



下载APP



## 23 | ER模型：如何理清数据库设计思路？

2021-05-04 朱晓峰

MySQL 必知必会

[进入课程 >](#)



讲述：朱晓峰

时长 10:33 大小 9.68M



你好，我是朱晓峰。

在超市项目的数据库设计阶段，超市经营者交给我们一大堆 Excel 表格。我们需要把这些表格的数据都整理清楚，并且按照一定的规则存储起来，从而进行高效的管理。

比如，当时我们有这样一张进货表：



| listnumber<br>(单号) | supplierid<br>(供货商编号) | Suppliername<br>(供货商名称) | stock<br>(仓库) | barcode<br>(条码) | goodsname<br>(名称) | property<br>(属性) | quantity<br>(数量) | price<br>(进货<br>价格) | importvalue<br>(进货金额) |
|--------------------|-----------------------|-------------------------|---------------|-----------------|-------------------|------------------|------------------|---------------------|-----------------------|
| 3478               | 1                     | 出版社                     | 仓库            | 0001            | 书                 | 16开/本            | 10               | 45                  | 450                   |
| 3478               | 1                     | 出版社                     | 仓库            | 0002            | 地图                | 张                | 5                | 9.9                 | 49.5                  |
| 3490               | 2                     | 文具厂                     | 卖场            | 0003            | 笔                 | 10支/包            | 1                | 3.5                 | 3.5                   |
| 3492               | 1                     | 出版社                     | 仓库            | 0002            | 地图                | 张                | 10               | 9.5                 | 95                    |

为了提高数据存储的效率，我们按照第三范式的原则进行拆分，这样就得到了 4 个表，分别是供货商表、进货单头表、进货单明细表和商品信息表。

供货商表：

| supplierid | suppliername |
|------------|--------------|
| 1          | 出版社          |
| 2          | 铅笔厂          |

进货单头表：

| listnumber | supplierid | stock |
|------------|------------|-------|
| 3478       | 1          | 仓库    |
| 3490       | 2          | 卖场    |
| 3492       | 1          | 卖场    |

进货单明细表：

| listnumber | itemnumber | quantity | importprice | importvalue |
|------------|------------|----------|-------------|-------------|
| 3478       | 1          | 10       | 45          | 450         |
| 3478       | 2          | 5        | 9.9         | 49.5        |
| 3490       | 3          | 1        | 3.5         | 3.5         |
| 3492       | 2          | 10       | 9.5         | 95          |

商品信息表：

| itemnumber | barcode | goodsname | specification | unit |
|------------|---------|-----------|---------------|------|
| 1          | 0001    | 书         | 16开           | 本    |
| 2          | 0002    | 地图        | NULL          | 张    |
| 3          | 0003    | 笔         | 10支           | 包    |

其中，商品信息表、供货商表和进货单头表都满足第三范式的原则，进货单明细表虽然不满足第三范式的原则，但是满足第二范式的要求，而且保留的冗余字段也是基于业务优先

的原则保留的。因此，超市经营者给我们提供的进货单表，经过我们的拆解，已经是存取效率最佳的方案了。在进货管理这个局部模块中，是最优的数据库设计方案。

但是，当我们按照这样的方式拆分一连串数据表时，却发现越拆越多，而且支离破碎。事实上，**局部最优的表，不仅有可能存在进一步拆分的情况，还有可能会出现数据缺失。**

毕竟，数据库设计是牵一发而动全身的。那有没有什么办法提前看到数据库的全貌呢？比如需要哪些数据表、数据表中应该有哪些字段，数据表与数据表之间有什么关系、通过什么字段进行连接，等等。这样我们才能进行整体的梳理和设计。

其实，ER 模型就是一个这样的工具。ER 模型也叫作实体关系模型，是用来描述现实生活中客观存在的事物、事物的属性，以及事物之间关系的一种数据模型。**在开发基于数据库的信息系统的设计阶段，通常使用 ER 模型来描述信息需求和信息特性，帮助我们理清业务逻辑，从而设计出优秀的数据库。**

今天，我还是借助实际案例，带你使用 ER 模型分析一下超市的业务流程，具体给你讲一讲怎么通过 ER 模型来理清数据库设计的思路，从而设计出优秀的数据库。

在使用之前，咱们得先知道 ER 模型里都包括啥。

## ER 模型包括哪些要素？

**在 ER 模型里面，有三个要素，分别是实体、属性和关系。**

实体，可以看做是数据对象，往往对应于现实生活中的真实存在的个体。比如，这个连锁超市就可以看做一个实体。在 ER 模型中，用矩形来表示。实体分为两类，分别是**强实体和弱实体**。强实体是指不依赖于其他实体的实体；弱实体是指对另一个实体有很强的依赖关系的实体。

