COMS W4111: Introduction to Databases Sections 002, V02 Fall 2022

Homework 1, Part 1

Introduction/Overview

Please consult CourseWorks and ED for submission instructions.

To convert the notebook to PDF, do one of the following:

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

Due date: 10/3 (Mon) at 11:59PM 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!

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 = "Tong Wu"
uni = "tw2906"
track = "Programming"

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

```
Tong Wu
tw2906
Programming
```

Homework Overview

Note: The track specific sections will come out in a couple of days.

The homework has 3 sections:

1. **All** students must complete *Common Questions and Tasks.*

2. Students on the **Non-Programming Track** complete the section *Non-Programming Track Tasks.*

3. Students on the **Programming Track** complete the section *Programming Track Tasks.*

Common Questions and Tasks

Written Questions

Questions 1: DML and DDL

- Briefly explain the concepts of data definition language and data manipulation language.
- Give one example of a SQL DDL statement and a SQL DML statement.

Answer:

The data definition language is used to define the structure of the database. It is used to create the database and define attributes.

The example of SQL DDL is:

%%sql CREATE DATABASE f22_hw2;

The data manipulation language is used to add or update data in the database.

The example of SQL DML is:

%sql SELECT * from f22_hw2.person;

Questions 2: Database Management System Functions

- We have seen that we can manipulate data in CSV files using either Google Sheets/Excel/Numbers or a DBMS.
- Give three benefits of a DBMS over spreadsheet programs for a scenario in which many users are editing a very large CSV dataset.

Answer:
Easy to share the CSV files to the data through webpage.
Can be edited by few peoples at the same time.
Flexibility, easy to manipulate, visualize the data.

Questions 3: Types of Data

- Briefly explain the concepts of structured, semi-structured and unstructured data.
- Give an example of each type of data.

Answer:
Structured data is represented by rows and columns in a database table. All rows have same set of attributes (columns). There are some relations between tables. The example for this type of data is sql.
Semi-structured data allows same type of data has difference attributes. The example for this type of data is JSON and XML.
Unstructured data has no structure in data, are not defined their attrubutes and grouped. The example of this type of data is PDF and image.

Questions 4: Data Abstraction Levels

- Briefly explain the concepts of physical level, logical level and view level in databases.

Answer:
Physical level has the lowest level of data modeling, with the least abstraction to describe how data is stored. It is often used by developer to do some technical implementation.
Logic level use few abstractive relatives to show the structure of data. It is often used by analysts to clarification the structure of data.
View level use the highest level of abstraction to describe the data structure, by only providing the name and relationship of entities. It is often used by stakeholders to communication and definition of terms and roles.

Questions 5: Instance and Schema

- Briefly explain the concepts of database schema and data instance.
- Would you use DML or DDL statements to create and update a schema?

Answer:
Database schema is the logical structure of the database,which can be written in a programming language.
Data instance describes the collection of information stored in the database at a particular moment.
Use DDL to create and update the schema.

Questions 6: Declarative versus Procedural Languages

- Briefly explain the difference between a procedure data manipulation language and a procedural data manipulation language.
- Briefly explain why the *relational algebra* is declarative and python is procedural.

Answer:
The procedure DML do not require user to specify how to get data, while procedural DML need user to specify.

Beacuse the python only need user to declare where to get data, and specify the needs of the data. Relational algebra need user to declare how to get data from a specific place by using codes or library.

Questions 7: Application Architectures

- Briefly explain the concepts of two-tier and three-tier application architectures.
- Are Jupyter notebooks two-tier or three-tier applications?

Answer:
The two-tier application architectures is partitioned into two parts, the application and database system. The application as the front-end directly send query language statements to the database server.
The three-tier application architecture is partitioned into three parts, the application, application server and database system. The application communicates with the application server, and the query statements are sent from application server to the database system.
Jupyter notebook should be a three-tier application, since the jupyter notebook are often used web environment, and it can be connected remotely.

Questions 8: User and Tools

- We have used several tools so far in class. Two examples are Jupyter Notebooks and DataGrip.
- Which tool would a database administrator use?
- Which tool would a sophisticated database user use?

