

# 说明文档

## 说明文档

- 1. 配置环境
- 2. 数据库配置说明
- 3. ER图
- 4. 初始数据
- 5. 操作说明
  - 5.0
  - 5.1 查询所有套餐
  - 5.2 某个用户进行套餐的查询（包括历史记录）
  - 5.3 某个用户的月账单
  - 5.4 通话情况下的资费生成
  - 5.5 使用本地情况下的资费生成
  - 5.6 订购
  - 5.7 立即退订
  - 5.8 下月退订
- 6. 注意事项
- 7. 优化

## 1. 配置环境

项目	版本号
Java	1.8
MySQL	8.0.12
Mybatis	3.4.6

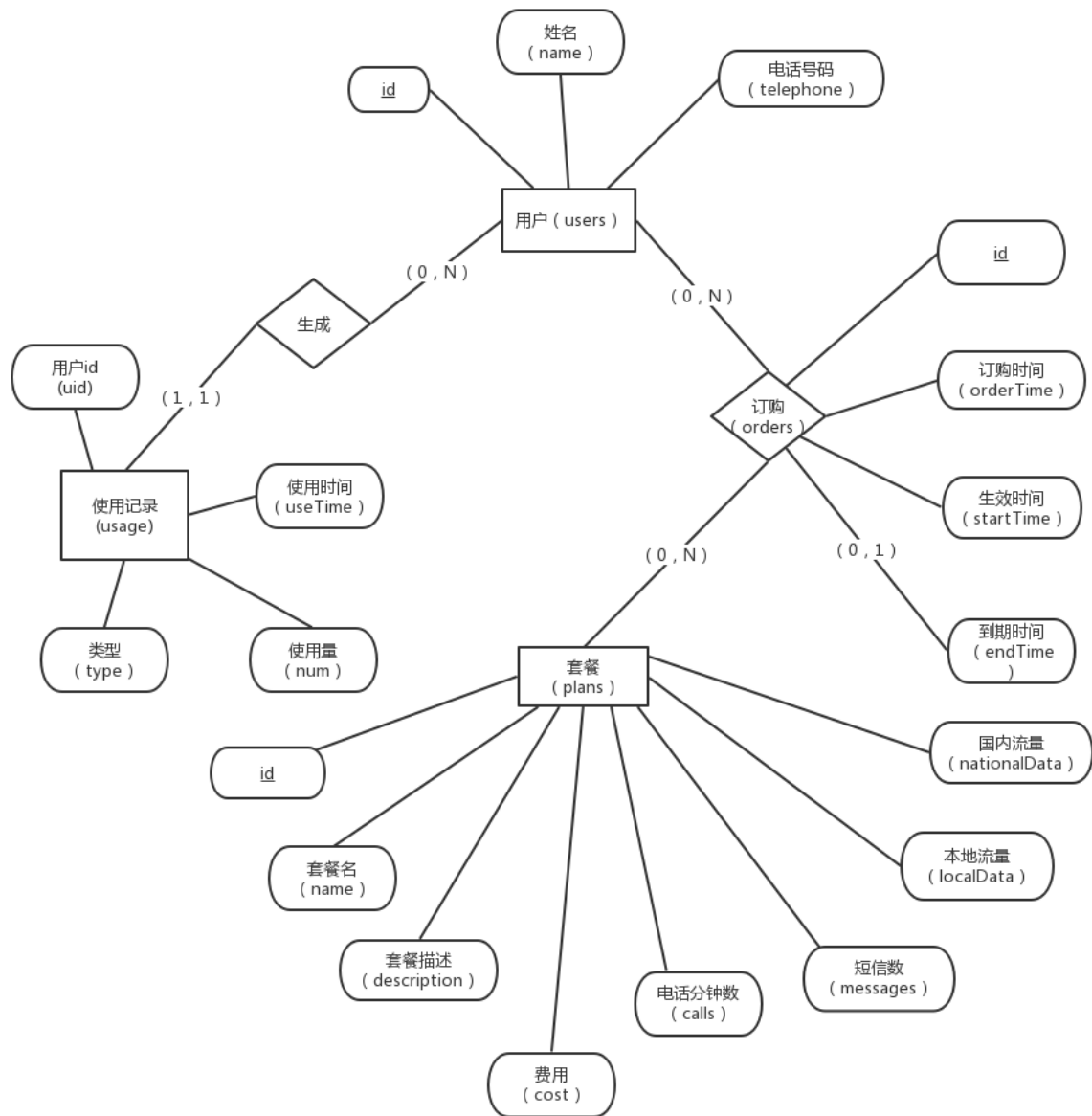
## 2. 数据库配置说明

数据库配置文件为项目文件夹resources下的jdbc.yml

```
driver: com.mysql.cj.jdbc.Driver
url: jdbc:mysql://localhost:3306/mobile?characterEncoding=utf8&useSSL=false&serverTimezone=Asia/Shanghai
username: root
password: 123456
```

其中数据库名为mobile、系统时间使用Asia/Shanghai、用户名为root、密码为123456，可根据自身情况进行更改

## 3. ER图



## 4. 初始数据

(电话基本单位为分、流量为M)

1. 10个用户(使用Faker生成的)

2. 5个套餐

1. 话费套餐：月功能费20元，最多可拨打100分钟电话，超出时间按照0.5元/分钟计费。
2. 短信套餐：月功能费10元，最多可发送200条短信，超出条数按0.1元/条计费。
3. 本地流量套餐：月功能费20元，最多可获得2G流量，仅在本地使用，超出流量按2元/M计费。
4. 国内流量套餐：月功能费30元，最多可获得2G流量，超出流量按5元/M计费。
5. 大王卡套餐：月功能费68元，最多可拨打100分钟电话，超出时间按照0.5元/分钟计费；最多可发送200条短信，超出条数按0.1元/条计费；最多可获得2G流量，仅在本地使用，超出流量按2元/M计费；最多可获得2G流量，超出流量按5元/M计费。

对应方法仍在Service中保留，为

```
addUser(User user);

addPlan(Plan plan);
```

可自行增加数据

3. 为测试数据，还进行了以下初始化操作

1. 用户1订购了1、3、5套餐，1、5立即生效，3下月生效。且发了7条短信，用了5G本地流量，打了13分钟电话
2. 用户2订购了1，2套餐，立即生效；后又取消了2（次月生效）。使用了3分钟电话
3. 用户3未订购套餐，且打了7分钟电话、使用了64M本地流量和124M国内流量

## 5. 操作说明

### 5.0

由service包下的MainService接口提供服务

```
public interface MainService {

    /**
     * 新增用户
     *
     * @param user 用户
     */
    void addUser(User user);

    /**
     * 新增套餐
     *
     * @param plan 套餐
