

# Database Project 2

Implementing a Simple Database Application

# Overview

- Pymysql을 이용해 MariaDB와 연동되는 응용 시스템 구현
- 공연 티켓 예매 시스템
  - 공연장, 공연, 관객 정보 삽입/삭제/출력
  - 공연을 공연장에 배정
  - 공연 예매
  - 좌석 별 예매 상황 출력
- Due : 6/22(월) 23:59

# Pymysql Tutorial

# Connect to Server DB

- 접속 정보
  - 서버 주소
    - astronaut.snu.ac.kr (포트: 3306)
  - DB 이름
    - 20DB\_[학번]
    - EX) 20DB\_2020\_12345
  - 유저 이름, 비밀번호
    - 20DB\_[학번]
    - EX) 20DB\_2020\_12345

# Python - MySQL 연동 예시 코드1

```
import pymysql
```

← pymysql 라이브러리 불러오기

```
# Connect to the database
```

```
connection = pymysql.connect(
```

```
    host='astronaut.snu.ac.kr',
```

← DB가 위치한 서버의 주소

```
    port=3306,
```

← 포트번호

```
    user='20DB_[학번]',
```

← ID

```
    password='20DB_[학번]',
```

← Password

```
    db='20DB_[학번]',
```

← 사용할 schema 이름

```
    charset='utf8',
```

← Character set 종류

```
    cursorclass=pymysql.cursors.DictCursor)
```

← 결과를 저장할 방식 지정 (\*)

```
try:
```

← 에러 발생 시에도 프로그램 종료하지 않기 위해...

```
    with connection.cursor() as cursor:
```

← connection으로부터 cursor 생성 (\*\*)

```
        # Select records from Student table
```

```
        sql = " "
```

← 사용할 SQL 문

```
        cursor.execute(sql)
```

← SQL 문 실행 (\*\*\*)

```
        result = cursor.fetchall()
```

← cursor에서 모든 결과를 불러와 result에 저장 (\*\*\*\*)

```
        print(result)
```

```
finally:
```

← 에러가 발생하든 아니든 이미 연결한 connection은 닫아야 하므로...

```
    connection.close()
```

# Python - MySQL 연동 예시 코드: 설명

- (\*) cursorclass에는 쿼리 실행 결과를 어떤 형태로 표현할 것인지를 지정합니다.
  - DictCursor의 경우 결과가 dictionary의 list로 표현됨을 의미합니다. 즉, student 테이블 내에 A, B, C 필드가 있을 경우, SELECT A, B FROM student의 실행 결과는 [{'A': ..., 'B': ...}, ..., {'A': ..., 'B': ...}] 와 같은 형태가 됩니다.
  - DictCursor 대신 Cursor를 사용할 경우 dictionary의 list가 아닌 tuple의 list로 표현됩니다. 즉, SELECT A, B FROM student의 실행 결과는 [(..., ...), (..., ...)] 와 같은 형태가 됩니다.
- (\*\*) Cursor란 데이터베이스와 상호작용할 때에 사용하는 오브젝트를 말합니다. 대부분의 database connector는 cursor를 통해 데이터베이스에 statement를 전송합니다.
  - 참고: <https://stackoverflow.com/questions/3861558/what-are-the-benefits-of-using-database-cursor>
- (\*\*\*) 생성한 cursor에 statement를 전송해 실행시키면, cursor에는 실행 결과가 저장됩니다.
- (\*\*\*\*) cursor에 저장된 실행 결과를 불러오는(fetch) 부분입니다.
  - fetchall()외에 fetchone, fetchmany 등을 사용할 수 있습니다.
  - fetchone(): 결과 하나를 불러 옴. 연속해서 호출할 경우 계속해서 다음 결과를 리턴.
  - fetchmany(k): 결과 k개를 불러 옴. 연속해서 호출할 경우 계속해서 다음 결과를 리턴.
  - fetchone과 fetchmany의 경우 연속해서 호출할 때마다 계속 다음 결과를 리턴하므로, 결과의 개수가 너무 많아 fetchall을 사용하기 어려운 경우 사용될 수 있습니다.

# Python - MySQL 연동 예시 코드2

```
import pymysql
```

```
# Connect to the database
```

```
class Data(object):
```

```
    def __init__(self, host='s.snu.ac.kr', port=3306, student_id='DB2018_20575'):
```

```
        print("Connect DB...", end='')
```

```
        self.db = pymysql.connect(
```

```
            host=host, port=3306,
```

```
            user=student_id,
```

```
            passwd=student_id,
```

```
            db=student_id,
```

```
            charset='utf8'
```

```
        )
```

```
        self.cursor = self.db.cursor()
```

```
        print("Done!")
```

```
    def __del__(self):
```

```
        print("Close database connection..")
```

```
        self.db.close()
```

```
        self.cursor.close()
```

```
    def _prepare_schema(self):
```

```
        truncate_sql = """
```

```
            TRUNCATE TABLE customer;
```

```
        """
```

```
        sql = """
```

```
            CREATE TABLE customer (
```

```
                ID INT PRIMARY KEY,
```

```
                name VARCHAR(255) NOT NULL,
```

```
                sex VARCHAR(255) NOT NULL,
```

```
                age INT NOT NULL);
```

```
        """
```

```
        self.cursor.execute(truncate_sql)
```

```
        self.cursor.execute(sql)
```

```
        self.db.commit()
```

```
        :
```

# 기타 공지

- Pymysql외의 외부 라이브러리 사용 불가능
- 프로젝트 관련 문의는 6/19 금요일까지만 가능
- 프로젝트 1-3의 채점 결과 공지는 6/10 수요일까지 공개될 예정
- 프로젝트 1-3 성적 관련 문의는 6/12 금요일까지 가능



# Simple Tutorial for MySQL Workbench (*optional*)



# Welcome to MySQL Workbench

MySQL Workbench is the official graphical user interface (GUI) tool for MySQL. It allows you to design, create and browse your database schemas, work with database objects and insert data as well as design and run SQL queries to work with stored data. You can also migrate schemas and data from other database vendors to your MySQL database.

[Browse Documentation >](#)[Read the Blog >](#)[Discuss on the Forums >](#)

MySQL Connections ⊕ ⊖

🔍 Filter connections

Setup New Connection

Connection Name: ABC

Type a name for the connection

Connection Method: Standard (TCP/IP)

Method to use to connect to the RDBMS

Parameters

SSL

Advanced

Hostname: s.snu.ac.kr

Port: 3306

Name or IP address of the server host - and TCP/IP port.

Username: ADB2018\_20555

Name of the user to connect with.

Password: Store in Vault ... Clear

Default Schema: ADB2018\_20555

Configure Server Management...


Test Connection

Cancel

OK

Connect to MySQL Server

Please enter password for the following service:



Service: Mysql@s.snu.ac.kr:3306

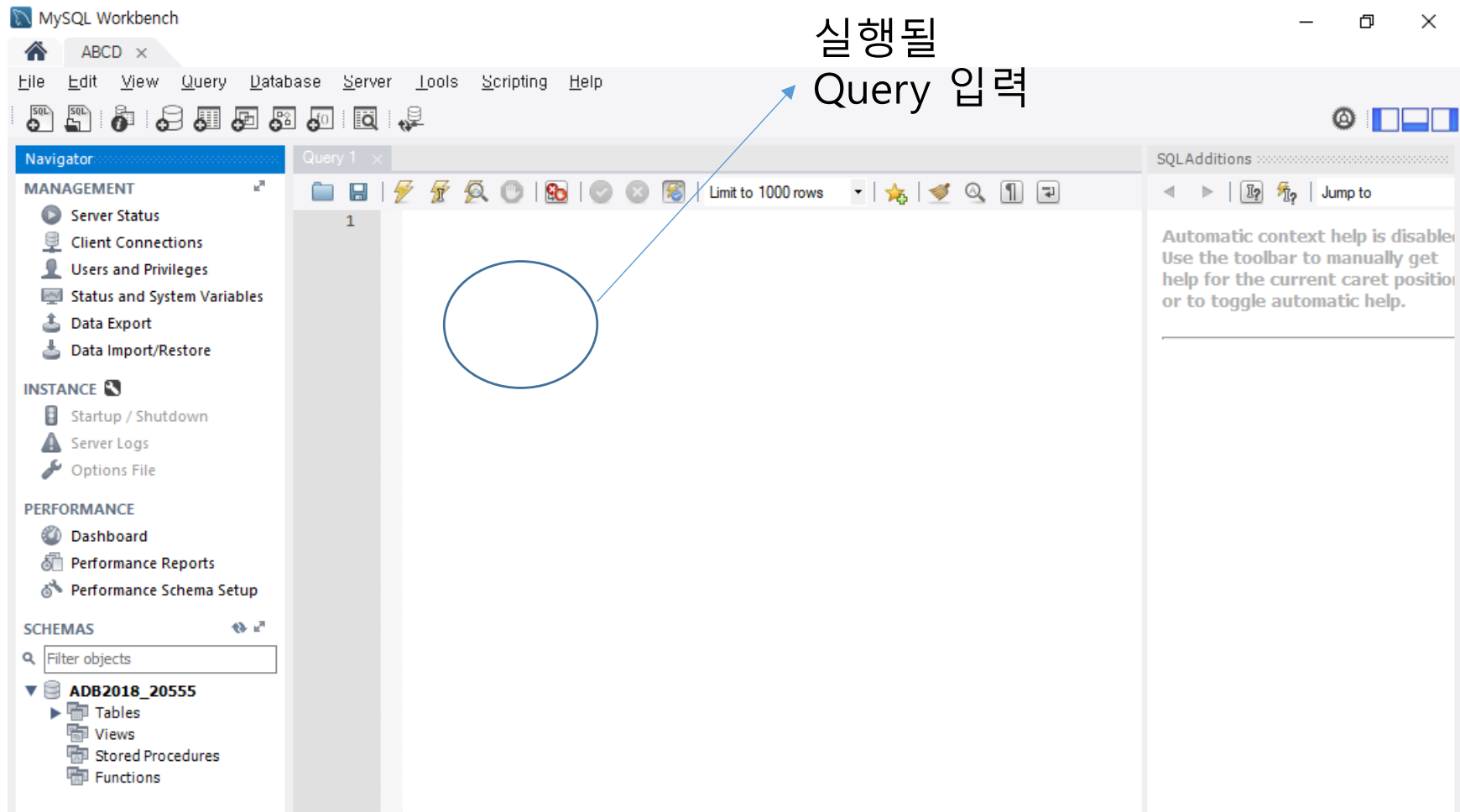
User: ADB2018\_20555

Password:

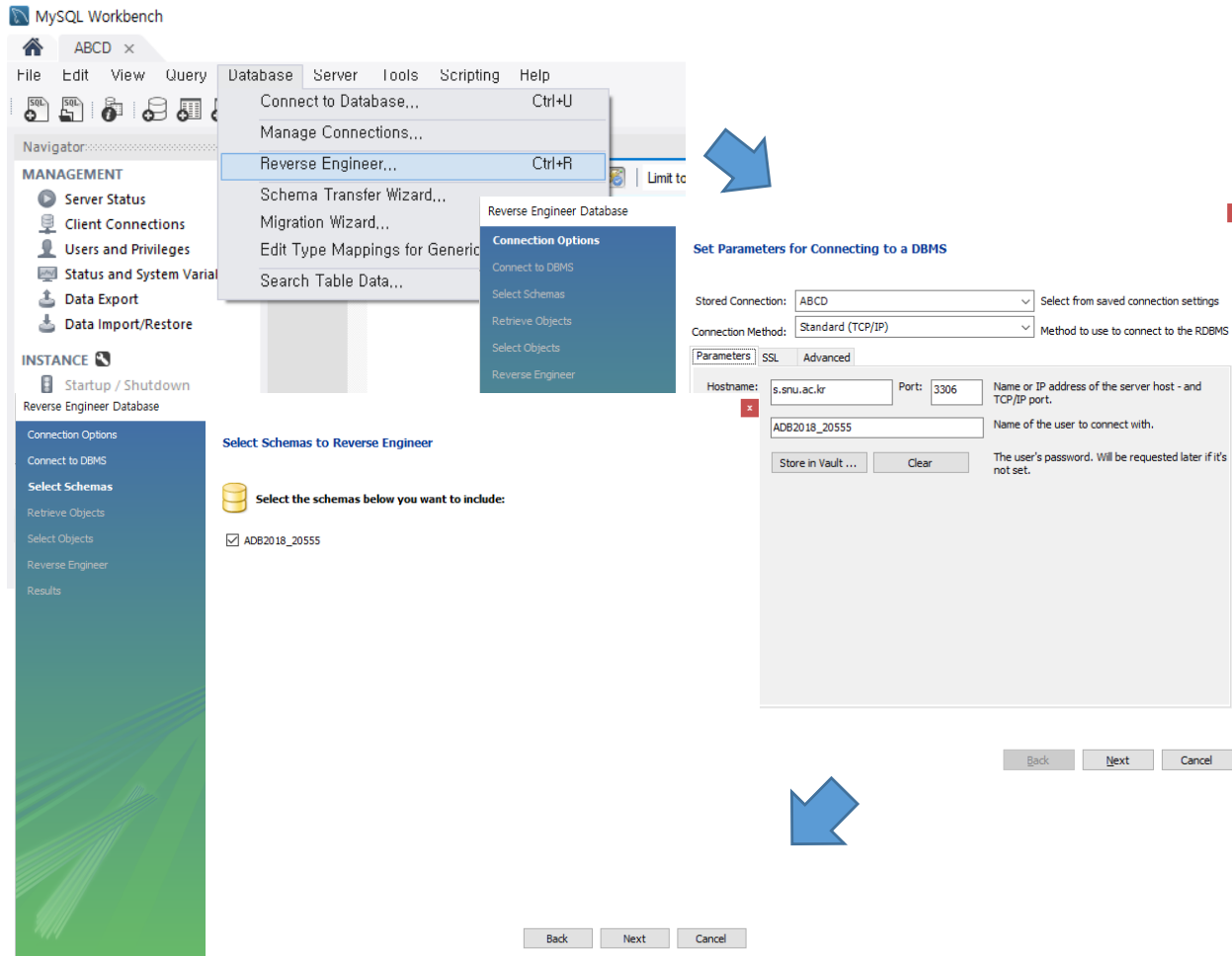
☐ Save password in vault

OK

Cancel



# 간단한 Schema Diagram



## 간단한 Schema Diagram (Cont.)

