UE14CS256 Database Management Systems Lab Exercises

Lab 1:

Introduction to database and PostgresSQL. Learn basic psql commands. Create a sample database, populate with data, execute given queries, explain the output produced.

Lab 2:

ERD diagram –Draw the ERD for the Company Database discussed in class.(given at end). Construct an E-R diagram for a car insurance company, details given below.

Company: is organized into departments. Each department has a unique name, a unique number, and a particular employee who manages the department. We keep track of the start date when the employee began managing the department. A department may have several locations A department controls a number of projects, each of which has a unique name, a unique number and a single location. We store each employee's name, social security number, address, salary, sex, and birthdate. An employee is assigned to one department, but may work on several projects, which are not necessarily controlled by the same department. We keep track of the number of hours per Lab that an employee works on each project. We also keep track of the direct supervisor or each employee. We want to keep track of the dependents of each employee for insurance purposes. We kep each dependent's first name, sex, birth date and relationship to the employee.

Car Insurance : Construct an E-R diagram for a car insurance company whose customers own one or more cars each. Each car has associated with it zero to any number of recorded accidents. Each insurance policy covers one or more cars, and has one or more premium payments associated with it. Each payment is for a particular period of time, and has an associated due date, and the date when the payment was received.

Lab 3: Railway Database: Create the following database with proper constraints and populate the corresponding tables with at least 10 records. Write and execute the given SQL queries.

- Trainhalts(<u>id, seqno</u>, stcode, timein, timeout)
- Track(<u>stcode1, stcode2</u>, distance)
- Station(<u>stcode</u>, name)
- Train(<u>id</u>, name)

1. Find pairs of stations (station codes) that have a track (direct connection) with distance less than 20Kms between them.
2. Find the IDs of all the trains which have a stop at THANE
3. Find the names of all trains that start at MUMBAI.
4. List all the stations in order of visit by the train 'CST-AMR_LOCAL'.
5. Find the name of the trains which have stop at Thane, before the 6th station in the route of the train.

Lab 4: Create the university database for the given schema of tables.

- Student(id, name, dep_name, tot_credits)
- Takes(id, course_id, sec_id, sem, year, grade)
- Section(course_id, sec_id, sem, year, building, room_no, time_slot_id)
- Time_slot(time_slot_id, day, start_time, end_time)
- Classroom(building, room_no, capacity)
- Teaches(i_id, course_id, sec_id, sem, year)
- Instructor(i_id, name, dep_name, salary)
- Dept(dep_name, building, budget)
- Advisor(id, i_id)
- Course(course_id, title, dep_name, credits)
- Prereq(course_id, prereq_id)

Lab 5: Write the following simple SQL Queries on the University Schema

1. Find the names of all the students whose total credits are greater than 100
2. Find the course id and grades of all courses taken by any student named 'Tanaka'
3. Find the ID and name of instructors who have taught a course in the Comp. Sci. department, even if they are themselves not from the Comp. Sci. department. To test this query, make sure you add appropriate data, and include the corresponding insert statements along with your query.
4. Find the courses which are offered in both 'Fall' and 'Spring' semester (not necessarily in the same year).

Additional queries

a) Find the names of all the instructors from Comp. Sci. department
b) Find the course id and titles of all courses taught by an instructor named 'Srinivasan'
c) Find names of instructors who have taught at least one course in Spring 2009

Lab 6: Write the following SQL Queries on the University Schema (use nested queries)

1. Find the id and title of all courses which do not require any prerequisites.
2. Find the names of students who have not taken any biology dept courses
3. Write SQL update queries to perform the following

4. Give a 10% hike to all instructors
5. Increase the tot_creds of all students who have taken the course titled "Genetics" by the number of credits associated with that course.
6. For all instructors who are advisors of at least 2 students, increase their salary by 50000.
7. Set the credits to 2 for all courses which have less than 5 students taking them (across all sections for the course, across all years/semesters).

<u>Lab7:</u> SQL DDL commands and Updates

1. Each offering of a course (i.e. a section) can have many Teaching assistants; each teaching assistant is a student. Extend the existing schema (Add/Alter tables) to accommodate this requirement.
2. According to the existing schema, one student can have only one advisor.
3. Alter the schema to allow a student to have multiple advisors and make sure that you are able to insert multiple advisors for a student.
4. Write SQL queries on the modified schema. You will need to insert data to ensure the query results are not empty.
5. Find all students who have more than 3 advisors
6. Find all students who are co-advised by Prof. Srinivas and Prof. Ashok.
7. Find students advised by instructors from different departments. etc.

Write SQL queries for the following:

a) Delete all information in the database which is more than 10 years old. Add data as necessary to verify your query.
b) Delete the course CS 101. Any course which has CS 101 as a prereq should remove CS 101 from its prereq set. Create a cascade constraint to enforce the above rule, and verify that it is working.

<u>Lab 8:</u>

Introduction to MongoDb. Connecting to the database, creating database and collection and populating it with documents.

Documents with varying fields such as books, film, person, hospital etc. can be added to the collection. Use find() and findOne() for querying the database.

<u>Lab 9:</u>

Write complex queries to retrieve the documents from the collection using various $opertors .

Lab 10 – 12 Mini Project

Lab 10: problem definition and draft ER diagram to explain the complexity of the project.

Lab 11: ER diagram to be drawn using any of the diagram tools as Edraw, lucidcharts or dia. Schema mapping steps applied to arrive to the final relational schema for the project. Sample queries listed.

Lab 12: complete database implementation, with all the constraints listed in the project definition to be shown. All the functionalities to be shown as query executions on the back end database.

Project Deliverables:

CD containing the following the details need to be submitted during the final presentation of the project.

