td1a_sql_correction

January 1, 2023

1 1A.soft - Notions de SQL - correction

Correction des exercices du premier notebooks relié au SQL.

[1]:
```
from jyquickhelper import add_notebook_menu
add_notebook_menu()
```

[1]:
```
<IPython.core.display.HTML object>
```

[2]:
```
%matplotlib inline
```

1.1 Recupérer les données

[3]:
```
import os
if not os.path.exists("td8_velib.db3"):
    from pyensae.datasource import download_data
    download_data("td8_velib.zip", website = 'xd')
    from pyensae.sql import import_flatfile_into_database
    dbf = "td8_velib.db3"
    import_flatfile_into_database(dbf, "td8_velib.txt")                    # 2 secondes
    import_flatfile_into_database(dbf, "stations.txt", table="stations")  # 2 minutes
```

```
  TextFile: opening file  td8_velib.txt
  TextFile.guess_columns: processing file  td8_velib.txt
  TextFile: opening file  td8_velib.txt
  TextFile.guess_columns: using  101  lines
  TextFile: closing file  td8_velib.txt
  TextFile.guess_columns: sep  '\t' nb cols 7  bestnb  101  more  {('\t', 6):
101, (' ', 2): 100}
  TextFile.guess_columns: header  True  columns  {0: ('collect_date', <class
'str'>), 1: ('last_update', <class 'str'>), 2: ('available_bike_stands', <class
'int'>), 3: ('available_bikes', <class 'int'>), 4: ('number', <class 'int'>), 5:
('heure', <class 'int'>), 6: ('minute', <class 'int'>)}
  compiling ^(?P<collect_date>.*)\t(?P<last_update>.*)\t(?P<available_bike_stand
s>([-]?[1-9][0-9]*?)|(0?))\t(?P<available_bikes>([-]?[1-9][0-9]*?)|(0?))\t(?P<nu
mber>([-]?[1-9][0-9]*?)|(0?))\t(?P<heure>([-]?[1-9][0-9]*?)|(0?))\t(?P<minute>([
-]?[1-9][0-9]*?)|(0?))$
  TextFile.guess_columns: regex  ^(?P<collect_date>.*)\t(?P<last_update>.*)\t(?P
<available_bike_stands>([-]?[1-9][0-9]*?)|(0?))\t(?P<available_bikes>([-]?[1-9][
0-9]*?)|(0?))\t(?P<number>([-]?[1-9][0-9]*?)|(0?))\t(?P<heure>([-]?[1-9][0-9]*?)
|(0?))\t(?P<minute>([-]?[1-9][0-9]*?)|(0?))$
```

```
  TextFile.guess_columns: header  True  columns  {0: ('collect_date', (<class
'str'>, 52)), 1: ('last_update', (<class 'str'>, 38)), 2:
('available_bike_stands', <class 'int'>), 3: ('available_bikes', <class 'int'>),
4: ('number', <class 'int'>), 5: ('heure', <class 'int'>), 6: ('minute', <class
'int'>)}
  [_guess_columns] sep=['\t']
  TextFile: closing file  td8_velib.txt
  [_guess_columns] columns_name=None
   guess with  1001 lines
   count_types  {0: {<class 'str'>: 1000}, 1: {<class 'str'>: 1000}, 2: {<class
'int'>: 1000}, 3: {<class 'int'>: 1000}, 4: {<class 'int'>: 1000}, 5: {<class
'int'>: 1000}, 6: {<class 'int'>: 1000}}
   columns  {0: ('collect_date', <class 'str'>), 1: ('last_update', <class
'str'>), 2: ('available_bike_stands', <class 'int'>), 3: ('available_bikes',
<class 'int'>), 4: ('number', <class 'int'>), 5: ('heure', <class 'int'>), 6:
('minute', <class 'int'>)}
   guess {0: ('collect_date', (<class 'str'>, 52)), 1: ('last_update', (<class
'str'>, 38)), 2: ('available_bike_stands', <class 'int'>), 3:
('available_bikes', <class 'int'>), 4: ('number', <class 'int'>), 5: ('heure',
<class 'int'>), 6: ('minute', <class 'int'>)}
SQL  'CREATE TABLE td8_velib(collect_date TEXT,'
'       last_update TEXT,'
'       available_bike_stands INTEGER,'
'       available_bikes INTEGER,'
'       number INTEGER,'
'       heure INTEGER,'
'       minute INTEGER);'
   column_has_space False ['collect_date', 'last_update',
'available_bike_stands', 'available_bikes', 'number', 'heure', 'minute']
   changes {}
    TextFileColumns (2): regex:  {0: ('collect_date', (<class 'str'>, 52)), 1:
('last_update', (<class 'str'>, 38)), 2: ('available_bike_stands', <class
'int'>), 3: ('available_bikes', <class 'int'>), 4: ('number', <class 'int'>), 5:
('heure', <class 'int'>), 6: ('minute', <class 'int'>)}
  TextFile.guess_columns: processing file  td8_velib.txt
  TextFile: opening file  td8_velib.txt
  TextFile.guess_columns: using  101  lines
  TextFile: closing file  td8_velib.txt
  TextFile.guess_columns: sep  '\t' nb cols 7  bestnb  101  more  {('\t', 6):
101, (' ', 2): 100}
  TextFile.guess_columns: header  True  columns  {0: ('collect_date', <class
'str'>), 1: ('last_update', <class 'str'>), 2: ('available_bike_stands', <class
'int'>), 3: ('available_bikes', <class 'int'>), 4: ('number', <class 'int'>), 5:
('heure', <class 'int'>), 6: ('minute', <class 'int'>)}
  compiling ^(?P<collect_date>.*)\t(?P<last_update>.*)\t(?P<available_bike_stand
s>([-]?[1-9][0-9]*?)|(0?))\t(?P<available_bikes>([-]?[1-9][0-9]*?)|(0?))\t(?P<nu
mber>([-]?[1-9][0-9]*?)|(0?))\t(?P<heure>([-]?[1-9][0-9]*?)|(0?))\t(?P<minute>([
-]?[1-9][0-9]*?)|(0?))$
  TextFile.guess_columns: regex  ^(?P<collect_date>.*)\t(?P<last_update>.*)\t(?P
<available_bike_stands>([-]?[1-9][0-9]*?)|(0?))\t(?P<available_bikes>([-]?[1-9][
0-9]*?)|(0?))\t(?P<number>([-]?[1-9][0-9]*?)|(0?))\t(?P<heure>([-]?[1-9][0-9]*?)
|(0?))\t(?P<minute>([-]?[1-9][0-9]*?)|(0?))$
  TextFile.guess_columns: header  True  columns  {0: ('collect_date', (<class
'str'>, 52)), 1: ('last_update', (<class 'str'>, 38)), 2:
```

```
('available_bike_stands', <class 'int'>), 3: ('available_bikes', <class 'int'>),
4: ('number', <class 'int'>), 5: ('heure', <class 'int'>), 6: ('minute', <class
'int'>)}
  TextFile: opening file  td8_velib.txt
adding 100000 lines into table td8_velib
adding 200000 lines into table td8_velib
adding 300000 lines into table td8_velib
adding 400000 lines into table td8_velib
adding 500000 lines into table td8_velib
adding 600000 lines into table td8_velib
adding 700000 lines into table td8_velib
adding 800000 lines into table td8_velib
adding 900000 lines into table td8_velib
adding 1000000 lines into table td8_velib
adding 1100000 lines into table td8_velib
^(?P<collect_date>.*)\t(?P<last_update>.*)\t(?P<available_bike_stands>([-]?[1-9]
[0-9]*?)|(0?))\t(?P<available_bikes>([-]?[1-9][0-9]*?)|(0?))\t(?P<number>([-]?[1
-9][0-9]*?)|(0?))\t(?P<heure>([-]?[1-9][0-9]*?)|(0?))\t(?P<minute>([-]?[1-9][0-9
]*?)|(0?))$
error regex 0 unable to interpret line  1103788 :  ''
  TextFile: closing file  td8_velib.txt
1103787  lines imported
  TextFile: opening file  stations.txt
  TextFile.guess_columns: processing file  stations.txt
  TextFile: opening file  stations.txt
  TextFile.guess_columns: using  101  lines
  TextFile: closing file  stations.txt
  TextFile.guess_columns: sep  '\t' nb cols 6  bestnb  101  more  {('\t', 5):
101, (' ', 13): 3, (' ', 21): 1, (' ', 14): 2, (' ', 12): 11, (' ', 16): 2, ('
', 9): 24, (' ', 10): 21, (' ', 8): 13, (' ', 11): 10, (' ', 7): 12, (' ', 6):
1}
  TextFile.guess_columns: header  True  columns  {0: ('address', <class 'str'>),
1: ('contract_name', <class 'str'>), 2: ('lat', <class 'float'>), 3: ('lng',
<class 'float'>), 4: ('name', <class 'str'>), 5: ('number', <class 'int'>)}
  compiling ^(?P<address>.*)\t(?P<contract_name>.*)\t(?P<lat>[-]?[0-9]*?([.][0-9
]*?)?([eE][-]?[0-9]{0,4})?)\t(?P<lng>[-]?[0-9]*?([.][0-9]*?)?([eE][-]?[0-9]{0,4}
)?)\t(?P<name>.*)\t(?P<number>([-]?[1-9][0-9]*?)|(0?))$
  TextFile.guess_columns: regex  ^(?P<address>.*)\t(?P<contract_name>.*)\t(?P<la
t>[-]?[0-9]*?([.][0-9]*?)?([eE][-]?[0-9]{0,4})?)\t(?P<lng>[-]?[0-9]*?([.][0-9]*?
)?([eE][-]?[0-9]{0,4})?)\t(?P<name>.*)\t(?P<number>([-]?[1-9][0-9]*?)|(0?))$
  TextFile.guess_columns: header  True  columns  {0: ('address', (<class 'str'>,
134)), 1: ('contract_name', (<class 'str'>, 10)), 2: ('lat', <class 'float'>),
3: ('lng', <class 'float'>), 4: ('name', (<class 'str'>, 98)), 5: ('number',
<class 'int'>)}
  [_guess_columns] sep=['\t']
  TextFile: closing file  stations.txt
  [_guess_columns] columns_name=None
   guess with  935 lines
   count_types  {0: {<class 'str'>: 934}, 1: {<class 'str'>: 934}, 2: {<class
'float'>: 934}, 3: {<class 'float'>: 934}, 4: {<class 'str'>: 934}, 5: {<class
'int'>: 934}}
   columns  {0: ('address', <class 'str'>), 1: ('contract_name', <class 'str'>),
2: ('lat', <class 'float'>), 3: ('lng', <class 'float'>), 4: ('name', <class
'str'>), 5: ('number', <class 'int'>)}
```

```
   guess {0: ('address', (<class 'str'>, 152)), 1: ('contract_name', (<class
'str'>, 10)), 2: ('lat', <class 'float'>), 3: ('lng', <class 'float'>), 4:
('name', (<class 'str'>, 98)), 5: ('number', <class 'int'>)}
SQL  'CREATE TABLE stations(address TEXT,'
'       contract_name TEXT,'
'       lat FLOAT,'
'       lng FLOAT,'
'       name TEXT,'
'       number INTEGER);'
   column_has_space False ['address', 'contract_name', 'lat', 'lng', 'name',
'number']
   changes {}
    TextFileColumns (2): regex:  {0: ('address', (<class 'str'>, 152)), 1:
('contract_name', (<class 'str'>, 10)), 2: ('lat', <class 'float'>), 3: ('lng',
<class 'float'>), 4: ('name', (<class 'str'>, 98)), 5: ('number', <class
'int'>)}
  TextFile.guess_columns: processing file  stations.txt
  TextFile: opening file  stations.txt
  TextFile.guess_columns: using  101  lines
  TextFile: closing file  stations.txt
  TextFile.guess_columns: sep  '\t' nb cols 6  bestnb  101  more  {('\t', 5):
101, (' ', 13): 3, (' ', 21): 1, (' ', 14): 2, (' ', 12): 11, (' ', 16): 2, ('
', 9): 24, (' ', 10): 21, (' ', 8): 13, (' ', 11): 10, (' ', 7): 12, (' ', 6):
1}
  TextFile.guess_columns: header  True  columns  {0: ('address', <class 'str'>),
1: ('contract_name', <class 'str'>), 2: ('lat', <class 'float'>), 3: ('lng',
<class 'float'>), 4: ('name', <class 'str'>), 5: ('number', <class 'int'>)}
  compiling ^(?P<address>.*)\t(?P<contract_name>.*)\t(?P<lat>[-]?[0-9]*?([.][0-9
]*?)?([eE][-]?[0-9]{0,4})?)\t(?P<lng>[-]?[0-9]*?([.][0-9]*?)?([eE][-]?[0-9]{0,4}
)?)\t(?P<name>.*)\t(?P<number>([-]?[1-9][0-9]*?)|(0?))$
  TextFile.guess_columns: regex  ^(?P<address>.*)\t(?P<contract_name>.*)\t(?P<la
t>[-]?[0-9]*?([.][0-9]*?)?([eE][-]?[0-9]{0,4})?)\t(?P<lng>[-]?[0-9]*?([.][0-9]*?
)?([eE][-]?[0-9]{0,4})?)\t(?P<name>.*)\t(?P<number>([-]?[1-9][0-9]*?)|(0?))$
  TextFile.guess_columns: header  True  columns  {0: ('address', (<class 'str'>,
134)), 1: ('contract_name', (<class 'str'>, 10)), 2: ('lat', <class 'float'>),
3: ('lng', <class 'float'>), 4: ('name', (<class 'str'>, 98)), 5: ('number',
<class 'int'>)}
  TextFile: opening file  stations.txt
^(?P<address>.*)\t(?P<contract_name>.*)\t(?P<lat>[-]?[0-9]*?([.][0-9]*?)?([eE][-
]?[0-9]{0,4})?)\t(?P<lng>[-]?[0-9]*?([.][0-9]*?)?([eE][-]?[0-9]{0,4})?)\t(?P<nam
e>.*)\t(?P<number>([-]?[1-9][0-9]*?)|(0?))$
error regex 0 unable to interpret line  1232 :  ''
  TextFile: closing file  stations.txt
1231  lines imported
```

[4]:
```
%load_ext pyensae
```

[5]:
```
%SQL_connect td8_velib.db3
```

[5]:
```
<pyensae.sql.sql_interface_database.InterfaceSQLDatabase at 0x206823f4100>
```

1.2 Exercice 1

[6]:
```
%%SQL
SELECT COUNT(*) FROM (
    SELECT DISTINCT last_update FROM td8_velib
) ;
```

[6]:

	COUNT(*)
0	15294

[7]:
```
%%SQL
SELECT MIN(last_update), MAX(last_update) FROM td8_velib ;
```

[7]:

	MIN(last_update)	MAX(last_update)
0	2013-07-22 09:00:19	2013-09-13 11:25:19

1.3 Exercice 2

[8]:
```
%%SQL
SELECT number, COUNT(*) AS nb
FROM td8_velib
WHERE available_bikes==0 AND last_update >= '2013-09-10 11:30:19'
GROUP BY number
ORDER BY nb DESC
```

[8]:

	number	nb
0	42704	864
1	41101	864
2	35008	864
3	35005	864
4	35003	864
5	33012	864
6	33011	864
7	33006	864
8	33005	864
9	32603	864

1.4 Exercice 3 : plage horaires de cinq minutes où il n'y a aucun vélo disponible

[9]:
```
%%SQL
SELECT nb, COUNT(*) AS nb_station
FROM (
  -- requête de l'exercice précédent
  SELECT number, COUNT(*) AS nb
  FROM td8_velib
  WHERE available_bikes==0 AND last_update >= '2013-09-10 11:30:19'
  GROUP BY number
)
GROUP BY nb
```

[9]:

	nb	nb_station
0	1	7

```
1   2           7
2   3           5
3   4           6
4   5           6
5   6           5
6   7           8
7   8           9
8   9           1
9  10           7
```

1.5 Exercice 4 : distribution horaire par station et par tranche de 5 minutes

[10]:
```
%%SQL
SELECT A.number, A.heure, A.minute, 1.0 * A.nb_velo / B.nb_velo_tot  AS␣
 ↪distribution_temporelle
FROM (
  SELECT number, heure, minute, SUM(available_bikes) AS nb_velo
  FROM td8_velib
  WHERE last_update >= '2013-09-10 11:30:19'
  GROUP BY heure, minute, number
) AS A
JOIN (
  SELECT number, heure, minute, SUM(available_bikes) AS nb_velo_tot
  FROM td8_velib
  WHERE last_update >= '2013-09-10 11:30:19'
  GROUP BY number
) AS B
ON A.number == B.number
--WHERE A.number in (8001, 8003, 15024, 15031)  -- pour n'afficher que quelques␣
 ↪stations
ORDER BY A.number, A.heure, A.minute
```

[10]:

	number	heure	minute	distribution_temporelle
0	901	0	0	0.001104
1	901	0	5	0.001104
2	901	0	10	0.001104
3	901	0	15	0.001104
4	901	0	20	0.001104
5	901	0	25	0.000946
6	901	0	30	0.000946
7	901	0	35	0.000946
8	901	0	40	0.000946
9	901	0	45	0.000946

1.6 Zones de travail et zones de résidence

[11]:
```
%%SQL --df=df
SELECT number, SUM(distribution_temporelle) AS velo_jour
FROM (
  -- requête de l'exercice 4
    SELECT A.number, A.heure, A.minute, 1.0 * A.nb_velo / B.nb_velo_tot  AS␣
 ↪distribution_temporelle
```

```
    FROM (
        SELECT number, heure, minute, SUM(available_bikes) AS nb_velo
        FROM td8_velib
        WHERE last_update >= '2013-09-10 11:30:19'
        GROUP BY heure, minute, number
    ) AS A
    JOIN (
        SELECT number, heure, minute, SUM(available_bikes) AS nb_velo_tot
        FROM td8_velib
        WHERE last_update >= '2013-09-10 11:30:19'
        GROUP BY number
    ) AS B
    ON A.number == B.number
)
WHERE heure >= 10 AND heure <= 16
GROUP BY number
```

[11]:

	number	velo_jour
0	901	0.555188
1	903	0.473137
2	904	0.357640
3	905	0.323395
4	906	0.117566
5	908	0.501727
6	1001	0.486796
7	1002	0.355434
8	1003	0.364749
9	1004	0.440549

[12]:

```
df = df.sort_values("velo_jour").reset_index()
df["index"] = range(0, df.shape[0])
df.head()
```

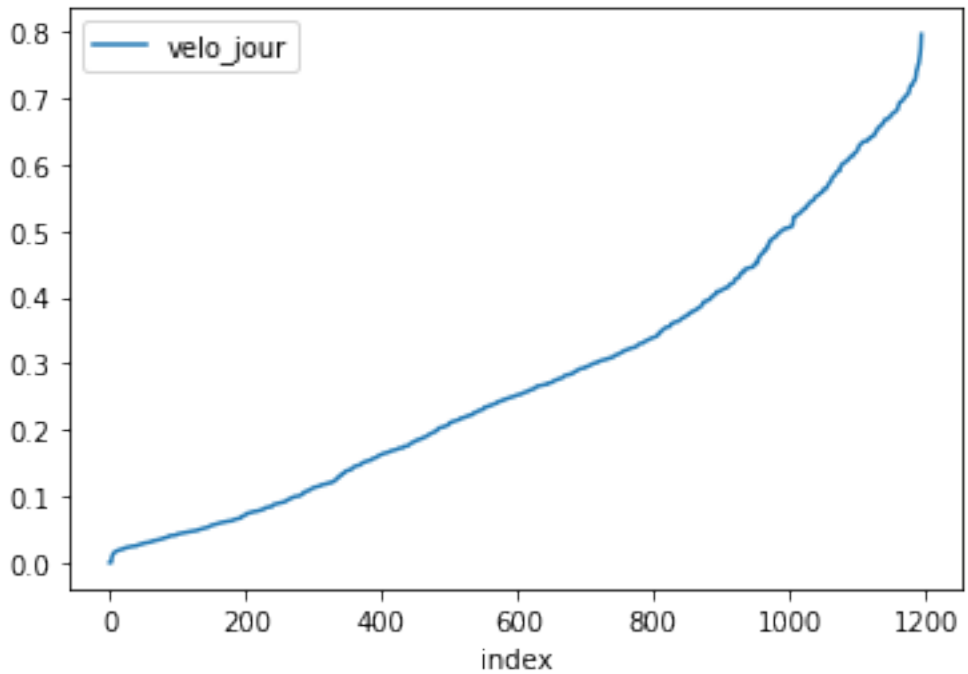
[12]:

	index	number	velo_jour
0	0	18113	0.000000
1	1	35013	0.000000
2	2	10010	0.000000
3	3	20037	0.007382
4	4	20119	0.010623

[13]:

```
df.plot(x="index", y="velo_jour")
```

[13]: <AxesSubplot:xlabel='index'>



1.7 JOIN avec la table stations et les stations "travail"

On trouve les arrondissements où les stations de vélib sont les plus remplies en journée au centre de Paris.

[14]:
```
%%SQL
SELECT C.number, name, lat, lng, velo_jour FROM
(
  -- requête de la partie précédente
    SELECT number, SUM(distribution_temporelle) AS velo_jour
    FROM (
        -- requête de l'exercice 4
        SELECT A.number, A.heure, A.minute, 1.0 * A.nb_velo / B.nb_velo_tot  AS␣
↪distribution_temporelle
        FROM (
            SELECT number, heure, minute, SUM(available_bikes) AS nb_velo
            FROM td8_velib
            WHERE last_update >= '2013-09-10 11:30:19'
            GROUP BY heure, minute, number
        ) AS A
        JOIN (
            SELECT number, heure, minute, SUM(available_bikes) AS nb_velo_tot
            FROM td8_velib
            WHERE last_update >= '2013-09-10 11:30:19'
            GROUP BY number
        ) AS B
        ON A.number == B.number
    )
    WHERE heure >= 10 AND heure <= 16
    GROUP BY number
```

```
) AS C
INNER JOIN stations
ON C.number == stations.number
```

[14]:

```
   number                                            name        lat  \
0     901            00901 - PORT SOLFÉRINO (STATION MOBILE)  48.861380
1     903              00903 - QUAI MAURIAC  / PONT DE BERCY  48.837134
2     904             00904 - PLACE JOFFRE / ECOLE MILITAIRE  48.852136
3     905  00905 - CONCORDE/BERGES DE SEINE (STATION MOBILE)  48.863140
4     906                              00906 - GARE DE L'EST  48.876420
5     908      00908 - PORT DU GROS CAILLOU (STATION MOBILE)  48.862880
6    1001                   01001 - ILE DE LA CITE PONT NEUF  48.857092
7    1002                          01002 - PLACE DU CHATELET  48.857940
8    1003                         01003 - RIVOLI SAINT DENIS  48.859150
9    1004                      01004 - MARGUERITE DE NAVARRE  48.859896

        lng  velo_jour
0  2.324420   0.555188
1  2.374341   0.473137
2  2.301961   0.357640
3  2.316690   0.323395
4  2.358630   0.117566
5  2.306520   0.501727
6  2.341748   0.486796
7  2.347010   0.355434
8  2.347620   0.364749
9  2.346757   0.440549
```

[15]:

```

```