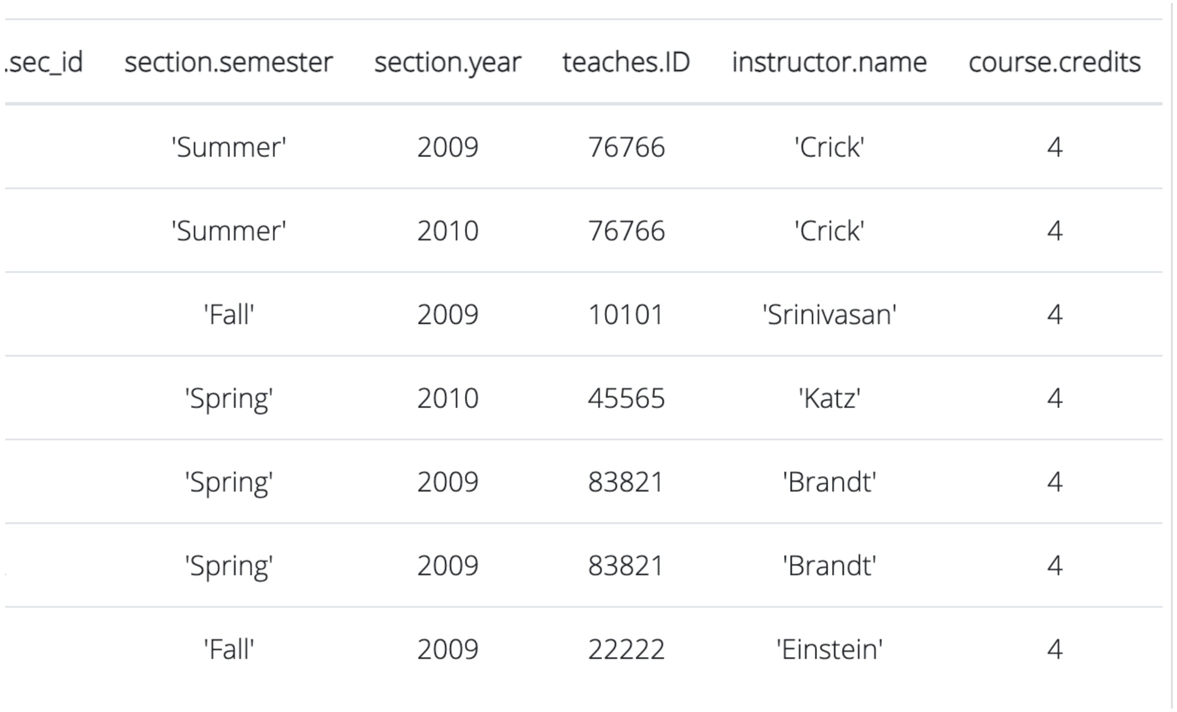


Result data part 1:



ection.course_id	section.sec_id	section.semester	section.year	teaches.ID	instruct
'BIO-101'	1	'Summer'	2009	76766	'Cr
'BIO-301'	1	'Summer'	2010	76766	'Cr
'CS-101'	1	'Fall'	2009	10101	'Srini
'CS-101'	1	'Spring'	2010	45565	'K
'CS-190'	1	'Spring'	2009	83821	'Bra
'CS-190'	2	'Spring'	2009	83821	'Bra
'PHY-101'	1	'Fall'	2009	22222	'Ein

