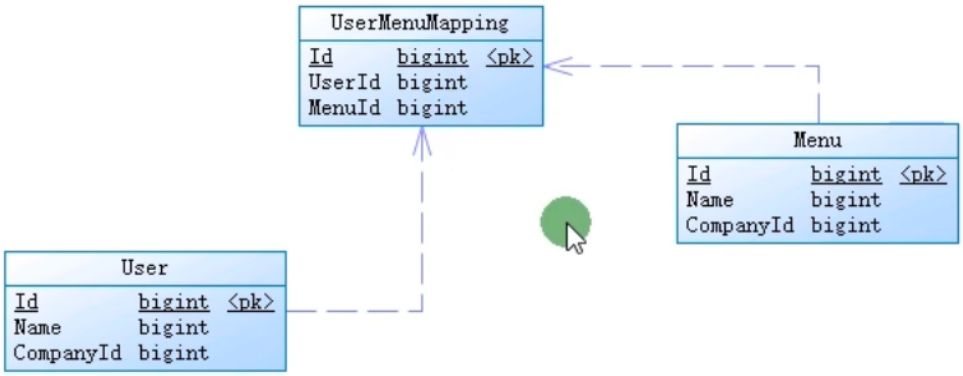
一：关系型数据库的设计实录

一对一：除非分表的情况，一般都是写在一张表里面

一对多：一般是通过外键来创建表的

多对多：一般是创建一个中间map表，然后两个表里面的数据通过map表来互相查询



二：三大范式

第一范式（1NF）：数据表中的每一列（每个字段）必须是不可拆分的最小单元，也就是确保每一列的原子性；

第二范式（2NF）：在1NF的基础上，非码属性必须完全依赖于候选码（在1NF基础上消除非主属性对主码的部分函数依赖）

第二范式需要确保数据库表中的每一列都和主键相关，而不能只与主键的某一部分相关（主要针对联合主键而言）。

举例说明：

订单号	产品号	产品数量	产品折扣	产品价格	订单金额	订单时间
2008003	205	100	0.9	8.9	2870	20080103
2008003	206	200	0.8	9.9	2870	20080103
2008005	207	200	0.75	10	2000	20080203
2008006	207	400	0.85	12	4800	20080206
2008007	207	1000	0.88	14	14000	20080209
2008008	210	240	0.95	8	12255	20100423
2008008	211	300	0.75	8	12255	20100423
2008008	212	350	0.8	15.9	12255	20100423

在上图所示的情况中，同一个订单中可能包含不同的产品，因此主键必须是“订单号”和“产品号”联合组成，

但可以发现，产品数量、产品折扣、产品价格与“订单号”和“产品号”都相关，但是订单金额和订单时间仅与“订单号”相关，与“产品号”无关，

这样就不满足第二范式的要求，调整如下，需分成两个表：

订单号	产品号	产品数量	产品折扣	产品价格
2008003	205	100	0.9	8.9
2008003	206	200	0.8	9.9
2008005	207	200	0.75	10
2008006	207	400	0.85	12
2008007	207	1000	0.88	14
2008008	210	240	0.95	8
2008008	211	300	0.75	8
2008008	212	350	0.8	15.9

订单号	订单金额	订单时间
2008003	2870	20080103
2008003	2870	20080103
2008005	2000	20080203
2008006	4800	20080206
2008007	14000	20080209
2008008	12255	20100423
2008008	12255	20100423
2008008	12255	20100423

第三范式（3NF）：在2NF基础上，任何非主属性不依赖于其它非主属性（在2NF基础上消除传递依赖）

第三范式需要确保数据表中的每一列数据都和主键直接相关，而不能间接相关。

举例说明：

学号	姓名	性别	家庭人口	班主任姓名	班主任性别	班主任年龄
20150001	李白	男	3口人	陈洁	女	35
20150002	杜甫	男	2口人	陈洁	女	35
20150003	王维	男	4口人	陈洁	女	35
20150004	白居易	男	3口人	李丽	女	32
20150005	刘禹锡	男	4口人	李丽	女	32
20150006	李清照	女	5口人	王安	男	29
20150007	苏轼	男	2口人	南林	男	34
20150008	屈原	男	4口人	南林	男	34
20150009	陶渊明	男	1口人	王安	男	29

上表中，所有属性都完全依赖于学号，所以满足第二范式，但是“班主任性别”和“班主任年龄”直接依赖的是“班主任姓名”，

而不是主键“学号”，所以需做如下调整：

学号	姓名	性别	家庭人口	班主任姓名
20150001	李白	男	3口人	陈洁
20150002	杜甫	男	2口人	陈洁
20150003	王维	男	4口人	陈洁
20150004	白居易	男	3口人	李丽
20150005	刘禹锡	男	4口人	李丽
20150006	李清照	女	5口人	王安
20150007	苏轼	男	2口人	南林
20150008	屈原	男	4口人	南林
20150009	陶渊明	男	1口人	王安

班主任姓名	班主任性别	班主任年龄
陈洁	女	35
陈洁	女	35
陈洁	女	35
李丽	女	32
李丽	女	32
王安	男	29
南林	男	34
南林	男	34
王安	男	29

这样以来，就满足了第三范式的要求。

ps:如果把上表中的班主任姓名改成班主任教工号可能更确切，更符合实际情况，不过只要能理解就行。

三：数据库相关优化

（1）死锁：死锁不可避免。但是可以尽量减少。遵循以下原则

1.让事务里面的表都按照固定顺序执行，避免A表等B表。B表等A表的情况

2.事务尽量简短，只把需要事务一起执行的表放到事物中

3.尽量减少并发；优化事务里面的SQL执行时间；设置锁的隔离级别

（2）存储过程：存储过程少用，因为大量占用数据库资源。除非是大量数据关联查询之类或者报表。