Group Assignment (40%) – Data Management & Ethics

Data management for social good

Assignment case: DonorsChoose.org

Founded in 2000 by a Bronx history teacher, DonorsChoose.org is an online, non-profit organization that has raised $685 million for America's classrooms. Teachers at three-quarters of all the public schools in the U.S. have come to DonorsChoose.org to request what their students need, making DonorsChoose.org the leading digital platform for supporting public education.

To date, 3 million people and partners have funded 1.1 million DonorsChoose.org projects. But teachers still spend more than a billion dollars of their own money on classroom materials. To get students what they need to learn, the team at DonorsChoose.org needs to be able to connect donors with the projects that most inspire them.

As part of their mission, DonorsChoose.org makes anonymized data available to researchers to help generate insights and to make public education smarter. However, they are now dealing with a large amount of data that needs to be properly managed and analyze.

Acting as data management team for DonorsChoose.org, you need to provide them support and develop an efficient database to manage and analyze the data that will capture their organizational needs. You have to think what kind of questions DonorsChoose.org might want to answer and create a database using the provided data that will allow your team to answer those questions by interrogating the data.

This assignment consists of a series of tasks (see all details below) and write a final report including the outcomes of all the tasks.

Please, carefully read the whole tasks list before starting with task 1.

Overview of provided data

A teacher from a school can initiate a funding project by posting a proposal on DonorsChoose.org and reaching out to potential donors via the platform. Donorschoose.org has a large quantity of data on projects, teachers, schools and donors.

For this assignment, you will work on public data provided by DonorChoose.org to Kaggle.com in occasion of the 'second Kaggle Data Science for Good challenge'. Tables have been prepared and adjusted accordingly in line with assignment objectives. You can find two data tables on Canvas: 'table1_donations" and 'table2_projects'.

table1_donations.csv

Donation ID	Unique ID of a donation
Donation Amount	Total amount donated for a project
Donation Date	Date and time on which the donation was received
Donation Included Optional Donation	Yes/No to give 15% of donation amount to Donoschoose
Project ID	Unique ID of the project to which the donation was given
Donor ID	Unique identifier of the donor making the donation
Donor City	The donor's city
Donor State	The donor's state
Donor Zip	The donor's zip code

table2_projects.csv

Project ID	Unique ID of a project
Project title	title of the project
Project_ description	Project description
Project_ category	Project subject category
grade_level	school grade project
cost_project	Project overall costs
posted_date	Project Posted Date
expiration_date	Project Expiration Date
status	Project Current Status (at the data collection moment): fully funded or expired
fully_funded_date	date when the project was fully funded
School ID	Unique identifier of a school
School Name	Name of the school
School Metro Type	One of four categories describing metro type, or urbanicity, of school area
School State	The state of the school
School Zip	The zip code of the school
School City	The city of the school
School County	The county of the school

Projects (N = 133,098) have a targeted costs amount and they can be either fully funded or they expire after 4 months they have been posted in the platform, if they don't meet their funding goal. Over 70% of the projects usually reach their funding goal. For each project, there is plenty of data about title, subjects, date of launch and expiration date, and project description. Be aware that some projects might have the same title, but they are actually different because they either belong to different school grades (e.g., preK2 or 3-12) or relate to different subjects.

One project is linked to one and only one teacher. Here, you will work directly at the school level, so the school where the teacher works, is the entity launching a project (or more projects, because a school has several teachers). Schools (N= 37,848) are public or public charter schools located in the US. For each school, there is quite detailed location data (please, familiarize with the US system of state>county>city>zipcode). This is helpful data because several schools have the same name, but they are located in completely different areas and states.

Donors (N = 341,232) can donate to one or more projects via Donorschoose.org. Donors are in the database because they made at least one donation (otherwise they won't be called donors…). A donation (N = 557,427) is a unique transaction made by one and only donor to one and only project. Donors might donate to the same project more than once, thus having multiple donations/transaction. Donors data is anonymized to protect their privacy, however location data for each donor is provided (yet, some donors might still want to not share this piece of information).

Assignment tasks

Table 1 shows the list of the tasks you have to perform in the assignment and related points. Full details of each task are provided separately below.

Table 1. Group assignment tasks.

Task	Description	Points
1	Given the above scenario, what could be interesting questions that the DonorChoose.org might have and that can be answered by interrogating the data?	10
2	Identify and collect the necessary requirements for the database based on the purpose, questions, data and information needs.	10
3	Translate the identified requirements into a relational database design: create the conceptual, logical and physical models (ERD models) and take care of normalization (to the extent to which normal forms are possible).	25
4	Implement your database using DB Browser and load data using the provided tables.	15
5	Carry out queries on the implemented database and answer your questions. Discuss your results.	30
6	Write a final report that summarizes the outputs of all the previous tasks (i.e., all outputs from the previous tasks are included + write a short teamwork reflection).	10
Total		100

1. Given the above scenario, what could be interesting questions that DonorChoose.org might have and that can be answered by interrogating the data?

Start with group brainstorming and identify at least **3 questions** of interest and explain why they are relevant for DonorChoose.org. You could also think of a **business problem** with **3 aspects or dimensions** to solve by interrogating the database.

During classes, tutorials and exercises, we saw several examples of formulating questions addressing specific organizational (business) needs and/or problems. You should think of questions (or a problem) that can be answered with the data at hand by setting the right relational database requirements (see Task 2), creating database models (Task 3), implement the database in DB Browser for SQLite (Task 4) and running correct SQL queries of various complexity (see Task 5 for requirements) to find an answer to your questions. Last, report and communicate your results to Donorschoose.org (Task 6).

2. Identify and collect the necessary requirements for the database based on the purpose, questions, data and information needs.

In class, we learnt that data management starts on day 1 with a lot of planning. Using DM plan frameworks, like the one provided by CESSDA, can help you with that. Have a look at the provided tables and their description above. For this task, you can use the following guideline questions:

- What data would help you in answering the proposed questions (Task 1)?
- What types of data might be useful?
- How can data be organized? What are the entities of importance to design this database? Select **at least 3** (see Task 3)
- Thinking about the quality of the (provided) data: are they complete, accurate? Do you spot any unusual data formats or issues in the data? Here you are not necessarily asked to fix all these issues now (some can be fixed, others not) but is important is that you are aware of the issues beforehand. You might want to avoid using data with low quality if not necessary. Otherwise, if you really need that piece of data, then you know it in advance about potential issues and can do the proper cleansing afterwards.
- What about normalization? Can you already foresee potential issues that you must be solved when implementing the database?

In answering these questions, team brainstorming and discussion are again very important as in Task 1. Keep track of what you discuss and decide, and **write** the identified requirements and any important things that you have to keep in mind when implementing the database (e.g., you can write a list of action points or 'to-do' list).

3. Translate the identified requirements into a relational database design: create the conceptual, logical and physical models (ERD models) and take care of normalization (to the extent to which normal forms are possible).

a. Draw the **conceptual model** that contains the entities identified in Task 2. Describe all your entities. Every entity should have at least one relationship connecting it to another entity and all entities need to be linked to each other (even if just through a third entity). There should be at least one many-to-many relationship.
In addition, do not forget to include cardinalities (i.e., full relationship information). Please explain the meaning of the relationships in the conceptual ERD using relationship sentences (for examples, see the SQL tutorials and related exercises).

b. Draw the **logical model** by choosing the appropriate primary keys (mark them with '(PK)", relevant attributes and data types. Briefly explain your choice of primary keys and be sure to define the meaning of each attribute.

c. Draw the **physical model** to be implement in DB Browser SQLite by resolving all many-to-many relationships. In addition to marking the primary keys with "(PK)", mark all foreign keys with "(FK)".

d. When designing the ERDs, you also have to take care of **normalization** so the tables meet these requirements as well. Do all the tables are properly normalized? Explain why and the appropriate transformations that are needed (e.g., write about normalized relations and functional dependencies). You might find useful to follow these guiding questions from Exercise 2, Part 1:
Does the table conform to 1NF? If not, which columns should be changed in what way in order to conform to 1NF? Which column(s) is/are the primary key?

If you change the table, does it conform to 2NF? If not, what should be changed? Next, does the table conform to 3NF? If not, what should be changed?

4. Implement your database in DB Browser and load data using the provided tables.

Now, it is time to move to SQL:

- Create a new database in DB browser and save it as a Project file '**Group#_databaseproject.sqbpro**'. In this way, all the queries can be saved and easily checked during grading.
- Write SQL statements to create each table of your physical ERD, including correct data types and setting all needed constraints. Use comments to explain your queries: This will increase their usability!
  Explain any choice you made about constraints. *Tip: when setting the constraints, in particular the FKs, the order of creating table is very important to avoid incurring in foreign key constraints. If you need to refresh how to set FK, you can check https://www.sqlitetutorial.net/sqlite-foreign-key/.*
- Populate the database tables using the provided data by writing appropriate SQL statements. *When importing the provided tables (1&2) into DB Browser, make sure to use semicolon as field separator and double check that datatypes are correct (sometimes types are set by default but they might be wrong).*
- All should be properly working and in line with your ERD!

5. Carry out queries on the implemented database to answer your initial questions. Discuss your results.

Write **queries** to answer your questions or problem. Some questions or problem dimensions might require more than one query to be answered/solved. Write clearly what a query does in each step. If needed, motivate specific choices made.

Important requirements for queries. Your set of queries must include:

- regular expressions, scalar functions and analytical functions (of which, at least 1 window function);
- date/time data;
- Joins operations, where at least there is
  - 1 query with 3 JOIN operations
  - 1 query with 1 LEFT JOIN. *Tip: think about your ERD models: which entities might have missing value for another entity?*
- 1 view
- 1 index: motivate why you create it and show proof that SQLite uses it
- 1 trigger, you can choose between:
  - Create a trigger that prevents updates on columns that you consider important and cannot be changed.
  - Create a trigger that records operations into a log table (e.g., insert, update, delete data) if you plan to manipulate the database in any way after it is implemented.

Last, write a short **discussion** of the obtained results so to provide an answer to your initial questions/problem.

For saving SQL syntax in Tasks 4 & 5. In principle, you can use the tabs in "Execute SQL" in DB Brower to write and save your queries. Name each tab properly so then it is easy to recollect them in the final report. Again, use commenting can be useful. Be sure to save all your queries in a **txt.file** as well for backup and submit it together with the other files.

6. Write a final report that summarizes the outputs of all the previous tasks.

The final report is a written, well-organized collection of all the outputs of the previous tasks (1-5). In addition, the report should demonstrate that you are able to find an answer to your questions (or solution to your problem) and to report and communicate the obtained results in an appropriate way.

The report must be structured in sections following the tasks' order and using a clear title. More specifically, you must include:

- A separate cover page indicating the title of your project, the full names of the group members (with e-mail and student numbers) and assigned Group Number.
- An introduction section with a description of the scenario and your questions/problems, and why they are relevant (Task 1).
- Overview of your data management plans (Task 2), including a short summary of discussion points, needed data, identified requirements and any relevant considerations about data quality and/or foreseen normalization issues.
- Entity Relationship Models with all related documented outputs, such as diagrams, descriptions, explanations, normalized relations and functional dependencies (as required in Task 3).
- Database implementation description with all queries used in the process (Task 4).
- The queries for database interrogation and related outputs. Report and discuss your findings in relation to your questions (Task 5).
- Conclusions with short reflection (max 150 words) on the group's experience with the project as a DM team (which steps were the most difficult? Which were the easiest? what is the main lesson learnt?).

Important remarks about Task 4 and Task 5 sections of the report:

In these sections, you have to include the syntax used and put it into context. For instance: *"To implement our database in DB browser, we first need to create a table for entity1 using a CREATE statement in SQL, as follows [either copy-paste the text of the query or make a screenshot]."*

Explain any query choices you made, in particular when required in the task (e.g., motivate the use of indexes in Task 5). For instance, *To speed up querying on column1, we decided to create an index called 'idx_index" using the following query [either copy-paste the text of the query or make a screenshot]. We did this because ...."*

For reporting the outputs of data interrogation (Task 5), you can either use a screenshot of the DB Browser output or a part of it. *Optional: feel free to use data visualizations tools to report your results (either using the Plots integrated in DB Browser or R, or other free DM* *visualization tools, such as Tableau or Power BI- limited user versions). Remember that SQL can be easily implemented in other environments, so feel free to explore your adventure there.*

Outputs required at submission

The group assignment requires the submission of the following:

1) A **database project file** as a Project 'Group#_databaseproject.sqbpro' including all the needed data and tables as well as commented SQL Syntax. Syntax should be saved in the 'Execute SQL" tabs AND also into **separate txt file** for backup. Submitting these files is of extreme importance because, during grading, we will run all the queries you provided to check whether they are correct. If we miss something, you will lose points.
2) Final **report** in either PDF or .doc format.

All outputs must be submitted on Canvas via the provided link by the deadline 25/10/2020 (23:59). This deadline must be considered as ultimate date for submission. This implies:

- Late work policy. Groups are expected to turn the assignment on time. There will be no exceptions. Late work is not accepted and will penalize the final grade. There will be an increasing penalty for every day past a missed deadline, as follows: 1 day after deadline: 25%; 2 days after deadline: 50%; 3 days after deadline: 75%. Example: if students submit within 1 day after the deadline, 7 becomes 7*0.75 = 5.25.
- Groups are welcome to submit the completed assignment at any time/day before the deadline.

Grading

The assignment must be completed as a group effort. Students are expected to contribute equally to the group assignment. The grade will be equal for all members of a group. The group grade is the results of the execution of each task and final report. All tasks will be examined against the expected answers and points will be given per task as explained in Table 1. 100 points equal to a grade of 10.

What is / is not allowed for this assignment?

- The assignment is supposed to be completed within your group, as a group effort.
- You are free to use the SQL lecture materials, textbooks, or Internet resources that explain the SQL concepts.
- Communication across groups (e.g., how to solve a task, what SQL concepts and syntax are required, etc.) is not allowed. Groups can ask clarification questions about the assignment tasks via the Canvas discussion forum or in class; questions should be limited to general matters (and not solutions) that can be beneficial all groups.
- If there is suspect of plagiarism or cheating, action will be taken accordingly.