LAB EXERCISES FOR DBS LAB ICT-3163 (OCT-DEC 2021)

SL. NO.	LAB EXERCISES
1	(Lab 1 from manual) 1. Design a scientific calculator using C#. Have at least 4 different kinds of scientific functions. 2. Develop a simple form to enter necessary details for online registration of students. Also, display the message (confirm or not) along with details entered on submit. Perform necessary validations. Use Text Box, Radio button, Combo box, Check box, Calendar, Label, Button, and Message Box.
2	(Lab 2 from manual) 1. Develop a notepad application using Rich Text Box, Menu Strip, File Dialog, Colour Dialog, Font Dialog components. 2. Develop a user interface for a banking application. A customer should be able to login with his/her credentials. Also, customer should be able to change his/her password. The second form should display the customer's username, balance, last access, date and last 5 transactions. The third form should facilitate money transfer by adding beneficiary. The amount transferred and the current balance in the account should be displayed as a message.
3	(Lab 3 and 4 from manual) **1.** Consider the Insurance database given below: PERSON (driver_id#: varchar(30), name: varchar(50), Address: varchar(100)) CAR (regno: varchar(20), model: varchar(30), Year:int) ACCIDENT (report_number: int, accd_date: date, location: varchar(50)) OWNS (driver_id#: varchar(30), regno: varchar(20)) PARTICIPATED (driver_id#: varchar(30), regno: varchar(20), report_number: int, damage_amount: int) i. Create the above tables by properly specifying the primary keys and the foreign keys. ii. Enter at least five tuples for each relation. (**Hint**: Date format is 'dd-mmm-yyyy') iii. Update the damage amount to 25000 for the car with a specific reg.

	no in a PARTICIPATED table with report number 12. iv. Delete the accident and related information that took place in a specific year. (**Hint**: Command to extract year component from the date attribute is, extract (year from accd_date)) v. Alter table to add and delete an attribute. vi. Alter table to add Check constraint. 1. Find the **total** number of people who owned cars that were involved in accidents in 2008. 2. Find the number of accidents in which cars belonging to a specific model were involved. 3. Produce a listing with **header as** OWNER_NAME, **No. of** Accidents, and **Total Damage** Amount in a descending **order** on total damage. 4. List the Owners who made **more than 2** accidents in a year. 5. List the owners who are **not involved** in any accident. Optional: 3*, 4*: Additional exercises from 3rd and 4th lab from the manual
4	(Lab 5 from manual) 1. Consider the following database of student enrolment in courses and books adopted for each course STUDENT (regno: varchar(20), name: varchar(50), major: varchar(20), bdate:date) COURSE (course#:int, cname: varchar(30), dept: varchar(30)) ENROLL (regno: varchar(20), course#:int) BOOK_ADOPTION (course#:int, sem:int, book_isbn:int) TEXT (book_isbn:int, booktitle: varchar(50), publisher: varchar(50), author: varchar(50)) i. Create the above tables by properly specifying the primary keys and the foreign keys. ii. Enter at least five tuples for each relation. iii. Execute following queries on the database using nested subquery concept: a. List the courses which uses more than 1 text book. b. List the departments whose all course text books are published by a particular publisher. c. Find the students who have enrolled for course of more than one department

	d. Produce a list of students who are not enrolled. e. List the books which are adopted by the course as well as enrolled by the student. f. List the courses which has adapted at least two books from a specific publisher. g. Identify the students who are enrolled for maximum number of books. h. List the publishers along with the number of books published by them. i. List the students who are enrolled to all their courses which adopts books. (note: some of the courses need not adopt books) j. Implement previous question i. using views. Optional: 5*: Additional exercises from lab 5
5	(Lab 6 from manual) Consider the database created in lab3 1. Generate a trigger displaying driver information, on participating in an accident 2. Create a function to return total number of accidents happened in a particular year. 3. Create a procedure to display total damage caused due to an accident for a particular driver on a specific year. 4. Create a procedure to display accident information which took place in a particular location. 5. Write a procedure which inserts customer information into a table T1, whose total order amount exceeds 10,000 INR. 6. Create a procedure to store accident details into a different table based on accident location information. Make use of Cursor concept.
6	(Lab 7 from manual) Connect the VC# front end of INSURANCE database with the back end. Execute the queries given under Lab Exercises (Lab 3) through front end.