Android数据库高手秘籍(四)--使用LitePal建立表关联

2015-09-08 安卓应用频道

(点击上方蓝字，可快速关注我们)

> 来源：郭霖
> 网址：http://blog.csdn.net/guolin_blog/article/details/39207945

目前我们已经对LitePal的用法有了一定了解，学会了使用LitePal来创建表和升级表的方式，那么今天就让我们一起继续进阶，探究一下如何使用LitePal来建立表与表之间的关联关系。还没有看过前一篇文章的朋友建议先去参考Android数据库高手秘籍(三)--使用LitePal升级表。

LitePal的项目地址是：https://github.com/LitePalFramework/LitePal

关联关系的基础知识

喜欢把所有的代码都写在一个类里的程序员肯定是个新手。没错，任何一个像样的程序都不可能仅仅只有一个类的，同样地，任何一个像样的数据库也不可能仅仅只有一张表。我们都知道，在面向对象的编程语言中，多个类之间可以相互关联引用，共同完成某项功能。那么在数据库当中，多个表之间可以相互关联吗？当然可以！只不过表与表之间的关联关系要比对象之间的关联关系复杂一些，也更加难懂，但是作为数据库的基本功，还是应该了解清楚的，那么我们就先来学习一下数据库表关联的基础知识。

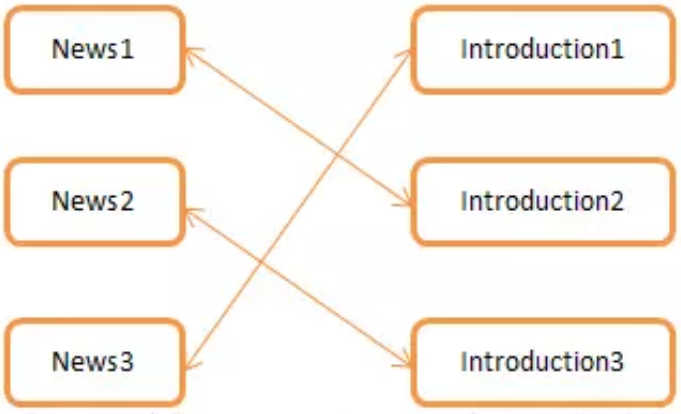
表与表之间的关联关系一共有三种类型，一对一、多对一、和多对多，下面我们分别对这三种类型展开进行讨论。

一对一

表示两个表中的数据必须是一一对应的关系。这种场景其实并不是很常见，我们还是通过例子来直观地体会一下，例子仍然是在之前文章的基础上展开的。

现在我们已经创建好了news这张表，里面主要记录了新闻的标题和内容，那么除了标题和内容之外，有些新闻还可能带有一些导语和摘要，我们把这两个字段放在一张introduction表中，作为新闻的简介。那么很显然，news表和introduction表就是一对一的关系了，因为一条新闻只能对应一个简介，一个简介也只能属于一条新闻。它们之间的对应关系大概如下图

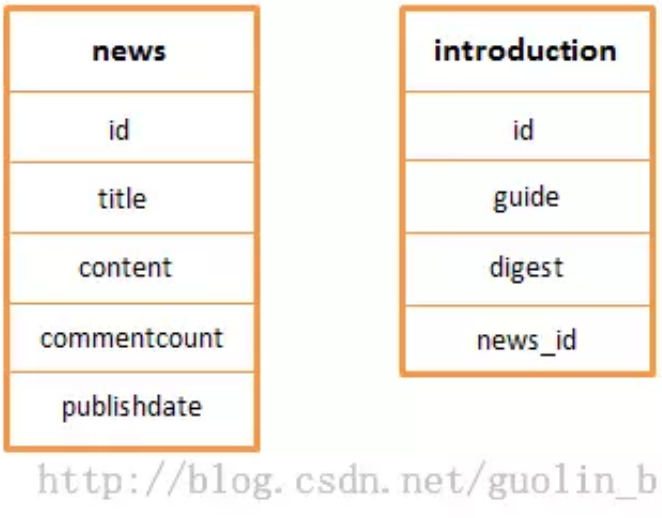
描述的一样：



可以看到，News1对应了Introduction2，News2对应了Introduction3，News3对应了Introduction1，但不管怎么样，它们都是一对一的关系。

那么这种一对一的关系，在编程语言中该怎么体现出来呢？相信熟悉面向对象设计的你，一定很轻松就能想出来吧，只需要在News类中持有一个Introduction类的引用，然后在Introduction类中也持有一个News类的引用，这样它们之间自然就是一对一的关系了。

没错，对象之间的一对一关系非常简单易懂，那么难点就在于，如何在数据库表中建立这样的一对一关系了。由于数据库并不像面向对象的语言一样支持相互引用，如果想让两张表之间建立一对一的关系，一般就只能通过外键的方式来实现了。因此，一对一关系的表结构就可以这样设计：



请注意，introduction表中有一个news_id列，这是一个外键列，里面应该存放一个具体的新闻id，这样一条introduction就能对应一条news，也就实现一对一的关系了，如下图所示：

news 表

id	……
1	……
2	……
3	……

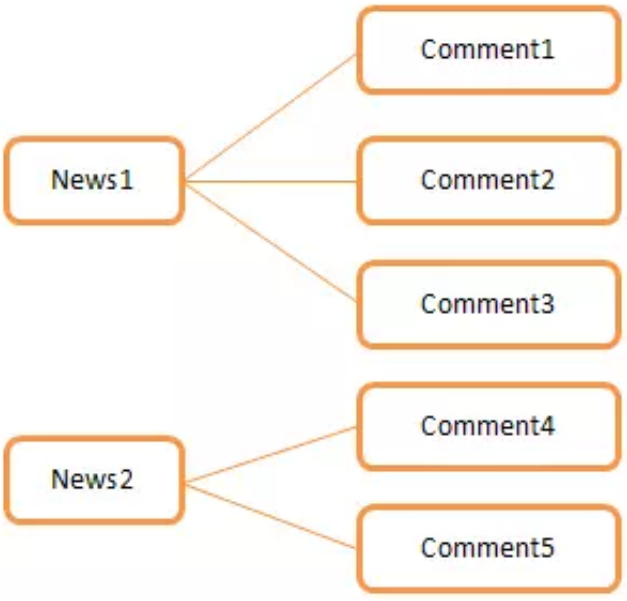
introduction 表

id	……	news_id
1	……	2
2	……	3
3	……	1

由此我们就能够看出，id为1的introduction对应着id为2的news，id为2的introduction对应着id为3的news，id为3的introduction对应着id为1的news。需要注意的是，一对一的关系并没有强制要求外键必须加在哪一张表上，你可以在introduction表中加一个news_id作为外键，也可以在news表中加一个introduction_id作为外键，不管使用哪一种，都可以表示出它们是一对一的关联关系。

多对一

表示一张表中的数据可以对应另一张表中的多条数据。这种场景比起一对一关系就要常见太多了，在我们平时的开发工作中多对一关系真的是比比皆是。比如说现在我们的数据库中有一个news表，还有一个comment表，它们两个之间就是典型的多对一关系，一条新闻可以有很多条评论，但是一条评论只能是属于一条新闻的。它们的关系如下图所示：



而这种多对一的关系在编程语言中是非常容易体现出来的，比如Java中就有专门集合类，如List、Set等，使用它们的话就能轻松简单地在对象之间建立多对一的关系，我们稍后就会看到。那么，这里的难点仍然是在数据库表中如何建立这样的多对一关系。现在说难点其实已经不难了，因为前面我们已经学会了一对一关系的建立方法，而多对一也是类似的。没错，数据库表中多对一的关系仍然是通过外键来建立的，只不过一对一的时候外键加在哪一张表上都可以，但多对一的时候关键必须要加在多方的表中。因此，多对一关系的表结构就可以这样设计：

在comment表中有一个news_id列，这是一个外键列，里面应该存放一个具体的新闻id，并且允许多条comment都存放同一个新闻id，这样一条评论就只能对应一条新闻，但一条新闻却可以有多条评论，也就实现多对一的关系了，如下图所示：

news 表

id	……
1	……
2	……

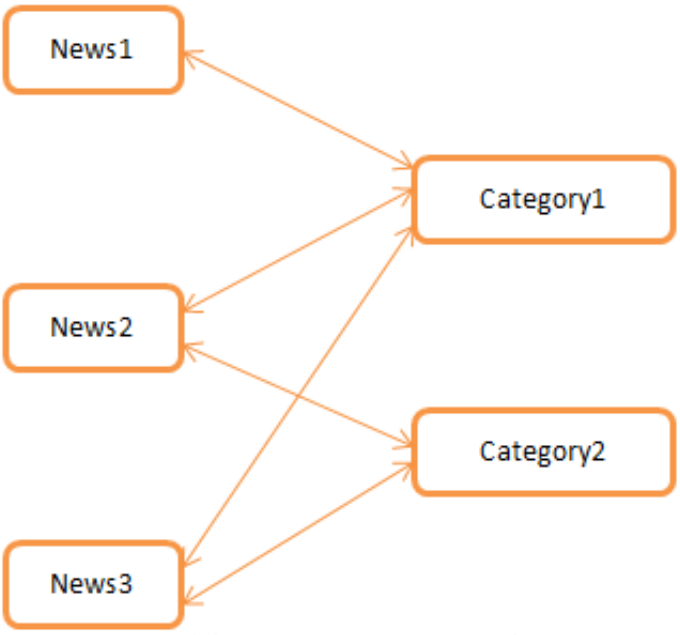
comment 表

id	……	news_id
1	……	1
2	……	1
3	……	1
4	……	2
5	……	2

由此我们就可以看出，id为1、2、3的三条评论是属于第一条新闻的，而id为4、5的两条评论是属于第二条新闻的。

多对多

表示两张关联表中的数据都可以对应另一张表中的多条数据。这种场景也不算是很常见，但比一对一关系要稍微更加常用一些。举个例子，我们都知道新闻网站是会将新闻进行种类划分的，这样用户就可以选择自己喜欢的那一类新闻进行浏览，比如说网易新闻中就会有头条、科技、娱乐、手机等等种类。每个种类下面当然都会有许多条新闻，而一条新闻也可能是属于多个种类的，比如iPhone6发布的新闻既可以属于手机种类，也可以属于科技种类，甚至还可以上头条。因此，新闻和种类之间就是一种多对多的关系，如下图所示：



可以看到，News1是属于Category1的，而News2和News3都是既属于Category1也属于Category2，如此复杂的关联关系该如何表示呢？在面向对象的编程语言中一切都是那么的简单，只需要在News类中使用集合类声明拥有多个Category，然后在Category类中也使用集合类声明拥有多个News就可以了，我们稍后就会看到。而难点仍然是留在了数据库上，两张表之间如何建立多对多的关联关系呢，还是用外键吗？肯定不行了，多对多的情况只能是借助中间表来完成了。也就是说，我们需要多建立一张表，这张表没什么其它作用，就是为了存放news表和category表之间的关联关系的，如下图所示：

注意这里我们建立一张名为category_news的中间表，中间表的命名并没有什么强制性的约束，但一个良好的命名规范可以让你一眼就明白这张表是用来做什么的。中间表里面只有两列，而且也只需要有两列，分别是news表的外键和category表的外键，在这里存放新闻和种类相应的id，就可以让它们之间建立关联关系了，如下图所示：

news 表

id	……
1	……
2	……
3	……

category_news 表

news_id	category_id
1	1
2	1
2	2
3	1
3	2

category 表

id	……
1	……
2	……

由此我们就可以看出，第一条新闻是属于第一个种类的，而第二和第三条新闻，则既属于第一个种类，也属于第二个种类。反过来也可以这样看，第一个种类下面有第一、第二、第三这三条新闻，而第二个种类下面只有第二、第三这两条新闻。不管怎么看，多对多的关系都是成立的。

好了，三种关联关系都讲完了，那我们来简单总结一下吧。虽说上面介绍了花了很大的篇幅讲解数据库的表关联知识，但其实最后的结论是非常简单的，大家可以当成口诀一样背下来。即一对一关联的实现方式是用外键，多对一关联的实现方式也是用外键，多对多关联的实现方式是用中间表。记下了这个口诀，在很多数据库设计的时候，你都可以发挥得更加游刃有余。

使用LitePal建立表关联

虽说口诀就是这个样子，但牵扯到表关联的时候毕竟增加了建表的难度，建表语句会更加复杂，你也需要格外地小心以防止出现什么错误。因此，使用LitePal来自动建立表关联又是一个非常不错的选择，我们不需要关心什么外键、中间表等实现的细节，只需要在对象中声明好它们相互之间的引用关系，LitePal就会自动在数据库表之间建立好相应的关联关系了，下面我们就来尝试一下吧。

首先确定一下一共涉及到了哪些实体类，News和Comment，这两个类我们在前两篇文章中就已经建好了，然后还需要有Introduction和Category这两个类，新建Introduction类，代码如下所示：

```
public class Introduction {

    private int id;
```

```
    private String guide;

    private String digest;

    // 自动生成get、set方法
}
```

接着新建Category类，代码如下所示：

```
public class Category {

    private int id;

    private String name;

    // 自动生成get、set方法
}
```

现在四个类都已经建好了，但目前它们都还是各自独立的，互相之间没有任何联系，那么我们现在就开始用极为简单易懂的方式来给它们建立关联吧。首先，News和Introduction是一对一的关系，那就可以在News类中添加如下引用：

```
public class News {
    ...
    private Introduction introduction;

    // 自动生成get、set方法
}
```

就是这么简单，在News类中可以得到一个对应的Introduction的实例，那么它们之间就是一对一关系了。

接着Comment和News是多对一的关系，因此News中应该包含多个Comment，而Comment中应该只有一个News，所以就可以这样写：

```
public class News {
    ...
    private Introduction introduction;
```

```
    private List<Comment> commentList = new ArrayList<Comment>();

    // 自动生成get、set方法
}
```

先使用一个泛型为Comment的List集合来表示News中包含多个Comment，然后修改Comment类的代码，如下所示：

```
public class Comment {
    ...
    private News news;

    // 自动生成get、set方法
}
```

Comment类中声明了一个News的实例，这样就清楚地表示出了News中可以包含多个Comment，而Comment中只能有一个News，也就是多对一的关系了。

最后News和Category是多对多的关系，相信聪明的你一定已经知道该怎么写了。News中可以包含多个Category，所以仍然应该使用List集合来表示：

```
public class News {
    ...
    private Introduction introduction;

    private List<Comment> commentList = new ArrayList<Comment>();

    private List<Category> categoryList = new ArrayList<Category>();

    // 自动生成get、set方法
}
```

而Category中也可以包含多个News，因此Category类也应该使用相同的写法，如下所示：

```
public class Category {
    ...
    private List<News> newsList = new ArrayList<News>();

    // 自动生成get、set方法
```

```
}
```

这样就清楚地表达出它们之间是多对多的关联了。
关联关系都声明好了之后，我们只需要将所有的实体类都添加到映射列表当中，并将数据库版本号加1就可以了。修改litepal.xml的代码，如下所示：

```
<?xml version="1.0" encoding="utf-8"?>
<litepal>
<dbname value="demo" ></dbname>

<version value="4" ></version>

<list>
<mapping class="com.example.databasetest.model.News"></mapping>
<mapping class="com.example.databasetest.model.Comment"></mapping>
<mapping class="com.example.databasetest.model.Introduction"></mapping>
<mapping class="com.example.databasetest.model.Category"></mapping>
</list>
</litepal>
```

基本上到这里就可以轻松地说结束了，现在只需要任意操作一下数据库，表之间的关联关系就将会自动建立，比如说调用一下Connector.getDatabase()方法。

下面我们来验证一下吧，输入.table命令查看一下当前数据库中的表，如下所示：

OK，news、comment、category、introduction这几张表全都有了，除此之外还有一张category_news中间表。那我们要来一一检查一下了，先查看一下introduction表的结构吧，如下所示：

可以看到，多了一个news_id列，说明introduction表和news表之间的一对一关系已经建立好了。

然后再检查一下comment表的结构，如下所示：

OK，comment表中也有一个news_id的列，那么comment表和news表之间的多对一关系也已经建立好了。

最后检查一下category_news这张中间表的结构，如下所示：

一共只有两列，一列是news_id，一列是category_id，分别对应着两张表的外键，这样news表和category表的多对多关系也建立好了。

借助LitePal的帮助，即使你并不熟悉数据库的表关联设计，只要你会面向对象编程，都可以轻松地将表与表之间的关联建立起来。创建表、升级表、表关联，这就是LitePal在数据库表管理方面给我们带来的巨大便利，相信大家都能体会到它的魅力所在了。那么到这里为止，我们就把使用LitePal进行表管理的知识全部学完了，从下一篇文章开始，我将会讲解如何使用LitePal进行CRUD的操作。感兴趣的朋友请继续阅读 Android数据库高手秘籍(五)--LitePal的存储操作 。

阅读原文

微信扫一扫
关注该公众号