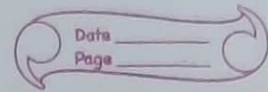


2020 Fall



2b i) Create a table named Automotor with chasis-number as primary key and following attributes:  
veh-brand, veh-name, veh-model, veh-year, veh-cost, veh-color, veh-weight.

→ CREATE TABLE Automotor (  
chasis-number INT PRIMARY KEY,  
veh-brand VARCHAR(255),  
veh-name VARCHAR(25),  
veh-model VARCHAR(100),  
veh-year ~~INT~~ DATE,  
veh-cost DECIMAL(10,2),  
veh-color VARCHAR(20),  
veh-weight DECIMAL(10,2));

ii) Enter a full detailed information of an automotor.

→ INSERT INTO Automotor VALUES  
(199278, 'Royce Roll', 'Phantom', '628G', '2017/08/27', '278658.25',  
'Purple', 586.92);

iii) Change any Automotor's year to 2019

→ UPDATE Automotor SET veh-year = 2019/08/27  
WHERE chasis-number = 199278;

iv) Remove all Automotor records whose model contains character 'i' in last position.

→ DELETE FROM Automotor  
WHERE veh-model LIKE '%i';

(v) Display the total cost of all vehicles of the table Automotor.

→ SELECT SUM(veh-cost) AS total-cost  
FROM Automotor;

(vi) Create a view from above table having vehicles only red color.

→ CREATE VIEW RedVehicles AS  
SELECT \* FROM Automotor  
WHERE veh-color = 'red';

(vii) Display details of Automotor ordering in descending manner by brand name and in ascending order on model when brand matches.

→ SELECT \* FROM Automotor  
ORDER BY veh-name DESC, veh-model ASC;

(viii) Change data type of color so that it only takes one character.

→ ALTER TABLE Automotor  
ALTER COLUMN veh-color SET DATA TYPE CHAR(1);



2b

Doctor (Name, Age, Address)

Works (Name, Dept-no., Salary)

Department (Depart-no, Dept-name, Floor, Room)

Write SQL statement.

(i) Display the name of doctor who do not work in any department.

→ SELECT d.Name FROM Doctor d

LEFT JOIN Works W ON d.Name = W.Name

WHERE W.Name IS NULL;

(ii) Modify the database so that Dr. Hari lives in Pokhara.

→ UPDATE Doctor SET Address = "Pokhara"

WHERE Name = "Hari";

(iii) Delete all records of Doctor working in OPD department.

→ DELETE FROM Works

WHERE Dept-no = (SELECT Dept-no FROM Department  
WHERE Dept-name = "OPD");

(iv) Delete the name of Doctors who work in at least two departments.

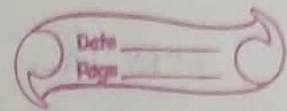
→ SELECT d.name FROM doctors d

INNER JOIN Works W ON d.name = W.Name

GROUP BY d.name

HAVING COUNT (DISTINCT W. Dept-no.) &gt;= 2;

2016 Spring



3 b. Doctors (DoctorID, DoctorName, Department, Address, Salary)  
Patients (PatientID, PatientName, Address, Age, Gender)  
Hospitals (PatientID, DoctorID, HospitalName, Location)

Write a SQL statement:

i. Display ID of patient admitted to hospital at Pokhara and whose name ends with 'a'.

→ SELECT p.patientID FROM patients p  
JOIN Hospitals ON p.patientID = h.patientID  
WHERE h.location = "Pokhara" AND p.patientName LIKE "%a";

ii. Delete the record of Doctors whose salary is greater than average salary of doctors.

→ DELETE FROM Doctors  
WHERE salary > (SELECT AVG(salary) FROM Doctors);

iii. Increase the salary of doctors by 18.5% who works in OPD department.

→ SELECT FROM Doctors SET Salary = Salary \* 1.185  
WHERE Department = "OPD";

iv. Find the average salary of Doctors for each address who have average salary more than 55k.

→ SELECT Address, AVG(Salary) AS AvgSalary  
FROM Doctors GROUP BY Address  
HAVING AVG(Salary) > 55000;

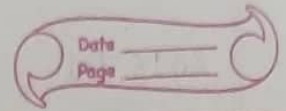


2b. Teacher (TeacherID, TeacherName, Office)

Write SQL statement:

- (i) To create a table from a table.  
→ CREATE TABLE new-table AS SELECT \* FROM Existing-table;
- (ii) To eliminate duplicate rows.  
→ SELECT DISTINCT \* FROM ~~Teacher~~ Teacher;
- (iii) To add a new column 'Gender' in the table.  
→ ALTER TABLE Teacher ADD Gender VARCHAR(20);
- (iv) To sort data in a table.  
→ SELECT \* FROM Teacher  
ORDER BY TeacherID;
- (v) To delete rows who works at admin  
→ DELETE FROM Teacher  
WHERE office = "admin";

2015 Spring



3a EMPLOYEE (eid, name, post, age)

POST (Post-title, salary)

PROJECT (Pid, Pname, duration, budget)

WORK-IN (Pid, eid, join-date)

Write SQL statement for

- (i) List the name of employees whose age is greater than average age of all employees.

→ SELECT name FROM EMPLOYEE  
WHERE age > (SELECT AVG(age) FROM EMPLOYEE);

- (ii) Display all employee numbers of those employee who are not working in any project.

→ SELECT eid FROM EMPLOYEE  
WHERE eid NOT IN (SELECT eid FROM WORK-IN);

- (iii) List name of employee and their salary who are working in the project "dbms"

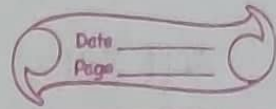
→ SELECT e.name, p.salary FROM EMPLOYEE e  
JOIN WORK-IN w ON e.eid = w.eid  
JOIN POST p ON e.post = p.post-title  
JOIN PROJECT pr ON w.pid = pr.id  
WHERE pr.name = "dbms"

- (iv) Update the database so that "Rishav" now lives in "Butwal"

→ UPDATE EMPLOYEE SET city = "Butwal"  
WHERE name = "Rishav";



2014 Spring:



3a Employee (employee-name, street, city)  
Works (employee-name, company-name, salary)  
Company (company-name, city)  
Manages (employee-name, manager-name)

① Modify the database so that Ram now lives in Kathmandu.

→ ~~ADD~~ UPDATE Employee SET City = "Kathmandu"  
WHERE employee-name = "Ram";

② Give all employees of First Bank Corporation a 10% rise

→ UPDATE Works SET salary =  $1.1 * \text{Salary}$   
WHERE company-name = "First Bank Corporation";

③ Give all managers of First Bank Corporation a 10% rise.

→ UPDATE WORKS SET salary =  $1.1 * \text{Salary}$   
WHERE company-name = "First Bank Corporation"  
AND employee-name IN (SELECT employee-name FROM Manages);

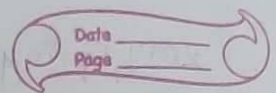
④ Delete all tuples in the works relation for employees of Small Bank Corporation.

→ DELETE FROM Works  
WHERE company-name = "Small Bank Corporation";

⑤ Find all employees who earn more than the average salary of all employees of their company

→ SELECT e.employee-name FROM employee e  
JOIN Works w ON e.employee-name = w.employee-name  
WHERE w.salary > c  
SELECT AVG(salary) FROM Works  
WHERE Company-name = w.company-name;

2014 Fall



2b employee (emp-name, street, city)  
works (emp-name, company, salary)  
company (comp-name, city)  
manages (emp-name, manager-name)

Write SQL statement for:

- ① Find employee name that lives in the city same as the company city.

→ SELECT e.emp-name FROM employee  
JOIN company c ON e.city = c.city;

- ② List all employee details who earns more than \$5000

→ SELECT e.\* FROM employee e  
JOIN works w ON e.emp-name = w.emp-name  
WHERE salary > 25000;

- ③ Update address of an employee "Sriyash" to pokhara.

→ UPDATE employee SET city = "Pokhara"  
WHERE emp-name = "Sriyash";

- ④ Create a view for which employee earn Rs. 20K or more.

→ CREATE VIEW high-earning-employee AS  
SELECT e.\* FROM employee  
JOIN works w ON e.emp-name = w.emp-name  
WHERE w.salary >= 20000;