### User表

用户id（user\_id 主键）

用户身份（user\_identity）

用户昵称(user\_virtual\_name)

用户密码(user\_password)

手机号(user\_phone)

地址(user\_address)

头像(user\_image)

性别(user\_sex)

身份证号(user\_card\_number)

服务评分(user\_grade)

微信（user\_wechat）

支付宝(user\_alipay)

微博（user\_blog）

用户是否处于订单中(user\_flag)

//我的订单(user\_order\_id 外键，参考order表的order\_id),

//我的狗(user\_dog\_id 外键，参考dog表的dog\_id),

### Order表

订单id(order\_id 主键)

订单委托人(order\_client\_id 参考user表的user\_id)

订单被委托人(order\_ mandatary\_id 参考user表的user\_id)

照顾的狗(order\_dog\_id 参考dog表中的 dog\_id)

订单处理状态(order\_state)

评价(order\_evaluate)

订单的开始日期(order\_start\_time)

订单的结束日期(order\_end\_time)

单次遛狗时间(time)

订单的金额（money）

### Dog表

狗的id( dog\_id 主键)

小狗的名字（dog\_name）

狗的种类(dog\_type)

狗的主人(dog\_user\_id 参考user表的user\_id)

照片(dog\_image)

狗狗的过敏原（allergen）

狗狗喜欢的食物(dog\_like\_food)

狗狗不喜欢的食物 (dog\_bad\_food)

### Dynamic表 （动态）

动态的id(dynamic\_id 主键)

动态的发起人(dynamic\_user\_id 参考user表的user\_id)

动态的发布时间(dynamic\_time)

动态的文字内容(dynamic\_content)

动态的图片（dynamic\_image）

//初步定义为string类别的 多个图片的时候，将他们的路径地址拼接起来，拼接的地方加上分隔符，读取的时候先用分隔符区分开

//动态的评论(dynamic\_ comment\_id 参考comment表中的comment\_id)

### Forward表（转发）

转发的id

转发的时间（forwardTime）

转发的人（User）

转发的动态（Dynamic）

### Fabulous表（赞）

赞的id

赞的时间（fabulousTime）

赞的动态（Dynamic）

点赞的人（User）

### Address表（地址）

地址id

用户（user）

地址内容（address）

### Comment表（评论）

评论的id(comment\_id)

评论人(comment\_user\_id 参考user表的user\_id)

评论的时间（time）

评论的内容(comment\_content)

动态的id(comment\_dynamic\_id 参考dynamic表的dynamic\_id

/\*一、数据库设计的考虑过程

1.android连接数据库的方式有两种，第一种是通过连接服务器，再由服务器读取数据库来实现数据的增删改查，这也是我们常用的方式。第二种方式是android直接连接数据库，这种方式非常耗手机内存，而且容易被反编译造成安全隐患，所以在实际项目中不推荐使用。所以我们采用**连接服务器的方式来进行数据库的连接**。

2.读取到的数据如何显示到APP上面，由于执行网络操作是耗时操作并且android4.0要求必须把和网络相关的操作放到子线程中。

3. 用到的知识有Android和JavaEE并且之间通过JSON传送数据。

4. 网路上具有一定局限性，因为用的服务器是电脑上的tomcat，所以手机和电脑上需要在一个局域网内（电脑连手机热点或者手机连电脑热点）才能够进行数据之间的传输。可以单独将写一个类来获取本地IP地址。

4. 高内聚低耦合。

5. 数据库文件（sql）运行的时候是在数据库上点击运行的（所以需要提前建立一个与导入的数据库相同的名字的数据库）。

二、数据库设计过程\*/

数据库采用第三方服务器（Bmob后端云）来进行数据的操作。这样采用第三方的类与接口可以不用写太多的后台代码。