Crash Course of Using Databases with Python

Lecture 3 of Application Part of 《Database Principles and Application》

TA: Cheng PENG
kris.dacpc@gmail.com

2018/06/27

Goal of This Lecture

Database Principles and Application

Goal of This Lecture

Database Principles and Application

"The goal of this course is to teach students the fundamental concepts underlying database system design, including not only the design of applications using databases, but also covering the fundamental implementation techniques used in database systems."

— 《Syllabus》Course Objectives

Goal of This Lecture

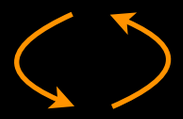
Database Principles and Application

"The goal of this course is to teach students the fundamental concepts underlying database system design, including not only the design of applications using databases, but also covering the fundamental implementation techniques used in database systems."

— 《Syllabus》 Course Objectives

SQL, E-R Diagram,
Relational Database Design,
Transaction…

Language: Python / Java / C#…



Database: Oracle / MySQL / SQLite…

Goal of This Lecture

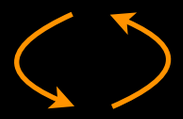
Database Principles and Application

"The goal of this course is to teach students the fundamental concepts underlying database system design, including not only the design of applications using databases, but also covering the fundamental implementation techniques used in database systems."

— 《Syllabus》 Course Objectives

SQL, E-R Diagram,
Relational Database Design,
Transaction…

*Language: **Python** / Java / C#…*



*Database: Oracle / **MySQL** / **SQLite**…*

Goal of This Lecture

If you know nothing about Python right now, After this lecture, you will get a basic sense of Python programming language, and have a preliminary knowledge of how to apply basic database operations (CRUD) with Python, and then know where to go deep if you are interested.

Content

- Quick Tutorial of Python
- Practice of Using Databases with Python
- Case: Excel Management

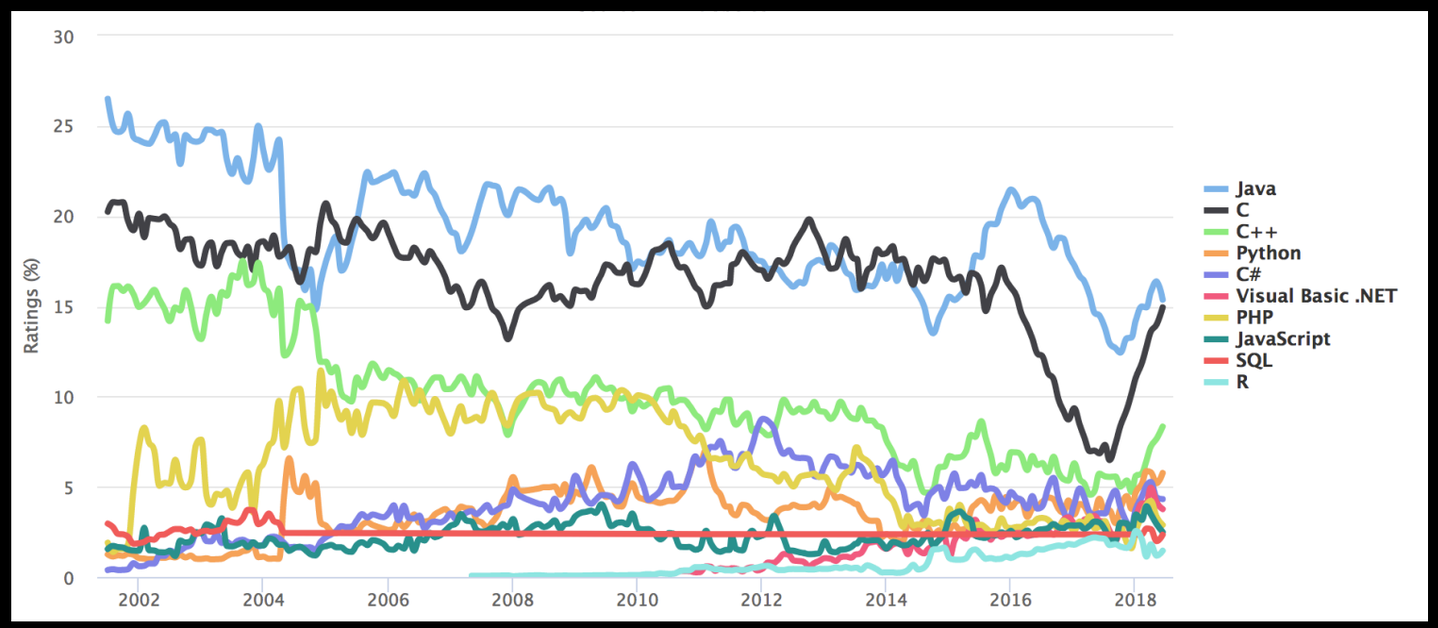
Main Refs:
[1] Python教程 廖雪峰
[2] 6 Reasons Why Python Is Suddenly Super Popular
[3] Using Databases with Python, Coursera
[4] SQLite official website

Content

- Quick Tutorial of Python
  - Why Python
  - Feature of Python
- Practice of Using Database with Python
- Case: Excel Management

Python is Popular

Jun 2018	Jun 2017	Change	Programming Language	Ratings	Change
1	1		Java	15.368%	+0.88%
2	2		C	14.936%	+8.09%
3	3		C++	8.337%	+2.61%
4	4		Python	5.761%	+1.43%
5	5		C#	4.314%	+0.78%
6	6		Visual Basic .NET	3.762%	+0.65%
7	8	⌃	PHP	2.881%	+0.11%
8	7	⌄	JavaScript	2.495%	-0.53%
9	-	⌃⌃	SQL	2.339%	+2.34%
10	14	⌃⌃	R	1.452%	-0.70%
11	11		Ruby	1.253%	-0.97%
12	18	⌃⌃	Objective-C	1.181%	-0.78%
13	16	⌃	Visual Basic	1.154%	-0.86%
14	9	⌄⌄	Perl	1.147%	-1.16%
15	12	⌄	Swift	1.145%	-1.06%

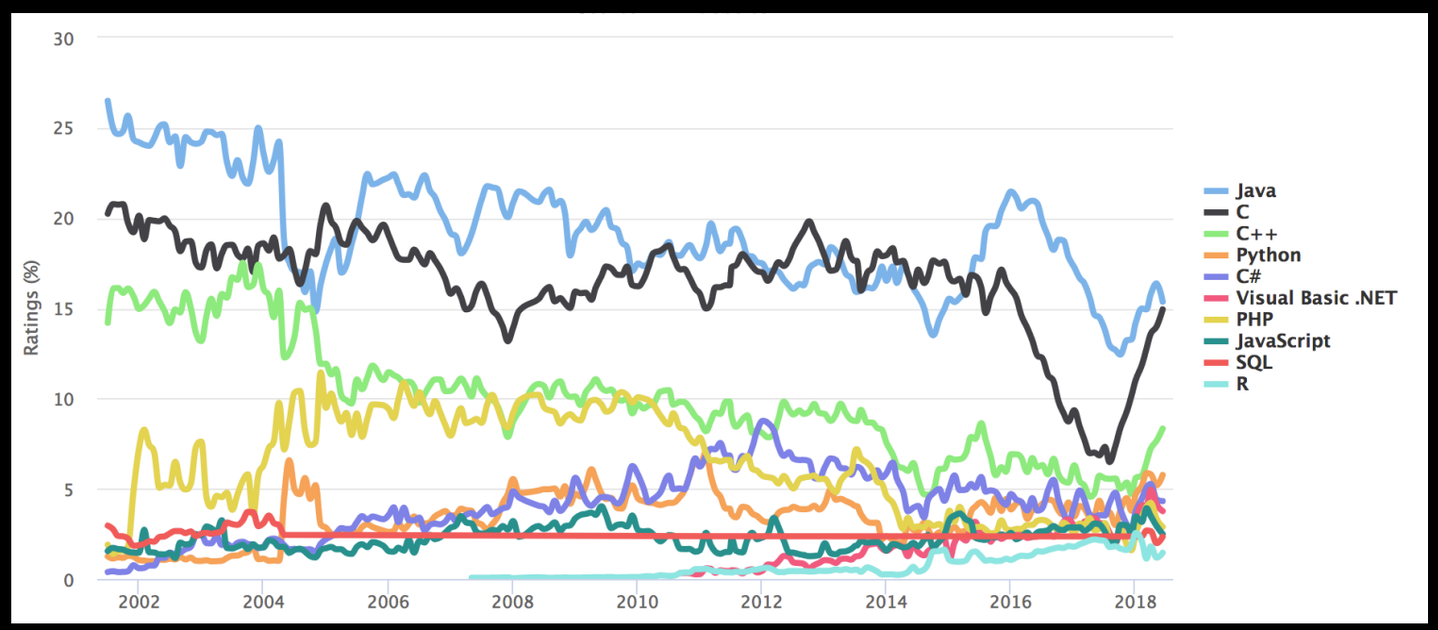


TIOBE Index for June 2018

Python is Popular

Jun 2018	Jun 2017	Change	Programming Language	Ratings	Change
1	1		Java	15.368%	+0.88%
2	2		C	14.936%	+8.09%
3	3		C++	8.337%	+2.61%
4	4		Python	5.761%	+1.43%
5	5		C#	4.314%	+0.78%
6	6		Visual Basic .NET	3.762%	+0.65%
7	8	⌃	PHP	2.881%	+0.11%
8	7	⌄	JavaScript	2.495%	-0.53%
9	-	⌃⌃	SQL	2.339%	+2.34%
10	14	⌃⌃	R	1.452%	-0.70%
11	11		Ruby	1.253%	-0.97%
12	18	⌃⌃	Objective-C	1.181%	-0.78%
13	16	⌃	Visual Basic	1.154%	-0.86%
14	9	⌄⌄	Perl	1.147%	-1.16%
15	12	⌄	Swift	1.145%	-1.06%

Main stream
&&
Increasing trend



TIOBE Index for June 2018

Reasons for Python

- *Python Has a Healthy, Active and Supportive Community*
- *Python Has Some Great Corporate Sponsors*
- *Python Has Big Data*
- *Python Has Amazing Libraries*
- *Python Is Reliable and Efficient*
- *Python Is Accessible*

Main Refs:
[1] 6 Reasons Why Python Is Suddenly Super Popular

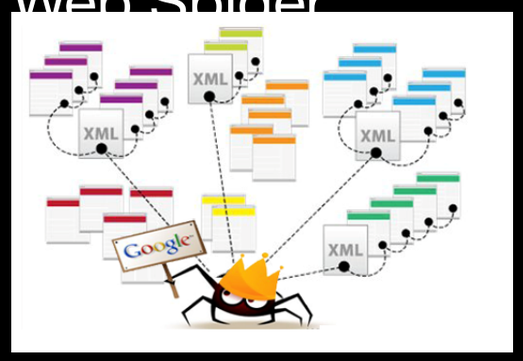
Reasons for Python

- *Python Has a Healthy, Active and Supportive Community*
- *Python Has Some Great Corporate Sponsors*
- *Python Has Big Data*
- *Python Has Amazing Libraries*
- *Python Is Reliable and Efficient*
- *Python Is Accessible*

Special Advantages

Python is Useful

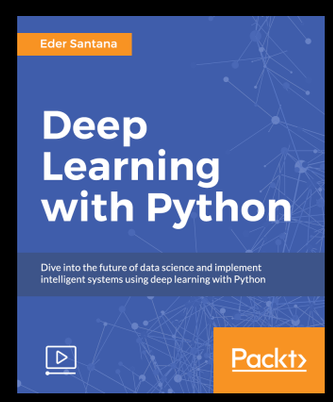
- Web Spider



- Big Data / Machine Learning / Deep Learning (AI, Data Science)

deeplearning.ai

Keras



- Database Operation

SQLAlchemy

SQLite

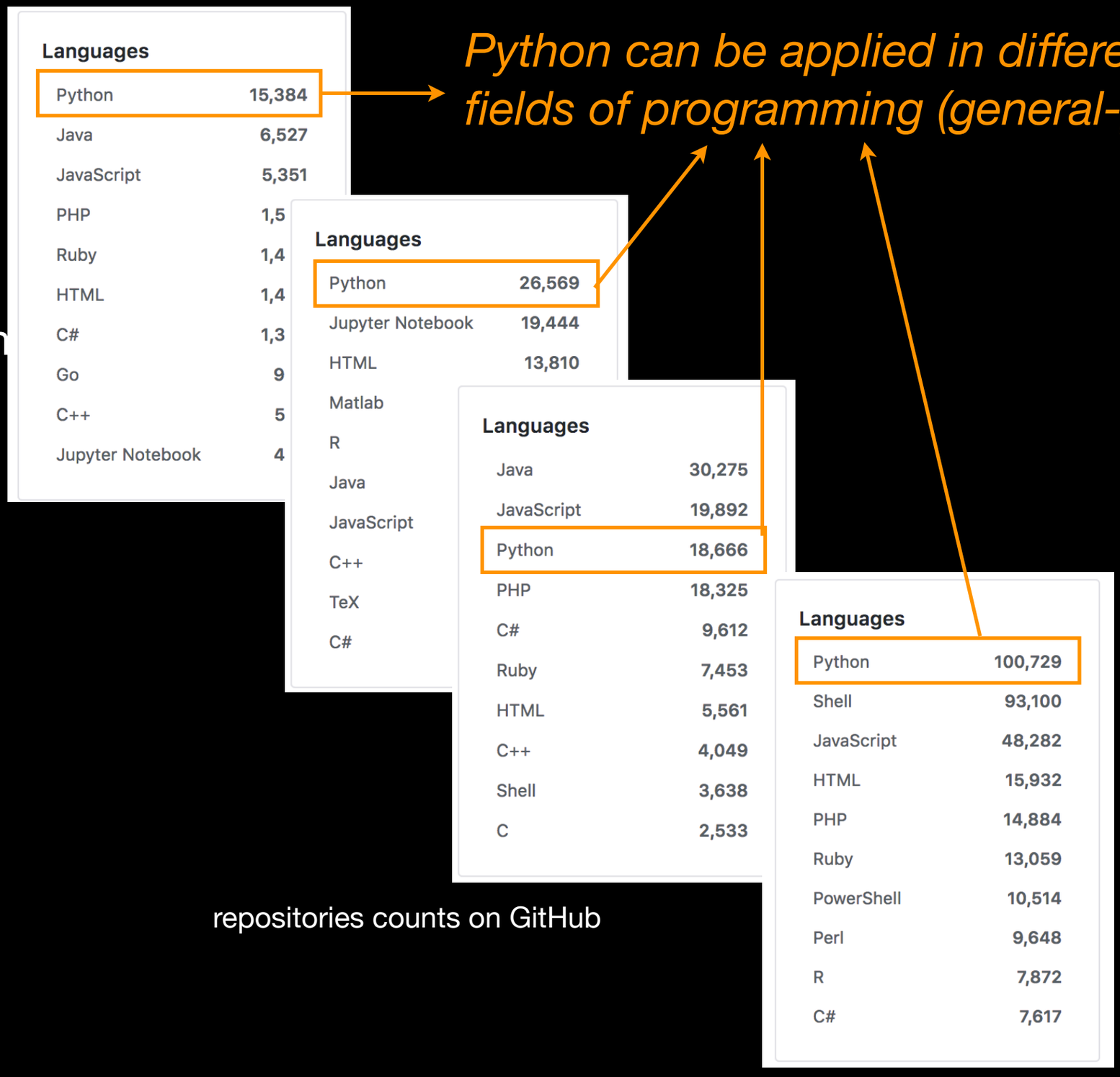
MySQL

- Script Handler

...

Python is Useful

- Web Spider
- Big Data / Machin
- Database
- Script Handler



Python can be applied in different fields of programming (general-purpose).

repositories counts on GitHub

Feature of Python

> *“Python is a programming language that lets you work more quickly and integrate your systems more effectively.”*
>
> — *python.org*

Feature of Python

> *“Python is a programming language that lets you work **more quickly** and integrate your systems **more effectively**.”*
>
> — *python.org*

Feature of Python

*“Python is a programming language that lets you work **more quickly** and integrate your systems **more effectively**.”*

— *python.org*

“Pythonic”

- Conciseness *(“Life is short, I use Python”)*
- Batteries Included
- …

Conciseness

```cpp
#include <iostream>
using namespace std;

int main()
{
      cout << "Hello, World!";
      return 0;
 }
```

Hello world in C++

```java
public class HelloWorld {
        public static void main(String[] args) {
                System.out.println("Hello World");
        }
}
```

Hello world in Java

Conciseness

```cpp
#include <iostream>
using namespace std;

int main()
{
      cout << "Hello, World!";
      return 0;
 }
```

Hello world in C++

```java
public class HelloWorld {
        public static void main(String[] args) {
                System.out.println("Hello World");
        }
}
```

Hello world in Java


```python
print("Hello World")
```

Hello world in Python

Just one single line!

In general, Python can implement the same functions with much less codes than Java/C++.

Conciseness (Cont.)

Change two Numbers

```
int t = a;
a = b;
b = t
```

C style

```
a, b = b, a
```

Python

Double all elements on a list

```
li = [1,2,3,4,5,6,7,8,9]
for i in range(len(li)):
    li[i] *=2
```

C style

```
li = [2*x for x in li]
```

or

```
li = list(map(lambda x:2*x,li))
```

Python

Batteries Included

Python have many well-designed packages, and you can use them directly to implement your idea quickly.

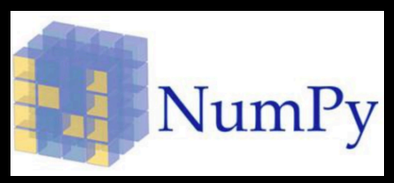
Particularly, on Machine Learning (AI) Field, Python has a complete ecosystem as follow:

Data Analysis and Visualization

Scipy

pandas

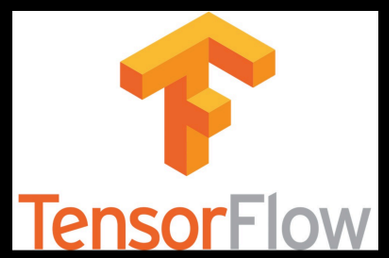
matplotlib



NumPy

Scikit-Learn

Deep Learning



TensorFlow

Keras

MxNet

... ...

Content

- Quick Tutorial of Python
- Practice of Using Databases with Python
  - SQLite with Python
  - ORM in Python
- Case: Excel Management

SQLite with Python

We firstly take SQLite as example here, which is integrated on Python3, you can use it directly without install any third-part libraries if you have finished set the Python environment correctly (to install Python, please click python.org).

A Lightweight, Embedded, User-Friendly database engine, which is a good start for Python Database Travel.

Main Refs:
[1] SQLite3 - Simple Databases with Python
[2] SQLite official website

Hello World on SQLite

3 Steps for SQLite:

- *Establish a connection*
- *Run the SQL commands*
- *Close the connection*

```
import sqlite3

conn = sqlite3.connect('tutorial.db')
cursor = conn.cursor()

def create_table():
    cursor.execute('CREATE TABLE IF NOT EXISTS user (id VARCHAR(20) PRIMARY KEY,\
    name VARCHAR(128),course VARCHAR(128), datastamp TEXT)')

def data_entry():
    cursor.execute("INSERT INTO user VALUES ('1', 'Cheng PENG', 'Python', '2018-06-27')")
    conn.commit()
    cursor.close()
    conn.close()

create_table()
data_entry()
```

Hello World on SQLite

3 Steps for SQLite:

- *Establish a connection*
- *Run the SQL commands*
- *Close the connection*

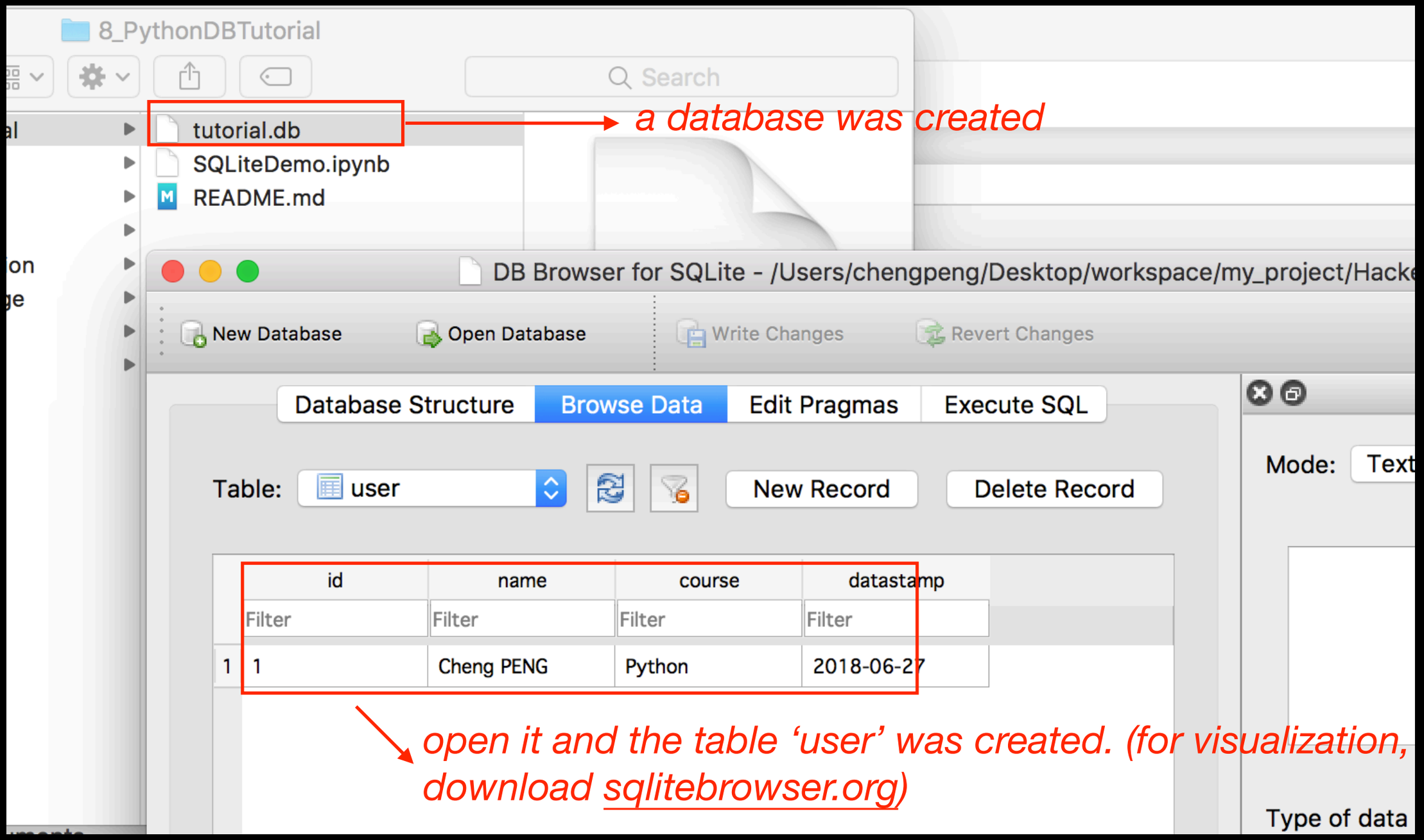
```
import sqlite3

conn = sqlite3.connect('tutorial.db')
cursor = conn.cursor()

def create_table():
    cursor.execute('CREATE TABLE IF NOT EXISTS user (id VARCHAR(20) PRIMARY KEY,\
    name VARCHAR(128),course VARCHAR(128), datastamp TEXT)')

def data_entry():
    cursor.execute("INSERT INTO user VALUES ('1', 'Cheng PENG', 'Python', '2018-06-27')")
    conn.commit()
    cursor.close()
    conn.close()

create_table()
data_entry()
```

1.establish the conncetion and use cursor

3.commit and close the connection

2.create the table and insert a column

Database was created



Insert More Data

```
def more_data():
    for i in range(10):
        current_time = time.time()
        date = str(datetime.datetime.fromtimestamp(current_time).strftime('%Y-%m-%d'))
        name = "Cheng PENG"
        course = "Python " + str(i)
        cursor.execute("INSERT INTO user (id, name, course, datastamp) \
        VALUES (?, ?, ?, ?)", (i, name, course, date))
        conn.commit()
    cursor.close()
    conn.close()
```

Database Structure | Browse Data | Edit Pragmas | Execute

Table: user | New Record | Delet

	id	name	course	datastamp
	Filter	Filter	Filter	Filter
1	0	Cheng PENG	Python 0	2018-06-20
2	1	Cheng PENG	Python 1	2018-06-20
3	2	Cheng PENG	Python 2	2018-06-20
4	3	Cheng PENG	Python 3	2018-06-20
5	4	Cheng PENG	Python 4	2018-06-20
6	5	Cheng PENG	Python 5	2018-06-20
7	6	Cheng PENG	Python 6	2018-06-20
8	7	Cheng PENG	Python 7	2018-06-20
9	8	Cheng PENG	Python 8	2018-06-20
10	9	Cheng PENG	Python 9	2018-06-20

Insert More Data

```
def more_data():
    for i in range(10):
        current_time = time.time()
        date = str(datetime.datetime.fromtimestamp(current_time).strftime('%Y-%m-%d'))
        name = "Cheng PENG"
        course = "Python " + str(i)
        cursor.execute("INSERT INTO user (id, name, course, datastamp) \
        VALUES (?, ?, ?, ?)", (i, name, course, date))
        conn.commit()
    cursor.close()
    conn.close()
```

Use placeholder "?" in SQLite
Update and Delete are similar to Insert, Just have a try!

```
def update_table():
    cursor.execute("UPDATE user SET name = 'Kris' where name = 'Cheng PENG'")
    conn.commit()
    cursor.close()
    conn.close()
```

