DS1EDP: Homework 04 – Solutions

1. Working with Text using Functions

Question 1:

```
def word_count(word):
     return len(word.split())
```

Question 2:

```
chapter_lengths = tale_chapters.apply(word_count, "Chapter text")
```

Question 3:

```
def character_count(word):
    word = word.replace(" ","")
    word = word.replace(".","")
    word = word.replace("!","")
    word = word.replace("?","")
    return len(word)
```

Note: You can also call **replace** chained:
`word.replace(" ","").replace(".","").replace("!","").replace("?","")`

Question 4:

```
def chapter_number(chapter_text):
    return text_before(chapter_text, ".")
```

Note: Look at the contents of the chapter. The roman letter that indicates the chapter is always followed by a '**.**'. This seems to be a good pattern to split the text by using **text_before**.

2. Uber

Question 1:

```
bins = np.arange(0, 120, 5)
boston.hist("ride time", bins=bins)
manila.hist("ride time", bins=bins)
city_with_long_ride_times = "Manila"
```

Question 2:

```
boston_under_10 = boston.where("ride time", are.below(10)).num_rows /
boston.num_rows
manila_under_10 = manila.where("ride time", are.below(10)).num_rows /
manila.num_rows
```

Question 3:

```
def average_ride_time_for_time(tbl, hod):
    return np.average(tbl.where("hod", are.equal_to(hod)).column("ride
time"))
```

Question 4:

```
larger_diff = "Manila"
```

3. Faculty salaries

Question 1:

```
departments = profs.group("department", first).column("department")
faculties = profs.group("department", identity).column("name identity")
prof_names = Table().with_columns("department", departments, "faculty",
faculties)
```

Note: Other approaches can lead to the correct result as well.

Question 3:

```
def salary_range(salaries):
     return max(salaries) - min(salaries)

department_ranges = profs.group("department", salary_range)
biggest_range_dept = department_ranges.sort("gross_salary salary_range",
descending=True).column("department").item(0)
biggest_range_dept
```

Note: The name of columns that are calculated during the **group** operation always consists of the name of the original column and the name of the function that was applied during grouping. In this example the name of the resulting column is **gross_salary salary_range**.

4. Causes of death by year

Question 1:

```
# Filter out HOM, HYP and NEP first
cleaned_causes = causes.where("Cause of Death", are.not_equal_to("HOM"))
cleaned_causes = cleaned_causes.where("Cause of Death",
are.not_equal_to("HYP"))
cleaned_causes = cleaned_causes.where("Cause of Death",
are.not_equal_to("NEP"))

# Join with the table abbreviation
cleaned_causes = cleaned_causes.join("Cause of Death", abbreviations)

# Drop and relable the columns
cleaned_causes = cleaned_causes.drop("Cause of Death")
cleaned_causes = cleaned_causes.relabel("Cause of Death (Full
Description)","Cause of Death")
```

Question 2:

```
cleaned_causes_by_year = cleaned_causes.pivot("Cause of Death", "Year",
values="Count", collect=sum)
```

Question 3:

```
most_happened = "Diseases of the Heart"
```

Question 4:

```
suspicion = True
```