**Feedback**

Step 1 Feedback

Dallas Mulkey

* Hi Connor and Tristan, here is my review of your draft.
* **Does the overview describe what problem is to be solved by a website with DB back end?**
* The overview states that the goal of STEMLeaf is to help its users learn a desired subject, as well as to track their progress in that subject.
* **Does the overview list specific facts?**
* It mentions the current membership size of 5,000 members as well as having 100 courses available.
* **Are at least four entities described and does each one represent a single idea to be stored as a list?**
* There are more than 4 entities, however I think that some of these entities could be combined into others. An entity such as password could probably be integrated into both users and staff. I also think that there is probably a better way to track all of the "has" tables, such as creating an extra list item in another table.
* **Does the outline of entity details describe the purpose of each, list attribute datatypes and constraints and describe relationships between entities?**
* The outline describes most of the entities, however all of the "has" entities in the ERD are not listed.
* **Are 1:M relationships correctly formulated? Is there at least one M:M relationship? Does the ERD present a logical view of the database?**
* The 1:M relationships between Subjects and Courses is correct, as one subject can have many different courses. I don't see any M:M relationships, I think that it should be Users to Courses, as multiple Users can be enrolled in multiple courses.
* **Is there consistency in a) naming between overview and entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?**
* Naming between overview and ERD are consistent.
* The naming for entities is consistent other than Staff, maybe renaming it Teachers would be more fitting? Also the attributes such as enrollments\_id might be better suited as enrollment\_id. But the naming convention is consistent, word\_word and no uppercase.
* All entity names are capitalized, no attributes are.
* Overall I think you guys have a solid base, and with a few tweaks to your schema you should have a great database!

Khadeeja Sajjad

* Does the overview describe what problem is to be solved by a website with DB back end? The overview clearly describes the project goals and details. It states that the purpose of the project is to help
* Does the overview list specific facts?
* Are at least four entities described and does each one represent a single idea to be stored a s a list?
* Does the outline of entity details describe the purpose of each, list attribute datatypes and constraints and describe relationships between entities?
* Are 1:M relationships correctly formulated? Is there at least one M:M relationship? Does the ERD present a logical view of the database?
* Is there consistency in a) naming between overview and entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?

Kavya Kolavasi

* STEMLeaf Team,
* **Does the overview describe what problem is to be solved by a website with DB back end?**
  + I think that the overview is appropriate for the Database Outline that was created. It appears that the goals is to track the progress of users and the subjects they are partaking in. Through the database the progress is monitored by looking at the enrollment details provided in the outline.
  + I think in general it is important to have a proper database especially when tracking the progress of students. This is also true for us in Oregon State University and therefore a good idea for a project idea as it is very relevant to us!
* **Does the overview list specific facts?**
  + The overview does list specific facts that help the user to understand the overall goals of the database and what it hopes to achieve. There are many courses (100 courses) offered and many members (5000 members) to keep track of. Therefore, it is important to have a database that is powerful enough to do so. Overall good job, but I just had a few suggestions below.
* **Are at least four entities described and does each one represent a single idea to be stored as a list?**
  + There are more than four entities listed but it may be the case that certain entities might be used as a single attribute inside of another entity.
  + Users: This entity indicates the users of the service with first and last name, along with email.
  + Passwords: This entity would have the password or security information for each user. Generally 1:1 relationships are not recommended, but I can see an understanding for this because it is for security purposes.
  + Staff: The use here records the staff their corresponding subjects. I think here, I would suggest using a M:1 relationship as many staff can teach one subject.
  + Subjects: Records subjects offered. Again you can use a 1:M relationship here with staff if one subject can be though by multiple staff members.
  + Courses: records the courses available. This entity may be a bit redundant with the subjects entity.
  + Enrollments: The enrollments here is appropriate to record the students enrolled per course.
  + All other attributes look good.
* **Does the outline of entity details describe the purpose of each, list attribute datatypes and constraints and describe relationships between entities?**
  + The purpose of each entity is described appropriately. There is also a link between the students and staff and the courses available. Each relationship is described, but I have listed some suggestions above of possible relationships that could be reviewed.
* **Are 1:M relationships correctly formulated? Is there at least one M:M relationship? Does the ERD present a logical view of the database?**
  + Generally a 1:1 entity relationship is not always used but I can understand the need here possibly for security purposes. It could also be possible to incorporate it as an attribute for Users and Staff.
  + For the Staff entity, you could also implement a M:1 relationship for multiple staff per subject.
  + In the Subjects entity, 1:M between subjects and courses is used correctly!
  + There also appear to be many M:M relationships used and seem appropriate. Some of these could possibly change if the group chooses to combine subjects and courses.
  + The ERD looks good with the current entities. As suggested in another comment, this could be changed slightly after review.
* **Is there consistency in a) naming between overview and entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?**
  + The naming seems to be appropriate between the entities and attributes. There is no confusion as to what the attributes could be referring too.
  + Staff could possibly be changed to Instructors to account for the plural condition. The attributes are all singular.
  + The attributes are all lower case and appropriate. Entities are capitalized. Good job on the naming conventions here.

