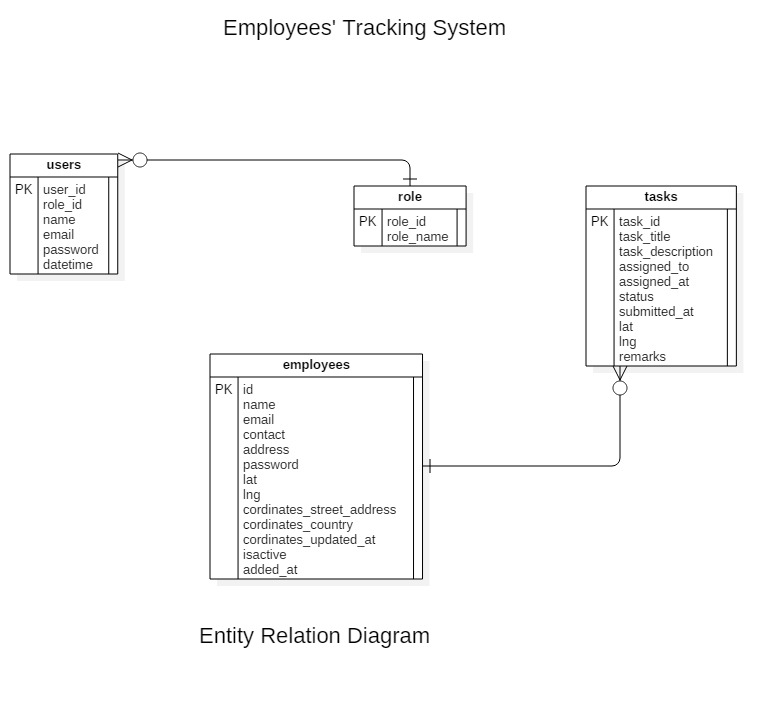
# Procedure CDP05- Databases

# Entity Relationship Diagram

After a detailed discussion with the team members and Business analyst, we have finalized 4 main entities now. Which are:

* Users
* Employees
* Tasks
* Roles

The image below shows the ER diagram of the system:



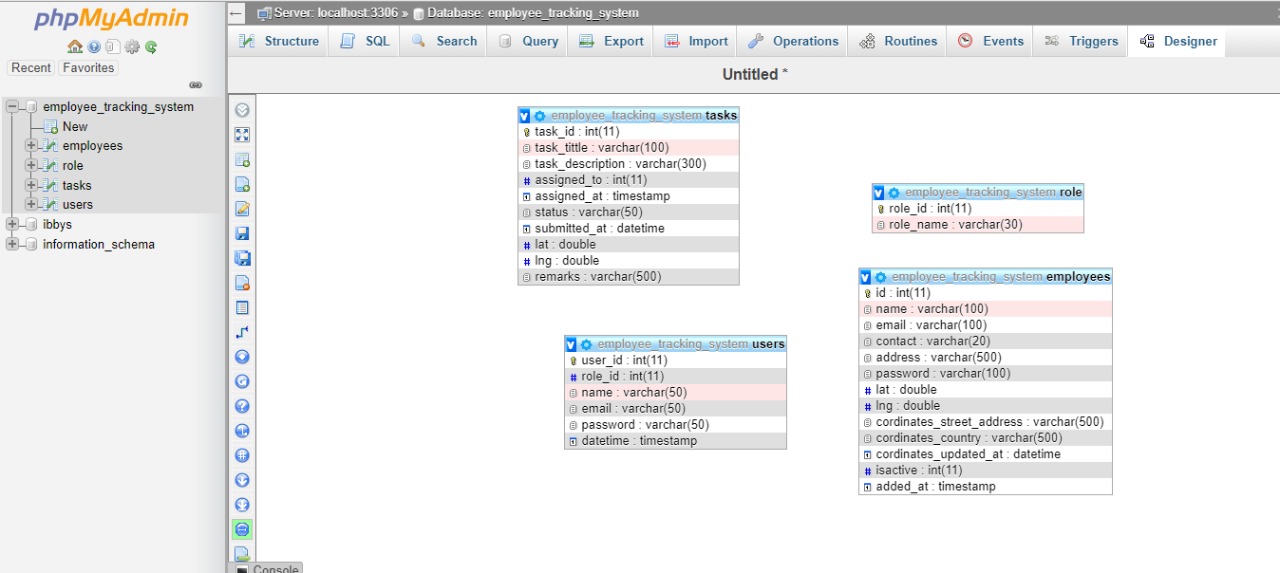
# DDL Scripts:

The conclusion of discussion with the team members was to implement the ERD in MySQL so it can be accessed by both website and mobile application. Using My SQLite for the android application will make it impossible to communicate with the website.

## Methodology:

A centralized database would be used so if the admin sends data from the website can be accessed directly by the employees in application.

# Screenshot of the Database:



# DDL Queries:

## Database Creation:

*Create database employee\_tracking\_system*

## Tables Creation:

**Users Table:**

*Create table users (*

*user\_id int NOT NULL,*

*role\_id int NOT NULL,*

*name varchar(50),*

*email varchar(50),*

*password varchar(50),*

*datetime timestamp,*

*PRIMARY KEY(user\_id)*

*FFOREIGN KEY (role\_id)*

*REFERENCES roles(role\_id)*

*)*

**Employees Table:**

*Create table employees (*

*id int NOT NULL,*

*name varchar(50),*

*email varchar(50),*

*contact varchar(50),*

*address varchar(500),*

*password varchar(50),*

*lat double,*

*lng double,*

*coordinates\_street\_address varchar(500),*

*coordinates\_country(500),*

*coordinates\_updated\_at datetime,*

*isactive int(11),*

*added\_at timestamp,*

*PRIMARY KEY(id)*

*)*

**Tasks Table:**

*Create table tasks (*

*task\_id int(11) NOT NULL,*

*task\_title varchar(100),*

*task\_description varchar(300),*

*assigned\_to int(11),*

*assigned\_at timestamp,*

*status varchar(50),*

*submitted\_at datetime,*

*lat double,*

*lng double,*

*remarks varchar (*500*),*

*PRIMARY KEY(task\_id)*

*FOREIGN KEY (assigned\_to)*

*REFRENCES employees(employee\_id)*

*)*

**Roles Table:**

*Create table roles (*

*role\_id int NOT NULL,*

*role\_name varchar(30),*

*PRIMARY KEY(role\_id)*

*)*

# Performance and Security Considerations:

Database performance improved and enhanced through many ways, some of them that we have used are:

## Optimize Queries:

In many cases, the issue of performance comes when poor queries are used for example pinging database for more times than needed, following a poor approach like selecting and searching through all the database. To enhance the database performance, we need to optimize our queries by using a single query for getting more data, by not pinging database for more times, by using where clause etc.

## Indexing SQL Query:

When indexing is done properly, it helps in optimizing the query execution duration and increase performance of the database. We actually need to implement a data structure to keep the things organized and well located. It retrieves data with more speed and saves time.

**Indexes can be created as:**

*CREATE INDEX index\_name*

*ON*

*Table\_name (col1, col2…);*

**Indexes can be dropped as:**

*ALTER TABLE table\_name*

*DROP INDEX index\_name;*

As we did not need any indexing in this system so we did not use it.

## Stronger CPU:

For higher performance, if the database is on a local memory or on any server, we need a strong CPU where the database is kept. In our case, we are keeping the database on an online server GoDaddy, which provides a very strong server to load database and database operations.

## Disk Types:

When we fetch data from a database, even a single query can require a lot of input output operations from the disk which depends on the amount of data we access and the amount of data returned. The type of disk on the server impact the SQL performance a lot. To enhance our database performance, we are using SSDs on our local disks.

## Updated SQL version:

We are using the latest version of SQL database to improve the performance of the database. Dependencies also make problems sometimes if the SQL version is not up to date.

## Ensure Normalization:

We have ensured the normalization of our database tables to improve the database efficiency. Data duplication and redundancy has been removed.

# References

1. <https://www.sentryone.com/blog/10-steps-to-better-database-performance>
2. <https://www.solarwinds.com/database-performance-analyzer/use-cases/database-performance-tuning>
3. <https://www.eversql.com/5-easy-ways-to-improve-your-database-performance/>
4. Self-research and study