**Information filled by student:**

**Course basic information**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Credit Hours** | | | **Course Name** | **Code** |
| **Total** | **Practice** | **Lecture** | **DATA BASE** |  |
| **3** |  |  |

**Research Title**

**(Mini Store Management System)**

|  |
| --- |
| **Student Name: Mina Ashraf Zaki Fahmi anis** |
| **Student ID:1618120180100079** |
| **Level: two** |
| **Department :Faculty of Computers and Information** |

**Table of Content**

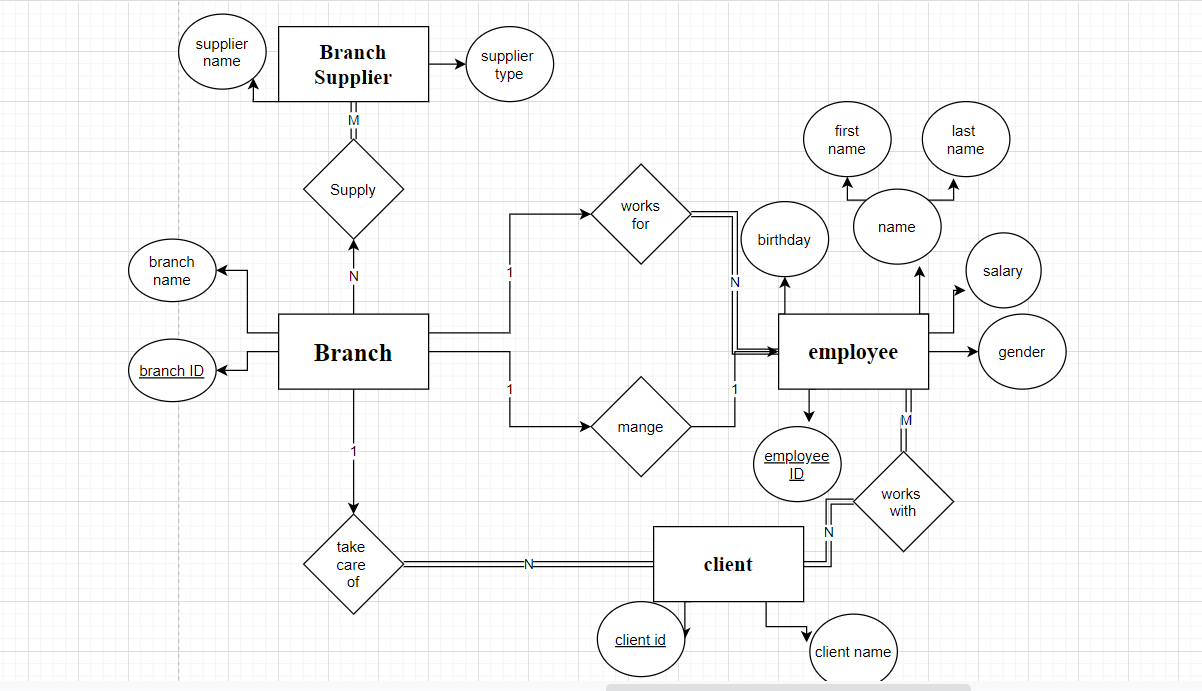
|  |  |
| --- | --- |
| Content | Page |
| Introduction | 2 |
| System Description | 2~3 |
| ER diagram | 3 |
| SQL Code | 3~5 |
| Insert | 5~6 |
| Queries and Select Statements | 6~7 |
| Count & Grouping | 7~8 |
| Sub-Query Statements | 8 |
| Joins | 8 |
| Update | 9 |
| Delete | 9 |
| Reference& Git-Hub Repository Link | 9 |
|  |  |

**Introduction**

**System Description**

This is mini store management system which handle any store and its branches around the countries in Egypt , It has a lot of benefits for people who work in store and for client the enter and also for the suppliers that supply the store so that the manger can handle everything around him. It give each employee that work an ID and handles there personal information and how much salary they take in a year and which employee supervise the other also it handle manger of each branch and connect between the manger table and employee table and also each branch has its own ID and its manger which is chosen by the supervisor and the manger should be one of the employee and the starting date of the each manger also it handle the client information that enter every branch and give them ID and there name should be in and which employee have helped them in there sales and which branch they have entered to do there sales also it handle the suppliers and give you the information about there supply and the supplier name and which branch have been supplied with the product ,The last table in this system is the sales Table which provide the total sales which each client have done and which employee have helped them and in which branch that he have spend his money

**ER Diagram**

****

**Theoretical analysis and discussion**

**[SQL Code]**

**-- Employees Table:**

CREATE TABLE employee (

emp\_id INT PRIMARY KEY,

first\_name VARCHAR(40),

last\_name VARCHAR(40),

birth\_day DATE,

gender VARCHAR(1),

salary INT,

super\_id INT,

branch\_id INT

);

**-- Branches of Store:**

CREATE TABLE branch (

branch\_id INT PRIMARY KEY,

branch\_name VARCHAR(40),

manger\_id INT,

manger\_start\_date DATE,

FOREIGN KEY(manger\_id) REFERENCES employee(emp\_id) ON DELETE SET NULL

);

**-- Making the branch id and supervisor id Foreign keys and the manger id is a foreign key since manger is on of the employees**

ALTER TABLE employee

ADD FOREIGN KEY(branch\_id)

REFERENCES branch(branch\_id)

ON DELETE SET NULL;

ALTER TABLE employee

ADD FOREIGN KEY(super\_id)

REFERENCES employee(emp\_id)

ON DELETE SET NULL;

**-- Client table:**

CREATE TABLE client (

client\_id INT PRIMARY KEY,

client\_name VARCHAR(40),

branch\_id INT,

FOREIGN KEY(branch\_id) REFERENCES branch(branch\_id) ON DELETE SET NULL

);

**-- Sales table**

CREATE TABLE Sales (

emp\_id INT,

client\_id INT,

total\_sales INT,

PRIMARY KEY(emp\_id, client\_id),

FOREIGN KEY(emp\_id) REFERENCES employee(emp\_id) ON DELETE CASCADE,

FOREIGN KEY(client\_id) REFERENCES client(client\_id) ON DELETE CASCADE

);

**-- Supplier of the branch**

CREATE TABLE branch\_supplier (

branch\_id INT,

supplier\_name VARCHAR(40),

supply\_type VARCHAR(40),

PRIMARY KEY(branch\_id, supplier\_name),

FOREIGN KEY(branch\_id) REFERENCES branch(branch\_id) ON DELETE CASCADE

);

**-- Data insertion**

**-- alex**

INSERT INTO employee VALUES(100, 'ahmed', 'mohmed', '1950-10-16', 'M', 250000, NULL, NULL);

INSERT INTO branch VALUES(1, 'alexandria', 100, '2005-01-08');

UPDATE employee

SET branch\_id = 1

WHERE emp\_id = 100;

INSERT INTO employee VALUES(101, 'noha', 'ahmed', '1960-03-12', 'F', 110000, 100, 1);

-- cairo

INSERT INTO employee VALUES(102, 'Michael', 'amgad', '1974-04-15', 'M', 75000, 100, NULL);

INSERT INTO branch VALUES(2, 'cairo', 102, '1992-04-06');

UPDATE employee

SET branch\_id = 2

WHERE emp\_id = 102;

INSERT INTO employee VALUES(103, 'mariam', 'shawky', '1981-06-25', 'F', 63000, 102, 2);

-- matroh

INSERT INTO employee VALUES(106, 'mahmoud', 'reda', '1959-02-05', 'M', 78000, 100, NULL);

INSERT INTO branch VALUES(3, 'matroh', 106, '1998-02-13');

INSERT INTO branch VALUES(4, "mansora", NULL, NULL);

UPDATE employee

SET branch\_id = 3

WHERE emp\_id = 106;

INSERT INTO employee VALUES(107, 'mina', 'ashraf', '1973-07-22', 'M', 65000, 106, 3);

INSERT INTO branch\_supplier VALUES(2, 'rizk', 'mobile');

INSERT INTO branch\_supplier VALUES(1, 'apple', 'pens');

INSERT INTO branch\_supplier VALUES(3, 'samsung', 'electronic devices');

INSERT INTO client VALUES(400, 'ziad', 2);

INSERT INTO client VALUES(401, 'rwan', 2);

INSERT INTO client VALUES(402, 'gaber', 3);

INSERT INTO `sales` VALUES (100, 402, 55000);

INSERT INTO sales VALUES(103, 400, 267000);

INSERT INTO `sales` VALUES (107, 401, 22500);

**\*Select Statement Using Different Functions**

**To change employee last name**

SELECT REPLACE(employee.last\_name,"ashraf", "Zaki") FROM employee

**How to Find all employees ordered by salary(ascending)**

SELECT \* from employee ORDER BY salary ASC

**How to Find all employees ordered by gender**

SELECT \* from employee ORDER BY gender;

**Finding the first 2 employees in the table**

SELECT \* from employee LIMIT 2;

**To find all gender in the table**

SELECT DISTINCT gender FROM employee;

**Find all employees at specific branch:**

SELECT \* FROM employee WHERE branch\_id = 2;

**Find the number of employees**

SELECT COUNT(emp\_id) FROM employee;

**Find the average of all employee's salaries**

SELECT AVG(salary) FROM employee;

**Find the sum of all employee's salaries**

SELECT SUM(salary) FROM employee;

**Find specific gender employees at specific branch**

SELECT \* FROM employee WHERE branch\_id = 2 AND gender = 'F';

**Find all employees born between two specified years**

SELECT \* FROM employee WHERE birth\_day BETWEEN '1970-01-01' AND '1975-01-01';

**Find any client with specific letter**

SELECT \*FROM client WHERE client\_name LIKE '%b(or any letter)%;

**To know the name of day which the manger start work**

SELECT DAYNAME(manger\_start\_date) from branch

**To know the age of employee**

SELECT ROUND (DATEDIFF(CURRENT\_DATE , employee.birth\_day) / 365) AS age FROM employee

**To Know the Greatest salary in employees**

SELECT first\_name,MAX(salary) FROM employee

**To Know the Greatest salary in employees**

SELECT first\_name,MIN(salary) FROM employee

**To Find the full name of Employee**

SELECT CONCAT(first\_name," ",last\_name) FROM employee

**To get the nickname of every branch name**

SELECT SUBSTRING(branch\_name,1 , 4) FROM branch

**Arranging sales from greatest to the least**

SELECT \* from sales ORDER BY total\_sales DESC

**Select Random Client to raise his salary**

SELECT emp\_id, first\_name FROM employee WHERE emp\_id = floor(RAND()\*(107-101)+100) LIMIT 1

**\*Select Using Counts&Group**

**Find out how many males and females there are**

SELECT COUNT(gender), gender FROM employee GROUP BY gender

**Find the total sales of each employee**

SELECT SUM(total\_sales), emp\_id FROM Sales GROUP BY client\_id;

**Find the total amount of money spent by each client**

SELECT SUM(total\_sales), client\_id FROM sales GROUP BY client\_id;

**\*Select Using Sub Query**

**Find names of all employees who have sold over 50,000**

SELECT employee.first\_name, employee.last\_name FROM employee WHERE employee.emp\_id IN (SELECT sales.emp\_id FROM sales WHERE sales.total\_sales > 50000);

**Find all clients who are handles by a specific manger ID**

SELECT client.client\_id, client.client\_name FROM client WHERE client.branch\_id = (SELECT branch.branch\_id FROM branch WHERE branch.manger\_id = 102);

-- **Find the names of employees who work with clients handled by branch ID**

SELECT employee.first\_name, employee.last\_name FROM employee WHERE employee.emp\_id IN ( SELECT sales.emp\_id FROM sales ) AND employee.branch\_id = 2;

**\*Join statements**

**To find the manger of each branch**

SELECT employee.emp\_id , employee.first\_name , branch.branch\_name FROM employee JOIN branch ON emp\_id = branch.branch\_id

**To get the employees that are not branch’s manger**

SELECT employee.emp\_id, employee.first\_name, branch.branch\_name FROM employee LEFT JOIN branch ON employee.emp\_id = branch.manger\_id

**To get the Branches That don’t have mangers**

SELECT employee.emp\_id, employee.first\_name, branch.branch\_name FROM employee RIGHT JOIN branch ON employee.emp\_id = branch.manger\_id

**To find the name of employees that help in sales**

SELECT employee.emp\_id , employee.first\_name , sales.total\_sales FROM employee JOIN sales ON employee.emp\_id = sales.emp\_id

**To get the name of branches that have clients**

SELECT client.client\_name , branch.branch\_name FROM client JOIN branch ON branch.branch\_id = client.branch\_id

**\*Update Statements:**

**To change the first name**

UPDATE employee SET first\_name = "morkos" where emp\_id =102

**To increase the salary of the employee by specific percentage**

UPDATE employee SET salary = salary+(salary\*0.1)

**To Change the supervisor of the Employees**

UPDATE employee SET super\_id = 107 where super\_id = 100

**To give Discount for total sales**

UPDATE sales SET total\_sales = total\_sales-(total\_sales\*0.2)

**\*Delete Statement:**

**To remove a client from your data base with ID**

DELETE FROM `client` WHERE `client`.`client\_id` = 400;

**To remove and client from your data base with name**

DELETE FROM `client` WHERE `client`.`client\_name` = “ziad”;

**To remove Client that has less than sales that you want**

DELETE FROM sales where total\_sales <20000

**To remove every male in the employee**

DELETE FROM employee WHERE gender = 'M'

**To remove client that has more that specific ID**

DELETE FROM client where client\_id > 401

**References**

[1] SQL Tutorial

[2]W3School.com

[3]Microsoft SQL Documentation Manual