Group 42 Project Step 4 draft

Team Members: Leala Aljehane & Edgar Palaquibay

Project Title: Workplace Organizer

URL: http://flip1.engr.oregonstate.edu:42543/

Fixes based on Feedback from Step 3:

-employee_id in projects_mapping is now default NULL, it's part of the M:N relationship between projects and employees. This was suggested in a feedback, since our database lacked a nullable relationship. In context, this means a project has yet to be assigned employees.

-Included DML queries for project_mapping in order to be able to assign employees to projects

Upgrades to the Draft version:

-implemented our projects_mapping entity in our UI

Step 3 Feedback:

Feedback-1

Hello! Here is my review. Overall, I think there could be a few more changes to make the UI more accessible and intuitive. I also think your entities in general are a little bare bones and you may want to add more attributes. Please see below for the specifics.

1. 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.

There is a SELECT for each entity and the whole table is displayed. The tables are not joined into a single query.

2. Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?

Not yet, there is no functionality for searching specific entities. I suggest adding an attribute under Employees that lists that individual's skills. For example, one employee could have Microsoft Office skills, Java, Python, etc. That way you could filter out available employees who have certain skillsets that may be required for projects and the manager could assign them a task. However, it may require adding a skills entity to implement this idea.

3. 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 are INSERTS for each table according to the UI.

4. Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship? In other words, if there is an 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).

For the single entities, the insertions do add foreign keys. The manager_id foreign key is added into the employee and client entities. The M:M relationship is projects to employees and currently I do not see functionality that assigns employees to a project. As a result, there is also no functionality for inserting into the junction table between the two.

5. Is there at least one DELETE and does at least one DELETE remove things from an 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.

According to the webpage UI, there are DELETEs for every entity and it allows deletion of a single row in each table. In the SQL, this is reflected but there is no DELETE query under managers.

6. 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, UPDATE is allowed for all entities.

7. 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.

At this time, I do not see a NULLable or optional relationship.

8. Do you have any other suggestions for the team to help with their HTML UI?

I think you could make a few meaningful changes in the UI of your webpage. I would put the projects under each client instead of having an entire projects page. If a manager needed to prioritize one client over all the others, you could just see which projects needed to get done by client instead of an entire list of projects. You could also maybe add an attribute under projects for due date. That way, you can search which projects need to be done first.

Feedback-2

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, all of the sample data is displayed on the table on each of their won pages.

Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?

In the DML file, the single client filter is incremented and displays one client based on the selected project_id.

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.

There is an insert operation shown for all entity pages, as well as an update and delete.

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).

Each insert operation adds the corresponding foreign keys for each entity as needed.

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.

There is a delete operation for each entity and the cascade delete operation is used to delete the corresponding rows when they are present in another table.

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?

Every entity in the schema is implemented with an update. I also noticed that the update cascade was implemented so that each corresponding field regarding updates changes as well.

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.

It looks like most of the foreign key fields are nullable in the entities.

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 like how you used some CSS to make the page more differentiated from the other projects I've viewed.

Feedback-3

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.

All entities are represented except for the project_mapping entity. One of the TAs recommended for us to put the 2 main tables that had a M:M relationship on one page

and do a separate page for the intersection table. So in that case you could try one page for Employees/Projects and then project_mapping would have its own.

Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?

Yes, for example, the update SELECT for the managers entity is dynamically populated based on the value of manager id specified in the UI.

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.

No, there is no INSERT for the project_mapping entity. This makes it so that it's not possible to add to the M:M relationship. This should be added in the DML as well after making a page for the project_mapping entity in the UI.

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).

As stated in response to the last question it's not currently possible to INSERT into the M:M relationship between employees and projects. The other insertions do give the option to insert their required FKs.

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.

There is a DELETE for all entities except for managers and the project_mapping entity. Because the project_mapping entity does not have a DELETE there is no way to remove

an entry from the M:M relationship. This can be solved by adding the project_mapping page and implementing CRUD for that table.

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, there is UPDATE for every entity except for project_mapping.

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.

No, there are no relationships that are NULLable. My suggestion is to make it so that you can have projects without employees. This can be changed by allowing employee_id to be NULL in the project_mapping table.

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.

Nice work on the layout of the UI, it looks great! My one suggestion would be allowing FKs to be dropdown menus instead of text boxes in order to make insertion/updates easier. This way the user doesn't have to have those values memorized.

Awesome job! Look forward to seeing the finished product.

Feedback-4

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 utilizes a SELECT for every table. There are not any missing attributes. All entities and their corresponding attributes from the schema are present on the UI.

Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?

No, there is no SELECT present in the UI that utilizes a search/filter function. This will be easy to implement for any of the entities, although I recommend the Projects entity because it will fit with the goal of being used by managers to organize the workplace and keep track of clients and employees.

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.

No, the UI implements an INSERT for 4/5 of the tables in the schema, although I see the 5th INSERT in the DDL file. The intersection table is missing, but it doesn't necessarily have to be there as long as it corresponds to a page which could be the Employees page, Projects page, or both. It would be good to know which projects are attached to employees and be able to search through them.

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, each INSERT present in the UI adds the corresponding FK attributes. Both employees and clients have manager_id listed as a column in the table.

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 at least one DELETE. In fact, there is a DELETE for every entity currently displayed in the UI. No, it does not remove anything from a M:M relationship or

otherwise just yet. The DDL file contains the CASCADE operations for clients and employees, so you will be good to go when you implement the DELETE operation to the UI.

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, there is at least one UPDATE. In fact, there is an UPDATE for every entity currently displayed in the UI. All corresponding attributes to each entity are displayed in the UI.

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.

No, it is not immediately clear there is a NULLable relationship. I recommend this be fixed within Projects or Projects_Mapping (whichever you decide upon) where employee can be null, since you may have new projects not yet assigned by the manager.

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.

No, I do not have any other suggestions at this time. I do want to add that the UI looks good and I wish you best of luck on Step 3 Final Version! Also, make sure you add the website URL to the top of your PDF file to avoid points off.

Fixes based on Feedback from Step 2:

Feedback by the peer reviewer:

Feedback-1

Does the schema present a physical model that follows the database outline and the ER logical diagram exactly?

Yep the Schema looks good, though the ERD is cut off in the PDF, but from the other info it looks good.

Is the schema easy to read (e.g. diagram is clear and readable with relationship lines not crossed)?

Yep its easy to read

Are intersection tables properly formed (e.g. two FKs and facilitate a M:N relationship)?

Yep their intersection tables are good

Does the sample data suggest any non-normalized issues, e.g. partial dependencies or transitive dependencies?

I'm just going to say I actually don't know in this case.

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!)

In the SQL, are the data types appropriate considering the description of the attribute in the database outline?

Yep it looks syntactically correct.

In the SQL, are the primary and foreign keys correctly defined when compared to the Schema? Are appropriate CASCADE operations declared?

Yep the keys are matching the outline side of things.

In the SQL, are relationship tables present when compared to the ERD/Schema?

Yep the tables are the same as the diagrams on the paper

In the SQL, is all example data shown in the PDF INSERTED?

yep all the example data is inserted.

I will also add that some of the SQL was formated a bit oddly, but not incorrectly. And the PDF was hard to read as the the text was spread across the entire page instead of justified left(or is right when its going from left to right)

Feedback-2

Does the schema present a physical model that follows the database outline and the ER logical diagram exactly?

I believe so. Some of the ERD has been cut off in formatting, but what I can see does look good.

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?

Yes, snake_case is used throughout and pluralization is appropriate.

Is the schema easy to read (e.g. diagram is clear and readable with relationship lines not crossed)?

Yes, it's easy to read. There is a pair of crossed lines, but I don't believe that impacts readability.

Are intersection tables properly formed (e.g. two FKs and facilitate a M:N relationship)?

Yes.

Does the sample data suggest any non-normalized issues, e.g. partial dependencies or transitive dependencies?

I don't believe so, but am not super comfortable on this topic yet.

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!)

Looks good! I did not encounter any errors and can DESCRIBE the tables without issue.

In the SQL, are the data types appropriate considering the description of the attribute in the database outline?

Yes.

In the SQL, are the primary and foreign keys correctly defined when compared to the Schema? Are appropriate CASCADE operations declared?

Yes.

In the SQL, are relationship tables present when compared to the ERD/Schema?

Yes.

In the SQL, is all example data shown in the PDF INSERTED?

Yes, the data was successfully inserted and does match.

Feedback-3

Does the schema present a physical model that follows the database outline and the ER logical diagram exactly?

ER diagram only shows the projects and clients table which correctly have a 1:1 relationship consistent with assumptions and schema. The major difference I noticed was that ER diagram projects table lists the task field as varchar(30) while the schema lists the project table task field as varchar(50).

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) Naming is consistent between overview, outline, ER and schema. Underscore consistently used for multiple word attributes.

b) Entities are plural and attributes are singular.

c) Lowercase used in entity names and attributes.

Is the schema easy to read (e.g. diagram is clear and readable with relationship lines not crossed)?

The schema diagram is clear and readable with uncrossed independent relationship lines.

Are intersection tables properly formed (e.g. two FKs and facilitate a M:N relationship)?

Intersection table projects_mapping appears properly formed with two foreign keys.

Does the sample data suggest any non-normalized issues, e.g. partial dependencies or transitive dependencies?

The sample data did not include the employees table. All table designs and relationships under assumptions seem to allow for proper normalization.

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!)

The sql file is mostly syntactically correct. In MySQL workbench, comments for the manager table creation and employees data insert caused a 1064 error. Upon inserting a space after the two dashes resolved the error.

In the SQL, are the data types appropriate considering the description of the attribute in the database outline?

Yes. ID's had data types of integer and descriptive fields had varchar data types.

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 appear correctly defined along with cascade operations.

In the SQL, are relationship tables present when compared to the ERD/Schema?

In the SQL, correct relationship tables are present.

In the SQL, is all example data shown in the PDF INSERTED?

All the data is shown in SQL including the employees table.

Does the schema present a physical model that follows the database outline and the ER logical diagram exactly?

Yes, the ER diagram was posted in a response to the post and follows it follows the database outline.

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?
Yes, there is consistency in naming between the overview, outline, ER, and schema. Each entity is written as a plural while each attribute is written as a singular. In both the schema and the ER diagram, each attribute and name are in lower case.
Is the schema easy to read (e.g. diagram is clear and readable with relationship lines not crossed)?
Yes, there is one point where lines are crossed at the employees entity, but it is still easy to read.
Are intersection tables properly formed (e.g. two FKs and facilitate a M:N relationship)?
Yes, intersection table is properly formed.
Does the sample data suggest any non-normalized issues, e.g. partial dependencies or transitive dependencies?
I believe the data is correctly in 3rd normalized form (in 2nd normalized form and no transitive dependencies) based on my current understanding of the material.
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!)
When importing using PhPMyAdmin I received an error: “#1064 - You have an error in your SQL syntax”
In the SQL, are the data types appropriate considering the description of the attribute in the database outline?
Yes, the data types are appropriate.
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 and appropriate CASCADE operations declared.
In the SQL, are relationship tables present when compared to the ERD/Schema?
Relationship tables present in the SQL.

In the SQL, is all example data shown in the PDF INSERTED?
Yes, all example data is shown.

Fixes based on Feedback from Step 1:

Feedback by the peer reviewer:

Feedback-1:

Hello,

- Does the overview describe what problem is to be solved by a website with DB back end?
    - Yes, managing what employees are working on what task definitely benefits from a database when there are many employees
- Does the overview list specific facts?
    - It would be nice to have some more specific facts, such as the amount of employees. This would give a sense of scale for why a database is needed.
- Are at least four entities described and does each one represent a single idea to be stored as a list?
    - Yes, each entity is distinct.
- Does the outline of entity details describe the purpose of each, list attribute datatypes and constraints, and describe relationships between entities? Does the outline clearly indicate which entities (tables) will be implemented and which team member is primarily assigned to the associated page(s)?
    - Each entity and attribute and relationship is described appropriately. The outline doesn't indicate which team member is responsible for which entity.
- Are 1:M relationships correctly formulated? Is there at least one M:M relationship?

- 1: M relationships seem correct. I would wonder if clients to assignments should be a 1:M relationship for repeat business from the same client though. The M:M relationship is also present, and an intersection table is described, but is not shown on the diagram. I am not sure if we are supposed to show the intersection table on the diagram or not.

- Is there consistency in a) naming between overview and entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?
  - For the most part naming is consistent. There are 2 discrepancies though: emp_name in the employees entity, as it is the only entity to have a shorthand name, and employees_under in the managers entity, as it is the only renamed FK.

Feedback-2:

1. Does the overview describe what problem is to be solved by a website with DB back end?

The overview is clear and concise in its description of the problem, and it's solution.

2. Does the overview list specific facts?

The overview lacks specific facts, such as selecting an example company and listing how many employees it currently has and how many new hires they hired last year as a way of quantifying the scheduling problem.

3. Are at least four entities described and does each one represent a single idea to be stored as a list?

There are four entities: Employees, Assignments, Clients, and Managers. These are all very distinct and can be stored as lists.

However, Managers are also employees, so the team may want to address whether a Manager needs to be a separate entity, or a boolean attribute that an Employee has.

4. Does the outline of entity details describe the purpose of each, list attribute datatypes and constraints and describe relationships between entities? Does the outline clearly indicate which entities (tables) will be implemented and which team member is primarily assigned to the associated page(s)?

The outline does not clearly state the purpose of each entity, but from the overview and the names of the entities it is very easy to tell their purpose.

The outline does not state which team member will develop what, but as far as I can recall that was not a requirement for the outline at this step.

5. 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 relationships seem to be correctly formulated for the most part, however I believe it is an oversight to state that employees can only have one manager, as this is rarely the case in my experience. The ERD is well defined and clear.

6. Is there consistency in a) naming between overview and entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?

The naming conventions are consistent! Good job!

Feedback-3:

- Does the overview describe what problem is to be solved by a website with DB back end?

Yes, it describes the issue and resolution their database will bring.

- Does the overview list specific facts?

Yes, it lists functionality and why this would be a useful tool for management. It could be more specific and include some additional details, as people have mentioned, about the quantity of projects, employees, etc to get a better feel for the scope of the data being stored.

- Are at least four entities described and does each one represent a single idea to be stored a s a list?

Yes, the four entities described represent single ideas to be stored as a list. They are all directly related to one another and present a simple and concise outline.

- Does the outline of entity details describe the purpose of each, list attribute data types and constraints and describe relationships between entities? Does the outline clearly indicate which entities (tables) will be implemented and which team member is primarily assigned to the associated page(s)?

The outline is very easy to understand logically however the specific roles are not explicitly described as to how they are related to one another. A brief description of what the Managers, Employees, Clients, and Assignments keep track of and how they relate to other 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?

Yes, there is a 1:M relationship between Managers and Employees as well as Managers and Clients which seems to be labeled and logically correct. There is also a M:M relationship between Employees and Assignments which again, seems correctly labeled and is logically correct.

- Is there consistency in a) naming between overview and entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?

Yes, consistency across the board. Very nice example of an outline! Great job Leala and Edgar!

Summary of reviews: give examples of why a backend is necessary for this project, mention we are talking about direct managers (so each employee only has 1 direct manager), talk more about numbers and facts such as how big the organizations are, how many managers, employees, and clients.

Actions based on the feedback from step1:

After reviewing the feedback that we have received from the reviewers, we have decided to add a purpose section to each entity outlining the purpose of it. In addition; we have changed the entity name format to be all lowercase characters as reviewers suggested. We also set character restrictions to attributes that were of type varchar in order to save space.

1. Upgrades to the draft version:

   Entities have been assigned to group members, entity assignment has been changed to projects. Overview has been expanded. Some normalization that can be done is having a managers to employee table. This is due to managers managing multiple employees, so the attribute employees_under will have a list of FK of

employees. Instead a separate table can be created where it shows multiple entries for managers, one for each employee they manage.

B. Fixes based on Feedback from Step 1:

Intersection table is now included in the ERD, emp_name attribute was changed to employee_name.

Overview:

As people progress through their careers, they may eventually end up in management level positions. It is a rewarding position, yet it brings many challenges. Managers can lead their teams to success when they have the right understanding of the relationships between their employees, the clients, and the projects. We have looked up at some of the top challenges that managers face, one of which is keeping track of what everyone they manage is doing, work wise.

This workplace organizer will help managers to manage, track, and update projects. It is meant to be used by managers, so they can keep track of who their clients are, what projects were given by clients, and their employees assigned to the projects. This application will be useful because a manager can also update the database if a project is completed, employees are reassigned to different projects, and any updates/deletions are necessary. It also consolidates how managers keep track of employees, their workload, and the clients expectations by using one system to reduce inefficiencies. As a result, managers will be able to solve and manage many conflicts within the team, creating reassurance, and reducing the chance of burn out. It is meant to be used in large scale organizations, being able to handle thousands of entries for each entity.

Database Outline:

managers:

Purpose:

An entity that stores information about managers including manager_id, manager_name, organization, employee_under, client_id. Managers will be able to keep track of employees who work under them and their clients. It shows what organization they are a part of (marketing department, technology department, HR department, legal department)

Attributes:

-manager_id: int, unique, not NULL, PK

-manager_name: varchar(30), not unique, not NULL

-organization: varchar(30), not unique, not NULL

Relationship:

There is a 1:M relationship between manager and employees, since an employee can only have 1 manager, while a manager can have many employees working under them.

Assigned to Edgar Palaquibay

clients:

Purpose:

An entity that stores information about client_id, project_id, and organization. This will demonstrate the relationship of the client with the organization as one client can give only one project.

Attributes:

-client_id: int, auto-increment, not NULL, PK

-project_id: int, unique, not NULL, FK

-manager_id: int, not unique, not NULL, FK

-client_organization: varchar(30), not unique, not NULL

Relationships:

There is a 1:1 relationship between client and projects, each client can only give one project and each project was given by only 1 client.

There is a 1:M relationship between manager and client, since a client has contact with only 1 manager, while a manager can have various clients.

Assigned to Edgar Palaquibay

employees:

Purpose:

An entity that stores information about emloyee_id, manager_id, emp_name, and organization. This entity has a M:M relationship between projects and employees.

Attributes:

-employee_id: int, auto-increment, not NULL, PK

-manager_id: int, not unique, not NULL, FK

-employee_name: varchar(30), not unique, not NULL

-organization: varchar(30), not unique, not NULL

Relationship:

There is a M:M relationship between projects and employees. This will be shown in an intersection table [projects_mapping]. Since a project given by a client can have multiple employees working on it, while an employee can also work on multiple projects.

Assigned to Leala Aljehane

projects:

Purpose:

An entity that stores information about project_id, client_id, and task. Task is a description of the project. A manager can search for a project by using a client_id.

Attributes:

-project_id: int, auto-increment, not NULL, PK

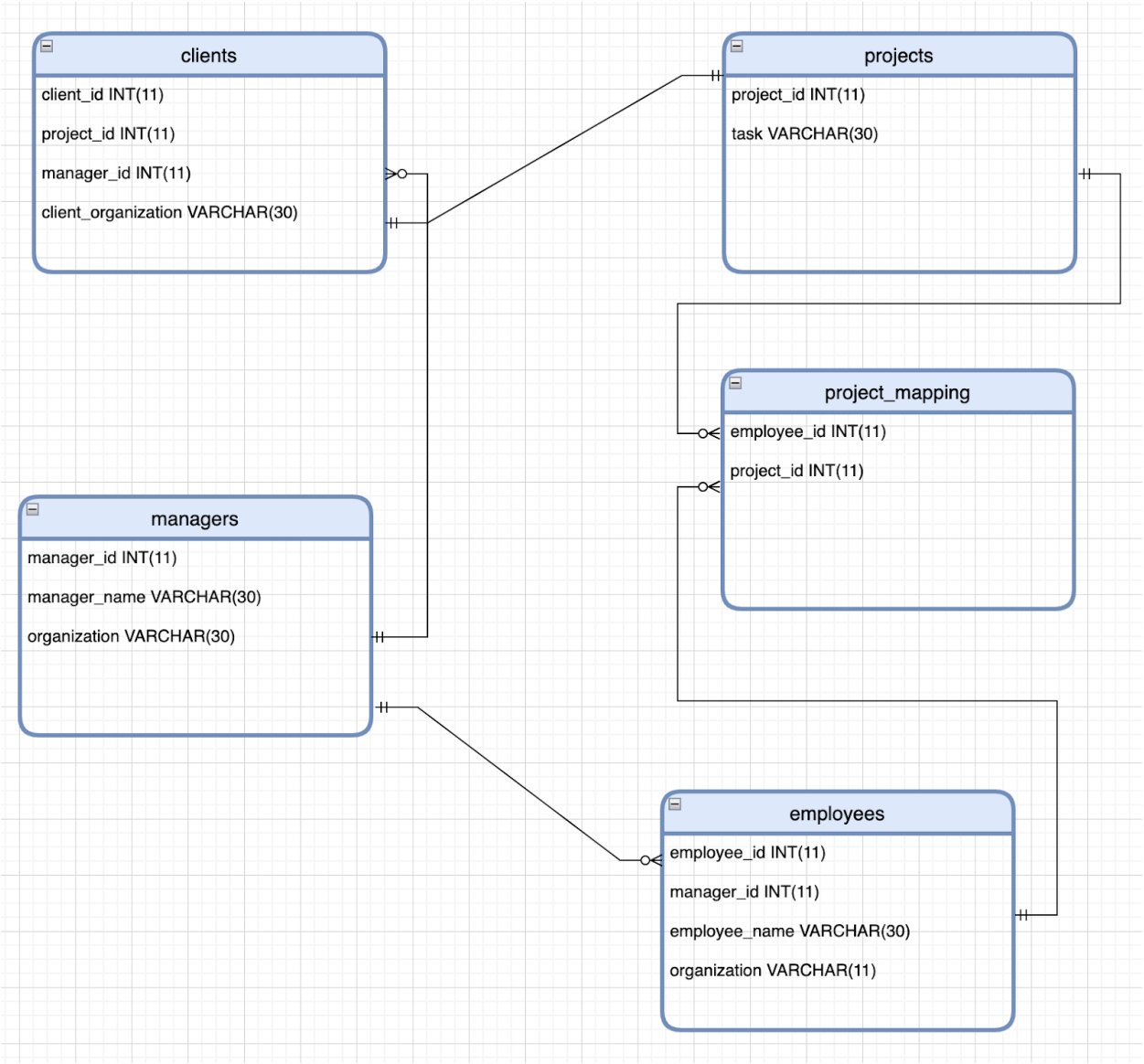
-client_id: int, unique, not NULL FK

-task: varchar(50), unique, not NULL

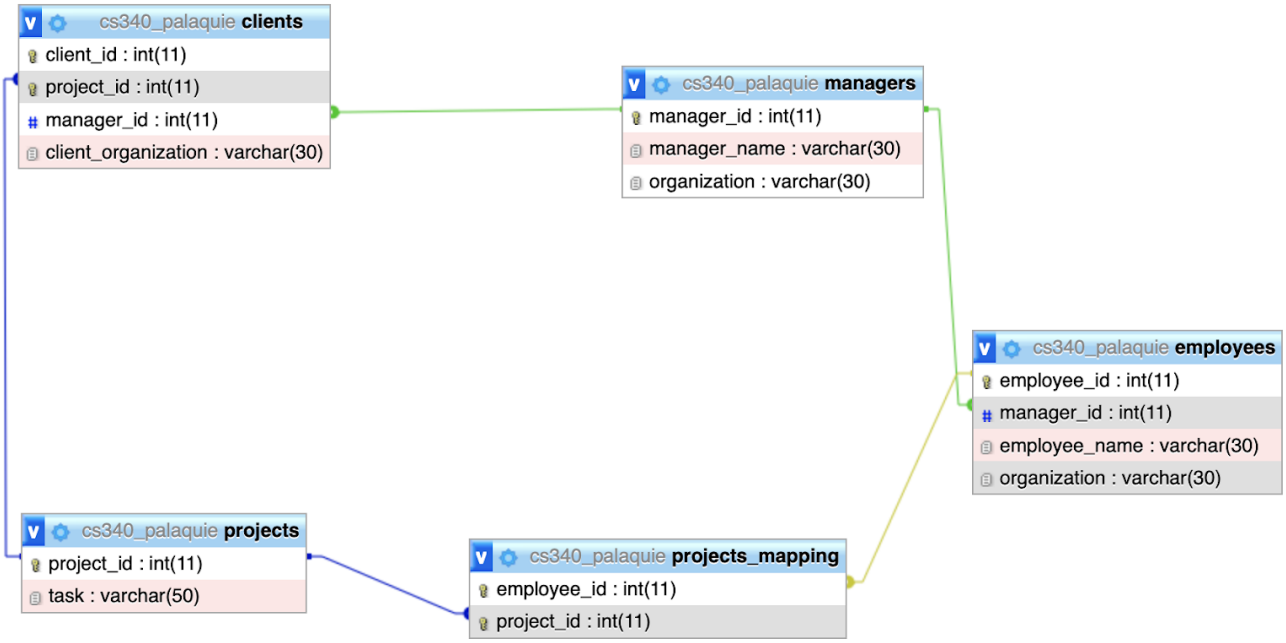
Relationship:

There is a M:N relationship between projects and employees. This will be shown in an intersection table [projects_mapping]. Since a project given by a client can have multiple employees working on it, while an employee can also work on multiple projects.

Entity Relationship Diagram



Schema:


cs340_palaquie clients
client_id : int(11)
project_id : int(11)
manager_id : int(11)
client_organization : varchar(30)
cs340_palaquie managers
manager_id : int(11)
manager_name : varchar(30)
organization : varchar(30)
cs340_palaquie employees
employee_id : int(11)
manager_id : int(11)
employee_name : varchar(30)
organization : varchar(30)
cs340_palaquie projects
project_id : int(11)
task : varchar(50)
cs340_palaquie projects_mapping
employee_id : int(11)
project_id : int(11)

Sample Data:

```
MariaDB [cs340_palaquie]> select * FROM managers;
+------------+-----------------+--------------+
| manager_id | manager_name    | organization |
+------------+-----------------+--------------+
|          1 | Josh Porter     | marketing    |
|          2 | Jan Levinson    | legal        |
|          3 | Toby Flenderson | HR           |
+------------+-----------------+--------------+
3 rows in set (0.001 sec)

MariaDB [cs340_palaquie]> select * FROM clients;
+-----------+------------+------------+--------------------+
| client_id | project_id | manager_id | client_organization |
+-----------+------------+------------+--------------------+
|         1 |          2 |          3 | HR                 |
|         2 |          1 |          2 | marketing          |
|         3 |          4 |          1 | marketing          |
|         4 |          3 |          2 | legal              |
+-----------+------------+------------+--------------------+
4 rows in set (0.000 sec)

MariaDB [cs340_palaquie]> select * FROM projects;
+------------+---------------------------------------+
| project_id | task                                  |
+------------+---------------------------------------+
|          1 | Fix brochure design                   |
|          2 | Update HR policy on gun violence      |
|          3 | fix our liability policies            |
|          4 | design new commercial for new product |
+------------+---------------------------------------+
4 rows in set (0.001 sec)

MariaDB [cs340_palaquie]> select * FROM employees;
+-------------+------------+---------------+--------------+
| employee_id | manager_id | employee_name | organization |
+-------------+------------+---------------+--------------+
|           1 |          1 | Dwight Shrute | marketing    |
|           2 |          1 | Karen Filapeli | marketing    |
|           3 |          3 | Creed Bratton | HR           |
|           4 |          2 | Michael Scott | legal        |
|           5 |          3 | Andy Bernard  | HR           |
+-------------+------------+---------------+--------------+
5 rows in set (0.000 sec)
```

The first table is managers, and all of that data was inserted manually.

The second table is clients, the data was inserted manually for the exception of manager_id and project_id. Both of those columns of data were retrieved with a subquery from the managers and projects tables respectively.

The data for the projects table were inputted manually.

The employees table was inserted manually except manager_id and organization. The manager_id and organization was retrieved via subquery based on manager_id.

```
MariaDB [cs340_palaquie]> select * FROM projects_mapping;
+-------------+------------+
| employee_id | project_id |
+-------------+------------+
|           1 |          1 |
|           3 |          2 |
|           5 |          2 |
|           0 |          3 |
|           4 |          3 |
|           1 |          4 |
|           2 |          4 |
+-------------+------------+
```

Projects_mapping table was created via subqueries. employee_id and project_id were retrieved from the employees and projects table. This is an intersection table which is used to show the M:N relationship between employees and projects. Also employee_id is nullable as a foreign key, the null example is when project_id=3 and employee_id=0 (equivalent to NULL).