David McGee

* Does the overview describe what problem is to be solved by a website with DB back end?
  + Yes, the problem identified is to identify/track the site users and their desired STEM course(s) and progress in those courses.
  + The problem statement is clear and matches the project requirements.
  + The scope is large (5000 users).
* Does the overview list specific facts?
  + The overview lists that the scope is 5000 users and that there are 100 STEM courses.
* Are at least four entities described and does each one represent a single idea to be stored a s a list?
  + There are 6 identified entities: Users, Passwords, Staff, Subjects, Courses and Enrollments. It is not immediately clear to me why Passwords and Subjects are separate entities- it seems like their attributes would be better represented as attributes of the Users and Courses entities.
* Does the outline of entity details describe the purpose of each, list attribute datatypes and constraints and describe relationships between entities?
  + Yes, each entity lists a primary key, at least one datatype for each attribute, and >=2 relationships. The only exceptions are the grade and date attributes that are missing constraints.
  + The purposes are described for each entity, but could use additional detail For example, the courses identity purposes says it records who is using the course but there isn't an atttribute to do that - instead there is an attribute to identify the staff member.
* Are 1:M relationships correctly formulated? Is there at least one M:M relationship? Does the ERD present a logical view of the database?
  + the 1:M relationship identified is between subjects and courses. It appears to be correctly formulated.
  + There is no M:M relationship identified. I think the Users to Subjects and the Users to Courses could be represented as M:M.
  + The 1:1 relationships seem to validate that Passwords is an unneccessary entity / same with Subjects.
  + The ERD represents a logical view of the database. Recommend that the above consolidation of entities be implemented to reduce additional complexity while remaining under the 4 Entity requirement of the project.
* Is there consistency in a) naming between overview and entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?
  + Capitalization is implemented correctly and equivalently across Entities and Attributes
  + All entities (except Staff) are plural and all attributes are singular.
  + There is consistency in naming implemented.

Step 2 Feedback

Tee Nguyen

* Does the schema present a physical model that follows the database outline and the ER logical diagram exactly?
  + Yes it does!
* Is there consistency in a) naming between overview, outline, ER and schema entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?
  + There are some capitalization issues between the schema and the outline (eg staff\_id and Staff\_id. This should be corrected.
* Is the schema easy to read (e.g. diagram is clear and readable with relationship lines not crossed)?
  + The schema is clear and easy to understand. The relationship lines do not appear to be crossed.
* Are intersection tables properly formed (e.g. two FKs and facilitate a M:N relationship)?
  + The Instructors table has two FKs and facilitates a M:N relationship.
* Does the sample data suggest any non-normalized issues, e.g. partial dependencies or transitive dependencies?
  + The sample data does not suggest any non-normalized issue like partial dependencies or transitive dependencies.
* Is the SQL file syntactically correct? This can be easily verified by using PhPMyAdmin and your CS 340 database (do not forget to take backup of your own database before you do this!)
  + SQL file imported fine, no issues adding database.
* In the SQL, are the data types appropriate considering the description of the attribute in the database outline?
  + Only slight nitpicking with the size of the variables used, but not a big deal. Exploring more efficient data types could be an option.
* In the SQL, are the primary and foreign keys correctly defined when compared to the Schema? Are appropriate CASCADE operations declared?
  + All PKs and FKs are correctly defined. CASCADE is set up for deletion.
* In the SQL, are relationship tables present when compared to the ERD/Schema?
  + Relationship tables are present when compared to the schema.
* In the SQL, is all example data shown in the PDF INSERTED?
  + the screenshots are present in the pdf.
* Is the SQL well-structured and commented (e.g. hand authored) or not (e.g. exported from MySQL)?
  + the SQL is well-structured and commented.

Nira Patel

* **Does the schema present a physical model that follows the database outline and the ER logical diagram exactly?**
* Yes, the schema closely aligns with both the database outline and the ERD.
* **Is there consistency in a) naming between overview, outline, ER and schema entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?**
* There is consistency in naming between the overview, outline, ER and schema, though there is a slight issue with staff being singular while other entities are plural.
* **Is the schema easy to read (e.g. diagram is clear and readable with relationship lines not crossed)?**
* The schema is reasonably easy to read.
* **Are intersection tables properly formed (e.g. two FKs and facilitate a M:N relationship)?**
* Yes, intersection tables are properly formed with two foreign keys referencing the respective tables, showing the M:N relationship between staff and courses.
* **Does the sample data suggest any non-normalized issues, e.g. partial dependencies or transitive dependencies?**
* The sample data seems to be normalized with no issue.
* **Is the SQL file syntactically correct?**
* Yes, the SQL file appears to be correct, follows MySQL syntax.
* **In the SQL, are the data types appropriate considering the description of the attribute in the database outline?**
* Data types in the SQL file generally align with the description of attributes in the database outline.
* **In the SQL, are the primary and foreign keys correctly defined when compared to the Schema? Are appropriate CASCADE operations declared?**
* Yes, primary and foreign keys are correctly defined in the SQL file, so references are correct. CASCADE is used for deletion.
* **In the SQL, are relationship tables present when compared to the ERD/Schema?**
* Relationship tables are present in the SQL file, matching the ERD and schema.
* **In the SQL, is all example data shown in the PDF INSERTED?**
* Yes, example data provided in the PDF is inserted into SQL file without issue.
* **Is the SQL well-structured and commented (e.g. hand authored) or not (e.g. exported from MySQL)?**
* The SQL file is well structured and commented.
* Good work!

Gervasius Juanda

* Does the schema present a physical model that follows the database outline and the ER logical diagram exactly?
  + **The schema follows the database outline quite nicely as it addresses everything written in the outline in a diagram form.**
* Is there consistency in a) naming between overview, outline, ER and schema entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?
  + **I'm seeing a good use of uniform naming conventions as well as** **capitalizations. However, I am seeing an unconsistency where 'Staff' is labelled in singular while other entities are in plural. It would be good to change this to match the other entities.**
* Is the schema easy to read (e.g. diagram is clear and readable with relationship lines not crossed)?
  + **The schema is really easy to read, by giving enough space within each tables and clearly defined arrows.**
* Are intersection tables properly formed (e.g. two FKs and facilitate a M:N relationship)?
  + **The ERD or schema does not seem to contain any intersection tables explicitly, but the Instructors table serves as an intersection table with two foreign keys. So, it is properly formed.**
* Does the sample data suggest any non-normalized issues, e.g. partial dependencies or transitive dependencies?
  + **From what I've seen, the database has already been normalized and issues like partial or transitive dependencies are not present.**
* Is the SQL file syntactically correct? This can be easily verified by using PhPMyAdmin and your CS 340 database (do not forget to take backup of your own database before you do this!)
  + **Syntax wise, the SQL file has been correctly following conventions and looks good.**
* In the SQL, are the data types appropriate considering the description of the attribute in the database outline?
  + **I would say for the most part, the data types are already appropriate to the descriptions. However, I personally think using other data types like decimal(19,2) would be better for chg\_hour in the Staff table.**
* In the SQL, are the primary and foreign keys correctly defined when compared to the Schema? Are appropriate CASCADE operations declared?
  + **All primary and foreign keys are all properly defined inside the SQL file, as well as foreign keys having appropriate ON DELETE CASCADE operations.**
* In the SQL, are relationship tables present when compared to the ERD/Schema?
  + **Considering the ERD is made using phpMyAdmin from the database itself, it looks like all the relationship tables are present in the SQL itself.**
* In the SQL, is all example data shown in the PDF INSERTED?
  + **The screenshots for the example data provided in the PDF are from the phpMyAdmin API, and they are all present inside the SQL.**
* Is the SQL well-structured and commented (e.g. hand authored) or not (e.g. exported from MySQL)?
  + **The SQL is well-commented and well-structured, and I can see that it is clearly hand authored, as well as not being an import from mySQL.**

Katherine Sandeen

* Great draft, Group 81!
* It was hard to find any errors in this, but I dove as deep as I could to find you some feedback.
* Here is my review:
* Does the schema present a physical model that follows the database outline and the ER logical diagram exactly?
  + Yes, the schema follows the outline exactly. The entity names match the table titles, the attributes, and their data types are listed correctly and the primary and foreign keys are referenced in the appropriate places. All relationships are described in the outline and reflected correctly on the schema.
* Is there consistency in a) naming between overview, outline, ER and schema entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?
  + a) The PK for Users, Staff, Courses, and Enrollments is just 'id' on the ERD, but in the Outline and Schema, they are called 'user\_id', 'staff\_id', 'course\_id', and enrollment\_id' respectively. Under 'Instructors' in the ERD, 'Courses\_id' and 'Staff\_id' are uppercase, but lowercase everywhere else.
  + b) All the entity names are plural and the attributes are singular, consistently throughout the document.
  + c) Capitalization is used in the naming of entities, and snake case is used for the naming of attributes in lieu of camel case.
* Is the schema easy to read (e.g. diagram is clear and readable with relationship lines not crossed)?
  + The schema is easy to read for the most part. There is a slight overlapping of line where both the Staff and Enrollments tables both key into Users, and there is a convergence of the relationship line between Users and Enrollments on the corner of the Users table. These could be slightly straightened, but I'm nitpicking because this project is very well done, and I have to nitpick anything I can find!
* Are intersection tables properly formed (e.g. two FKs and facilitate a M:N relationship)?
  + The Instructors table is properly formed with a foreign key each from the Staff and Courses table. This intersection table uses the composite of the two FKs to create the primary key. This combination is properly constructed in the SQL file, but the document outline could be made stronger by adding that detail to the Instructors specification.
* Does the sample data suggest any non-normalized issues, e.g. partial dependencies or transitive dependencies?
  + The sample data does not indicate any partial or transitive dependencies. All attributes appear to be properly and fully dependent to the entity under which they are defined. The database project is normalized.
* Is the SQL file syntactically correct? This can be easily verified by using PhPMyAdmin and your CS 340 database (do not forget to take backup of your own database before you do this!)
  + I was able to import the SQL file into my database without error or issue.
* In the SQL, are the data types appropriate considering the description of the attribute in the database outline?
  + The status attribute in the Enrollments table is correctly cast to a tinyint data type, but can be limited to 1 integer, instead of the default 4, since it is being used as a boolean value (tinyint(1)). Is the chg\_hour in the Staff table so large that it must be cast as a double instead of a float?
* In the SQL, are the primary and foreign keys correctly defined when compared to the Schema? Are appropriate CASCADE operations declared?
  + All the primary and foreign keys are correctly defined and the CASCADE operations are set for the deletion of foreign keys, but not their update.
* In the SQL, are relationship tables present when compared to the ERD/Schema?
  + All relationship tables are present and defined in the SQL file in comparison to the ERD and Schemas represented on the document.
* In the SQL, is all example data shown in the PDF INSERTED?
  + All the example data shown on the PDF is inserted by the SQL file without error.
* Is the SQL well-structured and commented (e.g. hand authored) or not (e.g. exported from MySQL)?
  + The SQL file is beautifully structured, commented well, and easy to read. It is hand-authored instead of exported from MySQL.

Step 3 Feedback

Carly Sherman

* *Does the UI utilize a SELECT for every table in the schema?*
  + I am seeing a select for every table yes!
* *Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?*
  + Yes, all tables also have a more specific select for getting data!
* *Does the UI implement an INSERT for every table in the schema?*
  + There is an insert for every table where users can insert to any page they would like.
* *Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?*
  + The intersection table between staff and courses does seem to successfully add the FK of both tables involved
* *Is there at least one DELETE, and does at least one DELETE remove things from a M:M relationship?*
  + There is at least one delete and it is part of the M:M relationship but doesn't delete the actual staff or courses rather just their pairing!
* *Is there at least one UPDATE for any one entity?*
  + There do seem like there is updates for all the entities too, yes!
* *Is at least one relationship NULLable?*
  + All the FK are set to being Not NULL so there wouldn't be one NULLable relationship, however, this could be easily fixed by finding an optional relationship. Maybe one like teachers and courses where one teacher could potentially not have any active courses.
* *Do you have any other suggestions for the team to help with their HTML UI? For example, using AS aliases to replace obscure column names such as fname with First Name.*
  + Everything looks good to me, simple and understandable. It might be easier to just have any Entity\_ID be listed as Entity ID in the html since it doesn't have to be as formally written but that is just me being specific ahah! Other than that I would maybe just add CSS to make it a bit more aesthetically pleasing!

Kimberly Wooten

* *Does the UI utilize a SELECT for every table in the schema?* In other words, data from each table in the schema should be displayed on the UI. Note: it is generally not acceptable for just a single query to join all tables and displays them.
  + It does show data in every table for Courses, Enrollments, Staff, Instructors, Users, Passwords allowing for us to view all the data.
* *Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?*
  + Yes, all of the tables have the select option which are showing the data available
* *Does the UI implement an INSERT for every table in the schema?* In other words, there should be UI input fields that correspond to each table and attribute in that table.
  + It does, each table has an option for inserting a new row. Each insert option also has each attribute that corresponds to the schema.
* *Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?* In other words if there is a M:M relationship between Orders and Products, INSERTing a new Order (e.g. orderID, customerID, date, total), should also INSERT row(s) in the intersection table, e.g. OrderDetails (orderID, productID, qty, price and line\_total).
  + You have a M;N relationship between courses and staff with an intersection table as instructors.
* *Is there at least one DELETE, and does at least one DELETE remove things from a M:M relationship?* In other words, if an order is deleted from the Orders table, it should also delete the corresponding rows from the OrderDetails table, BUT it should not delete any Products or Customers.
  + Yes, there is a delete on every table as an option so a row can be deleted. You do have the intersection table instructors, which won't effect staff or courses when deleted.
* *Is there at least one UPDATE for any one entity?* In other words, in the case of Products, can productName, listPrice, qtyOnHand, e.g. be updated for a single ProductID record?
  + There is an update on each table, so for all entities, as an option to use.
* *Is at least one relationship NULLable?* In other words, there should be at least one optional relationship, e.g. having an Employee might be optional for any Order. Thus, it should be feasible to edit an Order and change the value of Employee to be empty.
  + There isn't a nullable relationship so it's showing that all relationships are required.
* *Do you have any other suggestions for the team to help with their HTML UI? For example, using AS aliases to replace obscure column names such as fname with First Name.*
  + Just to update to have a nullable relationship!

Cameron Canfield

* ***Does the UI utilize a SELECT for every table in the schema?*** **In other words, data from each table in the schema should be displayed on the UI. Note: it is generally not acceptable for just a single query to join all tables and displays them.**
* Yes the UI uses a select to grab every table. The tables include Courses, Enrollments, Staff, Instructors, Users, and Passwords.
* You have them separated by pages which is the approach I took as well. After scrutinizing the details from the schema to make sure it matches, you did a good job of that. Each data entry that was present in the schema is present in the UI.
* ***Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?***
* Yes, actually each entity has a specific select that allows the user to specify and dynamically populate a list of properties. All of them use the entity id, or primary key.
* ***Does the UI implement an INSERT for every table in the schema?*** **In other words, there should be UI input fields that correspond to each table and attribute in that table.**
* Yes each table has an "Add" section where the user can create a new database entry.
* ***Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?*** **In other words if there is a M:M relationship between Orders and Products, INSERTing a new Order (e.g. orderID, customerID, date, total), should also INSERT row(s) in the intersection table, e.g. OrderDetails (orderID, productID, qty, price and line\_total).**
* Yes there is a M:N relationship between staff and courses. The Staff gets populated before the Instructors (intersection) so when ever a new staff member gets added it also gets added into the instructors.
* ***Is there at least one DELETE, and does at least one DELETE remove things from a M:M relationship?*** **In other words, if an order is deleted from the Orders table, it should also delete the corresponding rows from the OrderDetails table, BUT it should not delete any Products or Customers.**
* Yes there's a DELETE for every table and the intersection table deletes both the staff\_id\_key and course\_id but doesn't delete it from their respective tables.
* ***Is there at least one UPDATE for any one entity?*** **In other words, in the case of Products, can productName, listPrice, qtyOnHand, e.g. be updated for a single ProductID record?**
* Yes all entities have the option of updating any of the information. It's a little barebones right now and hard codes it so you can only update the first entry of every table. Obviously, I'm sure this if for draft purposes and this won't be your final design.
* ***Is at least one relationship NULLable?*** **In other words, there should be at least one optional relationship, e.g. having an Employee might be optional for any Order. Thus, it should be feasible to edit an Order and change the value of Employee to be empty.**
* After looking over you ddl it would appear that every relationship is required. Every foreign key reference adds NOT NULL therefore making it a mandatory relationship. Perhaps you can make the email address for Users optional meaning that not every user has to have an email. Food for thought.
* ***Do you have any other suggestions for the team to help with their HTML UI? For example, using AS aliases to replace obscure column names such as fname with First Name.***
* I think you guys have a solid UI so far one recommendation I would have would be maybe try your hand at some CSS. As of now you use CSS to add borders around the inputs which is appropriate. There's no color though which would make it a great portfolio project you can put on a resume!

Jacob Caudill

* *Does the UI utilize a SELECT for every table in the schema?* In other words, data from each table in the schema should be displayed on the UI. Note: it is generally not acceptable for just a single query to join all tables and displays them.
  + Yes there is a SELECT for every table in the schema. All data is shown on every page.
* *Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?*
  + Yes there are multiple SELECT queries that filter.
* *Does the UI implement an INSERT for every table in the schema?* In other words, there should be UI input fields that correspond to each table and attribute in that table.
  + Yes there is an add section for each page.
* *Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?* In other words if there is a M:M relationship between Orders and Products, INSERTing a new Order (e.g. orderID, customerID, date, total), should also INSERT row(s) in the intersection table, e.g. OrderDetails (orderID, productID, qty, price and line\_total).
* *Is there at least one DELETE, and does at least one DELETE remove things from a M:M relationship?* In other words, if an order is deleted from the Orders table, it should also delete the corresponding rows from the OrderDetails table, BUT it should not delete any Products or Customers.
  + It appears that it is set up correctly. It would only delete within the certain table that it was declared to delete in.
* *Is there at least one UPDATE for any one entity?* In other words, in the case of Products, can productName, listPrice, qtyOnHand, e.g. be updated for a single ProductID record?
  + As of now it would only update the first attribute in the table, but if this is just for the draft version that's acceptable. Needs elaborated on.
* *Is at least one relationship NULLable?* In other words, there should be at least one optional relationship, e.g. having an Employee might be optional for any Order. Thus, it should be feasible to edit an Order and change the value of Employee to be empty.
  + Every relationship is declared as NOT NULL. There could be some opportunities to create some NULLable relationships. Maybe make the last name optional if you make the email unique. That way someone could sign up without having to make a last name if they have to have a unique email.
* *Do you have any other suggestions for the team to help with their HTML UI? For example, using AS aliases to replace obscure column names such as fname with First Name.*
  + The outline looks really good and the UI is easy to follow. I know this is just a draft but if you were to add some styling to the UI it would be very easy to follow and look ascetically pleasing.

**Actions Taken based on Feedback**

* Subjects offered no value and should not be an entity as seen in the feedback, so it is now a single attribute apart of courses.
* Staff and courses was made into an M:N relationship so staff can instruct several courses.
* Rather than staff and users being separate causing an issue with their relationship to the passwords entity. Staff is now a part of users, and some users are staff, with this, passwords now only has a 1:1 relationship with users.
* Feedback had stated that there were no M:M relationships, but there was between users and enrollments.
* Added attributes so that staff has a biography for users to see, courses had a description, and removed the attribute for when a user finished a course so that enrollment focuses on the status of their completion.
* The M:N relationships between enrollment and users as well as enrollment and courses was transformed into a 1:M relationship. This is because each enrollment has only one student and one course.
* Instructors was made to document the M:N relationship between staff and courses, so documentation can be made about which staff members have access to which courses
* Status inside of enrollments is a TINYINT, there were suggestions made to specify to use only 1 bit for this. However, it shall remain as a TINYINT to account for more than two statuses. At minimum there is completed, in progress, and required.
* chg\_hour for staff was changed to decimal(19, 2) as recommended in the feedback to have more efficient data types.
* The ERD was corrected to have consistent naming for the id’s throughout the database design.
* There was no NULLable relationship, the relationship between staff and users is now nullable. This will allow for a staff member to delete their account but their past information regarding the courses they instructed will remain.
  + It was recommended that the relationship between instructors and courses was nullable, but this is already a transaction that the instructors entity is used to create the M:N relationship with staff and courses. This change was not made.

Project Title: STEMLeaf

CS 340: Project Step 1

Team: Group 81

URL: https://web.engr.oregonstate.edu/~vosburgt/cs340/Step3Final/index.html

Connor Schutze & Tristan Vosburg

**Overview**

STEMLeaf is an online course structured service, offering 100 STEM related courses each with their own explorations and tests. STEMLeaf has 5,000 members, each requiring the use of one or more courses with their saved state. A database driven website will record the information regarding these courses, alongside the information of users partaking in these courses. The overall purpose is to help users learn subjects they wish to learn and track their progress.

**Outline**

The entities a part of the database include users, passwords, staff, instructors, courses, and enrollments. Users will be used to record details of the users using the service so they can login and view their courses. Passwords is used to store the hashed passwords and identify when the last password change was made for a particular user. Staff is an entity used to record the details of the staff so that they can access their courses and make changes with certain permissions. Instructors documents what staff members are instructors what courses along with their biography for that particular course. Courses is an entity that will keep a record of all courses and their subjects. Enrollments is used to keep track of the students in particular courses as well as their status of the course.

To ensure that the schema is in 3 NF, staff was incorporated into users so that the password's primary key was not a combination of any sorts, but instead a unique value determined by the user’s information. Passwords are kept separate for security reasons but have information regarding changes to the password validating its entity to be separate from users. Additionally, staff was made separate from users so staff could have information regarding their biography ensuring that this entity did not create any anomalies or redundant data in users.

Users: records the details of Users using our service

* user\_id: int, auto\_increment, not NULL, primary key
* first\_name: varchar, not NULL
* last\_name: varchar, not NULL
* email: varchar, not NULL, unique, foreign key
* Relationship: a 1:1 between users and passwords
* Relationship: a 1:1 between users and staff
* Relationship: a M:1 between users and enrollments

Passwords: records the details of Users login information for authentication

* email: varchar, not NULL, primary key
* password\_hash: varchar, not NULL
* password\_update: date, not NULL
* Relationship: a 1:1 between users and passwords

Staff: records the Staff’s biography and courses they contribute to

* staff\_id: int, auto\_increment, not NULL, primary key
* hire\_date: date
* chg\_hour: decimal(19, 2)
* user\_id: int, not NULL, foreign key
* Relationship: a 1:1 between users and staff
* Relationship: a M:N between staff and courses
  + Using the Instructors entity

Courses: records the Courses offered on our service and who is using them

* course\_id: int, auto\_increment, not NULL, primary key
* course\_name: varchar, not NULL, unique
* subject: varchar, not NULL
* description: varchar
* staff\_id: int, not NULL, foreign key
* Relationship: a M:N between staff and courses (multiple staff per course)
  + Using the Instructors entity
* Relationship: a M:1 between courses and enrollments

Enrollments: records the Enrollments of users in courses and their status

* enrollment\_id: int, auto\_increment, not NULL, primary key
* status: boolean, not NULL
* grade: varchar
* user\_id: int, not NULL, foreign key
* course\_id: int, not NULL, foreign key
* Relationship: a M:1 between students (users) and enrollments
* Relationship: a M:1 between courses and enrollments

Instructors: used to form a M:N relationship between Staff and Courses

* staff\_id: int not NULL, foreign key
* course\_id: int not NULL, foreign key
* staff\_bio: varchar (unique for each course per instructor

**Entity-Relationship (ER) Diagram**

A diagram of a computer

Description automatically generated with medium confidence

**Scheme**

A screenshot of a computer

Description automatically generated

**Sample Data**

Users

A screenshot of a computer

Description automatically generated

Passwords

A screenshot of a computer

Description automatically generated

Staff

A screenshot of a computer

Description automatically generated

Courses

A screenshot of a computer

Description automatically generated

Enrollments

A screenshot of a computer

Description automatically generated

Instructors

A screenshot of a computer

Description automatically generated