



MODULE 06 - Piscine Python for Data Science

SQL and Pandas

Summary: Today we will help you acquire skills with SQL.

Contents

I	Foreword	2
II	Instructions	3
III	Specific instructions of the day	4
IV	Exercise 00 : Select	5
V	Exercise 01 : Subquery	7
VI	Exercise 02 : Join	8
VII	Exercise 03 : Aggregations	10
VIII	Exercise 04 : A/B-testing	12

Chapter I

Foreword

Sequel or Ess QueKyew Ell? How to pronounce SQL correctly? Some interviewers even reject interviewees if they fail of pronounce it the “right” way. Apparently, it is really important to know which is correct.

SQL was originally spelled SEQUEL (Structured English Query Language), but it turned out later that this was the registered trademark of an aircraft company. The authors had to change it. Since then, it has been SQL (Structure Query Language). One of the authors [was asked](#) which pronunciation is correct. He answered:

Since the language was originally named SEQUEL, many people continued to pronounce the name that way after it was shortened to SQL. Both pronunciations are widely used and recognized. As to which is more "official", I guess the authority would be the ISO Standard, which is spelled (and presumably pronounced) S-Q-L.

Thanks for your interest,

Don Chamberlin

So one of the authors said that you can use both ways. S-Q-L is a more official way of doing it (presumably, huh?). What do other “authority figures” think?

It is said that the official way to pronounce “MySQL” is “My Ess Kyew Ell” (not “my sequel”). At the same time, the official Oracle documentation says that the right way to pronounce it is “sequel”.

Before diving into the exercises you should make an important choice are you going to pronounce Sequel or S-Q-L? More informal or more formal? Where does your heart lie?

Chapter II

Instructions

- Use this page as your only reference. Do not listen to any rumors or speculations about how to prepare your solution.
- Here and further on we use Python 3 as the only correct version of Python.
- The solutions for python exercises (d01, d02, d03) must have the following block in the end: `if __name__ == '__main__':`
- Pay attention to the permissions of your files and directories.
- To be assessed your solution must be in your GIT repository.
- Your solutions will be evaluated by your piscine peers.
- You should not leave any additional files in your directory other than those explicitly specified in the subject. It is recommended that you modify your .gitignore to avoid any accidents.
- When you need to get precise output in your programs, it is forbidden to display a precalculated output instead of performing the exercise correctly.
- Have a question? Ask your neighbor on the right. If that fails, try your neighbor on the left.
- Your reference material: peers / Internet / Google.
- You can ask questions in Slack.
- Read the examples carefully. They may require things that are not otherwise specified in the subject.
- And may the Force be with you!


Chapter III

Specific instructions of the day

- Use Jupyter Notebook to work with your code
- For each major subtask in the list of any exercise (black bullets), your ipynb file should have an h2 heading to help your peer navigate easily within your code
- No imports allowed, except those explicitly mentioned in the section “Authorized functions” of the title block of each exercise
- You can use any built-in function, as long as it is not prohibited in the exercise
- Today, you can use only the following methods from Pandas: `io.sql.read_sql` and `to_sql`, unless otherwise explicitly prescribed in the exercise.
- Save and load all the required data in the subfolder `data/`

Chapter IV

Exercise 00 : Select

	Exercise 00
Select	
Turn-in directory : <i>ex00/</i>	
Files to turn in : <i>ex00_first_select.ipynb</i>	
Allowed functions : <code>import pandas as pd, import sqlite3</code>	

Sometimes only having Python in your toolbox can be limiting. Remember that during the first day of Piscine, you were working with command-line tools because they can be more efficient for some tasks. Today you will work with SQL. Why can it be useful for you? Sometimes your data might be not in convenient CSVs or JSONs but stored in a database, and you have to extract it from there somehow. Also, SQL can be used in Apache Spark and in Hive (but with HQL) tools that are used for processing big data.

Download [the SQLite database](#). Over the course of the day, you will work with different tables from it using Pandas. They are all connected and refer to the same project. It comprises a real dataset from an educational company. They have their own platform where every student can check if their solution is correct and receive some other feedback. The table checker stores the logs of when and which labs the users checked.

Лаба 5: Прогнозирование оттока клиентов

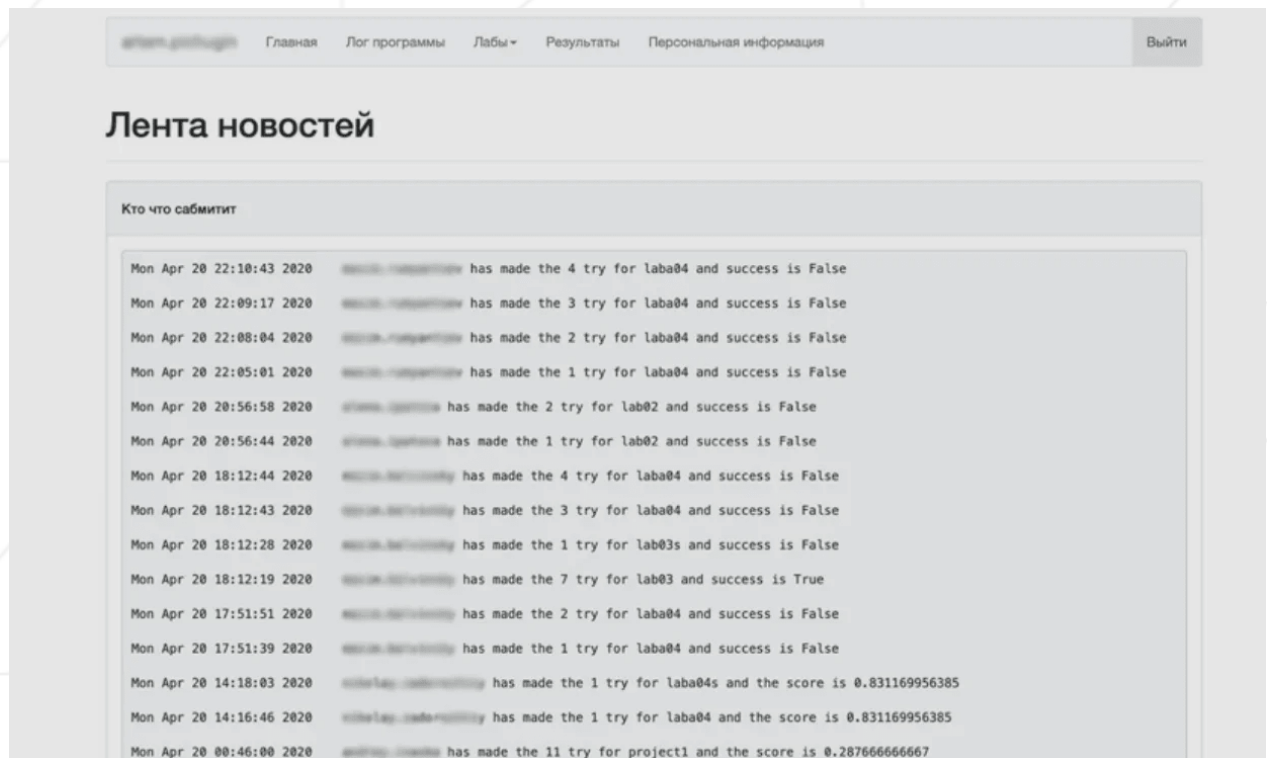
Вариант 1

✓ Проверить

↻ Обновить

The company decided to create a new page on the platform, Newsfeed, where those logs are visible to all the students in the program. The logs of the page visits are stored in another table – pageviews. The hypothesis was that the page would create peer pressure and the students would start working on the labs earlier. That could be good because they could make more iterations and try different approaches. In this series of exercises,

you will try to figure out if this hypothesis is correct.



But let us start with something super simple. In this exercise, you will need to get the filtered data from a table in the database. Why is it important to filter the data in the query but not afterward in Pandas? Because tables can be enormous. If you try to get the whole table, you will not be able to process it. Always keep this in mind.


The first method of filtering is to only choose those columns that you really need. The second is to choose the rows that you really need.

In more detail:

- put the database in the subfolder data in the root directory of the day
- create a connection to the database using the library sqlite3
- get the schema of the table pageviews using `pd.io.sql.read_sql` and the query `"PRAGMA table_info(pageviews);"`
- get only the first 10 rows of the table pageviews to check what the table looks like
- get the subtable using only one query where:
 - only uid and datetime are used
 - only user data (`user_*`) is used and not admin data
 - it is sorted by uid in ascending order
 - the index column is datetime
 - datetime is converted to `DatetimeIndex`
 - the name of the dataframe is `pageviews`
- close the connection to the database

Chapter V

Exercise 01 : Subquery

	Exercise 01
Subquery	
Turn-in directory : <i>ex01/</i>	
Files to turn in : ex01_subquery.ipynb	
Allowed functions : import pandas as pd, import sqlite3	


Ok, let us make something more complicated. Have you heard about subqueries? Like a query inside a query. Why can it be useful to you? In general, you might want to make some aggregations over a select that you had made before. Beware though that nested queries run first and alone, then the main query runs.

Here is what you need to do:

- create a connection to the database using the library `sqlite3`
- get the schema of the table checker
- get only the first 10 rows of the table checker to check what the table looks like
- count how many rows satisfy the following conditions using only one query with any number of subqueries:
 - count the rows from the pageviews table but only with users from the checker table with:
 - * `status = 'ready'`, we do not want to analyze the logs that are in status checking
 - * `numTrials = 1`, we want to analyze only the first commits, because only they can tell us when a student started working on a lab
 - * labnames should be from the list: `'laba04', 'laba04s', 'laba05', 'laba06', 'laba06s', 'project1'`. Only they were active during the experiment
 - store in the dataframe checkers with the column `cnt`
- close the connection

Chapter VI

Exercise 02 : Join

	Exercise 02
	Join
	Turn-in directory : <i>ex02/</i>
	Files to turn in : ex02_joins.ipynb
	Allowed functions : import pandas as pd, import sqlite3

In this exercise, you will create a so-called datamart. It is a table that can be used for analytics purposes. Usually, it is created by joining different tables together. In this exercise, we will collect various bits of data about our users: when they did their first commits, when they visited the newsfeed for the first time, etc. This will help us analyze it later.

What you need to do in this exercise (read the full task):


- create a connection to the database using the library `sqlite3`
- create a new table `datamart` in the database by joining the tables `pageviews` and `checker` using only one query
 - the table should have the following columns: `uid`, `labname`, `first_commit_ts`, `first_view_ts`
 - `first_commit_ts` is just a new name of the column `timestamp` from the `checker` table, it shows the first commit from a particular lab and from a particular user
 - `first_view_ts` is the first visit of a user to the table `pageviews`, timestamp when a user visited the newsfeed
 - `status = 'ready'` should still be a filter
 - `numTrials = 1` should still be a filter
 - `labnames` should still be from the list: `'laba04'`, `'laba04s'`, `'laba05'`, `'laba06'`, `'laba06s'`, `'project1'`
 - the table should contain only the users (`uids` with `user_*`) and not the admins

- `first_commit_ts` and `first_view_ts` should be parsed as `datetime64[ns]`
- using Pandas methods, create two dataframes: `test` and `control`
 - `test` should have the users that have the values in `first_view_ts`
 - `control` should have the users that have missing values in `first_view_ts`
 - replace the missing values in the `control` with the average `first_view_ts` of the `test` users, we will use this value for the future analysis
 - save both tables into the database, you will use them in the next exercises
- close the connection

A small piece of advice – do this step by step, from simple to more complex. It will help you to debug your queries.

Chapter VII

Exercise 03 : Aggregations

	Exercise 03
Aggregations	
Turn-in directory : <i>ex03/</i>	
Files to turn in : ex03_aggs.ipynb	
Allowed functions : <code>import pandas as pd, import sqlite3</code>	

What we did before was just data preparation. We have not gotten any insights into the data. It is time to change that. Remember how we had the hypothesis that the users would start working on the labs earlier if they saw the newsfeed? It means that the key metric for us is a delta between when a user started working on a lab (their first commit) and the deadline of the lab.


What you need to do in this exercise:

- create a connection to the database using the library `sqlite3`
- get the schema of the table `test`
- get only the first 10 rows of the table `test` to check what the table looks like
- find among all the users the minimum value of the delta between the first commit of the user and the deadline of the corresponding lab using only one query
 - do this by joining the table with the table `deadlines`
 - the difference should be displayed in hours
 - do not take the lab 'project1' into account, it has longer deadlines and will be an outlier
 - the value should be stored in the dataframe `df_min` with the corresponding `uid`
- do the same thing, but for the maximum, using only one query, the dataframe name is `df_max`

- do the same thing but for the average, using only one query, this time your dataframe should not include the uid column, and the dataframe name is df_avg
- we want to test the hypothesis that the users who visited the newsfeed just a few times have the lower delta between the first commit and the deadline. To do this, you need to calculate the correlation coefficient between the number of pageviews and the difference
 - using only one query, create a table with the columns: uid, avg_diff, pageviews
 - uid is the uids that exist in the test
 - avg_diff is the average delta between the first commit and the lab deadline per user
 - pageviews is the number of Newsfeed visits per user
 - do not take the lab 'project1' into account
 - store it to the dataframe views_diff
 - use the Pandas method corr() to calculate the correlation coefficient between the number of pageviews and the difference
- close the connection

Chapter VIII

Exercise 04 : A/B-testing

	Exercise 04
A/B-testing	
Turn-in directory : <i>ex04/</i>	
Files to turn in : <code>ex04_ab-test.ipynb</code>	
Allowed functions : <code>import pandas as pd, import sqlite3</code>	

So... let us finally find out if the Newsfeed affected the behavior of the students. Did they start to work on the labs earlier? Remember that we have two prepared tables in the database test and control. We are going to conduct something similar to an A/B-test. We need to calculate what the delta between the first commit and the deadline was before they visited the page for the first time and afterward. We need to do the same thing for the control group too.

In other words, each user in the test has their own timestamp for their first newsfeed visit. We want to calculate the average delta (first commit - deadline) before that timestamp and after that timestamp. We will do the same thing for the users in the control group. You may say: “but they did not visit the newsfeed at all”. That is correct, and we decided earlier to use the average timestamp of the first view from the test group for the users in the control group.

If the delta before the first Newsfeed visit is significantly different compared to the delta afterward in the test group, and we do not see the same effect in the control group, then creating the page was a great idea. We can roll it out to the whole group.

In more detail:

- create a connection to the database using the library `sqlite3`
- using only one query for each of the groups, create two dataframes: `test_results` and `control_results` with the columns `time` and `avg_diff` and only two rows
 - `time` should have the values: after and before
 - `avg_diff` contains the average delta among all the users for the time period before each of them made their first visit to the page and afterward

- only take into account the users that have observations before and after
- we still are not using the lab 'project1'
- close the connection
- have the answer: did the hypothesis turn out to be true and the page does affect the students' behavior?