CS 4611 – Spring 2017 – Laboratory 2
Assigned: 09/12/2017
Due: 09/12/2017 at 11:59 PM. Submit your file(s) to canvas.
Maximum Grade: 100 pts.

Notes:

- **If you want, you can work in pairs for this assignment.**

Objectives: After this lab you will learn the following:

- **Learn a slight variation on the syntax to insert rows into a table by listing the attributes you want to change, i.e. INSERT INTO my_table(name, age) VALUES('Methuselah', 969)**
- **Delete and update rows in tables**
- **Issue nested sub-queries, i.e. queries that have other queries inside their FROM or WHERE clauses**
- **Deal with violations of referential integrity constraints (a.k.a. foreign key constraint) using "on delete cascade" and "on delete set null"**
- **Do string pattern matching with "LIKE"**
- **Practice set membership testing ('IN', 'EXISTS', 'ALL')**
- **Learn how to sort query results and do top-K queries.**

Activity 1: Use SQL Developer to connect to the Oracle server running in the machine akka.d.umn.edu. For your username use the first four letters of your last name followed by the first letter of your name. Example: "leale" would be my username. If your last name doesn't have more than four letters, then use your full last name followed by the first letter of your name.

Activity 2: Consider the following scenario.

- There are three police departments: Duluth PD, UMD PD, and Cloquet PD (*Remember this when you think about your constraints*).
- There are also several police officers working for each of these departments.
- Every lead detectives of a case is a police officer.
- Each case as exactly one lead detective.
- A person can be a suspect for more than one case and a case can have several suspects.

The following tables capture this scenario:

police_officer

officer_id	officer_name	rank	dept_name	Salary
7A39	Bilbo	Rookie	Duluth PD	40K
8B67	Sauron	Chief	UMD PD	80K
7X93	Legolas	Lieutenant	UMD PD	65K
3E52	Lady Galadriel	Rookie	Duluth PD	45K

9A89	Gimli	Captain	Cloquet PD	60K

case

case_num	date	lead_detective_id	dept_name
12	01/02/1987	8B67	Duluth PD
15	02/04/1979	8B67	UMD PD
26	06/12/2005	3E52	Duluth PD
22	09/7/2017	3E52	UMD PD
17	04/05/2016	9A89	Cloquet PD

s**uspect**

suspect_id	name	age	description
A12	Saruman	1287	'Tall with white hair and beard and carries a staff'
B87	Azog	389	'Pale white orc'
C34	Sauron	8942	'Big fiery eye'
D67	Frodo	34	'3 feet tall'

is_suspect_of

case_num	suspect_id
12	D67
26	B87
12	A12
17	C34
15	B87

Now, write a file names *hw2_yourname.sql* containing code to perform the following:

a) Create the tables above in Oracle. The table definitions must define the necessary primary keys, foreign keys, not-null constraints and check constraints. This time, however, when you type a foreign key constraint you need to specify how you are going to deal with violations of referential integrity constraints (i.e., use "on cascade delete" or "on cascade set null" as you see fit).

b) (Deletion query) Delete all those cases that have dates older than 20 years. For this query, compute the current date and then subtract 20 years.

c) (Update query) Promote all those officers that work at Duluth PD with a rank of "Rookie" to "Lieutenant." Note: *You must use a single SQL query, i.e., you can't do separate update queries*.

d) Retrieve the top-2 highest earning police officers.

e) An officer at UMD PD wants to know the count of the number of cases where each officer has been the lead detective (detective id's are officer id's) and the officer's name.

f) Retrieve the names of the suspects in all cases where officer 3E52 has been the lead detective.

g) Find the lead detectives in crime cases with suspects whose description is like "staff."

h) Find the rank of the officers that have been lead detectives in cases associated with UMD PD and in cases associated with Duluth PD.
i) Retrieve the names of all the officers in all departments that earn more than the average salary at ‘UMD PD’
j) Find the average salary among those officers earning more than 41K in the police departments that have at least 2 people.
k) Find the names of those officers that earn more than at least one other officer. Write this query in SQL in two different ways.