The Best-P Company Database Management

Payroll Management
and Project Management

CS631 Database Management

Wei Zhang
YuQing Ding

Email:
wz27@njit.edu
yxd1372@njit.edu

TABLE OF CONTENTS

1. Introduction

This document is to simulate The Best-P personnel company database system to implement Human Resource management and Project management applications.

2. Requirement Description

The following list is a collection of information to be recorded in Best-P Company database management system:

2.1 Database Design Requirements

a. The company is organized into division and each division has a set of departments. Each department oversees a set of projects. The department has a set of rooms in different buildings that are used as offices or conference rooms.
b. An employee works for a department, but a few employees work for the divisions with no affiliation to any department.
c. The buildings are identified by a code but have names and the company also records the year each building was built or bought and how much it cost.
d. Every department has a department head, and every division has a division head. The head a division is picked from a department and keeps his affiliation to that department.
e. An employee can work on 1 project at a time, but the company would like to keep track of all the project an employee has worked on, the time he/she has spent on each project and the role they played in the project
f. Each project has only one manager.
g. Several employees can share an office; each office has a set of phones, one for each of the employee sharing the office.
h. A preliminary analysis has listed the following possible attributes but feel free to challenge them and add more:
i. For each department: department name (unique), and budget.
j. For each employee: employee number (unique), employee name, current project, title, office number, phone number; also, title of each job the employee has held, plus date (starting
k. date) and salary for each distinct salary received in that job. For each project: project number (unique), budget, date started and date ended.
l. For each office: office number (unique), area in square feet, and numbers (unique) of all the phones in that office.

2.2 Application Development Requirements

The company Best-P would implement 2 applications: Human Resource management and Project management.

2.2.1 Human Resources and Payroll

In addition to the regular employees of the company that are all salaried, the company employs people for specific projects. These employees are paid hourly, and their employment is limited to the duration of the project. The company would like an application that help manage all the employees and pay their salaries every month. For the salaries, we will assume 10% federal tax, 5% state tax and 3% for other taxes and deductions for simplicity. The company keeps the salary and tax history for all the employees as it has to report them to IRS at the end of each year.

The payroll application will allow:

a. Generate paychecks with monthly salary and tax history for all types of employees
b. Generate cost reports for all departments,
c. Generate cost reports for all divisions
d. Generate IRS tax forms for all types of employees at the end of each year.
e. Users can download history IRS reports for the entire company.

2.2.2 Project Management

The project management application will allow:

a. Creating a project and assigning the project manager and the project team
b. Viewing some statistics on the project (e.g, number of employees working on the project, total person-hours per project).
c. The project management application will also help keep track of the project milestones with details on progress status (% done).

3. Purpose

The main purpose of this project is to provide a database system as well as a management interface to support Human Resources/Payroll and project management for Best-P personnel company. This document provides the necessary information to develop the database and the interface for Human Resources/Payroll and project management applications.

4. Software Requirements

The applications uses the following software:

Component	Requirement
Database	MySQL community-8.0.27.1
Front End	Html 4.0/CSS 2.0 Apache Tomcat 11

Back End	Java JDK 1.6+ Java Servlet JDBC 4
Windows	10

5. API Programs and Web HTML Pages

Appendix A Java Code	**AlertReminder.java** **MainPayroll. java** **MainProject.java** **ConnectionReset.java** (JDBC connections, reset, update DBs) Login.java Supportive classes to convert and calculate difference in dates, days, month, and years from SQL to java
Appendix B **Web-Interface Code**	loginPayroll.html loginProject.html MainPayroll.html MainProject.html
Appendix C SQL Source	for all function required by payroll and project APIs
Appendix D Reports	IRS report for entire company Project Status Report for entire company

6. System Design

6.1 Overview

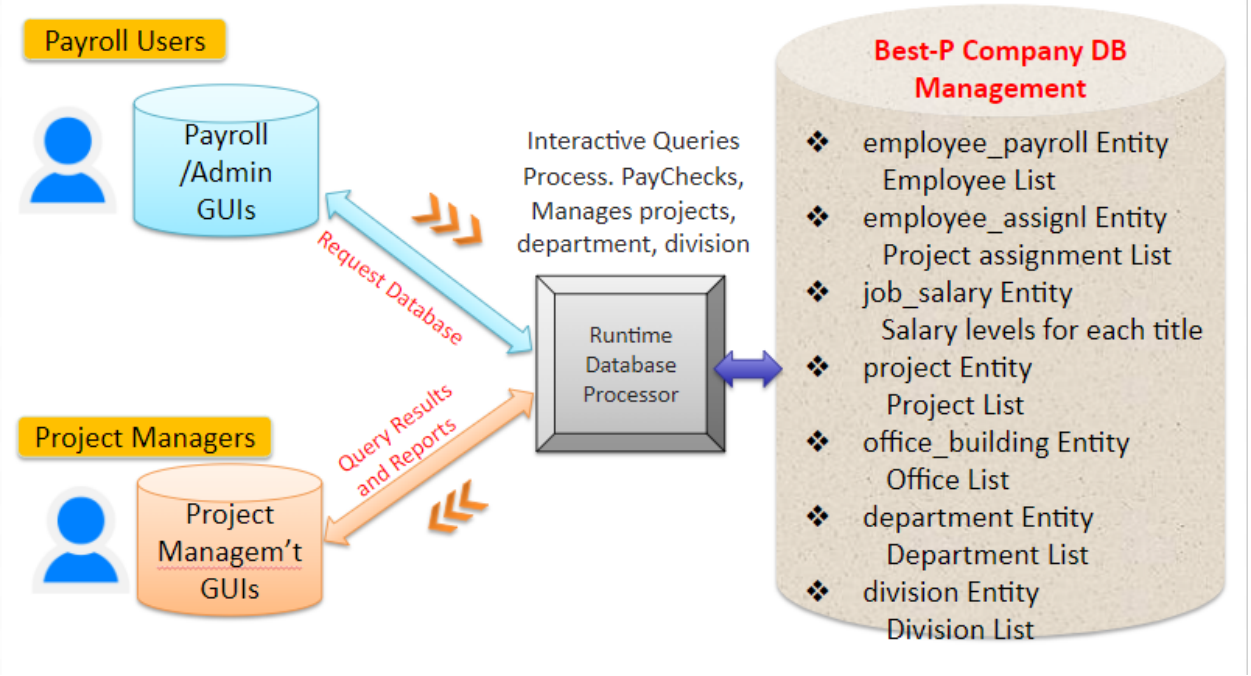
The system consists of two parts: backend and frontend. The backend defines the database that will be used to manage the HR/Payroll and project management information. The frontend is a Java API using JDBC driver connection to execute reports and SQL queries. The API has a GUI interface to manage the same operation as the JAVA API. The GUI uses web HTML and Servlet to connect to the Java API.

6.2 Database design overview

The database consists of several tables to identify each aspect required by the Best-P personnel company management team. The tables define the Employee entity, Salary level, Project, project status, Office, Department, Division. See Figure 1.

Detailed information about each table is described in the below section.

Figure 1. Best-P Database Structure



6.3 ER Diagram

The ER diagram represents the enhanced entity relations between the tables. It is used as a graphical view of the database design to quickly identify attributes and relationships between tables.

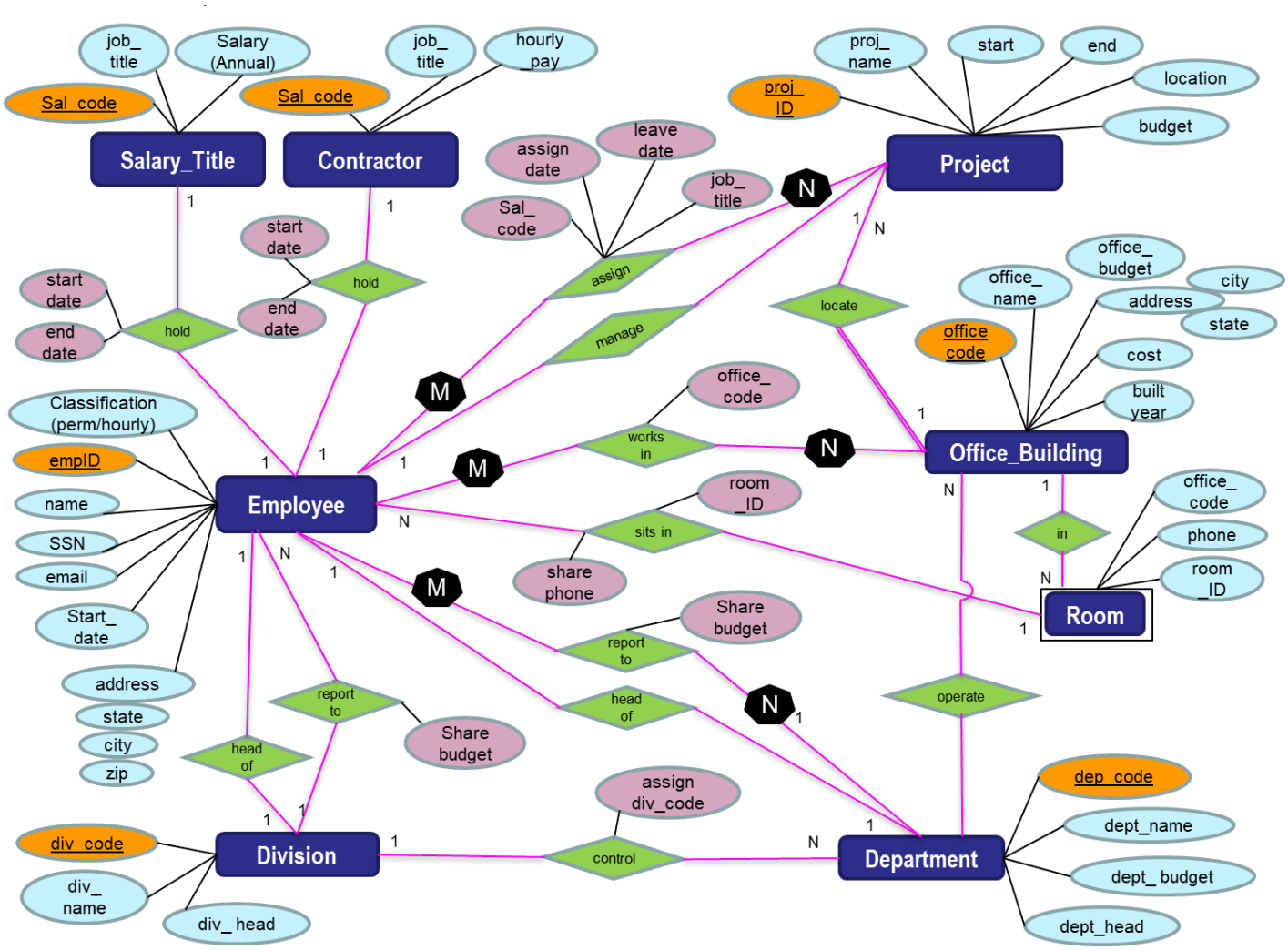


Figure 2. Best-P Database ER Diagram

6.4 ER Design and Database Normalization

Normalization is a process of concept separation which applies a top-down methodology for producing a schema by subsequent refinements and decompositions. It provides a theoretical foundation to well-known best practices in database design that state not to combine unrelated facts in one relation and that each relation should contain an independent set of facts.The normalization rules are called normal forms.
The First Normal Form aims at eliminating repeating groups in individual relations. By definition, every column is atomic.

The Second Normal Form aims at eliminating partial functional dependencies of non-prime attributes to key attributes. By definition, an entity that is 1NF and one of its attributes is defined as the primary key and the remaining attributes are dependent on the primary key.

The Third Normal Form aims at eliminating transitive functional dependencies of prime (key) attributes to keys. By definition, a table is considered in third normal if the table/entity is already in the second normal form and the columns of the table/entity are non-transitively dependent on the primary key.

The last Normal Form is called Boyce-Codd Normal Form(BCNF). By definition, the table is considered Boyce-Codd Normal Form, if it's already in the Third Normal Form and for every functional dependency between A and B, A should be a super key.

Best-P Company database shall satisfy the 3rd normal form. All the relations have semantic unity and all the attributes are not null. Data integrity has been achieved.

6.4.1 Five-Step ER Design

Step 1: Handling/Major Entities:

Employee (empID, name, title, starting _date, address, SSN, email, classification)
Salary (sal_code, job_title, salary(annual based)
Contractor: (sal_code, job_title, hourly_pay)
Project (projID, proj_name, budget, start, end)
Office_Building (office_code, office_name, address, city, state, area_sqrt, built_year, cost)
Room (RoomID, office_code, phone)
Department (dept_code, dept_name, dept_head, dept_budget)
Division (div_code, div_name, div_head)

Step 2: Weak entity:

Room is weak entity for the Best-P Company database since it's inside an office building and depending on existence of a building.

Step 3: 1:1 relationships

Relationship between Employee and Project: MANAGE(1:1). Each project is managed by one project manager.

Relationship between Employee and Division: HEAD OF (1:1). Each division has a division head, who oversees all the work in each division.

Relationship between Employee and Department: HEAD OF (1:1). Every department has a department head who supervises all the work in each department.

All other attributes are depending on the primary key in both entities, NO partial or transitive relationships in both entities.

Step 4: 1:N relationships

Relationship between Employee and Salary / Contractor: HAS(N:1). Each employee has a matching salary level depending on their title. Several employees may have held the same title.

Relationship between Employee and Office_building WORKS IN (N:1). Several employees(N) work in one office.

Relationship between Employee and Rooms: SITS IN IN (N:1). Several employees(N) work in one room in one office. Each office has a set of phones, one for each of the employees sharing the rooms and phones.

Relationship between Division and Department: CONTROL(1:N). Each division has a set of departments(N).

Relationship between Project and Office_building: LOCATE(N:1). Each department oversees a set of projects. The department has a set of rooms in different buildings that are used as offices. Therefore, A set of projects(N) locate in one office location.

Relationship between Employee and Salary/Contractor: HOLD(1:N). Each employee holds one or more job positions in the company. In other words, an employee will be assigned to different projects with different job titles over the time (for example, an employee may get promoted from junior to senior over different projects).

Relationship between Department and Office_building: HAS(1:N). Each department is operating different Office buildings(N).

All other attributes are depending on the primary key in both entities, NO partial or transitive relationships in both entities.

Step 5: M:N relationships

Relationship between Employee and Project: ASSIGN(M:N). Although each employee has been assigned to one project at a given time, one employee can work on different projects over the years and each project has a number of employees working for a specific period of time (permanent employees will work longer than contractors). So the employee and project are many to many relationships.

Relationship between Employee and Department: REPORT TO (M:N). Multiple employees work in one department and report to one department head. One employee can be assigned to different projects in different departments over the time. So the employee and department are many to many relationships.

Relationship between Employee and Office_building: WORKS IN(M:N). Multiple employees work in one office. One employee can be assigned to different offices in different projects over the time. So the employee and office are many to many relationships.

6.4.2 Final set of relations

Figure 3. Best-P Database ER Relationships

Relationship Name	Relationship Type	Between Two Entities
HAS	1: N relationship	Salary/Contractor& Employee
HOLD	1: N relationship	Employee & Salary/Contractor
ASSIGN	M: N relationship	Employee & Project
WORKS IN	M: N relationship	Employee & Office_building
REPORT TO	M: N relationship	Employee & Department
LOCATE	N: 1 relationship	Project & Office_building
CONTROL	1: N relationship	Division & Department
MANAGE	1: 1 relationship	Employee & Project
HEAD OF	1: 1 relationship	Employee & Department
HEAD OF	1: 1 relationship	Employee & Division
OPERATE	1: N relationship	Department & Office_building
IN	1: N relationship	Office_building & Rooms
SITS IN	N: 1 relationship	Employee & Rooms

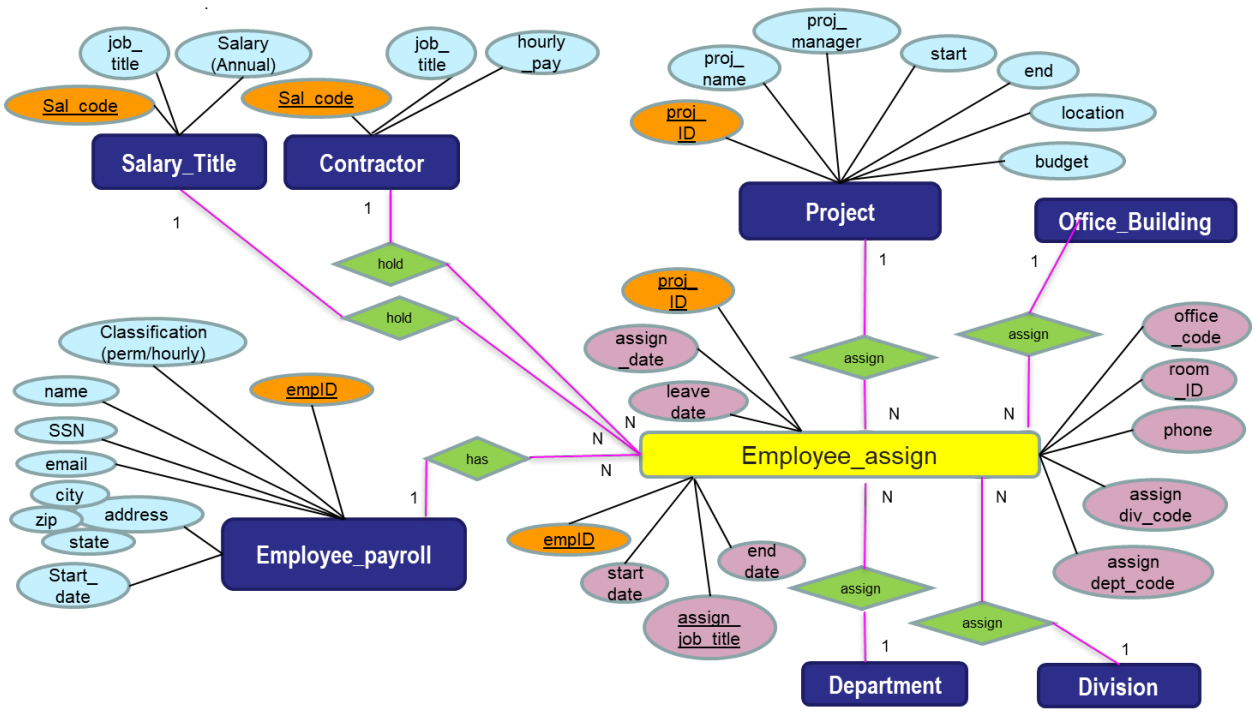
6.5 Decompose M:N Relationships in ER Design

The two M:N relationships shall be decomposed into 1:N relationships to conform with the 3rd Norm Form (otherwise, SQL queries will not be able to produce accurate results). This is done by matching the two entities using the relationships shown in the ER mapping table. A new table is created to account for all the assignments of each employee over the time, including his job_titiles, projects, along with start and end times of that assignment and associated locations, departments, etc.

The primary key of the new table Employee_assign table is combination of empID, projID, job_title, start_date and end_date to ensure it's a unique tuple (otherwise, SQL queries will fail).

Note that all the attributes in the Employee_assign table are derived from various relationships to break out the M:N relationships as shown in the new diagram below.

Figure 4. Tables Derived from ER



6.6 Final Database Tables by ER Design

Figure 5. Best-P Database Tables

Entities (Primary Key & Major Attributes & Foreign Keys)

- Employee:
  - Employee_payroll (empID, name,SSN, start_date, address, city, state, zip, email, classification of permanent or hourly contractor)
  - Employee_assign (empID, job_title, proj_ID, proj_startdate, proj_enddate, office_code, dept_code, div_code, proj_code)
- Salary_title
  (sal_code, job_title, salary(annual))
- Contractor
  (sal_code, job_title, hourly_pay)
- Project
  (projID, proj_name, proj_manager, location, budget, start, end, status, milestone)
- Office_building
  (office_code, office_name, room_ID, phone, address, city, state, area_sqrt, built_year, cos, budget)
- Department
  (dept_code, dept_name, dept_head, dept_budget, div_code, office_code)
- Division
  (div_code, div_name, div_head)

Employee_payroll:

The table attributes are very basic information about one employee, such as name, SSN, start_date, address, city, state, zip, email, classification of permeant or hourly contractor. Primary key for this table is empID,

SPECIAL NOTE: the classification attribute will specify hourly and permanent employees, since the benefit cost and IRS report are entirely different.

Employee_assign:

The table attributes, including office_code, dept_code, div_code and proj_code, etc. are relationships to other entities to achieve Boyce-Codd Normal Form(BCNF). The primary key for this table is a combination of empID, projID, job_title, office_code, dept_code, div_code along with the assigned project start and end times (we assume the title assigned time is the same as project assigned time). This is to ensure no attributes have NO partial or transitive relationships.

SPECIAL NOTE: A VP or secretary working on overhead doesn't have any project or department assignment.

Salary (for permanent):

The table is salary levels to match the title of each permanent employee. The primary key for this table is sal_code. All other attributes are depending on the primary key, NO partial or transitive relationships in this entity.

Contractor (for hourly):

The table is salary levels to match the title of each hourly employee. The primary key for this table is sal_code. All other attributes are depending on the primary key, NO partial or transitive relationships in this entity.

Project:

The table is project available to charge by employee or contractor. The primary key for this table is projID. All other attributes are depending on the primary key, NO partial or transitive relationships in this entity.

Office_building:

The table is office_building for each employee to work in. The primary key for this table is office_code. All other attributes are depending on the primary key, NO partial or transitive relationships in this entity.

Room:

The table is depending on table of office_building. Only a list of rooms, phones for each office.

Department:

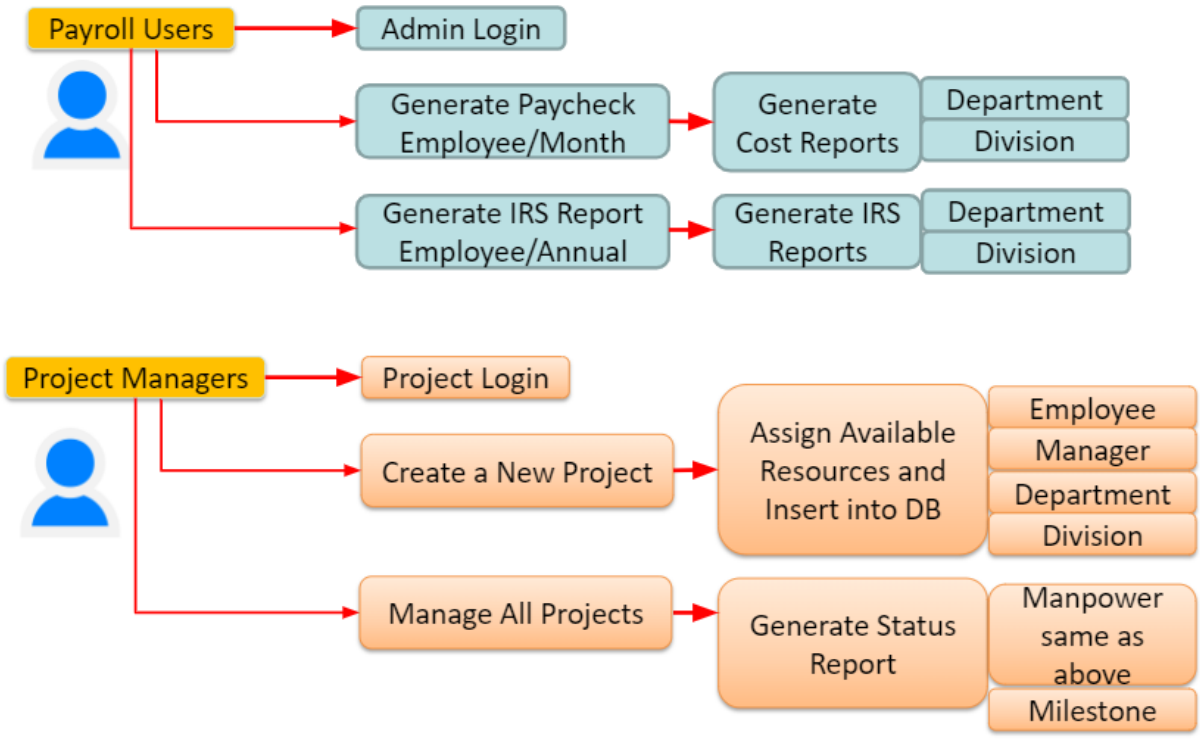
The table is department assigned to each employee. The primary key for this table is dept_code. All other attributes are depending on the primary key, NO partial or transitive relationships in this entity.

Division:

The table is division assigned to each department. The primary key for this table is div_code. All other attributes are depending on the primary key, NO partial or transitive relationships in this entity.

6.7 Database Queries & Data Flows

Figure 6. Best-P Database User Queries & Data Flows



6.8 Java Program Interface Design Overview

The Java program provides the payroll users and project manager with interfaces to manage all the project, employee database and view reports. The program authenticates the user before accessing the database.

The UI will provide the company employees an interface to view reports, search the database, and perform queries. The UI uses JSP(Java Servlet Pages) to query the database. The interface authenticates the users before accessing the database. The code is available in the Appendix A.

7. Database Generation

7.1 Create Tables

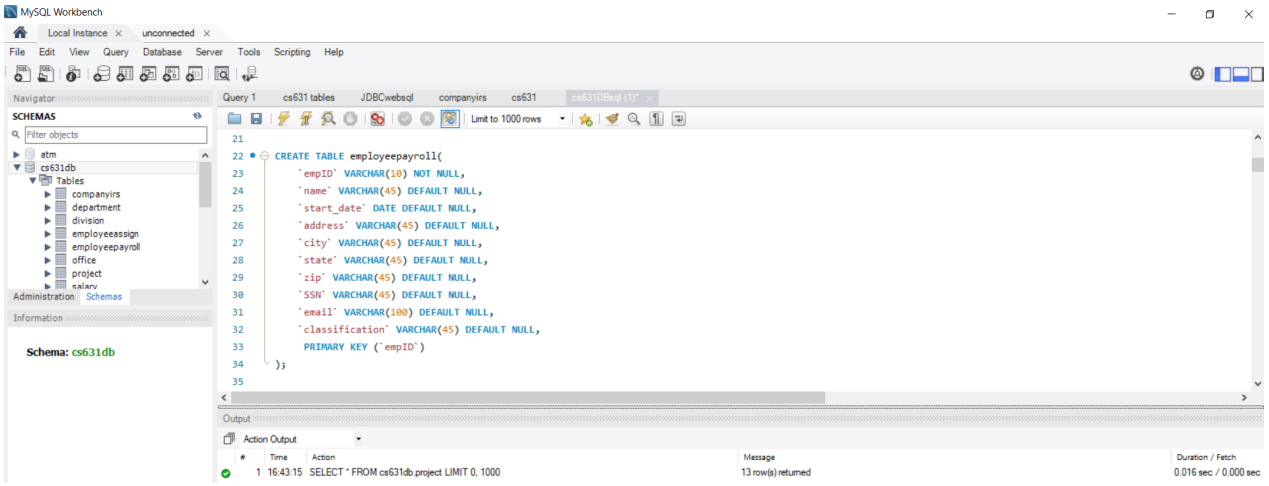
This section provides the SQL queries required to create The Best-P company database tables.

1) Create Table employee_payroll

First, create employee_payroll table. The employee_payroll table includes the empID, name, title, address, city, SSN, office_code, classification, etc. See the SQL statement below.

Appendix C SQL Source (completed and tested)

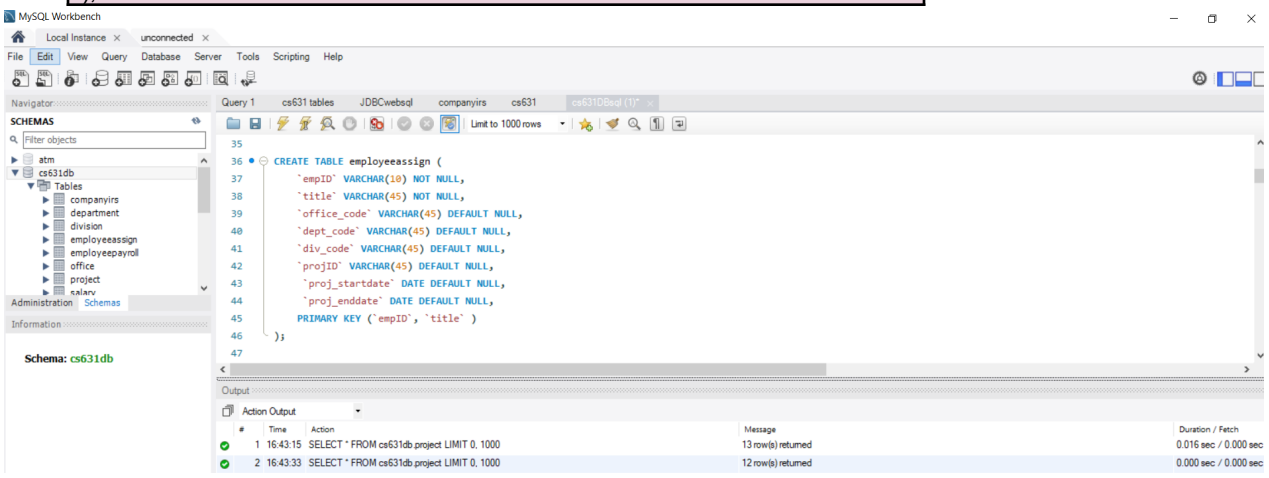
```
CREATE TABLE employeepayroll(
    `empID` VARCHAR(10) NOT NULL,
    `name` VARCHAR(45) DEFAULT NULL,
    `start_date` DATE DEFAULT NULL,
    `address` VARCHAR(45) DEFAULT NULL,
    `city` VARCHAR(45) DEFAULT NULL,
    `state` VARCHAR(45) DEFAULT NULL,
    `zip` VARCHAR(45) DEFAULT NULL,
    `SSN` VARCHAR(45) DEFAULT NULL,
    `email` VARCHAR(100) DEFAULT NULL,
    `classification` VARCHAR(45) DEFAULT NULL,
     PRIMARY KEY (`empID`)
);
```



2) Create table employee_assign

Second, create employee_assign table. The employee_assign table includes the empID, job_title,projID, proj_startdate, etc. See the SQL statement below. Appendix C SQL Source (completed and tested)

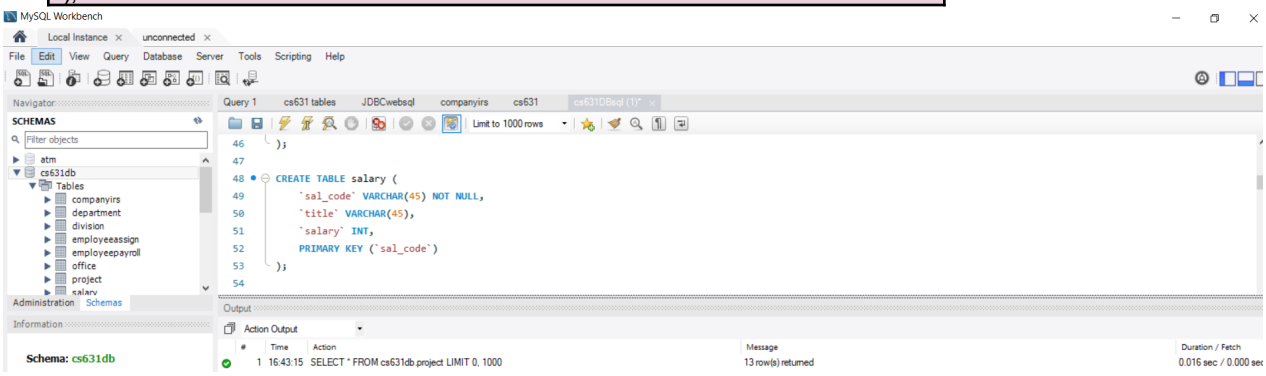
```
CREATE TABLE employeeassign (
    `empID` VARCHAR(10) NOT NULL,
    `title` VARCHAR(45) NOT NULL,
    `office_code` VARCHAR(45) DEFAULT NULL,
    `dept_code` VARCHAR(45) DEFAULT NULL,
    `div_code` VARCHAR(45) DEFAULT NULL,
    `projID` VARCHAR(45) DEFAULT NULL,
     `proj_startdate` DATE DEFAULT NULL,
     `proj_enddate` DATE DEFAULT NULL,
 PRIMARY KEY (`empID`, `title` )
);
```



3) Create Table salary

This table includes sal_code, title, salary, classification, and hourly_pay. See the SQL statement below.

