## SQL and Final Review

## CS 61A Group Mentoring

April 23 to April 25, 2018

## 1 Creating Tables, Querying Data

Examine the table, mentors, depicted below.

| Name | Food | Color | Editor | Language |
| :---: | :---: | :---: | :---: | :---: |
| Jacob | Thai | Purple | Notepad++ | Java |
| Rachel | Pie | Green | Sublime | Java |
| Jemmy | Sushi | Orange | Emacs | Ruby |
| Daniel | Tacos | Blue | Vim | Python |
| Amy | Ramen | Green | Vim | Python |

1. Create a new table mentors that contains all the information above. (You only have to write out the first two rows.)
2. Write a query that lists all the mentors along with their favorite food if their favorite color is green.
Rachel|Pie
Amy | Ramen
3. Write a query that lists the food and the color of every person whose favorite language is not Python.

Sushilorange<br>PielGreen<br>Thai|Purple

4. Write a query that lists all the pairs of mentors who like the same language. (How can we make sure to remove duplicates?)
Rachel|Jacob
Daniel|Amy

## 2 Aggregation

CS 61A wants to start a fish hatchery, and we need your help to analyze the data we've collected for the fish populations! Running a hatchery is expensive - we'd like to make some money on the side by selling some seafood (only older fish of course) to make delicious sushi.

The table fish contains a subset of the data that has been collected. The SQL column names are listed in brackets.

| Species <br> [species] | Population <br> [pop] | Breeding Rate <br> [rate] | \$/piece <br> [price] | \# of pieces per fish <br> [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 |

Hint: The aggregate functions MAX, MIN, COUNT, and SUM return the maximum, minimum, number, and sum of the values in a column. The GROUP BY clause of a select statement is used to partition rows into groups.

1. Write a query to find the three most populated fish species.
2. 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.
3. 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.
[^0]The table competitor contains the competitor's price for each species.

| Species <br> [species] | \$/piece <br> [price] |
| :---: | :---: |
| Salmon | 2 |
| Eel | 3.4 |
| Yellowtail | 3.2 |
| Tuna | 2.6 |

4. Business is good, but a bunch of competition has sprung up! Through some cunning corporate espionage, we have determined one such competitor's selling prices.

Write a query that returns, for each species, the difference between our hatchery's revenue versus the competitor's revenue for one whole fish. For example, the table should contain the following row Salmonl 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 .

[^1]5. Suppose these fish breed every day. The population of each fish gets multiplied by its breeding rate every year. Write a series of SQL statements which will create a table containing each species of fish, their population, and the year for this year (year 0), next year, and the year after. You have access to all tables defined in previous parts. CREATE TABLE yearly_pop(species, pop, n);
6. Mutate the same table to only include rows for year 2.
7. Tragedy strikes! In year 2, all fish with a population less than 1,000 die, so remove them from the table. In addition, the remaining fish have their population halved, so mutate the table for them similarly.

[^2]
[^0]:    CS61A Summer 2018: Alex Stennet and Chris Allsman, with content by Jennie Chen and
    Ajay Raj, Alex Yang, Annie Tang, Brandon Fong, Catherine Han, Danelle Nachum, Elaine Park, Hyun Jae Moon, Kevin Tsang, Lindsay Yang, Michelle Cheung, Ryan Moughan, Ryan Roggenkemper, Shreya Sahoo, Surya Duggirala, Thomas Zhang

[^1]:    CS61A Summer 2018: Alex Stennet and Chris Allsman, with content by Jennie Chen and
    Ajay Raj, Alex Yang, Annie Tang, Brandon Fong, Catherine Han, Danelle Nachum, Elaine Park, Hyun Jae Moon, Kevin Tsang, Lindsay Yang, Michelle Cheung, Ryan Moughan, Ryan Roggenkemper, Shreya Sahoo, Surya Duggirala, Thomas Zhang

[^2]:    CS61A Summer 2018: Alex Stennet and Chris Allsman, with content by Jennie Chen and
    Ajay Raj, Alex Yang, Annie Tang, Brandon Fong, Catherine Han, Danelle Nachum, Elaine Park, Hyun Jae Moon, Kevin Tsang, Lindsay Yang, Michelle Cheung, Ryan Moughan, Ryan Roggenkemper, Shreya Sahoo, Surya Duggirala, Thomas Zhang

