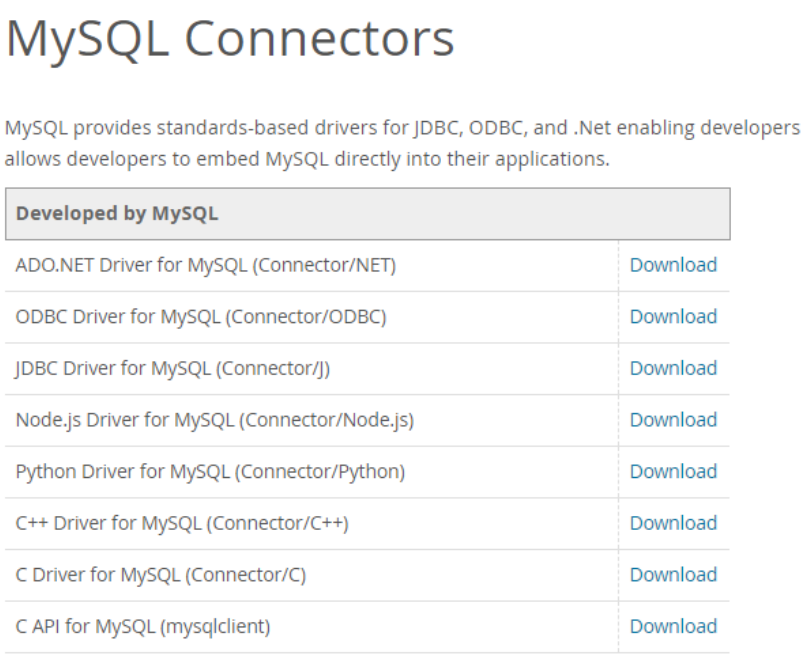
ILE3-034 psycopg

RDBMS인 PostgreSQL을 Python으로 사용

DB Connector

각 언어에서 RDBMS를 조작하기 위해서 사용하는 Library를 통칭



MySQL Connectors

MySQL provides standards-based drivers for JDBC, ODBC, and .Net enabling developers allows developers to embed MySQL directly into their applications.

Developed by MySQL	
ADO.NET Driver for MySQL (Connector/NET)	Download
ODBC Driver for MySQL (Connector/ODBC)	Download
JDBC Driver for MySQL (Connector/J)	Download
Node.js Driver for MySQL (Connector/Node.js)	Download
Python Driver for MySQL (Connector/Python)	Download
C++ Driver for MySQL (Connector/C++)	Download
C Driver for MySQL (Connector/C)	Download
C API for MySQL (mysqlclient)	Download

psycopg

Python에서 PostgreSQL에 연결하기 위한 DB Connector

설치

```
$ pip install psycopg
```

사용 방법

모듈 Import

```
import psycopg
```

Connection

생성

`psycopy2.connect()` 함수를 이용하여 PostgreSQL Instance와 연결

```
conn = psycopg.connect(host="localhost",
                       port="5432"
                       user="postgres",
```

```
                      password="abc123",
                      database="testpython")
```

종료

Connection instance의 `close()` 함수를 이용하여 연결 종료

```
conn.close()
```

Query

Cursor 생성

Connection instance의 `cursor()` 함수를 이용하여 생성
해당 cursor를 이용하여 Query의 전달 및 결과 읽기가 가능

```
cur = conn.cursor()
```

Query 전송

Cursor instance의 `execute()` 함수를 이용하여 Query 구문 전달

```
cur.execute("SELECT version()")
```

> 해당 함수를 호출 시 Query는 수행되었으나, 결과는 받아오지 않은 상태
> INSERT / UPDATE / DELETE와 같이 결과가 중요하지 않은 함수는 해당 함수 호출 시 해당 내용이 DB에 반영되어 있음

1개의 Row 읽기

Cursor instance의 `fetchone()` 함수를 이용하여 1개의 Row 읽기

```
>>> cur.execute("SELECT version()")
>>> cur.fetchone()
('PostgreSQL 12.7, compiled by Visual C++ build 1914, 64-bit',)
>>> cur.fetchone()
None

```

> 읽어온 Row는 튜플 형대로 반환

> 데이터의 마지막까지 도달 한 경우 `None` 반환

모든 Row 읽기

Cursor instance의 `fetchall()` 함수를 이용하여 해당 Query의 모든 Row 읽기

```
```python
>>> cur.execute("SELECT _id FROM testTable")
>>> cur.fetchall()
[(557631909,), (557631910,), (557631919,), (557631929,)]

```

> 읽어온 List는

Commit / Rollback

`INSERT` / `UPDATE` / `DELETE` 등의 구문으로 인하여 데이터가 수정되었을 때 해당 Transaction을 반영 / 복원하는 기능
Cursor instance의 `commit()`, `rollback()` 함수를 이용
```

```
cursor.commit()
cursor.rollback()
```

사용 예제

```
import psycopg

conn = psycopg.connect(host="localhost",
                       port="5432"
                       user="postgres",
                       password="abc123",
                       database="testpython",
                       )

cur = conn.cursor()
cur.execute("""CREATE TABLE FRUITS (
               id          INT ,
               fruit_name  TEXT,
               color       TEXT,
               price       REAL
            )""")
conn.commit()

conn.close()
```

과제

Geometry를 사용하지 않는 방식의 POI Table에 아래와 같이 임의의 데이터 1,000건을 기록

1. 테이블이 있는 경우, Table 내의 데이터를 제거
2. 좌표의 범위는 127.0, 37.0 ~ 128.0, 38.0
3. 명칭은 10글자의 임의의 문자열
4. 종별은 1~10 중 하나를 임의로 설정

데이터 기록 후 아래의 조건에 해당하는 데이터를 출력

1. 좌표의 범위가 127.4, 37.4 ~ 127.6, 37.6 내에 존재하는 항목 출력

임의의 문자열 만들기

```
import string
import random

result = ''
for i in range(10):
  result += random.choice(string.ascii_uppercase)
```