**DL703/824 Y2 – Database Management Systems**

**Continuous Assessment 2**

Date: **13/3/17**

CA weight: 25%

Total marks: 100

Student name: EMILY MEAGHER

Student number: N00150623

* For this in class test you may use the Query Express to form the queries required to solve each of the problems below.
* Please copy and paste your completed SQL query solution from Query Express to the answer column below. If in any case you cannot produce the correct solution, please submit your best attempt at a solution.
* When you are finished, save this document to the appropriate folder in:

T:\Mohammed Cherbatji\CC2 – DBMS\CA2 Submissions\

* You may refer to the SQL notes documents that you have been provided with.
* For this test with you will be working with your own Database. In SQL query analyser ensure you choose the Database with the same name as your student number, e.g. ca2\_n12345678

This Database contains some tables that have been created for you. You will also be required to create your own tables in this Database.

* You may **not** access **any** other resource on the IADT network or on the Internet during the test. If you do so a mark of 0 may be awarded.

|  |  |  |  |
| --- | --- | --- | --- |
| Creating and modifying tables (3 marks each) | | | |
| Question | | | Your Answer |
| 1 | Create a table called restaurant which has the following columns (choose appropriate data types):  restaurantNum (an identity column)  restaurantName  restaurantPhoneNum (which can be null)  managerName  numTable | | create table restaurant  (  restaurantNum int identity,  restaurantName varchar(20),  restaurantPhoneNum int null,  managerName varchar(20),  numTable int  ) |
| 2 | Insert the following rows into the restaurant table: | | insert restaurant  (restaurantName, restaurantPhoneNum, managerName, numTable)  values  ('Los Pollos Hermanos', 353265412, 'Gus Fring', 50),  ('Snack Attack', 454656323, 'John Gruber', 45),  ('Rusty Fork', 787945131, 'Kate Rogers', 25),  ('Babs BBQ', 454554545, 'Barbara Roe', 80) |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | restaurantNum | restaurantName | restaurantPhoneNum | managerName | numTable | | 1 | Los Pollos Hermanos | 353265412 | Gus Fring | 50 | | 2 | Snack Attack | 454656323 | John Gruber | 45 | | 3 | Rusty Fork | 787945131 | Kate Rogers | 25 | | 4 | Babs BBQ | 454554545 | Barbara Roe | 80 | | | | |
| 3 | | Create a view named largeRests on the restaurant table that displays the restaurantNames of restaurants with 50 or more tables. | CREATE VIEW largeRests AS  SELECT restaurantName  FROM restaurant  WHERE numTable >= 50 |
| 4 | | What command would you run to verify that the view you created above is working correctly? | Select \* From largeRests |
| 5 | | Create a view named rRests on the restaurant table that displays all rows where the restaurantName contains the letter ‘r’ | ALTER VIEW rRests AS  SELECT restaurantName  FROM restaurant  WHERE restaurantName LIKE '%r%' |
| 6 | | Change the restaurant table so that all restaurants with less than 30 tables now have 10 tables. | update restaurant  set numTable = 10  where numTable <= 30 |
| After the change:   |  |  |  |  |  | | --- | --- | --- | --- | --- | | restaurantNum | restaurantName | restaurantPhoneNum | managerName | numTable | | 1 | Los Pollos Hermanos | 353265412 | Gus Fring | 50 | | 2 | Snack Attack | 454656323 | John Gruber | 45 | | 3 | Rusty Fork | 787945131 | Kate Rogers | **10** | | 4 | Babs BBQ | 454554545 | Barbara Roe | 80 | | | | |
| 7 | | Change the restaurants table so that all restaurant with phone numbers starting with ‘4’ now have 100 tables. | update restaurant  set numTable = 100  where restaurantPhoneNum LIKE '4%' |
| After the change:   |  |  |  |  |  | | --- | --- | --- | --- | --- | | restaurantNum | restaurantName | restaurantPhoneNum | managerName | numTable | | 1 | Los Pollos Hermanos | 353265412 | Gus Fring | 50 | | 2 | Snack Attack | 454656323 | John Gruber | **100** | | 3 | Rusty Fork | 787945131 | Kate Rogers | 10 | | 4 | Babs BBQ | 454554545 | Barbara Roe | **100** | | | | |
| 8 | | Delete the row containing restaurant number 4. | delete restaurant  where restaurantNum = 4 |
| 9 | | Remove the managerName column of the restaurant table. | alter table restaurant  drop column managerName |
| 10 | | Add a column named numOfEmployees to the restaurant table. | alter table restaurant  add numOfEmployees int |
| 11 | | Delete the entire restaurant table. | drop table restaurant |

