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 [4]:
```
# Print your name, uni, and track below

name = "William Das"
uni = "whd2108"
track = "Programming"

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

```
William Das
whd2108
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: Put your answer in the markdown cell.

- A data defintiion language (DDL) specifies information about data stored in a database, particularly about its relations, such as the schema for each relation, value types of attributes, Integrity constraints, which is used to create and modify tables in SQL.
- A data manipulation language (DML) then uses the database schema from the DDL to manage data stored in database, such as inserting and altering the actual data in a database.

- An example of an SQL DDL statement would be to create a table with schema:

```
create table patient ( ID char(5),
                       first_name varchar(20),
                       last_name varchar(20),
                       diagnosis varchar(20))
```

- An example of an SQL DML statement would be to insert a value into this table:
  `insert into patient values ('10450', 'John', 'Doe', 'alzheimer')`

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: Put your answer in the markdown cell.

- Faster access to data: DBMS allows faster access of data and queries to a database, as opposed to a large spreadsheet with many users editing its contents, which often suffers in performance.
- Increased security and privacy: Spreadsheet programs have poor security compared to a DBMS—-a DBMS has better control of user access and data security in a database.
- Efficient sharing and updating of data: For sharing data, especially with many users editing a dataset, a DBMS can effectively control user access, and any changes or updates in a database can be reflected in tables, as opposed to having to update multiple spreadsheets, ensuring that data is updated more efficiently.

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: Put your answer in the markdown cell.

- Structured data consists of data organized into a clear and defined data model that can be stored in a relational database--data can be mapped to predesigned fields and easily extracted using SQL (structured query language). Semi-structured data consists of data that has partial structure and definition, but cannot be stored in a relational database. Unstructured data consists of data that does not have a predefined structure or underlying data model, and is often data in its absolute raw form.
- Structured data can be presented in columns and rows in a database, and can thus be stored as relations in a database—-as such examples of structured data can be as simple as storing attributes like names or birth dates (any type of relational data). An example of semi-structured data is HTML code, which is structured in the format of a webpage, but cannot be stored by itself in a relational database. Examples of unstructured data include images, videos, or emails, which don't necessarily each follow a specific, predefined method of organization, and cannot be stored in a relational database.

Questions 4: Data Abstraction Levels

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

Answer: Put your answer in the markdown cell.

- The lowest level, the physical level, in a database deals with how data is stored in the database, and baseline rules and structure, which includes defining entity names and relationships. The logical level deals with the schema and representation of relational data stored in the database, defining relationships between tables and schemas, which includes entity names and relationships, as well as primary and foreign keys. The view level is the topmost layer, closest to the user, which defines how a database is implemented and displayed, which includes primary and foreign keys, as well as table names, column names, and column data types.

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: Put your answer in the markdown cell.

- A database schema specifies how data is organized, and structured, in a relational database; it defines the logical structure of the database, which includes its different tables and keys. A database instance is a snapshot of a database at a given point in time, which takes into account all the data and its relations within the database.
- You would use DDL statements to create and update a schema, since DDL is used to define and update database schemas, as opposed to updating records in the database.

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: Put your answer in the markdown cell.

- A procedural data manipulation language specifies the necessary steps, step-by-step, to achieve an end result, while a declarative data manipulation language describes an end result or goal.
- Relational algebra is procedural, as it uses relational input to produce a new relation as an end result, where we specify the necessary relations to achieve the output relation. Python is declarative as it allows you to specify what you want calculated, or functions you wish to apply to a set of data, without having to explicitly define how these computations are made.

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: Put your answer in the markdown cell.

- Two-tier application architectures refers to direct communication between a client and database server, where all application logic for processing data from the database exists within the client interface or database, without an intermediate, middle-tier, channel. Three-tier application architectures include an additional middle layer (client, business, and database layers), where logic for processing data from the server and client resides in this intermediate channel.
- Jupyter notebooks are three-tier applications since it acts as a web-based platform for interfacing with servers; the web-based app acts as a client layer for displaying the user interface, the middle-tier layer executes the python code, which then interacts with the database layer to retrive and manipulate data from a database.

