**Project Time Recorder Database Project**

**Hood College, IT530, Fall 2017**

**Table of Contents**

Project Description and Scope------------------------------------------------------3

ER Design--------------------------------------------------------------------------------4

Relational Schema and SQL implementation-------------------------------------8

User Interface Design----------------------------------------------------------------14

**Project Description and Scope: Project Time Recorder**

**Description**

MMJ Project Management (MMJ), consisting of members Michael, Michelle, Junyan and Dami, must automate its project time recording processes to remain a competitor in the marketplace. The current flat file database cannot support the increased demand for MMJ services. The increasing number of projects that each employee is working on and need to know how much time is required at faster turn-around demands a better way to record time. Paper versions and multiple files on different computers do not allow for efficient and accurate analysis. Introduction of a relational database system accessed through a web-based user interface will provide the access required to support the business goals of MMJ both now and in the future.

**Scope**

The proposed system utilizes a relational database as an information repository. The database will capture both metadata and detailed information about project time management. Metadata reflects those data needed to identify the projects that need more employees and time; detailed data will describe the projects that may need to be cancelled. Detailed data will be used to make future project decisions.

The database also captures both metadata and detailed information about projects. Metadata includes employee information, project information, client information, project timetable, invoice and department information.

The user interface provides web-based access to the underlying relational database. Data entry forms provide the means to enter and edit projects employees are working on and the time they spend on each project. The user interface provides reports to identify projects, time and cost. The web-based interface will utilize PHP processes to access the database and provide web access. Client-side processing will utilize JavaScript and CSS as necessary for entry validation and visualization.

# Data Requirements Analysis

For each employee, the employee relation contains employee ID, employee name, employee title and their hourly rate, identified by employee ID.

For each dependent, the dependent relation contains name, age, gender, relationship, identified by name and employee ID. The dependent information should be deleted or updated when employee information is deleted or updated.

For each department, the department relation contains department ID and department name, identified by department ID.

For each project, the project relation contains project ID, project description and project name, identified by project ID.

For each project, there should be a time recorder that keeps track of the working hours each employee has worked on the project.

For each time, the time relation contains all employees that work on a particular project, project ID, the date, and the working hours that each employee works on a particular date.

For each client, the client relation should contain client address, client name and client ID, indentified by client ID.

Each employee must be working for at least one departments.

A department can have multiple projects.

Each employee can manage and work on multiple projects.

Each employee records their own working hours to the database in the end of each working day.