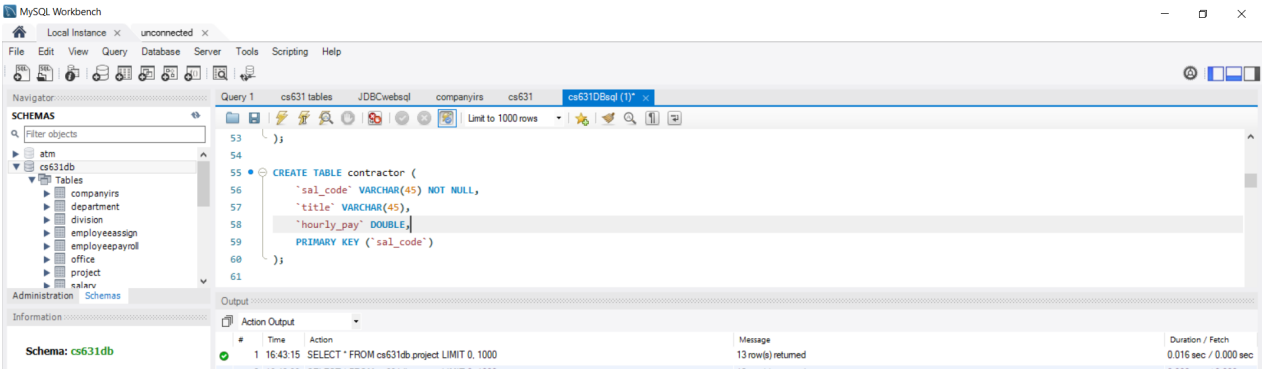
```
CREATE TABLE salary (
`sal_code`  VARCHAR (45) NOT NULL,
`title` VARCHAR (45),
`salary` INT,
primary key (`sal_code`)
);
```



4) Create Table contractor

This table includes sal_code, title, and hourly_pay. See the SQL statement below.

```
CREATE TABLE contractor (
    `sal_code` VARCHAR(45) NOT NULL,
    `title` VARCHAR(45),
    `hourly_pay` DOUBLE,
    PRIMARY KEY (`sal_code`)
);
```

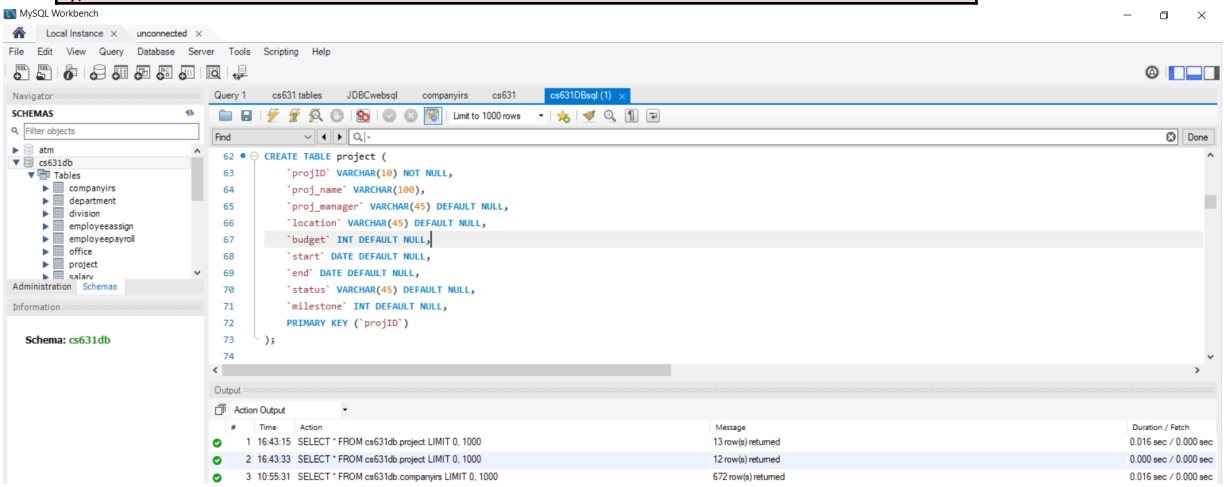


5) Create Table project

This table includes the projID, name, manager, location, budget, start/end date, status and milestone. See the SQL statement below.

```
CREATE TABLE project (
    `projID` VARCHAR(10) NOT NULL,
    `proj_name` VARCHAR(100),
    `proj_manager` VARCHAR(45) DEFAULT NULL,
    `location` VARCHAR(45) DEFAULT NULL,
    `budget` INT DEFAULT NULL,
    `start` DATE DEFAULT NULL,
    `end` DATE DEFAULT NULL,
    `status` VARCHAR(45) DEFAULT NULL,
    `milestone` INT DEFAULT NULL,
```

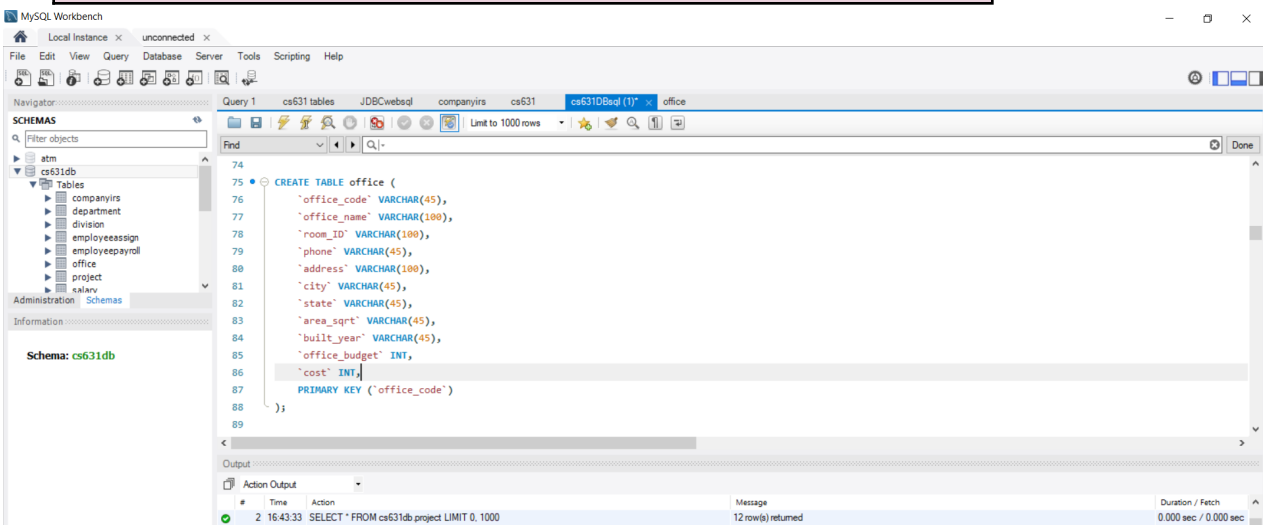
```
    PRIMARY KEY (`projID`)
);
```



6) Create Table office

This table includes the office_code, office_name, phone, address, area_sqrt, office_budget, cost and room_ID. See the SQL statement below.

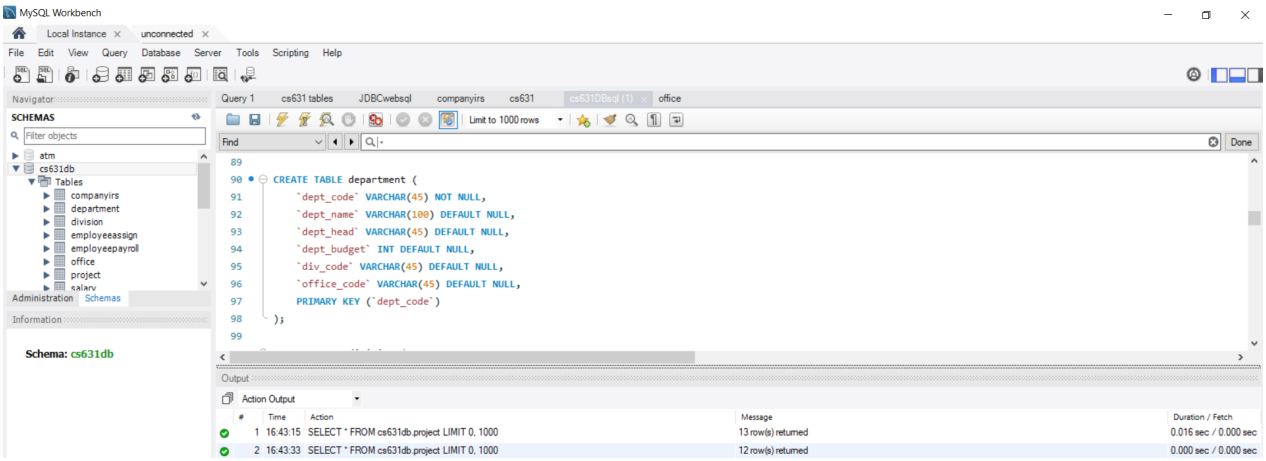
```
CREATE TABLE office (
    `office_code` VARCHAR(45),
    `office_name` VARCHAR(100),
    `room_ID` VARCHAR(100),
    `phone` VARCHAR(45),
    `address` VARCHAR(100),
    `city` VARCHAR(45),
    `state` VARCHAR(45),
    `area_sqrt` VARCHAR(45),
    `built_year` VARCHAR(45),
    `office_budget` INT,
    `cost` INT,
    PRIMARY KEY (`office_code`)
);
```



7) Create Table department

This table includes the dept_code, dept_name, dept_head, dept_budget, div_code, etc. See the SQL statement below.

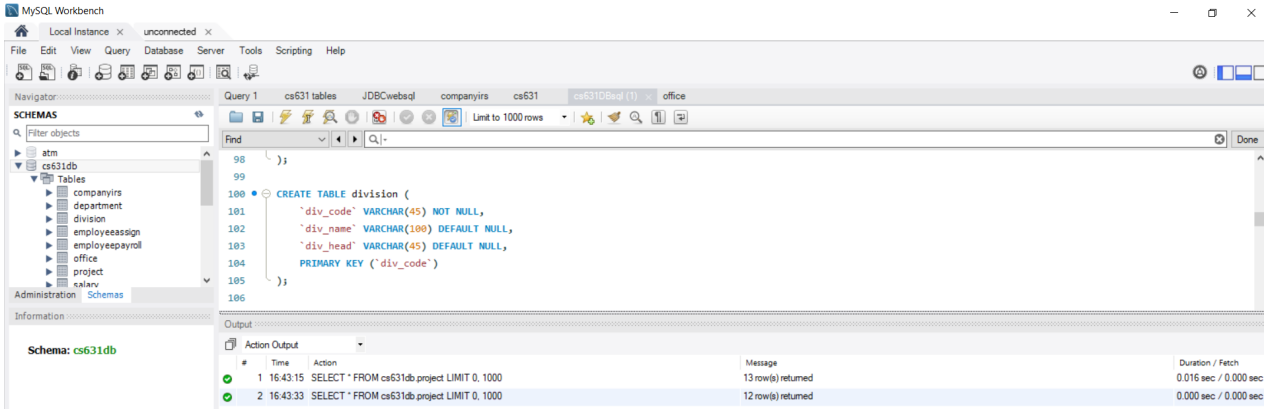
```
CREATE TABLE department (
    `dept_code` VARCHAR(45) NOT NULL,
    `dept_name` VARCHAR(100) DEFAULT NULL,
    `dept_head` VARCHAR(45) DEFAULT NULL,
    `dept_budget` INT DEFAULT NULL,
    `div_code` VARCHAR(45) DEFAULT NULL,
    `office_code` VARCHAR(45) DEFAULT NULL,
    PRIMARY KEY (`dept_code`)
);
```



8) Create Table division

This table includes the div_code, div_name and div_head. See the SQL statement below.

```
CREATE TABLE division (
    `div_code` VARCHAR(45) NOT NULL,
    `div_name` VARCHAR(100) DEFAULT NULL,
    `div_head` VARCHAR(45) DEFAULT NULL,
    PRIMARY KEY (`div_code`)
);
```

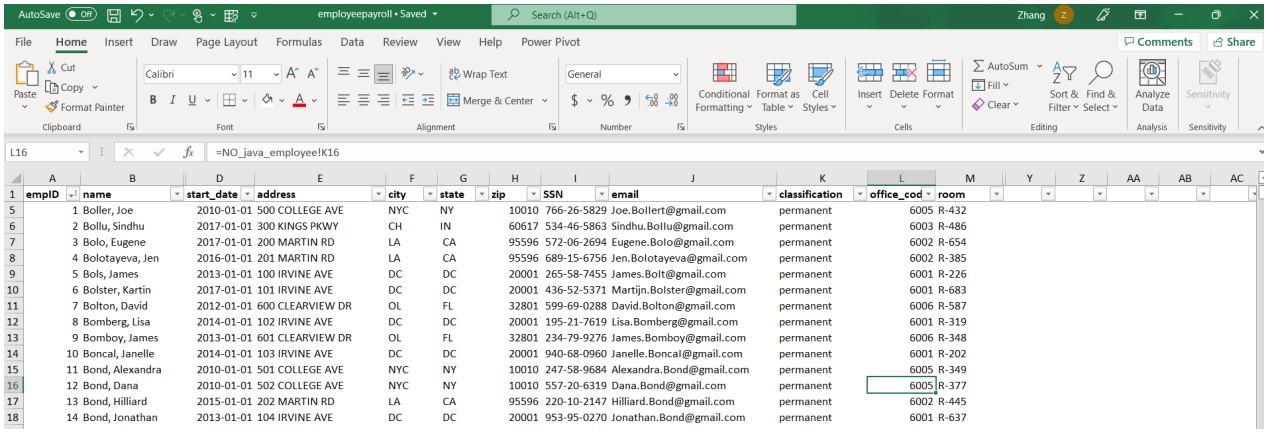


7.2. Import Data into Tables

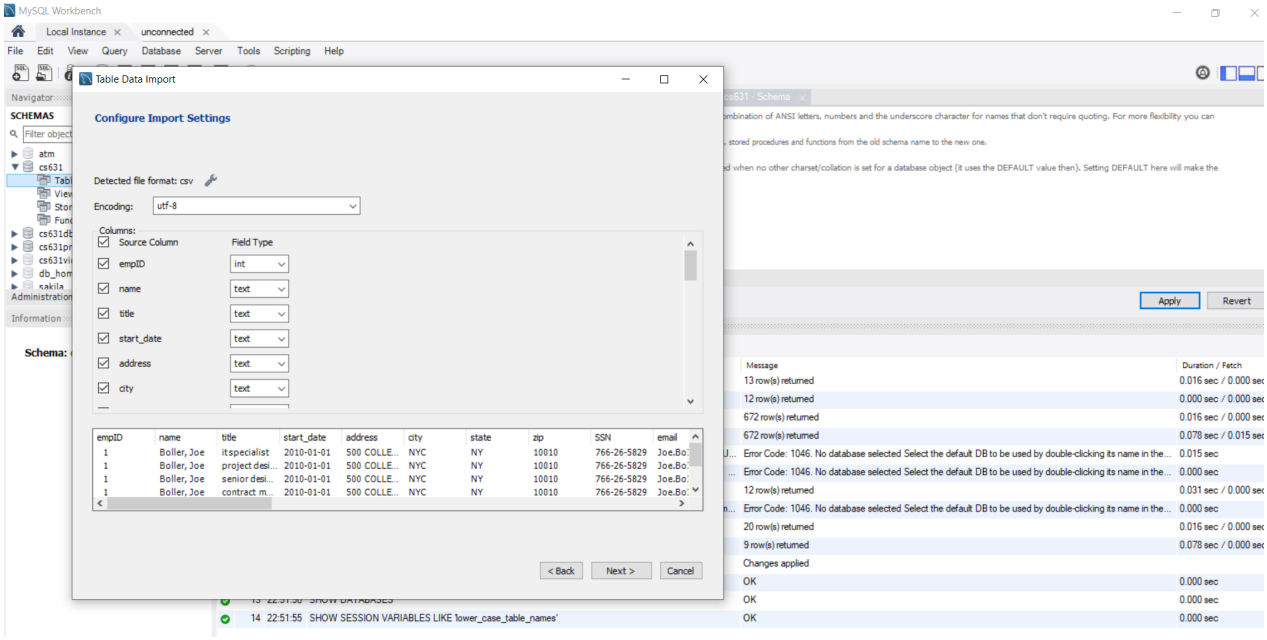
This section provides the SQL commands to import necessary and made-up data into all tables. The order of importing data into each table is important because some tables require data to exist in another table first.

1) Import data into table employee_payroll

First add all the employee_payroll data into Excel and save it as a csv file. Then import the csv file into mySQL using the Table Data Import Wizard option. See the screenshots below.

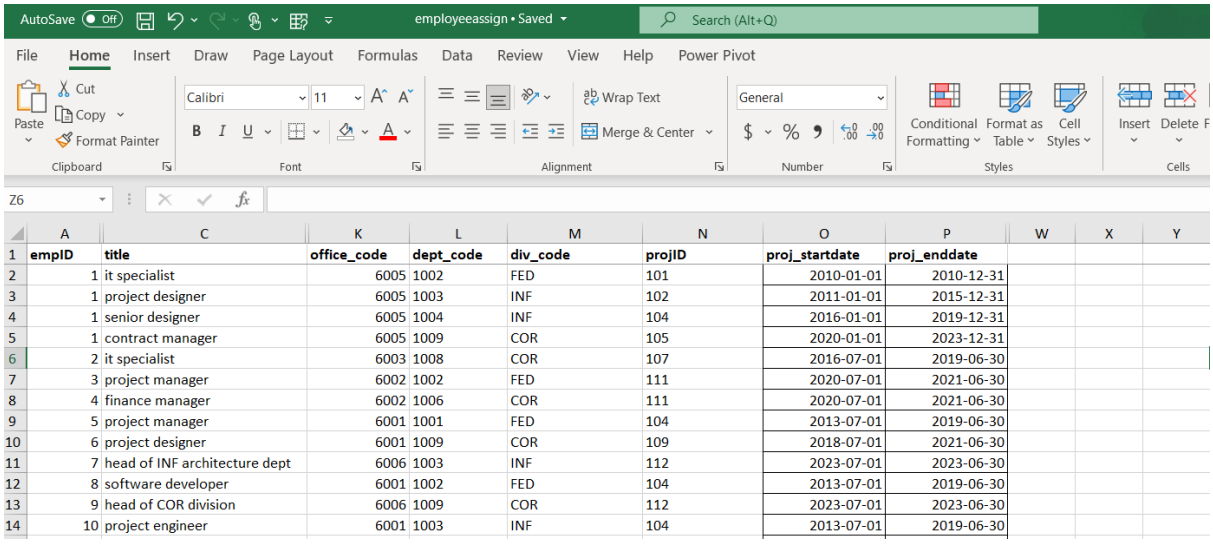


empID	name	start_date	address	city	state	zip	SSN	email	classification	office_cod	room
1	Boller, Joe	2010-01-01	500 COLLEGE AVE	NYC	NY	10010	766-26-5829	Joe.Bollert@gmail.com	permanent	6005	R-432
2	Bollu, Sindhu	2017-01-01	300 KINGS PKWY	CH	IN	60617	534-46-5863	Sindhu.Bollu@gmail.com	permanent	6003	R-486
3	Bolo, Eugene	2017-01-01	200 MARTIN RD	LA	CA	95596	572-06-2694	Eugene.Bolo@gmail.com	permanent	6002	R-654
4	Bolotayeva, Jen	2016-01-01	201 MARTIN RD	LA	CA	95596	689-15-6756	Jen.Bolotayeva@gmail.com	permanent	6002	R-385
5	Bols, James	2013-01-01	100 IRVINE AVE	DC	DC	20001	265-58-7455	James.Bolt@gmail.com	permanent	6001	R-226
6	Bolster, Kartin	2017-01-01	101 IRVINE AVE	DC	DC	20001	436-52-5371	Martijn.Bolster@gmail.com	permanent	6001	R-683
7	Bolton, David	2012-01-01	600 CLEARVIEW DR	OL	FL	32801	599-69-0288	David.Bolton@gmail.com	permanent	6006	R-587
8	Bomberg, Lisa	2014-01-01	102 IRVINE AVE	DC	DC	20001	195-21-7619	Lisa.Bomberg@gmail.com	permanent	6001	R-319
9	Bomboy, James	2013-01-01	601 CLEARVIEW DR	OL	FL	32801	234-79-9276	James.Bomboy@gmail.com	permanent	6006	R-348
10	Boncal, Janelle	2014-01-01	103 IRVINE AVE	DC	DC	20001	940-68-0960	Janelle.Boncal@gmail.com	permanent	6001	R-202
11	Bond, Alexandra	2010-01-01	501 COLLEGE AVE	NYC	NY	10010	247-58-9684	Alexandra.Bond@gmail.com	permanent	6005	R-349
12	Bond, Dana	2010-01-01	502 COLLEGE AVE	NYC	NY	10010	557-20-6319	Dana.Bond@gmail.com	permanent	6005	R-377
13	Bond, Hilliard	2015-01-01	202 MARTIN RD	LA	CA	95596	220-10-2147	Hilliard.Bond@gmail.com	permanent	6002	R-445
14	Bond, Jonathan	2013-01-01	104 IRVINE AVE	DC	DC	20001	953-95-0270	Jonathan.Bond@gmail.com	permanent	6001	R-637

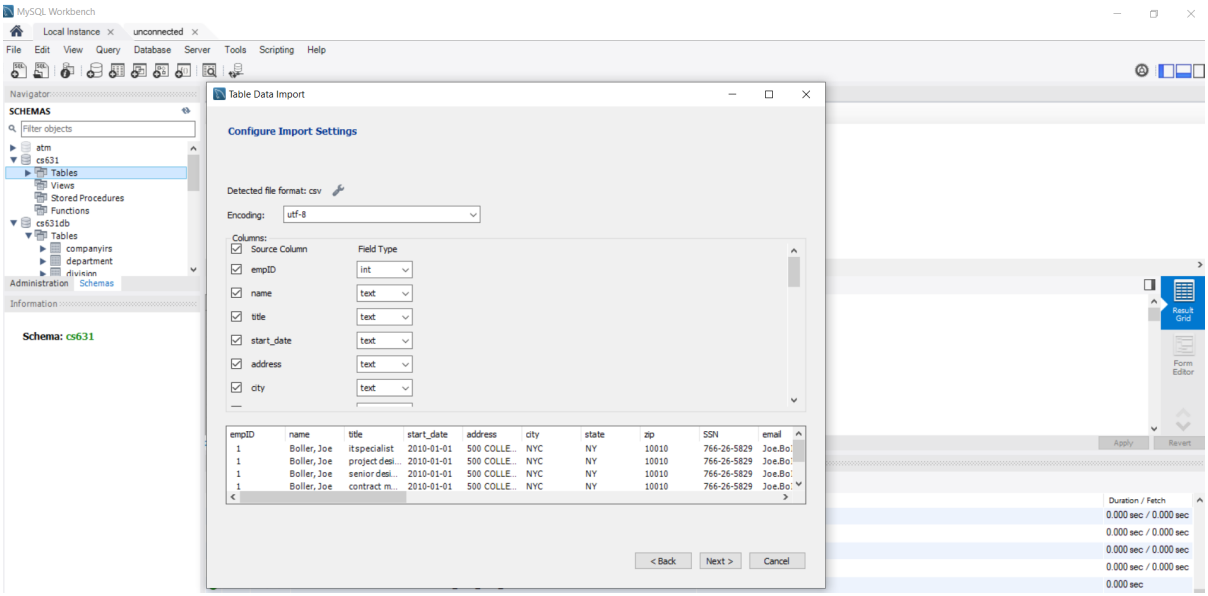


2) Import data into table employee_assign

Add data into employee_assign csv file and import it into mySQL using the Table Data Import Wizard option. See the screenshots below.

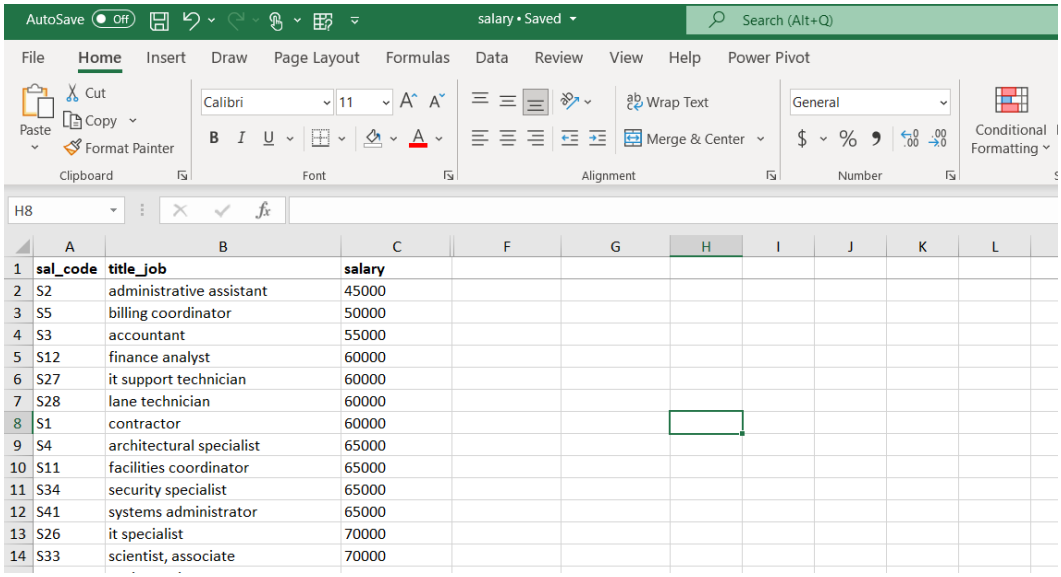


	A	C	K	L	M	N	O	P	W	X	Y
1	empID	title	office_code	dept_code	div_code	projID	proj_startdate	proj_enddate			
2	1	it specialist	6005	1002	FED	101	2010-01-01	2010-12-31			
3	1	project designer	6005	1003	INF	102	2011-01-01	2015-12-31			
4	1	senior designer	6005	1004	INF	104	2016-01-01	2019-12-31			
5	1	contract manager	6005	1009	COR	105	2020-01-01	2023-12-31			
6	2	it specialist	6003	1008	COR	107	2016-07-01	2019-06-30			
7	3	project manager	6002	1002	FED	111	2020-07-01	2021-06-30			
8	4	finance manager	6002	1006	COR	111	2020-07-01	2021-06-30			
9	5	project manager	6001	1001	FED	104	2013-07-01	2019-06-30			
10	6	project designer	6001	1009	COR	109	2018-07-01	2021-06-30			
11	7	head of INF architecture dept	6006	1003	INF	112	2023-07-01	2023-06-30			
12	8	software developer	6001	1002	FED	104	2013-07-01	2019-06-30			
13	9	head of COR division	6006	1009	COR	112	2023-07-01	2023-06-30			
14	10	project engineer	6001	1003	INF	104	2013-07-01	2019-06-30			

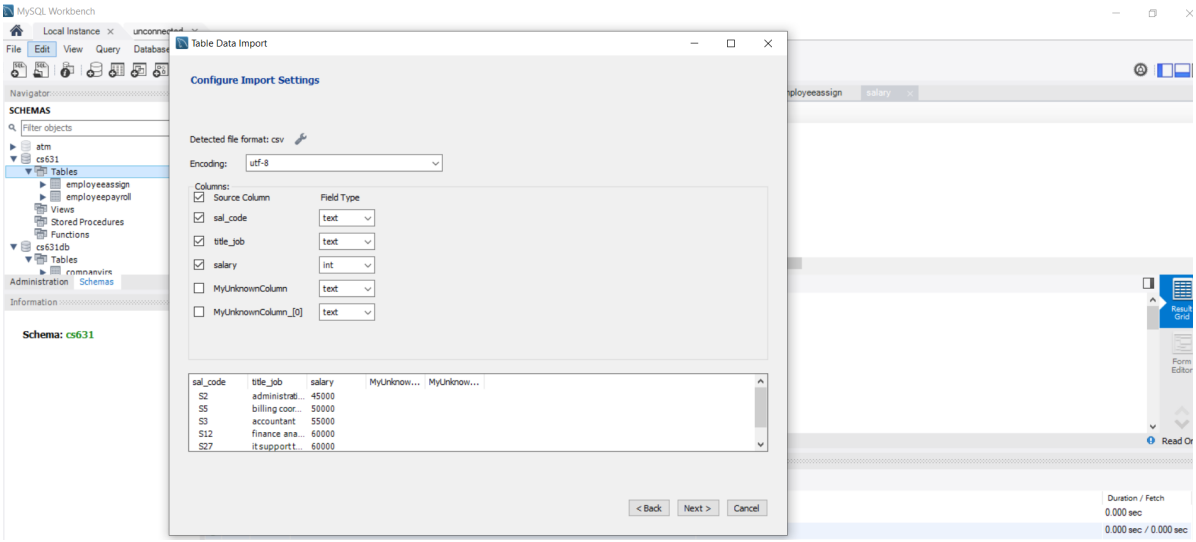


3) Import data into table salary

Add data into salary csv file and import it into mySQL using the Table Data Import Wizard option. See the screenshots below.

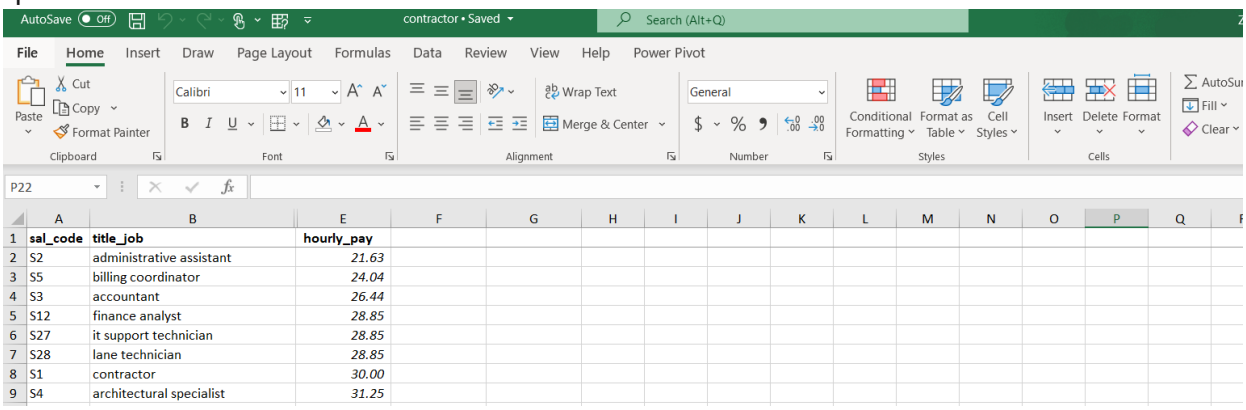


	A	B	C	F	G	H	I	J	K	L
1	**sal_code**	**title_job**	**salary**							
2	S2	administrative assistant	45000							
3	S5	billing coordinator	50000							
4	S3	accountant	55000							
5	S12	finance analyst	60000							
6	S27	it support technician	60000							
7	S28	lane technician	60000							
8	S1	contractor	60000							
9	S4	architectural specialist	65000							
10	S11	facilities coordinator	65000							
11	S34	security specialist	65000							
12	S41	systems administrator	65000							
13	S26	it specialist	70000							
14	S33	scientist, associate	70000							

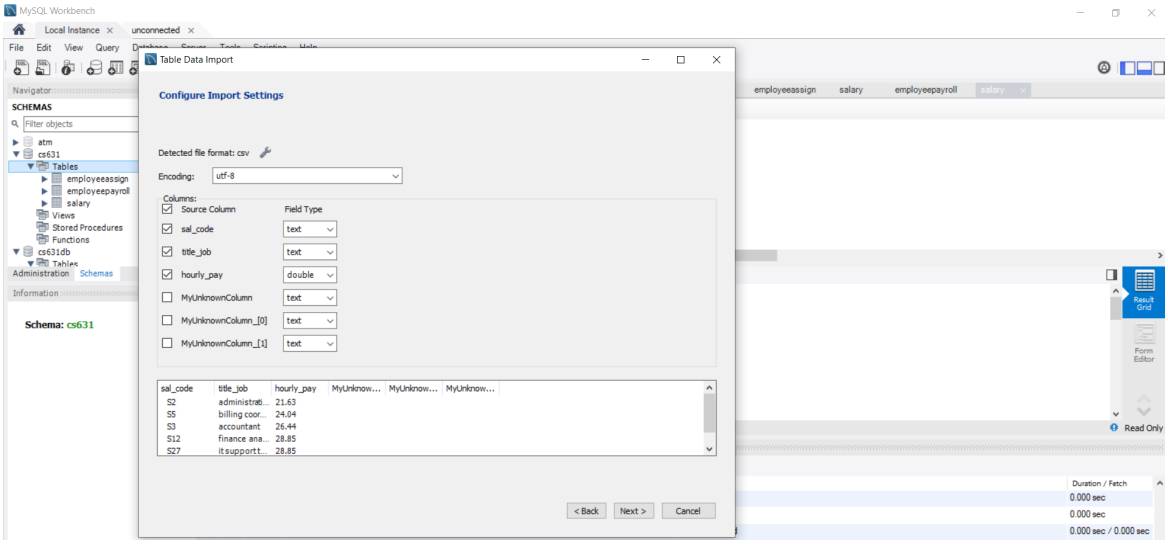


4) Import data into table contractor

Add data into contractor csv file and import it into mySQL using the Table Data Import Wizard option. See the screenshots below.

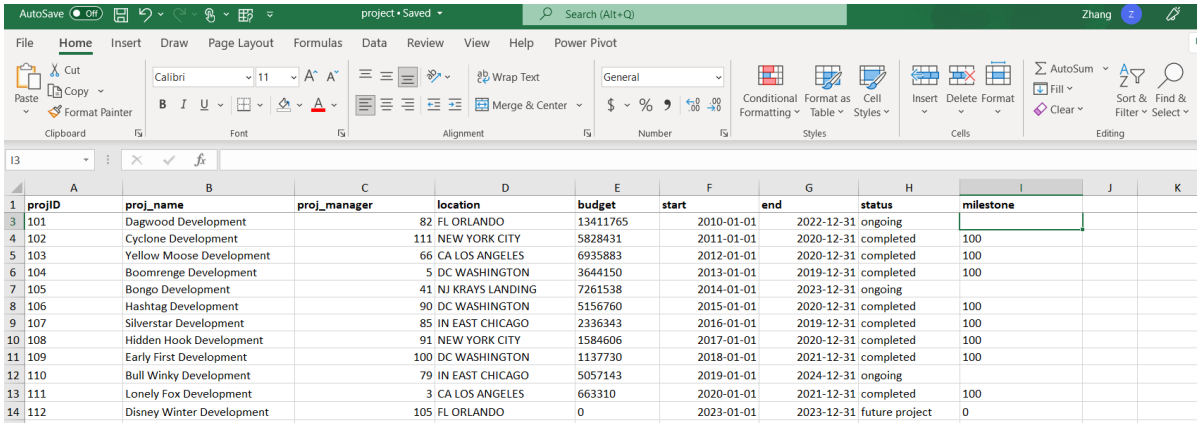


	A	B	E
1	sal_code	title_job	hourly_pay
2	S2	administrative assistant	21.63
3	S5	billing coordinator	24.04
4	S3	accountant	26.44
5	S12	finance analyst	28.85
6	S27	it support technician	28.85
7	S28	lane technician	28.85
8	S1	contractor	30.00
9	S4	architectural specialist	31.25

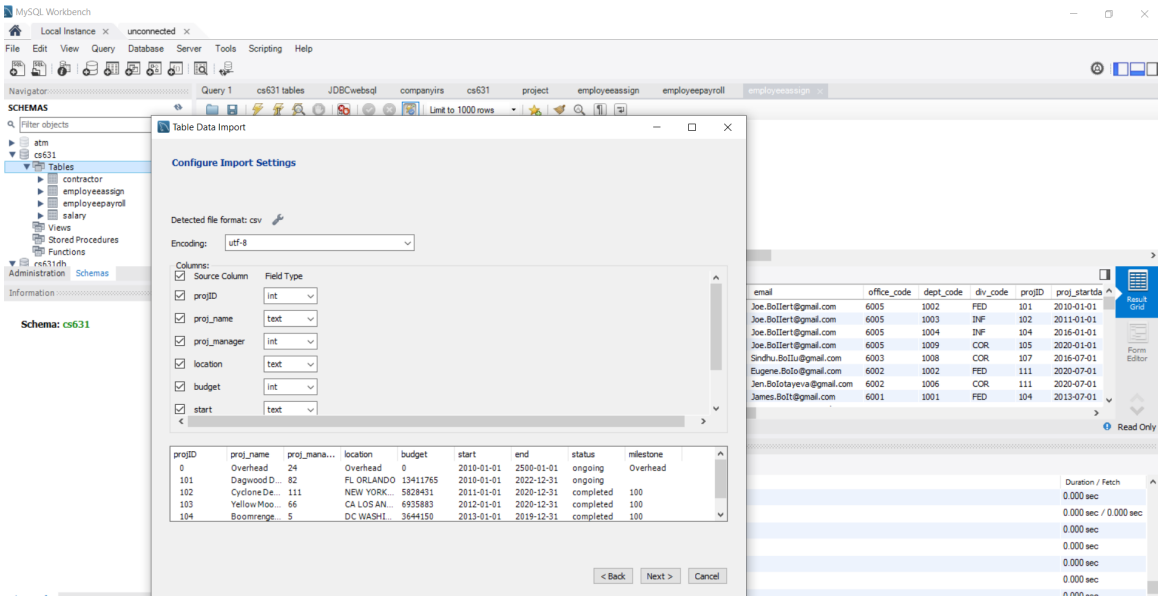


5) Import data into table project

Add data into project csv file and import it into mySQL using the Table Data Import Wizard option. See the screenshots below.

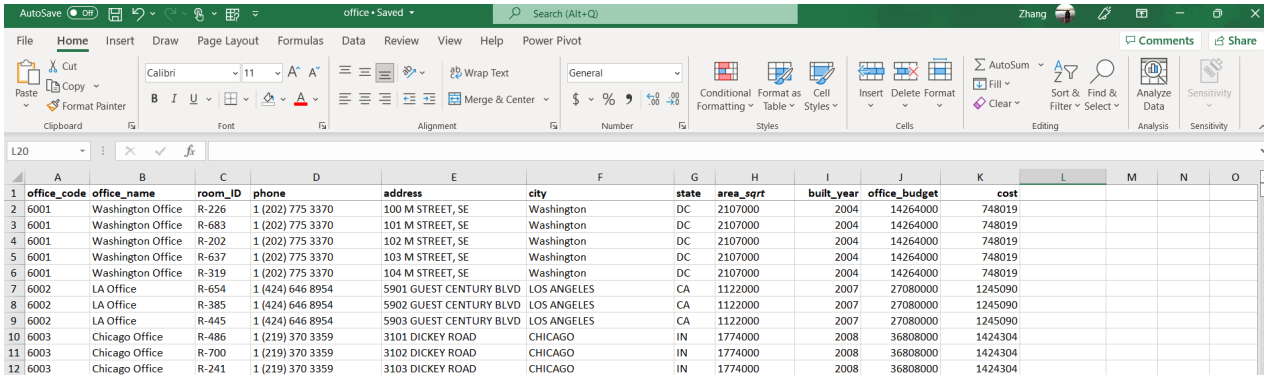


projID	proj_name	proj_manager	location	budget	start	end	status	milestone
101	Dagwood Development	82	FL ORLANDO	13411765	2010-01-01	2022-12-31	ongoing	
102	Cyclone Development	111	NEW YORK CITY	5828431	2011-01-01	2020-12-31	completed	100
103	Yellow Moose Development	66	CA LOS ANGELES	6935883	2012-01-01	2020-12-31	completed	100
104	Boomrenge Development	5	DC WASHINGTON	3644150	2013-01-01	2019-12-31	completed	100
105	Bongo Development	41	NJ KRAYS LANDING	7261538	2014-01-01	2023-12-31	ongoing	
106	Hashtag Development	90	DC WASHINGTON	5156760	2015-01-01	2020-12-31	completed	100
107	Silverstar Development	85	IN EAST CHICAGO	2336343	2016-01-01	2019-12-31	completed	100
108	Hidden Hook Development	91	NEW YORK CITY	1584606	2017-01-01	2020-12-31	completed	100
109	Early First Development	100	DC WASHINGTON	1137730	2018-01-01	2021-12-31	completed	100
110	Bull Winky Development	79	IN EAST CHICAGO	5057143	2019-01-01	2024-12-31	ongoing	
111	Lonely Fox Development	3	CA LOS ANGELES	663310	2020-01-01	2021-12-31	completed	100
112	Disney Winter Development	105	FL ORLANDO	0	2023-01-01	2023-12-31	future project	0

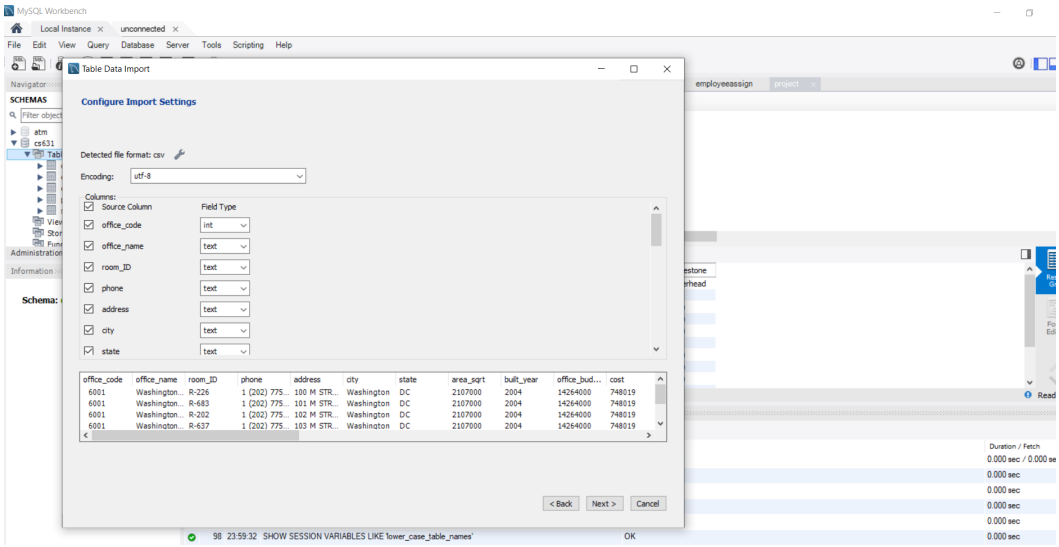


6) Import data into table office_building

Add data into office csv file and import it into mySQL using the Table Data Import Wizard option. See the screenshots below.

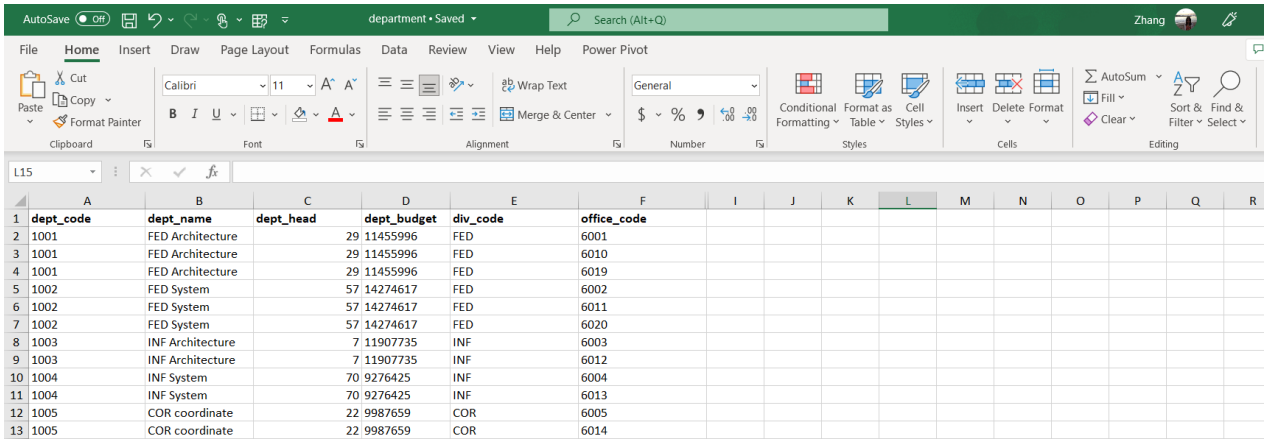


office_code	office_name	room_ID	phone	address	city	state	area_sqrt	built_year	office_budget	cost
6001	Washington Office	R-226	1 (202) 775 3370	100 M STREET, SE	Washington	DC	2107000	2004	14264000	748019
6001	Washington Office	R-683	1 (202) 775 3370	101 M STREET, SE	Washington	DC	2107000	2004	14264000	748019
6001	Washington Office	R-202	1 (202) 775 3370	102 M STREET, SE	Washington	DC	2107000	2004	14264000	748019
6001	Washington Office	R-637	1 (202) 775 3370	103 M STREET, SE	Washington	DC	2107000	2004	14264000	748019
6001	Washington Office	R-319	1 (202) 775 3370	104 M STREET, SE	Washington	DC	2107000	2004	14264000	748019
6002	LA Office	R-654	1 (424) 646 8954	5901 GUEST CENTURY BLVD	LOS ANGELES	CA	1122000	2007	27080000	1245090
6002	LA Office	R-385	1 (424) 646 8954	5902 GUEST CENTURY BLVD	LOS ANGELES	CA	1122000	2007	27080000	1245090
6002	LA Office	R-445	1 (424) 646 8954	5903 GUEST CENTURY BLVD	LOS ANGELES	CA	1122000	2007	27080000	1245090
6003	Chicago Office	R-486	1 (219) 370 3359	3101 DICKEY ROAD	CHICAGO	IN	1774000	2008	36808000	1424304
6003	Chicago Office	R-700	1 (219) 370 3359	3102 DICKEY ROAD	CHICAGO	IN	1774000	2008	36808000	1424304
6003	Chicago Office	R-241	1 (219) 370 3359	3103 DICKEY ROAD	CHICAGO	IN	1774000	2008	36808000	1424304

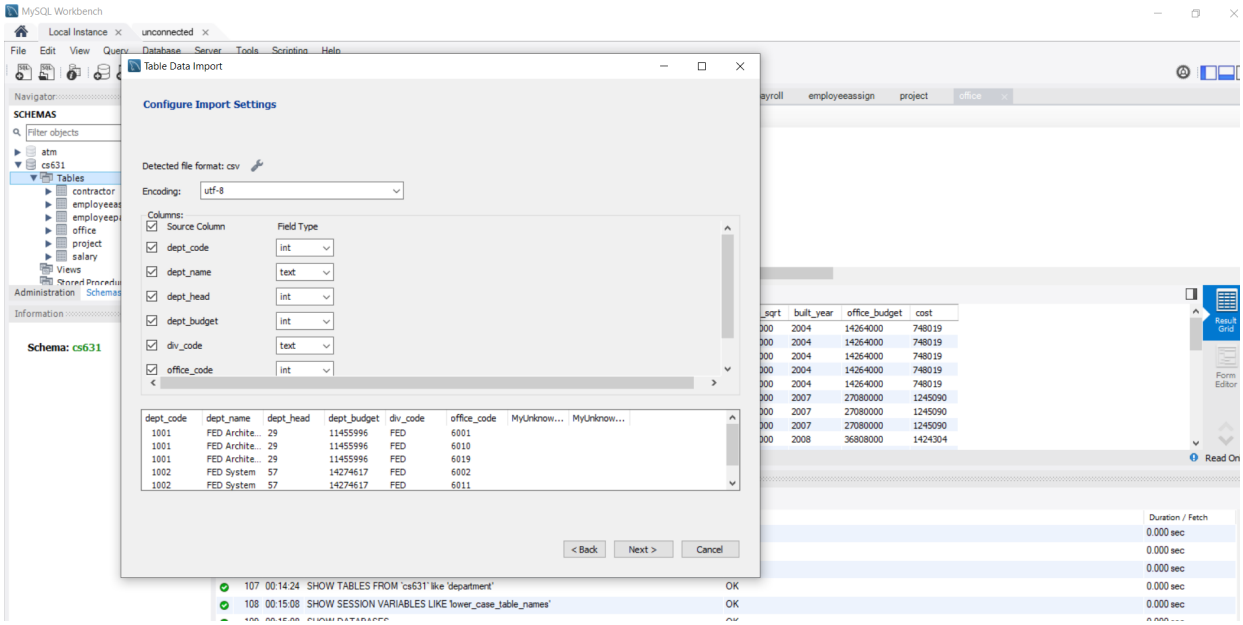


7) Import data into table department

Add data into department csv file and import it into mySQL using the Table Data Import Wizard option. See the screenshots below.

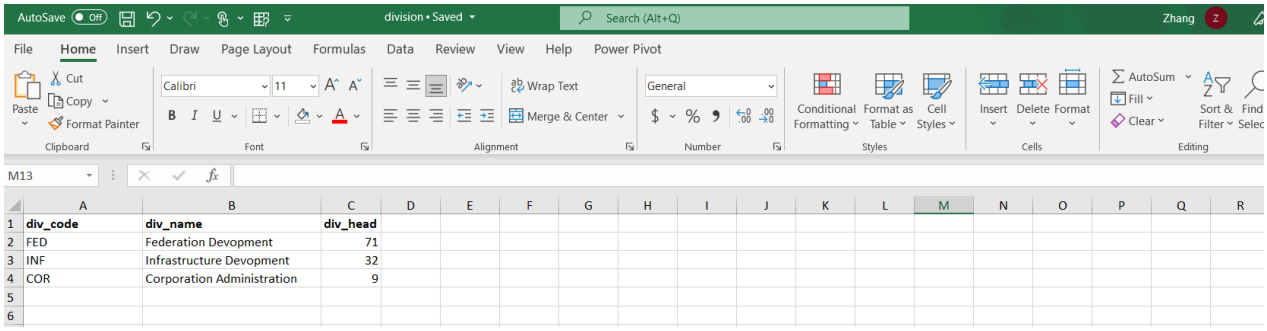


	A	B	C	D	E	F
1	dept_code	dept_name	dept_head	dept_budget	div_code	office_code
2	1001	FED Architecture	29	11455996	FED	6001
3	1001	FED Architecture	29	11455996	FED	6010
4	1001	FED Architecture	29	11455996	FED	6019
5	1002	FED System	57	14274617	FED	6002
6	1002	FED System	57	14274617	FED	6011
7	1002	FED System	57	14274617	FED	6020
8	1003	INF Architecture	7	11907735	INF	6003
9	1003	INF Architecture	7	11907735	INF	6012
10	1004	INF System	70	9276425	INF	6004
11	1004	INF System	70	9276425	INF	6013
12	1005	COR coordinate	22	9987659	COR	6005
13	1005	COR coordinate	22	9987659	COR	6014

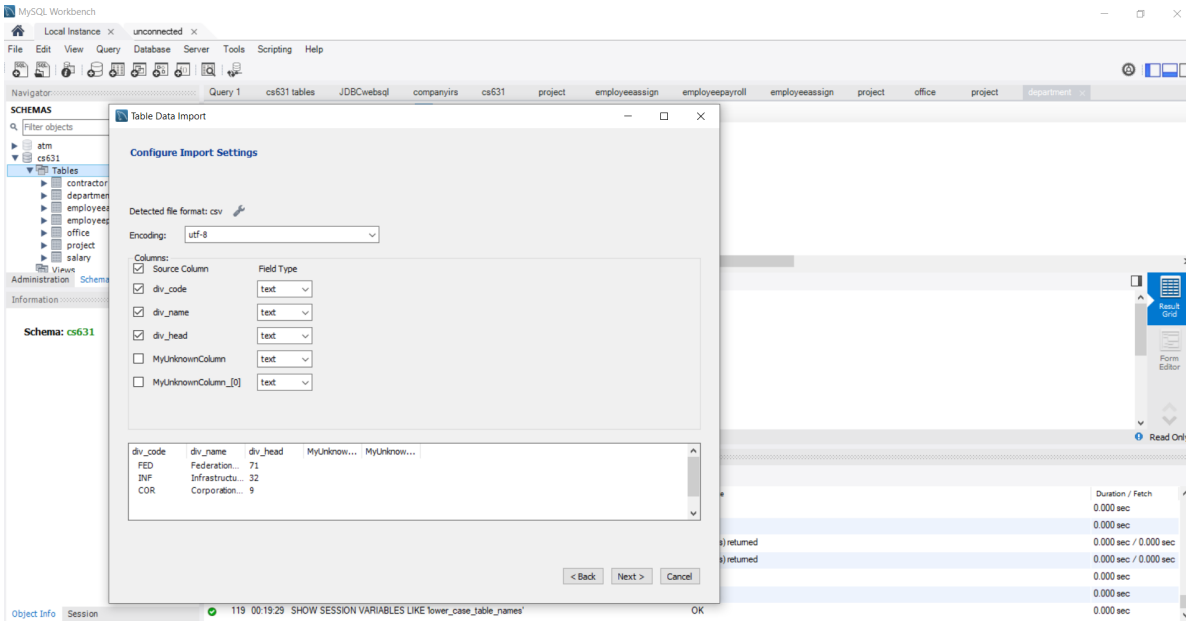


8) Import data into table division

Add data into division csv file and import it into mySQL using the Table Data Import Wizard option. See the screenshots below.



	A	B	C
1	**div_code**	**div_name**	**div_head**
2	FED	Federation Devopment	71
3	INF	Infrastructure Devopment	32
4	COR	Corporation Administration	9
5			
6			



8. Database Query

The following are major database views and queries for payroll users and project management users. The views and queries can be updated by users from web applications as shown in Appendix C. More precisely, these queries are imported into java/html web applications to allow users to interact with the database, query, update, and generate all reports including paychecks, monthly cost reports, IRS reports, project status reports, and project assignments and updates, etc. as required and shown in Section 9.

8.1 Payroll application

8.1.1 Create SQL employee view for the company

This view is used for online users to limit attributes that can be revealed to users, such as empID, associated assignments and time periods to improve real-time queries from all clients. This view is dynamic and updated each time upon changes to employee or project assignment dates. The hiring dates and project assignment dates are included to support queries to check valid income, e.g., only employees with valid projects within the subject month will have paychecks, only projects with employees assigned will count for work-hours.

```
CREATE VIEW empview AS
SELECT e.empID, e.name, ea.title, e.start_date, e.address, e.city, e.state,
e.zip, e.SSN, e.email, e.office_code, office.phone, office.office_name, ea.dept_code, ea.div_code, ea.projID,
ea.proj_startdate as start, ea.proj_enddate as end, salary.salary, salary.classification, salary.hourly_pay,
 project.proj_name
FROM employeepayroll as e
JOIN employeeassign as ea ON e.empID=ea.empID
JOIN office ON e.office_code =office.office_code
JOIN project ON ea.projID =project.projID
```

```
JOIN salary ON ea.title=salary.title
;
```

8.1.2 Generate monthly paychecks for permanent employee

Use empID number 1 as an example , that he's hired and charged at the subject month. For the salaries, we will assume 10% federal tax, 5% state tax and 3% for other taxes and deductions for simplicity. This view is dynamic and updated each time upon changes to employee or project assignment dates. Only employees with valid projects will have paychecks.

```
CREATE VIEW empsal AS
SELECT * FROM cs631db.empview
WHERE empID = '1' AND start_date<='2020-12-31' AND start<='2020-01-01' AND '2020-01-01'<=end;
```

8.1.3 Generate monthly paychecks for hourly employee

In addition to the regular employees of the company that are all salaried, the company employs contractors for specific projects. These employees are paid hourly, and their benefit and work period is limited to the duration of the project. Meanwhile, a VP is on overhead, not assign to any project. The application would help manage all the employee and pay their salaries every month. Use empID number 103 as an example , that he's hired and charged at the subject month.

```
CREATE VIEW empsalhour AS
SELECT * FROM cs631db.empview
WHERE empID = '103' AND start_date<='2020-12-31' AND start<='2020-01-01' AND '2020-01-01'<=end;
```

8.1.4 Generate monthly cost reports across a department

The department cost report combines two parts: for permanent employees and for hourly employees:

This view is to generate monthly report across all departments for permanent employees. This view is dynamic and updated each time upon changes to periods and employee salary. Only employees with valid projects will have paychecks and will be included in the summary. Use dept_code 1001 as an example below.

```
CREATE VIEW deptsalperm AS
SELECT d.dept_code,d.dept_name, e.name as manager, e.phone, d.email, count(*) as permEmp,
sum(view.hourly_pay) as permSal
FROM cs631db.empview as view, department as d, empview as e
WHERE view.dept_code= d.dept_code AND view.classification='permanent' AND d.dept_head=e.empID
AND view.start_date<='2020-12-31'AND view.start<='2020-01-01' AND '2020-01-01'<=view.end
AND d.dept_code='1001';
```

This view is to generate monthly report across all departments for hourly employees. This view is dynamic and updated each time upon changes to periods and employee hourly pay. Only employees with valid projects will have paychecks and will be included in the summary. Use dept_code 1001 as an example below.

```
CREATE VIEW deptsalhour AS
SELECT d.dept_code,d.dept_name, e.name as manager, e.phone, d.email, count(*) as hourEmp,
sum(view.hourly_pay) as hourSal
FROM cs631db.empview as view, department as d, empview as e
WHERE view.dept_code= d.dept_code AND view.classification='hourly' AND d.dept_head=e.empID
AND view.start_date<='2020-12-31'AND view.start<='2020-01-01' AND '2020-01-01'<=view.end
AND d.dept_code='1001';
```

8.1.5 Generate monthly cost reports across a division

The division cost report combines two parts: for permanent employees and for hourly employees.

This view is to generate monthly report across all divisions of permanent employees. Only employees with valid projects will have paychecks and will be included in the summary. Use div_code FED as an example below.

```
CREATE VIEW divsalperm AS
SELECT d.div_code,d.div_name, e.name as manager, e.phone, d.email, count(*) as permEmp,
sum(view.hourly_pay) as permSal
FROM cs631db.empview as view, division as d, empview as e
WHERE view.div_code= d.div_code AND view.classification='permanent' AND d.div_head=e.empID
AND view.start_date<='2020-12-31'AND view.start<='2020-01-01' AND '2020-01-01'<=view.end
AND d.div_code='FED';
```

This view is to generate monthly cost report across all divisions of hourly employees. Only employees with valid projects will have paychecks and will be included in the summary. Use div_code FED as an example below.

```
CREATE VIEW divsalhour AS
SELECT d.div_code,d.div_name, e.name as manager, e.phone, d.email,count(*) as hourEmp,
sum(view.hourly_pay) as hourSal
FROM cs631db.empview as view, division as d, empview as e
WHERE view.div_code= d.div_code AND view.classification='hourly' AND d.div_head=e.empID
AND view.start_date<='2020-12-31'AND view.start<='2020-01-01' AND '2020-01-01'<=view.end
AND d.div_code='FED';
```

8.1.6 Generate IRS tax forms for permanent employees at the end of each year

This view is to prepare annual IRS tax forms for permanent employees. Since not all employees work the entire span of every project, we have to calculate partial IRS years for those who have only assigned late to the project and who terminated early. This query seeks for instances of

- Through the entire IRS year, the salary will be used as the social security salary,

- First half of the year of IRS reports, the working hours from the beginning of the year to the termination date will be used to get the social security salary,
- Second half of the year of IRS reports, the working hours from the project assignment date to the end of the year will be used to get the social security salary,

All calculations are depending on the exact date assigned to the project which can start late or leave early, or throughout the IRS year. For state taxes and salaries, we will assume 10% federal tax, 5% state tax and 3% for other taxes and deductions for simplicity. We also assume 6.2% social security taxes, 1.45% medical taxes in order to calculate state wages deducted from social security wages. Those calculations are performed in web applications as shown in Appendix A. Use empID number 1 as an example below.

```
/* check if full year for IRS report */
DROP view if exists empIRS20;
CREATE VIEW empIRS20 AS
SELECT *, salary as income FROM cs631db.empview
WHERE empID = '1' AND start_date<='2020-12-31' AND start<='2020-01-01' AND '2020-12-31'<=end;

/* check if project start at last half partial year */
DROP view if exists empIRS20p1;
CREATE VIEW empIRS20p1 AS
SELECT *, hourly_pay*DATEDIFF('2020-12-31', start)/7*40 as income  FROM cs631db.empview
WHERE empID = '1' AND start_date<='2020-12-31' AND start>='2020-01-01' AND start<='2020-12-31';

/* check if project end at first half partial year */
DROP view if exists empIRS20p2;
CREATE VIEW empIRS20p2 AS
SELECT *, hourly_pay*DATEDIFF(end, '2020-01-01')/7*40  FROM cs631db.empview
WHERE empID = '1' AND start_date<='2020-12-31' AND end>='2020-01-01' AND end<='2020-12-31';
```

8.1.7 Generate IRS tax forms for hourly employees at the end of each year

This view is to prepared annual IRS tax forms for hourly employees. All calculations are depending on the exact date assigned to the project which can start late or leave early, or throughout the IRS year. For state taxes and salaries, we will assume 10% federal tax, 5% state tax and 3% for other taxes and deductions for simplicity. We also assume 6.2% social security taxes, 1.45% medical taxes in order to calculate state wages deducted from social security wages. Those calculations are performed in web applications as shown in Appendix A. Use empID number 103 as an example below.

```
/* for late join/start, check if hourly at 2011 start at last half partial year */
DROP view if exists empIRS11hr;
CREATE VIEW empIRS11hr AS
SELECT *, hourly_pay*DATEDIFF('2011-12-31',start)/7*40 as income  FROM cs631db.empview
WHERE empID = '103' AND start_date<='2011-12-31' AND start>='2011-01-01' AND start<='2011-12-31';

/* for early leave, check if hourly at 2018 end at first half partial year */
DROP view if exists empIRS18hr;
CREATE VIEW empIRS18hr AS
SELECT *, hourly_pay*DATEDIFF(end, '2018-01-01')/7*40 as income  FROM cs631db.empview
WHERE empID = '103' AND start_date<='2018-12-31' AND end>='2018-01-01' AND end<='2018-12-31';
```

8.1.8 History IRS reports for entire company

This table provides history IRS reports for payroll users, which includes all taxes and salaries for all IRS years. For the taxes and salaries, we will assume 10% federal tax, 5% state tax and 3% for other taxes and deductions for simplicity. We also assume 6.2% social security taxes, 1.45% medical taxes in order to calculate state wages deducted from social security wages.

Due to the nature of random assignment dates of an employee to projects and different types of employees (permanent will work longer but contractors are shorter since he can join late and terminate early), the IRS reports shall cover all scenarios of assignment periods thus a series of SQL queries are used to scan each employee to get the employee's IRS income based on duration on each project in each IRS year.

- Both the start date and end date are in the same IRS year for contractors, working days are from the start date to the end date (one partial IRS year in the report)
- Start and end date are in two IRS years for contractors, working days are from the start date to the end of first year, and from the beginning of second year to the end date (two partial IRS years in the report)
- Work 3+ years on a project but start late at the Second half of the year, working days are from the project assignment date to the end of the year for the first IRS year
- Work 3+ years on a project but terminate early at the First half of the year, working days are from the beginning of the year to the termination date for the last IRS year
- Through the entire IRS year, the salary will be used as the employee's IRS income.

See the SQL queries below to generate view of the employee's IRS income (social security salary) for all employees.

```
/* generate IRS for all employees work for only one IRS year */
DROP view if exists companyirshr;
CREATE VIEW companyirshr AS
SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state,
e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification, e.hourly_pay as unit_pay, 0 as
span, CAST(year(e.start) as signed integer) as IRSyear, (DATEDIFF(e.end,e.start))/7*40*e.hourly_pay as
SSincome
FROM empview as e
where year(e.start)=year(e.end)
UNION
/* generate IRS for all employees work for only two IRS years */
SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state,
e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification, e.hourly_pay as unit_pay, 0 as
span, CAST(year(e.start)as signed integer) as IRSyear,
(DATEDIFF(CONCAT(year(e.start),'-12-31'),start))/7*40*e.hourly_pay  as SSincome
FROM empview as e
where year(e.end)-year(e.start)=1
UNION
SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state,
e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification, e.hourly_pay as unit_pay, 0 as
span, CAST(year(e.end)as signed integer) as IRSyear,
(DATEDIFF(end,CONCAT(year(e.end),'-1-1')))/7*40*e.hourly_pay  as SSincome
FROM empview as e
```

```
where year(e.end)-year(e.start)=1
UNION
/* generate IRS for all employees work for >=three IRS years */
SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state,
e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification, e.hourly_pay as unit_pay, 0 as
span, CAST(year(e.start) as signed integer) as IRSyear,
(DATEDIFF(CONCAT(year(e.start),'-12-31'),start))/7*40*e.hourly_pay as SSincome
FROM empview as e
where year(e.end)-year(e.start)>1
UNION
SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state,
e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification, e.hourly_pay as unit_pay, 0 as
span, CAST(year(e.end) as signed integer)as IRSyear,
(DATEDIFF(end,CONCAT(year(e.end),'-1-1')))/7*40*e.hourly_pay as SSincome
FROM empview as e
where year(e.end)-year(e.start)>1
UNION
SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state,
e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification, e.salary as unit_pay,
(year(e.end)-year(e.start)-1) as span, CAST((year(e.start)+1)as signed integer) as IRSyear, e.salary as SSincome
FROM empview as e
where year(e.end)-year(e.start)>1
;
```

After the employee's IRS income (social security salary) is obtained, another SQL query is performed to calculate all sorts of federal, state, social security, medicare taxes and salaries, which will be used as the final IRS reports for the entire company for users to query from web applications.

We scanned 100 employees in our sample database with random start and end dates to 10 projects, and total entries of IRS reports during 2010 to 2022 are around 700, see Appendix D for details.

```
Integer taxFed = (int) (empIRSIncome * 0.1);
Integer taxSta = (int) (empIRSIncome * 0.05);
Integer taxSS = (int) (empIRSIncome * 0.062);
Integer medWage=empIRSIncome-taxFed-taxSS;
Integer taxMedi = (int) (empIRSIncome * 0.0145);
Integer netWage=empIRSIncome-taxFed-taxSS-taxMedi;
```

```
/*create a raw table of all taxes and salaries for all IRS years*/
DROP table if exists companyirs;
CREATE table companyirs AS
SELECT c.empID, c.name, c.address, c.city, c.state, c.zip, c.SSN, c.office_code, c.dept_code, c.div_code,
c.classification,
c.span, c.IRSyear, c.SSincome, SSincome*0.1 as FederalTax, SSincome*0.05 as StateTax, SSincome*0.062 as
SSTax, SSincome*0.0145 as MedicalTax, SSincome*(1-0.1-0.05-0.062-0.0145) as StateWage
FROM companyirshr as c
WHERE c.start_date<=CURDATE();

/* select years of IRS report to generate a IRS report for the entire company*/
SELECT * FROM companyirs WHERE IRSyear>=2010 AND IRSyear<=2020 ORDER BY IRSyear, empID ASC;
```

8.2 Project Management Application

8.2.1 Insert a new project

A new project with ID, name, budget, location, duration, etc. will insert into the table and a trigger will update the table project. Below is an example of the SQL queries. Note that all parameters in the SQL query will be provided by users from the html web applications.

```
INSERT INTO project VALUES ('112', 'Disney Winter Development', null, 'FL ORLANDO', '10000000', null, null);
```

8.2.2 Assign the project manager and the project team

Users can assign the available project manager and the project team members to the new project. A trigger will update the table project and table employee_assign. Below is an example of the SQL queries. Note that all parameters in the SQL query will be provided by users from the html web applications.

```
UPDATE project SET proj_manager = '105', start = '2023-1-1', end = '2024-1-1' WHERE projID = '112';
UPDATE employeeassign SET projID = '112', proj_startdate='2023-1-1', proj_enddate='2024-1-1'  WHERE empID
= '105';
UPDATE employeeassign SET projID = '112', proj_startdate='2023-1-1', proj_enddate='2024-1-1'  WHERE empID
= '7';
UPDATE employeeassign SET projID = '112', proj_startdate='2023-1-1', proj_enddate='2024-1-1'  WHERE empID
= '9';
UPDATE employeeassign SET projID = '112', proj_startdate='2023-1-1', proj_enddate='2024-1-1'  WHERE empID
= '10';
```

8.2.3 Generate progress statistics on all projects

A series of SQL queries are developed to track progress statistics on the project, including number of employees working on the project, total person-hours per project and total expenditure as of today.

- Project Current Status: check if a project is completed, new or on-going by the end date
- Total employees assigned and total person-hours: scan all employees to aggregate totals for each project
- Total charges and percent (%) of progress based on total budget of each project.

The query results are shown in Appendix D for all projects including existing in database and new inserted by online users. Below is the list of queries developed and used in web applications.

```
/*Track project status and mark up for each project -3 status, completed, ongoing and future   */
DROP view if exists projectSta;
```

```
CREATE VIEW  projectSta AS
(SELECT projID, year(end)-year(start) AS span, 'COMPLETED' as status, '100' as milestone FROM project
WHERE (CURDATE()>=end AND projID != '0')
UNION
SELECT projID, '0' AS span, 'FUTURE' as status, '0' as milestone FROM project WHERE (CURDATE()<=start
AND projID != '0')
UNION
SELECT projID, year(CURDATE())-year(start) AS span, 'ONGOING' as status, null milestone FROM project
WHERE (start<=CURDATE() AND CURDATE()<=end AND projID != '0')
);

/*Track man-hour and cost for each emp on the project, note that VP on overhead thus not included and contractor
on shorter durations   */
DROP view if exists projectEmpHr;
CREATE VIEW  projectEmpHr AS
(SELECT *, DATEDIFF(end, start)/7*40 AS workhour, DATEDIFF(end, start)/7*40*hourly_pay as charge FROM
empview2 WHERE (CURDATE()>=end AND projID != '0')
UNION
SELECT *, '0' AS workhour, '0' as charge FROM empview2 WHERE (CURDATE()<=start AND projID != '0')
UNION
SELECT *, DATEDIFF(end, start)/7*40 AS workhour, DATEDIFF(end, start)/7*40*hourly_pay as charge FROM
empview2 WHERE (start<=CURDATE() AND CURDATE()<=end AND projID != '0')
);

/* Track progress statistics, total man-hour and cost on ALL projects   */
DROP view if exists projectProgress;
CREATE VIEW projectProgress AS
SELECT * FROM
(SELECT p.projID, p.proj_name, p.proj_manager, p.location, p.budget, p.start, p.end, s.span, s.status, e.name,
COUNT(DISTINCT e.empID) as totalEmp, SUM(eh.charge) as totalCharge, SUM(eh.workhour) as totalworkhour,
SUM(eh.charge)/p.budget as prog_milestone
FROM project p, projectSta s, projectEmpHr eh, empview2 e
WHERE p.projID=s.projID AND p.proj_manager=e.empID AND p.projID=eh.projID
GROUP BY p.projID) as results
ORDER BY results.projID ASC;
```

8.2.4 Track project milestones with progress bars and status of one project

This view is to track project status and mark up progress status for a project requested by online users (use project 105 or “Bongo Development” in New Jersey as an example).

```
/* Track progress statistics, total man-hour and cost on each project   */
SELECT * FROM projectProgress WHERE projID='105';
```

9. Java API and HTML Web Applications

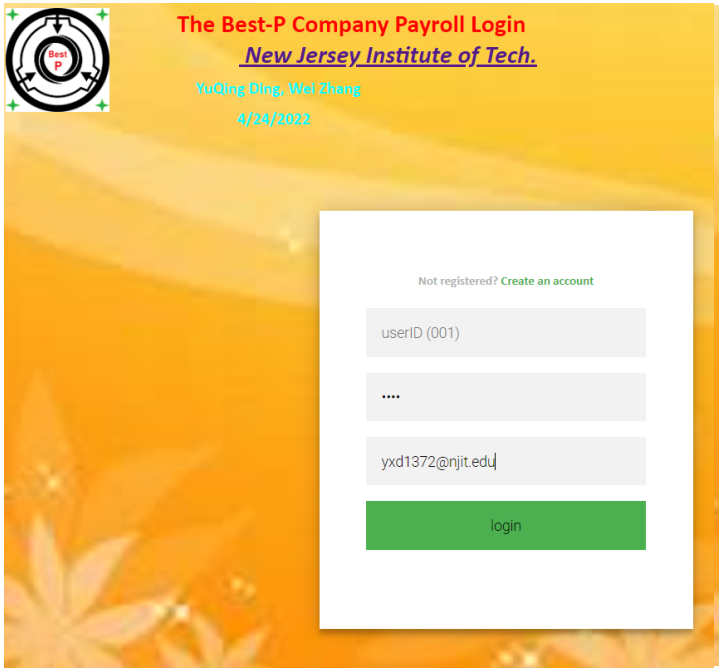
The java and web application for Best-P database system are designed with 3 types of html pages, including authentication, Human Resources and Payroll, and Project Management, as shown in section 8. All queries in section 8 are modified to prepared queries for Java servlets to

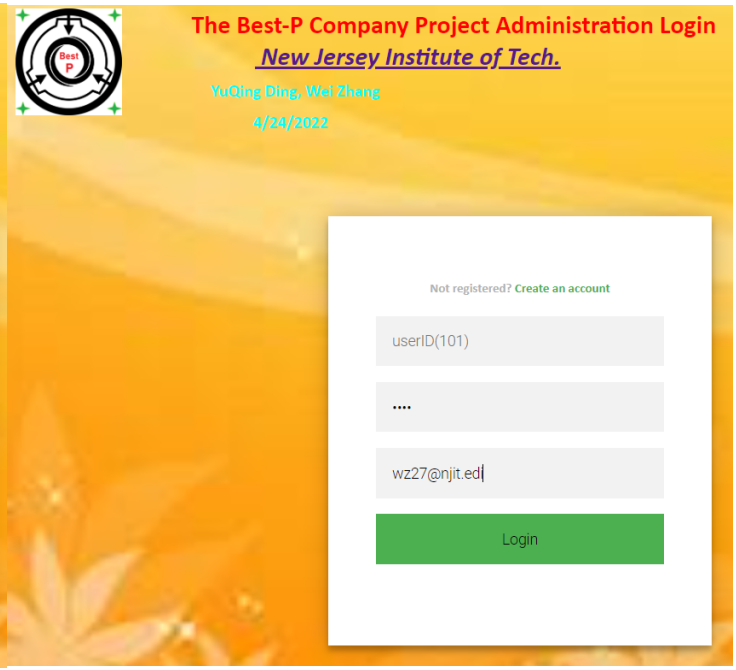
process online requests from web pages and perform SQL queries in backend database by JDBC query engine.

The online users can perform queries on the backend database in real-time simultaneously.

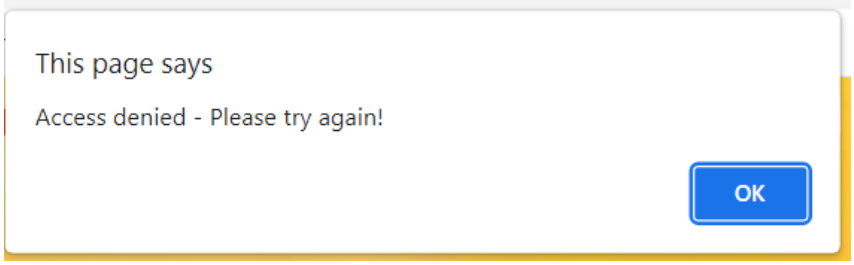
9.1 User Login Manuals for Payroll and Project Management

Each application has a login page with authentication process, to verify, grant or deny access to the applications. Each user will check against saved passwords. Access will be denied and html page disabled if failed. The GUIs below shows two login pages for Payroll user and Project management user, respectively. A user will need to enter a userID, password and email in order to access the interface.

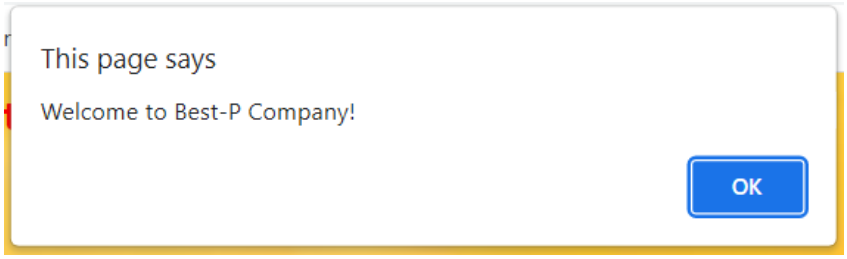




A popup alert message shows the error when user input wrong userID or password. Click on 'OK' to try input again.



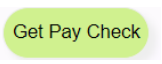
When the user input the correct userID and password, a successful authentication message popup to grant access. Click on ‘OK’ to get into the application interface.



9.2 User Manual for Payroll with (5) major functions

The main web page for payroll application provides pay check options and IRS report types for the payroll users to implement five major functions. When the user clicks on a checkbox and enters all required parameters, then submit the form by using one of colored round buttons, it will make a request to the back end and return the SQL result. See the sample pages as follows.

(1) Generate paychecks with monthly salary and tax history for all types of employees.

The GUI below shows an example to get paycheck report for Employee ID#1, payroll period January, 2020. The user clicks on the first checkbox and enters the required parameters, then submit the form by clicking 

button.

When user clicks on the button Get Pay Check , it will make a request to the back end DB and return the paycheck result with monthly salary for EmployeeID-01 in January 2020, along with his personal information such as ID, name, phone, email, etc.

Pay Check #: 101
Created: 4/24/2022

Time Period: January 2020

The Best-P Company
7545 MLK J. Blvd
Newark, NJ 07102
USA Headquarter

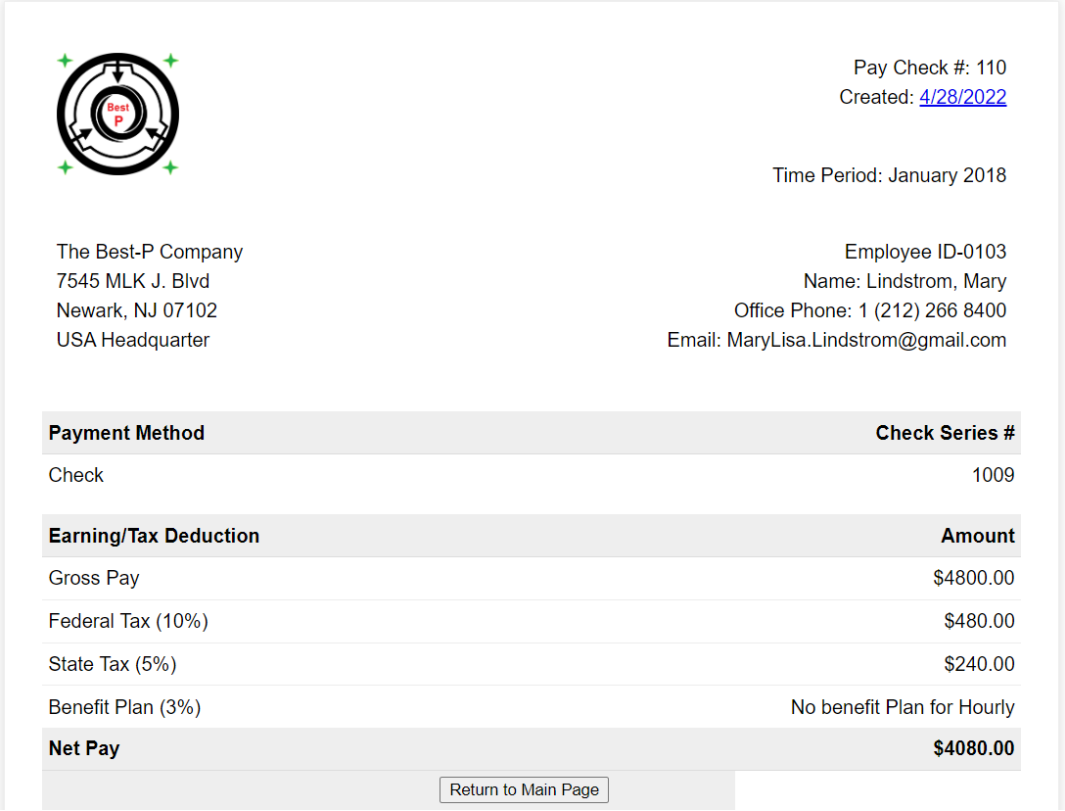
Employee ID-01
Name: Boller, Joe
Office Phone: 1 (212) 266 8400
Email: Joe.Bollert@gmail.com

Payment Method	Check Series #
Check	1001

Earning/Tax Deduction	Amount
Gross Pay	$7692.80
Federal Tax (10%)	$769.28
State Tax (5%)	$384.64
Benefit Plan (3%)	$230.78
Net Pay	**$6308.10**

Return to Main Page

If the employee is NOT permanent, his monthly salary will look differently due to no benefit plans. Below is the return of the paycheck result with monthly salary for EmployeeID-103 in June 2018, along with her personal information such as ID, name, phone, email, etc.



Pay Check #: 110
Created: 4/28/2022

Time Period: January 2018

The Best-P Company
7545 MLK J. Blvd
Newark, NJ 07102
USA Headquarter

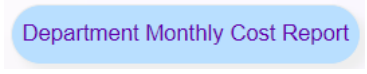
Employee ID-0103
Name: Lindstrom, Mary
Office Phone: 1 (212) 266 8400
Email: MaryLisa.Lindstrom@gmail.com

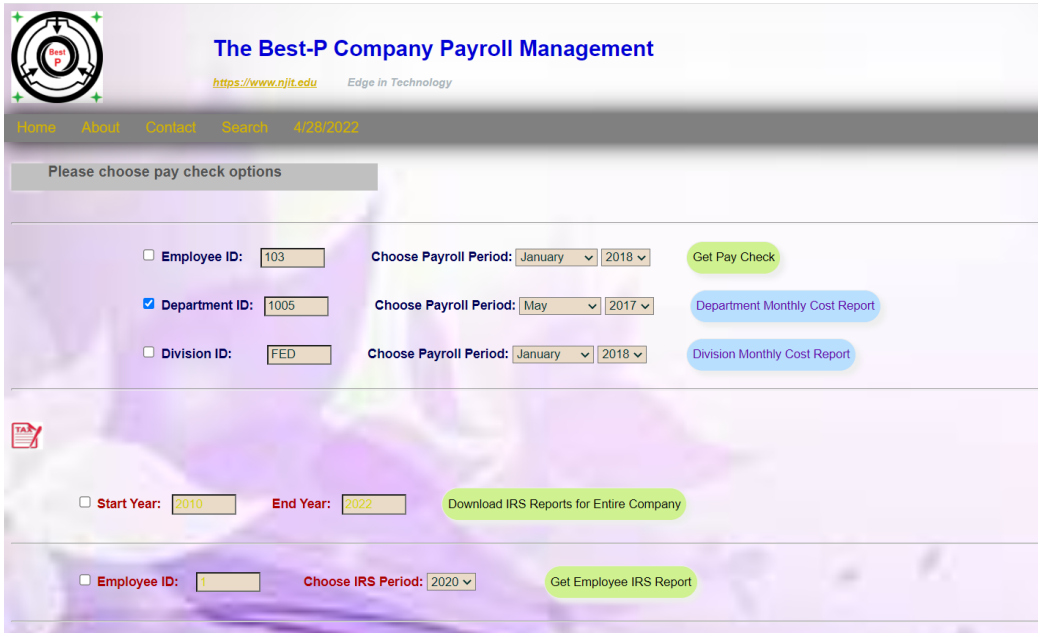
Payment Method	Check Series #
Check	1009

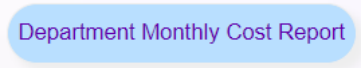
Earning/Tax Deduction	Amount
Gross Pay	$4800.00
Federal Tax (10%)	$480.00
State Tax (5%)	$240.00
Benefit Plan (3%)	No benefit Plan for Hourly
Net Pay	**$4080.00**

Return to Main Page

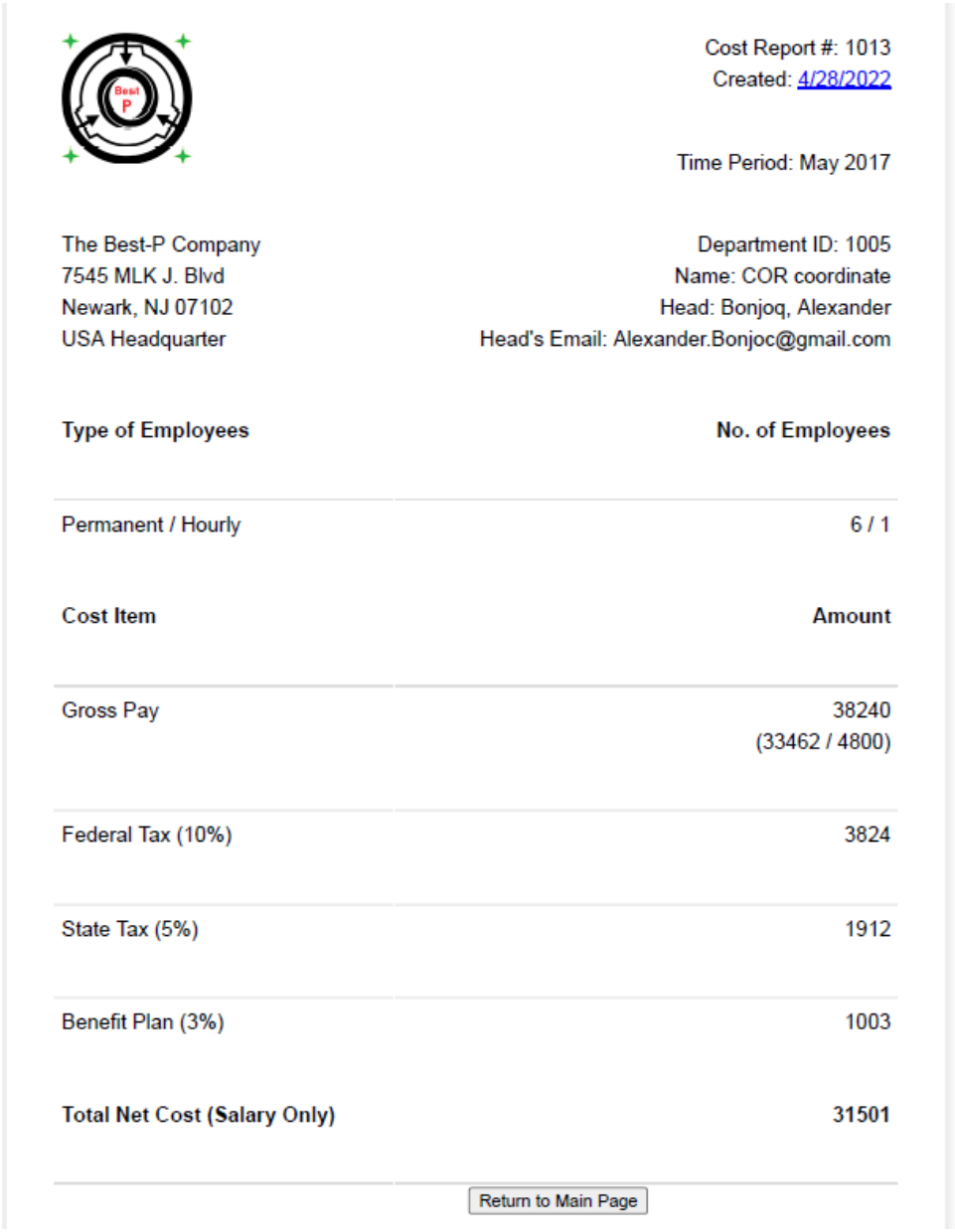
(2) Generate monthly cost reports for all departments.

The GUI below shows an example to get monthly cost report for department ID#1005, payroll period May, 2017. The user clicks on the second checkbox and enters the required parameters, then submit the form by clicking  button.



When user click on the button 

, it will make a request to the back end DB and return the detailed department cost report including both permanent and hourly employees, along with basic information such as department ID, head's name, email, etc.



Cost Report #: 1013
Created: 4/28/2022

Time Period: May 2017

The Best-P Company
7545 MLK J. Blvd
Newark, NJ 07102
USA Headquarter

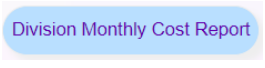
Department ID: 1005
Name: COR coordinate
Head: Bonjoq, Alexander
Head's Email: Alexander.Bonjoc@gmail.com

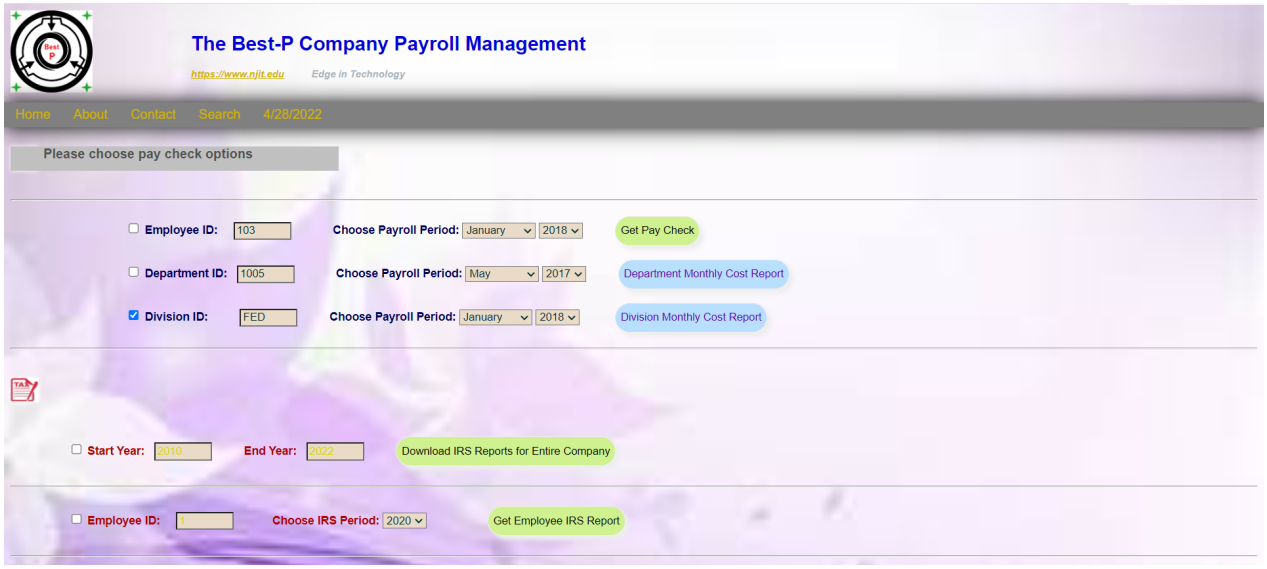
Type of Employees	No. of Employees
Permanent / Hourly	6 / 1

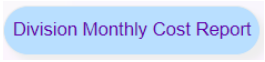
Cost Item	Amount
Gross Pay	38240 (33462 / 4800)
Federal Tax (10%)	3824
State Tax (5%)	1912
Benefit Plan (3%)	1003
Total Net Cost (Salary Only)	**31501**

Return to Main Page

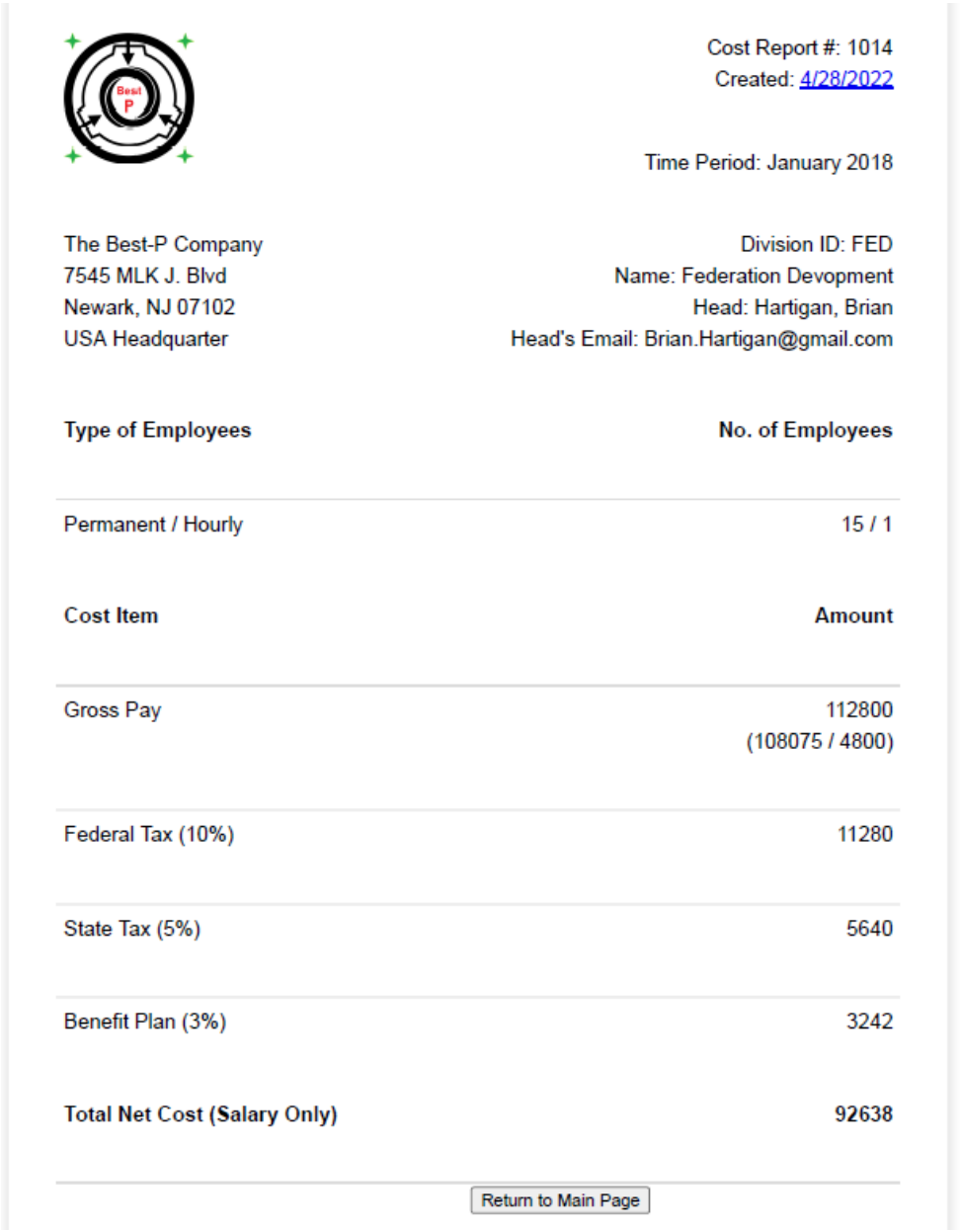
(3) Generate cost reports for all divisions.

The GUI below shows an example to get cost report for Division ID FED during January, 2018. The user clicks on the third checkbox and enters the required parameters, then submit the form by clicking  button.



When user click on the 

button, it will make a request to the back end DB and return the detailed division cost report including both permanent and hourly employees, along with basic information such as division ID, head's name, email, etc.



Cost Report #: 1014
Created: 4/28/2022

Time Period: January 2018

The Best-P Company
7545 MLK J. Blvd
Newark, NJ 07102
USA Headquarter

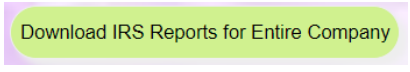
Division ID: FED
Name: Federation Devopment
Head: Hartigan, Brian
Head's Email: Brian.Hartigan@gmail.com

Type of Employees	No. of Employees
Permanent / Hourly	15 / 1

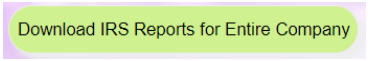
Cost Item	Amount
Gross Pay	112800 (108075 / 4800)
Federal Tax (10%)	11280
State Tax (5%)	5640
Benefit Plan (3%)	3242
Total Net Cost (Salary Only)	**92638**

(4) Users can download history IRS reports for entire company.

The GUI below shows an example to get IRS reports for entire company from 2010 to 2022. The user clicks on the checkbox and enters the required parameters, then submit the form by clicking



button.

When user click on the 

button, it will make a request to the back end DB and return the detailed IRS report for all employees sorted by all the requested years and then by employee ID, the logic details are in section 8. Due to the length of 700 records, screenshots are only partial, please refer to Appendix D for the entire report.

YuQing Ding, Wei Zhang, CS631 Term Project Date: 2022-04-24

The Best-P Company IRS Records

Year	Emp ID	Emp Name	address	city	state	zip	SSN	Social.S.Wage	Federal Tax	State Tax	Social.S.Tax	Medical Tax	State_Wage
2010	1	Boller, Joe	500 COLLEGE AVE	NYC	NY	10010	766-26-5829	69992	6999	3499	4339	1014	54138
2010	24	Bonzo, David	108 IRVINE AVE	DC	DC	20001	957-98-7369	249995	24999	12499	15499	3624	193371
2010	36	Gee, Jamie	604 CLEARVIEW DR	OL	FL	32801	281-02-5840	35188	3518	1759	2181	510	27218
2010	38	Gee, Vanzetta	605 CLEARVIEW DR	OL	FL	32801	357-07-4939	25138	2513	1256	1558	364	19444
2010	45	Ginnes, Christopher	607 CLEARVIEW DR	OL	FL	32801	442-51-4477	47757	4775	2387	2960	692	36940
2010	47	Ginnis, Marlene	608 CLEARVIEW DR	OL	FL	32801	929-92-8293	32678	3267	1633	2026	473	25276
2010	49	Glochlin, Linda	609 CLEARVIEW DR	OL	FL	32801	187-87-6197	40218	4021	2010	2493	583	31108
2010	54	Gowan, Drew	610 CLEARVIEW DR	OL	FL	32801	624-51-3875	32678	3267	1633	2026	473	25276
2010	55	Gowan, Matthew	611 CLEARVIEW DR	OL	FL	32801	561-42-5595	37708	3770	1885	2337	546	29167
2010	56	Gowen, Dennis	612 CLEARVIEW DR	OL	FL	32801	239-59-4439	42738	4273	2136	2649	619	33058
2010	82	Hartsough, Scott	618 CLEARVIEW DR	OL	FL	32801	259-65-6858	52787	5278	2639	3272	765	40831
2010	87	Lin, Danny	619 CLEARVIEW DR	OL	FL	32801	444-24-1460	60327	6032	3016	3740	874	46663
2010	88	Lin, Edward	620 CLEARVIEW DR	OL	FL	32801	965-75-7206	31371	3137	1568	1945	454	24265
2010	94	Linato, Angelo	622 CLEARVIEW DR	OL	FL	32801	121-02-7827	40218	4021	2010	2493	583	31108
2011	1	Boller, Joe	500 COLLEGE AVE	NYC	NY	10010	766-26-5829	79996	7999	3999	4959	1159	61877
2011	103	Lindstrom, Mary	515 COLLEGE AVE	NYC	NY	10010	417-93-8791	31371	3137	1568	1945	454	24265

(continued)

2022	42	Gill, John	307 KINGS PKWY	CH	IN	60617	796-32-5366	80000	8000	4000	4960	1160	61880
2022	43	Ginley, Stacey	402 REDSTONE RD	KL	NJ	18834	385-01-0067	60000	6000	3000	3720	870	46410
2022	45	Ginnes, Christopher	607 CLEARVIEW DR	OL	FL	32801	442-51-4477	46974	4697	2348	2912	681	36335
2022	47	Ginnis, Marlene	608 CLEARVIEW DR	OL	FL	32801	929-92-8293	32142	3214	1607	1992	466	24862
2022	48	Girr, Emily	308 KINGS PKWY	CH	IN	60617	573-89-4048	75000	7500	3750	4650	1087	58012
2022	49	Glochlin, Linda	609 CLEARVIEW DR	OL	FL	32801	187-87-6197	39558	3955	1977	2452	573	30598
2022	50	Glothlin, Jennifer	403 REDSTONE RD	KL	NJ	18834	671-00-3853	80000	8000	4000	4960	1160	61880
2022	51	Glynn, Colin	404 REDSTONE RD	KL	NJ	18834	872-21-1472	55000	5500	2750	3410	797	42542
2022	52	Govern, Susan	405 REDSTONE RD	KL	NJ	18834	304-40-2230	90000	9000	4500	5580	1305	69615
2022	54	Gowan, Drew	610 CLEARVIEW DR	OL	FL	32801	624-51-3875	32142	3214	1607	1992	466	24862
2022	55	Gowan, Matthew	611 CLEARVIEW DR	OL	FL	32801	561-42-5595	37090	3709	1854	2299	537	28689
2022	56	Gowen, Dennis	612 CLEARVIEW DR	OL	FL	32801	239-59-4439	42037	4203	2101	2606	609	32516
2022	57	Grath, Shaun	406 REDSTONE RD	KL	NJ	18834	105-11-9543	120000	12000	6000	7440	1740	92820
2022	58	Gregore, Chuck	309 KINGS PKWY	CH	IN	60617	799-13-7775	75000	7500	3750	4650	1087	58012
2022	79	Hartley, John	310 KINGS PKWY	CH	IN	60617	370-08-7964	105000	10500	5250	6510	1522	81217
2022	82	Hartsough, Scott	618 CLEARVIEW DR	OL	FL	32801	259-65-6858	51922	5192	2596	3219	752	40161
2022	84	Harvey, Peter	311 KINGS PKWY	CH	IN	60617	231-84-2562	75000	7500	3750	4650	1087	58012
2022	87	Lin, Danny	619 CLEARVIEW DR	OL	FL	32801	444-24-1460	59338	5933	2966	3678	860	45898
2022	89	Lin, Kevin	314 KINGS PKWY	CH	IN	60617	545-90-1938	60000	6000	3000	3720	870	46410
2022	94	Linato, Angelo	622 CLEARVIEW DR	OL	FL	32801	121-02-7827	39558	3955	1977	2452	573	30598

(5) Generate IRS tax forms for all types of employees at the end of each year.

The GUI below shows an example to get 2020 IRS tax form for employee ID #1. The user clicks on the checkbox and enters the required parameters, then submit the form by clicking Get Employee IRS Report button.

When user click on the Get Employee IRS Report button, it will make a request to the back end DB and return the detailed IRS tax form showing tax IDs for employee and employer, all sorts tax deductions, Social Security and State salaries, along with employee and employer information

required by IRS (the application uses faked tax IDs and faked addresses obtained from Google and randomized).

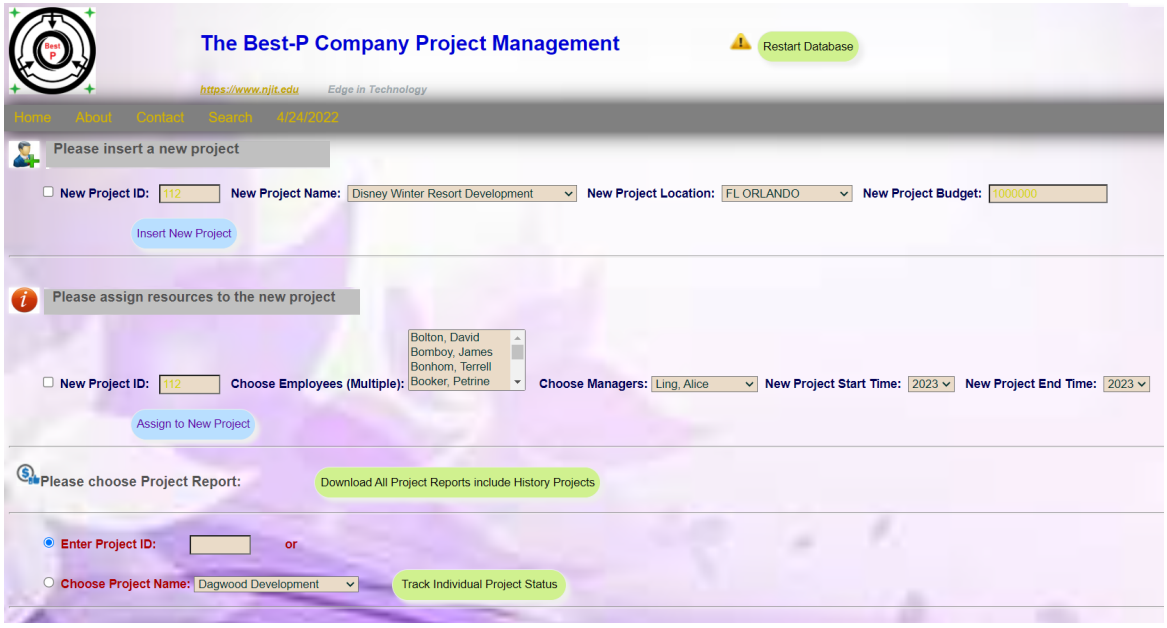
Copy B-To Be Filed with Employee's FEDERAL TAX Return	OMB #: 1234-5678 Created: 4/24/2022 **2020**
a. Employee's Soc. Sec. Number **766-26-5829**	**b. Employer's ID Number** **36-012341**
1. Wages, tips, other comp. $82350	2. Federal income tax. withheld $10000
3. Social security wages $100000	4. Social security tax. withheld $6200
5. Medical wages and tips $83800	6. Medical tax. withheld $1450
c. Employer's name, address, and ZIP code The Best-P Company 7545 MLK J. Blvd Newark, NJ 07102	
d. Employee's name, address, and ZIP code Boller, Joe 500 COLLEGE AVE NYC,NY,10010	

NY State	360-982-12345 Employer's state ID	$ 82350 State wages, tips, etc.	$ 6200 State income tax

Return to Main Page

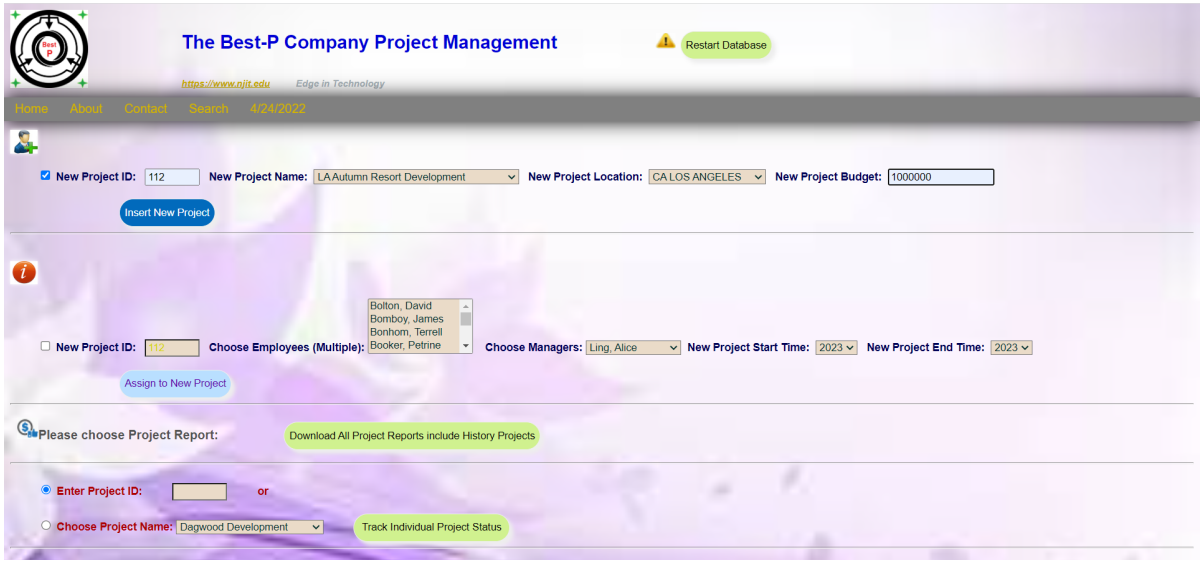
9.3 User Manual for Project Management with (4) major functions

The main web page for project management provides users to implement the major four functions, such as insert project, resource assignments for the project management, and track project status. When the user clicks on a checkbox and enters all required parameters, submits the form by using one of colored round buttons, it will make a request to the back end and return the SQL result. See the sample pages as follows.

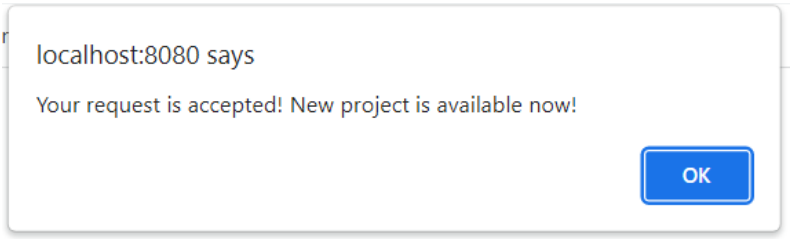


(1) Create a new project.

The GUI below shows an example to create a new project with new project ID '112', name 'LA Autumn Resort Development', budget '$1000000' located in LA. The user clicks on the checkbox and enters the required parameters, then submit the form by clicking Insert New Project button.

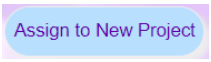


When a user clicks on the Insert New Project button, a pop message will appear and it will make a request to the back end and the database will be updated accordingly. API will inform users by popup message as shown.

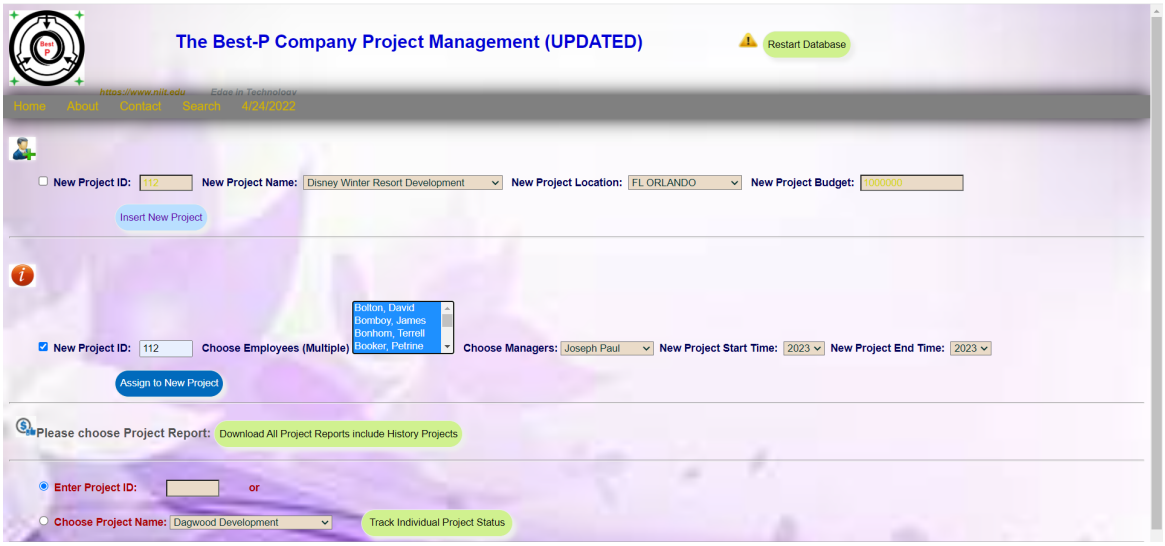


(2) Assign the project manager and project team to a new project.

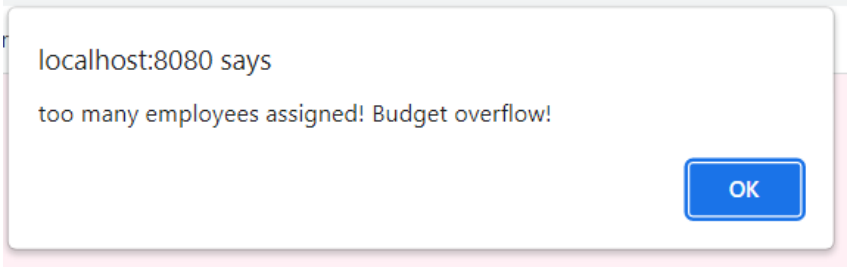
After a new project has been created, the interface will be updated to include the new project in the drop-down list and make it available for users to assign the employees. The GUI below shows an example to assign the project manager and project team to the new project ID 112 created by previous query (users can enter any new project). The user clicks on the checkbox and enters the required parameters from the drop-down list, then submit the form by clicking



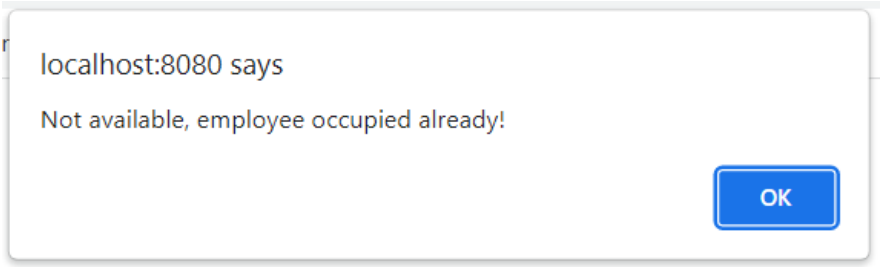
button.



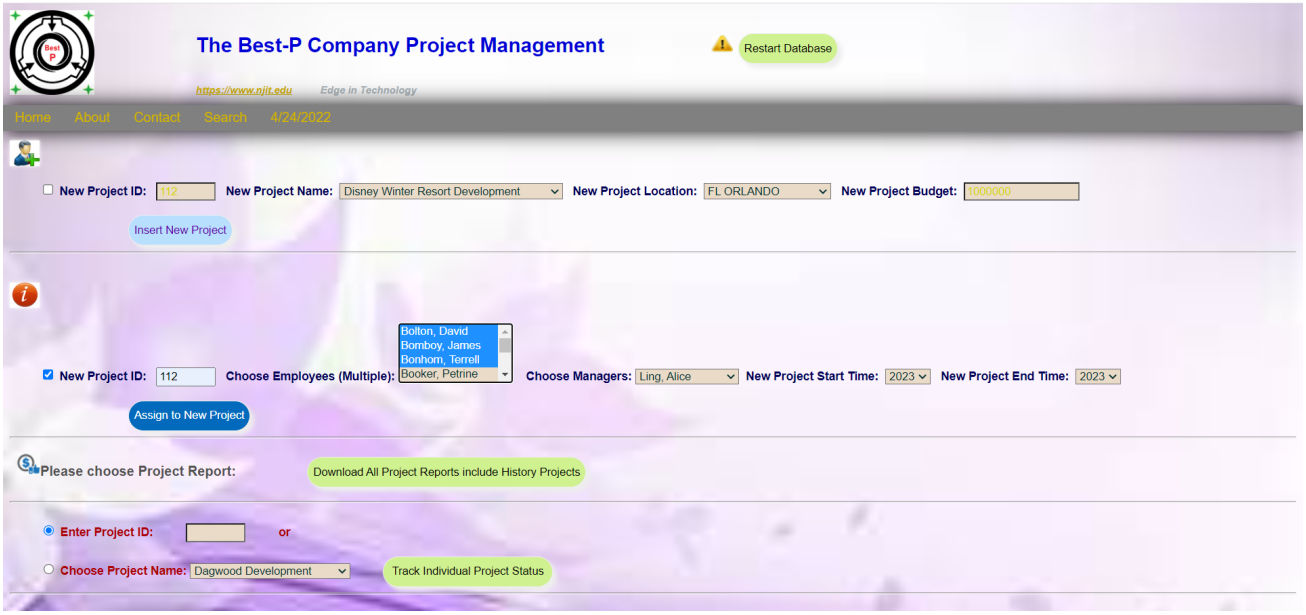
If the wages of assigned employees exceed the budget, a warning alert message will popup. Click on ‘OK’ to try again.



If the assigned employees are not available (this is b/c multiple users could update DB at the same time), a warning alert message will popup. Click on ‘OK’ to try again.



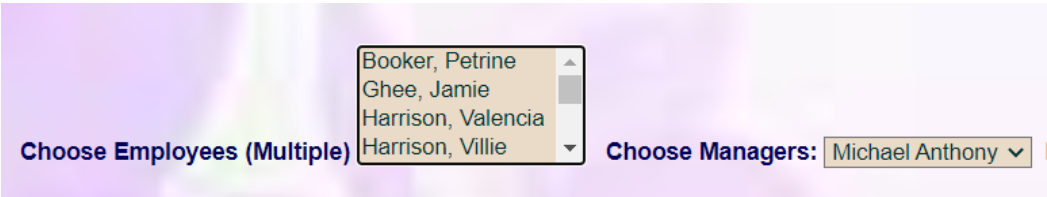
Second attempt to assign the employees to new project ID 112.



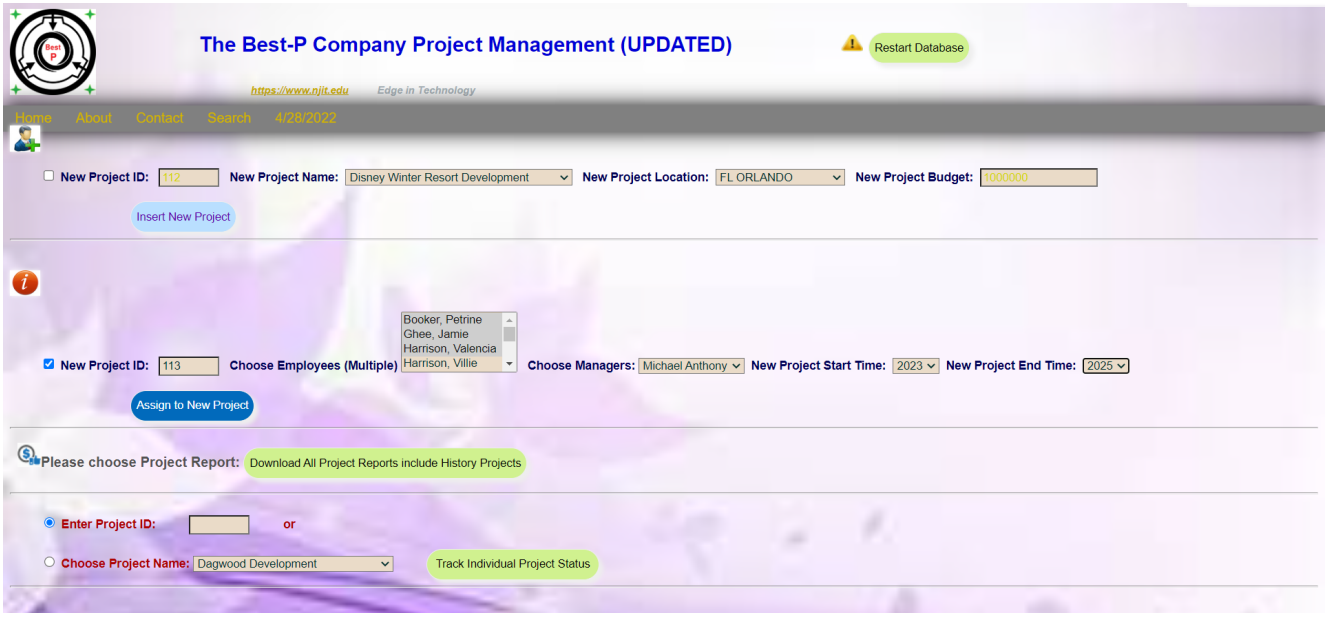
When the employees and managers are available and within budget, the request will go through by the back end DB and the web interface and database will be updated accordingly by notifying with a new title 

.

Meanwhile, Java API will refresh the GUI by removing the assigned employees (three employees starting with B) and manager (Ling, Alice) from the available list as shown. They disappear from the drop down list.



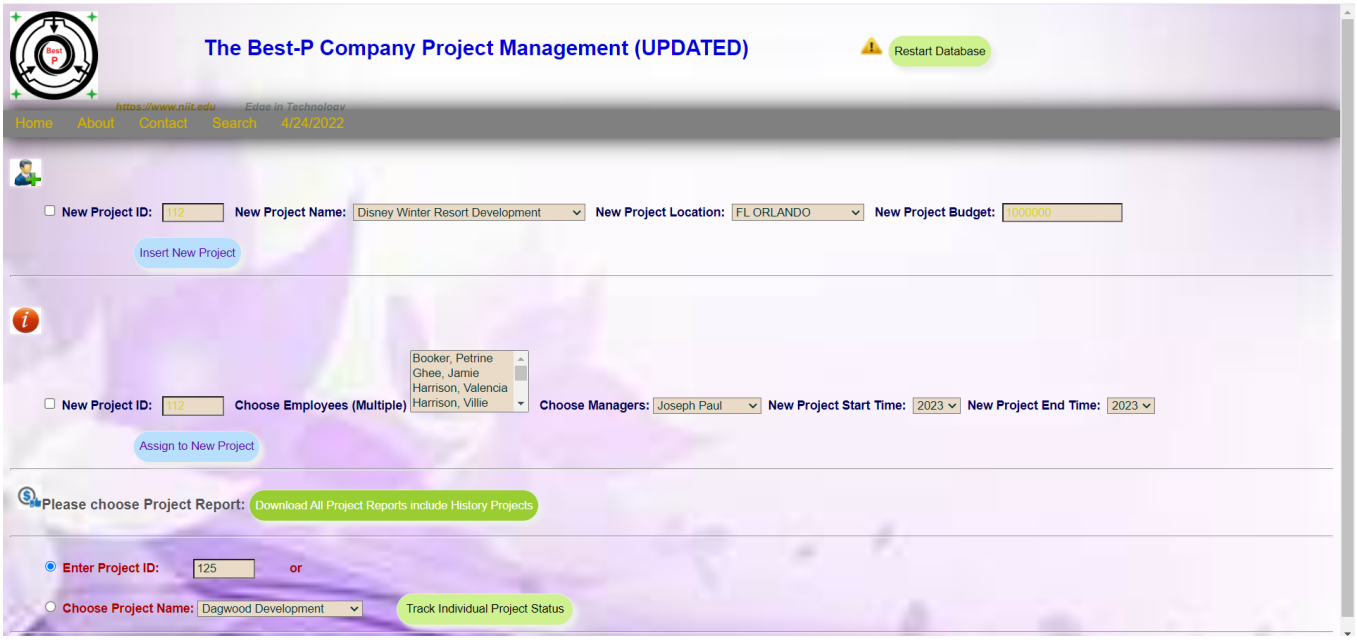
For illustration purposes, we continue to add another new project ID 113 in disney land, LA and assign resources during 2023 - 2025 the same way to it and continue to query if project status can include existing and new projects.



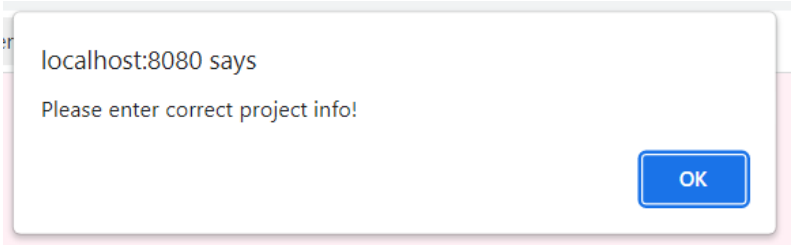
(3) Track project milestones with progress bars and status

The GUI below shows an example to track project status for project ID '105'. The user clicks on the checkbox and enters the required parameters, then submit the form by clicking

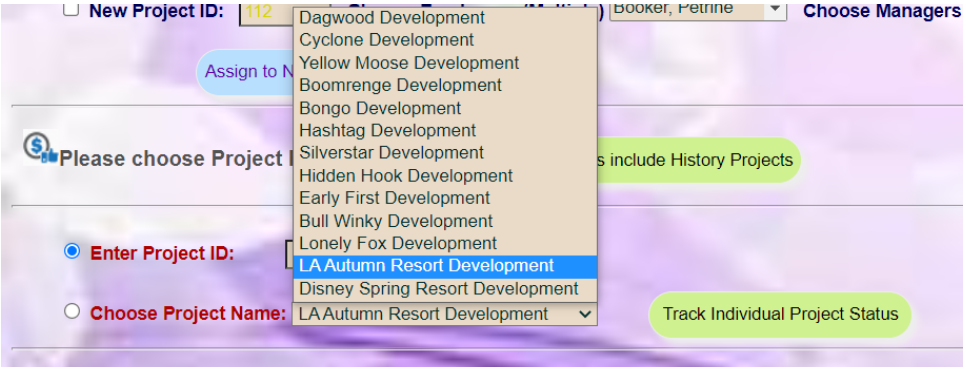
Track Individual Project Status button.



The user enters incorrect project ID '125' instead of '105'. when the project ID is not available, a popup alert message will appear. Click on 'OK' to try again.



To avoid above mistakes, the user can then choose a project from the drop down list (with two new projects included) to get project names and then click on the Track Individual Project Status button, it will make a request to the back end DB and return the detailed project progress status report.



Below is the project status for project ID 105, which is still on-going, along with its budget, manager and employees working on it, total charges and work hours as of today, along with milestone bar.

(4) Track progress statistics on all the project

The GUI below shows an example to track progress statistics on all the project by clicking 'Download All Project Reports include History Projects' button.

When user click on the Download All Project Reports include History Projects button, it will make a request to the back end DB and return all project reports with progress statistics, including the newly inserted project ID 112 in LA, CA and 113 in Disney Land, FL.

Users can refer to Appendix D sample outputs for more details.

YuQing Ding, Wei Zhang, CS631 Term Project Date: 2022-04-28

The Best-P Company Project Progress Reports

Project ID	Project Name	Location	Budget	Start Date	End Date	Project Manager	Total # Employees	Total Charge	Total Work-Hours	Milestone/Progress/
101	Dagwood Development	FL ORLANDO	13411765	2010-01-01	2022-12-31	Hartsough, Scott	13	11410164.34	298388.57	ONGOING 85%
102	Cyclone Development	NEW YORK CITY	5834793	2011-01-01	2020-12-31	Liou, Claire	9	5837527.77	152337.14	COMPLETED 100%
103	Yellow Moose Development	CA LOS ANGELES	6935365	2012-01-01	2020-12-31	Harsh, Christopher	9	6941297.49	150222.86	COMPLETED 100%
104	Boomrenge Development	DC WASHINGTON	3652095	2013-01-01	2019-12-31	Bols, James	8	3650000.00	95942.86	COMPLETED 100%
105	Bongo Development	NJ KRAYS LANDING	7261538	2014-01-01	2023-12-31	Gill, Christopher	9	5727261.14	139782.86	ONGOING 79%
106	Hashtag Development	DC WASHINGTON	5150248	2015-01-01	2020-12-31	Lin, Kevin	14	5153185.14	133548.57	COMPLETED 100%
107	Silverstar Development	IN EAST CHICAGO	2347638	2016-01-01	2019-12-31	Harvey, Ruth	8	2344223.20	50011.43	COMPLETED 100%
108	Hidden Hook Development	NEW YORK CITY	1600335	2017-01-01	2020-12-31	Lin, Michael	7	1594382.57	43800.00	COMPLETED 100%
109	Early First Development	DC WASHINGTON	1143198	2018-01-01	2021-12-31	Lindner, Scott	5	1135296.00	31285.71	COMPLETED 99%
110	Bull Winky Development	IN EAST CHICAGO	5057143	2019-01-01	2024-12-31	Hartley, John	7	2959789.49	73040.00	ONGOING 59%
111	Lonely Fox Development	CA LOS ANGELES	656408	2020-01-01	2021-12-31	Bolo, Eugene	8	664976.00	16640.00	COMPLETED 101%
112	LA Autumn Resort Development	CA LOS ANGELES	10000000	2023-01-01	2025-12-31	Joseph Paul	4	0.00	0.00	FUTURE 0%
113	Disney Spring Resort Development	FL ORLANDO	1000000	2023-01-01	2025-12-31	Maria Swabota	4	0.00	0.00	FUTURE 0%

Appendix Source Code

Appendix A Java Code

- AlertReminder.java

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.http.HttpServletResponse;

public class AlertReminder1  {
String warnMsg;
        public AlertReminder1(String warnMsg) {
                this.warnMsg=warnMsg;
        }
        public void popAlert1 (HttpServletResponse response) throws IOException {
                // @servlet writer to get results posted
                PrintWriter out = response.getWriter();
                out.println( "<script>alert(\""+warnMsg+"\")</script>");
        }
        public void popAlert1 (PrintWriter out) throws IOException {
```

```
                // @servlet writer to get results posted
                        out.println( "<script>alert(\""+warnMsg+"\")</script>");
        }
}
```

- Login.java

```
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
/**
 * Servlet implementation login
 */
@WebServlet("/login")
public class login extends HttpServlet {
        private static final long serialVersionUID = 1L;
        /**
         * @see HttpServlet#HttpServlet()
         */
        public login() {
                super();
                // TODO Auto-generated constructor stub
        }
        // the login user info. for validation
        static String useridPay, pwdPay, emailPay;
        static String useridProj, pwdProj, emailProj;

        protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
                // read login forms
                String warningMsg="No permission to sensitive database!";
                        if (request.getParameter("loginPay")!=null) {
                                useridPay = request.getParameter("useridPay");
                                pwdPay = request.getParameter("pwdPay");
                                emailPay=request.getParameter("emailPay");
                                if (pwdPay.equals("root") &&
(useridPay.equals("001")||useridPay.equals("002")||useridPay.equals("003")||useridPay.equals("004")))
                                        response.sendRedirect("MainPayroll.html");
                                else
                                new AlertReminder().popAlert(warningMsg);
                        }
                        if (request.getParameter("loginProj")!=null) {
                                useridProj = request.getParameter("useridProj");
                                pwdProj = request.getParameter("pwdProj");
                                emailProj=request.getParameter("emailProj");
                                if (pwdProj.equals("root") &&
```

```
(useridProj.equals("101")||useridPay.equals("102")||useridPay.equals("103")||useridPay.equals("104")))
                                        response.sendRedirect("MainProject.html");
                                else
                                        new AlertReminder().popAlert(warningMsg);
                        }
        }
}
```

- MainPayroll. java

```
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.Date;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.time.LocalDate;
import java.time.temporal.ChronoUnit;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/MainPayroll")
public class MainPayroll extends javax.servlet.http.HttpServlet implements javax.servlet.Servlet {
        static final long serialVersionUID = 1L;
        //this set of employee basic information is used by all applications, thus enlisted here for efficienc
        String empID, name, address, city, state, zip, SSN, email, office_code, title, dept_code, div_code, projID,
                        office_name, phone, classification, proj_name;
        Date start_date,start, end;
        Integer salary;
        Double hourly_pay;

        String errorMessage="SQL connection failed! ";
        String errorMsg1 = "Please enter correct employee -missing employee start at the date";
        String errorMsg2 = "Please enter correct Department ID!";
        String errorMsg3 = "Please enter correct Division ID!";
        String errorMsg4 = "Please enter correct employee ID!";
        String errorMsg5 = "Please enter correct IRS year";
        String errorMsgMiss = "Missing input from HTML forms. Please check again!";

        String fromGmail="eugina.ding@gmail.com";
        String toGmail="eugina.ding@gmail.com";
        static Integer reportID = 101;
        static Integer reportSeries = 1001;
```

```
	public MainPayroll() {
		super();
	}
	protected void doPost(HttpServletRequest request, HttpServletResponse response)
			throws ServletException, IOException {
		response.setContentType("text/html");
		PrintWriter out = response.getWriter();

		// @ initial messages
		ResultSet rs;
		// initial html checkbox parameters - myCheckBox is true if checked, false if
		// not checked
		boolean checkPayEmp = request.getParameter("checkPayEmp") != null;
		boolean checkPayDept = request.getParameter("checkPayDept") != null;
		boolean checkPayDiv = request.getParameter("checkPayDiv") != null;
		boolean checkComIRS = request.getParameter("checkComIRS") != null;
		boolean checkEmpIRS = request.getParameter("checkEmpIRS") != null;
		// @ create queries prepared statement for monthly paychecks.
		// note that hourly employees are contractors who has limited duration and no benefit plans!
		// note that in the following calculations use this view as a master for new start and end times
for contractors only

		String sqlMonthSal = "SELECT * FROM empview WHERE empID = ? AND start_date<=? AND
start<=? AND ?<=end";

		String sqlDeptPerm="SELECT d.dept_code,d.dept_name, e.name as manager, e.phone, d.email,
count(*) as permEmp, sum(view.hourly_pay) as permSal " +
		"FROM cs631db.empview as view, department as d, empview as e "+
		"WHERE view.dept_code= d.dept_code AND view.classification='permanent' AND
d.dept_head=e.empID AND "+
		"view.dept_code=? AND view.start_date<=? AND view.start<=? AND ?<=view.end ";

		String sqlDeptHour="SELECT d.dept_code,d.dept_name, e.name as manager, e.phone, d.email,
count(*) as hourEmp, sum(view.hourly_pay) as hourSal "+
		"FROM cs631db.empview as view, department as d, empview as e "+
		"WHERE view.dept_code= d.dept_code AND view.classification='hourly' AND
d.dept_head=e.empID AND "+
		"view.dept_code=? AND view.start_date<=? AND view.start<=? AND ?<=view.end ";

		String sqlDivPerm="SELECT d.div_code,d.div_name, e.name as manager, e.phone, d.email,
count(*) as permEmp, sum(view.hourly_pay) as permSal "+
		"FROM cs631db.empview as view, division as d, empview as e "+
		"WHERE view.div_code= d.div_code AND view.classification='permanent' AND
d.div_head=e.empID AND "+
		"view.div_code=? AND view.start_date<=? AND view.start<=? AND ?<=view.end ";

		String sqlDivHour="SELECT d.div_code,d.div_name, e.name as manager, e.phone,
d.email,count(*) as hourEmp, sum(view.hourly_pay) as hourSal  "+
		"FROM cs631db.empview as view, division as d, empview as e "+
		"WHERE view.div_code= d.div_code AND view.classification='hourly' AND d.div_head=e.empID
```

```
AND "+
                    "view.div_code=? AND view.start_date<=? AND view.start<=? AND ?<=view.end ";

                    // @ create queries prepared statement for annual IRS, entire year, first half or second half of
the year.
                    // note that 3 cases depending on date assigned to project (project is 10 yrs, one can start late
and leave early)

                    String sqlEmpWholeYear = "SELECT * FROM empview WHERE empID = ? AND start_date<=?
AND start<=? AND ?<=end";
                    //late start, so yearend-start, start_date<= yearend '2020-12-31' AND
start>=yearstart'2020-01-01' AND start<= yearend '2020-12-31'
                    String sqlEmpPartYear1 = "SELECT *, hourly_pay*DATEDIFF(?, start)/7*40 as income FROM
empview WHERE empID = ? AND start_date<=? AND start>=? AND start<=?";
                    //early leave, so end-yearstart, start_date<=yearend'2020-12-31' AND
end>=yearstart'2020-01-01' AND end<=yearend'2020-12-31'
                    String sqlEmpPartYear2 = "SELECT *, hourly_pay*DATEDIFF(end, ?)/7*40 as income FROM
empview WHERE empID = ? AND start_date<=? AND end>=? AND end<=?";


                    String sqlComIRS = "SELECT * FROM companyirs ORDER BY IRSyear, empID ASC";
                    String sqlComIRS1 = "SELECT * FROM companyirs WHERE IRSyear>=? AND IRSyear<=? ORDER BY
IRSyear, empID ASC";

                    // @ JDBC connection to SQL - create database and table
                    try {

                              ConnectionReset con=new ConnectionReset();
                              // this is done in project                    con.resetDB();
                              //initialize employee view for entire company
                              con.iniPayroll();
                              //sqlMonthSal = "SELECT * FROM empview WHERE empID = ? AND start_date<=? AND
start<=? AND ?<=end";

                              // @servlet writer to get results posted

                              // mySubmit button is true if checked, false if not checked
                              if (request.getParameter("payEmp") != null) {
                                        // myCheckBox is true if checked, false if not checked
                                        if (checkPayEmp == true) {
                                                  PreparedStatement pstmt =
con.connection.prepareStatement(sqlMonthSal);
                                                  empID = request.getParameter("empID");
                                                  String monthEmp = request.getParameter("monthEmp");
                                                  String yearEmp = request.getParameter("yearEmp");
                                                  // run sql query to get employee ID and print paycheck within the
project limit
                                                  // the charge to a project shall be between the live span of the
project
                                                  pstmt.setString(1, empID);
                                                  pstmt.setString(2, yearEmp+"-"+monthEmp+"-01");
```

```
                                        pstmt.setString(3, yearEmp+"-"+monthEmp+"-01");
                                        pstmt.setString(4, yearEmp+"-"+monthEmp+"-01");

                                        rs = pstmt.executeQuery();
                                        //generate monthly pay check for employee
                                        paycheckEmp(out, rs, monthEmp, yearEmp);
                                        pstmt.close();
                                        } else {
                                                System.out.println(errorMsgMiss);
                                        }

                        } else if (request.getParameter("payDept") != null ) {
                                if (checkPayDept == true) {
                                        dept_code = request.getParameter("deptID");
                                        String monthDept = request.getParameter("monthDept");
                                        String yearDept = request.getParameter("yearDept");

                                        PreparedStatement pstmt1 =
con.connection.prepareStatement(sqlDeptPerm);
                                        PreparedStatement pstmt2 =
con.connection.prepareStatement(sqlDeptHour);
                                        pstmt1.setString(1, dept_code);
                                        pstmt1.setString(2, yearDept+"-"+monthDept+"-01");
                                        pstmt1.setString(3, yearDept+"-"+monthDept+"-01");
                                        pstmt1.setString(4, yearDept+"-"+monthDept+"-01");
                                        ResultSet rs1=pstmt1.executeQuery();

                                        pstmt2.setString(1, dept_code);
                                        pstmt2.setString(2, yearDept+"-"+monthDept+"-01");
                                        pstmt2.setString(3, yearDept+"-"+monthDept+"-01");
                                        pstmt2.setString(4, yearDept+"-"+monthDept+"-01");
                                        ResultSet rs2=pstmt2.executeQuery();

                                        costDeptDiv(out, dept_code, "Department", monthDept, yearDept,
rs1, rs2);
                                        pstmt1.close();
                                        }else {
                                        System.out.println(errorMsgMiss);
                                }
                        } else if (request.getParameter("payDiv") != null) {
                                if (checkPayDiv == true) {
                                        div_code = request.getParameter("divID");
                                        String monthDiv = request.getParameter("monthDiv");
                                        String yearDiv = request.getParameter("yearDiv");
                                        PreparedStatement pstmt1 =
con.connection.prepareStatement(sqlDivPerm);
                                        PreparedStatement pstmt2 =
con.connection.prepareStatement(sqlDivHour);
                                        pstmt1.setString(1, div_code);
                                        pstmt1.setString(2, yearDiv+"-"+monthDiv+"-01");
                                        pstmt1.setString(3, yearDiv+"-"+monthDiv+"-01");
```

```
                                        pstmt1.setString(4, yearDiv+"-"+monthDiv+"-01");
                                        ResultSet rs1=pstmt1.executeQuery();

                                        pstmt2.setString(1, div_code);
                                        pstmt2.setString(2, yearDiv+"-"+monthDiv+"-01");
                                        pstmt2.setString(3, yearDiv+"-"+monthDiv+"-01");
                                        pstmt2.setString(4, yearDiv+"-"+monthDiv+"-01");
                                        ResultSet rs2=pstmt2.executeQuery();

                                        costDeptDiv(out, div_code, "Division", monthDiv, yearDiv, rs1, rs2);
                                        pstmt1.close();
                                        pstmt2.close();

                                }else {
                                        System.out.println(errorMsgMiss);
                                }

                        } else if (request.getParameter("comIRS") != null) {
                                if (checkComIRS == true) {

                                        //create the most updated IRS report for the entire company
                                        con.companyIRS();
                                        //generate the most updated IRS report for the entire company
                                        //sqlComIRS1 = "SELECT * FROM companyirs WHERE IRSyear>=? AND
IRSyear<=? ORDER BY IRSyear, empID ASC"

if(request.getParameter("startYr")!=null&&request.getParameter("endYr")!=null) {
                                        Integer startYr = Integer.parseInt(request.getParameter("startYr"));
                                        Integer endYr = Integer.parseInt(request.getParameter("endYr"));

                                        //send email out to user
                                        //new SendEmails(          fromGmail, toGmail,"IRS Reports Ready to
download").send();
                                        PreparedStatement
pstmt=con.connection.prepareStatement(sqlComIRS1);
                                        pstmt.setInt(1, startYr);
                                        pstmt.setInt(2, endYr);
                                        rs=pstmt.executeQuery();
                                        printIRSreport(out, rs);
                                        }
                                        //sent IRS report out to user
                                } else {
                                        System.out.println(errorMsgMiss);
                                }
                        } else if (request.getParameter("empIRS") != null) {
                                if (checkEmpIRS == true) {
                                        String empIDIRS=request.getParameter("empIDIRS");
                                        String yearIRS=request.getParameter("yearEmpIRS");

                                        // sqlEmpWholeYear = "SELECT * FROM empview WHERE empID = ?
AND start_date<=? AND start<=? AND ?<=end";
```

```
                                        //late start, so yearend-start, start_date<= yearend '2020-12-31' AND
start>=yearstart'2020-01-01' AND start<= yearend '2020-12-31'
                                        // sqlEmpPartYear1 = "SELECT *, hourly_pay*DATEDIFF(?, start)/7*40
as income FROM empview WHERE empID = ? AND start_date<=? AND start>=? AND start<=?";

                                        //early leave, so end-yearstart, start_date<=yearend'2020-12-31'
AND end>=yearstart'2020-01-01' AND end<=yearend'2020-12-31'
                                        // sqlEmpPartYear2 = "SELECT *, hourly_pay*DATEDIFF(end, ?)/7*40
as income FROM empview WHERE empID = ? AND start_date<=? AND end>=? AND end<=?";

                                        //check if whole year
                                        PreparedStatement
pstmtEmpIRS=con.connection.prepareStatement(sqlEmpWholeYear);

                                        pstmtEmpIRS.setString(1, empIDIRS);
                                        pstmtEmpIRS.setString(2, yearIRS+"-12-31");
                                        pstmtEmpIRS.setString(3, yearIRS+"-01-01");
                                        pstmtEmpIRS.setString(4, yearIRS+"-12-31");

                                        //check if late start
                                        PreparedStatement
pstmtEmpIRS1=con.connection.prepareStatement(sqlEmpPartYear1);

                                        pstmtEmpIRS1.setString(1, empIDIRS);
                                        pstmtEmpIRS1.setString(2, yearIRS+"-12-31");
                                        pstmtEmpIRS1.setString(3, yearIRS+"-01-01");
                                        pstmtEmpIRS1.setString(4, yearIRS+"-12-31");

                                        //check if early leave
                                        PreparedStatement
pstmtEmpIRS2=con.connection.prepareStatement(sqlEmpPartYear2);

                                        pstmtEmpIRS2.setString(1, empIDIRS);
                                        pstmtEmpIRS2.setString(2, yearIRS+"-12-31");
                                        pstmtEmpIRS2.setString(3, yearIRS+"-01-01");
                                        pstmtEmpIRS2.setString(4, yearIRS+"-12-31");

                                        //check if whole year
                                        if(pstmtEmpIRS.execute()) {
                                                rs=pstmtEmpIRS.executeQuery();
                                        //type=0: whole year to generate yearly IRS report for employee
                                                empIRSreport(out, yearIRS, rs, 0);
                                        } else if (pstmtEmpIRS1.execute()) {
                                                //check if late start
                                                rs=pstmtEmpIRS1.executeQuery();
                                        //type=1: late start to generate yearly IRS report for employee
                                        empIRSreport(out, yearIRS, rs, 1);

                                        } else if (pstmtEmpIRS2.execute()) {
                                                //check if early leave
```

```
                                                rs=pstmtEmpIRS1.executeQuery();
                                                //type=2: early leave to generate yearly IRS report for
employee
                                                empIRSreport(out, yearIRS, rs, 2);

                                        } else {
                                                System.out.println(errorMsg4);
                                        }
                                        pstmtEmpIRS.close();
                                        pstmtEmpIRS2.close();
                                }else {
                                        System.out.println(errorMsgMiss);
                                }
                        }
                } catch (Exception ex) {
                        System.out.println(errorMessage + ex);
                        System.exit(0);

                        // @servlet response

                }
                out.close();
        }
        public void paycheckEmp(PrintWriter out, ResultSet rs, String monthEmp, String yearEmp) throws
SQLException {
                // query has output not null
                if (rs.next()) {
                        // empID = rs.getString(1);
                        name = rs.getString(2);
                        title = rs.getString(3);
                        start_date = rs.getDate(4);
                        address = rs.getString(5);
                        city = rs.getString(6);
                        state = rs.getString(7);
                        zip = rs.getString(8);
                        SSN = rs.getString(9);
                        email = rs.getString(10);
                        office_code = rs.getString(11);
                        phone = rs.getString(12);
                        office_name = rs.getString(13);
                        dept_code = rs.getString(14);
                        div_code = rs.getString(15);
                        projID = rs.getString(16);
                        salary = rs.getInt(19);
                        classification = rs.getString(20);
                        hourly_pay = rs.getDouble(21);
                        proj_name = rs.getString(22);
                }else System.out.println(errorMsgMiss);
                out.println("<html>");
                out.println("<head> <meta charset=\"utf-8\" />");
                out.println("<title>The Best-P Company Payroll Pay Check</title>");
```

```
                    out.println("<link rel=\"stylesheet\" href=\"report.css\">");
                    out.println("<script type=\"text/javascript\"</script>");
                    out.println("<script src=\"https://code.jquery.com/jquery-1.12.4.min.js\"></script>");
                    out.println("</head> <body><div class=\"invoice-box\">");
                    out.println("<table cellpadding=\"0\" cellspacing=\"0\">");
                    out.println("<tr class=\"top\"><td colspan=\"2\"><table>");
                    out.println("<tr><td class=\"title\"><img src=\"adminLogo.jpg\" class=\"image1\" />");
                    out.println("<td>Pay Check #: " + reportID++);
                    out.println("<br/>Created: <a href=\"#\"><script>document.write(new
Date().toLocaleDateString()); </script>");
                    out.println("</a></p><br/>Time Period: " + MonthNumber.matchName(monthEmp) + " " +
yearEmp);
                    out.println("</td></tr></table></td></tr>");
                    out.println("<tr class=\"information\"><td colspan=\"2\"><table> <tr><td>");
                    out.println("The Best-P Company<br/>7545 MLK J. Blvd<br/>Newark, NJ 07102<br/>USA
Headquarter</td><td>");
                    out.println("Employee ID-0" + empID);
                    out.println("<br/>Name: " + name);
                    out.println("<br/>Office Phone: " + phone);
                    out.println("<br/>Email: " + email);
                    out.println("</td></tr></table></td></tr>");
                    out.println("<tr class=\"heading\"><td>Payment Method</td><td>Check Series #</td></tr>");
                    out.println("<tr class=\"details\"><td>Check</td><td>");
                    out.println("" + reportSeries++ + " </td></tr>");
                    out.println("<tr class=\"heading\"><td>Earning/Tax Deduction</td><td>Amount</td>");
                    out.println("</tr><tr class=\"item\"><td>Gross Pay</td><td>");
                    // calculate the salary
                    Double monthSal = hourly_pay * 160;
                    Double taxFed = monthSal * 0.1;
                    Double taxSta = monthSal * 0.05;
                    Double benefit = monthSal * 0.03;
                    if (classification.equalsIgnoreCase("hourly"))
                            benefit = 0.00;
                    Double netPay = monthSal - taxFed - taxSta - benefit;

                    out.println("$" + String.format("%.2f", monthSal) + "</td></tr>");
                    out.println("<tr class=\"item\"><td>Federal Tax (10%)</td><td>");
                    out.println("$" + String.format("%.2f", taxFed) + "</td></tr>");
                    out.println("<tr class=\"item\"><td>State Tax (5%)</td><td>");
                    out.println("$" + String.format("%.2f", taxSta) + "</td></tr>");
                    out.println("<tr class=\"item last\"><td>Benefit Plan (3%)</td><td>");
                    if (classification.equalsIgnoreCase("hourly")) {
                            out.println("No benefit Plan for Hourly");
                    } else {
                            out.println("$" + String.format("%.2f", benefit) + "</td></tr>");
                    }
                    out.println("<tr class=\"heading\"><td>Net Pay</td><td>");
                    out.println("$" + String.format("%.2f", netPay) + "</td></tr>");

                    out.println("<tr class=\"heading\"><td>");
                    out.println("<tab3><tab3><button onclick=\"load()\">Return to Main Page </button>");
```

```
                out.println("<script>    function load() {");
                out.println("location.assign(\"http://localhost:8080/cs631db/MainPayroll.html\");");
                out.println("} </script>       </td></tr></table></div></body></html>");
        }

        public void costDeptDiv(PrintWriter out, String code, String type, String month, String year, ResultSet
rsPerm, ResultSet rsHr) throws SQLException {

                String deptdivName = null;
                String headName = null;
                String phone = null;
                String email = null;
                Integer permEmp = null;
                Integer hrEmp = null;
                Double permSal = 0.0;
                Double hrSal=0.0;

                if (rsPerm.next() && rsHr.next()) {

                deptdivName= rsPerm.getString(2);
                headName=rsPerm.getString(3);
                phone=rsPerm.getString(4);
                email=rsPerm.getString(5);
                permEmp=rsPerm.getInt(6);
                hrEmp =rsHr.getInt(6);
                permSal=rsPerm.getDouble(7);
                hrSal=rsHr.getDouble(7);

                } else System.out.println(errorMsgMiss);

                out.println("<html><head><meta charset=\"utf-8\" />");
                out.println("<title>The Best-P Company Payroll Division/Department Cost Report</title>");
                out.println("<link rel=\"stylesheet\" href=\"report.css\">");
                out.println("<script type=\"text/javascript\"</script><script
src=\"https://code.jquery.com/jquery-1.12.4.min.js\"></script>");
                out.println("</head><body>           <div class=\"invoice-box\">");
                out.println("<table cellpadding=\"0\" cellspacing=\"0\">         <tr class=\"top\"><td
colspan=\"2\">");
                out.println("<table>          <tr><td class=\"title\"><img src=\"adminLogo.jpg\" class=\"image1\"
/></td>");
                out.println("<td>Cost Report #: "+ reportSeries++ +"<br />");
                out.println("Created: <a href=\"#\"><script>document.write(new Date().toLocaleDateString());
</script></a></p><br />");
                out.println("Time Period: " + MonthNumber.matchName(month) + "  " + year);
                out.println("</td></tr></table></td></tr><tr class=\"information\"><td
colspan=\"2\"><table><tr><td>");
                out.println("The Best-P Company<br/>7545 MLK J. Blvd<br/>Newark, NJ 07102<br/>USA
Headquarter</td><td>");
                out.println(type+" ID: "+code+"<br/>");
                out.println("Name: "+ deptdivName+"<br/>");
                out.println("Head: "+headName+"<br/>");
```

```
                    out.println("Head's Email: "+email+"</td></tr>");
                    out.println("<tr class=\"heading\"><td>Type of Employees</td><td> No. of Employees </td>");
                    out.println("</tr><tr class=\"details\"><td>Permanent / Hourly</td>");
                    out.println("<td>"+permEmp+" / "+hrEmp+"</td></tr>");

                    //calculate operating cost for each department or division
                    Integer totalSal=(int) (permSal+hrSal) * 160;
                    Integer totalSal1=(int) (permSal * 160);
                    Integer totalSal2=(int) (hrSal * 160);
                    Integer taxFed = (int) (totalSal * 0.1);
                    Integer taxSta = (int) (totalSal * 0.05);
                    Integer benefit = (int) (permSal * 160 * 0.03);
                    Integer netCost=totalSal-taxFed-taxSta-benefit;
                    out.println("<tr class=\"heading\"><td>Cost Item</td><td>Amount</td></tr><tr
class=\"item\">");
                    out.println("<td>Gross Pay</td><td>"+totalSal+"<br/>("+totalSal1+" / "+totalSal2+")</td>");
                    out.println("</tr><tr class=\"item\"><td>Federal Tax (10%)</td>");
                    out.println("<td>" + taxFed +"</td></tr>");
                    out.println("<tr class=\"item\"><td>State Tax (5%)</td>");
                    out.println("<td>"+ taxSta +"</td></tr>");
                    out.println("<tr class=\"item last\"><td>Benefit Plan (3%)</td>");
                    out.println("<td>"+ benefit +"</td></tr>");
                    out.println("<tr class=\"heading\"><td>Total Net Cost (Salary Only)</td>");
                    out.println("<td>"+netCost+"</td></tr></table>");
                    out.println("<tab3><tab3><button onclick=\"load()\">Return to Main Page </button>");
                    out.println("<script>    function load() {");
                    out.println("location.assign(\"http://localhost:8080/cs631db/MainPayroll.html\");");
                    out.println("} </script>      </td></tr></table></div></body></html>");
          }
          public void empIRSreport(PrintWriter out, String yearIRS, ResultSet rs, int type) throws SQLException {

                    int empIRSIncome = 0;
                    if(rs.next()) {

                    empID=rs.getString(1);
                    name=rs.getString(2);
                    address=rs.getString(5);
                    city=rs.getString(6);
                    state = rs.getString(7);
                    zip = rs.getString(8);
                    SSN = rs.getString(9);
                    start=rs.getDate(17);
                    end=rs.getDate(18);
                    salary = rs.getInt(19);
                    classification = rs.getString(20);
                    hourly_pay = rs.getDouble(21);
                    } else System.out.println(errorMsgMiss);
          //compare the days for partial year
                    try {
                              SimpleDateFormat datef = new SimpleDateFormat("yyyy-MM-dd");
               java.util.Date javaStartDate = datef.parse(yearIRS+"-01-01");
```

```
                java.util.Date javaEndDate = datef.parse(yearIRS+"-12-31");
                java.util.Date sqlStartDate = new java.util.Date(start.getTime());
                java.util.Date sqlEndDate = new java.util.Date(end.getTime());
                long effectiveDays;
                        switch (type) {
                        case 0:
                                //type=0: whole year
                                empIRSIncome=salary;
                                break;
                        case 1:
                                //type=1: late start
                                effectiveDays=((javaEndDate.getTime()-sqlStartDate.getTime())/ (1000 * 60 * 60 * 24));
                                 //convert to income
                                 empIRSIncome=(int) (hourly_pay* effectiveDays/7*40);
                                break;
                        case 2:
                                //type=2: early leave
                                effectiveDays=((sqlEndDate.getTime()-javaStartDate.getTime())/ (1000 * 60 * 60 * 24));
                                 //convert to income
                                 empIRSIncome=(int) (hourly_pay* effectiveDays/7*40);
                                break;
                        }
                        } catch (ParseException ex) {
                                System.out.println(errorMsg5 + ex.getMessage());
                        } catch (Exception e) {
                                System.out.println(errorMsg5 + e);
                        }
                        // Federal=15%, State = 5%, 3% benefit, plus 6.2% for Social Security Tax and 1.45% for Medical
                        Integer taxFed = (int) (empIRSIncome * 0.1);
                        Integer taxSta = (int) (empIRSIncome * 0.05);
//                      Integer benefit = (int) (empIRSIncome * 0.03);
                        Integer taxSS = (int) (empIRSIncome * 0.062);
                        Integer medWage=empIRSIncome-taxFed-taxSS;
                        Integer taxMedi = (int) (empIRSIncome * 0.0145);
                        Integer netWage=empIRSIncome-taxFed-taxSS-taxMedi;

                        out.println("<html><head><meta charset=\"utf-8\" />");
                        out.println("<title>The Best-P Company IRS Report</title>");
                        out.println("<link rel=\"stylesheet\" href=\"report.css\">");
                        out.println("<script type=\"text/javascript\"</script><script
src=\"https://code.jquery.com/jquery-1.12.4.min.js\"></script>");
                        out.println("</head><body>           <div class=\"invoice-box\">");
                        out.println("<table border=\"1\" cellpadding=\"0\" cellspacing=\"0\"><tr>");
                        out.println("<td><font color=RED> Copy B-To Be Filed with Employee's<br>");
                        out.println("FEDERAL TAX Return</font></td>");
                        out.println("<td>OMB #: 1234-5678<br> Created: <a href=\"#\">");
                        out.println("<script>document.write(new Date().toLocaleDateString());</script></a>");
                        out.println("</p><h1>"+yearIRS+"</h1></td></tr>");
                        out.println("<tr class=\"heading\">");
                        out.println("<td>a. Employee's Soc. Sec. Number<br> <tab1>");
                        out.println( ""+SSN+"</td>");
```

```
                    out.println("<td>b. Employer's ID Number<br> <tab1> 36-01234"+empID+"</td></tr>");
                    out.println("<tr class=\"details\">");
                    out.println("<td>1. Wages, tips, other comp.<br> <tab1>$"+netWage+"</td>");
                    out.println("<td>2. Federal income tax. withheld<br> <tab1>$"+ taxFed+"</td>");
                    out.println("</tr>");
                    out.println("<tr class=\"details\">");
                    out.println("<td>3. Social security wages<br> <tab1>$"+empIRSIncome+"</td>");
                    out.println("<td>4. Social security tax. withheld<br> <tab1>");
                    out.println("$"+taxSS+"</td>");
                    out.println("</tr>");
                    out.println("<tr class=\"details\">");
                    out.println("<td>5. Medical wages and tips<br> <tab1>$"+ medWage +"</td>");
                    out.println("<td>6. Medical tax. withheld<br> <tab1>$"+taxMedi+"</td>");
                    out.println("</tr>");
                    out.println("</table>");
                    out.println("<table border=\"1\" cellpadding=\"0\" cellspacing=\"0\">");
                    out.println("<tr class=\"information\">");
                    out.println("<td colspan=\"0\">c. Employer's name, address, and ZIP code<br />");
                    out.println("<tab2> The Best-P Company<br />");
                    out.println("<tab2> 7545 MLK J. Blvd<br />");
                    out.println("<tab2> Newark, NJ 07102 </td>");
                    out.println("</tr>");
                    out.println("<tr class=\"information\">");
                    out.println("<td colspan=\"0\">d. Employee's name, address, and ZIP code<br />");
                    out.println("<tab2>"+name+"<br />");
                    out.println("<tab2>"+address+"<br />");
                    out.println("<tab2>"+city+","+state+","+zip+"</td>");
                    out.println("</tr>");
                    out.println("</table>");
                    out.println("<table border=\"1\" cellpadding=\"0\" cellspacing=\"0\">");
                    out.println("<tr class=\"information\">");
                    out.println("<td colspan=\"4\">"+state+"<br> State");
                    out.println("<td>360-982-12345 <br> Employer's state ID");
                    out.println("</td>");
                    out.println("<td>$ "+netWage +"<br> State wages, tips, etc.");
                    out.println("</td>");
                    out.println("<td>$ "+taxSS+"<br> State income tax</td>");
                    out.println("</tr></table>");
                    out.println("<tab3><tab3><button onclick=\"load()\">Return to Main Page </button>");
                    out.println("<script>    function load() {");
                    out.println("location.assign(\"http://localhost:8080/cs631db/MainPayroll.html\");");
                    out.println("} </script></div></body></html>");
          }
          public void printIRSreport (PrintWriter out, ResultSet rs) throws SQLException  {

                    // query has output not null  >
                    out.println("<html><head><link rel=\"stylesheet\" href=\"./CS631.css\">");
                    out.println("<script type=\"text/javascript\"</script><script
src=\"https://code.jquery.com/jquery-1.12.4.min.js\"></script>");
                    out.println("<script src='js/autoresize.jquery.min.js'></script></head><body>");
                    out.println("<h4>"+"YuQing Ding, Wei Zhang, CS631 Term Project   Date: "+LocalDate.now());
```

```
                    out.println("</h4><h1><tab3><tab3> The Best-P Company IRS
Records<tab2></h1><br></head><body>");
                    //start of a table
                    out.println("<table border = \"1\" bordercolor = \"black\" bgcolor = \"lightgray\">
<tr><th>Year</th><th>Emp ID</th><th>Emp Name</th>");

out.println("<th>address</th><th>city</th><th>state</th><th>zip</th><th>SSN</th><th>Social.S.Wage</th><th
>Federal Tax</th>");
                    out.println("<th>State Tax</th><th>Social.S.Tax</th><th>Medical
Tax</th><th>State_Wage</th></tr>");

                    while (rs.next()) {
                              empID = rs.getString(1);
                              name = rs.getString(2);
                              address = rs.getString(3);
                              city = rs.getString(4);
                              state = rs.getString(5);
                              zip = rs.getString(6);
                              SSN = rs.getString(7);
                              office_code = rs.getString(8);
                              dept_code = rs.getString(9);
                              div_code = rs.getString(10);
                              classification = rs.getString(11);
                              Integer span = rs.getInt(12);
                              Integer IRSyear = rs.getInt(13);
                              //post all sorts of taxes and salaries
                              Integer SSincome = (int) rs.getDouble(14);
                              Integer FederalTax = (int)rs.getDouble(15);
                              Integer StateTax = (int)rs.getDouble(16);

                              Integer SSTax = (int)rs.getDouble(17);
                              Integer MedicalTax = (int)rs.getDouble(18);
                              Integer StateWage = (int) rs.getDouble(19);
                              out.println("<tr><th>"+IRSyear+"</th><th>"+empID+"</th><th>"+name+"</th>");
                              out.println("<th>"+ address
+"</th><th>"+city+"</th><th>"+state+"</th><th>"+zip+"</th><th>"+SSN+"</th><th>"+SSincome+"</th>");

out.println("<th>"+FederalTax+"</th><th>"+StateTax+"</th><th>"+SSTax+"</th><th>"+MedicalTax+"</th><th>"+
StateWage+"</th></tr>");
                    }
          }
}
```

- MainProject. java

```
import java.io.IOException;
import java.io.PrintWriter;
import java.math.BigInteger;
import java.sql.Date;
import java.sql.PreparedStatement;
```

```
import java.sql.ResultSet;
import java.sql.SQLException;
import java.time.LocalDate;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/MainProject")
public class MainProject extends javax.servlet.http.HttpServlet implements javax.servlet.Servlet {
	static final long serialVersionUID = 1L;

	//this set of employee basic information is used by all applications, thus enlisted here for efficienc
	String empID, name, address, city, state, zip, SSN, email, office_code, title, dept_code, div_code, projID,
	office_name, phone, classification, proj_name, proj_manager, location, start1, end1, status, mgr_name;
	Date start_date, start, end, proj_startdate, proj_enddate;
	Integer budget, salary;
	BigInteger totalEmp;
	Double hourly_pay,totalCharge,totalworkhour, prog_milestone;

	// @ initial messages
	String errorMsgCon="SQL connection failed! ";
	String errorMsgData = "Wrong Data Entry or No Data. Please try again!";
	String errorMsgMiss = "Missing input from HTML forms. Please check again!";
	String errorMsgTake = "Already in database. Please try again!";
	String errorMsg1 = "Project exists! Please try again!";
	String goodMsg1 = "Your request is accepted! New project is available now!";
	String errorMsg2 = "Not available, manager occupied already!!";
	String goodMsg2 = "Your request is accepted! Database assigned manager to project!";
	String errorMsg3 = "Not available, employee occupied already!";
	String goodMsg3 = "Your request is accepted! Database assigned employee to project!";
	String errorMsg4 = "Please enter correct project info!";
	String goodMsg4 = "Your request is accepted! Get project statistics!";
	String errorMsg5 = "too many employees assigned! Budget overflow!";
	// @ initial database for web page
	String sqlUpdateProj="SELECT * FROM (SELECT projID, proj_name FROM project WHERE projID>0) as p ORDER BY
p.projID ASC";
	String sqlUpdateMgr="SELECT * FROM (SELECT ea.empID, ep.name FROM employeeassign ea, employeepayroll ep "
			+ "WHERE ea.empID=ep.empID AND ea.projID is null AND ea.title = 'project manager') "
			+ "as e ORDER BY e.name ASC";
	String sqlUpdateEmp="SELECT * FROM (SELECT ea.empID, ep.name FROM employeeassign ea, employeepayroll ep "
			+ "WHERE ea.empID=ep.empID AND ea.projID is null AND ea.title != 'project manager') "
			+ "as e ORDER BY e.name ASC";

	public MainProject() {
		super();
	}
	protected void doGet(HttpServletRequest request, HttpServletResponse response)
			throws ServletException, IOException {
		try {
			// @ JDBC connection to SQL - create database and table
```

```
                        ConnectionReset con=new ConnectionReset();
                        prepRefreshPage(con, response);

                } catch (Exception ex) {
                        System.out.println(errorMsgCon + " sql error is "+ex);
                        // @servlet response
                        response.sendRedirect("MainProject.html");
                }
        }
        private void prepRefreshPage(ConnectionReset con, HttpServletResponse response) throws IOException {
                // @servlet writer to get newly updated lists and page refreshed
                PrintWriter out = response.getWriter();

                try {
                PreparedStatement pstmtProj = con.connection.prepareStatement(sqlUpdateProj);
                PreparedStatement pstmtMgr = con.connection.prepareStatement(sqlUpdateMgr);
                PreparedStatement pstmtEmp = con.connection.prepareStatement(sqlUpdateEmp);
                ResultSet updateEmp=pstmtEmp.executeQuery();
                ResultSet updateMgr=pstmtMgr.executeQuery();
                ResultSet updateProj=pstmtProj.executeQuery();
                //refresh the main page
                refreshPage(out, updateEmp, updateMgr, updateProj);

                } catch (Exception ex) {
                        System.out.println(errorMsgCon + " sql error is "+ex);
                        // @servlet response
                        response.sendRedirect("MainProject.html");
                }
        }
        protected void doPost(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, IOException {
                response.setContentType("text/html");
                // @servlet writer to get results posted
                PrintWriter out = response.getWriter();
                // sql query result set
                ResultSet rs;
                // initial html checkbox  - myCheckBox is true if checked, false if not
                boolean checkNewProj = request.getParameter("checkNewProj") != null;
                boolean checkAssignProj = request.getParameter("checkAssignProj") != null;
                String checkProjSta;

                // @ create queries prepared statement.

                String sqlInsertProj = "INSERT INTO project VALUES (?, ?, null, ?, ?, null, null)";
                String sqlAssignMgr = "UPDATE project SET proj_manager = ?, start =?, end = ? WHERE projID =?";
                String sqlAssignEmp = "UPDATE employeeassign SET projID = ?, proj_startdate=?, proj_enddate=? WHERE
empID = ?";
                String sqlProjStatus = "SELECT * FROM projectProgress WHERE projID=?";
                String sqlAllProj = "SELECT * FROM projectProgress";
```

```
		try {
			// @ JDBC connection to SQL - create database and table
			ConnectionReset con=new ConnectionReset();
			//	check out the four major functions, insert & assign resources to new project, quote all & each
project status
			if (request.getParameter("restart")!=null) {
				//initialize database
				//initialize SQL employee view and project view for entire company, remove wrong trials
				con.resetDB();
				con.iniProject();
				response.sendRedirect("MainProject.html");
			}
			// Submit button to insert a new proj is true if checked, false if not checked
			if (request.getParameter("projNew") != null) {
				// myCheckBox is true if checked, false if not checked
				if (checkNewProj == true) {
					//recalculate empview2 per most updated entity tables
					con.iniProject();
					PreparedStatement pstmt = con.connection.prepareStatement(sqlInsertProj);
					//sqlInsertProj = "INSERT INTO project VALUES (?, ?, null, ?, ?, null, null)";

					try {
						projID = request.getParameter("projNewID");
						proj_name = request.getParameter("projNewName");
						location = request.getParameter("projNewLoc");
						budget = Integer.parseInt(request.getParameter("projNewbgt"));

						// run sql query to insert new project ID
						pstmt.setString(1, projID);
						pstmt.setString(2, proj_name);
						pstmt.setString(3, location);
						pstmt.setInt(4, budget);

						Integer i= pstmt.executeUpdate();
						//check if the project is inserted to database
						if (i==1) new AlertReminder1(goodMsg1).popAlert1(response);
						else if (i==0) {
							new AlertReminder1(errorMsg1).popAlert1(response);
							throw new Exception ("errorMsg1");
						}

					} catch (NumberFormatException e) {
						System.out.println(errorMsgMiss);
						System.out.println(e.getMessage());
					} catch (Exception e) {
						System.out.println(errorMsgTake);
						System.out.println(e.getMessage());
					}
				pstmt.close();
				} else {
					System.out.println(errorMsgMiss);
```

```
                                        }
                                        //refresh the main page
                                        prepRefreshPage(con, response);
                                } else if (request.getParameter("projAssign")!=null) {
                                                // Submit button to assign resources to proj is true if checked, false if not checked
                                                if (checkAssignProj==true) {
                                                        //recalculate empview2 per most updated entity tables
                                                        con.iniProject();

                                                PreparedStatement pstmt1 = con.connection.prepareStatement(sqlAssignMgr);
                                                //sqlAssignMgr = "UPDATE project SET proj_manager = ?, start =?, end = ? WHERE
projID =?";

                                                String projIDr = request.getParameter("projNewIDr");
                                                proj_manager = request.getParameter("availMgr");
                                                start1 = request.getParameter("projStart")+"-01-01";
                                                end1 = request.getParameter("projEnd")+"-12-31";

                                                start = new StringToDate(start1).getSqlDate();
                                                end = new StringToDate(end1).getSqlDate();

                                                pstmt1.setString(1, proj_manager);
                                                pstmt1.setDate(2, start);
                                                pstmt1.setDate(3, end);
                                                pstmt1.setString(4, projIDr);
                                                //insert manager
                                                Integer j=pstmt1.executeUpdate();
                                                //check if the project manager is inserted to database
                                                if (j==1) System.out.println(goodMsg2);
                                                else if (j==0) {
                                                        new AlertReminder1(errorMsg2).popAlert1(response);
                                                }
                                                //insert employees
                                                PreparedStatement pstmt2 = con.connection.prepareStatement(sqlAssignEmp);
                                                //sqlAssignEmp = "UPDATE employeeassign SET projID = ?, proj_startdate=?,
proj_enddate=? WHERE empID = ?";
                                                //insert employee as a manager is inserted to database
                                                pstmt2.setString(1, projIDr);
                                                pstmt2.setDate(2, start);
                                                pstmt2.setDate(3, end);
                                                pstmt2.setString(4, proj_manager);
                                                Integer i=pstmt2.executeUpdate();
                                                //check if the employee as a manager is inserted to database
                                                if (i==1) System.out.println(goodMsg3);
                                                else if (i==0) {
                                                        System.out.println(errorMsg3);
                                                }
                                                String[] availEmp=request.getParameterValues("availEmp");
                                        //insert all empIDs newly assigned to identified project
                                                if (availEmp!=null) {
                                                        if (availEmp.length>3) new
```

```
AlertReminder1(errorMsg5).popAlert1(response);
                                                    else {
                                                            for (String emp:availEmp) {
                                                                    pstmt2.setString(1, projIDr);
                                                                    pstmt2.setDate(2, start);
                                                                    pstmt2.setDate(3, end);
                                                                    pstmt2.setString(4, emp);
                                                                    Integer k=pstmt2.executeUpdate();
                                                                    //check if the employee as employee is inserted to
database
                                                                    if (k==1) System.out.println(goodMsg3);
                                                                    else if (k==0) {
                                                                            new
AlertReminder1(errorMsg3).popAlert1(response);
                                                                    }
                                                            }
                                                    }
                                            }
                                            //refresh the main page
                                            prepRefreshPage(con, response);
                                            pstmt1.close();
                                            pstmt2.close();

                                            }else  {
                                                    System.out.println(errorMsgMiss);
                                            }

                                    } else if (request.getParameter("allProj")!=null) {
                                            // Submit button to get status of all proj is true if checked, false if not checked
                                            //recalculate status of all projects per most updated entity tables
                                            con.staProject();
                                            if(con.stmt.execute(sqlAllProj)) {
                                                    // sql has results
                                                            rs=con.stmt.executeQuery(sqlAllProj);
                                                            printAllProj(out, rs);
                                            } else System.out.println(errorMsgMiss);

                                            //refresh the main page
                                            //prepRefreshPage(con, response);

                                    } else if (request.getParameter("projSta")!=null) {
                                            // Submit button to get status of one proj is true if checked, false if not checked
                                                    //recalculate status of all projects per most updated entity tables
                                                    con.staProject();
                                                    //sqlProjStatus = "SELECT * FROM projectProgress WHERE projID=?";
                                                    PreparedStatement
pstmt3=con.connection.prepareStatement(sqlProjStatus);
                                            if (request.getParameter("checkProjSta")!=null) {
                                                    if(request.getParameter("checkProjSta").equals("1")) {
                                                            projID=request.getParameter("projID");
                                                            } else if (request.getParameter("checkProjSta").equals("2")) {
```

```
                                                projID=request.getParameter("projName");
                                                }

                                        pstmt3.setString(1, projID);
                                        rs=pstmt3.executeQuery();
                                        printAllProj(out, rs);
                                                pstmt3.close();
                                    } else System.out.println(errorMsg4);
                                            //refresh the main page
                                            //prepRefreshPage(con, response);

                                }
                                }catch (Exception ex) {
                                        System.out.println(errorMsgCon + " sql error is "+ex);

                                        // @servlet response
                                        response.sendRedirect("MainProject.html");
                                }
                out.close();
        }
        public void printAllProj (PrintWriter out, ResultSet rs)  {
                // query has output not null
                out.println("<html><head><link rel=\"stylesheet\" href=\"./CS631.css\">");
                out.println("<script type=\"text/javascript\"</script><script
src=\"https://code.jquery.com/jquery-1.12.4.min.js\"></script>");
                out.println("<script src='js/autoresize.jquery.min.js'></script></head><body>");
                out.println("<h4>"+"YuQing Ding, Wei Zhang, CS631 Term Project   Date: "+LocalDate.now());
                out.println("</h4><h1><tab3><tab3> The Best-P Company Project Progress
Reports<tab2></h1><br></head><body>");
                //start of a table
                out.println("<table> <tr><th>Project ID</th><th>Project Name</th><th>Location</th><th>Budget</th>");
                out.println("<th>Start Date</th><th>End Date</th><th>Project Manager</th><th>Total # Employees</th>");
                out.println("<th>Total Charge</th><th>Total Work-Hours</th><th>Milestone/Progress/</th></tr>");

                try {
                        while (rs.next()) {
                                projID = rs.getString(1);
                                proj_name = rs.getString(2);
//                              proj_manager = rs.getString(3);
                                location= rs.getString(4);
                                budget=rs.getInt(5);
                                proj_startdate = rs.getDate(6);
                                proj_enddate = rs.getDate(7);
//                              span = rs.getString(8);
                                status = rs.getString(9);
                                mgr_name = rs.getString(10);
                                totalEmp = BigInteger.valueOf(rs.getLong(11));
                                totalCharge = rs.getDouble(12);
                                totalworkhour = rs.getDouble(13);
                                prog_milestone = rs.getDouble(14);
                                int mstone=(int)Math.round(prog_milestone*100.00);
```

```
                                        out.println("<tr><td>"+projID+"</td>");
                                        out.println("<td>"+proj_name+"</td>");
                                        out.println("<td>"+location+"</td>");
                                        out.println("<td>"+budget+"</td>");

                                        out.println("<td>"+proj_startdate+"</td>");
                                        out.println("<td>"+proj_enddate+"</td>");
                                        out.println("<td>"+mgr_name+"</td>");
                                        out.println("<td>"+totalEmp+"</td>");

                                        out.println("<td>"+String.format("%.2f", totalCharge)+"</td>");
                                        out.println("<td>"+String.format("%.2f", totalworkhour)+"</td>");
                                        out.println("<td>"+status+"<progress id=\"progressBar\" max=\"100\" value=\""+mstone+"\"
> </progress>");
                                        out.println("</td></tr>");
                                }
                        out.println("</table><tab3><tab3><button onclick=\"load()\">Return to Main Page </button>");
                        out.println("<script>    function load() {");
                        out.println("location.assign(\"http://localhost:8080/cs631db/MainProject.html\");");
                        out.println("} </script>      </div></body></html>");
                        } catch (SQLException e) {
                                System.out.println(errorMsgData);
                                e.printStackTrace();
                        }
                }
public void refreshPage(PrintWriter out, ResultSet newEmp, ResultSet newMgr, ResultSet newProj) throws SQLException {

                //title
                out.println("<html><head><link rel=\"stylesheet\" href=\"./CS631.css\">");
                out.println("<script type=\"text/javascript\"</script><script
src=\"https://code.jquery.com/jquery-1.12.4.min.js\"></script> ");
                out.println("<script src='js/autoresize.jquery.min.js'></script></head><body><img src=\"adminLogo.jpg\"
class=\"image1\"> ");

                out.println("<h1><br> <form method=\"post\" action=\"MainProject\"><tab2>The Best-P Company Project
Management (UPDATED)");
                out.println("<tab2><img src=\"warning.jpg\" style=\"width: 25px; height: 20px;\">");
                out.println("<input type=\"submit\" value=\"Restart Database\" name=\"restart\" class=\"button2\"></tab1></h1>
</form>");
//              out.println("");
                out.println("<h2><tab2><tab3> <a href=\"https://www.njit.edu\">https://www.njit.edu</a>");
                out.println("<tab1>Edge in Technology </h2>");
                out.println("<div id=\"navbar\"><p>");
                out.println("<a href=\"www.njit.edu\">Home</a> <a href=\"www.google.com\">About</a> ");
                out.println("<a href=\"#contact\">Contact</a> <a href=\"www.google.com\">Search</a> ");
                out.println("<a href=\"#\"><script>document.write(new Date().toLocaleDateString());</script></a></p></div>");
                out.println("<form id=\"vform\" method=\"post\" action=\"MainProject\"><! insert a new project query/>");
                // insert a new project
                out.println("<h3>  <div class=\"container\">");
                out.println(" <img src=\"insert.jpg\" style=\"width: 35px; height: 30px;\" class=\"image\">");
```

```
            out.println(" <div class=\"overlay\"> <tab1>Please insert a new project </div></div></h3> ");
            out.println("<h4><tab1><input type=\"checkbox\" name=\"checkNewProj\"> New Project ID:");
            out.println("  <input type=\"text\" maxlength=\"5\" size=\"5\" name=\"projNewID\"placeholder=\"112\"> ");
            //newProj
            out.println("  New Project Name:  <select name=\"projNewName\">");
            out.println("<option value=\"Disney Spring Resort Development\" selected>Disney Winter Resort
Development</option> ");
            out.println("<option value=\"Cape May Summer Resort Development\">Cape May Summer Resort
Development</option> ");
            out.println("<option value=\"LA Autumn Resort Development\">LA Autumn Resort Development</option> ");
            out.println("<option value=\"DC Winter Musium Development\">DC Autumn Resort Development</option></select> ");

            out.println("  New Project Location:  <select name=\"projNewLoc\">");
            out.println("<option value=\"FL ORLANDO\" selected>FL ORLANDO</option>");
            out.println("<option value=\"NEW YORK CITY\">NEW YORK CITY</option>");
            out.println("<option value=\"CA LOS ANGELES\">CA LOS ANGELES</option>");
            out.println("<option value=\"DC WASHINGTON\">DC WASHINGTON</option>");
            out.println("<option value=\"NJ KRAYS LANDING\">NJ KRAYS LANDING</option>");
            out.println("<option value=\"IN EAST CHICAGO\">IN EAST CHICAGO</option></select>");
            out.println("  New Project Budget:  <input type=\"text\" maxlength=\"15\" size=\"15\"
name=\"projNewbgt\" placeholder=\"1000000\">");
            out.println("<tab1> <br><br><tab3> <input type=\"submit\" value=\"Insert New Project\" name=\"projNew\"
class=\"button\">");
            // assign resources to a new project
            out.println("<br> <hr> <! assign new resources query/>");
            // static
            out.println("<h3>  <div class=\"container\">");
            out.println("<img src=\"aboutinfo.jpg\" style=\"width: 35px; height: 30px;\" class=\"image\">");
            out.println("<div class=\"overlay\"> <tab1>Please assign resources to the new project </div></div></h3> ");
            out.println("<h4> <tab1> <input type=\"checkbox\" name=\"checkAssignProj\"> New Project ID:");
            out.println("  <input type=\"text\" maxlength=\"5\" size=\"5\" name=\"projNewIDr\" placeholder=\"112\">");
            out.println("  Choose Employees (Multiple)");
            out.println("<select name=\"availEmp\" multiple=\"multiple\">");
            //refresh and update the available employees ResultSet newEmp
            while (newEmp.next()) {
                  out.println("<option value=\""+newEmp.getString(1)+"\">"+newEmp.getString(2)+"</option>");
            }
            //refresh and update the available managers ResultSet newMgr
            out.println("</select>   Choose Managers: <select name=\"availMgr\">");
            while (newMgr.next()) {
                  out.println("<option value=\""+newMgr.getString(1)+"\">"+newMgr.getString(2)+"</option>");
            }
            //static
            out.println(" </select>  New Project Start Time:  <select name=\"projStart\">");
            out.println("<option value=\"2023\" selected>2023</option>");
            out.println("<option value=\"2024\">2024</option>");
            out.println("<option value=\"2025\">2025</option> </select>");
            out.println(" New Project End Time:  <select name=\"projEnd\">");
            out.println("<option value=\"2023\" selected>2023</option>");
            out.println("<option value=\"2024\">2024</option>");
            out.println("<option value=\"2025\">2025</option> </select> <tab1> <br> <br>");
```

```
        out.println("<tab3> <input type=\"submit\" value=\"Assign to New Project\"  name=\"projAssign\" class=\"button\">");
        //download all project statistics
        out.println("<br>  <hr> <! start project query/> <h3>   <img src=\"projstatus.jpg\" style=\"width: 27px; height:
28px;\">Please choose Project Report:");
//      out.println("<tab2> (or Click and Download All)");
        out.println("<input type=\"submit\" name=\"allProj\"  value=\"Download All Project Reports include History
Projects\"");
        //choose one project to quote status
        out.println("class=\"button2\"> </h3> <hr> <! start each project query/> <h5>");
        out.println("<tab1>  <input type=\"radio\" name=\"checkProjSta\" value=\"1\" checked> Enter Project ID:");
        out.println("<tab1> <input type=\"text\" maxlength=\"5\" size=\"5\" name=\"projID\"> <tab1> or <br><br>");
        out.println("<tab1> <input type=\"radio\" name=\"checkProjSta\" value=\"2\">  Choose Project Name: <select
name=\"projName\">");

        //refresh and update the available project names ResultSet newProj
        while (newProj.next()) {
                out.println("<option value=\""+newProj.getString(1)+"\">"+newProj.getString(2)+"</option>");
        }
        out.println("</select> <tab1> <input type=\"submit\" value=\" Track Individual Project Status \" name=\"projSta\"
class=\"button2\"><hr>");
        out.println("</form></body></html>");
}
}
```

- ConnectionReset.java (JDBC connections, reset, update DBs)

```
import java.sql.*;
import java.io.IOException;
import java.io.PrintWriter;
import java.math.BigInteger;
public class ConnectionReset {

        static final String SQL_DRIVER = "com.mysql.cj.jdbc.Driver";
        static final String DB_URL = "jdbc:mysql://localhost/cs631db";
        static final String USER = "root";
        static final String PASS = "root";

        public Connection connection;
        public Statement stmt;
        public PreparedStatement pstmt;
        public ResultSet rs;

        String resetDB1 = "UPDATE employeeassign SET projID=null WHERE projID>='112'";
        String resetDB2 = "DELETE FROM project WHERE projID>='112'";

        String sqlEmpViewPay1 = "DROP view if exists empview";
        String sqlEmpViewPay2 = "CREATE VIEW empview AS " +
                        "SELECT e.empID, e.name, ea.title, e.start_date, e.address, e.city, e.state, " +
                        "e.zip, e.SSN, e.email, e.office_code, office.phone, office.office_name, ea.dept_code,
ea.div_code, ea.projID, " +
```

```
                                "ea.proj_startdate as start, ea.proj_enddate as end, salary.salary, salary.classification,
salary.hourly_pay, " +
                                " project.proj_name " +
                                "FROM employeepayroll as e " +
                                "JOIN employeeassign as ea ON e.empID=ea.empID " +
                                "JOIN office ON e.office_code =office.office_code " +
                                "JOIN project ON ea.projID =project.projID " +
                                "JOIN salary ON ea.title=salary.title";

        String sqlEmpViewProj1 = "DROP view if exists empview2";
        String sqlEmpViewProj2 = "CREATE VIEW empview2 AS " +
                                "SELECT e.empID, e.name, ea.title, e.start_date, e.address, e.city, e.state, " +
                                "e.zip, e.SSN, e.email, e.office_code, office.phone, office.office_name, ea.dept_code,
ea.div_code, ea.projID, " +
                                "ea.proj_startdate as start, ea.proj_enddate as end, salary.salary, salary.classification,
salary.hourly_pay, " +
                                " project.proj_name " +
                                "FROM employeepayroll as e " +
                                "JOIN employeeassign as ea ON e.empID=ea.empID " +
                                "JOIN office ON e.office_code =office.office_code " +
                                "JOIN project ON ea.projID =project.projID " +
                                "JOIN salary ON ea.title=salary.title";

        String sqlProjectSta1 = "DROP view if exists projectSta";
        String sqlProjectSta2 = "CREATE VIEW  projectSta AS "
                                + "(SELECT projID, year(end)-year(start) AS span, 'COMPLETED' as status, '100' as
milestone FROM project WHERE (CURDATE()>=end AND projID != '0') " +
                                "UNION  " +
                                "SELECT projID, '0' AS span, 'FUTURE' as status, '0' as milestone FROM project WHERE
(CURDATE()<=start AND projID != '0') " +
                                "UNION  " +
                                "SELECT projID, year(CURDATE())-year(start) AS span, 'ONGOING' as status, null
milestone FROM project WHERE (start<=CURDATE() AND CURDATE()<=end AND projID != '0') " +
                                ")";
        String sqlProjectEmpHr1 = "DROP view if exists projectEmpHr";
        String sqlProjectEmpHr2 = " CREATE VIEW  projectEmpHr AS "
                                + "(SELECT *, DATEDIFF(end, start)/7*40 AS workhour, DATEDIFF(end,
start)/7*40*hourly_pay as charge FROM empview2 WHERE (CURDATE()>=end AND projID != '0') " +
                                "UNION  " +
                                "SELECT *, '0' AS workhour, '0' as charge FROM empview2 WHERE (CURDATE()<=start
AND projID != '0') " +
                                "UNION  " +
                                "SELECT *, DATEDIFF(end, start)/7*40 AS workhour, DATEDIFF(end,
start)/7*40*hourly_pay as charge FROM empview2 WHERE (start<=CURDATE() AND CURDATE()<=end AND projID
!= '0') " +
                                ")";
        String sqlProgress1 = "DROP view if exists projectProgress";
        String sqlProgress2 = "CREATE VIEW projectProgress AS "+"SELECT * FROM "
                                +"(SELECT p.projID, p.proj_name, p.proj_manager, p.location, p.budget, p.start, p.end,
s.span, s.status, e.name, " +
                                "COUNT(*) as totalEmp, SUM(eh.charge) as totalCharge, SUM(eh.workhour) as
```

```
totalworkhour, SUM(eh.charge)/p.budget as prog_milestone " +
                                "FROM project p, projectSta s, projectEmpHr eh, empview2 e " +
                                "WHERE p.projID=s.projID AND p.proj_manager=e.empID AND p.projID=eh.projID " +
                                "GROUP BY p.projID" + ") as results " + "ORDER BY results.projID ASC";

        String sqlComIRS1 = "DROP view if exists companyirshr";
        String sqlComIRS2 = "CREATE VIEW companyirshr AS \r\n" +
                                "SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state, \r\n" +
                                "e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification,
e.hourly_pay as unit_pay, '0' as span, CAST(year(e.start) as signed integer) as IRSyear,
(DATEDIFF(e.end,e.start))/7*40*e.hourly_pay as SSincome\r\n" +
                                "FROM empview as e \r\n" +
                                "where year(e.start)=year(e.end)\r\n" +
                                "UNION\r\n" +
                                "/* generate IRS for all employees work for only two IRS years */\r\n" +
                                "SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state, \r\n" +
                                "e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification,
e.hourly_pay as unit_pay, '0' as span, CAST(year(e.start)as signed integer) as IRSyear,
(DATEDIFF(CONCAT(year(e.start),'-12-31'),start))/7*40*e.hourly_pay  as SSincome\r\n" +
                                "FROM empview as e\r\n" +
                                "where year(e.end)-year(e.start)=1\r\n" +
                                "UNION\r\n" +
                                "SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state, \r\n" +
                                "e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification,
e.hourly_pay as unit_pay, '0' as span, CAST(year(e.end)as signed integer) as IRSyear,
(DATEDIFF(end,CONCAT(year(e.end),'-1-1')))/7*40*e.hourly_pay  as SSincome\r\n" +
                                "FROM empview as e\r\n" +
                                "where year(e.end)-year(e.start)=1\r\n" +
                                "UNION\r\n" +
                                "/* generate IRS for all employees work for >three IRS years */\r\n" +
                                "SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state, \r\n" +
                                "e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification,
e.hourly_pay as unit_pay, '0' as span, CAST(year(e.start) as signed integer) as IRSyear,
(DATEDIFF(CONCAT(year(e.start),'-12-31'),start))/7*40*e.hourly_pay as SSincome\r\n" +
                                "FROM empview as e\r\n" +
                                "where year(e.end)-year(e.start)>1\r\n" +
                                "UNION\r\n" +
                                "SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state, \r\n" +
                                "e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification,
e.hourly_pay as unit_pay, '0' as span, CAST(year(e.end) as signed integer)as IRSyear,
(DATEDIFF(end,CONCAT(year(e.end),'-1-1')))/7*40*e.hourly_pay as SSincome\r\n" +
                                "FROM empview as e\r\n" +
                                "where year(e.end)-year(e.start)>1\r\n" +
                                "UNION\r\n" +
                                "SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state, \r\n" +
                                "e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification, e.salary
as unit_pay, (year(e.end)-year(e.start)-1) as span, CAST((year(e.start)+1)as signed integer) as IRSyear, e.salary as
SSincome\r\n" +
                                "FROM empview as e\r\n" +
                                "where year(e.end)-year(e.start)>1";
```

```
	String sqlComIRS3 = "DROP table if exists companyirs";
	String sqlComIRS4 = "CREATE table companyirs AS \r\n" +
				"SELECT c.empID, c.name, c.address, c.city, c.state, c.zip, c.SSN, c.office_code,
c.dept_code, c.div_code, c.classification,\r\n" +
				"c.span, c.IRSyear, c.SSincome, SSincome*0.1 as FederalTax, SSincome*0.05 as
StateTax, SSincome*0.062 as SSTax, SSincome*0.0145 as MedicalTax, SSincome*(1-0.1-0.05-0.062-0.0145) as
StateWage\r\n" +
				"FROM companyirshr as c\r\n" +
				"WHERE c.start_date<=CURDATE() " ;

	//these tupples are multiple years
//	String sqlComIRS5 ="DROP view if exists allirs";
	String sqlComIRS6 ="SELECT * FROM companyirs WHERE span>1";

	String sqlPrepIRS = "INSERT INTO companyirs VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?,?, ?, ?,?)";

	String errorMessage1 = "SQL connection failed! ";
	String errorMessage2 = "Wrong Reset, Missing Project Data!";
	String goodMessage2 = "Database reset, good to go!";
	String errorMessage3 = "Wrong Reset, Missing Employee Data!";
	String goodMessage3 = "Employee reset, good to go!";
	String errorMessage4 = "Wrong Reset, Missing Emp-Project Data!";
	String goodMessage4 = "Emp-Project reset, good to go!";
	String errorMessage5 = "Wrong Reset, Missing Status-Project Data!";
	String goodMessage5 = "Status-Project reset, good to go!";

	public ConnectionReset() {
		try {
			Class.forName(SQL_DRIVER).newInstance();
			connection = DriverManager.getConnection(DB_URL, USER, PASS);
			stmt = connection.createStatement();
			// pstmt = connection.prepareStatement(sqlQuerySal);

		} catch (Exception e) {
			new AlertReminder().popAlert(errorMessage1);
			System.out.print(e.getMessage());
		}
	}
	public void resetDB() {
		try {
			stmt.executeUpdate(resetDB1);
			stmt.executeUpdate(resetDB2);
			new AlertReminder().popAlert(goodMessage2);
		} catch (Exception e) {
			new AlertReminder().popAlert(errorMessage2);
			System.out.print(e.getMessage());
		}
	}
	public void iniPayroll() {
		//generate empview with most updated entity tables for payroll analysis
		try {
```

```
                        stmt.executeUpdate(sqlEmpViewPay1);
                        stmt.execute(sqlEmpViewPay2);
                        new AlertReminder().popAlert(goodMessage3);
                } catch (Exception e) {
                        new AlertReminder().popAlert(errorMessage3);
                        System.out.print(e.getMessage());
                }
        }
        public void iniProject() {
                //generate empview2 with most updated entity tables for project analysis
                try {
                        stmt.executeUpdate(sqlEmpViewProj1);
                        stmt.execute(sqlEmpViewProj2);
                        new AlertReminder().popAlert(goodMessage4);
                } catch (Exception e) {
                        new AlertReminder().popAlert(errorMessage4);
                        System.out.print(e.getMessage());
                }
        }
        public void staProject() {
                //initiate and generate project status with most updated entity tables for project progress
                try {
                        iniProject();
                        stmt.executeUpdate(sqlProjectSta1);
                        stmt.execute(sqlProjectSta2);
                        stmt.executeUpdate(sqlProjectEmpHr1);
                        stmt.execute(sqlProjectEmpHr2);
                        stmt.executeUpdate(sqlProgress1);
                        stmt.execute(sqlProgress2);
                        new AlertReminder().popAlert(goodMessage5);
                } catch (Exception e) {
                        new AlertReminder().popAlert(errorMessage5);
                        System.out.print(e.getMessage());
                }
        }
                public void companyIRS() {
                        //initiate empview with most updated entity tables and generate the IRS report for
entire company
                        Statement dupStmt = null;
                        ResultSet rs = null;

                        try {
                                iniPayroll();
                                stmt.executeUpdate(sqlComIRS1);
                                stmt.execute(sqlComIRS2);
                                stmt.executeUpdate(sqlComIRS3);
                                stmt.execute(sqlComIRS4);
//                              stmt.executeUpdate(sqlComIRS5);

                        // regenerate the duplicated years for employees working for more than 2 year span
                        //sqlPrepIRS = "INSERT INTO companyirs VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?,?,
```

```
?, ?,?)";
                                        pstmt=connection.prepareStatement(sqlPrepIRS);
                                        dupStmt = connection.createStatement();
                                        rs=dupStmt.executeQuery(sqlComIRS6);

                                        while (rs.next()) {
                                                int counter=rs.getInt(12);
                                                int year=rs.getInt(13);
                                                while (counter>1) {
                                                        counter--;
                                                        year++;
                                                        pstmt.setString(1, rs.getString(1));
                                                        pstmt.setString(2, rs.getString(2));
                                                        pstmt.setString(3, rs.getString(3));
                                                        pstmt.setString(4, rs.getString(4));
                                                        pstmt.setString(5, rs.getString(5));
                                                        pstmt.setString(6, rs.getString(6));
                                                        pstmt.setString(7, rs.getString(7));
                                                        pstmt.setString(8, rs.getString(8));
                                                        pstmt.setString(9, rs.getString(9));
                                                        pstmt.setString(10, rs.getString(10));
                                                        pstmt.setString(11, rs.getString(11));
                                                        //set missing IRS year
                                                        pstmt.setInt(12, counter);
                                                        pstmt.setInt(13, year);
                                                        //set salaries in IRS form
                                                        pstmt.setDouble(14, rs.getDouble(14));
                                                        pstmt.setDouble(15, rs.getDouble(15));
                                                        pstmt.setDouble(16, rs.getDouble(16));
                                                        pstmt.setDouble(17, rs.getDouble(17));
                                                        pstmt.setDouble(18, rs.getDouble(18));
                                                        pstmt.setDouble(19, rs.getDouble(19));
                                                        pstmt.execute();
                                                }
                                        }
                                        rs.close();
                                        dupStmt.close();

                                        new AlertReminder().popAlert(goodMessage5);
                                } catch (Exception e) {
                                        new AlertReminder().popAlert(errorMessage5);
                                        System.out.print(e.getMessage());
                                }
                }
}
```

- Supportive classes to convert and calculate difference in dates, days, month, and years from SQL to java

```
import java.text.ParseException;
```

```
import java.text.SimpleDateFormat;
import java.util.Date;

class TimeDiff {
            static long difference_In_Years;
            static long difference_In_Days;
   // Function to print difference in
   // time start_date and end_date
   static long findDifference(String start_date,  String end_date)
   {
      // SimpleDateFormat converts the
      // string format to date object
      SimpleDateFormat sdf  = new SimpleDateFormat( "yyyy-MM-dd" );

      // Try Block
      try {
         // parse method is used to parse
         // the text from a string to
         // produce the date
         Date d1 = sdf.parse(start_date);
         Date d2 = sdf.parse(end_date);

         // Calucalte time difference
         // in milliseconds
         long difference_In_Time  = d2.getTime() - d1.getTime();

         // Calucalte time difference in
         // seconds, minutes, hours, years,
         // and days
         long difference_In_Seconds  = (difference_In_Time  / 1000) % 60;
         long difference_In_Minutes  = (difference_In_Time  / (1000 * 60)) % 60;
         long difference_In_Hours  = (difference_In_Time  / (1000 * 60 * 60))  % 24;
         difference_In_Years = (difference_In_Time  / (1000l * 60 * 60 * 24 * 365));
         difference_In_Days  = (difference_In_Time  / (1000 * 60 * 60 * 24))  % 365;

         // Print the date difference in
         // years, in days, in hours, in
         // minutes, and in seconds
         System.out.print(  "Difference "  + "between two dates is: ");
         System.out.println(  difference_In_Years  + " years, "  + difference_In_Days
            + " days, "
            + difference_In_Hours
            + " hours, "
            + difference_In_Minutes
            + " minutes, "
            + difference_In_Seconds
            + " seconds");
      }
      // Catch the Exception
      catch (ParseException e) {
         e.printStackTrace();
```

```
    }
    return difference_In_Days;
  }
  // Driver Code
  public static void main(String[] args)
  {
    // Given start Date
    String start_date
      = "2018-10-01";
    // Given end Date
    String end_date
      = "2018-12-01";
    // Function Call
    findDifference(start_date, end_date);
  }
}

import java.io.IOException;
import java.io.PrintWriter;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.*;
import java.sql.Date;

public class StringToDate {
        //convert a string to sql date
        SimpleDateFormat datef = new SimpleDateFormat("yyyy-MM-dd");
        String date;
        private java.sql.Date sqlDate;
        private java.util.Date utilDate;
        public StringToDate(String date) {
                this.date=date;
        }
        private Date stringToDate(String date) throws Exception {
    java.sql.Date sqlDate = null;
    if( !date.isEmpty()) {

      try {
        java.util.Date normalDate = datef.parse(date);
        sqlDate = new java.sql.Date(normalDate.getTime());
      } catch (ParseException e) {
        throw new Exception("Not able to Parse the date", e);
      }
    }
    return sqlDate;
  }
        private String dateToString (java.util.Date utilDate) {
                String stringDate = datef.format(utilDate);
                return stringDate;
        }
        public void setSqlDate(java.sql.Date sqlDate) {
```

```
                this.sqlDate = sqlDate;
        }
        public java.sql.Date getSqlDate( ) {
                try {
                        sqlDate=stringToDate(date);
                } catch (Exception e) {
                        e.printStackTrace();
                }
                return sqlDate;
        }
}
```

Appendix B Web-Interface Code

- loginPayroll.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Payroll Login Form</title>
<link rel="stylesheet" href="login.css">
<script type="text/javascript"</script>
<script src="https://code.jquery.com/jquery-1.12.4.min.js">  </script>
</head>
<body class="body">
        <!-- partial:index.html -->
        <img src="adminLogo.jpg"
                style="float: left; width: 100px; height: 100px;">
        <b> <font color="#ff0000" face="Calibri" size="5">  
                                  The Best-P Company Payroll Login <br>
                                           
                           <i><a href="https://www.njit.edu"> New Jersey
                                        Institute of Tech. </a></i><font face="Calibri" size="3"
color="#00ffff">
                                <br>               
                                        YuQing Ding, Wei Zhang<br>  
                                               
                                              <script
                                        class=message>
          document.write(new Date().toLocaleDateString());
        </script>
                                <div class="login-page">
                                        <div class="form">
                                                <form class="register-form">
                                                        <input type="text" name="loginName"
placeholder="name" /> <input
                                                                        type="email" name="email"
placeholder="email address" />
```

```
                                                        <button>create</button>
                                                        <p class="message">
                                                                Already registered? <a href="#">Sign
In</a>
                                                        </p>
                                                </form>
                                                <form class="login-form" action="#">
                                                        <p class="message">
                                                                Not registered? <a href="#">Create an
account</a>
                                                        </p>
                                                        <br> <input type="text" name="useridPay"
                                                                placeholder="userID (001)" /> <input
type="password"
                                                                name="pwdPay" placeholder="password"
/> <input type="text"
                                                                name="emailPay" placeholder="email
address" /> <input
                                                                type="submit" value="login"
name="loginPay"
                                                                onclick="return check(this.form)"
                                                                style="background-color: #4CAF50;/>
   <p class="message">
                                                        <a href="#"> </a>
                                                        </p>
                                                </form>
                                        </div>
                                </div> <!-- checkout user account --> <script type="text/javascript">
   function check(form) {
    if (form.useridPay.value == "001"|"002"|"003"|"004" && form.pwdPay.value == "root") {
    alert("Welcome to Best-P Company!")
    window.load()
     return true;
    } else {
      alert("Access denied - Please try again!")
      return false;
    }
   }
      <!-- Script to use Location assign() Method -->
  function load() {
location.assign("http://localhost:8080/cs631db/MainPayroll");
}
  </script>
</body>
</html>
```

- loginProject.html

```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
<meta charset="UTF-8">
<title>Administration Login Form</title>
<link rel="stylesheet" href="login.css">
<script type="text/javascript"></script>
<script src="https://code.jquery.com/jquery-1.12.4.min.js">  </script>
</head>
<body class="body">
		<!-- partial:index.html -->
		<img src="adminLogo.jpg"
			style="float: left; width: 100px; height: 100px;">
		<b> <font color="#ff0000" face="Calibri" size="5">  
				          The Best-P Company Project
				Administration Login <br>          
				             <i><a
					href="https://www.njit.edu"> New Jersey Institute of Tech. </a></i><font
				face="Calibri" size="3" color="#00ffff"> <br>  
					                  
					  YuQing Ding, Wei Zhang<br>        
					                  
					       <script class=message>
	  document.write(new Date().toLocaleDateString());
	</script>
					<div class="login-page">
						<div class="form">
							<form class="register-form">
								<input type="text" name="loginName"
placeholder="name" /> <input
									type="email" name="email"
placeholder="email address" />
								<button>create</button>
								<p class="message">
									Already registered? <a href="#">Sign
In</a>
								</p>
							</form>
							<form class="login-form" action="#">
								<p class="message">
									Not registered? <a href="#">Create an
account</a>
								</p>
								<br> <input type="text" name="useridProj"
									placeholder="userID(101)" /> <input
type="password"
									name="pwdProj" placeholder="password"
/> <input type="text"
									name="emailProj" placeholder="email
address" /> <input
									type="submit" value="Login"
name="loginProj"
									onclick="return check(this.form)"
```

```
                                                                    style="background-color: #4CAF50;/>
  <p class="message">
                                                                    <a href="#"> </a>
                                                                    </p>
                                                            </form>
                                                    </div>
                                            </div> <!-- checkout user account --> <script type="text/javascript">
  function check(form) {
    if (form.useridProj.value == "101"|"102"|"103"|"104" && form.pwdProj.value == "root") {
    alert("Welcome to Best-P Company!")
      return true;
    } else {
      alert("Access denied - Please try again!")
      return false;
    }
  }
 </script>
</body>
</html>
```

- MainPayroll.html

```
<html>
<head>
<link rel="stylesheet" href="./CS631.css">
<script type="text/javascript"></script>
<script src="https://code.jquery.com/jquery-1.12.4.min.js">
</script>
</head>
<body>
        <img src="adminLogo.jpg" class="image1">
        <h1>
                <br>
                <tab2> The Best-P Company Payroll Management
        </h1>
        <h2>
                <br>
                <tab3> <a href="  https://www.njit.edu">https://www.njit.edu</a>
                <tab1>Edge in Technology
        </h2>
        <br>
        <br>
        <div id="navbar">
                <p>
                        <a href="www.njit.edu">Home</a> <a href="www.google.com">About</a> <a
                                href="#contact">Contact</a> <a href="www.google.com">Search</a> <a
                                href="#"><script>
                                        document.write(new Date().toLocaleDateString());
                                </script></a>
                </p>
```

```
</div>
<form id="vform" method="post" action="MainPayroll">
	<h3>
		 
		<div class="container">
			<img src="paycheck.jpg" style="width: 34px; height: 30px;"
				class="image">
			<div class="overlay">
				<tab1>Please choose pay check options
			</div>
		</div>
		<br>
	</h3>
	<hr>
	<! start emp query/>
	<h4>
		<tab3> <input type="checkbox" name="checkPayEmp">
		Employee ID:     <input type="text" maxlength="5" size="5"
			name="empID" placeholder="1">        
		    Choose Payroll Period: <select name="monthEmp">
			<option value="1">January</option>
			<option value="2">February</option>
			<option value="3">March</option>
			<option value="4">April</option>
			<option value="5">May</option>
			<option value="6">June</option>
			<option value="7">July</option>
			<option value="8">August</option>
			<option value="9">September</option>
			<option value="10">October</option>
			<option value="11">November</option>
			<option value="12">December</option>
		</select> <select name="yearEmp">
			<option value="2015">2015</option>
			<option value="2016">2016</option>
			<option value="2017">2017</option>
			<option value="2018">2018</option>
			<option value="2019">2019</option>
			<option value="2020" selected>2020</option>
			<option value="2021">2021</option>
		</select> <tab1> <input type="submit" value="Get Pay Check"
			name="payEmp" class="button2"> <br>
		<br>
		<! start dep query/> <tab3> <input type="checkbox"
			name="checkPayDept"> Department ID:   <input
			type="text" maxlength="5" size="5" name="deptID" placeholder="1001">
		            Choose Payroll Period: <select
			name="monthDept">
			<option value="1">January</option>
			<option value="2">February</option>
			<option value="3">March</option>
```

```
                    <option value="4">April</option>
                    <option value="5">May</option>
                    <option value="6">June</option>
                    <option value="7">July</option>
                    <option value="8">August</option>
                    <option value="9">September</option>
                    <option value="10">October</option>
                    <option value="11">November</option>
                    <option value="12">December</option>
                </select> <select name="yearDept">
                    <option value="2015">2015</option>
                    <option value="2016">2016</option>
                    <option value="2017">2017</option>
                    <option value="2018">2018</option>
                    <option value="2019">2019</option>
                    <option value="2020" selected>2020</option>
                    <option value="2021">2021</option>
                </select> <tab1> <input type="submit"
                    value="Department Monthly Cost Report" name="payDept" class="button">
                <br>
                <br>
                <! start div query/> <tab3> <input type="checkbox"
                    name="checkPayDiv"> Division ID: <tab1> <input
                    type="text" maxlength="5" size="5" name="divID" placeholder="FED">
                <tab1> Choose Payroll Period: <select name="monthDiv">
                    <option value="1">January</option>
                    <option value="2">February</option>
                    <option value="3">March</option>
                    <option value="4">April</option>
                    <option value="5">May</option>
                    <option value="6">June</option>
                    <option value="7">July</option>
                    <option value="8">August</option>
                    <option value="9">September</option>
                    <option value="10">October</option>
                    <option value="11">November</option>
                    <option value="12">December</option>
                </select> <select name="yearDiv">
                    <option value="2015">2015</option>
                    <option value="2016">2016</option>
                    <option value="2017">2017</option>
                    <option value="2018">2018</option>
                    <option value="2019">2019</option>
                    <option value="2020" selected>2020</option>
                    <option value="2021">2021</option>
                </select> <tab1>  <input type="submit"
                    value="Division Monthly Cost Report" name="payDiv" class="button">
            </h4>
            <hr>
            <! start IRS query/>
            <h3>
```

```
                         
                        <div class="container">
                                <img src="tax.jpg" style="width: 34px; height: 30px;" class="image">
                                <div class="overlay">
                                        <tab1>Please choose IRS Report Types
                                </div>
                        </div>
                        <br>
                </h3>
                <h5>
                        <tab2> <input type="checkbox" name="checkComIRS">
                        Start Year:   <input type="text" maxlength="5" size="5"
                                name="startYr" placeholder="2010"> <tab1> End Year:
                          <input type="text" maxlength="5" size="5" name="endYr"
placeholder="2022">
                                <tab1> <input type="submit"
                                name="comIRS" value="Download IRS Reports for Entire Company"
                                class="button2""> </a></tab2>
                </h5>
                <hr>
                <! start emp query/>
                <h5>
                        <tab2> <input type="checkbox" name="checkEmpIRS">
                        Employee ID:     <input type="text" maxlength="5" size="5"
                                name="empIDIRS" placeholder="1">        
                           Choose IRS Period: <select name="yearEmpIRS">
                                <option value="2015">2015</option>
                                <option value="2016">2016</option>
                                <option value="2017">2017</option>
                                <option value="2018">2018</option>
                                <option value="2019">2019</option>
                                <option value="2020" selected>2020</option>
                                <option value="2021">2021</option>
                        </select> <tab2> <input type="submit" value="Get Employee IRS Report"
                                name="empIRS" class="button2"> <br>
                        <br>
                        <hr>
        </form>
</body>
</html>
```

- MainProject.html

```
<html>
<head>
<link rel="stylesheet" href="./CS631.css">
<script type="text/javascript"</script>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
<script src='js/autoresize.jquery.min.js'></script>
</head>
```

```
<body>
	<img src="adminLogo.jpg" class="image1">
	<h1>
		<br>
		<form method="post" action="MainProject">
		<tab2>  The Best-P Company Project Management
			<tab2>
			<img src="warning.jpg" style="width: 25px; height: 20px;"> <input
				type="submit" value="Restart Database" name="restart"
				class="button2"></tab1>
	</h1>
	</form>
	<h2>
		<tab3><tab2> <a href="    https://www.njit.edu">https://www.njit.edu</a>
		<tab1>Edge in Technology
	</h2>
	<div id="navbar">
		<p>
			<a href="www.njit.edu">Home</a> <a href="www.google.com">About</a> <a
				href="#contact">Contact</a> <a href="www.google.com">Search</a> <a
				href="#"><script>document.write(new
Date().toLocaleDateString());</script></a>
		</p>
	</div>
	<form id="vform" method="post" action="MainProject">
		<! insert a new project query/>
		<h3>
			 
			<div class="container">
			<br>	<img src="insert.jpg" style="width: 35px; height: 30px;"
					class="image">
				<div class="overlay">
					<tab1>Please insert a new project
				</div>
			</div>
		</h3>
		<h4>
			<tab1>
			<input type="checkbox" name="checkNewProj"> New Project ID:
			  <input type="text" maxlength="5" size="5" name="projNewID"
				placeholder="112">   New Project Name:  <select
				name="projNewName">
				<option value="Disney Spring Resort Development" selected>Disney
					Winter Resort Development</option>
				<option value="Cape May Summer Resort Development">Cape May
					Summer Resort Development</option>
				<option value="LA Autumn Resort Development">LA Autumn
					Resort Development</option>
				<option value="DC Winter Musium Development">DC Autumn
					Resort Development</option>
			</select>   New Project Location:  <select name="projNewLoc">
```

```
                <option value="FL ORLANDO" selected>FL ORLANDO</option>
                <option value="NEW YORK CITY">NEW YORK CITY</option>
                <option value="CA LOS ANGELES">CA LOS ANGELES</option>
                <option value="DC WASHINGTON">DC WASHINGTON</option>
                <option value="NJ KRAYS LANDING">NJ KRAYS LANDING</option>
                <option value="IN EAST CHICAGO">IN EAST CHICAGO</option>
        </select>   New Project Budget:  <input type="text" maxlength="15"
                size="15" name="projNewbgt" placeholder="1000000"> <tab1>
        <br>
        <br>
        <tab3> <input type="submit" value="Insert New Project"
                name="projNew" class="button"> <br>
        <hr>
        <! assign new resources query/>
        <h3>
                 
                <div class="container">
                        <img src="aboutinfo.jpg" style="width: 35px; height: 30px;"
                                class="image">
                        <div class="overlay">
                                <tab1>Please assign resources to the new project
                        </div>
                </div>
        </h3>
        <h4>
                <tab1> <input type="checkbox" name="checkAssignProj">
                New Project ID:   <input type="text" maxlength="5" size="5"
                        name="projNewIDr" placeholder="112">   Choose
                Employees (Multiple): <select name="availEmp" multiple="multiple">
                        <option value="7">Bolton, David</option>
                        <option value="9">Bomboy, James</option>
                        <option value="19">Bonhom, Terrell</option>
                        <option value="27">Booker, Petrine</option>
                        <option value="40">Ghee, Jamie</option>
                        <option value="63">Harrison, Valencia</option>
                        <option value="64">Harrison, Villie</option>
                        <option value="68">Harski, Richard</option>
                        <option value="70">Hartiens, Benjamin</option>
                        <option value="76">Lin, Qinglu</option>
                        <option value="96">Lincoln, Jonh</option>
                        <option value="105">Ling, Alice</option>
                        <option value="109">Lionberger, Brian</option>
                </select>    Choose Managers: <select name="availMgr">
                        <option value="105">Ling, Alice</option>
                        <option value="116">Joseph Paul</option>
                        <option value="117">Maria Swabota</option>
                        <option value="118">Michael Anthony</option>
                        <option value="119">Pitbull Perez</option>
                        <option value="120">Richard Steven</option>
                        <option value="121">Robert Daniel</option>
                        <option value="122">Thomas Andrew</option>
```

```
                                        <option value="123">William Mark</option>
                                    </select>  New Project Start Time:  <select name="projStart">
                                        <option value="2023" selected>2023</option>
                                        <option value="2024">2024</option>
                                        <option value="2025">2025</option>
                                    </select>   New Project End Time:  <select name="projEnd">
                                        <option value="2023" selected>2023</option>
                                        <option value="2024">2024</option>
                                        <option value="2025">2025</option>
                                    </select> <tab1> <br>
                                    <br>
                                    <tab3> <input type="submit" value="Assign to New Project"
                                        name="projAssign" class="button"> <br>
                                    <hr>
                                    <! start project query/>
                                    <h3>
                                          <img src="projstatus.jpg" style="width: 27px; height:
28px;">Please
                                        choose Project Report:
                                        <tab2> <input type="submit" name="allProj"
                                            value="Download All Project Reports include History
Projects"
                                            class="button2">
                                    </h3>
                                    <hr>
                                    <! start each project uery/>
                                    <h5>
                                        <tab1> <input type="radio" name="checkProjSta" value="1"
                                            checked> Enter Project ID: <tab1> <input
                                            type="text" maxlength="5" size="5" name="projID"> <tab1>
                                        or <br>
                                        <br>
                                        <tab1> <input type="radio" name="checkProjSta" value="2">
                                        Choose Project Name: <select name="projName">
                                            <option value="101">Dagwood Development</option>
                                            <option value="102">Cyclone Development</option>
                                            <option value="103">Yellow Moose Development</option>
                                            <option value="104">Boomrenge Development</option>
                                            <option value="105">Bongo Development</option>
                                            <option value="106">Hashtag Development</option>
                                            <option value="107">Silverstar Development</option>
                                            <option value="108">Hidden Hook Development</option>
                                            <option value="109">Early First Development</option>
                                            <option value="110">Bull Winky Development</option>
                                            <option value="111">Lonely Fox Development</option>
                                            <option value="112">Disney Winter Development</option>
                                        </select> <! start Status Report/> <tab1> <input type="submit"
                                            value=" Track Individual Project Status " name="projSta"
                                            class="button2"> <br>
                                        <hr>
        </form>
```

```
</body>
</html>
```

Appendix C SQL Source

```
/* Generate paychecks with monthly salary and tax history for all types of employees  */
/* note that hourly employees are contractors who has limited duration and no benefit plans!*/
/* note that in the Salary paychecks/cost/IRS/Project calculations use new start and end times for contractors
only*/
/* PART 1. START OF PAYROLL APP   */

DROP view if exists empview;
CREATE VIEW empview AS
SELECT e.empID, e.name, ea.title, e.start_date, e.address, e.city, e.state,
e.zip, e.SSN, e.email, e.office_code, office.phone, office.office_name, ea.dept_code, ea.div_code, ea.projID,
ea.proj_startdate as start, ea.proj_enddate as end, salary.salary, salary.classification, salary.hourly_pay,
 project.proj_name
FROM employeepayroll as e
JOIN employeeassign as ea ON e.empID=ea.empID
JOIN office ON e.office_code =office.office_code
JOIN project ON ea.projID =project.projID
JOIN salary ON ea.title=salary.title
;
/* Generate paychecks with monthly salary  */
DROP view if exists empsal;
CREATE VIEW empsal AS
SELECT * FROM cs631db.empview
WHERE empID = '1' AND start_date<='2020-12-31' AND start<='2020-01-01' AND '2020-01-01'<=end;

/* Generate monthly cost reports for all departments   */
DROP view if exists deptsalperm;
CREATE VIEW deptsalperm AS
SELECT d.dept_code,d.dept_name, e.name as manager, e.phone, d.email, count(*) as permEmp,
sum(view.hourly_pay) as permSal
FROM cs631db.empview as view, department as d, empview as e
WHERE view.dept_code= d.dept_code AND view.classification='permanent' AND d.dept_head=e.empID
AND view.start_date<='2020-12-31'AND view.start<='2020-01-01' AND '2020-01-01'<=view.end
AND d.dept_code='1001';

/*Generate monthly cost reports for all departments    */
DROP view if exists deptsalhour;
CREATE VIEW deptsalhour AS
SELECT d.dept_code,d.dept_name, e.name as manager, e.phone, d.email, count(*) as hourEmp,
sum(view.hourly_pay) as hourSal
FROM cs631db.empview as view, department as d, empview as e
WHERE view.dept_code= d.dept_code AND view.classification='hourly' AND d.dept_head=e.empID
AND view.start_date<='2020-12-31'AND view.start<='2020-01-01' AND '2020-01-01'<=view.end
```

```
AND d.dept_code='1001';
/* Generate monthly cost reports for all divisions  */
DROP view if exists divsalperm;
CREATE VIEW divsalperm AS
SELECT d.div_code,d.div_name, e.name as manager, e.phone, d.email, count(*) as permEmp,
sum(view.hourly_pay) as permSal
FROM cs631db.empview as view, division as d, empview as e
WHERE view.div_code= d.div_code AND view.classification='permanent' AND d.div_head=e.empID
AND view.start_date<='2020-12-31'AND view.start<='2020-01-01' AND '2020-01-01'<=view.end
AND d.div_code='FED';
/* Generate monthly cost reports for all divisions  */
DROP view if exists divsalhour;
CREATE VIEW divsalhour AS
SELECT d.div_code,d.div_name, e.name as manager, e.phone, d.email,count(*) as hourEmp,
sum(view.hourly_pay) as hourSal
FROM cs631db.empview as view, division as d, empview as e
WHERE view.div_code= d.div_code AND view.classification='hourly' AND d.div_head=e.empID
AND view.start_date<='2020-12-31'AND view.start<='2020-01-01' AND '2020-01-01'<=view.end
AND d.div_code='FED';

/* Generate IRS tax form for employee #001 in 2020, 2010, 2015, 2018, and hourly emp. #103 partial year */
/* check if full year for IRS report */
DROP view if exists empIRS20;
CREATE VIEW empIRS20 AS
SELECT *, salary as income FROM cs631db.empview
WHERE empID = '1' AND start_date<='2020-12-31' AND start<='2020-01-01' AND '2020-12-31'<=end;

/* check if project start at last half partial year */
DROP view if exists empIRS20p1;
CREATE VIEW empIRS20p1 AS
SELECT *, hourly_pay*DATEDIFF('2020-12-31', start)/7*40 as income  FROM cs631db.empview
WHERE empID = '1' AND start_date<='2020-12-31' AND start>='2020-01-01' AND start<='2020-12-31';

/* check if project end at first half partial year */
DROP view if exists empIRS20p2;
CREATE VIEW empIRS20p2 AS
SELECT *, hourly_pay*DATEDIFF(end, '2020-01-01')/7*40  FROM cs631db.empview
WHERE empID = '1' AND start_date<='2020-12-31' AND end>='2020-01-01' AND end<='2020-12-31';

/* for late join/start, check if hourly at 2011 start at last half partial year */
DROP view if exists empIRS11hr;
CREATE VIEW empIRS11hr AS
SELECT *, hourly_pay*DATEDIFF('2011-12-31',start)/7*40 as income  FROM cs631db.empview
WHERE empID = '103' AND start_date<='2011-12-31' AND start>='2011-01-01' AND start<='2011-12-31';

/* for early leave, check if hourly at 2018 end at first half partial year */
DROP view if exists empIRS18hr;
CREATE VIEW empIRS18hr AS
SELECT *, hourly_pay*DATEDIFF(end, '2018-01-01')/7*40 as income  FROM cs631db.empview
WHERE empID = '103' AND start_date<='2018-12-31' AND end>='2018-01-01' AND end<='2018-12-31';
```

```
 /* Generate IRS tax forms for all types of employees -see a series of SQL queries  */

/* generate IRS for all employees work for only one IRS year */
DROP view if exists companyirshr;
CREATE VIEW companyirshr AS
SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state,
e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification, e.hourly_pay as unit_pay, 0 as span,
CAST(year(e.start) as signed integer) as IRSyear, (DATEDIFF(e.end,e.start))/7*40*e.hourly_pay as SSincome
FROM empview as e
where year(e.start)=year(e.end)
UNION
/* generate IRS for all employees work for only two IRS years */
SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state,
e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification, e.hourly_pay as unit_pay, 0 as span,
CAST(year(e.start)as signed integer) as IRSyear,
(DATEDIFF(CONCAT(year(e.start),'-12-31'),start))/7*40*e.hourly_pay  as SSincome
FROM empview as e
where year(e.end)-year(e.start)=1
UNION
SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state,
e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification, e.hourly_pay as unit_pay, 0 as span,
CAST(year(e.end)as signed integer) as IRSyear, (DATEDIFF(end,CONCAT(year(e.end),'-1-1')))/7*40*e.hourly_pay
as SSincome
FROM empview as e
where year(e.end)-year(e.start)=1
UNION
/* generate IRS for all employees work for >=three IRS years */
SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state,
e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification, e.hourly_pay as unit_pay, 0 as span,
CAST(year(e.start) as signed integer) as IRSyear,
(DATEDIFF(CONCAT(year(e.start),'-12-31'),start))/7*40*e.hourly_pay as SSincome
FROM empview as e
where year(e.end)-year(e.start)>1
UNION
SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state,
e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification, e.hourly_pay as unit_pay, 0 as span,
CAST(year(e.end) as signed integer)as IRSyear, (DATEDIFF(end,CONCAT(year(e.end),'-1-1')))/7*40*e.hourly_pay as
SSincome
FROM empview as e
where year(e.end)-year(e.start)>1
UNION
SELECT e.empID, e.name, e.title, e.start_date, e.address, e.city, e.state,
e.zip, e.SSN, e.email, e.office_code, e.dept_code, e.div_code, e.classification, e.salary as unit_pay,
(year(e.end)-year(e.start)-1) as span, CAST((year(e.start)+1)as signed integer) as IRSyear, e.salary as SSincome
FROM empview as e
where year(e.end)-year(e.start)>1
;
/*          Integer taxFed = (int) (empIRSIncome * 0.1);
            Integer taxSta = (int) (empIRSIncome * 0.05);
           Integer taxSS = (int) (empIRSIncome * 0.062);
            Integer medWage=empIRSIncome-taxFed-taxSS;
```

```
            Integer taxMedi = (int) (empIRSIncome * 0.0145);
            Integer netWage=empIRSIncome-taxFed-taxSS-taxMedi; */

/*create a raw table of all taxes and salaries for all IRS years*/
DROP table if exists companyirs;
CREATE table companyirs AS
SELECT c.empID, c.name, c.address, c.city, c.state, c.zip, c.SSN, c.office_code, c.dept_code, c.div_code,
c.classification,
c.span, c.IRSyear, c.SSincome, SSincome*0.1 as FederalTax, SSincome*0.05 as StateTax, SSincome*0.062 as SSTax,
SSincome*0.0145 as MedicalTax, SSincome*(1-0.1-0.05-0.062-0.0145) as StateWage
FROM companyirshr as c
WHERE c.start_date<=CURDATE()
;
/* duplicate those rows of multiple year span>0 */
DROP view if exists allirs;
CREATE VIEW allirs as
SELECT * FROM companyirs WHERE span>1;

/* select years of IRS report to generate a IRS report for the entire company*/
SELECT * FROM companyirs WHERE IRSyear>=2010 AND IRSyear<=2020 ORDER BY IRSyear, empID ASC;

/* PART 2. START OF PROJECT MANAGEMENT APP   */
/* update employ list */
DROP view if exists empview2;
CREATE VIEW empview2 AS
SELECT e.empID, e.name, ea.title, e.start_date, e.address, e.city, e.state,
e.zip, e.SSN, e.email, e.office_code, office.phone, office.office_name, ea.dept_code, ea.div_code, ea.projID,
ea.proj_startdate as start, ea.proj_enddate as end, salary.salary, salary.classification, salary.hourly_pay,
 project.proj_name
FROM employeepayroll as e
JOIN employeeassign as ea ON e.empID=ea.empID
JOIN office ON e.office_code =office.office_code
JOIN project ON ea.projID =project.projID
JOIN salary ON ea.title=salary.title
;
/* Create a new a project (with ID, name, budge, location, duration, etc.)  */
INSERT INTO project VALUES ('112', 'Disney Winter Development', null, 'FL ORLANDO', '10000000', null, null);
/* Assign the project manager and the project team  */
UPDATE project SET proj_manager = '105', start = '2023-1-1', end = '2024-1-1' WHERE projID = '112';
UPDATE employeeassign SET projID = '112', proj_startdate='2023-1-1', proj_enddate='2024-1-1'  WHERE empID =
'105';
UPDATE employeeassign SET projID = '112', proj_startdate='2023-1-1', proj_enddate='2024-1-1'  WHERE empID =
'7';
UPDATE employeeassign SET projID = '112', proj_startdate='2023-1-1', proj_enddate='2024-1-1'  WHERE empID =
'9';
UPDATE employeeassign SET projID = '112', proj_startdate='2023-1-1', proj_enddate='2024-1-1'  WHERE empID =
'10';
/* update projects */
SELECT * FROM
(SELECT projID, proj_name FROM project WHERE projID>0)
as p ORDER BY p.projID ASC;
```

```
/* update employees */
SELECT * FROM
(SELECT ea.empID, ep.name FROM employeeassign ea, employeepayroll ep WHERE ea.empID=ep.empID AND
ea.projID is null AND ea.title != 'project manager')
as e ORDER BY e.name ASC;

/* update managers */
SELECT * FROM (SELECT ea.empID, ep.name FROM employeeassign ea, employeepayroll ep WHERE
ea.empID=ep.empID AND ea.projID is null AND ea.title = 'project manager')
as e ORDER BY e.name ASC;

/*Track project status and mark up for each project -3 status, completed, ongoing and future   */
DROP view if exists projectSta;
CREATE VIEW  projectSta AS
(SELECT projID, year(end)-year(start) AS span, 'COMPLETED' as status, '100' as milestone FROM project WHERE
(CURDATE()>=end AND projID != '0')
UNION
SELECT projID, '0' AS span, 'FUTURE' as status, '0' as milestone FROM project WHERE (CURDATE()<=start AND
projID != '0')
UNION
SELECT projID, year(CURDATE())-year(start) AS span, 'ONGOING' as status, null milestone FROM project WHERE
(start<=CURDATE() AND CURDATE()<=end AND projID != '0')
);

/*Track man-hour and cost for each emp on the project, note that VP on overhead thus not included and
contractor on shorter durations   */
DROP view if exists projectEmpHr;
CREATE VIEW  projectEmpHr AS
(SELECT *, DATEDIFF(end, start)/7*40 AS workhour, DATEDIFF(end, start)/7*40*hourly_pay as charge FROM
empview2 WHERE (CURDATE()>=end AND projID != '0')
UNION
SELECT *, '0' AS workhour, '0' as charge FROM empview2 WHERE (CURDATE()<=start AND projID != '0')
UNION
SELECT *, DATEDIFF(end, start)/7*40 AS workhour, DATEDIFF(end, start)/7*40*hourly_pay as charge FROM
empview2 WHERE (start<=CURDATE() AND CURDATE()<=end AND projID != '0')
);

/* Track progress statistics, total man-hour and cost on ALL projects   */
DROP view if exists projectProgress;
CREATE VIEW projectProgress AS
SELECT * FROM
(SELECT p.projID, p.proj_name, p.proj_manager, p.location, p.budget, p.start, p.end, s.span, s.status, e.name,
COUNT(DISTINCT e.empID) as totalEmp, SUM(eh.charge) as totalCharge, SUM(eh.workhour) as totalworkhour,
SUM(eh.charge)/p.budget as prog_milestone
FROM project p, projectSta s, projectEmpHr eh, empview2 e
WHERE p.projID=s.projID AND p.proj_manager=e.empID AND p.projID=eh.projID
GROUP BY p.projID) as results
ORDER BY results.projID ASC;

/* Track progress statistics, total man-hour and cost on each project   */
```

```
SELECT * FROM projectProgress WHERE projID='105';

/* This is the end of Project Queries */
```

Appendix D Database Output Reports

1. Project Status Report for entire company 2010-2022
2. Sample Query Outputs for individual employees, departments, divisions. etc
3. IRS report for entire company 2010-2022