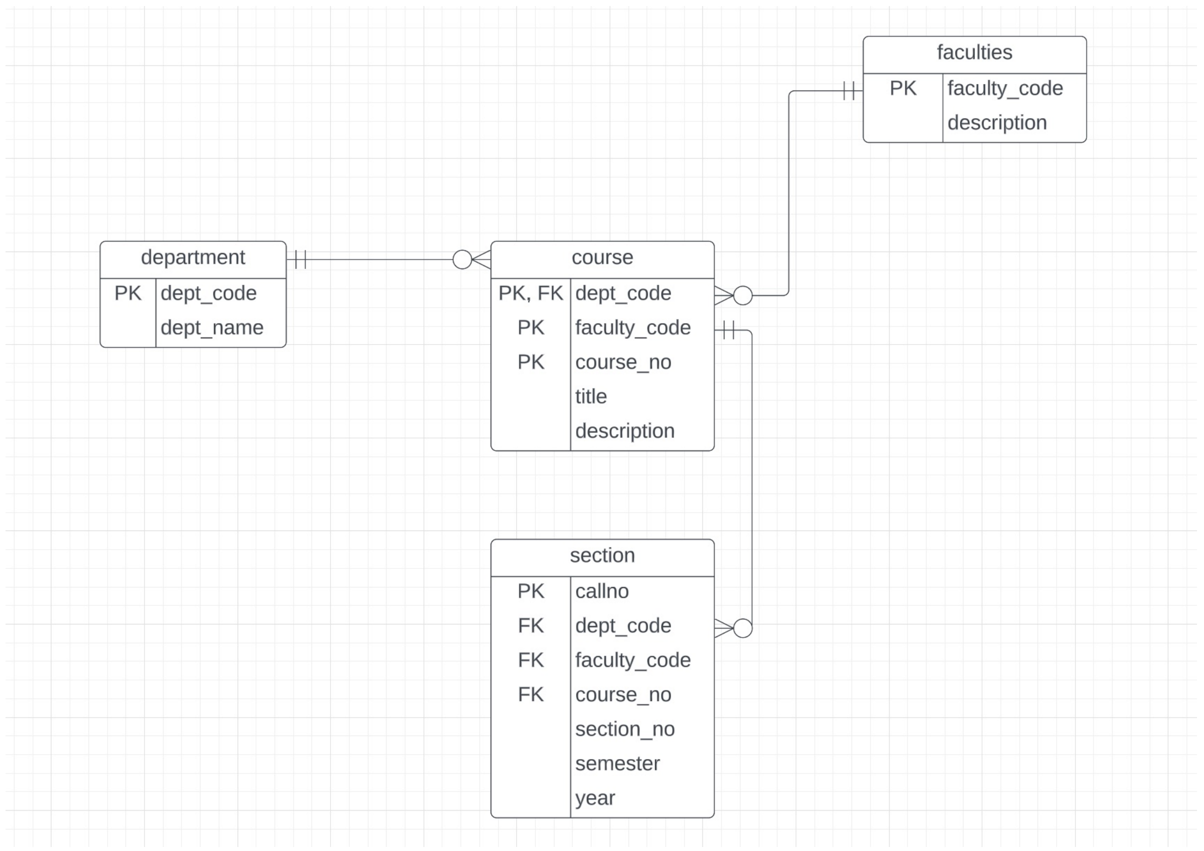
Result data part 2:



.sec_id	section.semester	section.year	teaches.ID	instructor.name	course.credits
	'Summer'	2009	76766	'Crick'	4
	'Summer'	2010	76766	'Crick'	4
	'Fall'	2009	10101	'Srinivasan'	4
	'Spring'	2010	45565	'Katz'	4
	'Spring'	2009	83821	'Brandt'	4
	'Spring'	2009	83821	'Brandt'	4
	'Fall'	2009	22222	'Einstein'	4

Data Modeling

ER Diagram to SQL DDL



- *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 [ ]:
```
%%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_cc
);
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
(pymysql.err.OperationalError) (3734, "Failed to add the foreign key constrain
t. Missing column 'department_code' for constraint 'simple_course_departments_
null_fk' in the referenced table 'departments'")
[SQL: 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_c
ode)
);]
(Background on this error at: https://sqlalche.me/e/14/e3q8)
```

In [ ]:
```
%%sql

use w4111_hw1;
drop table if exists W4111_HW1.simple_course;
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.
0 rows affected.
```

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

In [ ]:

In [ ]:
```
%%sql

drop table if exists section;
create table section
(
    callno       char(5)    not null,
    dept_code    varchar(4) not null,
    faculty_code char(2)    not null,
    course_no    char(4)    not null,
    section_no   int(3)     not null,
    semester     char(4)    not null,
    year         int(4)     not null,
    constraint section_pk
        primary key (callno),
    constraint section_fk
        foreign key (dept_code, faculty_code, course_no) references simple_cou
);
```

```
drop table if exists faculties;
create table faculties
(
    faculty_code char(4)     not null,
    description varchar(64) not null,
    constraint faculties_pk
        primary key (faculty_code)
);
ALTER TABLE simple_course
ADD FOREIGN KEY (faculty_code) REFERENCES faculties(faculty_code);
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
(pymysql.err.OperationalError) (1824, "Failed to open the referenced table 'si
mple_course'")
[SQL: create table section
(
    callno       char(5)    not null,
    dept_code    varchar(4) not null,
    faculty_code char(2)    not null,
    course_no    char(4)    not null,
    section_no   int(3)     not null,
    semester     char(4)    not null,
    year         int(4)     not null,
    constraint section_pk
        primary key (callno),
    constraint section_fk
        foreign key (dept_code, faculty_code, course_no) references simple_cou
rse(dept_code, faculty_code, course_no)
);]
(Background on this error at: https://sqlalche.me/e/14/e3q8)
```

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`

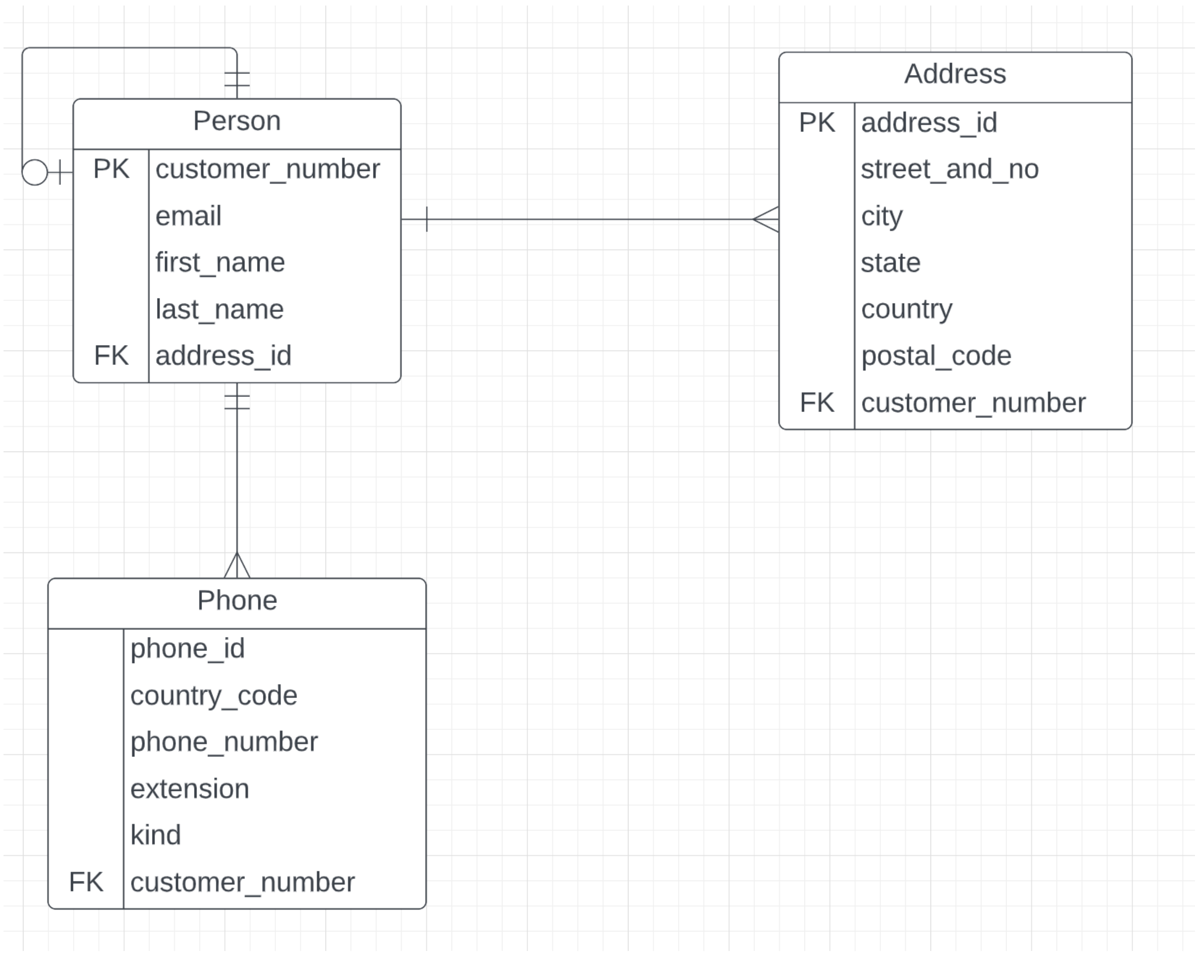
- `country_code`
- `phone_number`
- `extension`
- `kind` (e.g. 'Home', 'Mobile', 'Work')

