LAB 5 Worksheet

Q1. See the following useful features of PostgreSQL from the manual. Download it if not already done. The following refers to sections in the manual.

3.4 : transactions

7.6 LIMIT and OFFSET clauses

8.1.4 : serial type

8.2 : monetary type

8.5 and 9.9 - date/time types and functions

8.7 : enumerated type

8.15 : arrays and 8.16 : composite types

8.8 : geometric types

5.6 : on privileges

Read create trigger statement details in reference part of the manual

Ch. 34 : information schema

Q2. Find top 5 students (get their names and department) by tot_cred (ie, those top 5 who have completed highest total credits).

Q3. You will notice in the university schema that we are repeating course_id, sec_id, semester, year in 3 tables : teacher, takes and section. Can we simplify this by using serial data type in section, which will act as a key or all sections. It will be auto-incremented (see sec 8.1.4 in postgresqlmanual) ? Table takes and teaches then can use thatserial number as foreign keys and simplify the table definitions and queries. Write new create table statements for these 3 table with serial type column in the section table. Insert some sample data. See how postgresql maintains the serial data type.

Q4. Create a table which will store aggregate data about department. It should have dept name, budget, total-salaries, no-of-faculty.

Then write a trigger for the following:

- insert on faculty should chanfe total salary and faculty count
- update on instructor salary should adjust the total salary of the department.

Q5. Use begin/end transaction SQL statements along with commit/rollback. Consider a case where we want all or none of the following to happen:

i) add a course

ii) create a section for it with some year/semester etc.

iii) assign a teacher to it

Create a file (.sql type) containing the above statements within begin and end transaction.

Try all of the above ending with commit (execute the above .sql file). Then check into database if inserts are present.

Try and put a rollback after step (ii) above and check if the database has rolled both inserts. Try the same after step (iii).