-- /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--  \* NAME: George Calvert

--  \* CLASS: CPSC 321

--  \* DATE: 10/2/22

--  \* HOMEWORK: Problem Set 4

--  \* DESCRIPTION: develop five “interesting” and “relevant” analytical queries for your final project

--  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

-- -- Question 1

SELECT set\_id, created\_by, username AS saved\_by

FROM Saves RIGHT OUTER JOIN StudySet USING(set\_id);

-- -- This query is interesting since it returns all sets in the database but leaves a NULL value

-- -- in created by if a set is not saved by another user

-- Question 2

(SELECT set\_id, "SET" AS type FROM StudySet)

UNION

(SELECT set\_id, "SAVED" AS type FROM Saves);

-- This query is interesting since it selects all set IDS and compares them to ones that were saved.

-- Meaning that you can easily see the number of sets that were saved compared to not saved.

-- Question 3

WITH saves AS(

    SELECT set\_id, set\_name, created\_by, username AS saved\_by

    FROM Saves RIGHT OUTER JOIN StudySet USING(set\_id)

)

SELECT set\_id, created\_by,

    CASE

        WHEN saved\_by IS NULL THEN "saved"

        ELSE "not saved"

    END AS saved

FROM saves

GROUP BY set\_id;

-- This query is interesting since it changes the null values to more readable values like saved or not saved

-- if a study set tat is created by a user is svaed by another user

-- Question 4

WITH number\_created AS(

    SELECT username, COUNT(\*) AS number\_sets

    FROM User JOIN StudySet ON(created\_by = username)

    GROUP BY username

    ORDER BY number\_sets DESC

)

SELECT username, number\_sets, DENSE\_RANK() OVER (ORDER BY number\_sets DESC) AS rank

FROM number\_created;

This query is interesting since it ranks the users based on the number of study sets they have created/

For instance rank 1 has the most sets and then descending down

-- Question 5

WITH most\_saved\_sets AS(

    SELECT set\_name, COUNT(\*) AS number\_saves

    FROM StudySet JOIN Saves USING(set\_id)

    GROUP BY set\_id

    HAVING number\_saves >= 1

    ORDER BY number\_saves DESC

)

SELECT set\_name, number\_saves, DENSE\_RANK() OVER (ORDER BY number\_saves DESC) AS rank

FROM most\_saved\_sets;

-- This query is interesting since it ranks the sets based on the number of saves each one has.

-- For example sets with the most saves would have rank 1. This allows me to rank the sets for users

-- for the most used sets.