Python数据库操作 教案

|  |  |  |  |
| --- | --- | --- | --- |
| **周次** | **课次** | **章节内容** | **课时** |
| 7 | 1 | SQLite数据库 | 2 |
| **单元教学目标** | | | |
| - 知识目标：能够描述SQLite的轻量级特性、嵌入式数据库模式以及基本数据类型（如整数、文本、布尔值）  - 技能目标：能够使用Python连接SQLite数据库并编写创建表的SQL语句，完成数据的插入与查询操作  - 素养目标：具备数据库操作规范意识，能够遵循安全规范进行数据操作，养成严谨的编程习惯 | | | |
| **教学重点** | | **教学难点** | |
| • 创建SQLite数据库和表的基本操作  • 使用SQL语句插入和查询数据  • 掌握使用sqlite3模块执行基础查询的技能 | | • 学生可能难以掌握如何正确使用sqlite3模块进行数据库连接与操作，例如未理解`connect()`函数的参数含义（如数据库文件路径）或`cursor()`方法的使用场景。  • 如何正确构造和执行SQL语句（如`INSERT`、`SELECT`）并处理结果集，尤其是对SQL语法（如表名、字段名、参数绑定）的掌握程度不足，容易导致语法错误或数据插入异常。 | |
| **教学活动** | | | |
| 新课导入【10分钟】  1. 教师用故事导入：讲述医院管理系统中数据存储的重要性，引出SQLite数据库的应用场景。  2. 播放短视频展示SQLite在医院管理中的实际应用案例，激发学生兴趣。  3. 提问引导：提问“如何用数据库管理医院患者信息？”引发学生思考，为后续学习铺垫。   讲授新课【120分钟】  1. \*\*安装与基本操作（20分钟）\*\*   - 教师讲解SQLite安装步骤，分步演示命令行操作（`sqlite3`）。   - 学生分组练习：使用`CREATE TABLE`创建示例表，教师巡视指导。   - 案例分析：分析医院患者信息表的结构，讨论字段类型选择。   2. \*\*数据查询与操作（30分钟）\*\*   - 讲解`SELECT`语句基础，演示查询患者信息的案例。   - 学生动手操作：使用`WHERE`条件筛选数据，教师点评错误操作。   - 小组竞赛：限时完成“查询患者年龄大于30岁的记录”任务，增强实践能力。   3. \*\*数据插入与更新（20分钟）\*\*   - 教师演示`INSERT INTO`语句，讲解`UPDATE`修改数据的语法。   - 学生分组实践：插入新患者信息并修改年龄，教师实时反馈问题。   - 角色扮演：模拟医生录入患者信息，巩固操作流程。   4. \*\*数据删除与事务（20分钟）\*\*   - 讲解`DELETE`语句使用场景，演示事务处理（`BEGIN TRANSACTION`）。   - 学生分组练习：删除错误记录并提交事务，教师指导常见错误。   - 教师总结：强调事务在数据完整性中的重要性，引发学生讨论。   巩固练习【30分钟】  1. \*\*基础练习（15分钟）\*\*   - 学生独立完成“创建患者表并插入3条数据”，教师巡视检查。   - 限时10分钟完成，教师提供错误提示，引导学生修正。   2. \*\*进阶练习（15分钟）\*\*   - 分组任务：用`UPDATE`修改数据并使用`SELECT`验证，教师分组指导。   - 学生展示成果，教师点评操作规范性，强调逻辑关系。   归纳总结【10分钟】  1. 教师总结关键点：SQLite安装步骤、基本操作命令、事务处理。  2. 学生复述课堂内容，教师补充常见误区（如字段类型错误）。  3. 提问互动：学生回答“如何用SQLite管理医院数据？”强化知识留存。 | | | |
| **教学资源** | | | |
| </think>  • \*\*教学设备和工具\*\*  - 计算机（带Python环境）  - 屏幕（用于展示教学内容）  - 本地SQLite数据库（如使用`sqlite3`模块）  - 教学用的SQLite数据库文件（如`example.db`）  - 用于演示的SQLite命令行工具（如`sqlite3`）  - 白板或投影仪（用于展示代码和数据库结构）  - 互联网连接（用于推荐参考资料和网站）   • \*\*推荐的参考资料或网站\*\*  - 《Python编程：从入门到实践》——适合初学者  - Python官方文档：https://docs.python.org/3/  - SQLite官方文档：https://www.sqlite.org/docs/  - GitHub上的SQLite教程：https://github.com/SQLZoo/SQLZoo  - 书籍推荐：《Python数据库开发》——适合进阶学习  - 网站推荐：Stack Overflow（解决编程问题）  - 教育网站：Coursera、edX（提供Python数据库课程）  - 交互式学习平台：Python for Data Analysis（提供数据库操作练习） | | | |
| **教学反思** | | | |
| • 教学效果：学生是否能够正确安装和使用SQLite数据库，是否掌握基本操作如创建表、插入数据、查询数据？是否存在学生对SQL语法或数据库操作的混淆？  • 学生反馈：学生对SQLite安装流程是否感到困惑？是否在数据插入/查询过程中遇到技术障碍？是否对实际应用场景（如数据可视化）感兴趣？  • 教学难点：是否因时间限制导致部分知识点（如复杂查询）讲解不够深入？学生是否因缺乏实践机会而难以理解抽象概念？  • 课堂互动：学生是否积极参与讨论或实践？是否出现分组合作中角色分配不均的问题？  • 技术实现：是否因教学资源（如安装工具、示例代码）不足导致学生操作困难？是否需要提供更详细的调试指导？  • 教学节奏：是否因内容密度导致学生疲劳？是否需要分阶段拆分内容（如先学安装，再学查询）？  • 实践应用：学生是否能将SQLite知识应用于实际项目（如小型数据管理应用）？是否缺乏项目驱动的练习环节？  • 教学资源：是否因示例代码不够贴近实际场景（如复杂查询、异常处理）而影响学习兴趣？是否需要补充案例库或可视化工具辅助教学？  • 学习动机：是否因教学方法（如讲授为主）导致学生兴趣下降？是否需要引入案例分析、竞赛或小组挑战提升参与感？  • 教学评估：是否通过练习题或项目作业有效检验学生掌握程度？是否存在因评估方式单一导致反馈不足的问题？ | | | |
| **教学评价** | | | |
| </think>  • \*\*过程性评价\*\*：   - 课堂参与度：观察学生在课堂中的发言、提问、合作情况，记录其参与积极性。   - 任务完成情况：检查学生是否按要求完成数据库操作任务，如创建表、插入数据、查询数据等。   - 课堂小结与反思：学生是否能总结本节课内容，提出疑问或改进意见。   • \*\*结果性评价\*\*：   - 作业完成情况：检查学生提交的数据库操作作业是否符合要求，如代码是否正确、是否能正常运行。   - 项目展示：学生是否能独立完成数据库操作项目，并进行展示与讲解。   - 试卷或测试：设计简短的测试题，检验学生对SQLite基本操作（如创建表、插入数据、查询数据）的理解与掌握程度。   • \*\*评价方式\*\*：   - 课堂观察记录表（教师记录学生表现）   - 作业提交与代码检查   - 项目展示与答辩   - 试卷或小测验（如选择题、填空题、简答题）   • \*\*评价标准\*\*：   - 课堂参与度：40%（如积极发言、合作良好）   - 任务完成质量：40%（代码正确性、功能完整性）   - 项目展示表现：20%（表达清晰、逻辑合理）   • \*\*评价工具\*\*：   - 课堂观察表   - 作业评分表   - 项目展示评分表   - 试卷评分表   • \*\*评价周期\*\*：   - 课前：预习情况评估   - 课中：课堂表现与任务完成情况   - 课后：作业与项目展示   • \*\*评价反馈\*\*：   - 评价结果记录与反馈给学生   - 评价建议与改进措施（如针对薄弱环节进行辅导） | | | |