hw66

October 8, 2018

0.0.1 Load the data

```
In [1]: import pandas as pd
        from datetime import datetime
        import pymysql
```

```
In [2]: department = pd.read_csv('departments.csv', sep=',')
        department.columns = ['department_name']
        shift = pd.read_csv('shifts.csv', sep=',')
        shift.columns = ['from_time', 'length']
        employees = pd.read_csv('employees.csv', sep=',')
        employees.emptype = employees.emptype.fillna('')
        schedule = pd.read_csv('schedule.csv', sep=',')
        schedule.columns = ['date', 'empid', 'dept', 'start_time', 'shift_length']
```

0.0.2 Manipulate the data

```
In [3]: class DataSqlLoader:
            def __init__(self, database):
                # connect to mysql local server
                self.db = pymysql.Connect(
                    host='localhost',
                    port=3308,
                    user='root',
                    passwd='',
                    db=database)
                self.c = self.db.cursor()

            # convert the shift and schedule time to `time` format compatible in MySQL
            def convert_time_format(self, time):
                return datetime.strptime('{}'.format(time), '%I%p').strftime('%H:%M:%S')

            # convert schedule date to `date` format compatible in MySQL
            def convert_date_format(self, time):
                return datetime.strptime('{}'.format(time), '%m/%d/%Y').strftime('%Y-%m-%d')

            def creat_tables(self):
                self.c.execute('''
```

```
        create table if not exists department
        (
          department_id  int auto_increment
            primary key,
          department_name varchar(50) not null
        );
        ''')
self.c.execute('''
    create table if not exists shift
    (
      shift_id int auto_increment
        primary key,
      from_time   time not null,
      length   int  not null
    );
    ''')
self.c.execute('''
    create table if not exists schedule
    (
      schedule_id  int auto_increment
        primary key,
      date         date        not null,
      empid        varchar(10) not null,
      dept         varchar(50) not null,
      start_time   time        not null,
      shift_length int         not null
    );
    ''')
self.c.execute('''
    create table if not exists employees
    (
      empid     varchar(10) not null primary key,
      lastname  varchar(20) not null,
      firstname varchar(20) not null,
      emptype   varchar(3)  null,
      cellphone varchar(20) null,
      homephone varchar(20) null,
      ftpt      varchar(2)  not null,
      constraint employee_empid_uindex
      unique (empid)
    );
''')

self.c.execute('''
    create table if not exists department_managers
    (
        dept_name varchar(20),
        manager varchar(30)
```

```
                );
            ''')

        def insert_into_tables(self, table, table_name):
            for i in range(len(table)):
                attributes = '{}'.format(tuple(table.columns.tolist())).replace("'","")
                query = "insert into {} {} values {};".format(
                    table_name, attributes, tuple(table.iloc[i,:].values))
                query = query.replace("(none)","")
                query = query.replace(r",)",")")
                # print(query)
                try:
                    self.c.execute(query)
                    self.db.commit()
                except Exception as e:
                    print(e)

        def close(self):
            self.db.close()
```

```
In [4]: dsl = DataSqlLoader('cs431_project')
        dsl.creat_tables()

        schedule['start_time'] = list(map(lambda x: dsl.convert_time_format(x),
                                          schedule['start_time']))
        shift['from_time'] = list(map(lambda x: dsl.convert_time_format(x),
                                      shift['from_time']))
        schedule['date'] = list(map(lambda x: dsl.convert_date_format(x),
                                    schedule['date']))

        dsl.insert_into_tables(department, 'department')
        dsl.insert_into_tables(shift, 'shift')
        dsl.insert_into_tables(schedule,'schedule')
        dsl.insert_into_tables(employees, 'employees')
```