Answer:
Database administrator should use Datagrip, sophisticated database user should use Jupyter Notebooks.

Questions 9: Relational Algebra

- Briefly explain the concept of the relational algebra being *closed* under the operations/operators.
- Give an example.

Answer:

Relation algebra use some actions to calculate a relation from two or more relations. Relation algebra's operands and results are both relations, so it can be called as closure.
For example, the relational algebra statment

$$\sigma_{dept-name="Physics"}(instructor)$$

will get several results of instructors who is under physics department, which is still in the set of instructor. It should be proved that the relational algebra is closed.

Questions 10: Good and Evil

- Briefly explain why Thursday and Friday (22-SEP, 23-SEP) were bad days for Professor Ferguson.
- Hint: It has something to do with baseball and evil triumphing over good.

Answer: Put your answer in the markdown cell.

Answer:
Because the Red Sox lose the MLB games to the New York Yankee in those days.

Data Modeling

ER Diagrams

Using Lucidchart draw an example of a logical ER model using Crow's Foot notation for the following scenario:

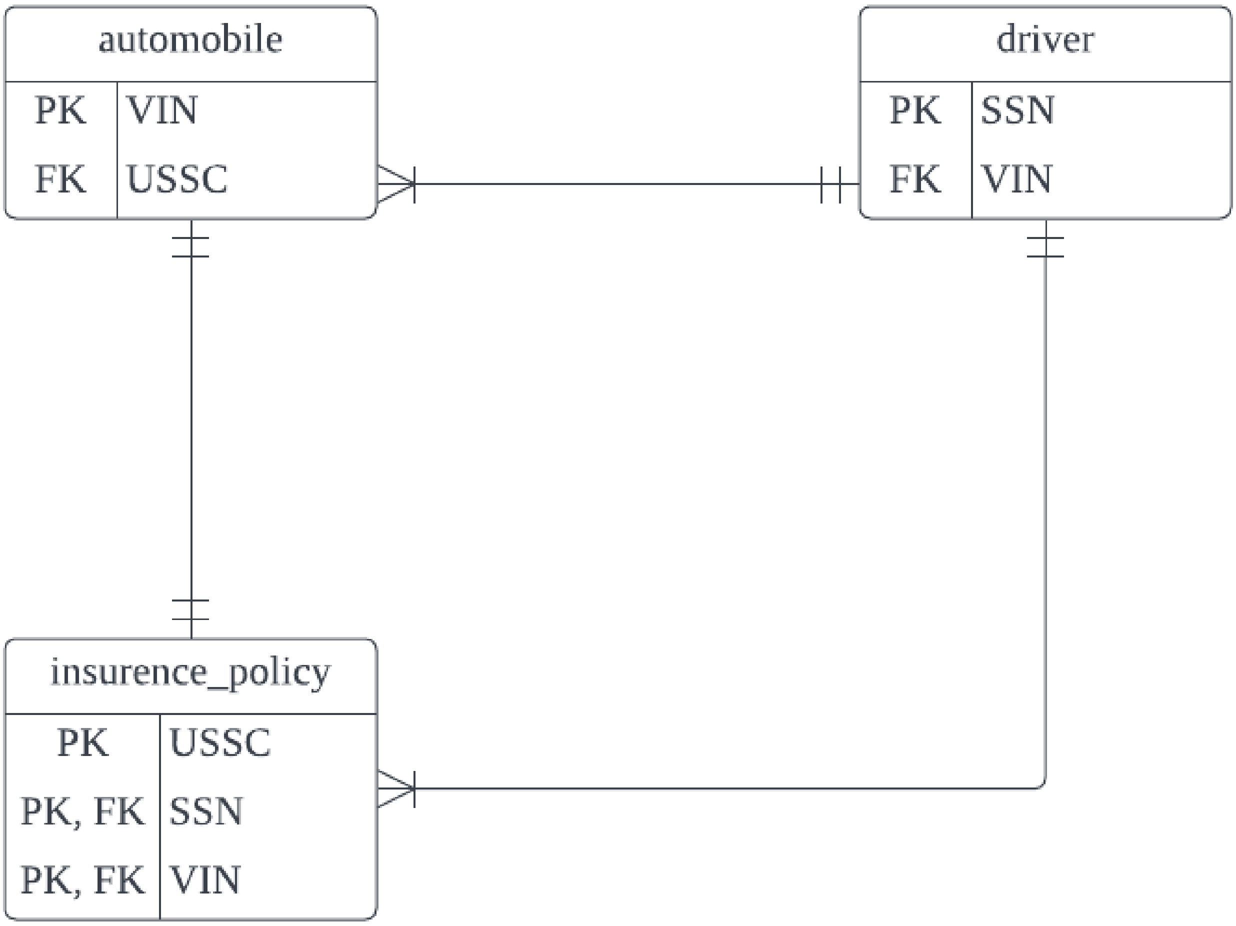
- Entity Types:
  - `automobile` (primary key is VIN)
  - `driver` (primary key is SSN)
  - `insurance_policy` (The primary key is a two letter US state code is a number. This information is just hints to help you with your logical modeling definitions).