     */
    void addPlan(Plan plan);

    /**
     * @return 所有套餐
     */
    List<Plan> getPlans();

    /**
     * @param uid 用户id
     * @return 用户的订购记录
     */
    List<OrderVO> getOrders(int uid);

    /**
     * 订购套餐
     *
     * @param uid 用户id
```

```

    * @param pid          套餐id
    * @param nowTime      现在时间
    * @param immediately 是否立即订购
    * @return 订单id
    */
    int orderPlan(int uid, int pid, LocalDateTime nowTime, boolean
immediately);

/**
 * 取消订单
 *
 * @param oid          订单id
 * @param nowTime      现在时间
 * @param immediately 是否立即退订
 */
void cancelPlan(int oid, LocalDateTime nowTime, boolean immediately);

/**
 * 获取用户资费信息
 *
 * @param uid  用户id
 * @param now  现在时间
 * @param type 数据类型
 * @return 资费账单
 */
Expense getExpense(int uid, LocalDateTime now, Type type);

/**
 * 获取用户月账单
 *
 * @param uid  用户id
 * @param year 年
 * @param month 月
 * @return 月账单
 */
Bill getBill(int uid, int year, int month);

/**
 * 打电话
 *
 * @param uid          用户id
 * @param startTime 开始时间
 * @param minutes      时长
 */
void call(int uid, LocalDateTime startTime, int minutes);

/**
 * 发信息
 *

```

```
    * @param uid      用户id
    * @param startTime 开始时间
    */
    void sendMessage(int uid, LocalDateTime startTime);

    /**
     * 使用流量
     *
     * @param uid      用户id
     * @param startTime 开始时间
     * @param num      使用量
     * @param local    是否本地
     */
    void useData(int uid, LocalDateTime startTime, int num, boolean local);
}
```