Database Structure | Browse Data | Edit Pragmas | Execute

Table: user | New Record | Delet

	id	name	course	datastamp
	Filter	Filter	Filter	Filter
1	0	Cheng PENG	Python 0	2018-06-20
2	1	Cheng PENG	Python 1	2018-06-20
3	2	Cheng PENG	Python 2	2018-06-20
4	3	Cheng PENG	Python 3	2018-06-20
5	4	Cheng PENG	Python 4	2018-06-20
6	5	Cheng PENG	Python 5	2018-06-20
7	6	Cheng PENG	Python 6	2018-06-20
8	7	Cheng PENG	Python 7	2018-06-20
9	8	Cheng PENG	Python 8	2018-06-20
10	9	Cheng PENG	Python 9	2018-06-20

Fetch Data From Database

```
def read_from_db():
    cursor.execute("SELECT * FROM user")
    data = cursor.fetchall()
    print(data)
    for item in data:
        print(item)
create_table()
# more_data()
# data_entry()
read_from_db()
```

fetchall() returns a list which contains all the data

```
[('0', 'Cheng PENG', 'Python 0', '2018-06-20'), ('1', 'Cheng PENG', 'Python 1', '2018-06-20'), ('2', 'Cheng PENG', 'P
ython 2', '2018-06-20'), ('3', 'Cheng PENG', 'Python 3', '2018-06-20'), ('4', 'Cheng PENG', 'Python 4', '2018-06-2
0'), ('5', 'Cheng PENG', 'Python 5', '2018-06-20'), ('6', 'Cheng PENG', 'Python 6', '2018-06-20'), ('7', 'Cheng PEN
G', 'Python 7', '2018-06-20'), ('8', 'Cheng PENG', 'Python 8', '2018-06-20'), ('9', 'Cheng PENG', 'Python 9', '2018-0
6-20')]
('0', 'Cheng PENG', 'Python 0', '2018-06-20')
('1', 'Cheng PENG', 'Python 1', '2018-06-20')
('2', 'Cheng PENG', 'Python 2', '2018-06-20')
('3', 'Cheng PENG', 'Python 3', '2018-06-20')
('4', 'Cheng PENG', 'Python 4', '2018-06-20')
('5', 'Cheng PENG', 'Python 5', '2018-06-20')
('6', 'Cheng PENG', 'Python 6', '2018-06-20')
('7', 'Cheng PENG', 'Python 7', '2018-06-20')
('8', 'Cheng PENG', 'Python 8', '2018-06-20')
('9', 'Cheng PENG', 'Python 9', '2018-06-20')
```

print the result (unformatted and formatted)

Object-Relational Mapping

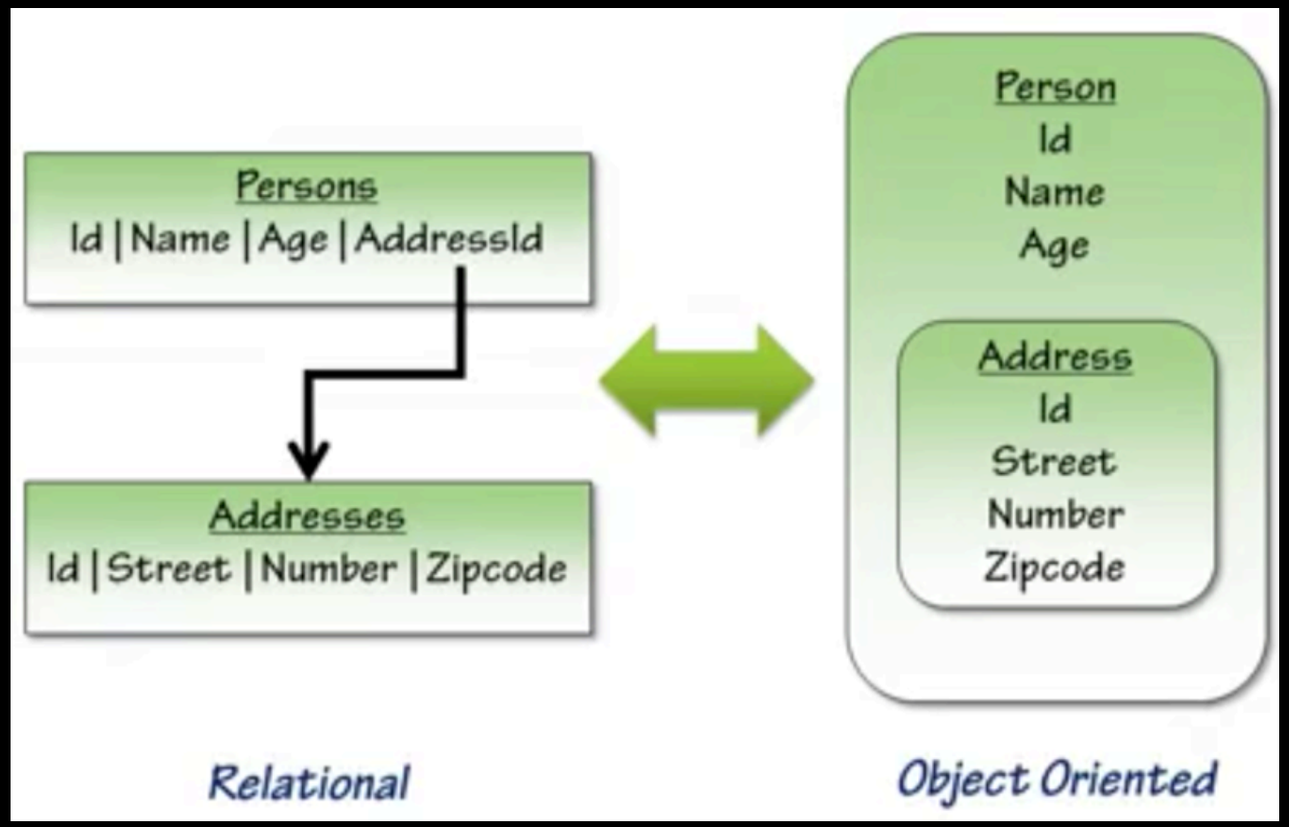
> *Object-relational mapping in computer science is a programming technique for converting data between incompatible type systems using object-oriented programming languages.*
>
> — *Wikipedia ORM*

Refs:
[1] What is an ORM (YouTube Video)
[2] Object-relational mapping (Wikipedia)

Object-Relational Mapping

Object-relational mapping in computer science is a programming technique for converting data between incompatible type systems using object-oriented programming languages.

— *Wikipedia ORM*



Database is not design for OOP, with ORM, you can focus on using objects *(eg: Java, C#, Python …)* rather than handle complex SQL statements.
In short, ORM creates a "Virtual Object Database".

Refs:
[1] What is an ORM (YouTube Video)
[2] Object-relational mapping (Wikipedia)

ORM with Python

Preliminaries:
SQLAlchemy— install via *“pip install sqlalchemy”*
MySQL— install via official tutorial, mysql.com

```
from sqlalchemy import Column, String,Integer,DateTime, create_engine
from sqlalchemy.orm import sessionmaker
from sqlalchemy.ext.declarative import declarative_base
from datetime import datetime
Base = declarative_base()

class User(Base):
    __tablename__ = 'user'
    id = Column(Integer, primary_key=True)
    name = Column(String(20))
    course = Column(String(20))
    create_time = Column(DateTime)


engine = create_engine('mysql+mysqlconnector://root:pengcheng00@localhost:3306/test')
DBSession = sessionmaker(bind=engine)

session = DBSession()
new_user = User(name='Cheng PENG', course="Database Principle", create_time=datetime.now())
session.add(new_user)
session.commit()
session.close()
```

ORM with Python

```python
from sqlalchemy import Column, String,Integer,DateTime, create_engine
from sqlalchemy.orm import sessionmaker
from sqlalchemy.ext.declarative import declarative_base
from datetime import datetime
Base = declarative_base()

class User(Base):
    __tablename__ = 'user'
    id = Column(Integer, primary_key=True)
    name = Column(String(20))
    course = Column(String(20))
    create_time = Column(DateTime)

engine = create_engine('mysql+mysqlconnector://root:pengcheng00@localhost:3306/test')
DBSession = sessionmaker(bind=engine)

session = DBSession()
new_user = User(name='Cheng PENG', course="Database Principle", create_time=datetime.now())
session.add(new_user)
session.commit()
session.close()
```

1. Define the Object

2.Create the object and commit

Search: id =

id	name	course	create_time
3	Cheng PENG	Database Principle	2018-06-20 17:37:49

3. a new column is created, no SQL needed!

Fetch Data Model

```
session = DBSession()
users = session.query(User).all()
for user in users:
    print('Name:', user.name)
    print('Course:', user.course)
    print('Create Time:', user.create_time)
session.close()
```

```
Name: Cheng PENG
Course: Database Principle
Create Time: 2018-06-20 17:37:49
```

Just use query, and you can add filter, which plays the role of "Where" part on SQL context.

Search: id =

id	name	course	create_time
3	Cheng PENG	Database Principle	2018-06-20 17:37:49

Record on Database

```
session = DBSession()
users = session.query(User).filter(User.name=="Kris").all()
for user in users:
    print('Name:', user.name)
    print('Course:', user.course)
    print('Create Time:', user.create_time)
session.close()
```

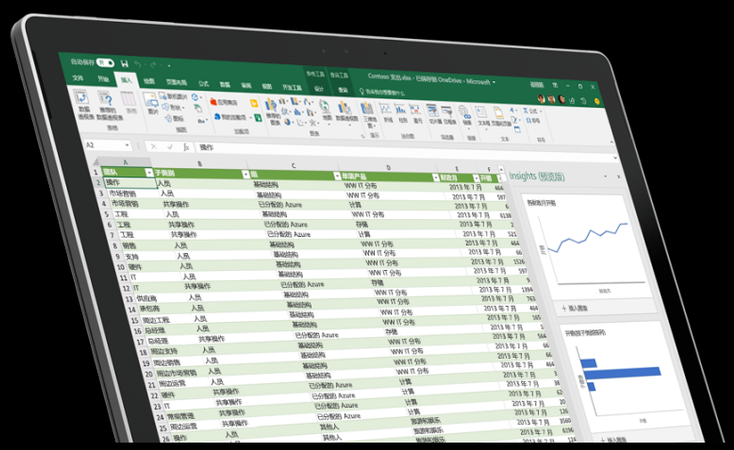
no suitable result returned

Content

- Quick Tutorial of Python
- Practice of Using Database with Python
- Case: Excel Management
  - Scene
  - Formulation
  - Steps
  - Conclusion

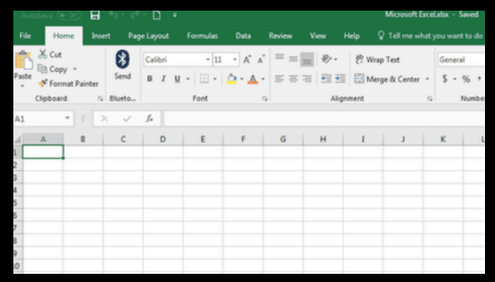
Scene

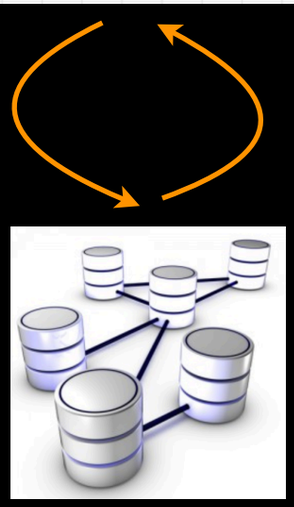
You are the manager of a car shop, right now you want to design a management system to manage your employees, but all the raw data you get from the data stewards are excels, if you want establish you system, you have to handle those mussy data first…



Formulation

raw data





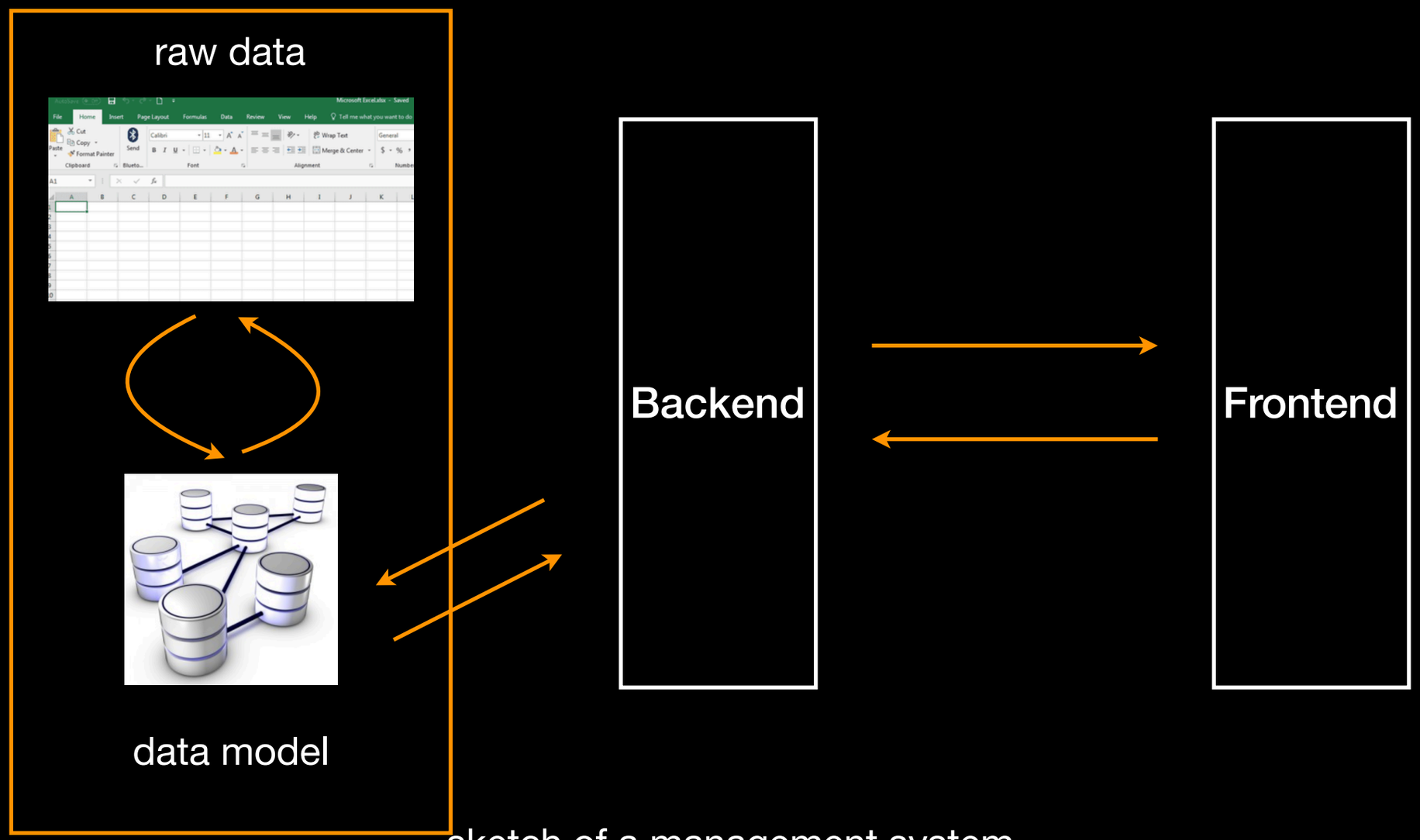
data model

Backend

Frontend

sketch of a management system

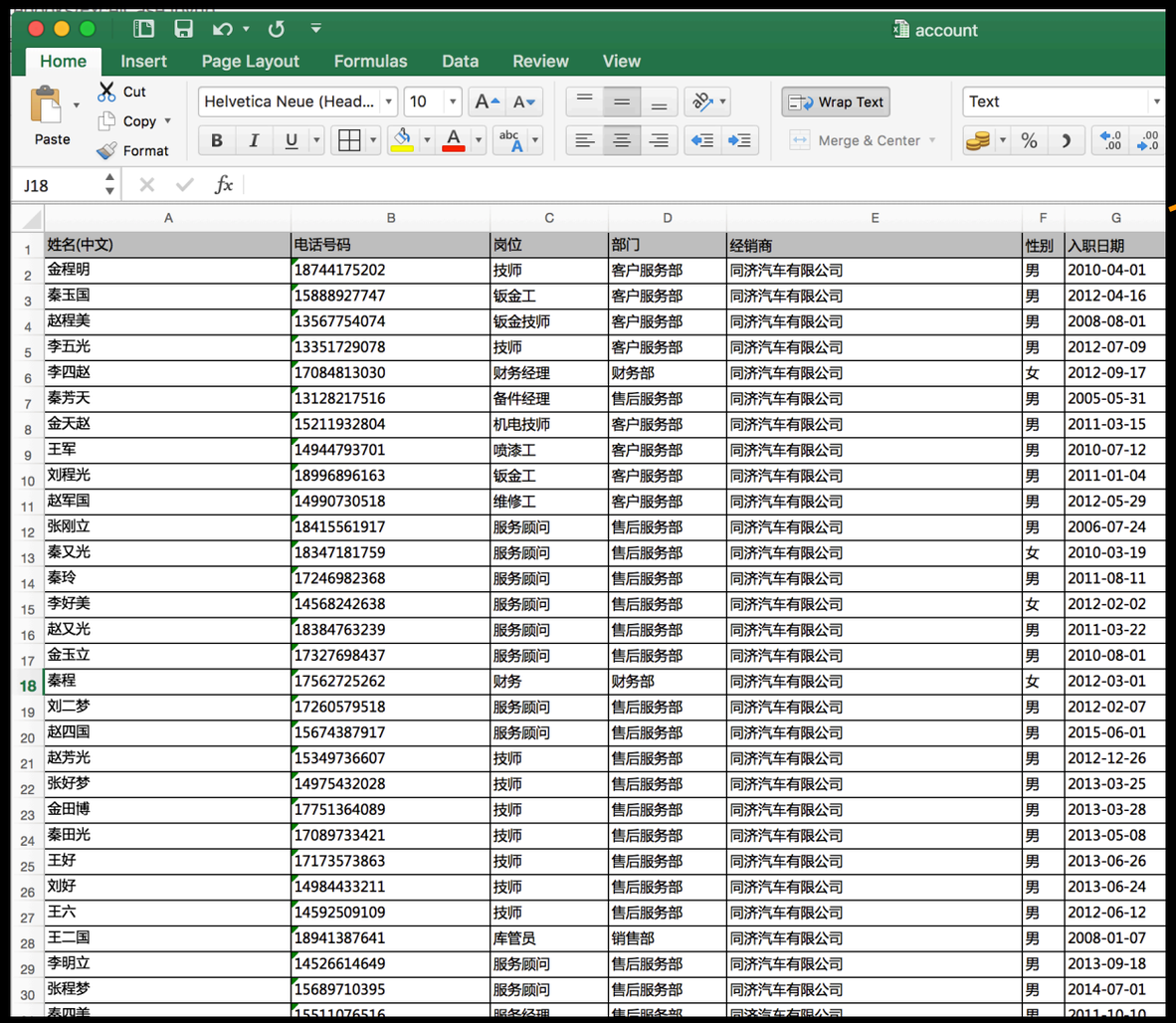
Formulation



sketch of a management system

What we learn today can handle this part — 1. Design Table; 2. Fetch data; 3. Load Data

Design Table



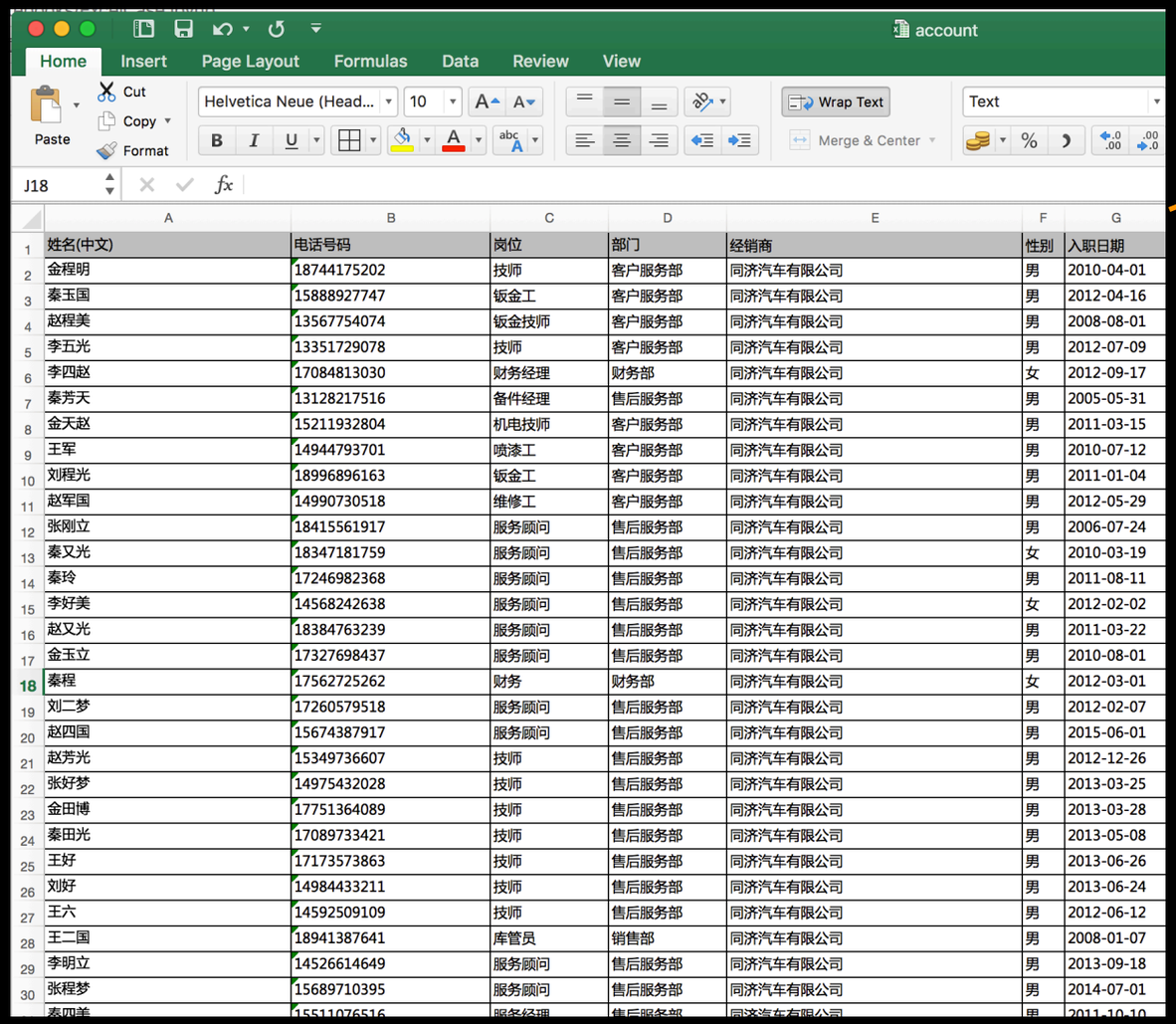
	A	B	C	D	E	F	G
1	姓名(中文)	电话号码	岗位	部门	经销商	性别	入职日期
2	金程明	18744175202	技师	客户服务部	同济汽车有限公司	男	2010-04-01
3	秦玉国	15888927747	钣金工	客户服务部	同济汽车有限公司	男	2012-04-16
4	赵程美	13567754074	钣金技师	客户服务部	同济汽车有限公司	男	2008-08-01
5	李五光	13351729078	技师	客户服务部	同济汽车有限公司	男	2012-07-09
6	李四赵	17084813030	财务经理	财务部	同济汽车有限公司	女	2012-09-17
7	秦芳天	13128217516	备件经理	售后服务部	同济汽车有限公司	男	2005-05-31
8	金天赵	15211932804	机电技师	客户服务部	同济汽车有限公司	男	2011-03-15
9	王军	14944793701	喷漆工	客户服务部	同济汽车有限公司	男	2010-07-12
10	刘程光	18996896163	钣金工	客户服务部	同济汽车有限公司	男	2011-01-04
11	赵军国	14990730518	维修工	客户服务部	同济汽车有限公司	男	2012-05-29
12	张刚立	18415561917	服务顾问	售后服务部	同济汽车有限公司	男	2006-07-24
13	秦又光	18347181759	服务顾问	售后服务部	同济汽车有限公司	女	2010-03-19
14	秦玲	17246982368	服务顾问	售后服务部	同济汽车有限公司	男	2011-08-11
15	李好美	14568242638	服务顾问	售后服务部	同济汽车有限公司	女	2012-02-02
16	赵又光	18384763239	服务顾问	售后服务部	同济汽车有限公司	男	2011-03-22
17	金玉立	17327698437	服务顾问	售后服务部	同济汽车有限公司	男	2010-08-01
18	秦程	17562725262	财务	财务部	同济汽车有限公司	女	2012-03-01
19	刘二梦	17260579518	服务顾问	售后服务部	同济汽车有限公司	男	2012-02-07
20	赵四国	15674387917	服务顾问	售后服务部	同济汽车有限公司	男	2015-06-01
21	赵芳光	15349736607	技师	售后服务部	同济汽车有限公司	男	2012-12-26
22	张好梦	14975432028	技师	售后服务部	同济汽车有限公司	男	2013-03-25
23	金田博	17751364089	技师	售后服务部	同济汽车有限公司	男	2013-03-28
24	秦田光	17089733421	技师	售后服务部	同济汽车有限公司	男	2013-05-08
25	王好	17173573863	技师	售后服务部	同济汽车有限公司	男	2013-06-26
26	刘好	14984433211	技师	售后服务部	同济汽车有限公司	男	2013-06-24
27	王六	14592509109	技师	售后服务部	同济汽车有限公司	男	2012-06-12
28	王二国	18941387641	库管员	销售部	同济汽车有限公司	男	2008-01-07
29	李明立	14526614649	服务顾问	售后服务部	同济汽车有限公司	男	2013-09-18
30	张程梦	15689710395	服务顾问	售后服务部	同济汽车有限公司	男	2014-07-01
	秦四美	15511076516	服务经理	售后服务部	同济汽车有限公司	男	2011-10-10

Table: account

Field	Type	Length
id	INT	11
name	VARCHAR	128
telephone_number	VARCHAR	128
position	VARCHAR	128
department	VARCHAR	128
company	VARCHAR	128
gender	VARCHAR	8
entry_time	DATE	
registered	TINYINT	1
password	VARCHAR	128
register_time	TIMESTAMP	
create_time	TIMESTAMP	

Disclaimer: all data are fake.

Design Table



	A	B	C	D	E	F	G
1	姓名(中文)	电话号码	岗位	部门	经销商	性别	入职日期
2	金程明	18744175202	技师	客户服务部	同济汽车有限公司	男	2010-04-01
3	秦玉国	15888927747	钣金工	客户服务部	同济汽车有限公司	男	2012-04-16
4	赵程美	13567754074	钣金技师	客户服务部	同济汽车有限公司	男	2008-08-01
5	李五光	13351729078	技师	客户服务部	同济汽车有限公司	男	2012-07-09
6	李四赵	17084813030	财务经理	财务部	同济汽车有限公司	女	2012-09-17
7	秦芳天	13128217516	备件经理	售后服务部	同济汽车有限公司	男	2005-05-31
8	金天赵	15211932804	机电技师	客户服务部	同济汽车有限公司	男	2011-03-15
9	王军	14944793701	喷漆工	客户服务部	同济汽车有限公司	男	2010-07-12
10	刘程光	18996896163	钣金工	客户服务部	同济汽车有限公司	男	2011-01-04
11	赵军国	14990730518	维修工	客户服务部	同济汽车有限公司	男	2012-05-29
12	张刚立	18415561917	服务顾问	售后服务部	同济汽车有限公司	男	2006-07-24
13	秦又光	18347181759	服务顾问	售后服务部	同济汽车有限公司	女	2010-03-19
14	秦玲	17246982368	服务顾问	售后服务部	同济汽车有限公司	男	2011-08-11
15	李好美	14568242638	服务顾问	售后服务部	同济汽车有限公司	女	2012-02-02
16	赵又光	18384763239	服务顾问	售后服务部	同济汽车有限公司	男	2011-03-22
17	金玉立	17327698437	服务顾问	售后服务部	同济汽车有限公司	男	2010-08-01
18	秦程	17562725262	财务	财务部	同济汽车有限公司	女	2012-03-01
19	刘二梦	17260579518	服务顾问	售后服务部	同济汽车有限公司	男	2012-02-07
20	赵四国	15674387917	服务顾问	售后服务部	同济汽车有限公司	男	2015-06-01
21	赵芳光	15349736607	技师	售后服务部	同济汽车有限公司	男	2012-12-26
22	张好梦	14975432028	技师	售后服务部	同济汽车有限公司	男	2013-03-25
23	金田博	17751364089	技师	售后服务部	同济汽车有限公司	男	2013-03-28
24	秦田光	17089733421	技师	售后服务部	同济汽车有限公司	男	2013-05-08
25	王好	17173573863	技师	售后服务部	同济汽车有限公司	男	2013-06-26
26	刘好	14984433211	技师	售后服务部	同济汽车有限公司	男	2013-06-24
27	王六	14592509109	技师	售后服务部	同济汽车有限公司	男	2012-06-12
28	王二国	18941387641	库管员	销售部	同济汽车有限公司	男	2008-01-07
29	李明立	14526614649	服务顾问	售后服务部	同济汽车有限公司	男	2013-09-18
30	张程梦	15689710395	服务顾问	售后服务部	同济汽车有限公司	男	2014-07-01

Disclaimer: all data are fake.

Table: account

Field	Type	Length
id	INT	11
name	VARCHAR	128
telephone_number	VARCHAR	128
position	VARCHAR	128
department	VARCHAR	128
company	VARCHAR	128
gender	VARCHAR	8
entry_time	DATE	
registered	TINYINT	1
password	VARCHAR	128
register_time	TIMESTAMP	
create_time	TIMESTAMP	

functionality consideration (registration)

basic information

Design Table

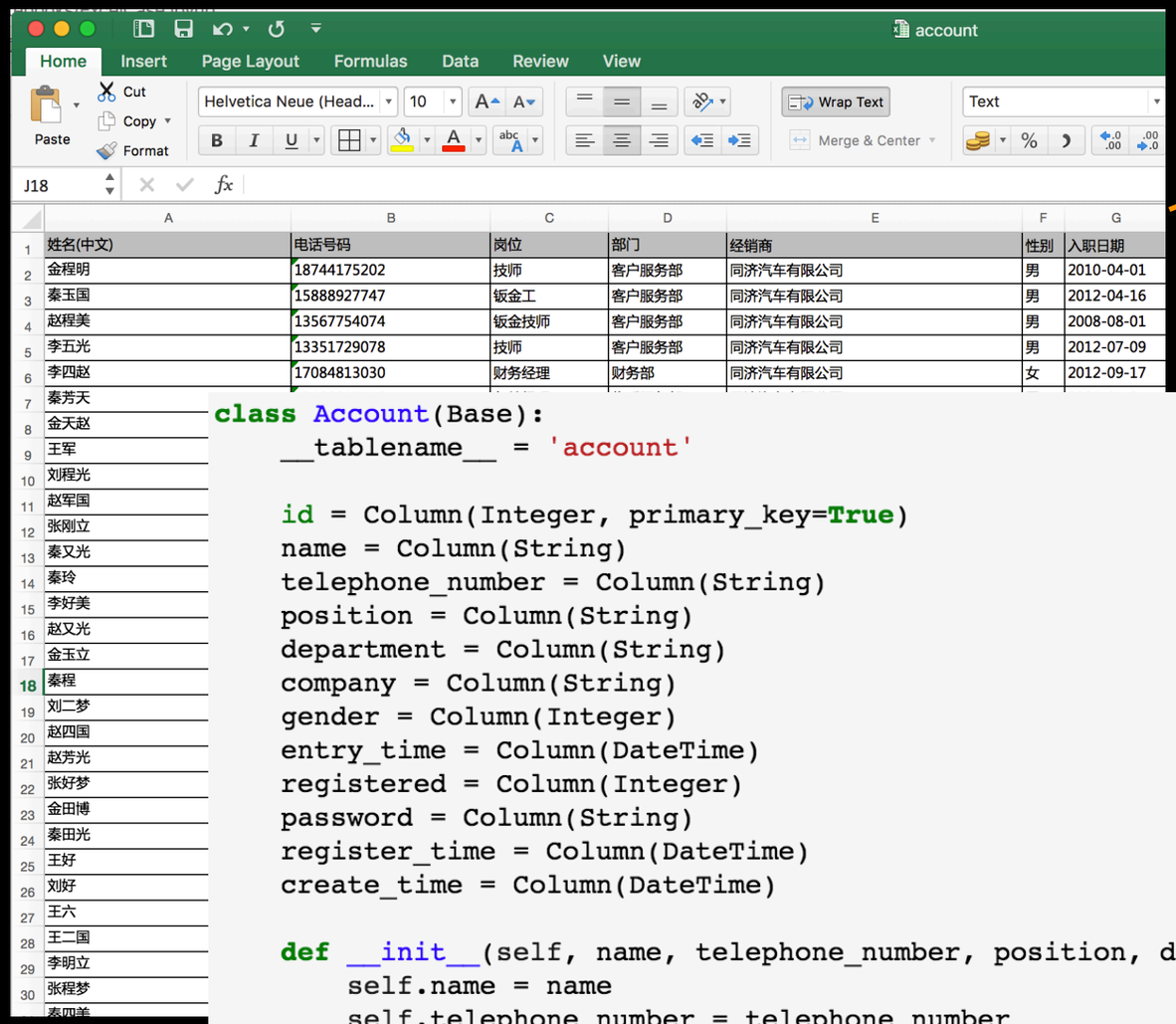


Table: account

Field	Type	Length
id	INT	11
		128
		128
		128
		128
		8
		1
		128

```python
class Account(Base):
    __tablename__ = 'account'

    id = Column(Integer, primary_key=True)
    name = Column(String)
    telephone_number = Column(String)
    position = Column(String)
    department = Column(String)
    company = Column(String)
    gender = Column(Integer)
    entry_time = Column(DateTime)
    registered = Column(Integer)
    password = Column(String)
    register_time = Column(DateTime)
    create_time = Column(DateTime)

    def __init__(self, name, telephone_number, position, department, company, gender, entry_time):
        self.name = name
        self.telephone_number = telephone_number
        self.position = position
        self.department = department
        self.company = company
        self.gender = gender
        self.entry_time = entry_time
        self.registered = 0
        self.create_time = datetime.now()
```

Account Object

Fetch Data

Preliminaries:
openpyxl —>*“pip install openpyxl”*

```
def load_data():
    file_name = "account.xlsx"
    wb = load_workbook(filename = file_name, data_only=True, read_only=True)
    ws = wb.get_sheet_by_name(name="account")
    idx = [cell.value for cell in ws[1]]
    list = []
    for row in ws.iter_rows(row_offset=1):
        dict = {}
        for cell in row:
            cell_value = cell.value
            if cell_value is not None:
                cell_key = idx[cell.column - 1]
                if isinstance(cell_value, datetime):
                    cell_value = cell_value.strftime("%Y-%m-%d")
                elif "日期" in cell_key:
                    cell_value = dateutil.parser.parse(cell_value).date().strftime("%Y-%m-%d")
                dict[cell_key] = str(cell_value)
        if dict:
            list.append(dict)
    return list
```

Fetch Data

Preliminaries:
openpyxl —>*“pip install openpyxl”*

```
def load_data():
    file_name = "account.xlsx"
    wb = load_workbook(filename = file_name, data_only=True, read_only=True)
    ws = wb.get_sheet_by_name(name="account")
    idx = [cell.value for cell in ws[1]]
    list = []
    for row in ws.iter_rows(row_offset=1):
        dict = {}
        for cell in row:
            cell_value = cell.value
            if cell_value is not None:
                cell_key = idx[cell.column - 1]
                if isinstance(cell_value, datetime):
                    cell_value = cell_value.strftime("%Y-%m-%d")
                elif "日期" in cell_key:
                    cell_value = dateutil.parser.parse(cell_value).date().strftime("%Y-%m-%d")
                dict[cell_key] = str(cell_value)
        if dict:
            list.append(dict)
    return list
```

1.as formal program, set them into configuration files!

2.please take care of some data format in Excel and handle them respectively!

Load Data

```
def list_to_db(list):
    for item in list:
        data_result = Account(
        name=item["姓名(中文)"],
        telephone_number=item["电话号码"],
        position=item["岗位"],
        department=item["部门"],
        company=item["经销商"],
        gender=item["性别"],
        entry_time=item["入职日期"]
        )
        session.add(data_result)

list = load_data()
engine = create_engine('mysql+mysqlconnector://root:pengcheng00@localhost:3306/test')
DBSession = sessionmaker(bind=engine)
session = DBSession()
try:
    list_to_db(list)
    session.commit()
except:
    print("DATA LOAD ERROR")
    session.rollback()
    raise
finally:
    print("DATA LOAD END")
    session.close()
```

Load Data

```python
def list_to_db(list):
    for item in list:
        data_result = Account(
        name=item["姓名(中文)"],
        telephone_number=item["电话号码"],
        position=item["岗位"],
        department=item["部门"],
        company=item["经销商"],
        gender=item["性别"],
        entry_time=item["入职日期"]
        )
        session.add(data_result)

list = load_data()
engine = create_engine('mysql+mysqlconnector://root:pengcheng00@localhost:3306/test')
DBSession = sessionmaker(bind=engine)
session = DBSession()
try:
    list_to_db(list)
    session.commit()
except:
    print("DATA LOAD ERROR")
    session.rollback()
    raise
finally:
    print("DATA LOAD END")
    session.close()
```

load data model (Account) one by one

handle exception, for more complex cases, make sure your program satisfy ACID properties.

Load Data

TABLES

account

Search: id = Search

id	name	telephone_number	position	department	company	gender	entry_time	registered	password	register_time	create_time
1	金程明	18744175202	技师	客户服务部	同济汽车有限公司	男	2010-04-01	0	NULL	NULL	2018-06-24 11:06:27
2	秦玉国	15888927747	钣金工	客户服务部	同济汽车有限公司	男	2012-04-16	0	NULL	NULL	2018-06-24 11:06:27
3	赵程美	13567754074	钣金技师	客户服务部	同济汽车有限公司	男	2008-08-01	0	NULL	NULL	2018-06-24 11:06:27
4	李五光	13351729078	技师	客户服务部	同济汽车有限公司	男	2012-07-09	0	NULL	NULL	2018-06-24 11:06:27
5	李四赵	17084813030	财务经理	财务部	同济汽车有限公司	女	2012-09-17	0	NULL	NULL	2018-06-24 11:06:27
6	秦芳天	13128217516	备件经理	售后服务部	同济汽车有限公司	男	2005-05-31	0	NULL	NULL	2018-06-24 11:06:27
7	金天赵	15211932804	机电技师	客户服务部	同济汽车有限公司	男	2011-03-15	0	NULL	NULL	2018-06-24 11:06:27
8	王军	14944793701	喷漆工	客户服务部	同济汽车有限公司	男	2010-07-12	0	NULL	NULL	2018-06-24 11:06:27
9	刘程光	18996896163	钣金工	客户服务部	同济汽车有限公司	男	2011-01-04	0	NULL	NULL	2018-06-24 11:06:27
10	赵军国	14990730518	维修工	客户服务部	同济汽车有限公司	男	2012-05-29	0	NULL	NULL	2018-06-24 11:06:27
11	张刚立	18415561917	服务顾问	售后服务部	同济汽车有限公司	男	2006-07-24	0	NULL	NULL	2018-06-24 11:06:27
12	秦又光	18347181759	服务顾问	售后服务部	同济汽车有限公司	女	2010-03-19	0	NULL	NULL	2018-06-24 11:06:27
13	秦玲	17246982368	服务顾问	售后服务部	同济汽车有限公司	男	2011-08-11	0	NULL	NULL	2018-06-24 11:06:27
14	李好美	14568242638	服务顾问	售后服务部	同济汽车有限公司	女	2012-02-02	0	NULL	NULL	2018-06-24 11:06:27
15	赵又光	18384763239	服务顾问	售后服务部	同济汽车有限公司	男	2011-03-22	0	NULL	NULL	2018-06-24 11:06:27
16	金玉立	17327698437	服务顾问	售后服务部	同济汽车有限公司	男	2010-08-01	0	NULL	NULL	2018-06-24 11:06:27
17	秦程	17562725262	财务	财务部	同济汽车有限公司	女	2012-03-01	0	NULL	NULL	2018-06-24 11:06:27
18	刘二梦	17260579518	服务顾问	售后服务部	同济汽车有限公司	男	2012-02-07	0	NULL	NULL	2018-06-24 11:06:27
19	赵四国	15674387917	服务顾问	售后服务部	同济汽车有限公司	男	2015-06-01	0	NULL	NULL	2018-06-24 11:06:27
20	赵芳光	15349736607	技师	售后服务部	同济汽车有限公司	男	2012-12-26	0	NULL	NULL	2018-06-24 11:06:27
21	张好梦	14975432028	技师	售后服务部	同济汽车有限公司	男	2013-03-25	0	NULL	NULL	2018-06-24 11:06:27
22	金田博	17751364089	技师	售后服务部	同济汽车有限公司	男	2013-03-28	0	NULL	NULL	2018-06-24 11:06:27
23	秦田光	17089733421	技师	售后服务部	同济汽车有限公司	男	2013-05-08	0	NULL	NULL	2018-06-24 11:06:27
24	王好	17173573863	技师	售后服务部	同济汽车有限公司	男	2013-06-26	0	NULL	NULL	2018-06-24 11:06:27
25	刘好	14984433211	技师	售后服务部	同济汽车有限公司	男	2013-06-24	0	NULL	NULL	2018-06-24 11:06:27
26	王六	14592509109	技师	售后服务部	同济汽车有限公司	男	2012-06-12	0	NULL	NULL	2018-06-24 11:06:27
27	王二国	18941387641	库管员	销售部	同济汽车有限公司	男	2008-01-07	0	NULL	NULL	2018-06-24 11:06:27
28	李明立	14526614649	服务顾问	售后服务部	同济汽车有限公司	男	2013-09-18	0	NULL	NULL	2018-06-24 11:06:27
29	张程梦	15689710395	服务顾问	售后服务部	同济汽车有限公司	男	2014-07-01	0	NULL	NULL	2018-06-24 11:06:27

data has been loaded, then you can establish your management system right now!

Further

Codes of this Lecture
https://github.com/KrisCheng/HackerProblem/tree/master/Python/8_PythonDBTutorial

Python教程（廖雪峰）
https://www.liaoxuefeng.com/wiki/001431608955726 4a6b348958f449949df42a6d3a2e542c000

SQLite Docs
https://www.sqlite.org/index.html

SQLAlchemy Docs
https://www.sqlalchemy.org/

Using Databases with Python (Coursera)
https://www.coursera.org/learn/python-databases

Machine Learning with Python *(if you are interested)*
https://machinelearningmastery.com/