MSiA-413 Introduction to Databases and Information Retrieval

Homework 2: Data modeling: Data Sets, Normalization, and ER Diagrams

Name 1: Laurie Merrell_____

NetID 1: lmr8733_____

Name 2: Kristiyan Dimitrov_____

NetID 2: ktd5131_____

Instructions

You should submit this homework assignment via Canvas. Acceptable formats are word files, text files, and pdf files. Paper submissions are not allowed and they will receive an automatic zero.

As explained during lecture and in the syllabus, assignments are done in groups. The groups have been created and assigned. Each group needs to submit only one assignment (i.e., there is no need for both partners to submit individually the same homework assignment).

Each group can submit solutions multiple times (for example, you may discover an error in your earlier submission and choose to submit a new solution set). We will grade only the last submission and ignore earlier ones.

Make sure you submit your solutions before the deadline. The policies governing academic integrity, tardiness and penalties are detailed in the syllabus.

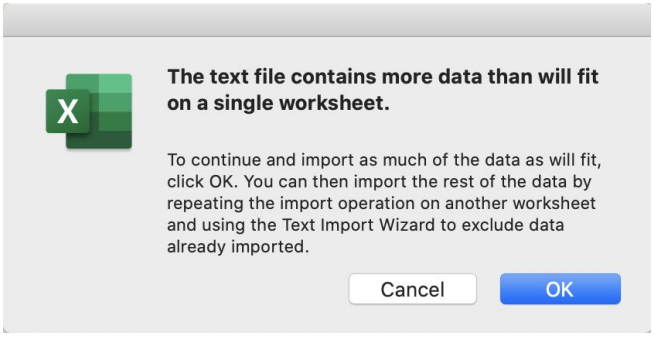
Due Date: Thursday October 17, 11:59 pm

Question 1. Dataset Exploration (6 points)

Download the data sets using the links below and import them to either Excel, Numbers, or another spreadsheet processing program of your choice. Below you can find a tutorial on how to download a data set and import it to Excel or Numbers. Then, proceed to answer the following questions:

a. **(3 points)** Did you encounter any problems in importing any of these datasets into a spreadsheet? If yes, describe which dataset(s) you encountered the problem with, and explain the reasons you believe it failed to be imported.
   - **For unique names, we are using the username+data set name+filename, since it seems like that's the only combination that ensures uniqueness.**
   - **We tried loading data in Excel, Google Sheets, and Numbers**

Data set name: benhamner/sf-bay-area-bike-share#status.csv.
For the San Francisco bike data, the "status.csv" file was not imported successfully on the first attempt. It gave this error in Excel:



Based on the error text, we believe the issue is there is too much data in the csv file for Excel to handle.

We also tried uploading to Google Drive and opening in Google Sheets. After waiting for the file to open for 15 minutes, we concluded it will not load within a reasonable amount of time. We suspect the reason for this is Google Sheets needs to expend a lot of resources to visualize the data in a GUI, on top of storing the data in memory. Similar behavior was exhibited in MacOS Numbers. The file did load, but was very slow and practically unusable. At their core, both issues stem from the size of the data. (~1.4 million rows)

For the Pokemon.csv data: Here the data loaded fairly easily into all three spreadsheet tools.

b. For the dataset(s) that were successfully imported, please answer the following questions:
   i. **(1 point)** What is the data set's name? You have to be really precise with the name; after all, there may be multiple datasets at Kaggle.com with similar names. Find the name that uniquely identifies it.
      Data set name: abcsds/pokemon#Pokemon.csv
   ii. **(1 points)** How many rows does it have? **801**
   iii. **(1 points)** How many columns does it have? **13**

Link to Project in Kaggle	CSV file to download
https://www.kaggle.com/benhamner/sf-bay-area-bike-share	status.csv
https://www.kaggle.com/abcsds/pokemon	pokemon.csv

How to download a data set from Kaggle

The links above point to two data predictive modelling and analytics projects on Kaggle. You can read the project and data description on the corresponding project overview page. If you click the project data page, you will see a list of comma-separated values (CSV) files and other file types on the left hand side. When you click on any file, you can preview the first 100 columns, and read the column metadata or column metrics. There is a download button to download that CSV file to your computer. Once downloading is done, you can follow the guidelines below to import the dataset into the spreadsheet program of your choice.

How to import a CSV file into Excel

Go to the "File" main menu and select the submenu "import" to import a csv file in Excel. Choose the appropriate file type to import the file (CSV), and then click the import button. Next, select which file to import, and click "Get Data". Most of the CSV files are delimited by commas to separate each column, and you can import it starting at row *n* if that file is too large. After that, there are a few options to opt for a proper column data format, and a spreadsheet which you want to put the data. It is recommended import the new data into a new sheet. When you are done with all file configurations, click "Finish". Excel will try to import the CSV file, or throw an exception if any error happens.

How to open a CSV file in Number

For Numbers, you can choose which CSV file to open, and the program will do it for you. That's it.

Question 2. Data Type Exploration (6 points)

Assume the datasets provided below. Consider the following data formats:

a. **(1 point)** 32-bit integer**: san-francisco (dataset) > 311_service_requests (table) > unique_key (column). This column contains unique values identifying each 311 service request (i.e. it is the primary key). The largest value stored in it is 10,281,200 while the largest value that can be stored by a 32-bit int is 2,147,483,647. This is the most space-efficient way to store data in this column.**

b. **(1 point)** 64-bit integer:
datasf/san-francisco#sfpd_incidents (dataset) > sfpd_incidents (table) > pdid(column)
Column description: "Unique Identifier for use in update and insert operations" – it essentially just appears to be an ID value, but it's 14 digits long.
An example of the data in the column: "16601857306244"
The reason why your chosen data type is appropriate: These are all whole numbers (no fractional parts) so we do not need floating point, and at 14 digits long it's outside the range of a 32-bit integer. We checked online and verified on Wikipedia that this is within the range of 64-bit ints – largest possible value is 9,223,372,036,854,775,807.

c. **(1 point)** fixed point (and specify the number of decimal places):
sf-bay-area-bike-share (dataset) > weather (table) > mean/min/max sea level pressure (columns). These columns represent statistics related to sea level pressure and are reported with an accuracy of 2 decimal digits and further precision would not be relevant. Therefore, a fixed-point storage would be suitable for them (2 digits for the significant + 2 for decimal part).
An example of the data in the column: "30.07"

d. **(1 point)** floating point (either single or double precision):
sf-bay-area-bike-share (dataset) > stations (table) > lat & long (columns)
An example value is -121.88389099999999.
These columns correspond to geographic latitude & longitude location of the bike sharing stations. These should be represented as double-precision float. If we were using single precision then we could get an absolute error of ~1×10^{-7} degrees error, which corresponds to ~4 meters (source). For certain calculations this accuracy may be inadequate, therefore, we would opt in for double precision.

e. **(1 point)** date and time in epoch seconds:
datasf/san-francisco#311_service_requests > the table name: 311 Service Requests > created_date
Column description: "The date and time when the service request was made" – it's the time, down to the second, that the service call was received.
An example of the data in the column: "12/20/2013 14:48:14"
The reason why your chosen data type is appropriate: The timestamp is precise to the level of seconds, but not to the microseconds level, so epoch seconds is the appropriate datetime format to capture the level of precision present in the data.

f. **(1 point)** date and time in epoch microseconds: **We could not find any examples in the data – there did not appear to be any timestamps that were more precise than the seconds level or any that would require that level of precision.**

For each one of the data formats above, please answer the following:

i. Is there a column in one of these datasets that would be **best** stored in that format? (yes/no)
ii. If yes, please provide
    1. the data set
    2. the table name
    3. the column name
    4. a one-sentence description of the column
    5. an example of the data in the column
    6. the reason why your chosen data type is appropriate

iii. If no, explain why not (1-2 sentences)

Data sets:

1. https://www.kaggle.com/benhamner/sf-bay-area-bike-share
2. https://www.kaggle.com/datasf/san-francisco

Question 3. Data Types (8 points)

You are building a database for a credit card company. You need to select the best data types for the various parts of a credit card shown above. The database software you use supports the following types:

- *32-bit signed integer*: can store all integers between -2,147,483,648 and 2,147,483,647.
- *32-bit floating point*: can store numbers in the scientific notation with about 7 decimal digits of precision and exponent between 10^{-38} and 10^{+38}. It can precisely store all integers ≤ 16,777,215.
- *Epoch seconds*: a 32-bit unsigned integer that represents data and time by the number of seconds since midnight on Jan 1, 1970 in London.
- *UTF-8*: text, of any length.

You must use one of these four data types to store each of the values below. Each value should be stored in a single data element. Which of these types is the best to store:

a. **(1 point)** The bank name?

Answer: UTF-8 is the only storage option available for text data → UTF-8

b. **(1 point)** The CID field (4-digit decimal number)?

Answer: 32-bit signed integer is the most space-efficient method and provides more than enough space for 4-digits.

c. **(1 point)** The cardholder last name?

Answer: UTF-8 is the only storage option available for text data → UTF-8

d. **(1 point)** The expiration date?

Answer: Best use Epoch Seconds. Note: We assume we're storing the date and not the display value. If we need to store the display value (i.e. in case we need to store the "/") then we would use UTF-8.

e. **(2 points)** The credit card number (16-digit decimal number)?

Answer: 100% precision is essential here. Therefore, UTF-8 is the only option that will allow us to store the card number with no doubts, since 16-digit integers are outside the range that can be represented precisely by either 32-bit integers or 32-bit floating point numbers.

Note, however, that credit card numbers issued by a specific bank typically have the same first 6 digits (source) for all cards. Therefore, if these first 6 digits can be "hard-coded" and not stored in the database, then we only need to take care of the remaining 10 digits. Furthermore, the *last* digit is called a "check digit". From this source: "In a typical sixteen digit credit card number, the first fifteen digits are determined by the issuing bank, but the last digit, called the check digit, is mathematically determined based on all the other digits." Therefore, we don't need to worry about the final digit either. This leaves us with only 9 digits that actually identify the unique credit card

issued by our specific bank. In this case, we could store the data with 100% accuracy by using a 32-bit integer where the maximum value that can be stored is 2,147,483,647 > 999,999,999.

f. **(2 points)** The balance of the credit card?

NOTE: the balance is in USD, it is guaranteed to stay between -1,000,000.00 and +1,000,000.00 (negative values indicate credit), and must be accurate down to one cent (i.e., $1/100^{th}$ of a dollar).

Answer: We should use UTF-8 for the same reasons as the credit card number - it is essential to have accuracy. 32 bit integer doesn't work due to lack of decimal part. Float doesn't give required precision for integers. Epoch Seconds are not relevant. When doing calculations on this data (e.g. person makes a deposit/withdrawal) we would need a program interpreting string as fixed point.

Question 4. Database Normalization (10 points)

You work as a data analyst at a fishing/outdoors company. To identify new talents in database design, the company hosts an annual database schema competition. The winner takes home a commemorative statue known as the *Data Bass*. You won the competition last year, so a friend asked you to review his submission. Unfortunately, your friend did not take MSiA-413 and put all his data in a single table, shown below:

Employees_Database					
Empl. ID	*Name*	*Favorite Record*	*Department*	*Dept. Manager ID*	*Fav. Record's Artist*
1	Nancy	Abbey Road	Sales	1	The Beatles
2	John	Porgy and Bess	Accounting	2	Gershwin
3	Bill	Kind of Blue	Operations	6	Miles Davis
4	Tracy	A Night At The Opera	Sales	1	Queen
5	Muji	La Revancha del Tango	Sales	1	Gotan Project
6	Ohana	Ka 'Ano'i	Operations	6	Kamakawiwo'ole
7	Jill	Porgy and Bess	Accounting	2	Gershwin
8	Gloria	La Revancha del Tango	Operations	6	Gotan Project
9	Frank	Abbey Road	Accounting	2	The Beatles

a. (6 points) Help him by normalizing the database to remove redundancy. Show the normalized database **schema**.

Solution

Employees		**Departments**		**Records**
		Department ID*(int 32)*		Record ID*(int 32)*
Empl. ID (int 32)		Department Name *(UTF-8)*		Record Name *(UTF-8)*
Name (UTF-8)		Manager ID*(int 32)*		Artist *(UTF-8)*
Department *(UTF-8)*				
Fav Record *(UTF-8)*				

b. (4 points) Show the current **instance** of the database in the normalized schema.

Solution

Employee

EMPL ID	*Employee Name*	*Department ID*	*Favorite Record ID*
1	Nancy	3	1
2	John	1	6
3	Bill	2	4
4	Tracy	3	2
5	Muji	3	5
6	Ohana	2	3
7	Jill	1	6
8	Gloria	2	5
9	Frank	1	1

Record

Record ID	*Record Name*	*Record Artist*
1	Abbey Road	The Beatles
2	A Night At The Opera	Queen
3	Ka 'Ano'i	Kamakawiwo'ole
4	Kind of Blue	Miles Davis
5	La Revancha del Tango	Gotan Project
6	Porgy and Bess	Gershwin

Department

Department ID	*Department Name*	*Department Manager ID*
1	Accounting	2
2	Operations	6
3	Sales	1

Question 5. ER Diagram (20 points)

The main entities that participate in an online bookstore enterprise are as follows:

- A book has the information about the year that it was published, its title, the (current) price and its ISBN number. **Assume that ISBN is the unique number assigned to each edition/version of the book.**
- An author has information which includes his/her name and contact-address, along with a URL.
- Each publishing house/company has a name, postal address, phone number, email and URL.
- Each customer has a customer ID, name, address, email, credit card, and phone number, and **each customer must provide only one set of information.**
- A particular "shopping session" is typically recorded as a shopping basket, which is assigned a unique basket ID and has the information about the date of the given shopping session.
- Since this is an online bookstore, there must be physical locations where (copies of) the books are stored. A given warehouse has its address, name, and phone number available.

The associations among the various entities listed above are as follows:

- Each book is written by some author(s).
- Each book is published by a particular publishing house and information is kept about the publishing date, the edition number, and the number of copies.
- Each shopping basket is associated with a particular customer.
- Each shopping basket may contain several books and even several copies of a particular book.
- Each warehouse keeps/stocks different books, and for each book it also records the number of copies that it currently has.

Please draw an ER Diagram that models the online bookstore according to the information and rules provided above. If you make any assumptions along the way, please write them down. Note: there are a number of online tools to draw ER diagrams (e.g., https://www.lucidchart.com).

Solution

Assumptions/notes:

- **General/formatting:**
  - **I am listing foreign keys in parentheses for relationship tables to indicate that they're kind of implied.**
- **Books table:**
  - **Reviewed ISBN format here: https://www.isbn-international.org/content/what-isbn - "they now always consist of 13 digits", so assume we need a 64-bit (long) integer to represent them.**
  - **Since the specifications specifically mention that the book "has a year" I have a separate "year" field in the book entity, even though the "publishes" relationship table includes the full publication date for the edition.**
  - **I am assuming that we want to link author information with book information, and that authors can write multiple books and books can have multiple authors.**
  - **Assume that there are books in the database which the site does not current stock (i.e., are not stocked by a warehouse).**
  - **Assume that not all books have an author (ex. Bible, dictionary.)**
- **Publishes table:**
  - **I am assuming that books generally have less than 2^15 editions so that editions can be represented as short unsigned integers.**
- **Address table:**
  - **For address data, I am assuming we want it broken out the way I have seen most address data broken out in databases, with street, city, state, etc. I am also assuming we**

want at least some support for international addresses so I have made the address field names a little more general in their data types to support non-US.
    - I wondered if a publisher might also have literary agents whose addresses would be used as author addresses, so I opted for a single "addresses" table which can be used for both publishers and authors.
    - Assume that there may be multiple phone numbers/email addresses associated with a given physical address, so phone number/email should be stored separately from addresses (for example, in the scenario above where an author's address is the same as a publishing company's address).
    - Assuming that unique address information for all parties is required based on the wording of the description.
- Author table:
    - Assume "contact-address" for authors includes a phone number and email address.
- Customer table:
    - Assume we're ignoring the security concerns about how to store credit card information.