SQL 基础教程

一、SQL 基础核心概念

1. 数据库与表结构

数据库是数据存储容器，表由行（记录）和列（字段）组成。例如：

```
CREATE TABLE Persons (
    Id_P INT PRIMARY KEY,
    LastName VARCHAR(255) NOT NULL,
    FirstName VARCHAR(255),
    Address VARCHAR(255),
    City VARCHAR(255)
);
```

数据类型：INT（整数）、VARCHAR(255)（可变字符串）、DATE（日期）等 1。

2. SQL 分类与作用

DDL(数据定义语言):创建 / 修改 / 删除数据库对象(如 CREATE TABLE、ALTER TABLE)。

DML（数据操作语言）：增删改数据（如 INSERT、UPDATE、DELETE）。

DQL（数据查询语言）：检索数据（如 SELECT、JOIN）。

DCL（数据控制语言）：权限管理（如 GRANT、REVOKE）。

TLC（事务控制语言）：管理事务（如 COMMIT、ROLLBACK）7。

二、基础操作实战

1. 数据定义（DDL）

创建数据库：

```
CREATE DATABASE IF NOT EXISTS mydb;
USE mydb; -- 切换数据库
```

修改表结构：

```
ALTER TABLE Persons ADD COLUMN Age INT; -- 添加字段
ALTER TABLE Persons DROP COLUMN Age; -- 删除字段
```

删除表：

```
DROP TABLE IF EXISTS Persons;
```

2. 数据操作（DML）

插入数据：

```
INSERT INTO Persons (Id_P, LastName, FirstName) VALUES (1, 'Gates', 'Bill');
INSERT INTO Persons VALUES (2, 'Jobs', 'Steve', 'Cupertino', 'CA'); -- 全字段插入
```

更新数据：

```
UPDATE Persons SET City = 'Seattle' WHERE LastName = 'Gates';
```

删除数据：

```
DELETE FROM Persons WHERE LastName = 'Jobs';
```

3. 数据查询（DQL）

基础查询：

```
SELECT LastName, FirstName FROM Persons WHERE City = 'Beijing';
SELECT * FROM Persons ORDER BY LastName DESC; -- 按姓氏降序排列
```

去重与限制结果：

```
SELECT DISTINCT City FROM Persons; -- 去重
SELECT * FROM Persons LIMIT 5; -- 取前 5 条记录
```

聚合函数：

```
SELECT COUNT(*) AS TotalRows, AVG(Age) AS AvgAge FROM Persons;
```

三、多表关联与高级查询

1. 表连接（JOIN）

内连接：仅返回匹配行：

```
SELECT Orders.OrderID, Customers.CustomerName
FROM Orders
INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
```

左连接：返回左表所有记录：

```
SELECT Employees.EmployeeID, Orders.OrderID
FROM Employees
LEFT JOIN Orders ON Employees.EmployeeID = Orders.EmployeeID;
```

全连接：返回所有表记录：

```
SELECT * FROM TableA
FULL OUTER JOIN TableB ON TableA.ID = TableB.ID;
```

2. **子查询**

嵌套查询示例：

```
SELECT ProductName
FROM Products
WHERE CategoryID IN (SELECT CategoryID FROM Categories WHERE CategoryName
= 'Electronics');
```

3. **事务处理**

确保数据一致性：

```
START TRANSACTION;
UPDATE Accounts SET Balance = Balance - 100 WHERE AccountID = 1;
UPDATE Accounts SET Balance = Balance + 100 WHERE AccountID = 2;
COMMIT; -- 提交事务（或 ROLLBACK 回滚）
```

四、数据库方言差异与应对

不同数据库（如 MySQL、PostgreSQL、SQL Server）在语法和功能上存在差异：

分页查询：

MySQL/PostgreSQL：`LIMIT 10 OFFSET 0`

SQL Server：`OFFSET 0 ROWS FETCH NEXT 10 ROWS ONLY`

Oracle：`ROWNUM <= 10`4。

函数差异：

MySQL：`DATE_FORMAT()` 格式化日期

PostgreSQL：`TO_CHAR()` 实现相同功能 4。

解决方案：

使用 ORM 框架（如 Hibernate）自动适配方言。

编写多版本 SQL，根据数据库类型动态选择。

参考官方文档（如 MySQL 官方文档）46。

五、实战资源与学习路径

1. **示例数据库**

Sakila：MySQL 官方提供的电影租赁系统数据库，含复杂表结构，可用于练习多表查询 5。

Northwind：经典电商数据库，适合学习订单处理与报表生成。

下载地址：MySQL 示例数据库 56。

2. **学习资源推荐**

书籍：

《SQL 基础教程（视频教学版）》：含 300 + 案例与视频讲解，覆盖 MySQL 环境搭建与高级操作。

《SQL 必知必会》：适合快速入门，侧重实战技巧。

3. **练习方法**

模拟业务场景：设计班级管理系统，实现学生信息查询、缴费统计等功能，使用 `JOIN`、`GROUP BY` 等语句优化查询 8。

参与竞赛：在 LeetCode SQL 题目中提升逻辑思维。

项目实战：使用 Navicat 等工具设计 ER 图，建立表关联，编写存储过程处理复杂业务逻辑 78。

六、性能优化与安全实践

1. **索引优化**

创建索引加速查询：

```
CREATE INDEX idx_person_name ON Persons(LastName, FirstName);
```

使用 `EXPLAIN` 分析执行计划，识别慢查询瓶颈 9。

2. **权限管理**

授予用户特定权限：

```
GRANT SELECT, INSERT ON mydb.Persons TO 'user'@'localhost' IDENTIFIED BY 'password';
```

定期审计权限，及时撤销不必要的访问 7。

3. **数据备份**

使用 `mysqldump` 命令备份数据库：

```
mysqldump -u root -p mydb > backup.sql
```

七、常见问题与解决方案

1. **字段类型不匹配**

错误示例：`INSERT INTO Orders (OrderDate) VALUES ('2025-08-01');`（若 `OrderDate` 为 `DATETIME` 类型）。

解决：使用正确格式（如`'2025-08-01 12:00:00'`）或转换函数（如 `STR_TO_DATE()`）。

2. **外键约束冲突**

错误示例：删除被引用的主表记录。

解决：先删除从表关联记录，或设置 `ON DELETE CASCADE` 自动级联删除。

3. **事务回滚失败**

原因：未正确开启事务或存在隐性提交。

解决：确保在 `START TRANSACTION` 后执行所有操作，并检查是否有自动提交设置（如 MySQL 默认自动提交）。