# Python数据库操作 教案

周次：7

课次：1

章节内容：SQLite数据库

课时：2

## 单元教学目标

- 知识目标：能够正确使用SQLite的安装和基本命令，理解其核心概念如数据库、表、字段及数据类型   
- 技能目标：能够使用Python的sqlite3模块完成数据库的基本操作，包括创建表、插入数据、查询数据并处理简单错误   
- 素养目标：能够遵循代码规范，编写清晰的数据库操作代码，并能通过文档说明基本操作流程

## 教学重点

• 使用sqlite3模块连接和关闭数据库   
• 执行SQL查询并处理结果集   
• 插入、更新、删除数据操作

## 教学难点

• 学生可能难以掌握如何正确使用SQL语句进行数据插入和查询，例如参数化查询的语法（如`?`占位符）的使用方法及与Python变量的绑定逻辑。   
• 对SQLite数据库的事务处理机制理解不足，如如何通过`BEGIN`启动事务、`COMMIT`提交更改、`ROLLBACK`回滚操作，以及异常处理（如`sqlite3.OperationalError`）的常见场景。

## 教学活动

新课导入【5分钟】   
教师用故事导入：讲述医院管理中患者信息存储的痛点，引出需要数据库系统。通过提问引导学生思考：“若用Excel表格记录患者信息，存在什么问题？” 学生分组讨论后，教师总结数据库的重要性，激发学习兴趣。   
  
讲授新课【30分钟】   
1. \*\*讲授法\*\*（5分钟）   
 - 教师讲解SQLite基本概念，强调其轻量级、跨平台特性，通过PPT展示SQLite图形化界面操作示例。   
2. \*\*案例分析法\*\*（8分钟）   
 - 教师演示创建数据库“HospitalDB”及表“patients”（字段：id, name, age, disease）。学生分组分析案例，讨论如何设计表结构，教师点评优化建议。   
3. \*\*讨论法\*\*（7分钟）   
 - 学生讨论：“若需存储学生成绩，表字段应如何设计？” 学生代表发言，教师总结表结构设计原则（如主键、数据类型）。   
4. \*\*角色扮演法\*\*（6分钟）   
 - 学生分角色模拟数据库管理员，完成“创建表”任务，教师观察并点评操作准确性。   
5. \*\*讲授法\*\*（2分钟）   
 - 教师讲解查询语句基础，通过代码示例演示“SELECT \* FROM patients”操作，学生模仿执行。   
  
巩固练习【15分钟】   
1. \*\*练习法\*\*（8分钟）   
 - 学生独立完成“创建数据库并插入患者信息”练习，教师巡视指导，重点检查语法正确性。   
2. \*\*游戏法\*\*（7分钟）   
 - 学生分组竞赛，用SQLite工具完成“查询患者年龄大于30岁的记录”任务，优胜组获得“数据库操作小达人”称号。   
3. \*\*练习法\*\*（0分钟）   
 - 学生用代码实现“查询患者姓名和年龄”功能，教师批改并反馈错误点，强调条件语句使用。   
  
归纳总结【10分钟】   
1. \*\*总结法\*\*（5分钟）   
 - 教师梳理核心知识点：SQLite安装、表创建、查询语句，强调“主键唯一性”等关键概念。   
2. \*\*头脑风暴法\*\*（5分钟）   
 - 学生自由发言：“生活中哪些场景需要数据库？” 学生举例后，教师归纳为“数据存储、数据分析、决策支持”等应用场景。

## 教学资源

• 教学设备：计算机教室（需配备安装好Python环境的电脑）、投影仪、白板（用于演示）、网络设备（确保网络稳定）   
• 工具软件：   
 - Python编程环境（需安装SQLite库，如`pip install sqlite3`）   
 - SQLite管理工具（如SQLite Browser、sqlite3命令行工具）   
 - 图形化数据库管理工具（如SQLite Manager、Db Browser Pro）   
 - 数据可视化工具（如Matplotlib/Seaborn用于数据展示）   
  
• 推荐参考资料：   
 - 《Python编程：从入门到实践》（推荐章节：第12章）   
 - SQLite官方文档（https://www.sqlite.org/docs/）   
 - GeeksforGeeks Python教程（https://www.geeksforgeeks.org/python-sqlite/)   
 - SQLite教程网站（https://www.sqlitetutorial.com/）   
 - 书籍：《Python数据库编程》（作者：David Beazley）   
 - 在线练习平台：Exercism（https://exercism.org/）   
  
• 其他资源：   
 - Python官方文档（https://docs.python.org/3/）   
 - GitHub开源项目（如sqlite3示例代码）   
 - 教育网站：Coursera/edX（搜索“Python Database Programming”课程）

## 教学反思

• 教学效果方面：需评估学生是否能独立完成基础数据库操作（如创建表、插入数据、查询数据），是否掌握SQL语法基础，以及课堂参与度是否符合预期   
• 学生反馈方面：需关注学生对SQLite操作的熟悉程度（如是否理解连接/关闭数据库的逻辑），对实际应用场景（如数据存储与检索）的掌握情况，以及对错误处理（如SQL注入风险）的认知深度   
• 改进建议：增加实践环节，如通过分组练习巩固表操作；补充常见错误案例（如语法错误、表名误写）的解析；结合真实场景（如学生成绩表的构建）提升应用意识；针对薄弱环节（如SQL注入防范）进行针对性讲解   
• 教学设计优化：调整课堂节奏，确保时间分配合理（如预留10分钟演示操作步骤，20分钟实战练习）；增加可视化工具（如使用SQLite Browser辅助演示）提升学习体验   
• 学生能力培养：补充数据库设计基础（如范式理论简要介绍），引导学生思考数据结构与数据库设计的关系；通过对比MySQL与SQLite的差异，深化对数据库系统的理解   
• 评估反馈机制：引入即时反馈工具（如在线测试平台）实时监测学生掌握情况；针对难点问题（如事务操作）设置分层练习（基础题/进阶题）确保差异化教学

## 教学评价

• 课程目标：评估学生在SQLite数据库操作中的知识掌握程度与实践能力   
• 评价维度：   
 • 过程性评价（占40%）   
 - • 课堂参与度（20%）：观察学生在课堂讨论、操作练习中的积极性与互动情况   
 - • 作业完成情况（20%）：检查学生是否独立完成SQLite基础操作（如创建表、插入数据、查询）   
 - • 小组合作表现（20%）：评估小组成员分工协作、代码质量及问题解决能力   
 • 结果性评价（占60%）   
 - • 编程任务完成度（30%）：通过代码提交评估学生是否正确实现SQLite数据库操作（如连接、查询、关闭）   
 - • 代码质量（20%）：检查代码是否规范（如注释、语法正确、逻辑清晰）   
 - • 项目展示（30%）：通过答辩或演示展示学生对SQLite数据库操作的理解与应用能力   
  
• 评价方式：   
 • 过程性：课堂观察记录、作业批改、小组讨论实录   
 • 结果性：编程任务评分（代码正确性+逻辑性）、项目展示答辩评分（内容完整性+表达清晰度）   
  
• 评分标准：   
 • 过程性评价：   
 - 课堂参与度：1-5分（5分制）   
 - 作业完成情况：1-5分（5分制）   
 - 小组合作表现：1-5分（5分制）   
 • 结果性评价：   
 - 编程任务：1-5分（5分制）   
 - 代码质量：1-5分（5分制）   
 - 项目展示：1-5分（5分制）   
  
• 附加说明：   
 - 评分总分100分，需达到70分以上方可通过；   
 - 过程性评价占40%，结果性评价占60%；   
 - 项目展示可结合课堂表现与作业成果综合评分。