※ 파이썬에서 MySQL을 연동하여 프로그램을 작성하기 위한 개발 환경을 만들자.

Ⅰ. 파이썬에서 MySQL에 접속하기 위한 pymysql 설치하자

파이썬에서 대체로 오라클에서 공개한 mysql.connector라는 것을 사용해서 MySQL 처리를 했었는데, 파이썬 2.7.x에서 3.5로 갈아타면서 접속 라이브러리가 버전별로 달라서 사용자들이 어려움을 격고 있다. 그러나 파이썬은 pip를 이용하여 MySQL 접속 패키지인 pymysql을 쉽게 설치할 수 있다.

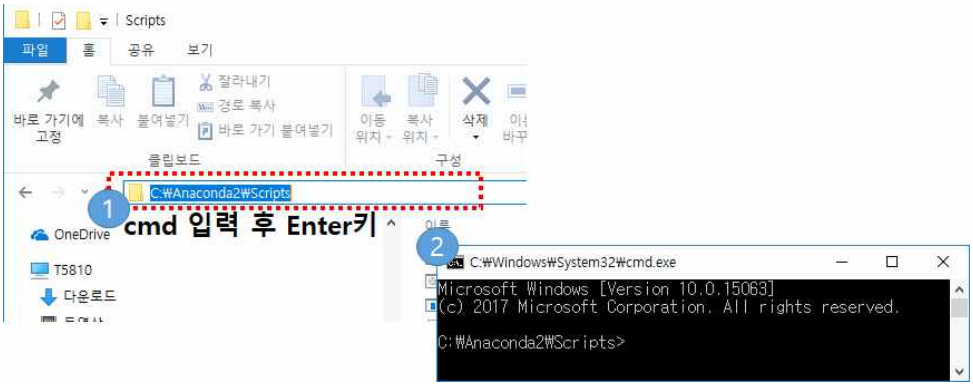
1. 요구 환경

- Python – one of the following:
  - CPython >= 2.6 or >= 3.3
  - PyPy >= 4.0
  - IronPython 2.7
- MySQL Server – one of the following:
  - MySQL >= 4.1 (tested with only 5.5~)
  - MariaDB >= 5.1

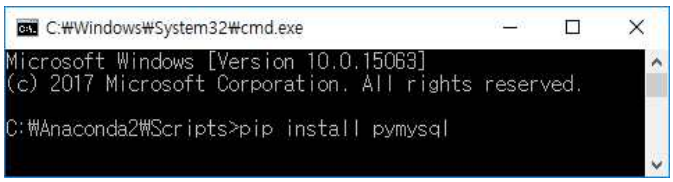
2. MySQL 연동 패키지인 PyMySQL 설치하기

※ PyMySQL 소스 공식 다운로드 경로 ⇒ https://github.com/PyMySQL/PyMySQL

1) "파일 탐색기"를 실행시킨 후 python 설치 폴더의 "script"폴더로 이동 → ①에 cmd 입력 후 Enter키를 치면 ②번과 같이 명령 프롬프트 창이 뜬다.



2) pip install pymysql 명령을 입력하여 설치한다.

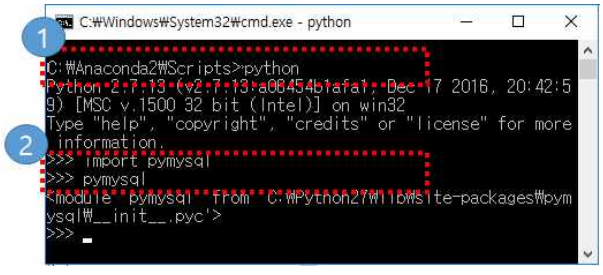


3) 이제 pymysql이 정상적으로 설치되었나를 확인해 보자.

① python을 입력하여 파이썬 실행 창으로 이동하자.

② 파이썬실행프롬프트(>>>) 상태에서 import pymysql을 입력한다.

③ pymysql을 입력하여 패키지 실행경로를 확인한다.