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: Put your answer in the markdown cell.

- A database administrator would use DataGrip, which allows a user to manage databases and execute SQL queries.
- A sophiticated database user would use Jupyter Notebooks, which allow users to manipulate data in Python, along with executing SQL queries, creating a more dynamic development environment and platform.

Questions 9: Relational Algebra

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

Answer: Put your answer in the markdown cell.

- Closure in relational algebra refers to how the outputs from one operation can acts as input into another operation, which allows the creation of nested expressions.
- For example, the query for extracting the rows asked for in the university database illustrates closure:
  ```
  π student.ID, name, course_id, dept_name, semester, year (σ
  student.ID = takes.ID and student.dept_name = 'Comp. Sci.'
  (student x takes))
  ```
  Here, we use nested operations for selecting fields and filtering out instances from the student and takes tables; we also employ cartesian product using corresponding IDs and department name of the student, which is used as input into the other queries.

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.

- The Red Sox lost to the Yankees on both days, unfortunately.

Data Modeling

ER Diagrams

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

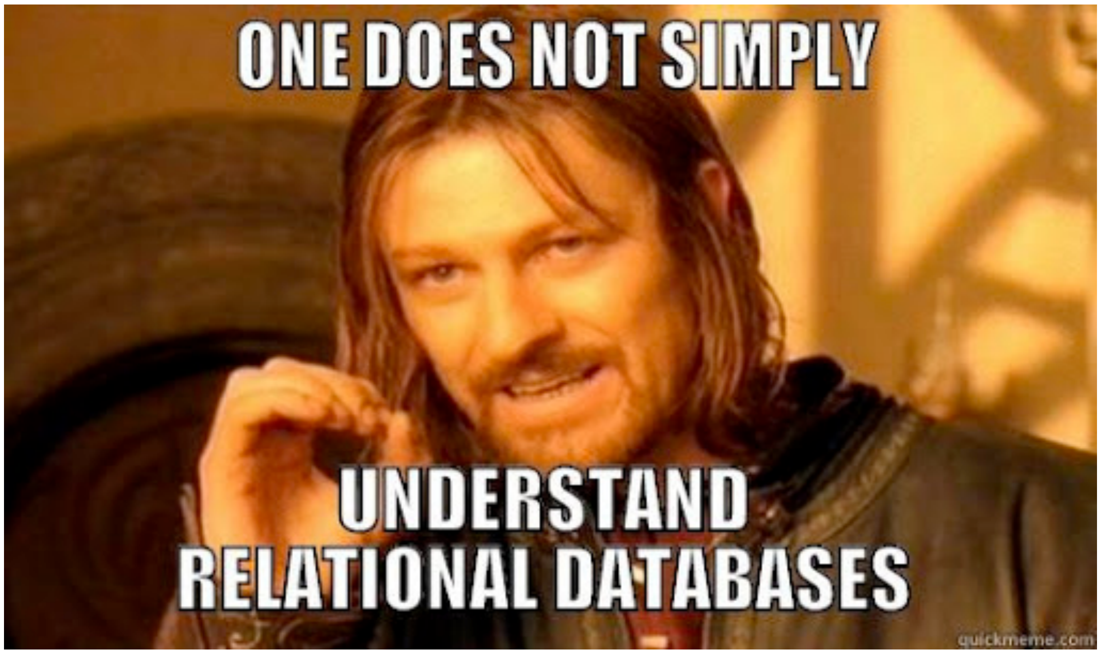
- Entity Types:
  - `automobile` (primary key is VIN)
  - `driver` (primary key is SSN)
  - `insurance_policy` (The primary key is a two letter US state code is a number. This information is just hints to help you with your logical modeling definitions).

