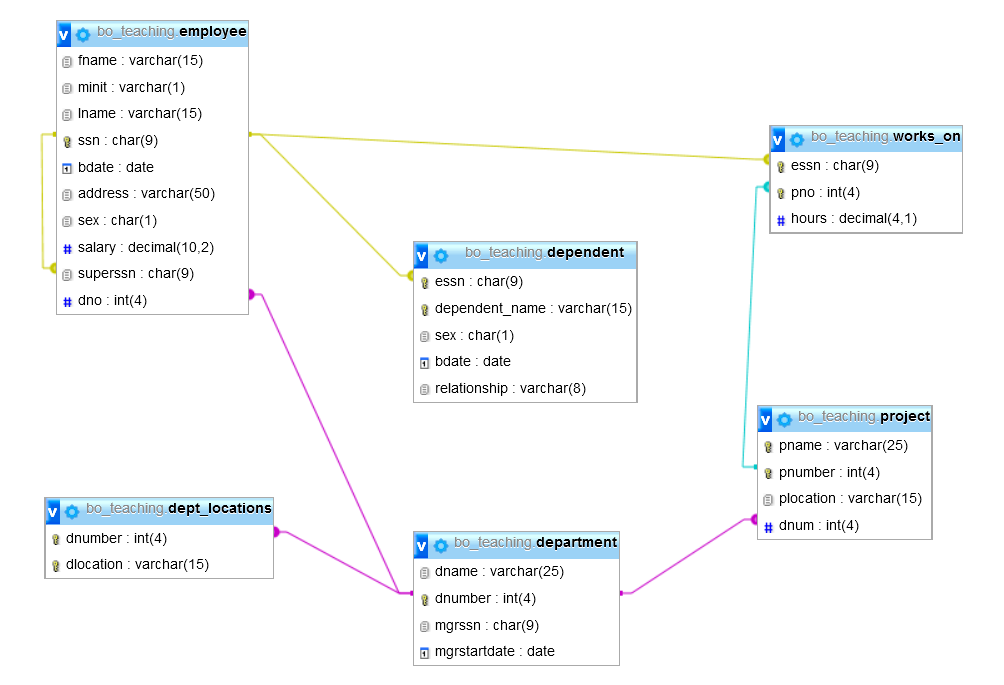
**CSC263 Database Systems**

**Laboratory Assignment #6**

**Given the following ERD for the COMPANY database,**

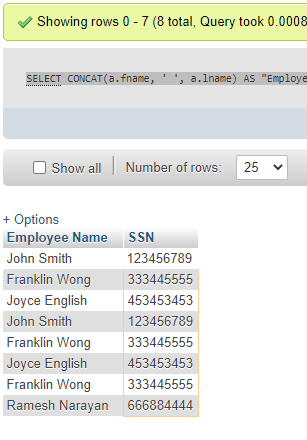


**Perform the following queries**

1. Retrieve the social security numbers of all employees who work on project numbers 1, 2, or 3.

Query used: “SELECT CONCAT(a.fname, ' ', a.lname) AS "Employee Name", a.ssn AS "SSN" FROM employee a, works\_on b WHERE a.ssn = b.essn AND b.Pno IN(1, 2, 3) “

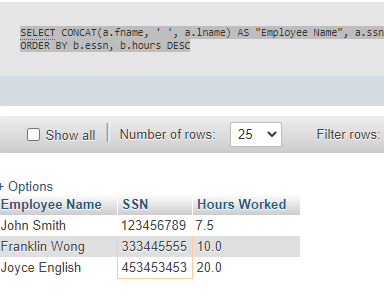
|  |  |
| --- | --- |
| Employee Name | SSN |



1. Retrieve all employees who are working on Project Number 2 and display them by their SSNs and the number of hours worked on Project 2.

Query Used “SELECT CONCAT(a.fname, ' ', a.lname) AS "Employee Name", a.ssn AS "SSN", b.hours AS "Hours Worked" FROM employee a,works\_on b WHERE a.ssn = b.essn AND b.Pno in(2) ORDER BY b.essn, b.hours DESC”

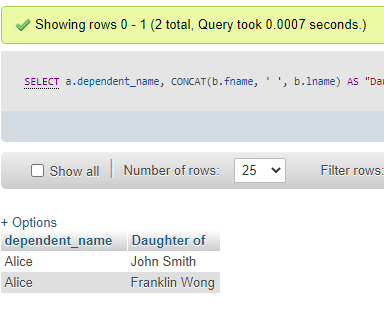
|  |  |  |
| --- | --- | --- |
| Employee Name | SSN | Hours Worked |



1. Retrieve all dependents who are daughters of some employees.

Query Used: SELECT a.dependent\_name, CONCAT(b.fname, ' ', b.lname) AS "Daughter of" FROM dependent a,employee b WHERE a.essn = b.ssn AND a.relationship LIKE "%Daughter%"”

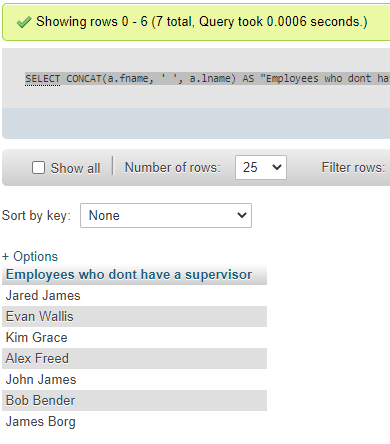
|  |  |
| --- | --- |
| Dependent Name | Daughter of |