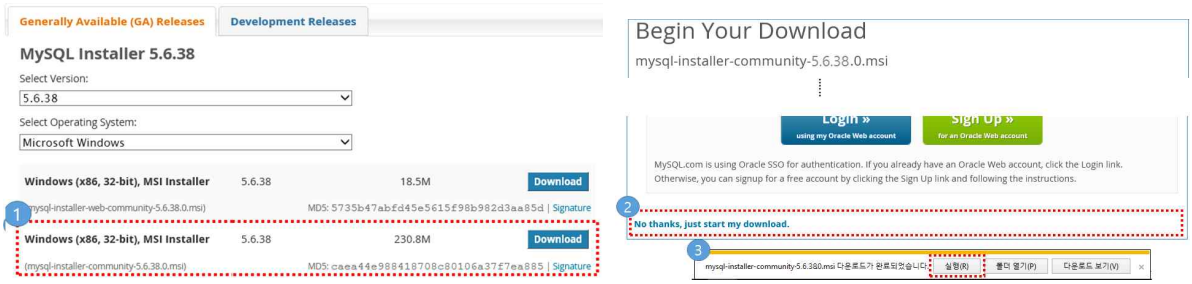


※ Documentation is available online: http://pymysql.readthedocs.io/

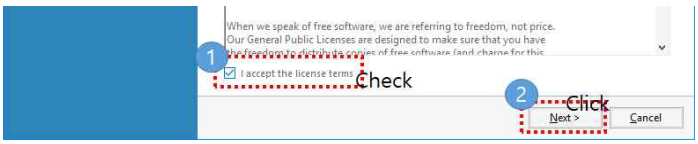
Ⅱ. windows에 MySQL 설치 및 연동 테스트하기

1. mysql 설치하기

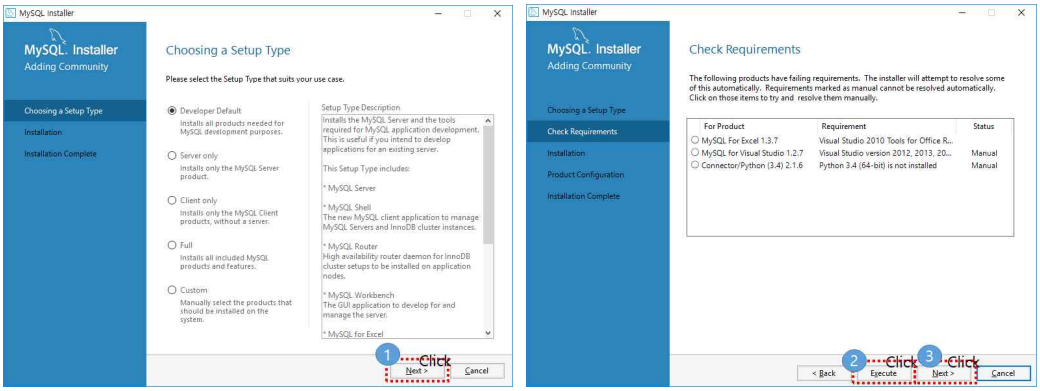
1) https://dev.mysql.com/downloads/windows/installer/5.6.html 사이트를 방문 후 하단으로 스크롤하여 원하는 버전을 [download] 버튼을 클릭하여 다운로드받은 후 실행시킨다.



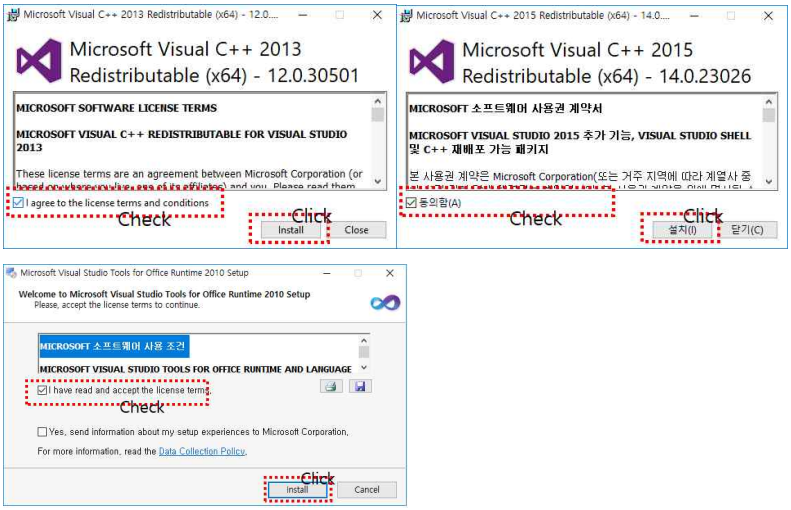
2) 설치 도중 “이 앱이 디바이스를 변경하도록 허용하시겠어요?”를 묻는 메시지가 나오면 [예] 버튼을 클릭하고, 라이센트 수용항목에 Check한 후 [Next] 버튼을 클릭한다.



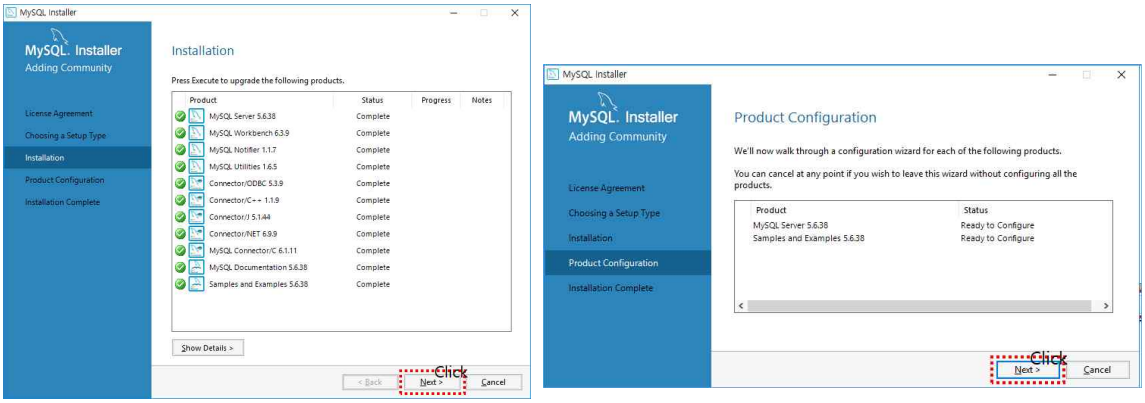
3) Setup Type은 “Full”을 선택하고, [Next] - [Execute] - [Next] 버튼을 클릭한다.



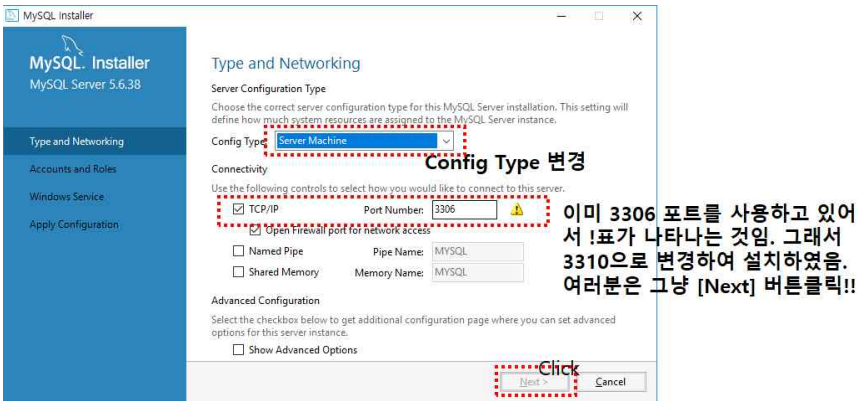
※ 아래와 같은 창이 뜨면, 수용(동의)항목에 Check한 후 [Install/설치] 버튼을 클릭한다.
(아래 항목들을 설치하는 도중 나오는 메시지 창에서는 [무시] 버튼을 클릭한다.)



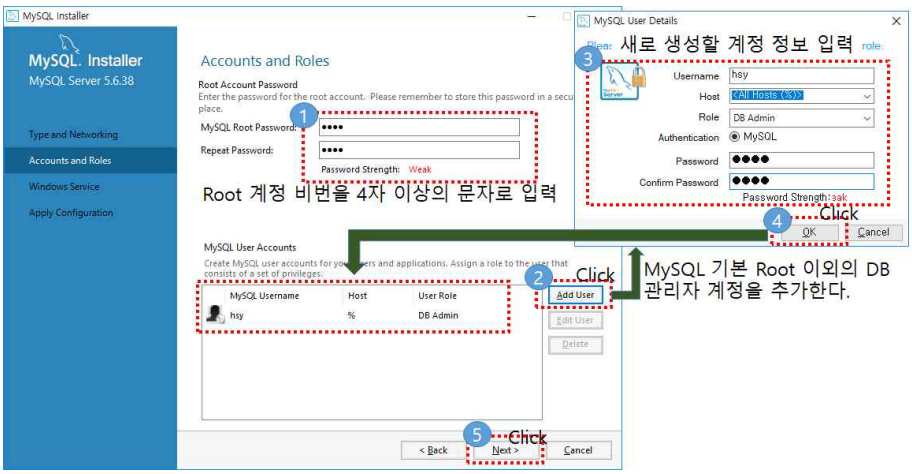
4) [Next] - [Next] 버튼을 클릭한다.



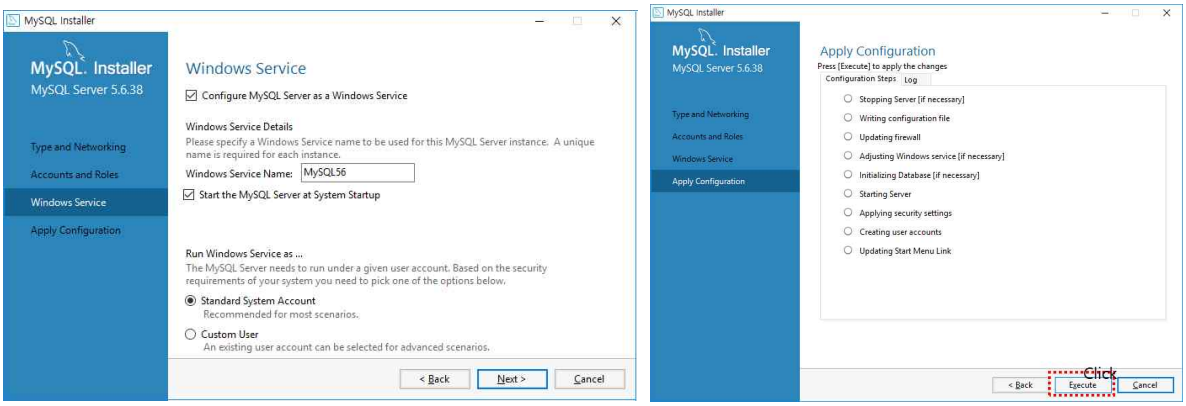
5) Config Type 변경 후 [Next] 버튼을 클릭한다.



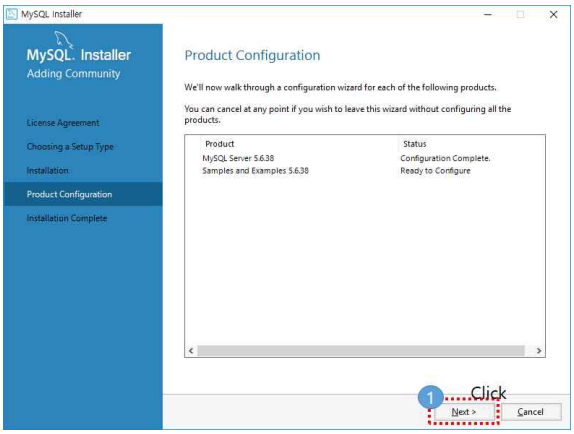
6) Root 비밀번호를 설정하고 필요한 관리자 및 사용자 계정을 추가한 후 [Next] 버튼을 클릭한다.

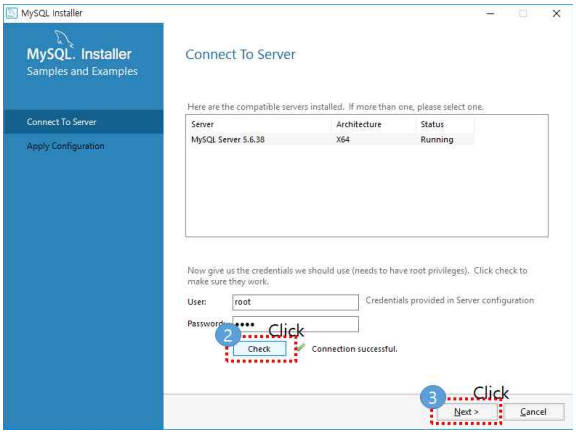


7) [Next] - [Execute] - [Finish] 버튼을 클릭한다.

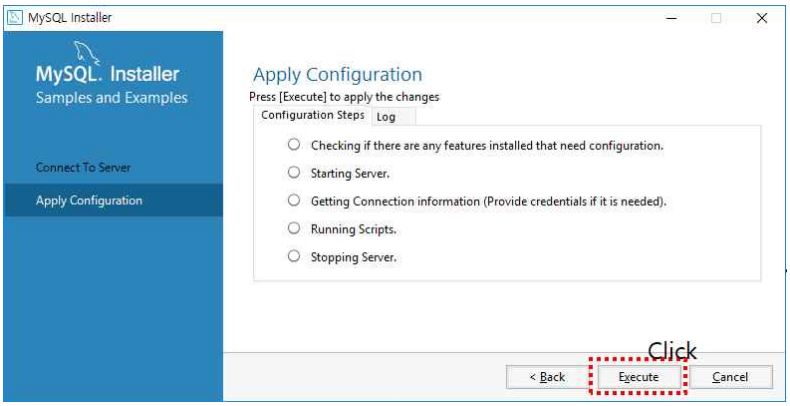


8) 아래 화면에서 [Next] - DB 연결을 [Check] - [Next] 버튼을 클릭한다.

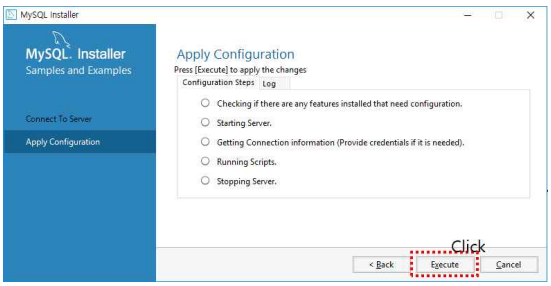


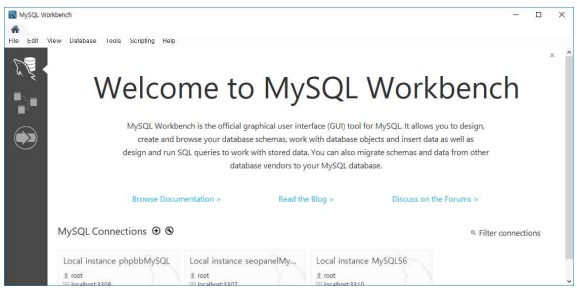


9) 다시 돌아온 Apply Configuration 화면에서 [Execute] - [Finish] 버튼을 클릭한다.



10) 다시 돌아온 Product Configuration 화면에서 [Next] - [Finish] 버튼을 클릭하면, 아래와 같은 MySQL Workbench화면이 뜬다.





2. mysql 연동하기

1) PyMySQL을 이용하여 MySQL 접속 방법을 알아보자

- MySQL 접속 객체 생성 방법

```
pymysql.connect(host='Mysql서버주소', port=포트번호, user='접속계정',
                password='root계정비밀번호',  db='사용할 데이터베이스명',
                charset='문자셋')
```

※ 종종 클라이언트의 charset이 제대로 설정되지 않으면 한글이 깨지는 경우가 발생할 수 있으니, charset을 utf8로 지정하는 편이 좋다.

[mysql의 기본 포트인 3306이고 문자셋이 utf8로 설치된 경우]

```
pymysql.connect(host='localhost', user='root', password='root계정비밀번호', db='world',
                charset='utf8')
```

[mysql의 기본 포트인 3310이고 문자셋이 utf8로 설치된 경우]

```
pymysql.connect(host='localhost', port=3310, user='root', password='root계정비밀번호',
```

```
                    db='world', charset='utf8')
```

- 데이터를 가져오는 방법

[데이터를 튜플 형태로 반환받는 방법]

```
# Connection 으로부터 mysql 기본 커서 생성(SQL 실행 객체 생성)
curs = conn.cursor()

# SQL문 실행
sql = "select id, name, countrycode from city" # SQL 문장
curs.execute(sql)

# data Fetch
rows = curs.fetchall()
for row in rows:
    print(row)  # 튜플 형식으로 출력
    print(row[0], row[1], row[2])  # Array 형식으로 출력
```

```
[실행결과]
(4079, u'Rafah', u'PSE')
(4079, u'Rafah', u'PSE')
```

[데이터를 딕셔너리 형태로 반환받는 방법 : 데이터프레임 및 튜플 형식 출력이 모두 가능]

```
# Connection 으로부터 Dictionary Cursor 생성(SQL 실행 객체 생성)
curs = conn.cursor(pymysql.cursors.DictCursor)

# SQL문 실행
sql = "select id, name, countrycode from city" # SQL 문장
curs.execute(sql)

# data Fetch
rows = curs.fetchall()
for row in rows:
    print(row)  # 튜플 형식으로 출력
    print(row[0], row[1], row[2])  # Array 형식으로 출력  : Error
    print(row['id'], row['name'], row['countrycode'])  # 데이터프레임 형식으로 출력
```

```
[실행 결과] : 실제 실행 결과는 오류가 나서 첫줄말 나오고 오류메시지를 출력함
{u'id': 4079, u'countrycode': u'PSE', u'name': u'Rafah'}
Error : Dictionary Cursor로 생성되는 경우 Array 형식으로의 출력은 불가능
(4079, u'Rafah', u'PSE')
```

2) 이클립스를 실행시켜 아래와 같은 코드를 입력한다.

정상적으로 설치된 경우라면, 아래와 같은 코드를 입력한 후 실행시키면 하단에 정보가 출력되는 것을 확인할 수 있다.

```
# -*- coding:utf-8 -*-
```

```
import pymysql

# MySQL Connection 연결(DB와 연결객체 생성)
conn = pymysql.connect(host='localhost', port=3310, user='root',
password='root계정비밀번호', db='world', charset='utf8')

# Connection 으로부터 Dictionary Cursor 생성(SQL 실행 객체 생성)
curs = conn.cursor(pymysql.cursors.DictCursor)

# SQL문 실행
sql = "select id, name, countrycode from city" # SQL 문장
curs.execute(sql)

# data Fetch
rows = curs.fetchall()
for row in rows:
    print(row)  # 튜플 형식으로 출력
    print(row['id'], row['name'], row['countrycode'])  # 데이터프레임 형식으로 출력

# Connection 닫기
conn.close()
```

```
[실행 결과]
{u'id': 4079, u'countrycode': u'PSE', u'name': u'Rafah'}
(4079, u'Rafah', u'PSE')
```

장고(Django) pymysql 이용해서 MySQL 연동하기

장고는 별다른 설정이 없을 시 기본적으로 sqlite를 사용한다. 장고에서 MySQL을 사용하기 위해서는 pymysql을 설치한다.

pip install pymysql

설치 후 장고의 settings.py를 열고 DATABASE관련 부분을 아래와 같이 바꿔준다.

```
78 DATABASES = {
79     'default': {
80         'ENGINE': 'django.db.backends.mysql',
81         'NAME': 'DB명 ',
82         'USER': 'root',
83         'PASSWORD': '',
84         'HOST': '127.0.0.1',
85         'PORT': '3306',
86     }
87 }
```

그 다음 settings.py의 상단 부분에 아래와 같은 코드를 넣어준다.

import pymysql

pymysql.install_as_MySQLdb()

수정한 이후 manage.py가 있는 폴더로 이동한 후

python manage.py makemigrations

python manage.py migrate

를 차례로 실행시킨다.

실행 후 디비에 장고 관련 테이블들이 생성되었다면 정상적으로 연동이 된 것이다.