# Python数据库操作 教案

周次：7

课次：2

章节内容：MySQL数据库连接

课时：2

## 单元教学目标

- 知识目标：能够正确安装MySQL驱动并理解连接字符串的构成要素   
- 技能目标：能够使用Python代码成功连接MySQL数据库，并能够处理常见的连接错误   
- 素养目标：能够遵守数据库安全规范，正确配置连接参数，避免泄露敏感信息

## 教学重点

• 掌握MySQL数据库的基本安装与配置方法   
• 理解Python连接MySQL的常用库（如mysql-connector-python或pymysql）   
• 学会通过参数建立安全的数据库连接（主机名、端口、用户名、密码、数据库名）

## 教学难点

• 学生可能难以掌握如何正确配置MySQL数据库连接参数（如host、user、password、database等），尤其是对不同数据库驱动（如MySQLdb、mysql-connector-python）的语法差异存在混淆。   
• 学生可能在实际操作中遇到连接失败问题，需理解异常处理机制（如try-except块）并掌握如何调试连接错误（如检查网络、权限、数据库是否存在）。

## 教学活动

新课导入【5分钟】   
教师通过故事导入：讲述一个电商公司需要记录用户订单数据，但因缺乏数据库连接导致效率低下。提问学生："如果要让系统自动记录订单，需要哪些技术？" 引导学生思考数据库连接的重要性，用案例分析法激发兴趣。   
  
讲授新课【20分钟】   
1. \*\*讲授法\*\*（5分钟）：讲解MySQL安装步骤，重点说明MySQL服务启动与关闭命令（如`service mysql start`）。   
2. \*\*案例分析法\*\*（7分钟）：展示Python代码片段，演示如何使用`mysql-connector-python`库连接数据库（示例代码：`import mysql.connector; conn = mysql.connector.connect(host='localhost', user='root', password='123456'））。   
3. \*\*讨论法\*\*（6分钟）：分组讨论"如何避免数据库连接错误"，教师引导学生总结常见错误（如密码错误、端口不对）并分享解决方案。   
4. \*\*练习法\*\*（2分钟）：让学生尝试编写简单连接代码，教师巡视指导，重点讲解异常处理（如`try-except`块）。   
  
巩固练习【15分钟】   
1. \*\*练习法\*\*（10分钟）：设计基础练习：   
 - 任务1：用Python连接本地MySQL数据库，输出"数据库连接成功"。   
 - 任务2：编写查询语句，从`employees`表中获取姓名为"John"的记录。   
2. \*\*游戏法\*\*（5分钟）：设置"数据库连接挑战赛"，学生分组竞赛，用最短时间完成任务，教师点评优胜组并讲解关键点（如参数传递、语句语法）。   
  
归纳总结【5分钟】   
教师通过提问总结：   
- 问学生："今天学习了哪些数据库连接技术？"   
- 引导学生复述Python连接MySQL的步骤，并强调异常处理的重要性。   
- 用讨论法鼓励学生分享学习收获，教师补充关键知识点（如驱动安装、连接参数）。

## 教学资源

</think>  
  
• \*\*教学设备和工具\*\*   
- 计算机（安装有Python环境）   
- 屏幕投影设备（用于展示教学内容）   
- MySQL数据库服务器（或本地安装MySQL）   
- Python开发环境（如PyCharm、VS Code、Jupyter Notebook）   
- 数据库管理工具（如MySQL Workbench）   
- 网络连接（用于远程访问MySQL服务器）   
  
• \*\*推荐的参考资料或网站\*\*   
- 《Python数据库操作》（书籍）   
- Python官方文档（https://docs.python.org/3/)   
- MySQL官方文档（https://dev.mysql.com/doc/）   
- 网易云课堂《Python数据库操作》课程   
- GitHub 上的Python数据库操作示例（如：https://github.com/PythonDB/PythonDB）   
- 知乎专栏《Python数据库实战》   
- 网易云课堂《Python数据库操作》课程（可免费观看）   
- LeetCode（用于练习SQL语句）   
- Stack Overflow（解答Python数据库相关问题）

## 教学反思

• 教学效果评估：需评估学生是否掌握MySQL连接的基本步骤（如安装库、配置参数、建立连接），以及能否独立完成简单查询操作，是否存在理解偏差或操作失误   
• 学生反馈分析：关注学生对数据库连接实际应用的反馈，如是否认可教学内容的实用性，是否对连接参数（如主机名、端口、用户名、密码）的配置存在困惑，或对异常处理机制的掌握程度   
• 教学内容深度：需判断是否覆盖了关键知识点（如连接对象的生命周期、事务处理），是否存在内容冗余或重点不突出的情况   
• 学习资源供给：评估学生是否能通过课后练习独立完成数据库操作，是否存在资源不足（如示例代码缺失、练习题难度适中）导致学习障碍   
• 技术实现难点：分析学生在安装MySQL客户端、配置环境变量等技术细节上的操作困难，是否需要增加可视化工具（如MySQL Workbench）辅助教学   
• 实操能力培养：观察学生是否能通过实际操作（如执行SQL语句、查看表结构）验证学习成果，是否存在理论与实践脱节现象   
• 异常处理教学：评估学生对连接失败、查询错误等异常情况的应对能力，是否需要补充错误码解释（如MySQL error 1045）   
• 配置环境指导：关注学生在安装依赖库（如mysql-connector-python）时的常见问题，是否需要提供详细安装步骤或依赖管理建议   
• 课堂互动效果：分析学生在分组练习中的参与度，是否存在课堂讨论不足导致的知识盲点，是否需要增加实时答疑环节   
• 课程节奏把控：评估2课时是否合理分配了理论讲解（如连接原理）与实践操作（如编写SQL语句）的时间比例，是否存在进度过快或过慢的问题

## 教学评价

• 过程性评价   
• 课堂参与度：观察学生在教师提问、小组讨论中的主动性和表达能力，记录课堂互动情况（如提问频率、发言质量）。   
• 作业完成情况：批改学生提交的代码，评估代码逻辑、语法正确性及是否按要求完成基础操作（如连接数据库、执行查询）。   
• 小组合作：通过小组任务完成情况（如分工是否明确、协作是否顺畅）评价团队协作能力。   
• 课堂表现：记录学生是否按时完成任务，是否主动解决疑难问题，是否存在懈怠现象。   
  
• 结果性评价   
• 项目成果：根据学生完成的数据库连接项目（如使用mysql-connector-python或pymysql）评分，重点考察代码完整性、功能实现及错误处理能力。   
• 代码质量：通过代码规范性（如缩进、注释、变量命名）和逻辑合理性（如是否能正确执行增删查改操作）进行评分。   
• 课堂测试：设计基础题（如连接MySQL数据库的Python代码编写）和拓展题（如处理异常或优化查询）进行测试，评估知识掌握程度。   
• 项目答辩：组织学生展示项目成果，通过答辩表现（如回答问题的准确性、逻辑性）综合评价学习效果。   
  
• 评价标准与方式   
• 课堂参与度：30%（观察记录+教师反馈）   
• 作业完成情况：40%（代码评分+逻辑分析）   
• 小组合作：20%（任务完成度+沟通记录）   
• 项目成果：10%（代码规范性+功能验证）   
  
• 评分细则   
• 课堂参与度：满分10分，扣分项包括：未主动提问、发言内容空洞、互动冷淡。   
• 作业完成情况：满分10分，扣分项包括：代码存在语法错误、逻辑不清晰、未按要求提交。   
• 项目成果：满分10分，扣分项包括：功能不完整、代码存在严重bug、文档缺失。   
  
• 备注   
• 评价周期：过程性评价贯穿课堂，结果性评价以项目成果为准。   
• 评价主体：教师、学生自评、同伴互评相结合。   
• 评分权重：过程性评价占60%，结果性评价占40%。