1> Problem definition with the details of the constraints.
2> ER diagram (using any tool).
3> Brief description of the back end DBMS chosen for implementation..
4> Schema mapping done using the steps discussed in the text book.
5> Script files for all the tables created for the database and also for the insert statements to populate the database.
6> List of all possible query statements to execute on the database along with the SQL statement written for the same.
7> This same text file could be used in the final demo of the project.

Lab 2 ERD Diagram for Company Database (Navathe page : 204)

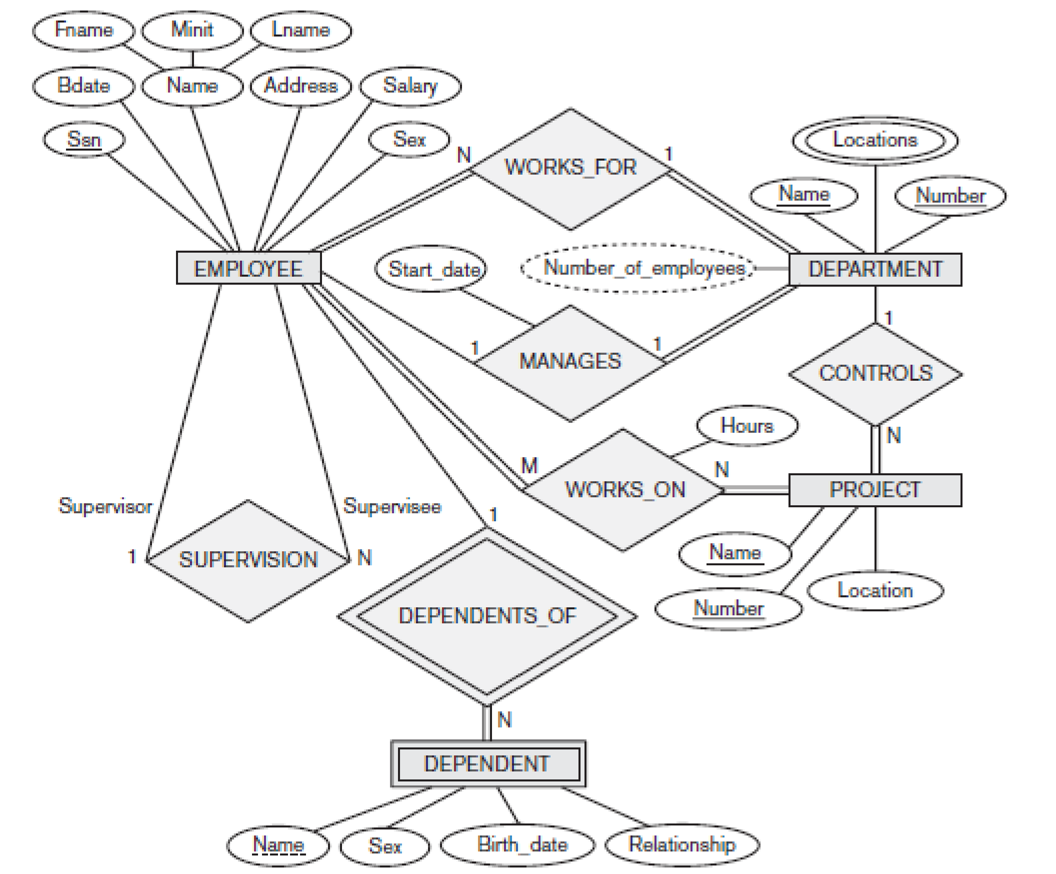

Figure 7.2

PostgreSql Commands:

sudo –u postgres psql postgres

should give postgres# prompt. You can use following commands:

\c *<dbname>*
\l to list databases
\d to list tables in the database
\d *<table>* to get the schema
\h to get help
\q to quit

Additional Lab Exercises:

(A) Consider the Movie Database given below. The Primary keys and Foreign keys are in bold,

Movies(**mid**:string,movie_name:string,year:int,reviews:string,rating:int)
Director(**d_id** int,**mid**:string,movie_name:string,schedule:string);
Producer(**pid**:int,**d_id**:int,budget:string);
Playin(actor_name:string,casttype:string)
Music_director(**music_id**:int,**d_id**:int,music_type:string,movie_name:string);

1. Create the above tables by properly specifying the primary keys and foreign keys.
2. Enter at least five tuples for each relation.
3. Update the schedule for a particular movie.
4. sort the name of the movie in descending order
5. Find the names of the movies in which a particular producer and director worked together
6. Write the query to ADD a new column salary in an existing movies table.
7. Display all the fields consisted in a movie name begining with “A”
8. Find a set of movies (title, reviews, rating) M, so that each movie in M is not dominated by any other movie in the database. Return the output on ascending order of title.
9. Delete the row of particular music type.
10. Drop the music_director table.

(B) Given below is the **Hospital Schema**,the Primary keys and Foreign keys are in bold,
 .
Doctor(**d_id**,d_name,d_phone)
Patient(**p_id**,p_name,diagnosis,age)
Medicine(**med_id**,med_name)
Prescription(**p_id,d_id,med_id**)
Bed(**B_id,**ward_no)
Bed_Patient(**p_id,b_id**)

1. Create the above tables by properly specifying the primary keys and foreign keys.
2. Enter at least five tuples for each relation.
3 List all medicine name which starts from alphabet ‘A’.
4 Add a column floor to Bed table
5.Find number of patients currently admitted
6.List all doctors along with the count of pateints treated by them
7. Display pateint name,diagnosis,medicine name and doctor name of all the patients who are admitted after specified date
8.Demonstrate update medicine name to a newer one
9. Delete a tuple from the Doctor table
10. Drop any one of the schema