数据库设计范式（没有数据冗余）

- 第一范式

1. 每一列属性都是不可再分的属性值，确保每一列的原子性
2. 两列的属性相近或相似或一样，尽量合并属性一样的列，确保不产生冗余数据。
3. 单一属性的列是由基本的数据类型所构成
4. 设计的表都是简单的二维表

- 第二范式
  满足1NF后，要求表中的所有列，都必须依赖于主键，而不能有任何一列与主键没有关系，也就是说一个表只描述一件事情

- 第三范式（3NF）：
  必须先满足第二范式（2NF），要求：表中的每一列只与主键直接相关而不是间接相关，（表中的每一列只能依赖于主键）

数据库事务四大特性ACID

- 原子性：一个事务是一个不可分割的最小工作单元，其操作要么全部成功，要么全部失败。
- 一致性：数据库总是从一个一致性状态转换为另一个一致性状态。所谓一致性状态，就是数据库的所有完整性约束（尤其注意用户定义约束）都被遵守，以银行转账为例，“转账操作必然导致一个账户减少金额，另一个账户增加金额，且这两个账户总金额之和不变”就是一个完整性约束。
- 持久性：一旦事务提交，则其所作的修改就会永久保存到数据库中。
- 隔离性：隔离性用于定义事务之间的相互隔离程度，存在四个隔离级别

基本操作

1. 插入

   - insert [into] 表名 [(列名1,列名2,列名3,…)] values (值1,值2,值3,…);
   - insert into students (name, sex, age) values("佳儿","男",21);

2. 查询

   - select 列名称 from 表名称 [查询条件];
   - select name,age from students where age=18;

3. 更新

   - update 表名称 set 列名称=新值 where 更新条件;
   - update students set tel="15200901235" where id=4;

4. 删除

   - delete from 表名称 where 删除条件;
   - delete from students where id=2;

5. 表的修改
   alter table 表名 add 列名 列数据类型 [after 插入位置];
   例：在表的最后追加列 address：
   alter table students add address char(60);

   修改列
   alter table 表名 change 列名称 列新名称 新数据类型;

   例：将students表中列tel改为telphone：
   alter table students change tel telphone char(13);

   删除列
   alter table 表名 drop 列名称;

   重命名表
   alter table 表名 rename 新表名;

   删除整张表
   drop table 表名;

6. 数据库管理
   添加用户：
   create user "test"@"localhost" identified by "123456";

   授权：
   grant all privileges on . to "test"@"localhost" with grant option;

   刷新权限：
   flush privileges

   查看用户权限：
   show grants for "root"@"localhost";

   修改指定用户密码：
   alter user "test"@"localhost" identified by "test123";

   删除用户：
   drop user "test"@"localhost";

MySQL触发器 aqlachemy触发器 使用类的继承实现触发器

- MySQL触发器
  - 触发器与存储过程非常相似，触发器也是SQL语句集，两者唯一的区别是触发器不能用EXECUTE语句调用，而是在用户执行Transact-SQL语句时自动触发（激活）执行。触发器是在一个修改了指定表中的数据时执行的存储过程。通常通过创建触发器来强制实现不同表中的逻辑相关数据的引用完整性和一致性。由于用户不能绕过触发器，所以可以用它来强制实施复杂的业务规则，以确保数据的完整性。触发器不同于存储过程，触发器主要是通过事件执行触发而被执行的，而存储过程可以通过存储过程名称名字而直接调用。当对某一表进行诸如UPDATE、INSERT、DELETE这些操作时，SQLSERVER就会自动执行触发器所定义的SQL语句，从而确保对数据的处理必须符合这些SQL语句所定义的规则
- aqlachemy触发器

```
1.在实体类中增加静态触发方法
@staticmethod
def on_created(target,value,initiator):
    target.role = Role.query.filter_by(name='Guests').first()

2.添加监听器
db.event.listen(User.name,'append',User.on_created)
```

- 使用类的继承实现触发器

```
class _ModelOp:
  def op_event_insert(self):
      return Event(Event.Type.BASIC,
                   Event.Action.INSERT,
                   unicode(self))

  def op_event_update(self, orig_name):
      event = Event(Event.Type.BASIC,
                    Event.Action.UPDATE,
                    unicode(self),
                    orig_name)
      return event

  def op_event_delete(self):
      return Event(Event.Type.BASIC,
                   Event.Action.DELETE,
                   unicode(self))

                   def op_insert(self):
                   pass

   def insert(self, commit_now=True):
       if self.exists():
           raise ConflictException(_('Object exists'))
       else:
           self.op_insert()
           db.session.add(self)
           event = self.op_event_insert()
           db.session.add(event)

           if commit_now:
               db.session.commit()
           else:
               db.session.flush()

在接口函数中创建时，初始化对象，然后调用obj.insert()
```

性能优化

- 使用索引
- 使用外键
- 使用连接（JOIN）来代替子查询(Sub-Queries)
  - 连表查询
  - User.query.w ith_entities(User.id,User.username,Department.department_name).join(User.dept).all()
  - 内连接：inner join 获取的就是两个表中的交集部分
  - 左连接：左连接会读取左边数据表的全部数据，即使右边数据表没有对应数据。(如果两个表中数据有相同部分，只显示一个)
  - 右连接：右连接会读取右边数据表的全部数据，即使左边数据表没有对应数据。(如果两个表中数据有相同部分，只显示一个)
- 选取最适用的字段属性
- 应该尽量把字段设置为NOT NULL

高可用

数据库备份

- 用mysqldump

数据库主从一致性检测

- percona-toolkit

数据库常见面试题

- 数据库常见面试题