## 5.1 查询所有套餐

设计：直接查plan表

```
select * from plans;
```

对应方法：

```
List<Plan> getPlans();
```

截图：

## 1. 查询所有套餐

```
[  
套餐名：话费套餐  
套餐费用：20.00  
  
,  
套餐名：短信套餐  
套餐费用：10.00  
  
,  
套餐名：本地流量套餐  
套餐费用：20.00  
  
,  
套餐名：国内流量套餐  
套餐费用：30.00  
  
,  
套餐名：大王卡套餐  
套餐费用：68.00  
]
```

开始时间：2018-10-30T21:16:10.226

结束时间：2018-10-30T21:16:12.198

运行：1972ms

## 5.2 某个用户进行套餐的查询（包括历史记录）

设计：先从orders表查该用户的所有订单，再根据pid与plans表连接

```
select o.id, p.*, orderTime, startTime, endTime  
from orders o,  
      plans p  
where uid = #{uid}  
      and p.id = o.pid  
order by orderTime
```

对应方法：

```
List<OrderVO> getOrders(int uid);
```

截图：

## 2.1 查询用户1套餐历史记录

[

套餐名：话费套餐

套餐费用：20.00

订购时间：2018-10-30T21:13:18

生效时间：2018-10-30T21:13:18

到期时间：

,

套餐名：本地流量套餐

套餐费用：20.00

订购时间：2018-10-30T21:14:18

生效时间：2018-11-01T00:00

到期时间：

,

套餐名：大王卡套餐

套餐费用：68.00

订购时间：2018-10-30T21:15:18

生效时间：2018-10-30T21:15:18

到期时间：

]

开始时间：2018-10-30T21:16:12.202

结束时间：2018-10-30T21:16:12.284

运行：82ms

## 2.2 查询用户2套餐历史记录

[

套餐名：话费套餐

套餐费用：20.00

订购时间：2018-10-30T21:13:18

生效时间：2018-10-30T21:13:18

到期时间：

,

套餐名：短信套餐

套餐费用：10.00

订购时间：2018-10-30T21:14:18

生效时间：2018-10-30T21:14:18

到期时间：2018-11-01T00:00

]

开始时间：2018-10-30T21:16:12.284

结束时间：2018-10-30T21:16:12.381

运行：97ms

## 5.3 某个用户的月账单

设计：首先确定时间段，该月第一天零点至该月最后一天的最后一刻（以月为单位），找出这个时间段已订购的可用的套餐，再找出在该月的各种数据的使用情况（以使用开始时间在该时段为基础）。

可用套餐计算套餐费用

而各种数据在减去套餐对应的数据量后若还有溢出，剩余部分进入基础资费的计算

值得注意的是，（本着人性化的态度），本系统中本地流量先使用套餐中的本地流量数，如果不够，再去看套餐中的国内流量是否有剩余（即套餐中的国内流量与实际使用的国内流量比较后的剩余）

同样之后的资费查询同样使用该算法

判断可用的套餐：



```
select *
from plans p
where p.id in (select pid
               from orders o
               where o.uid = #{uid}
                  and startTime < #{endTime}
                  and (endTime is null or endTime >= #{endTime}))
```

查询时间段内数据使用情况:

```
select *
from `usage`
where uid =#{uid}
      and type = #{type}
      and useTime >=#{startTime}
      and useTime < #{endTime}
```

代码:

```
public Bill getBill(int uid, int year, int month) {
    try (SqlSession sqlSession = MybatisUtil.getSqlSession()) {
        Bill bill = new Bill();

        bill.setYear(year);
        bill.setMonth(month);

        MainDao mainDao = sqlSession.getMapper(MainDao.class);

        bill.setTelephone(mainDao.findUserById(uid).getTelephone());

        List<Plan> activePlans = mainDao.findActivePlans(uid,
            DateUtil.getNextMonthFirstDayZero(year, month));

        bill.setPlanCosts(activePlans
            .stream()
            .collect(Collectors.toMap(Plan::getName,
                Plan::getCost)));

        int planCalls =
            activePlans.stream().mapToInt(Plan::getCalls).sum(),
            planMessages =
            activePlans.stream().mapToInt(Plan::getMessages).sum(),
            planLocalData =
            activePlans.stream().mapToInt(Plan::getLocalData).sum(),
            planNationalData =
            activePlans.stream().mapToInt(Plan::getNationalData).sum();
```

```

        LocalDateTime startTime =
DateUtil.getThisMonthFirstDayZero(year, month);
        LocalDateTime endTime = DateUtil.getNextMonthFirstDayZero(year,
month);

        long usedCalls = mainDao.findUsages(uid, Type.CALL, startTime,
endTime).stream().mapToLong(Usage::getNum).sum();
        long usedMessages = mainDao.findUsages(uid, Type.MESSAGE,
startTime, endTime).stream().mapToLong(Usage::getNum).sum();
        long usedLocalData = mainDao.findUsages(uid, Type.LOCAL_DATA,
startTime, endTime).stream().mapToLong(Usage::getNum).sum();
        long usedNationalData = mainDao.findUsages(uid,
Type.NATIONAL_DATA, startTime,
endTime).stream().mapToLong(Usage::getNum).sum();

        Map<String, BigDecimal> basicCosts = new HashMap<>();

        if (usedCalls > planCalls)
            basicCosts.put("通话费用",
BasicUtil.calculateBasicCallCost(usedCalls - planCalls));
        if (usedMessages > planMessages)
            basicCosts.put("短信费用",
BasicUtil.calculateBasicMessageCost(usedMessages - planMessages));

        long remainData = 0;

        if (usedNationalData > planNationalData)
            basicCosts.put("国内流量费用",
BasicUtil.calculateBasicNationalDataCost(usedNationalData -
planNationalData));
        else remainData = planNationalData - usedNationalData;

        if (usedLocalData > planLocalData + remainData)
            basicCosts.put("本地流量费用",
BasicUtil.calculateBasicLocalDataCost(usedLocalData - planLocalData -
remainData));

        bill.setBasicCosts(basicCosts);
        return bill;
    }
}

```

截图：

### 3.1 查询用户1的10月账单

2018 年 10 月

您好! 17118860822

本月账单 (单位: 元)

合计: 2136.00

套餐费用:

话费套餐—20.00

大王卡套餐—68.00

基础费用:

本地流量费用—2048.00

|

开始时间: 2018-10-30T21:22:25.222

结束时间: 2018-10-30T21:22:26.242

运行: 1020ms

### 3.2 查询用户2的10月账单

2018 年 10 月

您好! 14717299635

本月账单 (单位: 元)

合计: 30.00

套餐费用:

话费套餐—20.00

短信套餐—10.00

基础费用:

无

开始时间: 2018-10-30T21:22:26.249

结束时间: 2018-10-30T21:22:26.329

运行: 80ms

### 3.3 查询用户3的10月账单

2018 年 10 月

您好! 15881283916

本月账单 (单位: 元)

合计: 751.50

套餐费用:

无

基础费用:

国内流量费用—620.00

本地流量费用—128.00

通话费用—3.50

开始时间: 2018-10-30T21:22:26.330

结束时间: 2018-10-30T21:22:26.589

运行: 259ms

### 5.4 通话情况下的资费生成

设计: 因为有立即退订的存在, 且考虑到立即退订按百分比退款的不实际, 选择方案: 若立即退订, 则该套餐不参与该月数据费用的计算。所以在数据使用时也只记录使用时间和使用量, 之后查询时再来计算。

算法与 5.3 类型, 短信、国内流量与通话情况相同, 不再重复描述

对应方法:

```
Expense getExpense(int uid, LocalDateTime now, Type type);
```

截图:

#### 4.1 查询用户1在通话情况下的资费生成

2018 年 10 月

资费账单 (单位: 元)

套餐量: 200

使用量: 13

基础费用: 0.00

开始时间: 2018-10-30T21:53:25.235

结束时间: 2018-10-30T21:53:27.187

运行: 1952ms

#### 4.2 查询用户2在通话情况下的资费生成

2018 年 10 月

资费账单 (单位: 元)