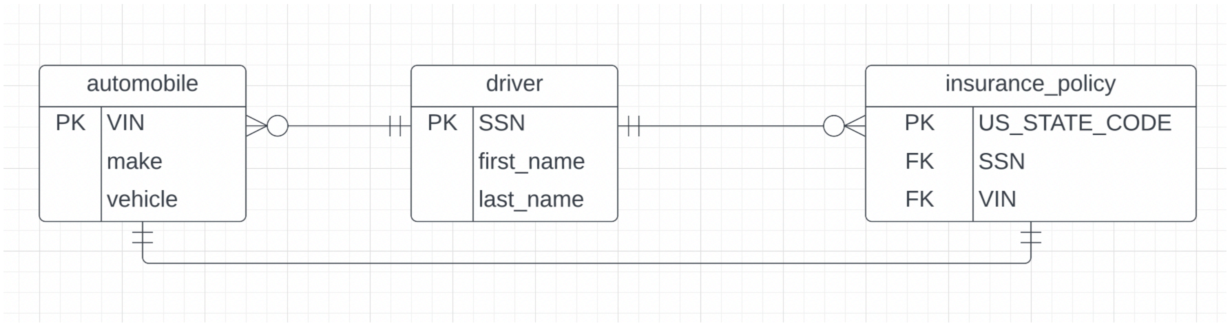
- The relationships are:
  - A `driver` is the driver for 0, 1 or many `automobiles`.
  - The relationship between `insurance_policy` and `automobile` is one-to-one. An `automobile` as exactly one `insurance_policy` and vice versa.
  - An `insurance_policy` has exactly one `driver` that is the primary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

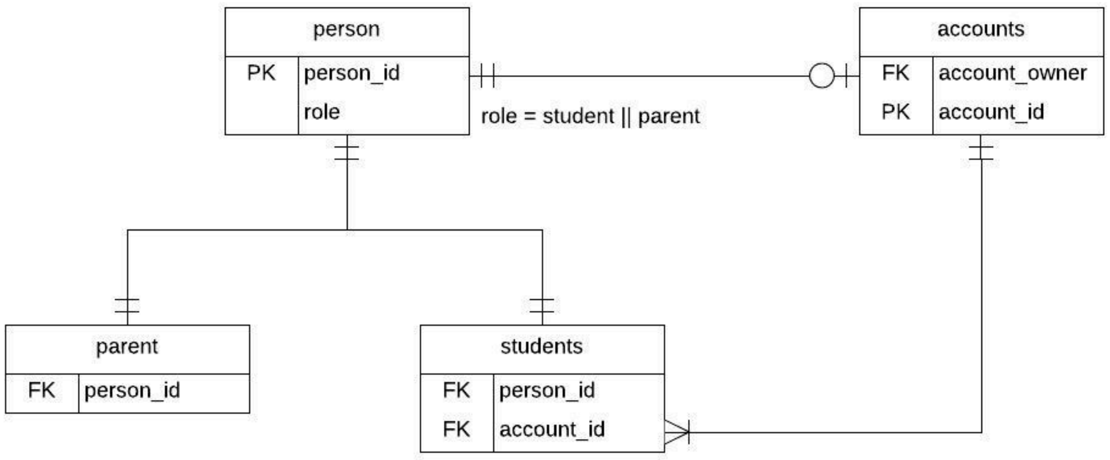
Use the following line to upload a photo of your Lucid Chart





In the diagram above, one driver can have many automobiles, one automobile can have one insurnace policy (one-to-one), and one driver can have many insurance policies.