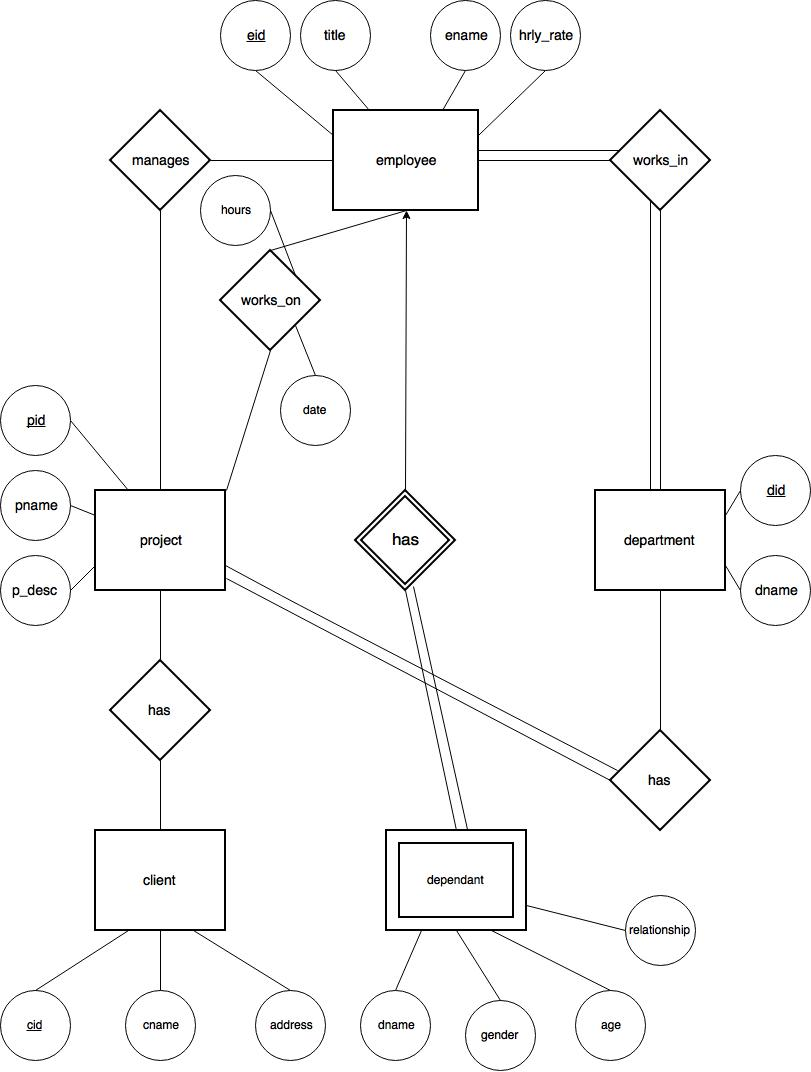
# Other Requirements Analysis

The HR department will enter the initial data of each employee and each department into the system.

The project manager will enter the initial data of their project(s) and related data needed in the time relation.

The project manager will review the data in the time relation and calculate the total hours needed for the project.

# ER Diagram



**ER Diagram Assumptions**

* A project must have many departments and a department can have many project
* A project must have time tracked
* An employee must work in many departments and a department must have many employees
* An employee can work on many projects and a project can have many employees working on it
* A project manager can manage many projects and a project can have many project managers
* A department can have many projects and a project must have many departments working on it
* A project can have many clients and a client can have many projects

**RELATIONAL SCHEMA AND EXPLANATIONS**

1.1 EMPLOYEE

Employee (eid: INTEGER, ename: CHAR(30), hourly\_rate: real, title: CHAR(15))

The employee table stores information about each employee, such as their name, hourly\_rate, and title. The employee’s ID is the primary key for the employee table. The employee table must have a department, may have an employee working on project and may have a manager of a project.

1.2 DEPENDENTS

Dependents (depend\_name: CHAR(30), age: VARCHAR(10), gender: CHAR(1), relationship: CHAR(6), eid: INTEGER).

The dependent table contains the dependent name and employee id for the dependent as a primary key. It also has their age, gender and relationship. It is used to store the dependents of employees and must have a matching employee.

1.3 DEPARTMENT

Department (did: INTEGER, dname: CHAR(30))

The department table contains the department id for the department as a primary key. It stores each department and which employees work in that department. The department table can also be used as a link to which departments have which projects. Each department must have at least one employee and each project must have a department.

1.4 WORKS\_IN

Works\_in (eid: INTEGER, did: INTEGER)

The works\_in table shows which departments the employee works in and links the employee table and department table. The primary keys are a combination of the employee id and department id.

1.5 WORKS-ON

Works\_on (eid: INTEGER, pid: INTEGER, hours: REAL, date: DATE)

The works\_on table shows which employee are working on a project. It records the hours and date that the employee spent on the project. There will only be one entry per day per employee per project. The works\_on table can be used to get the cost of the project.

1.6 MANAGES

Manages (eid: INTEGER, pid: INTEGER)

The manages table shows which manager is overseeing the project.

1.7 PROJECT

Project (pid INTEGER, pname CHAR(12), pdesc CHAR(30))

Project table stores information that are personally related to project. It contains a unique identifier, project id which serves as the primary key, project name and project description.

1.8 CLIENT

Client (cid INTEGER, cname CHAR(15), caddress STRING)

Client table contains information that are personally related to client. It contains a unique identifier, client id which is the primary key, client name and client address which will be used for billing.

1.9 PROJECT CLIENT

Project\_Client (cid INTEGER, pid INTEGER)

Project\_Client stores relationship that exist between client and project, shows the client that’s responsible for what project. The unique identifier is the combination of both primary key from Client and Project (cid and pid).

2.0 DEPARTMENT PROJECT

Dept\_Project (did INTEGER, pid INTEGER)

Dept\_Project table shows which departments oversees what project. The unique identifier is the combination of both primary key from Department and Project (did and pid).