- *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.*

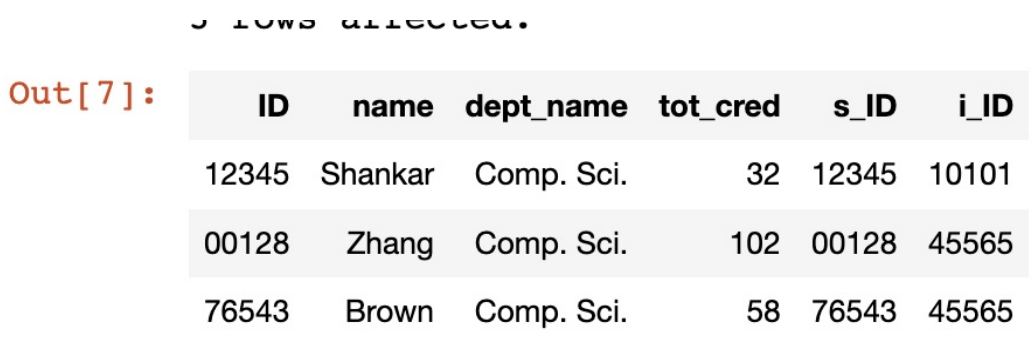


SQL

- *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 [ ]:

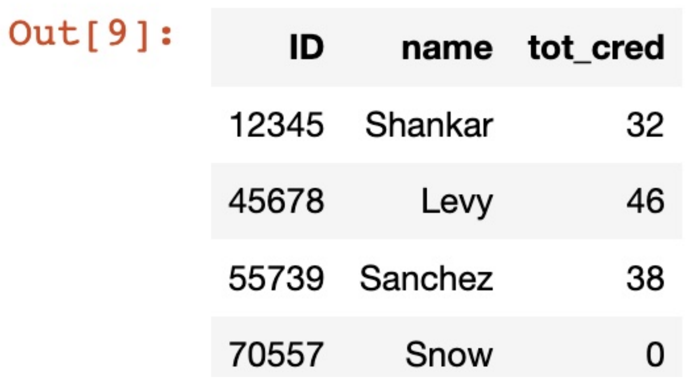
```
%%sql
select ID, name, dept_name, tot_cred, s_ID, i_ID from db_book.advisor join db_
where (ID = 12345 and i_ID = 10101) or (ID = 00128 and i_ID = 45565 and s_ID =
or (ID = 76543 and i_ID = 45565 and s_ID = 76543);
```

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

Out[ ]:

ID	name	dept_name	tot_cred	s_ID	i_ID
00128	Zhang	Comp. Sci.	102	00128	45565
12345	Shankar	Comp. Sci.	32	12345	10101
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 [ ]:

```
%%sql

select ID, name, tot_cred from db_book.student
where (ID = 12345 or tot_cred = 32)
or (ID = 45678)
```

```
OR (ID = 55739)
OR (ID = 70557);
```

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

Out[ ]:

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

In [ ]: