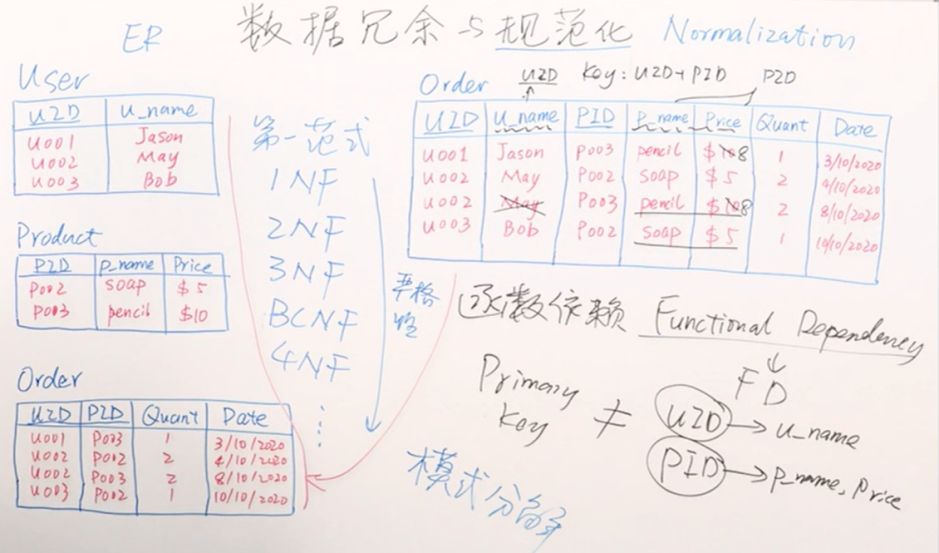
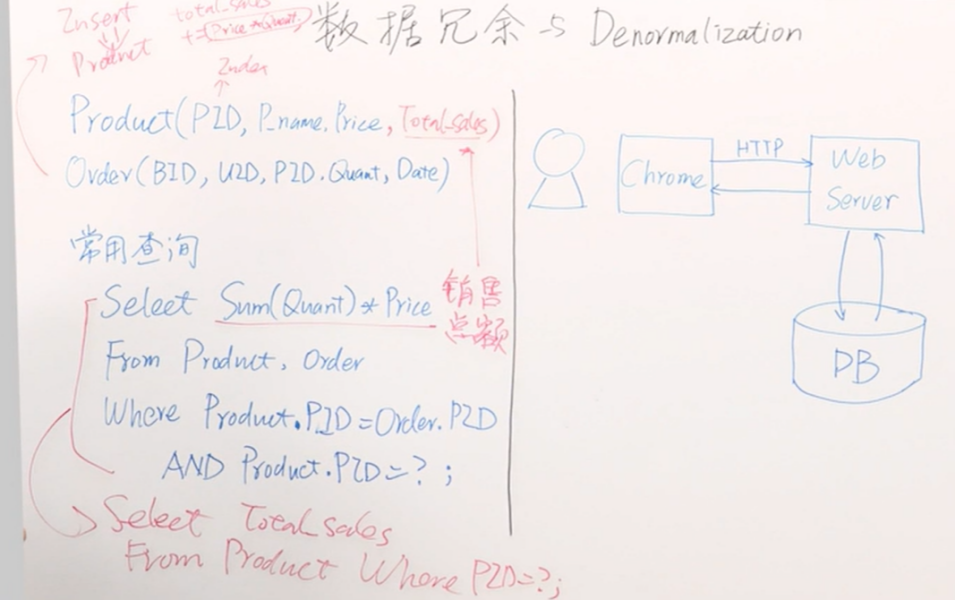
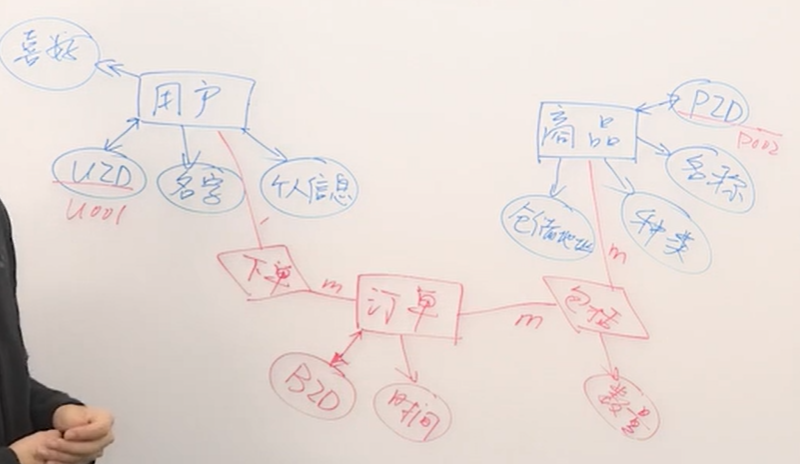
* 关系数据库中的冗余
* 数据库设计的规范化  
  
* 冗余带来的好处：提速
* 如何利用冗余  
  
* 网上购物实例
  1. 概念设计  
     
  2. 从ER图到关系设计
  3. 从ER图到文档模式

**第*1*题：假设有一个员工表 Employee(ID, Name, Address, Phone#)（其中ID为主键）。最初电话号码Phone#是一个单值属性，即每个人只有一个电话。后来发现，有少量员工有两个电话号码，且都需要记录。你会如何对数据库的结构设计进行调整？**

A：将Phone#的数据类型变成字符串，将两个电话号码用分号隔开，一起存在同一个字段中。

B：将主键变为 (ID, Phone#)，将有两个电话号码的人在表中存两份。

C：将模式修改为 Employee(ID, Name, Address)，Phone(ID, Phone#)。

D：将模式修改为 Employee(ID, Name, Address, Phone#, Phone#Alt)，其中Phone#Alt用于记录另一个电话号码。√

**第*2*题：关于数据冗余的说法，哪一项是不正确的？**

A：冗余会增加数据更新的代价

B：冗余会增加数据删除的代价

C：冗余不会增加数据查询的代价 √

D：冗余会增加软件开发的复杂

**第*3*题：对一张属性很多的表（通常称为“宽表”），为了提升表访问的性能，我们可以选择将表拆成两份或多份。比如，将 Employee (ID, name, salary, tax, mgr#, dept#) 拆成 Employee (ID, name, salary) 和 Employee (ID, tax, mgr#, dept#)。请问做表拆分的原则应该是什么？**

A：让同时被使用到的属性（即出现在同一个SQL中的属性）尽可能位于分拆后的同一张表中。√

B：让经常被更新的属性尽可能位于分拆后的同一张表中。

C：让被索引的属性尽可能位于分拆后的同一张表中。

D：让可能为空的属性尽可能位于分拆后的同一张表中。

**第*4*题：假设有一张表：行程表（旅客号码，导游号码，打卡景点），记录了哪些旅客被哪些导游带领去了哪些地方。我们发现了很多如下形式的规律：不同的旅客在同一位导游的带领下，如果去了景点A，那么必定也会去景点B。请问数据库中是否存在冗余信息，可以如何处理？**

A：没有冗余

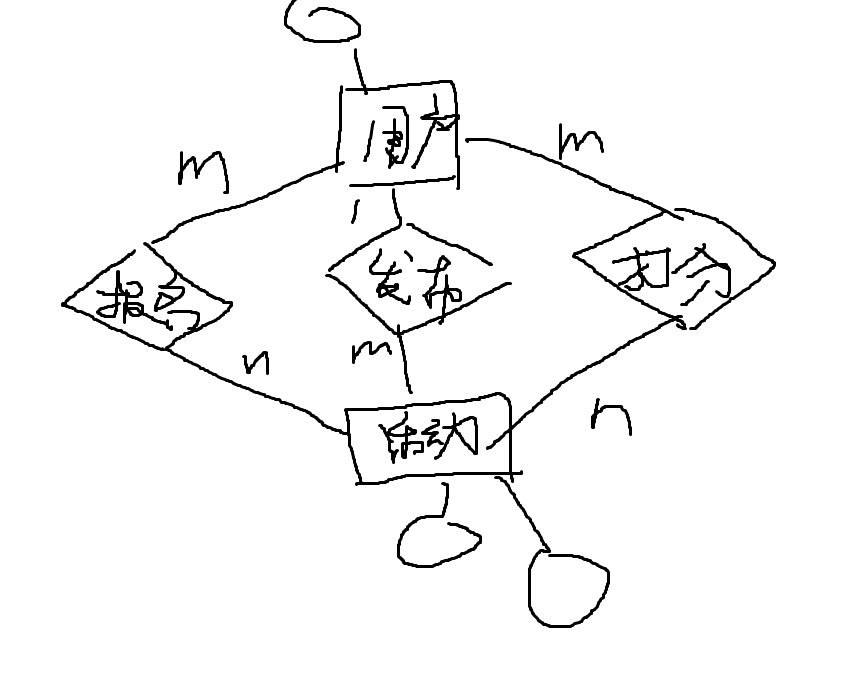
B：有冗余，但无法通过结构设计去除

C：有冗余，适当改变结构设计后可去除 √

D：以上说法均不对

一个校园学术活动网站。所有学生和老师都可以成为网站的用户，将自己组织的学术活动在网站上发布，比如学术讲座、读书会等等。发布活动时需要指定活动的标题、时间、地点、类别和内容。一旦活动发布，其他用户就可以浏览活动的内容并报名。通常活动有人数限制，活动发起人可以选择部分报名者成为活动参与人，也可以开放给所有报名者参与。活动结束后，参与人可以给活动打分，并写评论记录活动内容。每位用户发起的所有活动将永久被网站记录，其他用户可以随时查看他发起活动的历史，以及平均得分。

请先用ER图为以上场景设计数据库模式，然后再根据应用访问数据库的方式对模式进行优化。（提示：需考虑索引的使用。）



User (UID, UType, Name, Score…)

Activity (AID, UID, Title, Time, Address, AType, Content, Limit, IsOpen, AVGScore) 索引：UID

Signup (UID, AID, Comment, Score) 索引：AID