SQL AND TAIL RECURSION

COMPUTER SCIENCE MENTORS 61A

April 10 to April 14, 2017

1 Tail Recursion

1. What is a tail context/tail call? What is a tail recursive function?

2. Why are tail calls useful for recursive functions?

Answer the following questions with respect to the following function:

```
(define (sum-list lst)
  (if (null? lst)
    0
    (+ (car lst) (sum-list (cdr lst)))
  )
)
```

3. Why is sum-list not a tail call? Optional: draw out the environment diagram of this sum-list with list: (1 2 3). When do you add 2 and 3?

4. Rewrite sum-list in a tail recursive context.

2 SQL

Table name: `mentors`

Name	Food	Color	Editor	Language
Tiffany	Thai	Purple	Notepad++	Java
Diana	Pie	Green	Sublime	Java
Allan	Sushi	Orange	Emacs	Ruby
Alfonso	Tacos	Blue	Vim	Python
Kelly	Ramen	Green	Vim	Python

5. Create a new table **mentors** that contains all the information above.

6. Write a query that lists all the mentors along with their favorite food if their favorite color is green.
   Output:
   ```
   Diana|Pie
   Kelly|Ramen
   ```

7. Write a query that lists the food and the color of every person whose favorite language is NOT Python.
   Output:
   ```
   Sushi|Orange
   Pie|Green
   Thai|Purple
   ```

8. Write a query that lists all the pairs of mentors who like the same language. (How can we make sure to remove duplicates?)
   Output:
   ```
   Kelly|Alfonso
   Tiffany|Diana
   ```

9. Write a query that has the same data, but alphabetizes the rows by name. (Hint: Use order by.)
   Output:
   ```
   Alfonso|Tacos|Blue|Vim|Python
   Allan|Sushi|Orange|Emacs|Ruby
   Diana|Pie|Green|Sublime|Java
   Kelly|Ramen|Green|Vim|Python
   Tiffany|Thai|Purple|Notepad++|Java
   ```

3 Fish Population

The 61A mentors want to start a fish hatchery, and they need your help to analyze the data they've collected for the fish populations! Also, running a hatchery is expensive – they'd like to make some money on the side by selling some seafood (only older fish of course) to make delicious sushi.

The following table contains a subset of the data that has been collected. The SQL column names are listed in brackets. Note: we must be able to extend your queries to larger tables! (i.e, don't hard code your answers)

Table name: `fish`*

Species [species]	Population [pop]	Breeding Rate [rate]	$/piece [price]	# of pieces per fish [pieces]
Salmon	500	3.3	4	30
Eel	100	1.3	4	15
Yellowtail	700	2.0	3	30
Tuna	600	1.1	3	20

*(This was made with fake data, do not actually sell fish at these rates)

10. **Aggregation**

    (a) Profit is good, but more profit is better. Write a query to select the species that yields the most number of pieces for each price. Your output should include the species, price, and pieces.

    (b) Write a query to find the three most populated fish species.

    (c) Write a query to find the total number of fish in the "ocean." Additionally, include the number of species we summed. Your output should have the number of species and the total population.

(d) Business is good, but a bunch of competition has sprung up! Through some cunning corporate espionage, we have determined that one such competitor plans to open shop with the following rates:

Table name: `competitor`

Species [species]	**$/piece [price]**
Salmon	2
Eel	3.4
Yellowtail	3.2
Tuna	2.6

Write a query that returns, for each species, the difference between our hatcherys revenue versus the competitors revenue for one whole fish. For example, the table should contain the following row:

```
Salmon | 60
```

Because we make 30 pieces at $4 a piece for $120, whereas the competitor will make 30 pieces at $2 a piece for $60. Finally, the difference is 60. Remember to do this for every species!

11. **Recursive Select** Suppose these fish breed every day. The population of each fish gets multiplied by its breeding rate every year. Write a recursive select function that creates a table of fish 10 years from now.