属性，则是指实体的特性。比如超市的地址、联系电话、员工数等。在 ER 模型中用椭圆形来表示。

关系，则是指实体之间的联系。比如超市把商品卖给顾客，就是一种超市与顾客之间的联系。在 ER 模型中用菱形来表示。

需要注意的是，有的时候，实体和属性不容易区分。比如刚刚商品信息表中的商品的单位，到底是实体还是属性呢？如果从进货的角度出发，单位是商品的属性，但是从超市信息系统的整体出发，单位可以看做一个实体。

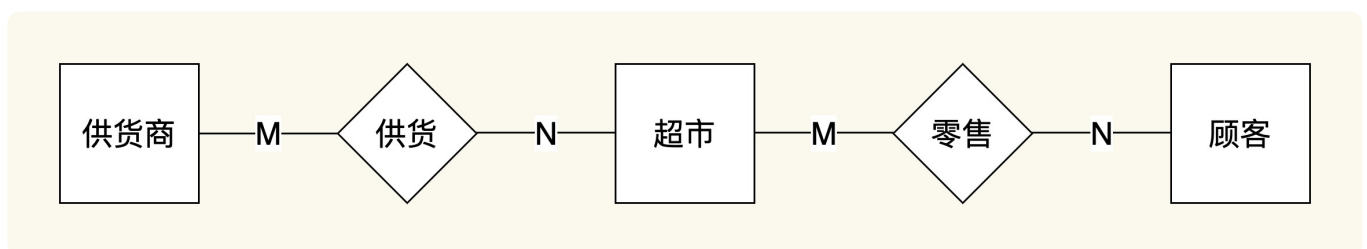
## 那么，该如何区分实体和属性呢？

我给你提供一个原则：我们要从系统整体的角度出发去看，**可以独立存在的是实体，不可再分的是属性**。也就是说，属性不需要进一步描述，不能包含其他属性。

在 ER 模型的 3 个要素中，关系又可以分为 3 种类型，分别是 **1 对 1**、**1 对多**和**多对多**。

1. 1 对 1：指实体之间的关系是一一对应的，比如个人与身份证信息之间的关系就是 1 对 1 的关系。一个人只能有一个身份证信息，一个身份证信息也只属于一个人。
2. 1 对多：指一边的实体通过关系，可以对应多个另外一边的实体。相反，另外一边的实体通过这个关系，则只能对应唯一的一边的实体。比如超市与超市里的收款机之间的从属关系，超市可以拥有多台收款机，但是每一条收款机只能从属于一个超市。
3. 多对多：指关系两边的实体都可以通过关系对应多个对方的实体。比如在进货模块中，供货商与超市之间的关系就是多对多的关系，一个供货商可以给多个超市供货，一个超市也可以从多个供货商那里采购商品。

知道了这些要素，我们就可以给超市业务创建 ER 模型了，如下图所示：



我来简单解释一下这个图。

在这个图中，供货商和超市之间的供货关系，两边的数字都不是 1，表示多对多的关系。同样，超市和顾客之间的零售关系，也是多对多的关系。

这个 ER 模型，包括了 3 个实体之间的 2 种关系：

1. 超市从供货商那里采购商品；
2. 超市把商品卖给顾客。

有了这个 ER 模型，我们就可以从整体上理解超市的业务了。但是，这里没有包含属性，这样就无法体现实体和关系的具体特征。现在，我们需要把属性加上，用椭圆来表示，这样我们得到的 ER 模型就更加完整了。

## ER 模型的细化

刚刚的 ER 模型展示了超市业务的框架，但是只包括了供货商、超市和顾客这三个实体，以及它们之间的关系，还不能对应到具体的表，以及表与表之间的关联。

因此，我们需要进一步去设计一下这个 ER 模型的各个局部，也就是细化下超市的具体业务流程，然后把它们综合到一起，形成一个完整的 ER 模型。这样可以帮助我们理清数据库的设计思路。

我们刚才的超市业务模型，包括了两个模块，分别是进货模块和销售模块。下面我们分别对这 2 个模块进行细化。

首先，我们来看一下超市业务中的进货模块的 ER 模型，整理一下其中包含哪些实体、哪些关系和哪些属性。

在我们的进货模块里，有 5 个实体：

1. 供货商
2. 商品
3. 门店
4. 仓库
5. 员工

其中，供货商、商品和门店是强实体，因为它们不需要依赖其他任何实体。而仓库和员工是弱实体，因为它们虽然都可以独立存在，但是它们都依赖门店这个实体，因此都是弱实体。

接下来，我们再分析一下各个实体都有哪些属性。

供货商：名称、地址、电话、联系人。

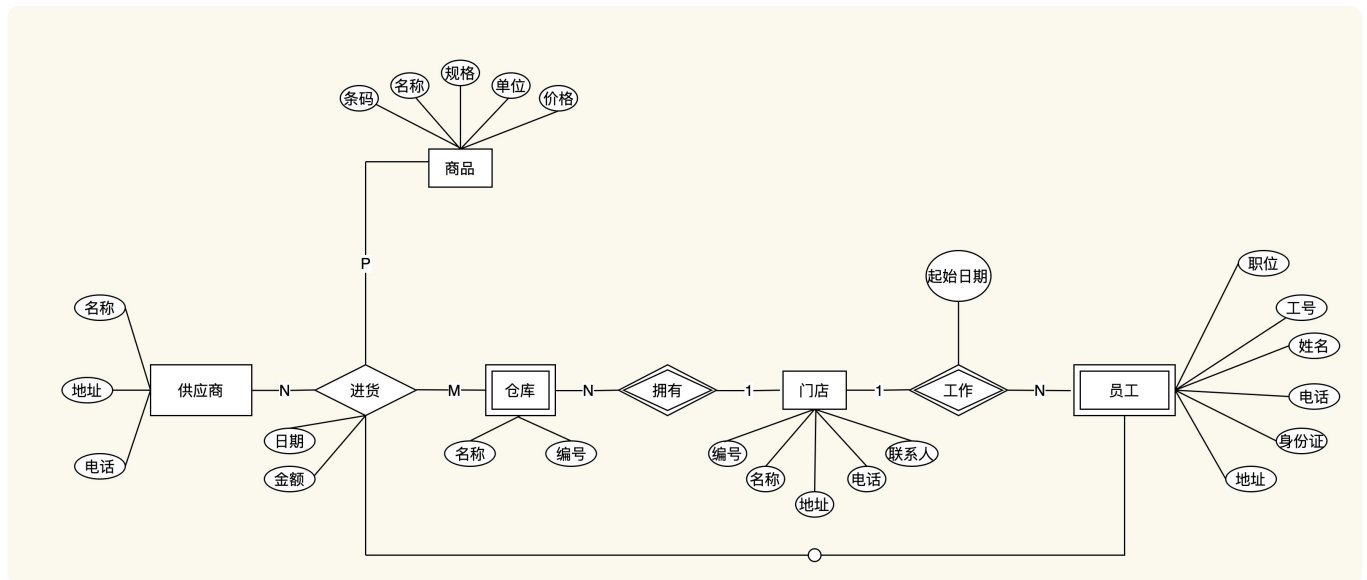
商品：条码、名称、规格、单位、价格。

门店：编号、地址、电话、联系人。

仓库：编号、名称。

员工：工号、姓名、住址、电话、身份证号、职位。

这样细分之后，我们就可以重新设计进货模块了，ER 模型如下：



需要注意的是，这里我是用粗框矩形表示弱实体，用粗框菱形，表示弱实体与它依赖的强实体之间的关系。

第二步，我们再分析一下零售模块。

经过分析，我们发现，在超市的业务流程中，零售业务包括普通零售和会员零售两种模式。普通零售包含的实体，包括门店、商品和收银款台；会员零售包含的实体，包括门店、商品、会员和收银款台。

这样我们就可以提炼出零售业务模块中的实体：

## 1. 商品

2. 门店
3. 会员
4. 收银款台。

其中，商品和门店不依赖于任何其他实体，所以是强实体；会员和收银款台都依赖于门店，所以是弱实体。

有了实体之后，我们就可以确定实体的属性了。

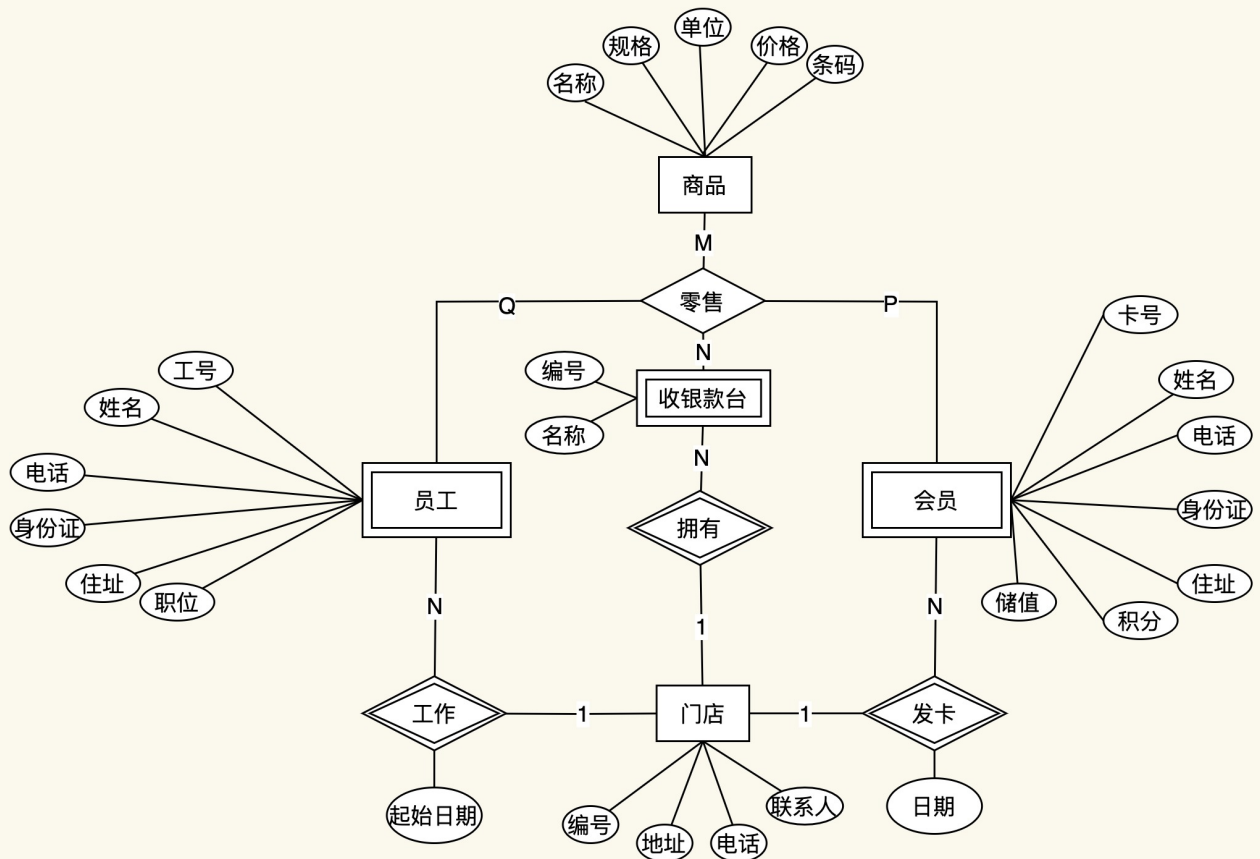
商品：条码、名称、规格、单位、价格。

会员：卡号、发卡门店、名称、电话、身份证、地址、积分、储值。

门店：编号、地址、电话、联系人。

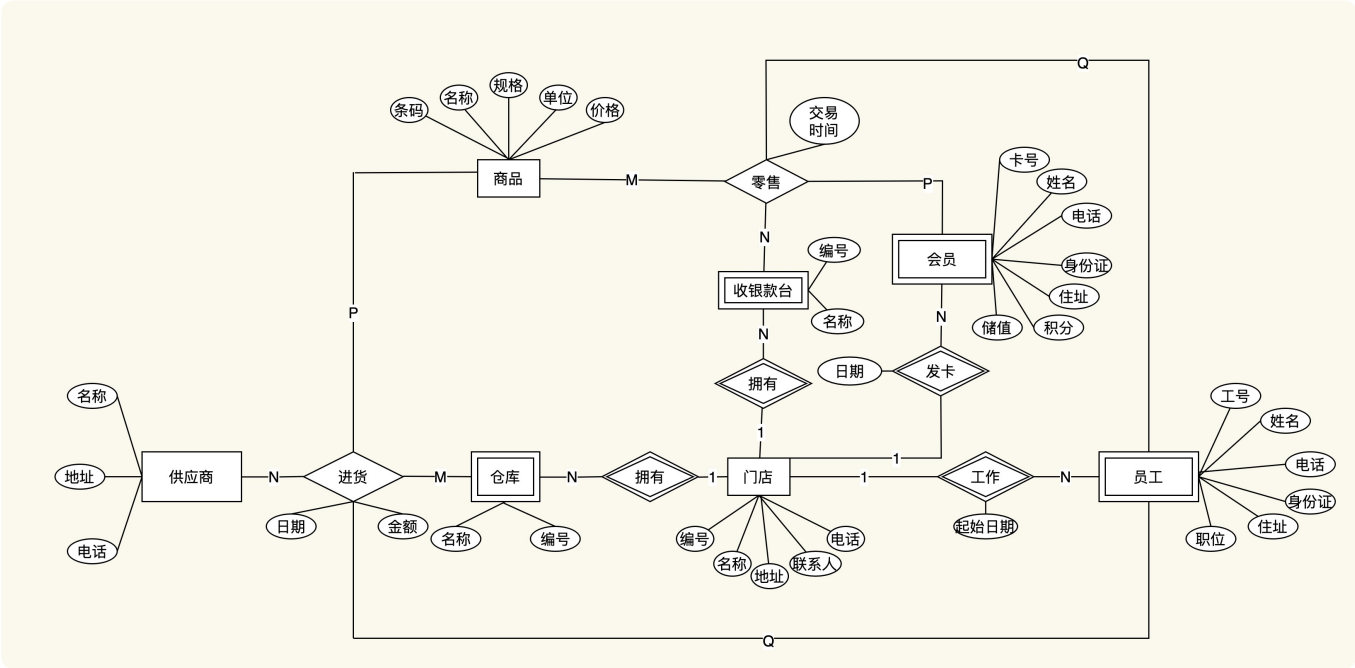
收银款台：编号、名称。

现在，我们就可以重新设计零售模块的 ER 模型了：





现在，我们把这两个图整合到一起，形成一个完整的 ER 模型：



## 如何把 ER 模型图转换成数据表？

通过绘制 ER 模型，我们经理清了业务逻辑，现在，我们就要进行非常重要的一步了：**把绘制好的 ER 模型，转换成具体的数据表。**

我来介绍下转换的原则。

1. 一个实体通常转换成一个数据表；
2. 一个多对多的关系，通常也转换成一个数据表；
3. 一个 1 对 1，或者 1 对多的关系，往往通过表的外键来表达，而不是设计一个新的数据表；
4. 属性转换成表的字段。

好了，下面我就结合前面的表格，给你具体讲解一下怎么运用这些转换的原则，把 ER 模型转换成具体的数据表，从而把抽象出来的数据模型，落实到具体的数据库设计当中。

### 一个实体转换成一个数据表


先来看一下强实体转换成数据表。

供货商实体转换成供货商表（demo.supplier）的代码如下所示：

 复制代码


```
1 mysql> CREATE TABLE demo.supplier
2 -> (
3 -> -- 我们给它添加一个与业务无关的字段“supplierid”为主键，并且设置自增约束。
4 -> supplierid INT PRIMARY KEY AUTO_INCREMENT,
5 -> suppliername TEXT,
6 -> address TEXT,
7 -> phone TEXT
8 -> );
9 Query OK, 0 rows affected (0.06 sec)
```

商品实体转换成商品表（demo.goodsmaster）：

 复制代码

```
1 mysql> CREATE TABLE demo.goodsmaster
2 -> (
3 -> --我们给商品信息表添加一个与业务无关的字段“itemnumber”为主键，采用手动赋值的方式，原因是
4 -> itemnumber INT PRIMARY KEY,
5 -> barcode TEXT,
6 -> goodsname TEXT,
7 -> specification TEXT,
8 -> unit TEXT,
9 -> salesprice DECIMAL(10,2)
10 -> );
11 Query OK, 0 rows affected (0.07 sec)
```

门店实体转换成门店表（demo.branch）：

 复制代码

```
1 mysql> CREATE TABLE demo.branch
2 -> (
3 -> -- 增加一个与业务无关的字段为主键，并且设置自增约束
4 -> branchid INT PRIMARY KEY AUTO_INCREMENT,
5 -> branchno TEXT,
6 -> branchname TEXT,
7 -> address TEXT,
8 -> phone TEXT,
9 -> contacter TEXT
10 -> );
11 Query OK, 0 rows affected (0.06 sec)
```

下面我们再把弱实体转换成数据表。

仓库转换成仓库表 (demo.stock)：

[复制代码](#)

```
1 mysql> CREATE TABLE demo.stock
2 -> (
3 -> --添加与业务无关的自增约束字段为主键
4 -> stockid INT PRIMARY KEY AUTO_INCREMENT,
5 -> -- 仓库是弱实体，依赖于强实体门店表，所以要把门店表的主键字段包括进来，作为与门店表关联的
6 -> branchid INT NOT NULL,
7 -> stockno TEXT NOT NULL,
8 -> stockname TEXT NOT NULL,
9 -> -- 设置外键约束，与门店表关联
10 -> CONSTRAINT fk_stock_branch FOREIGN KEY (branchid) REFERENCES branch (branchid)
11 -> );
12 Query OK, 0 rows affected (0.07 sec)
```

收银款台实体转换成收银款台表 (demo.cashier)：

[复制代码](#)

```
1 mysql> CREATE TABLE demo.cashier
2 -> (
3 -> -- 添加与业务无关的自增字段为主键
4 -> cashierid INT PRIMARY KEY AUTO_INCREMENT,
5 -> -- 收银款台是弱实体，依赖于强实体门店表，所以要把门店表的主键字段包括进来，所与与门店表关联
6 -> branchid INT NOT NULL,
7 -> cashierno TEXT NOT NULL,
8 -> cashiername TEXT NOT NULL,
9 -> -- 设置外键约束，与门店表关联
10 -> CONSTRAINT fk_cashier_branch FOREIGN KEY (branchid) REFERENCES branch (branchid)
11 -> );
12 Query OK, 0 rows affected (0.07 sec)
```

员工实体转换成员工表 (demo.operator)：

[复制代码](#)

```
1 mysql> CREATE TABLE demo.operator
2 -> (
3 -> -- 添加与业务无关的自增字段为主键
4 -> operatorid INT PRIMARY KEY AUTO_INCREMENT,
5 -> -- 员工是弱实体，依赖于强实体门店表，所以要把门店表的主键字段包括进来，所与与门店表关联的
6 -> branchid INT NOT NULL,
```

```
7 -> workno TEXT NOT NULL,
8 -> operatorname TEXT NOT NULL,
9 -> phone TEXT,
10 -> address TEXT,
11 -> pid TEXT,
12 -> duty TEXT,
13 -> -- 设置外键约束，与门店表关联
14 -> CONSTRAINT fk_operator_branch FOREIGN KEY (branchid) REFERENCES branch (bra
15 -> );
16 Query OK, 0 rows affected (0.11 sec)
```

## 会员实体转换成会员表 (demo.membermaster)：

[复制代码](#)


```
1 mysql> CREATE TABLE demo.membermaster
2 -> (
3 -> -- 添加与业务无关的自增字段为主键
4 -> memberid INT PRIMARY KEY,
5 -> -- 会员是弱实体，依赖于强实体门店表，所以要把门店表的主键字段包括进来，所为与门店表关联的:
6 -> branchid INT NOT NULL,
7 -> cardno TEXT NOT NULL,
8 -> membername TEXT,
9 -> address TEXT,
10 -> phone TEXT,
11 -> pid TEXT,
12 -> -- 设置默认约束，积分默认为0
13 -> memberpoints DECIMAL(10,1) NOT NULL DEFAULT 0,
14 -> -- 设置默认约束，储值默认为0
15 -> memberdeposit DECIMAL(10,2) NOT NULL DEFAULT 0,
16 -> -- 设置外键约束，与门店表关联
17 -> CONSTRAINT fk_member_branch FOREIGN KEY (branchid) REFERENCES branch (branc
18 -> );
19 Query OK, 0 rows affected (0.07 sec)
```

## 一个多对多的关系转换成一个数据表

这个 ER 模型中的多对多的关系有 2 个，分别是零售和进货。我们分别设计一个独立的表来表示，这种表一般称为中间表。

我们可以设计一个独立的进货单表，来代表进货关系。这个表关联到 4 个实体，分别是供货商、商品、仓库、员工，所以，表中必须要包括这 4 个实体转换成的表的主键。除此之外，我们还要包括进货关系自有的属性：进货时间、供货数量和进货价格。

按照数据表设计的第三范式的要求和业务优先的原则，我们把这个进货单表拆分成 2 个表，分别是进货单头表和进货单明细表：

 复制代码

```
1 CREATE TABLE demo.importhead
2 (
3   importid INT PRIMARY KEY,    -- 添加与业务无关的字段为主键
4   listnumber TEXT NOT NULL,
5   supplierid INT NOT NULL,    -- 供货商表的主键，反映了参与进货关系的供货商信息
6   stockid INT NOT NULL,       -- 仓库表的主键，反映了参与进货关系的参考信息
7   operatorid INT NOT NULL,    -- 员工表的主键，反映了参与进货关系的员工信息
8   recordingdate DATETIME NOT NULL,
9   totalquantity DECIMAL(10,3) NOT NULL DEFAULT 0,
10  totalvalue DECIMAL(10,3) NOT NULL DEFAULT 0,
11  CONSTRAINT fk_importhead_supplier FOREIGN KEY (supplierid) REFERENCES supplier
12  CONSTRAINT fk_importhead_stock FOREIGN KEY (stockid) REFERENCES stock (stockid)
13  CONSTRAINT fk_importhead_operator FOREIGN KEY (operatorid) REFERENCES operator
14 );
15 CREATE TABLE demo.importdetails
16 (
17   importid INT,
18   itemnumber INT,             -- 商品表的主键，反映了参与进货关系的商品信息
19   importquantity DECIMAL(10,3) NOT NULL DEFAULT 0,
20   importprice DECIMAL(10,2) NOT NULL DEFAULT 0,
21   importvalue DECIMAL(10,2) NOT NULL DEFAULT 0,
22   PRIMARY KEY (importid,itemnumber),
23   CONSTRAINT fk_importdetails_goodsmaster FOREIGN KEY (itemnumber) REFERENCES go
24 );
25
```

对于零售关系，我们可以设计一张流水单来表示。

这个表关联 4 个实体，分别是收银款台、商品、会员和员工。所以，表中也必须要包括这 4 个实体转换成的表的主键。除此之外，表中还要包括进货关系自有的属性：交易时间、数量、价格等。

按照数据表设计的第三范式的要求，我们把这个流水单表拆分成 2 个表，分别是流水单头表和流水单明细表：

 复制代码

```
1 CREATE TABLE demo.transactionhead
2 (
3   transactionid INT PRIMARY KEY,    -- 添加与业务无关的字段为主键
```


```

4 transactionno TEXT NOT NULL,
5 cashierid INT NOT NULL,          -- 收款机表的主键，反映了参与零售关系的收款机信息
6 memberid INT,                   -- 会员表的主键，反映了参与零售关系的会员的信息
7 operatorid INT NOT NULL,        -- 员工表的主键，反映了参与零售关系的员工信息
8 transdate DATETIME NOT NULL,
9 CONSTRAINT fk_transactionhead_cashier FOREIGN KEY (cashierid) REFERENCES cashi
10 CONSTRAINT fk_transactionhead_member FOREIGN KEY (memberid) REFERENCES member
11 CONSTRAINT fk_transactionhead_operator FOREIGN KEY (operatorid) REFERENCES ope
12 );
13 CREATE TABLE demo.transactiondetails
14 (
15 transactionid INT,
16 itemnumber INT,                 -- 商品表的主键，反映了参与零售关系的商品信息
17 quantity DECIMAL(10,3) NOT NULL DEFAULT 0,
18 price DECIMAL(10,2) NOT NULL DEFAULT 0,
19 salesvalue DECIMAL(10,2) NOT NULL DEFAULT 0,
20 PRIMARY KEY (transactionid,itemnumber),
21 CONSTRAINT fk_transactiondetails_goodsmaster FOREIGN KEY (itemnumber) REFERENC
22 );
23

```

## 通过外键来表达 1 对多的关系

在上面的表的设计中，我们已经完成了用外键来表达 1 对多的关系。比如：在流水单头表中，我们分别把 cashierid、memberid 和 operatorid 定义成了外键：


 复制代码

```

1 CONSTRAINT fk_transactionhead_cashier FOREIGN KEY (cashierid) REFERENCES cashi
2 CONSTRAINT fk_transactionhead_member FOREIGN KEY (memberid) REFERENCES member
3 CONSTRAINT fk_transactionhead_operator FOREIGN KEY (operatorid) REFERENCES ope

```

在流水单明细表中，我们把商品编号定义成了外键：

 复制代码

```

1 CONSTRAINT fk_transactiondetails_goodsmaster FOREIGN KEY (itemnumber) REFERENC

```

## 把属性转换成表的字段

在刚刚的设计中，我们也完成了把属性都转换成了表的字段，比如把商品属性（包括条码、名称、规格、单位、价格）转换成了商品信息表中的字段：

```
1 mysql> CREATE TABLE demo.goodsmaster
2 -> (
3 -> --我们给商品信息表添加一个与业务无关的字段“itemnumber”为主键，采用手动赋值的方式，原因是
4 -> itemnumber INT PRIMARY KEY,
5 -> barcode TEXT,           -- 条码属性
6 -> goodsname TEXT,        -- 名称属性
7 -> specification TEXT,    -- 规格属性
8 -> unit TEXT,             -- 单位属性
9 -> salesprice DECIMAL(10,2) -- 价格属性
10 -> );
11 Query OK, 0 rows affected (0.07 sec)
```

这样，我们就完成了 ER 模型到 MySQL 数据表的转换。

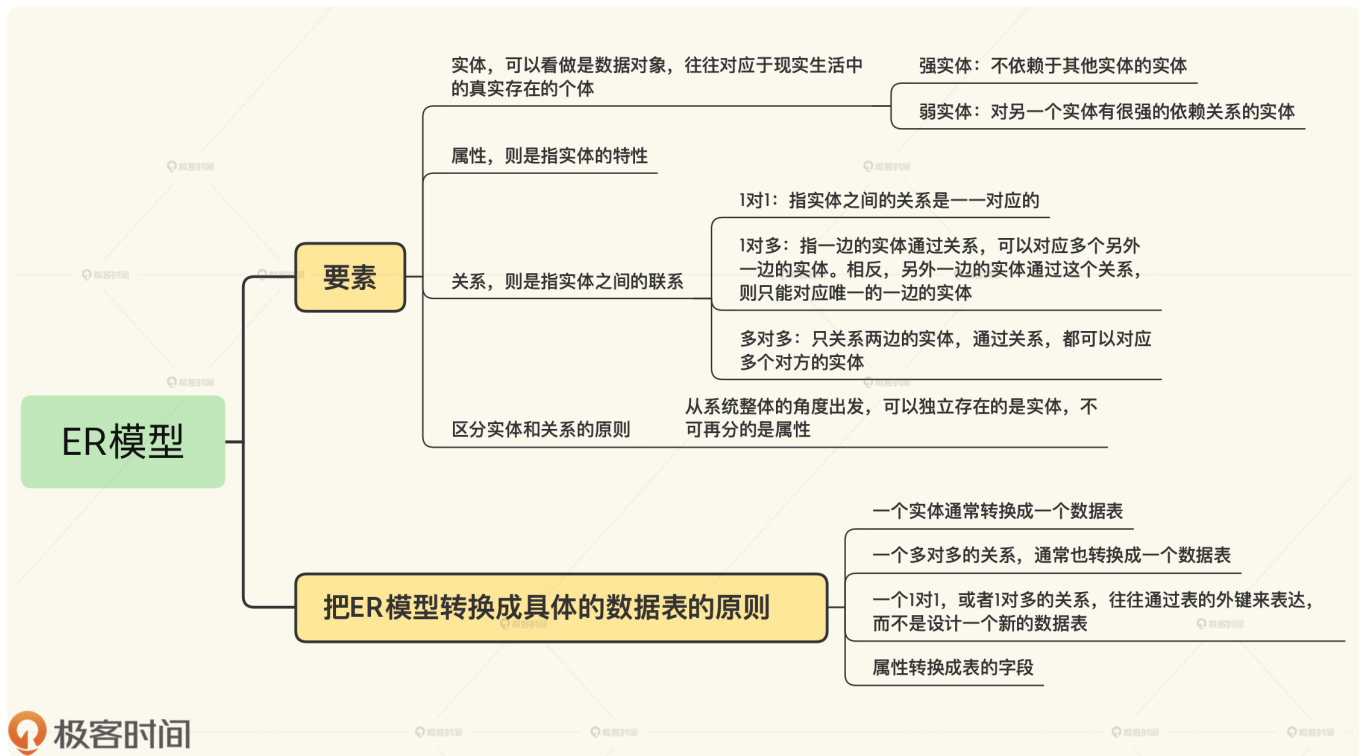
到这里，我们通过创建超市项目业务流程的 ER 模型，再把 ER 模型转换成具体的数据表的过程，完成了利用 ER 模型设计超市项目数据库的工作。

其实，任何一个基于数据库的应用项目，都可以通过这种先建立 ER 模型，再转换成数据表的方式，完成数据库的设计工作。ER 模型是一种工具。创建 ER 模型不是目的，目的是把业务逻辑梳理清楚，设计出优秀的数据库。我建议你不是为了建模而建模，要利用创建 ER 模型的过程来整理思路，这样创建 ER 模型才有意义。

## 总结

今天，我们学习了通过绘制 ER 模型图理清业务逻辑，以及怎么把 ER 模型转换成 MySQL 数据表，最终完成项目数据库设计。

这节课的知识点比较多，我用一张图来帮你回顾下重点。



最后，我还想再提醒你一下，ER 模型看起来比较麻烦，但是对我们把控项目整体非常重要。如果你只是开发一个小应用，或许简单设计几个表够用了，一旦要设计有一定规模的应用，在项目的初始阶段，建立完整的 ER 模型就非常关键了。开发应用项目的实质，其实就是建模。胸中有丘壑，才能下笔如有神。道理其实是一样的。

## 思考题

超市经营者每个月都要进行库房盘点，也就是在一个月的最后一天的营业结束之后，所有的员工一起把库房里的货品都数一遍，然后跟电脑上的库存比对，查看库存损耗。

我想请你思考一下，在这个业务模块中，涉及了哪些实体、属性和关系？另外，请你设计一下 ER 模型，通过它来整理一下数据库设计的思路。

欢迎在留言区写下你的思考和答案，我们一起交流讨论。如果你觉得今天的内容对你有所帮助，也欢迎你把它分享给你的朋友或同事，我们下节课见。

提建议



更多课程推荐

# 深入浅出计算机组成原理

带你掌握计算机体系全貌

徐文浩

bothub 创始人



涨价倒计时 🕒

今日订阅 **¥89**，5月12日涨价至 **¥199**

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 22 | 范式：如何消除冗余和高效存取？

下一篇 特别发送（一） | 经典面试题讲解第一弹

## 精选留言

💬 写留言

由作者筛选后的优质留言将会公开显示，欢迎踊跃留言。