ITC3160 A1

Dr. Ioannis Christou

Vasilis Pavlidis

Alexandros Synetos Konstantinidis

13/6/2022

Midterm Assignment, Coursework

Download and Run the script **WorkersDatabase.sql** in your database, in order to create the required tables. Then write the commands to accomplish the tasks below:

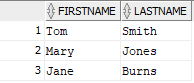
/\* 1. Display the names of all workers in the accounting department.

Use the ‘Accounting’ as a condition and not the department id. \*/

Select FIRSTNAME,LASTNAME

FROM WORKER

WHERE DEPARTMENTID = 1;



/\*2. Find the details of any project with the word “urn” anywhere in its name.

SELECT \* FROM PROJECT WHERE projName LIKE '%urn%';

Graphical user interface, text, application, Word

Description automatically generated

/\*3.Assuming that salary contains annual salary, find each worker’s ID,

name, and monthly salary. \*/

Select EMPID , FIRSTNAME, LASTNAME,SALARY/12

FROM WORKER; Table

Description automatically generated

/\*4. Display an alphabetical list of names of all workers assigned to project 1001, sorted by last name.

SELECT lastName, firstName FROM WORKER

WHERE empID IN(SELECT empID FROM ASSIGN WHERE projNo LIKE '1001')

ORDER BY lastName ASC;

Graphical user interface, text, application

Description automatically generated

/\*5. Display the name of the employee in the research department

who has the lowest salary. \*/

SELECT FIRSTNAME,LASTNAME

FROM WORKER

WHERE salary = (select MIN(salary) from WORKER);



/\*6. Display details of the project with the highest budget.

SELECT \* FROM PROJECT WHERE budget=(SELECT MAX(budget) FROM PROJECT);

Graphical user interface, application, Word

Description automatically generated

/\*7.Display the names and departments of all workers on project 1019. \*/

Select FIRSTNAME, LASTNAME, DEPARTMENTID

FROM WORKER

WHERE empID IN(SELECT empID FROM ASSIGN WHERE PROJNO LIKE '1019');

/\*8. Display an alphabetical list of names and corresponding ratings of all workers on any project that is managed by Michael Burns. Use ‘Michael and ‘Burns’ as conditions.

SELECT firstName, lastName, rating FROM WORKER, ASSIGN WHERE projNo IN(SELECT projNo FROM PROJECT WHERE projmgrid IN(SELECT empID FROM WORKER

WHERE firstName LIKE 'Michael' AND lastName LIKE 'Burns')) ORDER BY lastName ASC;

Table

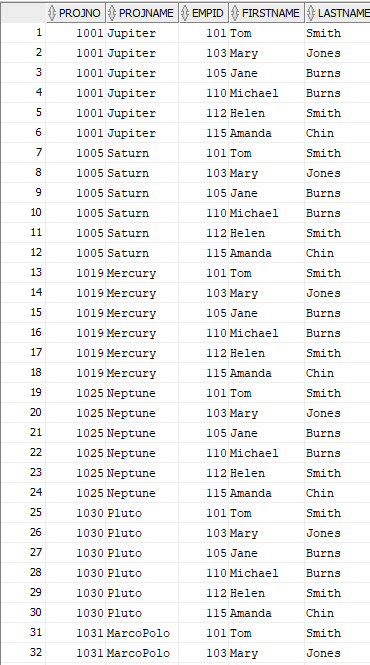
Description automatically generated

/\*9. Create a view that has project number and name of each project,

along with the IDs and names of all workers assigned to it. \*/

CREATE VIEW Overview

AS SELECT projno, projName, Empid, firstname, lastname FROM PROJECT, WORKER;



/\*10. Using the view created in Exercise 7, find the project number and project name of all projects to which employee 1001 is assigned.

SELECT projno, projname, empid FROM overview2

WHERE EMPID = '101'; (IN THE CASE THAT WE DIPLSAY BASED ON THE EMPLOYEE WITH ID 101)

Table, Excel

Description automatically generated

SELECT projno, projname FROM overview2

WHERE empid IN(SELECT empid FROM Assign WHERE projno LIKE '1001');) (IN THE CASE THAT WE DISPLAY THE EMPLOYEES AND THEIR PROJECTS WHO HAVE WOPRKED ON PROJECT 1001 AS WELL)

A picture containing table

Description automatically generated

/\*11. Add new workers, named with your names and your IDs to the research department.

Assign your selves into 2 projects each. Select everything from the view you created in Exercise 7. \*/

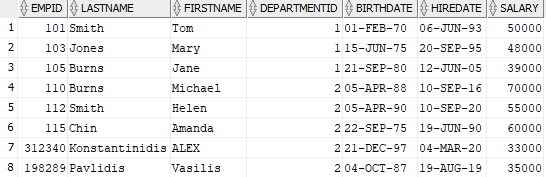
insert into worker values (312340, 'Konstantinidis', 'ALEX', 2,'21/DEC/97', '04/MAR/20', 33000);

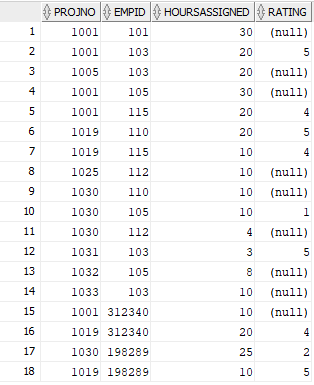
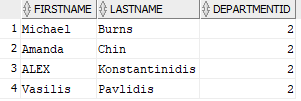
insert into worker values (198289, 'Pavlidis', 'Vasilis', 2,'04/OCT/87', '19/AUG/19', 35000);

insert into assign values(1001, 312340, 10, null);

insert into assign values(1019, 312340, 20, 4);

insert into assign values(1030, 198289, 25, 2);

insert into assign values(1019, 198289, 10, 5);



Bellow Query number 7 when done again.

/\*12. Change the hours, which employee 110 is assigned to project 1019, from 20 to 10.

UPDATE ASSIGN SET hoursAssigned=10 WHERE projNo IN

(SELECT projNo FROM ASSIGN WHERE empID=110 AND projNo=1019);

Graphical user interface, text, application

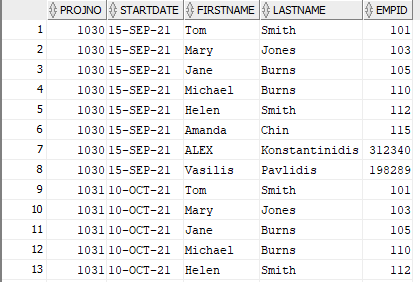
Description automatically generated

/\*13. For all projects starting after Sep 1, 2021, find the project number and the IDs and names of all workers assigned to them. \*/

SELECT projNo, startDate, firstname, lastname, empid FROM PROJECT, WORKER

WHERE startDate IN(Select startDate FROM PROJECT

GROUP BY startDate HAVING startdate > ('01/SEP/21'));



/\*14. For each project, list the project number, how many workers are assigned to it and how many hours they are assigned for.

SELECT projNo, COUNT(empID)AS workers, SUM(hoursAssigned) AS hours

FROM ASSIGN GROUP BY projNo;

Table

Description automatically generated

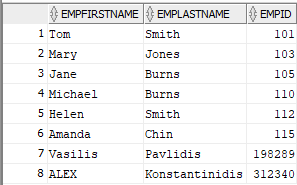
/\*15. Find the employee names and department manager names of all workers who are not assigned to any project. \*/

SELECT firstname AS EMPFIRSTNAME, lastname AS EMPLASTNAME, firstname AS MGRFIRSTNAME, lastname AS MGRLASTNAME FROM WORKER

WHERE empid IN(SELECT empid FROM assign

WHERE projno IS NOT NULL);

/\* We manually checked the tables and saw that the result was corrent since there were no unassigned people on any projects. \*/



/\*16. Display a list of project numbers and names and starting dates of all projects that have the same starting date.

SELECT projNo, projName, startDate FROM PROJECT

WHERE startDate IN(Select startDate FROM PROJECT

GROUP BY startDate HAVING COUNT(\*)>1);

Graphical user interface, text, application

Description automatically generated

/\*17. Add a field called status to the Project table (Sample values for this field are active, completed, planned, cancelled).

Update the Projects table and make some of them active, one completed and one cancelled. Display a list of all ‘active’ projects. \*/

ALTER TABLE project ADD status VARCHAR(20);

update project set status = 'planned';

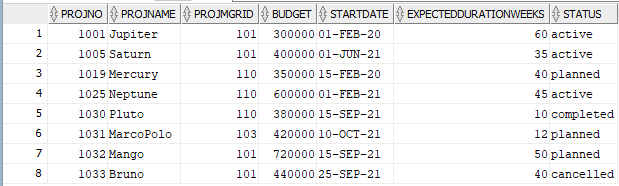
update project set status = 'completed' WHERE Projno = 1030;

update project set status = 'cancelled' WHERE Projno = 1033;

update project set status = 'active' WHERE Projno = 1001;

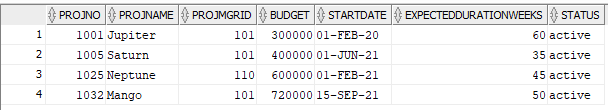
update project set status = 'active' WHERE Projno = 1005;

update project set status = 'active' WHERE Projno = 1025;

update project set status = 'active' WHERE Projno = 1032;

Select \* FROM PROJECT

WHERE status = 'active';



/\*18. Display the employee ID and project number of all employees who have no ratings on that project.

SELECT empID, projNo FROM ASSIGN WHERE rating IS null;

Table

Description automatically generated

/\*19. Add a field called numEmployeesAssigned to the Project table.

Use the UPDATE command to insert values into the field to correspond to the current information in the Assign table. \*/

ALTER TABLE project ADD numEmployeesAssigned VARCHAR(5);

update project set numEmployeesAssigned = 5 WHERE Projno = 1001;

update project set numEmployeesAssigned = 1 WHERE Projno = 1005;

update project set numEmployeesAssigned = 4 WHERE Projno = 1019;

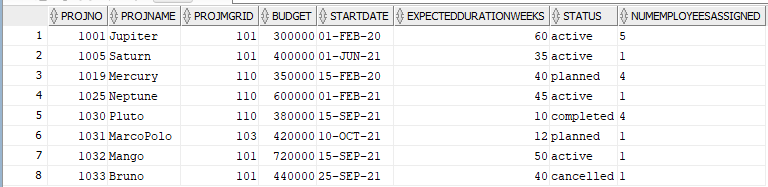
update project set numEmployeesAssigned = 1 WHERE Projno = 1025;

update project set numEmployeesAssigned = 4 WHERE Projno = 1030;

update project set numEmployeesAssigned = 1 WHERE Projno = 1031;

update project set numEmployeesAssigned = 1 WHERE Projno = 1032;

update project set numEmployeesAssigned = 1 WHERE Projno = 1033;



/\*20. Write a trigger that will update the numEmployeesAssigned field correctly whenever an assignment is made, dropped, or updated.

CREATE TRIGGER Employee\_Trigger

ON PROJECT

AFTER MAKE, UPDATE, DROP

AS DECLARE @projNo NUMBER, @projName VARCHAR(20),

@progMgrid NUMBER, @budget NUMBER, @startDate datetime NOT NULL DEFAULT GETDATE(),

@expectedDurationWeeks NUMBER, @status VARCHAR(10), @numEmployeesAssigned VARCHAR(5) ;

--MAKE

IF EXISTS(SELECT \* FROM inserted) AND NOT EXISTS(SELECT \* FROM deleted)

BEGIN

SELECT @numEmployeesAssigned=numEmployeesAssigned FROM inserted i;

INSERT INTO PROJECT(projNo, projName, projMgrid, budget, startDate, expectedDurationWeeks, status, numEmployeesAssigned)

VALUES(@projNo, @projName, @projMgrid, @budget, @startDate, @expectedDurationWeeks, @status, @numEmployeesAssigned);

END

--UPDATE

IF EXISTS(SELECT \* FROM inserted) AND EXISTS(SELECT \* FROM deleted)

BEGIN

SELECT @numEmployeesAssigned=numEmployeesAssigned FROM inserted i;

INSERT INTO PROJECT(projNo, projName, projMgrid, budget, startDate, expectedDurationWeeks, status, numEmployeesAssigned)

VALUES(@projNo, @projName, @projMgrid, @budget, @startDate, @expectedDurationWeeks, @status, @numEmployeesAssigned);

END

--DROP

IF EXISTS(SELECT \* FROM inserted) AND NOT EXISTS(SELECT \* FROM deleted)

BEGIN

SELECT @numEmployeesAssigned=numEmployeesAssigned FROM deleted i;

INSERT INTO PROJECT(projNo, projName, projMgrid, budget, startDate, expectedDurationWeeks, status, numEmployeesAssigned)

VALUES(@projNo, @projName, @projMgrid, @budget, @startDate, @expectedDurationWeeks, @status, @numEmployeesAssigned);

END;

Graphical user interface, text, application, email

Description automatically generated