**CS 122A: Introduction to Data Management – Fall 2020**

# Homework 1: E/R Modeling (100 points)

***Due Date:*** **Wednesday, Oct 14 (04:00 PM PDT)**

## Partner Policy

The partner policy for this HW (and subsequent) HW assignments will be that your CS122A partner is your "brainstorming buddy". You can discuss the HW with them freely w/o violating any rules. But - in the end, t*his is an individual assignment, and each of you should turn in your own work.* For this particular assignment, you should attempt the assignment **alone** first, and then talk to each other, and then write down some notes about the key differences between yours and your partner's ER diagram. If you do not have a partner, you will need to find someone to talk to after completing ER design on your own - i.e., students who opt to work solo will need to find some other student/team to compare diagrams with in order to get credit for this last part of the assignment (see below).

## HW Submissions

This HW assignment should be turned in with a filename that contains your student ID and name (e.g., 12345678_John_Doe.pdf) and it must be submitted online, as a **PDF** file, through the associated assignment in **Gradescope** (HW1 in this case). See the table below for HW 1 submission opportunities. Note that after the second deadline, Thursday October 15th, no further HW 1 submissions will be accepted at all. We will not accept **any** assignments after that time since we will publish a correct solution at that time. *Plan to turn your work in on time!* If possible, save your one dropped assignment for the end of the term when you are most likely to want/need it, or for a time when something unanticipated crops up.

| **Date / Time** | **Grade Implications** |
|---|---|
| Wednesday, Oct 14 (04:00 PM) | Full credit will be available |
| Thursday, Oct 15 (04:00 PM) | 10 points will be deducted |

## E-R Schema Design [90 pts]

In this age of information, *mis*information is all too common. Fed up with countless baseless arguments on Twitter, you and a group of your fellow CS122a students have decided to start a service through which users can subscribe to tweets they can trust. This service, dubbed ***CheckedTweets.org***, allows users to attach evidence to some tweet (be it a scholarly article, an interview, etc..). Special users dubbed as "Checkers" will then determine whether or not the aforementioned evidence - when there is enough of it, that is - is sufficient to verify the tweet. If so, the tweet is then marked as "verified" (not to be confused with Twitter's verified user concept) by that Checker. What's great about CheckedTweets is

that the Checkers will have been carefully vetted by your company beforehand - they are among the experts in their fields. Checkers can come from a variety of backgrounds: political scientists, epidemiologists, climate scientists, etc... Once these tweets are verified by a Checker, they will then be visible to (and presumably pushed to) users who only want to read "checked tweets."

In an interview with the CheckedTweets.org founders, they shared the following thoughts on the data model needed for their service:

1. Each user will be assigned a unique user id by CheckedTweets when they first join the service. Users will have a name consisting of a first name and a last name, an email address, a password (definitely hashed, don't worry :-)), a registration date, and an address consisting of a country, state, and city. Users may optionally provide a URL to a profile picture to be displayed on the front end.
2. There will be a special subset of users known as "Checkers" who are hand-picked experts in a certain field. A Checker has one or more fields-of-expertise associated with him/her, as well as a date corresponding to when they were appointed as a checker. These Checkers will also have one or more phone numbers associated with them, with each phone number consisting of a type (i.e., either a home, mobile, or work phone) and a number.
3. As the organization name "CheckedTweets" suggests, we are going to start by storing tweets obtained from the Twitter API. The tweets that come from the Twitter API are in a JSON format (you can look at an example in the Twitter API's documentation). For starters we are going to store the raw tweets (i.e., the JSON document of tweets) along with their provided unique tweet ids.
4. In addition to the raw tweets, we want to store processed tweets (which we will simply call tweets) that contain a selected subset of the information in the raw JSON documents. The desired information coming from each raw Tweet is its unique tweet id, the actual text of the tweet, and zero or more hashtags. (The plan is for some automated software to convert raw tweets into processed tweets on a periodic basis.)
5. Each tweet must be associated with the raw tweet that it was derived from.
6. Next we have information about the Tweeters (i.e., the Twitter users who are generating the tweets using their Twitter applications). Note that Tweeters are **NOT** users of CheckedTweets - they are Twitter's users. Each Tweeter has a unique id, a followers count, a twitter handle (e.g., @joe_doe), a display name (e.g., Joe Doe), and finally whether the Tweeter is a verified one or not (in Twitter's sense of the word verified).
7. Each tweet must be posted by exactly one Tweeter and each Tweeter must have at least one tweet. Each post of a tweet by a Tweeter has associated with it the posting datetime and optionally the posting location (which consists of longitude and latitude).
8. A tweet can be replied to by any number of other tweets (i.e., zero or more).
9. Also, a tweet could be quoted by any number of other tweets (again, zero or more).
10. Given a single tweet and its corresponding relationships, we are also able to compute a popularity index and a quality index. These are given by the equations:

$$Popularity = 0.4\ (Number\ of\ quotes) \times 0.6\ (Number\ of\ replies)$$

*Quality = Amount of associated evidence used for verification*

11. Users (whether a "Checker" or not) can submit any number of pieces of evidence. Each piece of evidence must be about one or more tweets. A piece of evidence contains a unique id and a URL link that refers to the supposedly trusted source of information (e.g., a medical journal article).
12. A Checker can fact-check (i.e., verify) a tweet by submitting the details of their verification. Each verification that is submitted will have a verification id, an optional comment about the verification process, and the date on which the verification is performed. (Each verification will be the result of the inspection and "use" of some or all of the evidence about the tweet being verified; more on this next.) A Checker can verify zero or more Tweets over the course of time.
13. Each verification that a checker submits must be about a posted tweet. Multiple checkers may (but not necessarily) verify the same tweets, in fact.
14. A checker's verification of a tweet must use **two or more** of the pieces of evidence that have been provided by users about a tweet. The idea here is that something will not be considered to be "true" unless there are at least two independent sources of information that can be used to attest to its truth. *(Hint: An ER diagram cannot capture the constraint of "using at least two pieces of evidence". However, your diagram should capture the notion that each verification must use evidence and that multiple pieces of evidence may be provided for a given tweet).*

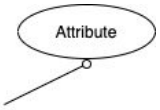
Develop an E-R schema to represent the required information and express its design in the form of an E-R diagram. Please use https://app.diagrams.net to draw your entities and relationships. (Do NOT invent your own notation or use UML! The site diagrams.net includes a variety of E-R modeling notation. *Please only use the E-R notation from the lectures or the textbook.* You can find a decent E-R model tutorial on the web here, but beware of its different ISA notation, which you should not use here.) Be sure that your E-R design captures all of the implications of the CheckedTweets user model and business model, including all aspects below (see parts (a) and (b)), and be sure to employ advanced E-R modeling features such as weak entities, ISA relationships, composite attributes, set-valued attributes, and optional attributes if/where appropriate. Please use the book's/lecture's ISA notation and be sure to think about and model the overlap and covering constraints when detailing your design. NOTE: Do NOT attempt to design your E-R schema by making just one pass over the bullets above! You will need to carefully analyze all of the requirements in order to identify the available attribute and relationship commonalities for ISA purposes; attempting a one-pass design is almost sure to lead you to miss important (inheritance-based) sharing opportunities and to lead you to a much messier-than-necessary design. Cleanliness matters!

**Notes on Diagrams.net:**

In the main interface of Diagrams.net, click **File → Open From → Device**. Then choose the provided file "**ER.drawio"**. You will see the entities drawn for you.

Under the Entity Relation category, you will find that some of the shapes are missing (e.g., ISA triangle). To get the triangle, go to **General** and pick the triangle and type the word **"ISA"** inside the triangle. Click

on the triangle again and then go to the right panel and click on the **Arrange** tab. You will find a button called **Rotate shape only by 90°**. Rotate the triangle until you get what you expect.

For an optional attribute, you will want to use a line that ends with a circle, i.e.: 


To get the empty circle at the end of a line, click on the line. From the right panel under the **Style** tab, you will find six drop-down menu options. The two on the right bottom can be used to modify the line ends to the shape you want.

Finally, once you are done with your diagram, you can export your drawing as a PNG (**File → Export As → PNG**). You can then copy and paste this into the homework template. Voila!

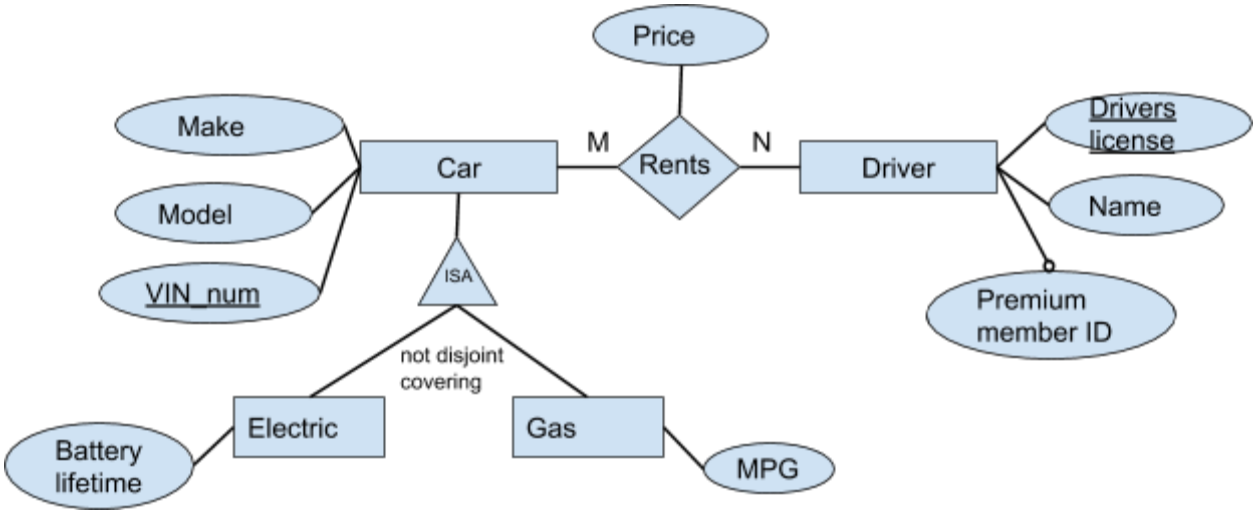
**Using the schema template that we will provide**, draw your E-R diagram, which must include:

(a) [45 pts] All of the relevant entities and their associated attributes, including their keys. (For the Entities, you may **ONLY** use "User", "Checker", "Evidence", "Verification", "Tweet", "RawTweet", and "Tweeter". Do **NOT** introduce any additional entities or use fewer entities than these!)

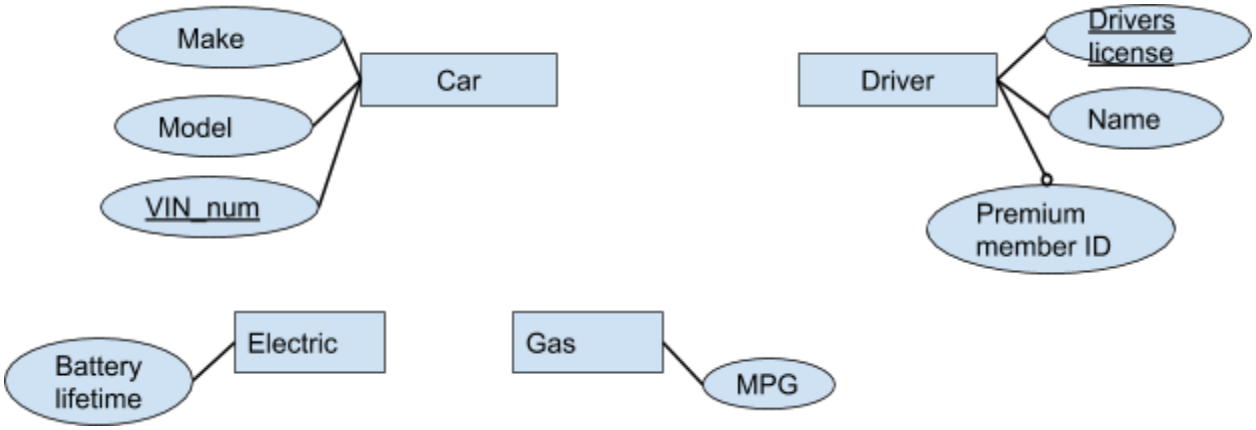
(b) [45 pts] All of the relevant relationships and their associated relationship attributes, if any. For the relationships, each one must also have appropriate cardinality and participation constraints. (***Note***: ISA "relationships" are special but are to be included in this portion of your design too; don't forget to include their overlap and covering constraints.)

Be sure to use both the HW #1 Diagrams.net schema template and the HW #1 submission template from the CS122a web page. The Diagrams.net template should be used as the basis for drawing the E-R schema that you turn in. (Your solution will **NOT** be accepted and will thus not be graded if you do not use **our** provided layout for your diagram! Without such visual standardization, it becomes too difficult to grade everyone's answers with sufficient thoroughness.) Please use this software to draw your E-R diagram, and again please **do not** move or otherwise change the spatial placement of the entities (!).

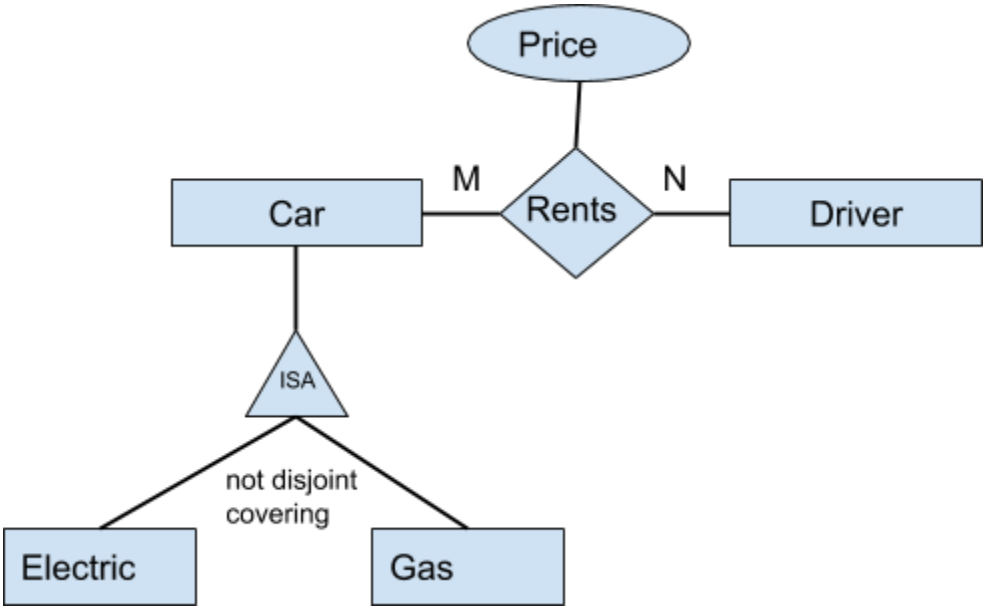
**Important:** As you will see in the templates, there are two similar pages that have all the necessary entities on them. **On the first page, just indicate the attributes for each entity without introducing any relationships. On the second page of your overall E-R diagram, just show the relationships, their cardinalities, their attributes (i.e., the relationship attributes), and the ISA relationships and constraints**. Please do not include any of the entity attributes in the second E-R diagram. As an example, suppose that your final design for CheckedTweets.org's conceptual schema is the following E-R diagram:




In your first E-R diagram you should show all of the entities with their attributes but no relationships and no IsA's. This diagram will be used to check that you have correctly identified all of the entities and all of the attributes needed for CheckedTweets.org. (Notice how the inherited attributes are handled.)




Then, in your second E-R diagram, you should show the ER graph of all entities (without attributes) and their regular relationships (with attributes, if any) and ISA relationships.




This diagram will be used to check your overall design w.r.t. its relationships -- and it will give you a nice overview to refer back to later on (again and again) in the project. You will eventually probably want to print out the CheckedTweets E-R diagram and keep it sitting next to you whenever you are working on any of the subsequent HW assignments!

Note that if the pictures shown above are indeed the same as your final design for CheckedTweets.org, you should probably read the founders' information again more carefully. (:-))

**E-R Design Comparison w/ Partner [10 pts]**

Given the description and the requirements above, there are still multiple different ways that one may interpret the requirements! After completing the first design pass for this assignment **on your own**, talk to your partner and compare your E-R diagrams. Take notes about the differences you find with your diagram and your partner's: look carefully at your respective relationships, their cardinality and/or participation constraints, the attributes, etc., and talk to each other about why your model looks the way it does. After this meeting, go ahead and update your model based on anything you may now want to model differently. To get full credit for this part of the assignment, the PDF that you turn in should include:

- Your ***final*** E-R diagram, laid out using the provided template(s).
- A list of the main ***differences*** that you identified in your partner meeting and a few sentences summarizing what you learned from doing the comparison exercise.
- Your ***initial*** E-R diagram, just for the record, also laid out using the template(s), so we can see what you started with versus what you ended up with. (*Note:* The details of this version will not be graded for correctness.)