The following questions use the **state** and **company** tables in your database. See the final page for the contents of these tables.

|  |  |  |
| --- | --- | --- |
| Stored Procedures (5 marks each) | | |
| Question | | Your Answer |
| 1 | Create a stored procedure named **avgPop** which calculates and displays the average population of the states in the state table. Hint: the output when this is executed should be: ‘Average population is 11154488’ | alter proc avgPop  as  declare @avgP int  select @avgP = AVG(pop) From state  select 'Average Population is ' + convert(varchar, @avgP) |
| 2 | What SQL statement did you use to execute the above stored procedure? | Exec avgPop |
| 3 | What SQL statement would you use to delete the stored procedure created in 1 above? | drop proc avgPop |
| 4 | Create a stored procedure named **largeComps** which displays the number of companies with more than 300000 employees. Hint: the output when this is executed should be 'Number of large companies is 3**’**. | create proc largeComps  as  declare @numC int  select @numC = COUNT(compNum)  FROM company  where numEmployees > 300000  select 'Number of large companies is ' + convert(varchar, @numC) |
| 5 | Create a stored procedure named **aboveAvg** which displays the number of states that have a population greater than the average population. Hint: the output should be: ‘Above average states: 3’ | create proc aboveAvg  as  declare @avgP int  declare @upAvg int  select @avgP = AVG(pop) FROM state  select @upAvg = COUNT(stateNum)  FROM state WHERE pop > @avgP  select ' Above average states: ' + convert(varchar, @upAvg) |
| 6 | Create a stored procedure named **firstLetter** thatdisplays all rows containing states that start with a particular letter. The letter is passed to the procedure as a parameter. | alter proc firstLetter  @fLetter varchar(1)  as  select stateName From state  where stateName LIKE '@fLetter%' |
| 7 | What SQL statement would you use to execute the above procedure and pass it ‘A’ as a parameter? | exec firstLetter A |

|  |  |  |
| --- | --- | --- |
| Triggers (8 marks each) | | |
| Question | | Your Answer |
| 1 | Create a trigger that ensures that any new company’s details that are entered include a valid hqLocation (i.e. one that matches a stateNum in the state table). If not the company’s information should not be allowed to be added. | create trigger trig1  on company  for insert  as  if not exists(select \* from state, inserted where state.stateNum = inserted.hqLocation)  begin  select 'Invalid company information'  rollback transaction  end  return |
| 2 | Create a trigger that ensures that the relevant noOfHqs value for the correct state in the state table is kept up to date when a new company is entered in the company table. | create trigger trig2  on state  for update  as  if @@rowcount = 0  return  if update(numOfHqs)  begin  update state  set state.numOfHqs = inserted.numOfHqs  from state, inserted, deleted  where state.numOfHqs = deleted.numOfHqs  end  return |
| 3 | Create a trigger that ensures that if a stateNum is changed in the state table, that the hqLocation in the company table is changed to the new number also. | create trigger trig3  on state  for update  as  if @@rowcount = 0  return  if update(stateNum)  begin  update state  set state.stateNum = inserted.stateNum  from state, inserted, deleted  where state.stateNum = deleted.stateNum  end  return |
| 4 | Create a trigger that ensures that if a state is deleted from the state table, all companies that have HQs in that state are deleted from the company table. | create trigger trig4  on state  for delete  as  if @@rowcount = 0  return  delete state  from state, deleted  where state.stateName = deleted.stateName  return |

state

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| stateNum | stateName | stateCapital | stateAbbrev | pop | numOfHqs |
| 1 | Alabama | Montgomery | AL | 4779736 | 3 |
| 2 | Arizona | Phoenix | AZ | 6392017 | 0 |
| 3 | Florida | Tallahassee | FL | 18801310 | 1 |
| 4 | Kansas | Topeka | KS | 4173405 | 1 |
| 5 | Massachusetts | Boston | MA | 6349097 | 0 |
| 6 | New York | Albany | NY | 18976457 | 0 |
| 7 | Texas | Austin | TX | 27000000 | 0 |

company

|  |  |  |  |
| --- | --- | --- | --- |
| compNum | compName | numEmployees | hqLocation |
| 1 | Super Corp | 400000 | 1 |
| 2 | Mega Inc | 500000 | 1 |
| 3 | Gigantosoft | 600000 | 1 |
| 4 | Big Chips | 50000 | 3 |
| 5 | Humungo Ltd | 45000 | 4 |