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 [2]:
```
%load_ext sql
```

In [3]:
```
# Modify the statement below with your MySQL ID and password.
#
%sql mysql+pymysql://root:dbuserbdbuser@localhost:3306
```

In [6]:
```
%%sql
create database f22_hw2;
```

```
 * mysql+pymysql://root:***@localhost:3306
1 rows affected.
```

Out[6]:

```
[]
```

In [7]:

```
%%sql
use f22_hw2;

create table person (
    person_id varchar(5),
    role varchar(10),
    primary key (person_id)
);
```

```
 * mysql+pymysql://root:***@localhost:3306
0 rows affected.
0 rows affected.
```

Out[7]:

```
[]
```

In [8]:

```
%%sql
create table accounts (
    account_id varchar(5),
    account_owner varchar(5),
    primary key (account_id),
    foreign key (account_owner) references person (person_id)
);
```

```
 * mysql+pymysql://root:***@localhost:3306
0 rows affected.
```

Out[8]:

```
[]
```

In [9]:

```
%%sql
create table students (
    person_id varchar(5),
    account_id varchar(5),
    foreign key (person_id) references person (person_id),
    foreign key (account_id) references accounts (account_id)
);
```

```
 * mysql+pymysql://root:***@localhost:3306
0 rows affected.
```

Out[9]:

```
[]
```

In [10]:

```
%%sql
create table parent (
    person_id varchar(5),
    foreign key (person_id) references person (person_id)
);
```

```
 * mysql+pymysql://root:***@localhost:3306
0 rows affected.
```

```
Out[10]: []
```

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.*

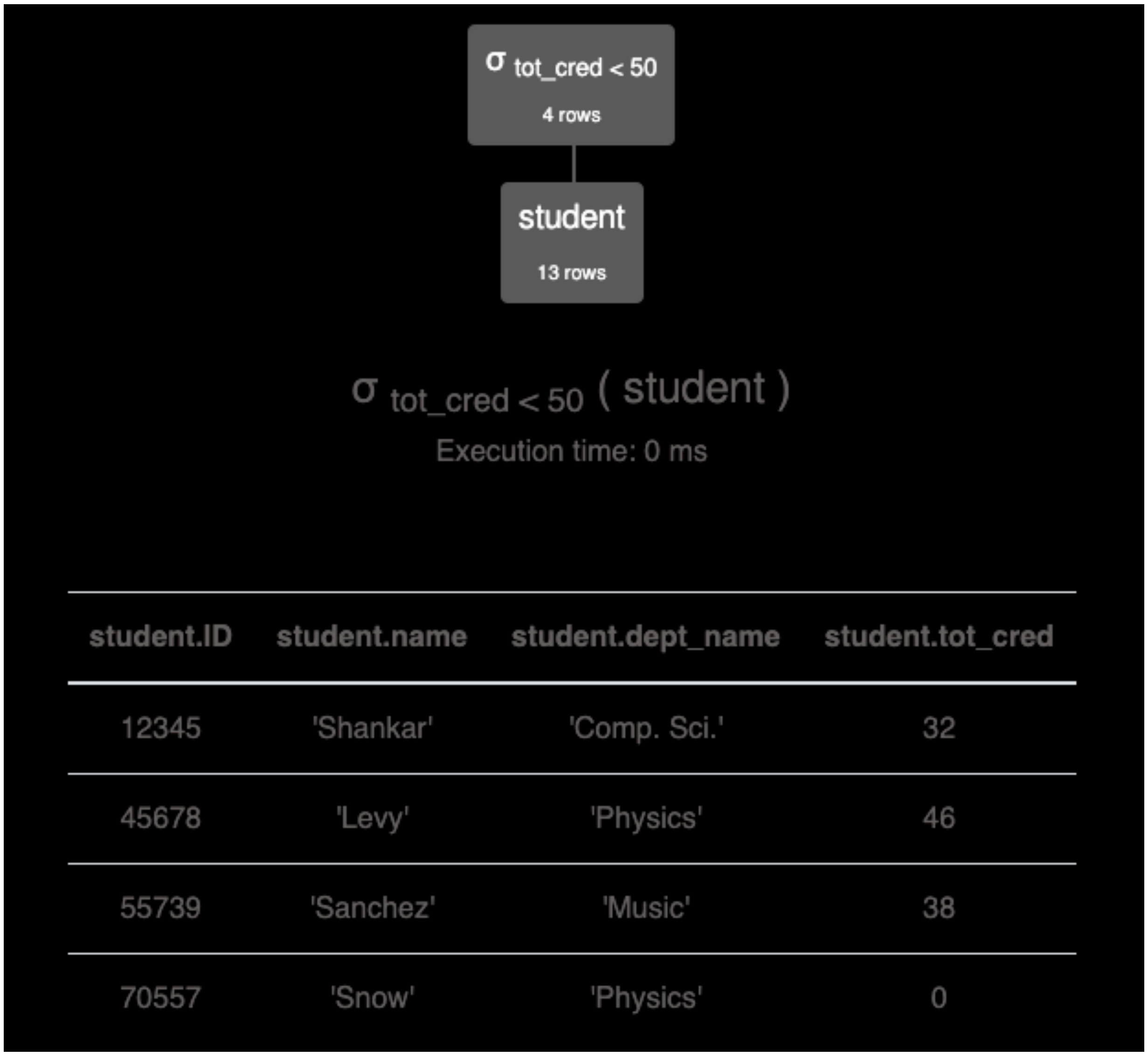
Example

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

Answer:

- *Algebra*

  $\sigma_{\ tot\_cred<50}\ (student)$



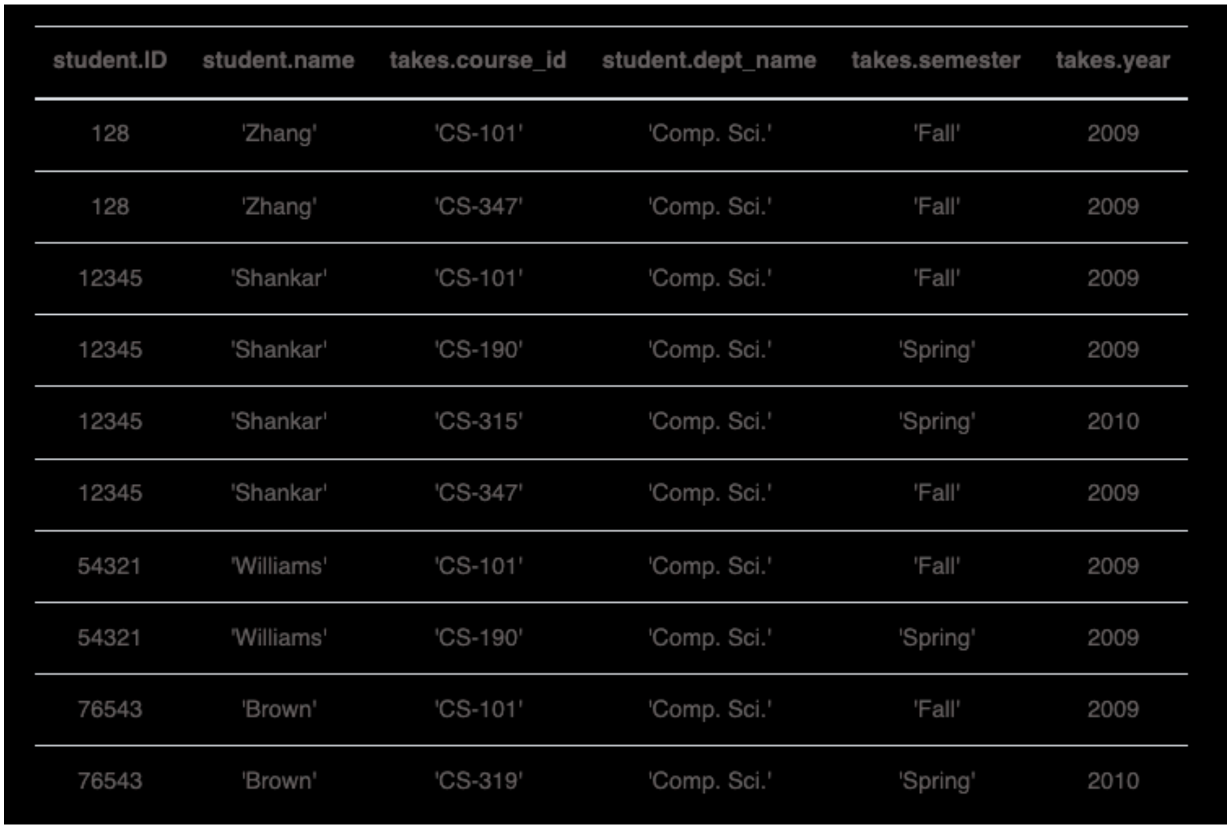
$\sigma_{\text{tot\_cred} < 50}$ (student)

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

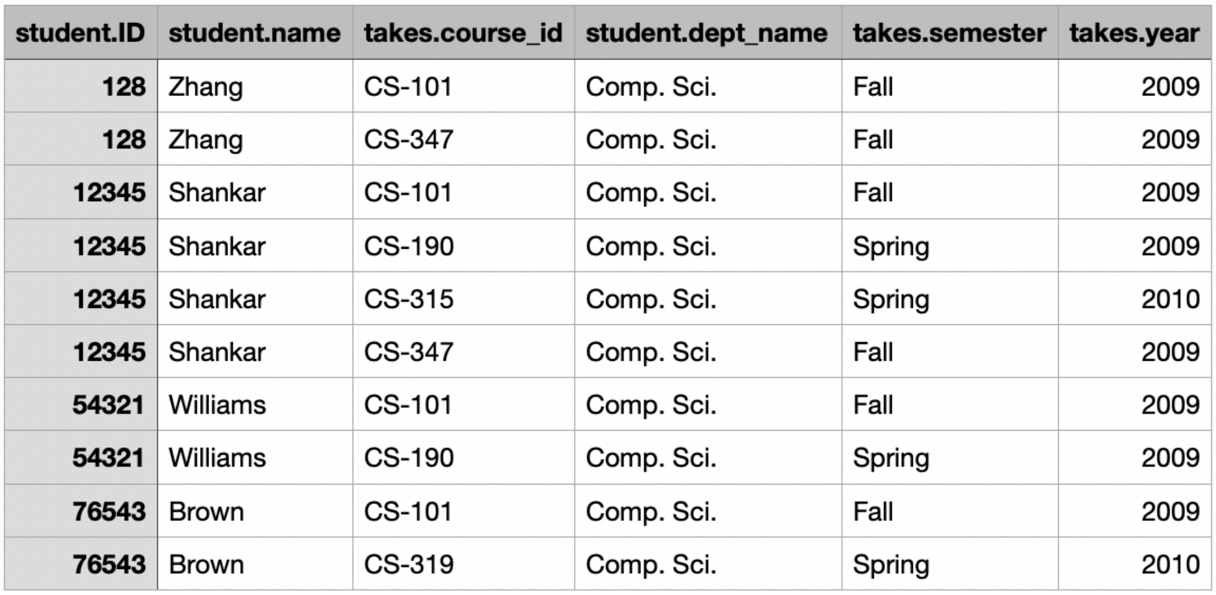
π student.ID, name, course_id, dept_name, semester, year (σ student.ID = takes.ID and student.dept_name = 'Comp. Sci.' (student x takes))

This relational algebra query outputs the table above.

SQL

- *Write the equivalent SQL statement and execute it below.*

```
In [ ]: %sql select student.ID, name, course_id, dept_name, semester, year from stuc
```



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

The result of the sql query is illustrated in the screenshot above.

In [ ]: