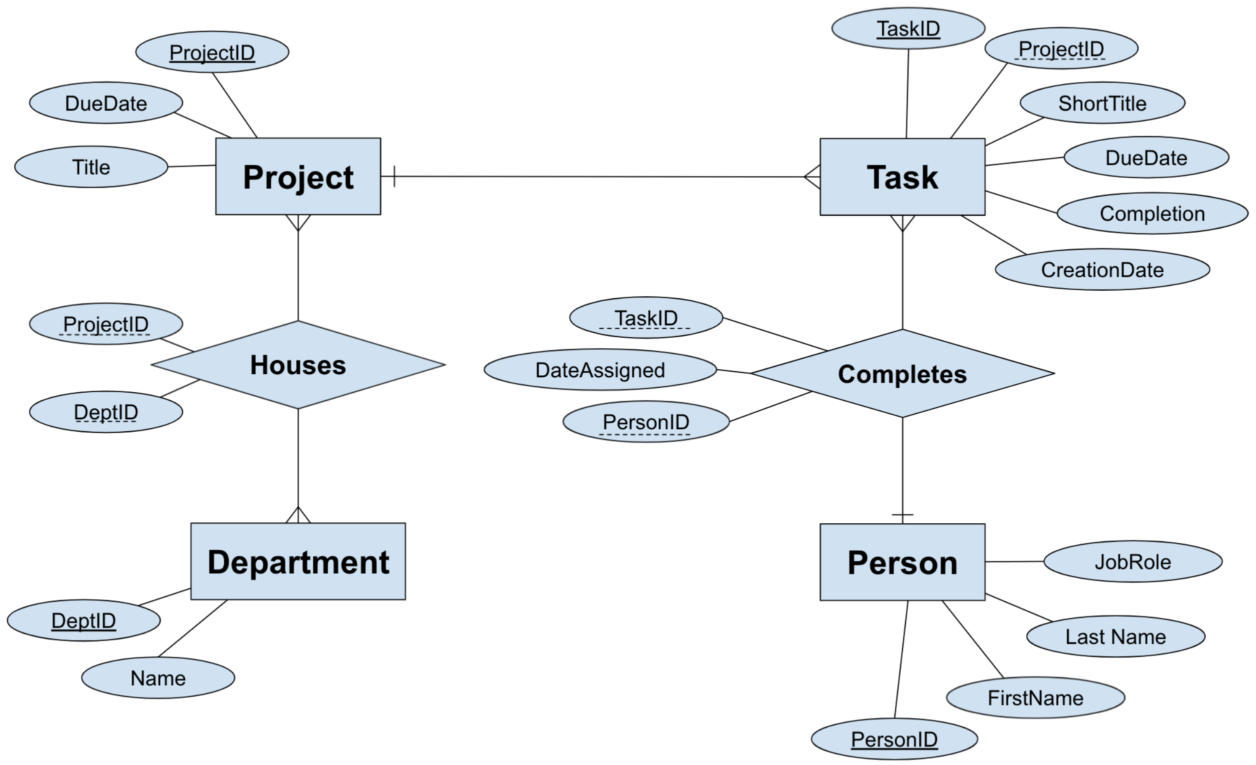
Task Organizer

Bennett Wendorf and Connor Marks

Abstract

The idea for this project comes largely from something that we, as students and software developers, are intimately familiar with. The need for most individuals and teams to have some way to manage their tasks on a given project, class, or day is something that many if not all students and developers have had to struggle with. Our goal with this project is to try to come up with a good solution for managing tasks for whatever use case our users might need. We also want to include a more overarching idea of a project that tasks can be a part of. This would be especially useful for teams that are working on multiple projects to help them organize what they need to work on and what those individual tasks are a part of. Much of the inspiration for how to format this project comes from Microsoft To Do. Microsoft is a task management software that allows management of tasks for a single user. One of our main goals is to take this idea and expand on it to allow management of tasks for multiple projects as well as teams of people. We chose to implement this as a web app so that it can conceivably be run on cloud infrastructure and be accessed from anywhere. In theory, this structure would also allow us to easily implement a mobile app as a secondary platform for accessing the same data. We wanted to provide a fully functional task management system that one could easily use to provide task and project management functionality for a single user, all the way up to a large team with a clean UI and easy access to all the data one might need.

Database Design



Project: This table will keep store the data for various projects. The idea with projects is to provide a way to organize groups of tasks so they can be more easily organized and referenced. Projects will have a projectID to uniquely identify them and will also have a title that may identify them but will be more of a description. Projects will also have a DueDate and this is when all tasks of a project should be completed by.

Task: Task management is the main focus of this program and that is why this table has the most attributes in it and is the most important. TaskID will be an AutoIncremented value and will be used to keep tasks organized. ShortTitle will give tasks a more meaningful name then just an ID number. All tasks will have a creation date and due date assigned to make sure tasks are completed in a timely manner. Tasks will also contain a boolean attribute called completion that will be marked true in our database when they are completed.

Completes: Every task could have a person assigned to it and that is the main purpose of this table. This table will keep account of what people are assigned to what tasks. It will have two foreign keys, those will be the primary keys from person and task but it will also have a date assigned attribute. This attribute will differ from the created date because not all tasks have to be assigned to a person right away. Completion date will allow us to see how long a task has been alive and date assigned will allow us to see how long someone has been working on a task.

Person: This is a straightforward table. People are what will be completing tasks and that is why this table is here. Each person will have a person ID that will be able to uniquely identify them. Each person will also have a first and last name, as well as their job title or role. There is no need to have a ton of information about the people because the main goal of this is task management so as long as each person can be uniquely identified the application will run well.

Houses: This table holds a relationship between a department and a project. It is necessary since that relationship is a many to many relationship, meaning that each department can house multiple projects and a single project might be co-managed by multiple departments. All we care about in this relationship is the department and project that are related, so we only have two foreign keys: ProjectID and DeptID.

Department: This table will hold some information about departments that projects are housed in. A project may be housed across multiple departments, and also each department might be managing or helping manage many projects, so the relationship between departments and projects is many to many. It is useful to store this information so we can see in the app what departments are doing well in terms of productivity. This entity only stores some basic information necessary to identify a department by name and a unique ID. This ID attribute will be auto incremented.

Functionality

- Project Page:
    - Filter what project is showing (GROUP BY, HAVING)
    - Users assigned to what projects (Joining 4 tables)
    - Tasks remaining in each project (Aggregate)
    - Ability to add, update, and remove tasks here as well

- Users Task Page:
    - Display Users Tasks (ORDER BY DueDate)
    - Allow for many tasks for a user, and limit the results to something like 10 or 20 per page (LIMIT and OFFSET)
    - Popup window for adding new tasks with spaces for user input for many of the attributes (Adding tuples)
    - Ability to click on an existing task to modify things like its title, due date, etc. (Updating tuples)
    - Button in this modification interface for deleting a given task (Deleting tuples)

- Management Page:
    - Find users that are free (all tasks are completed / has no tasks assigned)
        - This will likely require at least one subquery
    - Best User (Most Tasks completed) (Subquery with count and max)

Stakeholders

This web application will be multi purpose and very beneficial to a wide group of people. It could be used by an individual user that likes to manage their upcoming tasks and projects in a straightforward easy way to use. In this case there would only be one tuple in the person entity. It could also be advantageous to any business, large or small, that is interested in completing projects and tasks. A business would be able to add all of their employees (Persons) and then can start adding projects, tasks associated with each project and then managers could assign tasks to their employees, all in a very organized way. This application would be for anyone that wants to stay up to date and organized on tasks and projects they have to complete.

Technological Requirements

For this project we are building a web app using React for a front end and Node.js for the back end. Both React and Node are in javascript, so it will be fairly simple to be able to work with both. Neither of us have experience directly with these two frameworks, but we both have some experience with web design. I (Bennett) also have a friend who is knowledgeable about React and Node who is willing to help out if we get a bit stuck. Our plan at the moment is to use sqlite for the database. This makes the most sense to us since it is what we covered in class and what we are both the most familiar with. This will allow us to spend more of our time learning the UI frameworks and writing database queries, rather than trying to learn a different SQL as well. Node has a module called sqlite3 for integrating with this database, so that is our plan for the database connector. We have not looked into that connector at all yet, but since it is SQLite, I don’t think it will be terribly difficult to get into. In terms of sharing code, we have a GitHub repository set up to do this. Connor does not have much experience with git or GitHub, but I (Bennett) have been using git for a while and know my way around GitHub fairly well and am willing to help him wherever needed.