**IMPLEMENTATION OF THE SCHEMA IN MYSQL**

use jh26db;

# Database Table Creation

# First drop any existing tables. Any errors are ignored.

DROP TABLE IF EXISTS employee;

DROP TABLE IF EXISTS dependents;

DROP TABLE IF EXISTS department;

DROP TABLE IF EXISTS works\_in;

DROP TABLE IF EXISTS works\_on;

DROP TABLE IF EXISTS manages;

DROP TABLE IF EXISTS project;

DROP TABLE IF EXISTS client;

DROP TABLE IF EXISTS project\_client;

DROP TABLE IF EXISTS dept\_project;

# Now, add each table.

create table employee(

eid INTEGER UNSIGNED,

ename VARCHAR(30),

hourly\_rate REAL,

title VARCHAR(15),

primary key (eid)

)ENGINE=InnoDB;

# eid must have a not null constraint, because the dependent must be the children of the employees who work for the company. If the employee information is deleted or updated, we want to delete all tuples in the dependents table that relate to the deleted employee information. If the employee information is updated, we also want to update its data in the dependents table.

create table dependents(

depend\_name VARCHAR(30),

age VARCHAR(10),

gender VARCHAR(1),

relationship VARCHAR(6),

eid INTEGER UNSIGNED NOT NULL,

primary key (depend\_name, eid),

foreign key(eid) references employee(eid) ON DELETE CASCADE ON UPDATE CASCADE

)ENGINE=InnoDB;

create table department(

did INTEGER UNSIGNED,

dname VARCHAR(30),

primary key (did)

)ENGINE=InnoDB;

# If the employee information is deleted, we want to delete all tuples in the works\_in table that relate to the deleted employee. If the employee information is updated, we also want to update the data in the works\_in table. If the department information is deleted, we want to delete all tuples in the works\_in table that relate to the deleted department. If the department information is updated, we also want to update the data in the works\_in table.

create table works\_in(

eid INTEGER UNSIGNED,

did INTEGER UNSIGNED,

primary key(eid, did),

foreign key(eid) references employee(eid) ON DELETE CASCADE ON UPDATE CASCADE,

foreign key(did) references department(did) ON DELETE CASCADE ON UPDATE CASCADE

)ENGINE=InnoDB;

# If the employee information is deleted, we want to delete all tuples in the works\_on table that relate to the deleted employee. If the employee information is updated, we also want to update the data in the works\_on table. If the project information is deleted, we want to delete all tuples in the works\_on table that relate to the deleted project. If the project information is updated, we also want to update the data in the works\_on table.

create table works\_on(

eid INTEGER UNSIGNED,

pid INTEGER UNSIGNED,

hours REAL,

date DATE,

primary key(eid, pid, date),

foreign key(eid) references employee(eid) ON DELETE CASCADE ON UPDATE CASCADE,

foreign key(pid) references project(pid) ON DELETE CASCADE ON UPDATE CASCADE

)ENGINE=InnoDB;

# If the employee information is deleted, we want to delete all tuples in the manages table that relate to the deleted employee. If the employee information is updated, we also want to update the data in the manages table. If the project information is deleted, we want to delete all tuples in the manages table that relate to the deleted project. If the project information is updated, we also want to update the data in the manages table.

create table manages(

eid INTEGER UNSIGNED,

pid INTEGER UNSIGNED,

primary key(eid, pid),

foreign key(eid) references employee(eid) ON DELETE CASCADE ON UPDATE CASCADE,

foreign key(pid) references project(pid) ON DELETE CASCADE ON UPDATE CASCADE

)ENGINE=InnoDB;

create table project(

pid INTEGER UNSIGNED,

pname VARCHAR(12),

pdesc VARCHAR(30),

primary key(pid)

)ENGINE=InnoDB;

create table client(

cid INTEGER UNSIGNED,

cname VARCHAR(15),

caddress VARCHAR(20),

primary key(cid)

)ENGINE=InnoDB;

# If the project information is deleted, we want to delete all tuples in the project\_client table that relate to the deleted project. If the project information is updated, we also want to update the data in the project\_client table. If the client information is deleted, we want to delete all tuples in the project\_client table that relate to the deleted client. If the client information is updated, we also want to update the data in the client table.

create table project\_client(

pid INTEGER UNSIGNED,

cid INTEGER UNSIGNED,

primary key(pid, cid),

foreign key(pid) references project(pid) ON DELETE CASCADE ON UPDATE CASCADE,

foreign key(cid) references client(cid) ON DELETE CASCADE ON UPDATE CASCADE

)ENGINE=InnoDB;

# If the department information is deleted, we want to delete all tuples in the dept\_project table that relate to the deleted department. If the department information is updated, we also want to update the data in the dept\_project table. If the project information is deleted, we want to delete all tuples in the dept\_project table that relate to the deleted project. If the project information is updated, we also want to update the data in the dept\_project table.

create table dept\_project(

did INTEGER UNSIGNED,

pid INTEGER UNSIGNED,

primary key(did, pid),

foreign key(did) references department(did) ON DELETE CASCADE ON UPDATE CASCADE,

foreign key(pid) references project(pid) ON DELETE CASCADE ON UPDATE CASCADE

)ENGINE=InnoDB;