1. Retrieve the names of all employees who do not have a supervisor.

Query Used: “SELECT CONCAT(a.fname, ' ', a.lname) AS "Employees who dont have a supervisor" FROM employee a WHERE a.superssn IS NULL”

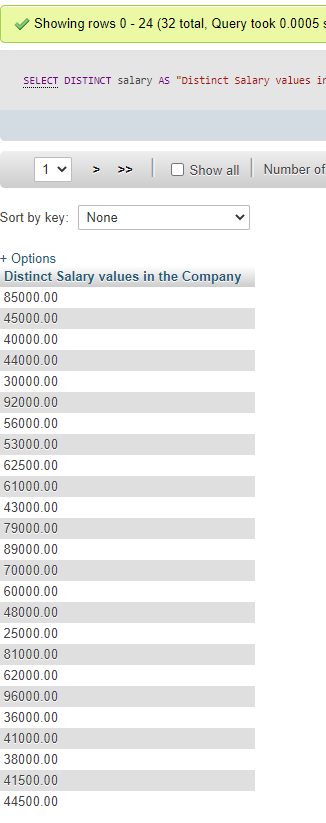
|  |
| --- |
| Employees who don’t have a supervisor |



1. Retrieve the distinct salary values.

Query Used “SELECT DISTINCT salary AS "Distinct Salary values in the Company" FROM employee”

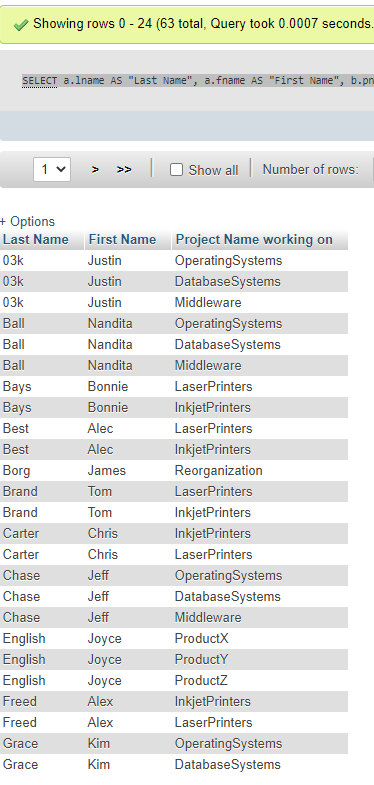
|  |
| --- |
| Distinct Salary values in the Company |



1. Ordered alphabetically by last name, first name of the employees, retrieve a list of employees and the projects they are working on.

Query Used “SELECT a.lname AS "Last Name", a.fname AS "First Name", b.pname AS "Project Name working on" FROM employee a, project b WHERE a.dno = b.dnum ORDER BY a.lname ASC”

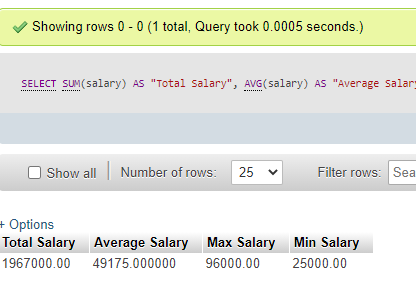
|  |  |  |
| --- | --- | --- |
| Last Name | First Name | Project Name working on |



1. Find the sum of the salaries of all employees, the maximum salary, the minimum salary, and the average salary.

Query Used “SELECT SUM(salary) AS "Total Salary", AVG(salary) AS "Average Salary", MAX(salary) AS "Max Salary", MIN(salary) AS "Min Salary" FROM employee”

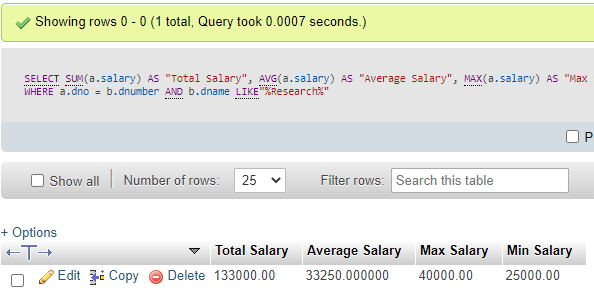
|  |  |  |  |
| --- | --- | --- | --- |
| Total Salary | Average Salary | Max Salary | Min Salary |



1. Find the sum of the salaries of all employees of the Research department, as well as the maximum salary, the minimum salary, and the average salary in this department.

Query Used “ SELECT SUM(a.salary) AS "Total Salary", AVG(a.salary) AS "Average Salary", MAX(a.salary) AS "Max Salary", MIN(salary) AS "Min Salary" FROM employee a,department b WHERE a.dno = b.dnumber AND b.dname LIKE"%Research%" ”

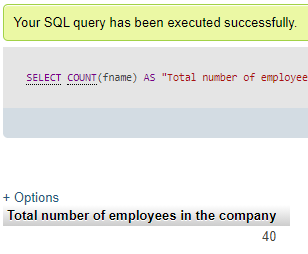
|  |  |  |  |
| --- | --- | --- | --- |
| Total Salary | Average Salary | Max Salary | Min Salary |



1. Retrieve the total number of employees in the company.

Query Used “SELECT COUNT(fname) AS "Total number of employees in the company" from employee”

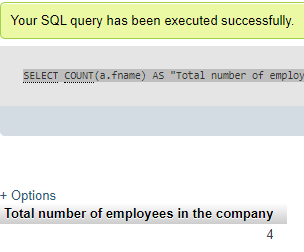
|  |
| --- |
| Total number of employees in the company |



1. Retrieve the total number of employees in the Research department.

Query Used “SELECT COUNT(a.fname) AS "Total number of employees in the company" from employee a, department b WHERE a.dno = b.dnumber and a.dno LIKE "%5%" “

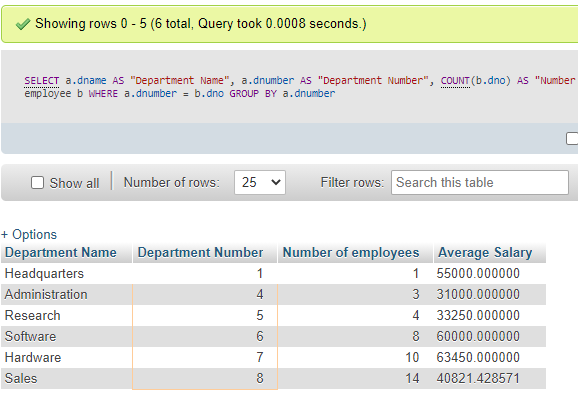
|  |
| --- |
| Total number of employees in the Research department |



1. For each department, retrieve the department number, the number of employees in the department, and their average salary.

Query Used “SELECT a.dname AS "Department Name", a.dnumber AS "Department Number", COUNT(b.dno) AS "Number of employees", AVG(b.salary) AS "Average Salary" FROM department a, employee b WHERE a.dnumber = b.dno GROUP BY a.dnumber”

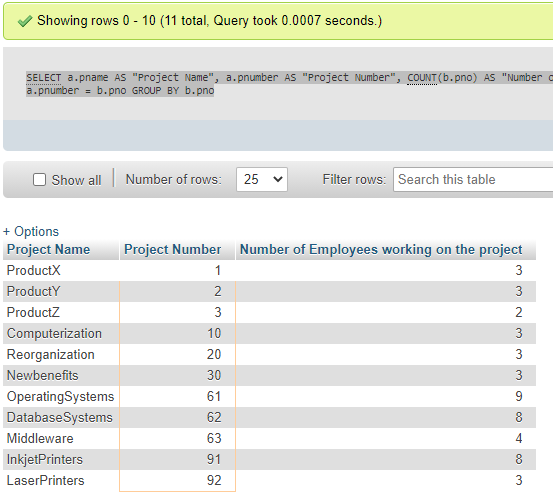
|  |  |  |  |
| --- | --- | --- | --- |
| Department Name | Department Number | Number of employees | Average Salary |



1. For each project, retrieve the project number, the project name, and the number of employees who work on that project.

Query Used “SELECT a.pname AS "Project Name", a.pnumber AS "Project Number", COUNT(b.pno) AS "Number of Employees working on the project" FROM project a, works\_on b WHERE a.pnumber = b.pno GROUP BY b.pno “

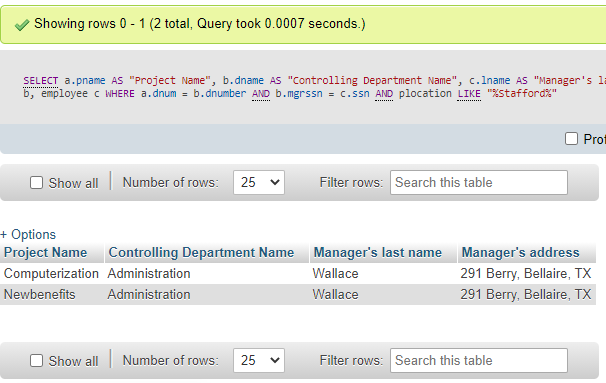
|  |  |  |
| --- | --- | --- |
| Project Name | Project Number | Number of Employees working on the project |



1. For every project located in Stafford, list the project name, the controlling department name, and the department manager's last name and address.

Query Used “SELECT a.pname AS "Project Name", b.dname AS "Controlling Department Name", c.lname AS "Manager's last name", c.address "Manager's address" FROM project a, department b, employee c WHERE a.dnum = b.dnumber AND b.mgrssn = c.ssn AND plocation LIKE "%Stafford%" ”

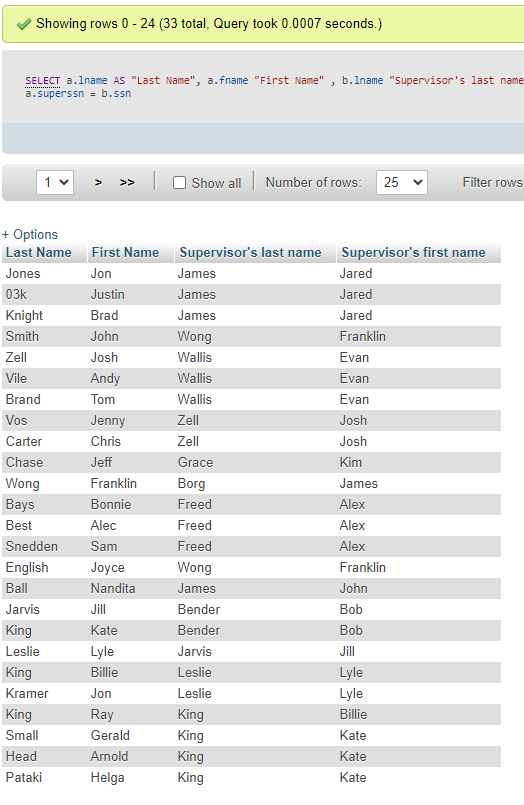
|  |  |  |  |
| --- | --- | --- | --- |
| Project Name | Controlling Department Name | Manager’s last name | Manager’s address |



1. For each employee, retrieve the employee's first and last name and the first and last name of his or her immediate supervisor.

Query Used “SELECT a.lname AS "Last Name", a.fname "First Name" , b.lname "Supervisor's last name", b.fname "Supervisor's first name" FROM employee a, employee b WHERE a.superssn = b.ssn”

|  |  |  |  |
| --- | --- | --- | --- |
| Last Name | First Name | Supervisor’s last name | Supervisor’s first name |

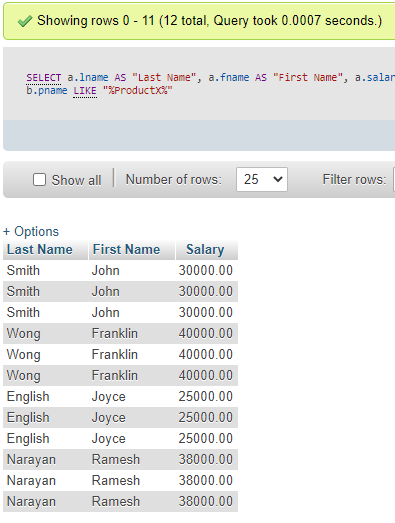


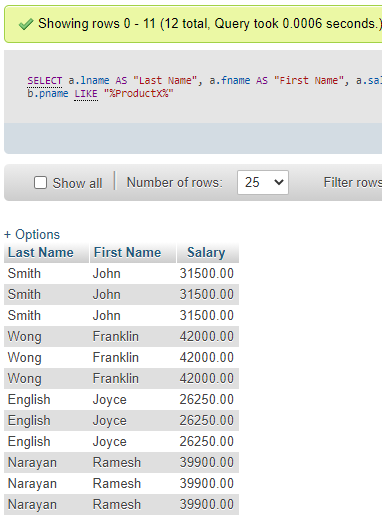
1. In the database, raise the salary of every employee who is working on the “ProductX” project by 5%. Show salary before and after the raise.

Before Query “SELECT a.lname AS "Last Name", a.fname AS "First Name", a.salary AS "Salary" FROM employee a, project b, works\_on c WHERE a.dno = b.dnum AND b.pnumber = c.pno AND b.pname LIKE "%ProductX%"”

Update Query “UPDATE employee set salary = salary\*1.05 WHERE dno LIKE "%5%" “

|  |  |  |
| --- | --- | --- |
| Last Name | First Name | Salary |

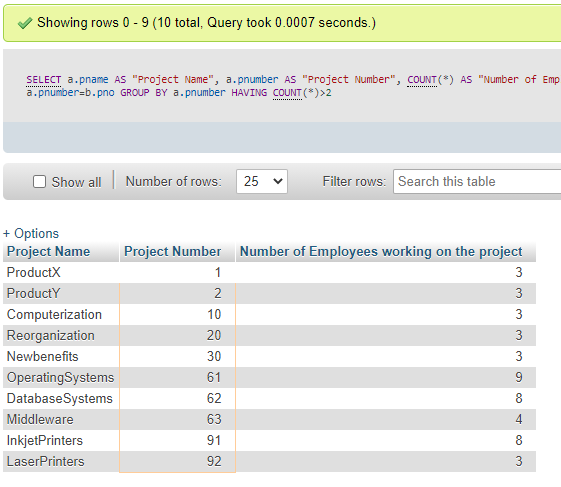




1. For each project on which more than two employees work, retrieve the project number, the project name, and the number of employees who work on the project.

Query Used “SELECT a.pname AS "Project Name", a.pnumber AS "Project Number", COUNT(\*) AS "Number of Employees working on the project" FROM project a, works\_on b WHERE a.pnumber=b.pno GROUP BY a.pnumber HAVING COUNT(\*)>2”

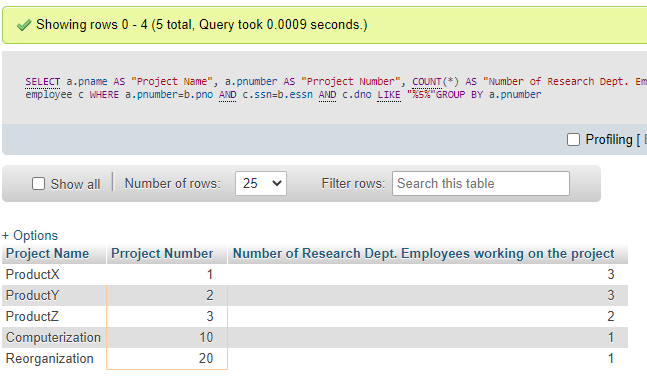
|  |  |  |
| --- | --- | --- |
| Project Name | Project Number | Number of Employees working on the project |



1. For each project, retrieve the project number, the project name, and the number of employees from “Research” department who work on the project.

Query Used “SELECT a.pname AS "Project Name", a.pnumber AS "Project Number", COUNT(\*) AS "Number of Research Dept. Employees working on the project" FROM project a, works\_on b, employee c WHERE a.pnumber=b.pno AND c.ssn=b.essn AND c.dno LIKE "%5%"GROUP BY a.pnumber”

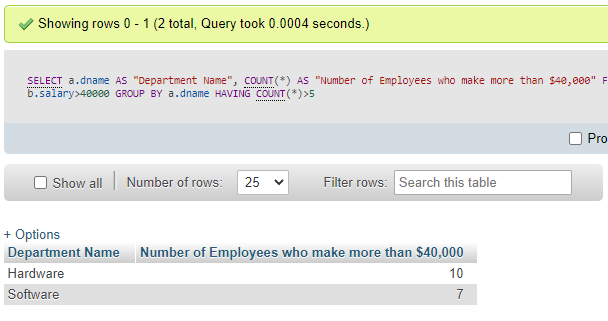
|  |  |  |
| --- | --- | --- |
| Project Name | Project Number | Number of Research Dept. Employees working on the project |



1. For each department that has more than five employees, retrieve the department number and the number of its employees who are making more than $40,000.

Query used “SELECT a.dname AS "Department Name", COUNT(\*) AS "Number of Employees who make more than $40,000" FROM department a, employee b WHERE a.dnumber=b.dno AND b.salary>40000 GROUP BY a.dname HAVING COUNT(\*)>5;”

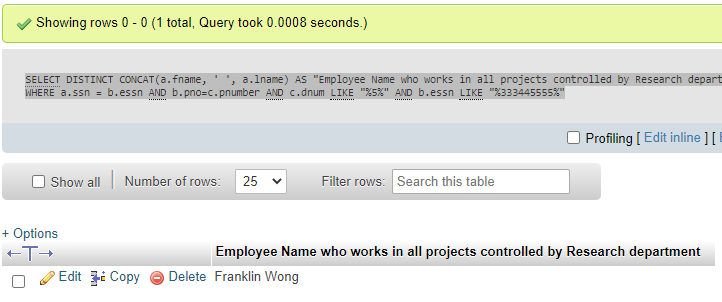
|  |  |
| --- | --- |
| Department Name | Number of Employees who make more than $40,000 |



1. Retrieve the names of each employee who works in all projects controlled by “Research” department.

Query Used “ SELECT DISTINCT CONCAT(a.fname, ' ', a.lname) AS "Employee Name who works in all projects controlled by Research department" FROM employee a, works\_on b, project c WHERE a.ssn = b.essn AND b.pno=c.pnumber AND c.dnum LIKE "%5%" AND b.essn LIKE "%333445555%"”

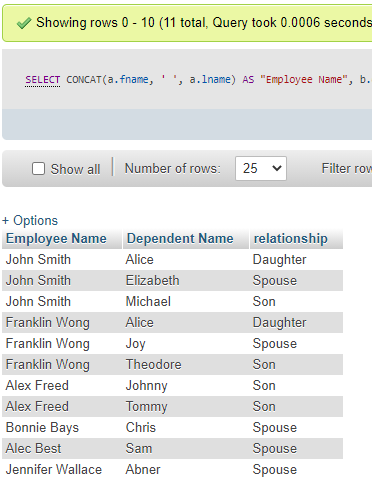
|  |
| --- |
| Employee Name who works in all projects controlled by Research department |



1. Retrieve dependent information in the database as required by the following result heading.

Query Used “SELECT CONCAT(a.fname, ' ', a.lname) AS "Employee Name", b.dependent\_name AS "Dependent Name", b.relationship FROM employee a, dependent b WHERE a.ssn = b.essn”

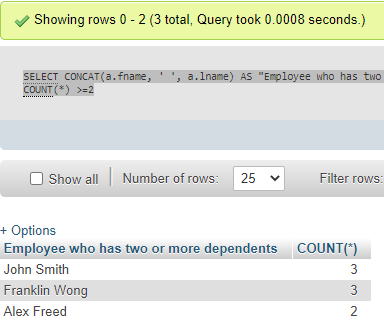
|  |  |  |
| --- | --- | --- |
| Employee Name | Dependent Name | Relationship |



1. Retrieve the names of all employees who have two or more dependents.

Query Used “SELECT CONCAT(a.fname, ' ', a.lname) AS "Employee who has two or more dependents", COUNT(\*) FROM employee a, dependent b WHERE a.ssn = b.essn GROUP BY a.ssn HAVING COUNT(\*) >=2”

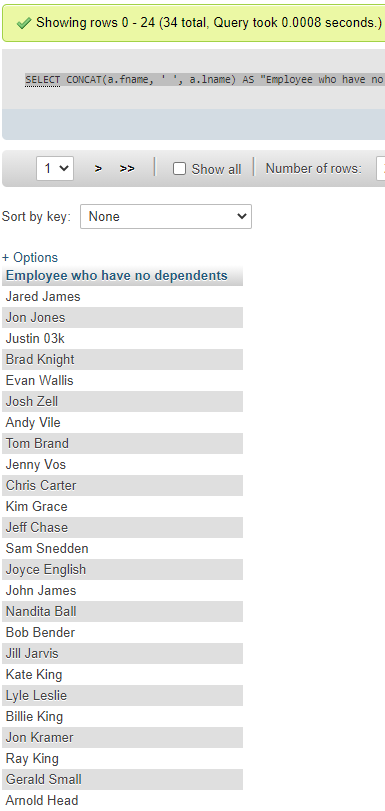
|  |
| --- |
| Employee who has two or more dependents |



1. Retrieve the names of employees who have no dependents.

Query Used “SELECT CONCAT(a.fname, ' ', a.lname) AS "Employee who have no dependents" FROM employee a WHERE NOT EXISTS (SELECT \*FROM dependent b WHERE a.ssn = b.essn)”

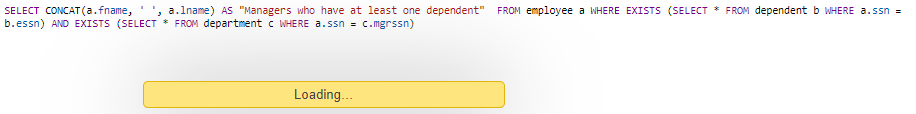
|  |
| --- |
| Employee who have no dependents |



1. List the names of managers who have at least one dependent.

Query Used “SELECT CONCAT(a.fname, ' ', a.lname) AS "Managers who have at least one dependent" FROM employee a WHERE EXISTS (SELECT \* FROM dependent b WHERE a.ssn = b.essn) AND EXISTS (SELECT \* FROM department c WHERE a.ssn = c.mgrssn)”

**QUERY SHOULD WORK HOWEVER MYPHP WILL NOT LOAD RESULTS**



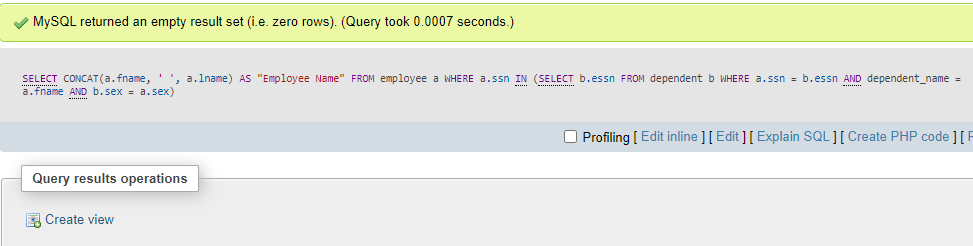
|  |
| --- |
| Managers who have at least one dependent. |

1. Retrieve the name of each employee who has a dependent with the same first name and same sex as the employee.

Query Used “SELECT CONCAT(a.fname, ' ', a.lname) AS "Employee Name" FROM employee a WHERE a.ssn IN (SELECT b.essn FROM dependent b WHERE a.ssn = b.essn AND dependent\_name = a.fname AND b.sex = a.sex)”

**THERE WERE NONE**

|  |  |  |
| --- | --- | --- |
| Employee Name | Dependent Name | Relationship |



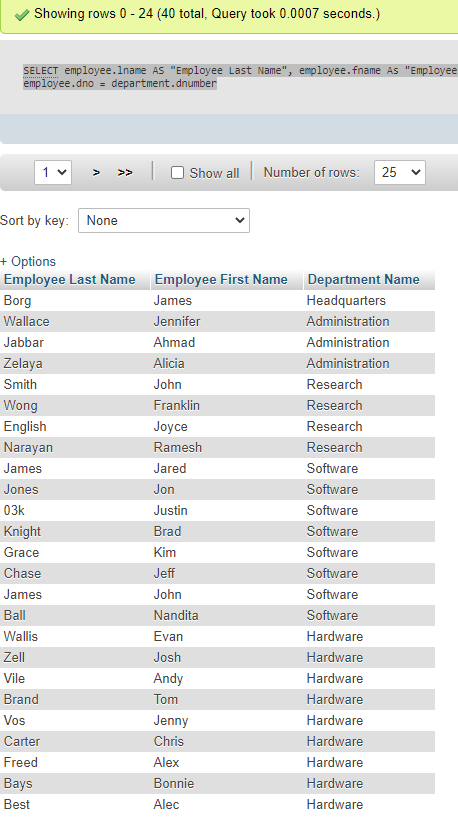
**JOIN, LEFT JOIN and RIGHT JOIN**

You must use JOIN, LEFT JOIN or RIGHT JOIN in the following queries.

1. Display ALL employees and the corresponding departments they work in. [You must use JOIN for this one.]

Query used “SELECT employee.lname AS "Employee Last Name", employee.fname As "Employee First Name", department.dname AS "Department Name" FROM employee RIGHT JOIN department ON employee.dno = department.dnumber “

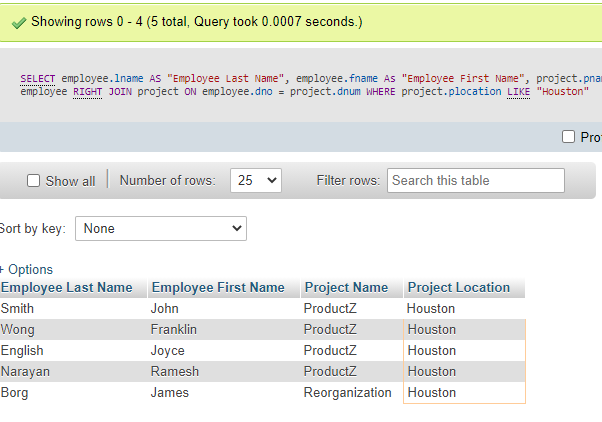
|  |  |  |
| --- | --- | --- |
| Employee Last Name | Employee First Name | Department Name |



1. Display all employees each is working on any project that is located in Houston. [You must use JOIN for this one.]

Query Used “SELECT employee.lname AS "Employee Last Name", employee.fname As "Employee First Name", project.pname AS "Project Name", project.plocation AS "Project Location" FROM employee RIGHT JOIN project ON employee.dno = project.dnum WHERE project.plocation LIKE "Houston"”

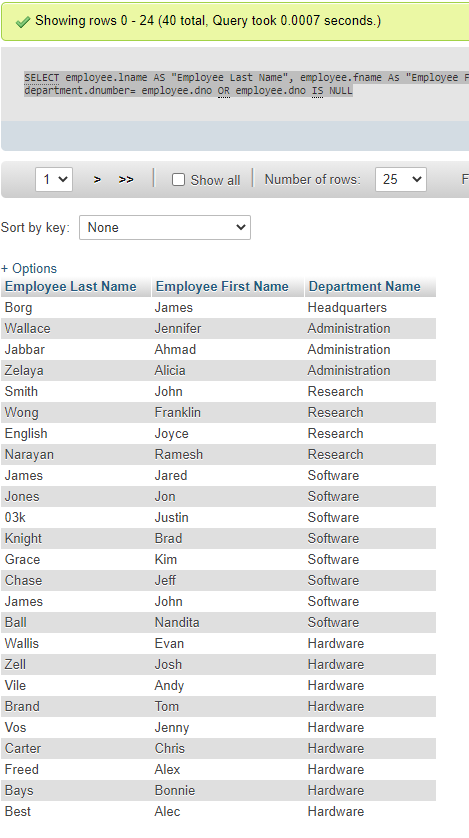
|  |  |  |  |
| --- | --- | --- | --- |
| Employee Last Name | Employee First Name | Project Name | Project Location |



1. Display ALL employees and the corresponding departments they work in. The list must show ALL employees including those who may have null information in their department work for. [You must use LEFT JOIN for this one.]

Query Used “SELECT employee.lname AS "Employee Last Name", employee.fname As "Employee First Name", department.dname AS "Department Name" FROM department LEFT JOIN employee ON department.dnumber= employee.dno OR employee.dno IS NULL”

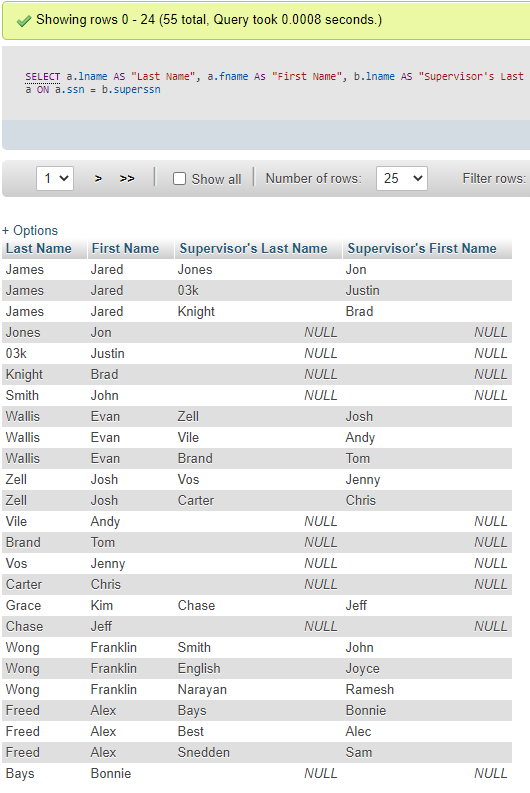
|  |  |  |
| --- | --- | --- |
| Employee Last Name | Employee First Name | Department Name |



1. For each employee, retrieve the employee's first and last name and the first and last name of his or her immediate supervisor. [You must use RIGHT JOIN for this one.]

Query Used “SELECT a.lname AS "Last Name", a.fname As "First Name", b.lname AS "Supervisor's Last Name", b.fname AS "Supervisor's First Name" FROM employee b RIGHT JOIN employee a ON a.ssn = b.superssn”

|  |  |  |  |
| --- | --- | --- | --- |
| Last Name | First Name | Supervisor’s last name | Supervisor’s first name |

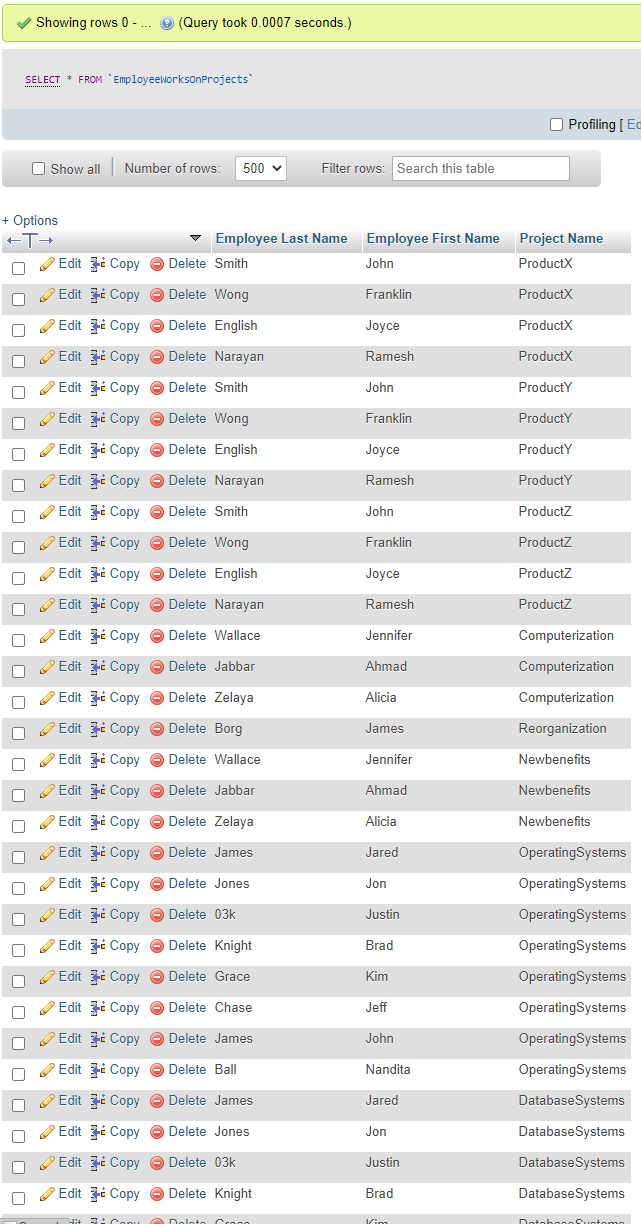


**Create VIEW and test VIEW**

1. Create and test a view EmployeeWorksOnProjects that has the following view headings. Make sure you show that view is indeed created in your database afterwards.

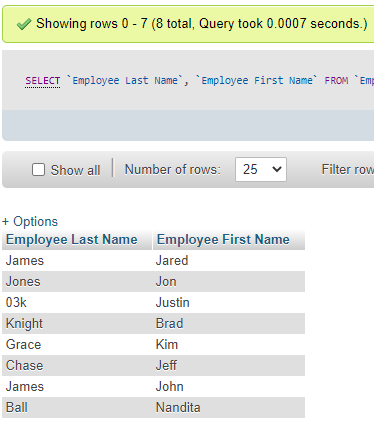
View Script “CREATE VIEW EmployeeWorksOnProjects AS SELECT a.lname AS "Employee Last Name", a.fname AS "Employee First Name", b.pname AS "Project Name" FROM employee a, project b WHERE a.dno =b.dnum”

|  |  |  |
| --- | --- | --- |
| Employee Last Name | Employee First Name | Project Name |

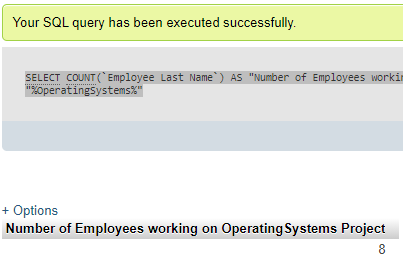


1. Test the view by using the view to perform the queries: (1) Display all employee who are working on “DatabaseSystems” project, and (2) How many employees are working on “OperatingSystems” project?
2. Query used “ SELECT `Employee Last Name`, `Employee First Name` FROM `EmployeeWorksOnProjects` WHERE `Project Name` LIKE "%DatabaseSystems%"”

|  |  |
| --- | --- |
| Employee Last Name | Employee First Name |



1. Query Used SELECT COUNT(`Employee Last Name`) AS "Number of Employees working on OperatingSystems Project" FROM `EmployeeWorksOnProjects` WHERE `Project Name` LIKE "%OperatingSystems%"



1. Create and test a view “SupervisorManagesEmployee” that has the following view headings. Make sure you show that view is indeed created in your database afterwards.

QUERY USED (NOT WORKING) ” CREATE VIEW SupervisorManagesEmployee AS SELECT b.lname "Supervisor's last name", b.fname "Supervisor's first name", COUNT(a.fname) AS "Number of employees Supervised" FROM employee a, employee b WHERE a.ssn = b.superssn GROUP BY b.lname”

|  |  |  |
| --- | --- | --- |
| Supervisor Last Name | Supervisor First Name | Number of Employees Supervised |

1. Test the view by using the view to perform the query: Which manager supervises the most number of employees?

|  |  |  |
| --- | --- | --- |
| Supervisor Last Name | Supervisor First Name | Number of Employees Supervised |

**Create TRIGGER and test TRIGGER**

1. Create and test a trigger named “employee\_salary\_change”.

Develop a trigger that logs each employee’s salary changes over time. In other words, the event when a change is made to an employee’s salary, it triggers the logging of such a change. The logging should show the employee’s name, employee ID, the current salary amount and the new salary amount, and the date and time this change is about to happen.

You need to create a new table, say, Employee\_Salary\_Audit, that keeps track all of above described information.

Show all necessary MySQL queries to make the above described task done. (1) Create the table, (2) Create the trigger, (3) Verify the trigger you created works by performing salary changes to at least (or more than) 3 different employees and twice salary changes to the same employee.

Table Creation CREATE TABLE `smoody`.`Employee\_Salary\_Audit` ( `Employee Name` VARCHAR(30) NOT NULL , `minit` VARCHAR(1) NOT NULL , `Current Salary` DECIMAL(10,2) NOT NULL , `New Salary` DECIMAL(10,2) NOT NULL , `Time` TIME NOT NULL , `Date` DATE NOT NULL ) ENGINE = InnoDB;

1. Because the original database does not store the date when a new employee joins the company, you are to create a trigger namely “employee\_joined” that causes the logging of the employee and his/her join date whenever a new employee tuple is inserted into the employee table. Create and test this trigger. When testing the trigger, you must make at least (or more) 3 new employees joining the company.