数据库

前端精进 - 后端方向

版权声明

联系方式

如果你想要购买本课程
请微信联系 xiedaimala02 或 xiedaimala03

如果你发现有人盗用本课程
请微信联系 xiedaimala02 或 xiedaimala03

关系型数据库的范式

范式可以理解为设计标准

第一范式 1NF

- 定义
  - 字段不可再分
- 举例
  - 我们要存储体检者的双眼视力
  - 那么应该存为左眼视力和右眼视力两个字段
  - 即 user 表里应该有 left_eye 和 right_eye
  - 而不能把它们存在一个字段

第一范式的缺点

学号	姓名	系名	系主任	课名	分数
1022211101	李小明	经济系	王强	高等数学	95
1022211101	李小明	经济系	王强	大学英语	87
1022211101	李小明	经济系	王强	普通化学	76
1022211102	张莉莉	经济系	王强	高等数学	72
1022211102	张莉莉	经济系	王强	大学英语	98
1022211102	张莉莉	经济系	王强	计算机基础	88
1022511101	高芳芳	法律系	刘玲	高等数学	82
1022511101	高芳芳	法律系	刘玲	法学基础	82

图片来自知乎问答

这是一个学生选课表，没有违反第一范式，但是存在如下问题：
数据冗余、创建系时插入异常、删除学生会导致系消失、学生转系时改动多处

结论：第一范式不够强

第二范式 2NF

- 定义（不标准）
  - 在 1NF 的基础上，要有键（键可由多个字段组合）
  - 所有字段分别完全依赖于键
  - 如果键是多个字段组合，则不允许部分依赖于该键
- 依赖关系
  - 给出键，就能唯一确定字段的值
  - 如给出学号，就能唯一确定姓名，反之则不行
  - 则称姓名依赖于学号
- 不满足第二范式的地方
  - 上表的键为 (学号, 课名)
  - 但存在部分依赖：姓名依赖于学号

第二范式 2NF

- 改进
  - ✓ 选课表（学号、课名、分数）
  - ✓ 学生表（学号、姓名、系名、系主任）

学号	课名	分数
1022211101	高等数学	95
1022211101	大学英语	87
1022211101	普通化学	76
1022211102	高等数学	72
1022211102	大学英语	98
1022211102	计算机基础	88
1022511101	高等数学	82
1022511101	法学基础	82

学号	姓名	系名	系主任
1022211101	李小明	经济系	王强
1022211102	张莉莉	经济系	王强
1022511101	高芳芳	法律系	刘玲

第三范式 3NF

- 定义（不标准）
  - 一个表里不能有两层依赖
  - 给出学号，就能确定系名：系名依赖于学号
  - 给出系名，就能确定系主任：系主任依赖于系名
  - 所以，系主任间接依赖于学号
- 解决办法
  - 把系名和系主任单独建表

第三范式 3NF

学号	课名	分数
1022211101	高等数学	95
1022211101	大学英语	87
1022211101	普通化学	76
1022211102	高等数学	72
1022211102	大学英语	98
1022211102	计算机基础	88
1022511101	高等数学	82
1022511101	法学基础	82

学号	姓名	系名
1022211101	李小明	经济系
1022211102	张莉莉	经济系
1022511101	高芳芳	法律系

系名	系主任
经济系	王强
经济系	王强
法律系	刘玲

总结

- 第一范式
  - ✓ 属性不可分割
- 第二范式
  - ✓ 字段完全依赖于键
- 第三范式
  - ✓ 字段没有间接依赖于键
- BC范式
  - ✓ 键中的属性也不存在间接依赖


无忌，你记住了吗
忘掉这些知识，只有面试才会用
我已经全忘了

数据库设计经验

- 高内聚
  - 把相关的字段放到一起，不相关的分开建表
  - 如果两个字段能够单独建表，那就单独建表
- 低耦合
  - 如果两个表之间有弱关系
  - 一对一可放在一个表，也可两个表加外键
  - 一对多一般用外键
  - 多对多一般建中间表

一对一

- 假设一个学生只能加入一个班级

- 可以把班级放在学生表里
  - 学生 id: 1001 姓名: 小明 班级id: 4002
  - 班级 id: 4002 名称: 入门1班

- 也可以单独建立关联表
  - 学生 id: 1001 姓名: 小明
  - 学生班级： id: 2003 学生id: 1001 班级id: 4002
  - 班级 id: 4002 名称: 入门1班

一对多

- 假设一个作者能写多本书

- 可以把书放在作者表里吗?
- 某些 DBMS 支持数组，可以存两个 id 到一个字段
- 作者 id: 1001, 姓名：大牛, books: [2001, 3002]
- 如果不支持数组，就不能这样做了

- 单独建立关系表（推荐）
- 作者 id: 1001 姓名: 大牛
- 出版：id: 2001 作者id: 1001 书id: 4002, 出版社id
- 出版：id: 2002 作者id: 1001 书id: 4003, 出版社id
- 书 id: 4002 名称: JS 入门

多对多

- 假设一个学生可以加入多个班级
- 当然每个班级也能有多个学生

- 可以把班级放在学生表里吗?
  - 某些 DBMS 支持数组就可以放
  - 如果不支持数组，就不能放了

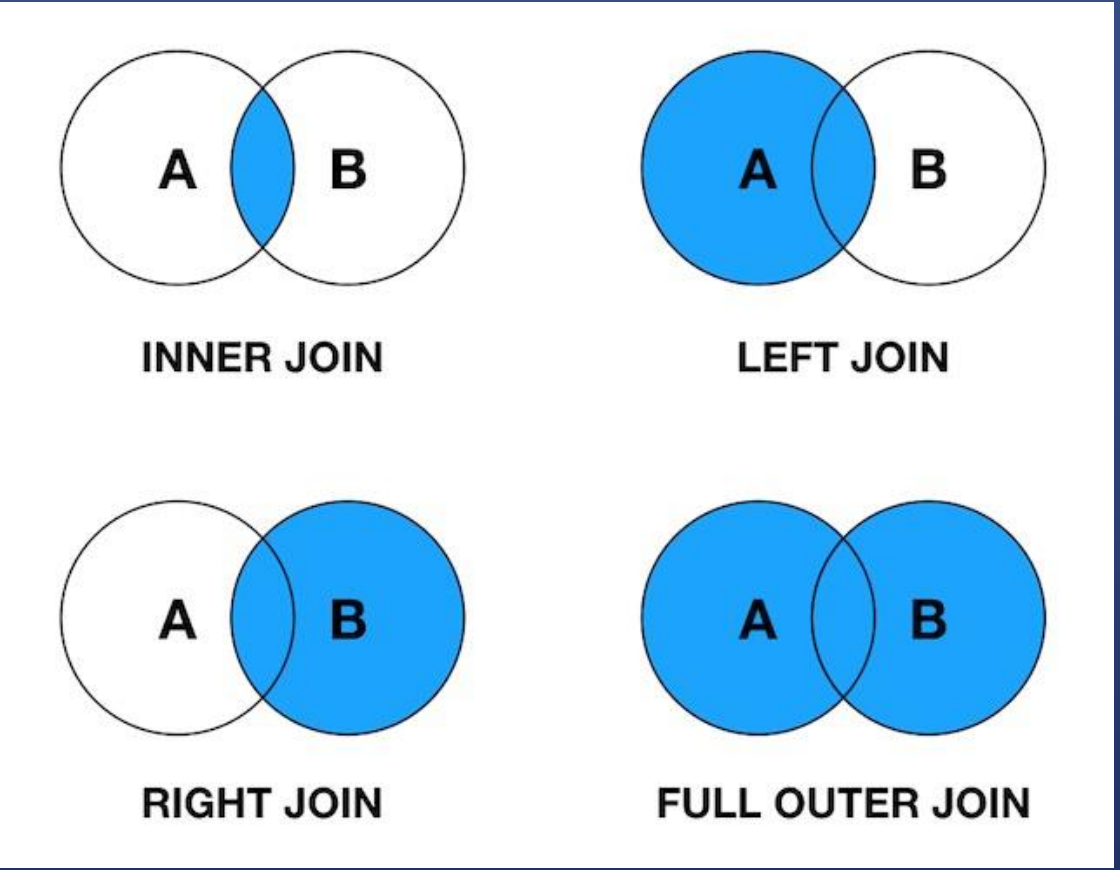
- 单独建立关系表（推荐）
  - 学生 id: 1001 姓名: 小明
  - 学生班级： id: 2001 学生id: 1001 班级id: 4002, 有效期
  - 学生班级： id: 2002 学生id: 1001 班级id: 4003, 有效期
  - 班级 id: 4002 名称: 入门1班

什么时候建关联表

- 当关联自身存在属性时
  - ✓ 比如关联的有效期，有效期为一年
  - ✓ 比如关联的级别，店铺会员分为 vip1~6

JOIN

- 连接表
  - ✓ inner join
  - ✓ left join
  - ✓ right join
  - ✓ full outer join
- 看图巧记
  - ✓ 参考文章



语法

把表名改为

T1 { [INNER] | { LEFT | RIGHT | FULL } [OUTER] } JOIN T2 ON boolean_expression

例如

```
SELECT A.PK AS A_PK, B.PK AS B_PK,
      A.Value AS A_Value, B.Value AS B_Value
FROM Table_A A
INNER JOIN Table_B B
ON A.PK = B.PK;
```

试试看

- 启动 mysql
  - docker container start mysql1
  - 或者
  - docker run --name mysql1 -e MYSQL_ROOT_PASSWORD=123456 -p 3306:3306 -d mysql:5.7.27
- 进入 mysql
  - docker exec -it mysql1 bash
  - mysql -u root -p
  - 输入密码 123456

试试看2

- 创建数据库

✓ CREATE DATABASE db1 CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;

✓ show databases;

✓ use db1;

- 创建表

✓ create table users(id serial, name text);

✓ create table staffs(id serial, name text);

✓ create table orders(id serial, user_id bigint unsigned, staff_id bigint unsigned, amount int unsigned);

试试看3

- 创建记录
  - insert into users (name) values ('XiaoMing');
  - insert into staffs (name) values ('XiaoHong');
  - insert into orders(user_id,staff_id, amount) values (1,1, 100);
- 使用 inner join
  - select users.name as uname, orders.amount as amount from users inner join orders on users.id =orders.user_id;
  - 得到 XiaoMing 100

其他 join

- Left join
  - ✓ 会保留右边的 null，以保证左边都显示
- Right join
  - ✓ 会保留左边的 null，以保证右边都显示
- Full outer join
  - ✓ 保留两边的 null，以保证两边都显示

缓存字段

- 假设一个博客 blog 包含多个评论 comments

- 如何获取博客的评论数
- select count(id) from comments where blog_id=8
- 这样太慢了
- 可不可以在 blog 表上加一个 comment_count 字段
- 每次添加 comment 则 +1
- 每次删除 comment 则 -1
- 可以的

事务

- 有些操作必须一次完成
  - 用户评论之后，要做两件事情
  - 第一步，在 comments 表新增记录
  - 第二步，在 blogs 表将对应的 comment_count +1
  - 如果第一步执行了，第二步没有执行怎么办
  - 数据就乱了
- 使用事务
  - 菜鸟的教程不错

```
start transaction;
语句1;  语句2; 语句3;
commit;
```

  - 只要有一句出错，则全都不生效。

MySQL 存储引擎

- 命令 SHOW ENGINES;

常见的

- InnoDB - 默认，目前版本是新版 InnoDB
- MyISAM - 拥有较高的插入、查询速度，但不支持事务
- Memory - 内存中，快速访问数据
- Archive - 只支持 insert 和 select

InnoDB

- InnoDB 是事务型数据库的首选，支持事务、遵循ACID、支持行锁和外键

索引

- 语法
  - CREATE UNIQUE INDEX index1 ON users(name(100))
  - show index in users;
  - 菜鸟的教程不错，有兴趣可以看看
- 用途
  - 提交搜索效率
  - where xxx>100 那么我们可以创建 xxx 的索引
  - where xxx>100 and yyy>200，创建 xxx,yyy 的索引

快速入门数据库

这些就是我们可能遇到的数据库知识

其他知识（如锁）用到了再去学习