- The relationships are:
  - A `driver` is the driver for 0, 1 or many `automobiles` .
  - The relationship between `insurance_policy` and `automobile` is one-to-one. An `automobile` as exactly one `insurance_policy` and vice versa.
  - An `insurance_policy` has exactly one `driver` that is the primary driver.

- Place a screenshot of your ER diagram below.

- We are not concerned about the properties that you choose for your entities that are not part of some key.
- You must correctly label primary key and foreign key attributes using the notation examples from class.

Answer: Please include a screenshot below.

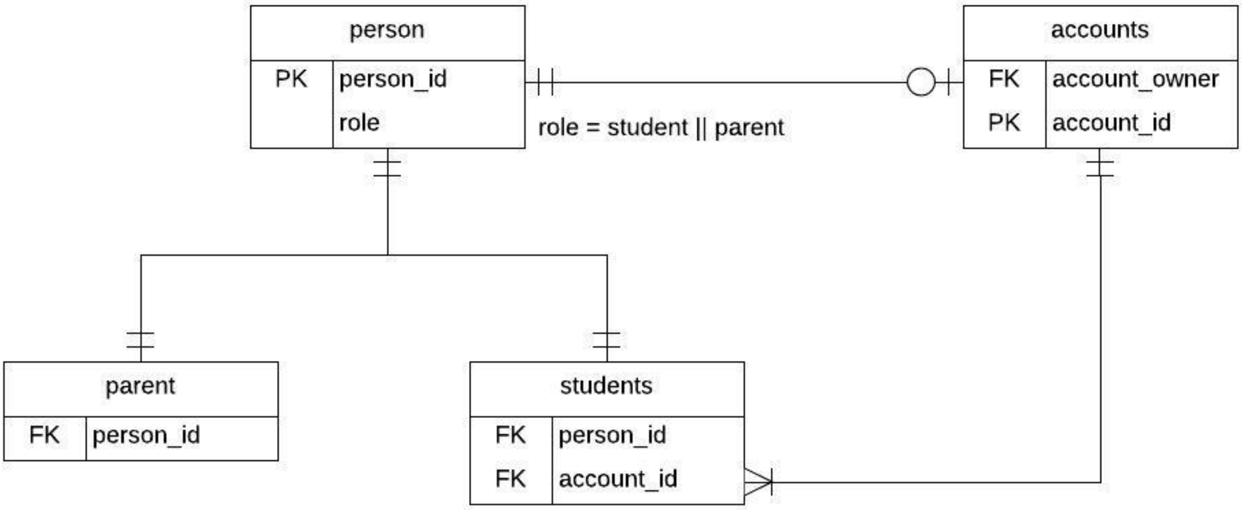


automobile	
PK	VIN
FK	USSC

driver	
PK	SSN
FK	VIN

insurence_policy	
PK	USSC
PK, FK	SSN
PK, FK	VIN

Create Tables



Simple Crow's Foot Diagrm

- The tasks are to write SQL statements that accomplish the following:

1. Create a database `f22_hw2` .
2. Create tables, including primary and foreign keys, for the ER diagram above.

- You must put and execute your SQL below.

In [1]:

```
%load_ext sql
```

In [2]:

```
# Modify the statement below with your MySQL ID and password.
#
%sql mysql+pymysql://root:dbuserdbuser@localhost
```

In [7]:

```
%%sql
drop database f22_hw2;
create database f22_hw2;
create table f22_hw2.person (
    person_id    varchar(8),
    role         enum('parent', 'students'),
    primary key (person_id));
create table f22_hw2.accounts (
    account_owner    varchar(8),
    account_id       varchar(8),
    primary key (account_id),
    foreign key (account_owner) references person(person_id));
create table f22_hw2.parent (
    person_id varchar(8),
    foreign key (person_id) references person(person_id));
create table f22_hw2.students (
    person_id        varchar(8),
    account_id       varchar(8),
    foreign key (person_id) references person(person_id),
    foreign key (account_id) references accounts(account_id));
```

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

Out[7]:

```
[]
```

Relational Algebra

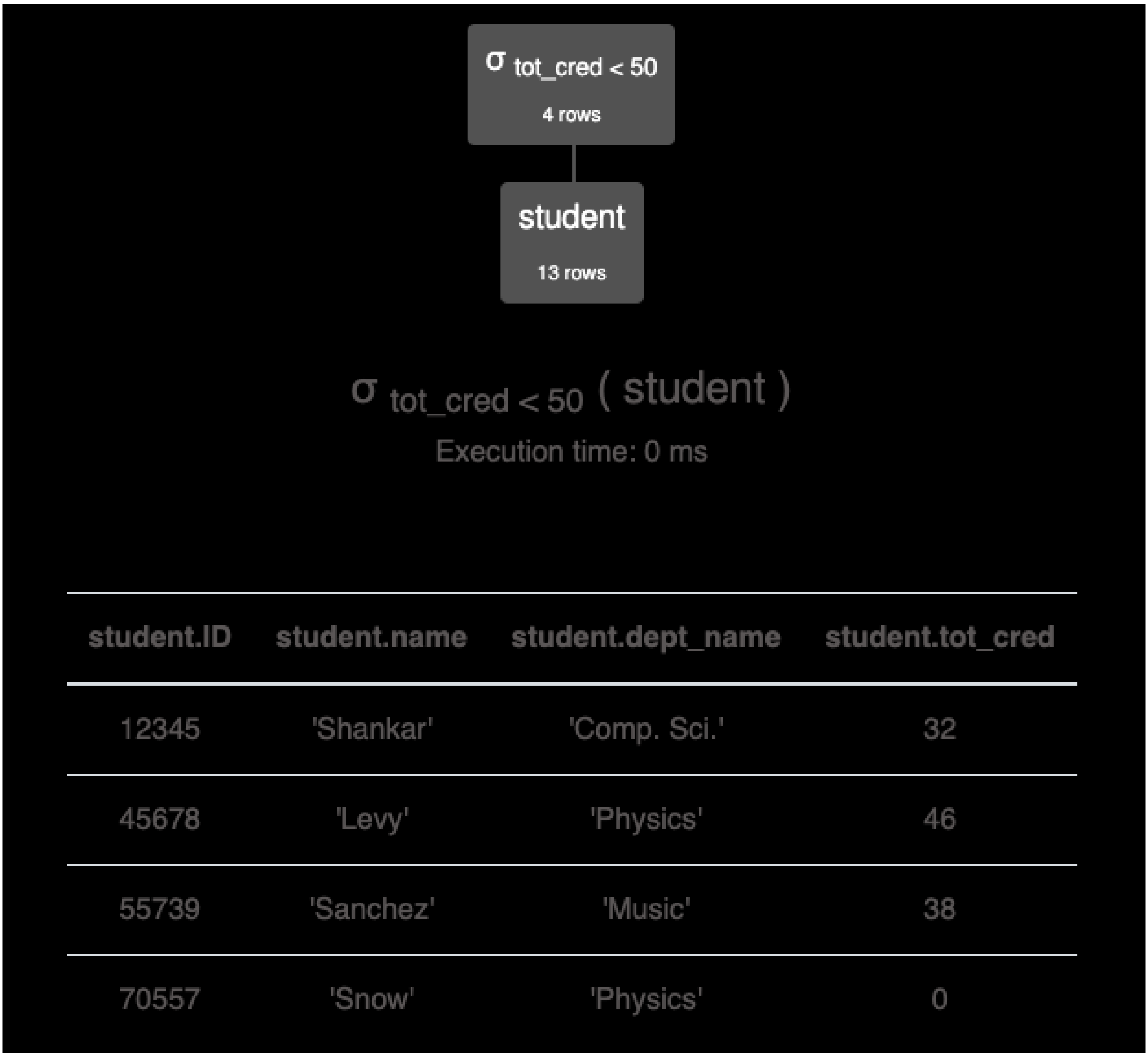
- The following is an example of how to show your relational algebra answers.
- You will use the following model for showing your answers.

Example

- "Write a query that returns all student with less than 50 tot_credits."

Answer:

- Algebra
    - σ tot_cred<50 (student)



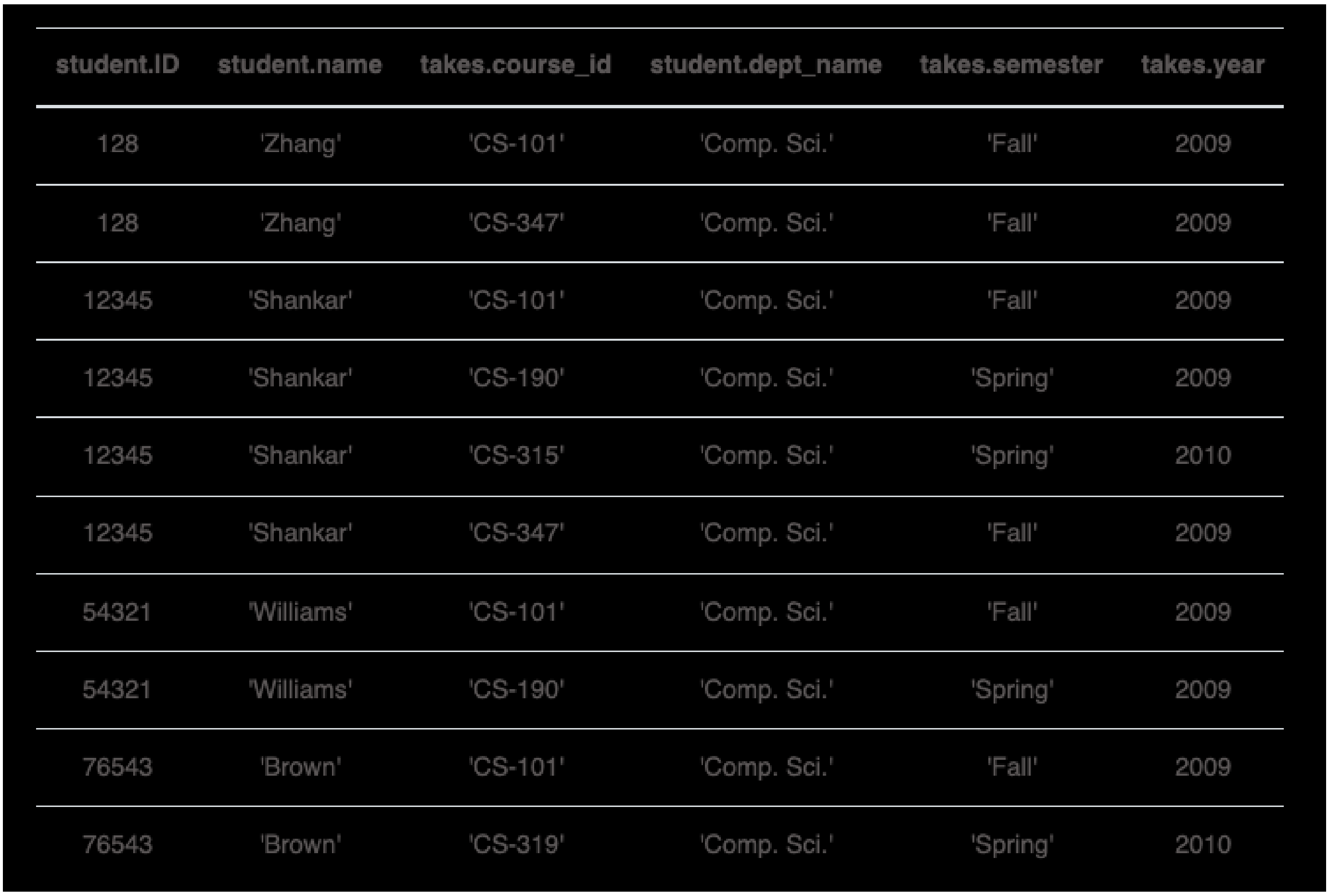
σ tot_cred < 50 (student)

Execution time: 0 ms

student.ID	student.name	student.dept_name	student.tot_cred
12345	'Shankar'	'Comp. Sci.'	32
45678	'Levy'	'Physics'	46
55739	'Sanchez'	'Music'	38
70557	'Snow'	'Physics'	0

Computer Science Students and Courses

- Write a query that produces the following result.



student.ID	student.name	takes.course_id	student.dept_name	takes.semester	takes.year
128	'Zhang'	'CS-101'	'Comp. Sci.'	'Fall'	2009
128	'Zhang'	'CS-347'	'Comp. Sci.'	'Fall'	2009
12345	'Shankar'	'CS-101'	'Comp. Sci.'	'Fall'	2009
12345	'Shankar'	'CS-190'	'Comp. Sci.'	'Spring'	2009
12345	'Shankar'	'CS-315'	'Comp. Sci.'	'Spring'	2010
12345	'Shankar'	'CS-347'	'Comp. Sci.'	'Fall'	2009
54321	'Williams'	'CS-101'	'Comp. Sci.'	'Fall'	2009
54321	'Williams'	'CS-190'	'Comp. Sci.'	'Spring'	2009
76543	'Brown'	'CS-101'	'Comp. Sci.'	'Fall'	2009
76543	'Brown'	'CS-319'	'Comp. Sci.'	'Spring'	2010

Answer:
π student.ID, name, course_id, dept_name, semester, year σ dept_name = 'Comp. Sci.' ∧ (year = 2009 ∨ year = 2010) (student ⋈ takes)

SQL

- Write the equivalent SQL statement and execute it below.

In [3]:

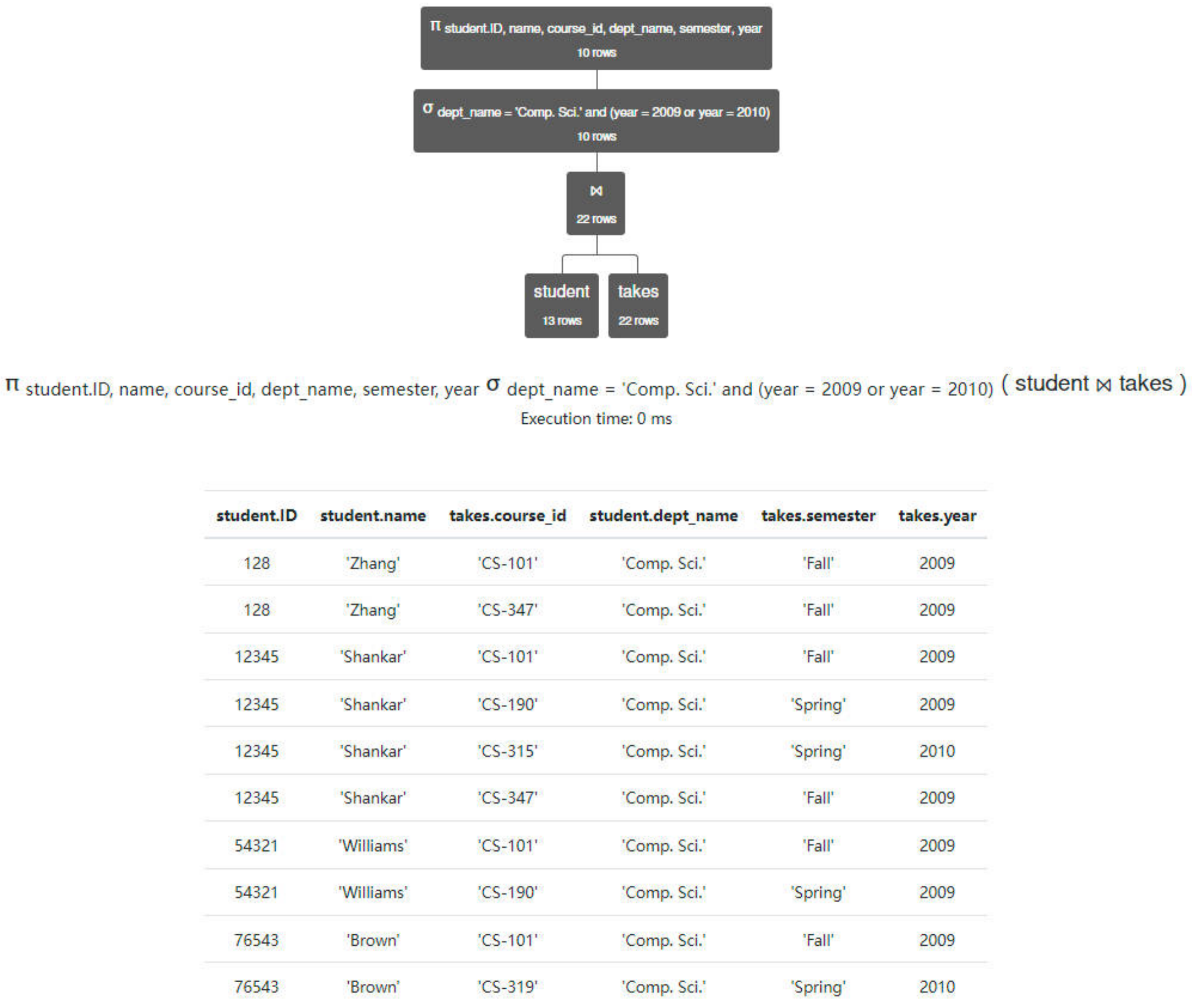
```
%sql use mysql;
%sql select student.ID, name, course_id, dept_name, semester, year \
from student natural join takes where dept_name = 'Comp. Sci.' \
and (year = 2009 or year = 2010)
```

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

Out[3]:

ID	name	course_id	dept_name	semester	year

The result from dbis-uibk.github.io:


π student.ID, name, course_id, dept_name, semester, year
10 rows
σ dept_name = 'Comp. Sci.' and (year = 2009 or year = 2010)
10 rows
⋈
22 rows
student
13 rows
takes
22 rows

π student.ID, name, course_id, dept_name, semester, year σ dept_name = 'Comp. Sci.' and (year = 2009 or year = 2010) (student ⋈ takes)

Execution time: 0 ms

student.ID	student.name	takes.course_id	student.dept_name	takes.semester	takes.year
128	'Zhang'	'CS-101'	'Comp. Sci.'	'Fall'	2009
128	'Zhang'	'CS-347'	'Comp. Sci.'	'Fall'	2009
12345	'Shankar'	'CS-101'	'Comp. Sci.'	'Fall'	2009
12345	'Shankar'	'CS-190'	'Comp. Sci.'	'Spring'	2009
12345	'Shankar'	'CS-315'	'Comp. Sci.'	'Spring'	2010
12345	'Shankar'	'CS-347'	'Comp. Sci.'	'Fall'	2009
54321	'Williams'	'CS-101'	'Comp. Sci.'	'Fall'	2009
54321	'Williams'	'CS-190'	'Comp. Sci.'	'Spring'	2009
76543	'Brown'	'CS-101'	'Comp. Sci.'	'Fall'	2009
76543	'Brown'	'CS-319'	'Comp. Sci.'	'Spring'	2010

In [ ]: