2020 Fall

2 b (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 on descending manner by brand name and on 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);

2018 FALL

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 Depart-no = (SELECT Depart-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. Depart-no.) >= 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;
```

2016 Fall

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;

(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.id = 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 "Richav" 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.

→ 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 * salary
WHERE company_name = "First Bank Corporation";

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

→ UPDATE WORKS SET salary = 1.1 * 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:

(i) 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;

(ii) List all employee details who earns more than 25000

→ SELECT e.* FROM employee e
JOIN works w ON e.emp_name = w.emp_name
WHERE salary > 25000;

(iii) Update address of an employee "Sriyash" to Pokhara.

→ UPDATE employee SET city = "Pokhara"
WHERE emp_name = "Sriyash";

(iv) 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;