**DATA**

Employee

1601030570,Rhoda Villarreal,96.31,Engineer

1658102615,Nadine Snider,85.91,Engineer

1621032699,Alec Gallagher,28.98,PR Officer

1686050122,Erin Houston,74.32,Customer Service Rep

1689020806,Felicia Vincent,73.45,Accountant

1673051184,Teegan Pierce,26.00,Accountant

1615051538,Cynthia Potter,20.51,HR Representative

1638072679,Heidi Owens,34.90,Manager

Department

1679099084,Accounting

1693032370,Payroll

1645010803,Customer Service

1697061210,Advertising

1608041590,Research and Development

1607071225,Sales

1654010708,Quality Assurance

1640020559,Tech Support

Dependents

George Pugh,3,M,Son,1658102615

Quincy Wilson,2,F,Daughter,1601030570

Illana Blankenship,1,F,Daughter,1686050122

Minerva Patel,2,F,Daughter,1686050122

Wyoming Melton,13,M,Son,1673051184

Castor Parks,16,M,Son,1638072679

Jennifer Parker,4,M,Son,1638072679

Giselle Kirby,3,F,Daughter,1689020806

Works\_in

1601030570,1640020559

1621032699,1645010803

1658102615,1645010803

1686050122,1679099084

1689020806,1679099084

1673051184,1679099084

1615051538,1608041590

1638072679,1693032370

Manages

1601030570,1676072322

1658102615,1624070310

1689020806,1675121349

1638072679,1663110336

1686050122,1674092116

1686050122,1629072586

1621032699,1678121708

1621032699,1665071211

Project

1676072322,Employee Handbook,Revising employee handbook

1624070310,Project Time Recorder,Records time on projects

1678121708,Secret Government,A secret government project

1675121349,Titan,A super big project

1663110336,Code Talker,Software development for outside company

1665071211,X Lab,Research and development for new aircraft

1674092116,404,Fixing website

1629072586,Hydra,Lots of heads

Client

1668112771,Mattis Institute,7869 Dolor. Ave

1674082419,Dolor Nulla Semper Limited,Ap #2-1 Tortor. Rd.

1647062250,Nunc Mauris Corporation,475-453 Per Avenue

1659021320,A Malesuada Id LLC,P.O. Box 371 8825 Egestas Avenue

1690022831,Sed Consulting,P.O. Box 319 4989 Praesent Street

1681110857,Quisque Corp.,P.O. Box 339 9725 Cras Rd.

1626053055,Quis Lectus Institute,375-54 Rutrum St.

1682050207,Posuere At Associates,Ap #570-81 Odio, Av.

Project\_client

1676072322,1674082419

1624070310,1647062250

1678121708,1659021320

1675121349,1690022831

1663110336,1681110857

1665071211,1626053055

1674092116,1668112771

1629072586,1647062250

Dept\_project

1679099084,1676072322

1693032370,1624070310

1645010803,1678121708

1645010803,1675121349

1697061210,1663110336

1608041590,1665071211

1607071225,1674092116

1654010708,1629072586

1640020559,1665071211

Works\_on

1601030570,1676072322,5,2017-05-01

1601030570,1678121708,2,2017-03-12

1621032699,1624070310,4,2017-06-20

1658102615,1624070310,10,2017-06-20

1686050122,1675121349,7,2017-06-12

1689020806,1663110336,8,2017-05-15

1673051184,1665071211,6,2017-04-30

1615051538,1665071211,9,2017-04-26

1638072679,1674092116,2,2017-06-04

1689020806,1629072586,1,2017-06-07

1638072679,1663110336,9,2017-07-06

**User Interface Design**

Home

**The home page:**

Functions as the main page to select links to our about us, login, search, add time and update time sites. It gives a brief description of the company.

**The About Us page:**

Tells why we started the company and the founders of the company.

**The Login page:**

Verifies that the employee is authorized to access the site. They will need to enter their username and password.

**The Search page:**

Allows the employee to search records in the database. They will need to enter their employee id, name, or project name.

**The Insert Record page:**

Allows the employee to enter new records into the database. They will need to enter their employee id, project id, hours and date.

**The Update Record page:**

Allows the employee to update existing records in the database. They will need to enter their employee id, project id, hours and date.

**Website Link:**

<http://pluto.hood.edu/~jh26/MMJ_Project_Time_Recorder/Index.html>