0.0.3 SQL Query

1. GENERATE THE FOLLOWING REPORT LISTING ALL EMPLOYEES WHO ARE "OVER-BOOKED" BY BEING SCHEDULED MORE THAN ONCE IN A DAY

```
In [24]: query = '''
             SELECT a.empid, a.lastname,a.firstname, a.date AS Overbooked_Date,
                date_format(a.start_time, '%I%p') AS sfrom, a.dept_name, a.manager
                 FROM
                 (SELECT s.empid, e.lastname, e.firstname,
                   s.date, s.start_time, d.dept_name, d.manager
                 FROM employees AS e, schedule AS s, department_managers AS d
                 WHERE e.empid=s.empid
                 and d.dept_name=s.dept) AS a
```

```
            INNER JOIN
                ( SELECT empid, date, start_time FROM schedule
                GROUP BY empid, date
                  HAVING count(*) >1) AS b
            ON (a.empid, a.date)=(b.empid, b.date)
            ORDER BY a.empid, a.date, a.start_time;
                '''

        df = pd.read_sql(query, dsl.db)
        print(df.iloc[150:, :].to_latex())
```

```
\begin{tabular}{llllllll}
\toprule
{} &   empid &  lastname & firstname & Overbooked\_Date & sfrom &       dept\_name &          ma
\midrule
150 &  967577 &   Swallow &    Oralia &      2018-09-24 &  03PM &       MATERNITY &    Carmel Me
151 &  967577 &   Swallow &    Oralia &      2018-10-03 &  07AM &       MATERNITY &    Carmel Me
152 &  967577 &   Swallow &    Oralia &      2018-10-03 &  11PM &         SURGERY &   Nadene Har
153 &  967577 &   Swallow &    Oralia &      2018-10-07 &  07AM &        ONCOLOGY &  Junita Lora
154 &  967577 &   Swallow &    Oralia &      2018-10-07 &  07PM &        ONCOLOGY &  Junita Lora
155 &  971559 &     Mumaw &   Marylin &      2018-09-30 &  07AM &        ONCOLOGY &  Junita Lora
156 &  971559 &     Mumaw &   Marylin &      2018-09-30 &  07PM &        ONCOLOGY &  Junita Lora
157 &  979951 &  Berryman &   Sabrina &      2018-10-04 &  03PM &       EMERGENCY &   Oralia Swa
158 &  979951 &  Berryman &   Sabrina &      2018-10-04 &  11PM &  INTENSIVE CARE &     Colby Wi
159 &  979951 &  Berryman &   Sabrina &      2018-10-04 &  11PM &        ONCOLOGY &  Junita Lora
160 &  979951 &  Berryman &   Sabrina &      2018-10-06 &  07AM &      PEDIATRICS &     Remona L
161 &  979951 &  Berryman &   Sabrina &      2018-10-06 &  07AM &       MATERNITY &    Carmel Me
162 &  979951 &  Berryman &   Sabrina &      2018-10-06 &  07AM &        ONCOLOGY &  Junita Lora
163 &  992463 &     Goode &       Rex &      2018-09-25 &  11PM &        ONCOLOGY &  Junita Lora
164 &  992463 &     Goode &       Rex &      2018-09-25 &  11PM &         SURGERY &   Nadene Har
165 &  992463 &     Goode &       Rex &      2018-09-26 &  07AM &       MATERNITY &    Carmel Me
166 &  992463 &     Goode &       Rex &      2018-09-26 &  03PM &       MATERNITY &    Carmel Me
167 &  995561 &   Labarre &   Virgina &      2018-09-24 &  11PM &       EMERGENCY &   Oralia Swa
168 &  995561 &   Labarre &   Virgina &      2018-09-24 &  11PM &         SURGERY &   Nadene Har
169 &  995561 &   Labarre &   Virgina &      2018-10-01 &  07AM &  INTENSIVE CARE &     Colby Wi
170 &  995561 &   Labarre &   Virgina &      2018-10-01 &  11PM &      PEDIATRICS &     Remona L
171 &  995636 &       Pye &    Deanna &      2018-09-30 &  07AM &       MATERNITY &    Carmel Me
172 &  995636 &       Pye &    Deanna &      2018-09-30 &  07AM &       EMERGENCY &   Oralia Swa
173 &  995636 &       Pye &    Deanna &      2018-10-01 &  07AM &      CARDIOLOGY &    Marylin M
174 &  995636 &       Pye &    Deanna &      2018-10-01 &  11PM &       RADIOLOGY &   Pamella De
175 &  995636 &       Pye &    Deanna &      2018-10-05 &  07AM &  INTENSIVE CARE &     Colby Wi
176 &  995636 &       Pye &    Deanna &      2018-10-05 &  11PM &       MATERNITY &    Carmel Me
177 &  995636 &       Pye &    Deanna &      2018-10-07 &  07PM &       RADIOLOGY &   Pamella De
178 &  995636 &       Pye &    Deanna &      2018-10-07 &  07PM &  INTENSIVE CARE &     Colby Wi
\bottomrule
\end{tabular}
```

2. GENERATE A REPORT OF ALL OF THE FULL TIME PEOPLE WHO ARE WORKING MORE THAN 80 HOURS, AND HOW MUCH OVERTIME THEY ARE SCHEDULED FOR.

```
In [26]: query = '''
                 SELECT concat(e.firstname, ' ', e.lastname) as fullname,
                 CAST(sum(s.shift_length)-80 AS INT) as scheduled_overtime
                 FROM schedule AS s, employees AS e
                 WHERE s.empid=e.empid
                 AND e.ftpt = 'ft'
                 GROUP BY s.empid
                 HAVING scheduled_overtime >0
                 ORDER BY scheduled_overtime DESC, fullname ASC;
                 '''

         df = pd.read_sql(query, dsl.db)
         print(df.to_latex())
```

```
\begin{tabular}{llr}
\toprule
{} &           fullname &  scheduled\_overtime \\
\midrule
0  &        Vern Cermak &                  60 \\
1  &         Deanna Pye &                  48 \\
2  &     Jeffry Clyburn &                  44 \\
3  &       Remona Locke &                  36 \\
4  &    Charlott Nalley &                  32 \\
5  &     Manie Zaragoza &                  32 \\
6  &    Junita Loranger &                  24 \\
7  &   Imelda Catanzaro &                  20 \\
8  &       Kenton Hauck &                  20 \\
9  &     Asa Noonkester &                  16 \\
10 &         Mark Hinze &                  16 \\
11 &      Shalanda Ober &                  16 \\
12 &  Barton Mcclelland &                  12 \\
13 &     Oralia Swallow &                   8 \\
14 &    Virgina Labarre &                   8 \\
15 &      Marylin Mumaw &                   4 \\
\bottomrule
\end{tabular}
```

```
In [7]: query = 'show tables;'
        tables = pd.read_sql(query, dsl.db)
        tables
```

```
Out[7]:   Tables_in_cs431_project
        0              department
        1     department_managers
```

```
        2               employees
        3                schedule
        4                   shift

In [9]: tables.values[0]

Out[9]: array(['department'], dtype=object)

In [14]: for table in tables.values:
             query = 'describe {};'.format(table[0])
             df = pd.read_sql(query, dsl.db)
             print(df)
             print('\n')

             Field         Type Null  Key Default           Extra
0    department_id      int(11)   NO  PRI    None  auto_increment
1  department_name  varchar(50)   NO         None


       Field         Type Null Key Default Extra
0  dept_name  varchar(20)  YES        None
1    manager  varchar(30)  YES        None


       Field         Type Null  Key Default Extra
0      empid  varchar(10)   NO  PRI    None
1   lastname  varchar(20)   NO         None
2  firstname  varchar(20)   NO         None
3    emptype   varchar(3)  YES         None
4  cellphone  varchar(20)  YES         None
5  homephone  varchar(20)  YES         None
6       ftpt   varchar(2)   NO         None


          Field         Type Null  Key Default           Extra
0   schedule_id      int(11)   NO  PRI    None  auto_increment
1          date         date   NO         None
2         empid  varchar(10)   NO         None
3          dept  varchar(50)   NO         None
4    start_time         time   NO         None
5  shift_length      int(11)   NO         None


       Field     Type Null  Key Default           Extra
0   shift_id  int(11)   NO  PRI    None  auto_increment
1  from_time     time   NO         None
2     length  int(11)   NO         None
```

In [ ]: