```
-- COSC265 S2 2017 Lab Test Solutions

-- Question 1 - 25 marks total
--
-- 1a. (10 marks) Write a single SQL statement to create a JobSkill table,
--   which holds a job name (e.g. SWDeveloper, Lifeguard, or
--   SystemsAnalyst) and a single character skill code (e.g. S, F, C, or D),
--   plus a third attribute to hold a Rank value (a number such as 1, 2 or 3)
--   for the ranked importance of the skill for that job.  Both the job name
--   and the skill code are to be used as the primary key.

-- Answer:
CREATE TABLE JobSkill
(
   J_Name  VARCHAR(20),                   // make sure is large enough
   S_Code  CHAR  REFERENCES Skill(S_Code),
   Rank    SMALLINT,
   PRIMARY KEY(J_Name, S_Code)
);

-- 1b. (5 marks) Correct the following SQL statements to properly insert the
--   included data into the JobSkill table you just created.
-- NOTE: These statements are also in the file Misc.sql; you can copy and past
--   them into your solutions file and then correct them to save you some
--   typing time.
-- NOTE: be sure to execute these statements in SQL once you have corrected
--   them.
INSERT INTO JobSkill VALUES (SWDeveloper, 'C', 2);
INSERT INTO JobSkill VALUES (SWDeveloper, 'D', 3);
INSERT INTO JobSkill VALUES (SWDeveloper, 'T', 1);
INSERT INTO JobSkill VALUES (Lifeguard, 'F', 2);
INSERT INTO JobSkill VALUES (Lifeguard, 'S', 1);

-- Answer
INSERT INTO JobSkill VALUES ('SWDeveloper', 'C', 2);
INSERT INTO JobSkill VALUES ('SWDeveloper', 'D', 3);
INSERT INTO JobSkill VALUES ('SWDeveloper', 'T', 1);
INSERT INTO JobSkill VALUES ('Lifeguard', 'F', 2);
INSERT INTO JobSkill VALUES ('Lifeguard', 'S', 1);


-- 1c. (5 marks) Write and execute a single SQL statement to show how many rows
--   are now in the JobSkill table

-- Answer: (5 rows)
SELECT COUNT(*)
FROM JobSkill;


-- 1d. (5 marks) Write a single SQL statement to change Gollum's name
--         to Smeagol in the creature table.
UPDATE Creature
SET C_Name = 'Smeagol'
WHERE C_Name = 'Gollum';


-- Question 2 - 50 marks total
```

```
--
-- 2a. (10 marks) Write a single SQL statement to find the name of each creature
--  who has achieved a skill that where the skill weight was less than or equal
--  to 0.5.  Display the names in alphabetical order, without duplicates.
-- NOTE: You must use JOIN clauses only (no nested sub-queries.)

-- Answer:
SELECT DISTINCT C_Name
FROM Creature C
JOIN Achievement A ON (C.C_ID = A.C_ID)
JOIN Skill S ON (A.S_Code = S.S_Code)
WHERE S.S_Weight <= 0.5
ORDER BY C_Name;

-- or

SELECT DISTINCT C_Name
FROM Creature C
NATURAL JOIN Achievement A
NATURAL JOIN Skill S
WHERE S.S_Weight <= 0.5
ORDER BY C_Name;

-- Results, 3 rows, using either approach:
-- Bannon
-- Gollum   (if old name)
-- Neff

-- or…
-- Bannon
-- Neff
-- Smeagol (if updated name)


-- 2b. (10 marks) Write a single SQL statement to find the name of each creature
--  that has achieved a skill where the skill weight was less than or equal
--  to 0.5.  Display the names in alphabetical order, without duplicates.
-- NOTE: You must use nested sub-queries only (no joins).

-- Answer:
SELECT DISTINCT C_Name
FROM Creature
WHERE C_ID IN
  (SELECT C_ID
   FROM Achievement
   WHERE S_Code IN
      (SELECT S_Code
       FROM Skill
       WHERE S_Weight <= 0.5))
ORDER BY C_Name;

-- results, 3 rows, same as above
-- Bannon   or  Bannon
-- Gollum       Neff
-- Neff         Smeagol
-- 2c. (10 marks) Write a single SQL statement to generate a list of each type of
--   creature in the database, a count of the number of skill achievements that
--   have been achieved by all creatures of that type, and the average
achievement
--   score (NOT skill weight) for those achievements.  Order your results by the
--   achievement count in descending order.
-- NOTE: you do NOT have to format the numeric results

-- Answer:
SELECT C.C_Type, COUNT(*) AS Count, AVG(A.Score) as AVG_Score
FROM Creature C
JOIN Achievement A ON C.C_id = A.C_id
GROUP BY C_Type
ORDER BY COUNT(*) DESC;

-- Results, 3 rows
-- Person, 11, 2.0909…
-- Hobbit,  2, 1.5
-- Dragon,  1, 1


-- 2d. (10 marks) Write a single SQL statement to find each pair of two
--   different skill codes where both skills were achieved at level 2.
--   Remove all duplicate pairs, including exact duplicates (e.g. A B and A B
--   are consider exact duplicate pairs) and all interchanged order pairs
--   (e.g. A B and B A are an example of an interchanged duplicate pair.)
-- NOTE: question wasn't clear on whether skill pairs could be achieved by any
--  creatures or had to be achieved by same creature, so both allowed

-- Answer, if assume pair of skills by one creature
SELECT DISTINCT A1.S_Code, A2.S_Code
FROM Achievement A1
JOIN Achievement A2 ON A1.C_id = A2.C_id
WHERE A1.S_Code < A2.S_Code
AND A1.Score = 2
AND A2.Score = 2
ORDER BY A1.S_Code, A2.S_Code;

-- Answer: 1 row
-- F, S

-- Answer, if allow pair of skills achieved by same or different creature
SELECT DISTINCT A1.S_Code, A2.S_Code
FROM Achievement A1
CROSS JOIN Achievement A2
WHERE A1.S_Code < A2.S_Code
AND A1.Score = 2
AND A2.Score = 2
ORDER BY A1.S_Code, A2.S_Code;

-- Answer: 6 rows
-- F, R
-- F, S
-- F, T
-- R, S
-- R, T
-- S, T
-- 2e. (10 marks) SQL to find each creature and the count of their achieved
--  skills.  Make sure that all creatures are included in the result regardless
--  of the number of achievements.

-- Answer:
SELECT C.C_id, COUNT(A.S_code) AS SkillCt
FROM Creature C
LEFT OUTER JOIN Achievement A ON (C.C_id = A.C_id)
GROUP BY C.C_id
ORDER BY C.C_id;

-- Result, 8 rows; left outer join needed to get creature 6 with no skills
-- 1, 3
-- 2, 1
-- 3, 2
-- 4, 2
-- 5, 3
-- 6, 0
-- 7, 2
-- 8, 1


-- Question 3 (25 marks total)
--
-- 3a. (8 marks) Define a view ACH_VIEW that includes the following
--  achievement-related information: creature id, creature name, creature type,
--  achievement skill code, achievement score, and skill name for that
--  achievement

-- Answer:
CREATE OR REPLACE VIEW Ach_View AS
SELECT C.C_id, C.C_Name, C.C_Type, A.S_Code, A.Score, S.S_Desc
FROM Creature C
JOIN Achievement A ON (C.C_id = A.C_id)
JOIN Skill S ON (A.S_Code = S.S_code);


-- 3b. (4 marks) Write a single SQL statement to display all information from
--   this view, but only for the rows for creatures 1 through 4 inclusive

-- Answer:
SELECT *
FROM Ach_View
WHERE C_id BETWEEN 1 and 4;

-- Result, 8 rows
-- 1, Bannon, Person, S, 1, Swim
-- 1, Bannon, Person, F, 3, Float
-- 1, Bannon, Person, C, 3, Code
-- 2, Myers,  Person, S, 3, Swim
-- 3, Neff,   Person, S, 2, Swim
-- 3, Neff,   Person, D, 1, Design
-- 4, Neff,   Person, S, 2, Swim
-- 4, Neff,   Person, F, 2, Float
-- 3c. (8 marks) We could try inserting a new creature and achievement in a
--      single SQL insert statement by using the Ach_View view instead
--      of working with the Creature and Achievements tables directly.
-- Try to execute the following statement (also in misc.scl for copy/paste)
-- INSERT INTO Ach_View (C_id, C_Name, C_Type, S_Code, Score)
--  VALUES (9, 'Fanghorn', 'Ent', 'W', 3);
-- However, this statement fails, as the joined view is not updatable.
-- So, your task here is to write an SQL trigger to successfully accomplish
--   the above task of updating two tables when someone does try the
--   view insert above.
--
-- Re-execute the Insert statement above to test your
--   trigger and make sure that the trigger fires correctly.

-- Answer:
CREATE OR REPLACE TRIGGER Insert_Creature
INSTEAD OF INSERT ON Ach_View
BEGIN
   INSERT INTO Creature (C_id, C_Name, C_Type)
      VALUES (:new.C_id, :new.C_Name, :new.C_Type);
   INSERT INTO Achievement (C_id, S_Code, Score)
      VALUES (:new.C_id, :new.S_Code, :new.Score);
END;
/


-- Question 4 - 5 marks total
--
-- This is a relatively low mark but more difficult question; work on this
--   question only after you have completed all of the other questions.
--
-- 4. (5 marks) Write one or more SQL statements to find a list of the
--   creatures, by id, who have achieved all software developer skills
--   (as would be listed in the JobSkill table you created and populated above).

-- Answers:

-- 1) using COUNTS
SELECT C.c_id
FROM Creature C
JOIN Achievement A ON C.c_id = A.c_id
JOIN Skill S ON A.s_code = S.s_code
WHERE S.s_code IN
   (SELECT s_code
    FROM JobSkill
    WHERE J_Name = 'SWDeveloper')
GROUP BY C.c_id
HAVING COUNT(*) =
   (SELECT COUNT(*)
     FROM JobSkill
     WHERE  J_Name = 'SWDeveloper');

-- result: 1 row
-- C_ID
-- 5
-- 2) using double NOT EXISTS
SELECT DISTINCT C_id              -- find each creature…
FROM Creature C
WHERE NOT EXISTS                  -- … where there doesn’t exist a SWDev job skill
  (SELECT *
   FROM JobSkill J
   WHERE J_Name = ‘SWDeveloper’
   AND NOT EXISTS                 -- … that has not been achieved by that creature
     (SELECT *
      FROM Achievement A
      WHERE A.C_id = C.C_id
      AND A.S_Code = J.S_Code) );

-- result: 1 row
-- C_ID
-- 5


-- 3) using double MINUS, in a single SQL query
SELECT DISTINCT C_id
FROM Creature
  MINUS
SELECT DISTINCT C_id
FROM
  (SELECT C_id, S_Code
   FROM Creature
   CROSS JOIN (SELECT S_Code
               FROM JobSkill
               WHERE J_Name = 'SWDeveloper')
     MINUS
   SELECT C_id, S_Code
   FROM Achievement);

-- Result, 1 row
-- C_ID
-- 5


-- 4) using double MINUS, broken into steps a) through e)
-- a) make universe of all possible creature/SWDeveloper skill combinations

CREATE TABLE Ach1 AS
  SELECT C_id, S_Code
  FROM Creature
  CROSS JOIN (SELECT S_Code FROM JobSkill WHERE J_Name = 'SWDeveloper');

SELECT *
FROM Ach1;

-- b) find each creature id/SWDeveloper skill pair that hasn't been achieved
CREATE TABLE Ach2 AS
   SELECT C_id, S_Code
   FROM Ach1
     MINUS
   SELECT C_id, S_Code
   FROM Achievement;

SELECT *
```

```
FROM Ach2;

-- c) find each creature that hasn't achieved at least one SWDeveloper skill
CREATE TABlE Ach3 AS
   SELECT DISTINCT C_id
   FROM Ach2;

SELECT *
FROM Ach3
ORDER BY C_id;

-- d) find each creature that has achieved all SWDeveloper skills
CREATE TABLE Ach4 AS
   SELECT C_id
   FROM Creature
     MINUS
   SELECT C_id
   FROM Ach3;

SELECT *
FROM Ach4;

-- Result, 1 row
-- 5
```