套餐量: 100

使用量: 3

基础费用: 0.00

开始时间: 2018-10-30T21:53:27.191

结束时间: 2018-10-30T21:53:27.298

运行: 107ms

## 4.3 查询用户3在通话情况下的资费生成

2018 年 10 月

资费账单 (单位: 元)

套餐量: 0

使用量: 7

基础费用: 3.50

开始时间: 2018-10-30T21:53:27.301

结束时间: 2018-10-30T21:53:27.381

运行: 80ms

## 5.5 使用本地情况下的资费生成

设计: 套餐量会加上国内流量未使用的部分, 其余与5.3相同。

截图:

5.1 查询用户1在使用本地流量情况下的资费生成  
2018 年 10 月  
资费账单 (单位: 元)  
套餐量: 4096  
使用量: 5120  
基础费用: 2048.00

开始时间: 2018-10-30T22:08:05.310  
结束时间: 2018-10-30T22:08:06.539  
运行: 1229ms

5.2 查询用户2在使用本地流量情况下的资费生成  
2018 年 10 月  
资费账单 (单位: 元)  
套餐量: 0  
使用量: 0  
基础费用: 0.00

开始时间: 2018-10-30T22:08:06.544  
结束时间: 2018-10-30T22:08:06.657  
运行: 113ms



5.3 查询用户3在使用本地流量情况下的资费生成  
2018 年 10 月  
资费账单 (单位: 元)  
套餐量: 0  
使用量: 64  
基础费用: 128.00

开始时间: 2018-10-30T22:08:06.658  
结束时间: 2018-10-30T22:08:06.718  
运行: 60ms

5.4 5.5代码:

```
public Expense getExpense(int uid, LocalDateTime now, Type type) {
    try (SqlSession sqlSession = MybatisUtil.getSqlSession()) {
        MainDao mainDao = sqlSession.getMapper(MainDao.class);

        LocalDateTime startTime =
            DateUtil.getThisMonthFirstDayZero(now);
        LocalDateTime endTime = DateUtil.getNextMonthFirstDayZero(now);

        if (type == Type.CALL || type == Type.MESSAGE || type ==
            Type.NATIONAL_DATA) {

            long useTotal = mainDao.findUsages(uid, type, startTime,
                endTime).stream().mapToLong(Usage::getNum).sum();

            long planTotal = mainDao.findActivePlans(uid,
                endTime).stream().mapToLong(x -> x.getTypeNum(type)).sum();

            BigDecimal basicCost = BigDecimal.valueOf(0, 2);

            if (useTotal > planTotal) {
                long difference = useTotal - planTotal;

                switch (type) {
                    case CALL:
                        basicCost =
                            BasicUtil.calculateBasicCallCost(difference);
                        break;
                    case MESSAGE:
```

```

        basicCost =
BasicUtil.calculateBasicMessageCost(difference);
        break;
        case NATIONAL_DATA:
            basicCost =
BasicUtil.calculateBasicNationalDataCost(difference);
            break;
    }
}

return new Expense(now.getYear(), now.getMonthValue(),
type, useTotal, planTotal, basicCost);
} else {
    long useLocalTotal = mainDao.findUsages(uid, type,
startTime, endTime).stream().mapToLong(Usage::getNum).sum();

    long useNationalTotal = mainDao.findUsages(uid,
Type.NATIONAL_DATA, startTime,
endTime).stream().mapToInt(Usage::getNum).sum();

    List<Plan> activePlan = mainDao.findActivePlans(uid,
endTime);

    long planNationalTotal = activePlan.stream().mapToLong(x ->
x.getTypeNum(Type.NATIONAL_DATA)).sum();

    long planLocalTotal = activePlan.stream().mapToLong(x ->
x.getTypeNum(Type.LOCAL_DATA)).sum();

    long nationRemain = (planNationalTotal > useNationalTotal)
? planNationalTotal - useNationalTotal : 0;

    BigDecimal basicCost = useLocalTotal > planLocalTotal +
nationRemain ? BasicUtil.calculateBasicLocalDataCost(useLocalTotal -
planLocalTotal - nationRemain) : BigDecimal.valueOf(0, 2);

    return new Expense(now.getYear(), now.getMonthValue(),
Type.LOCAL_DATA, useLocalTotal, planLocalTotal + nationRemain, basicCost);
}
}
}

```

## 5.6 订购

设计:

```

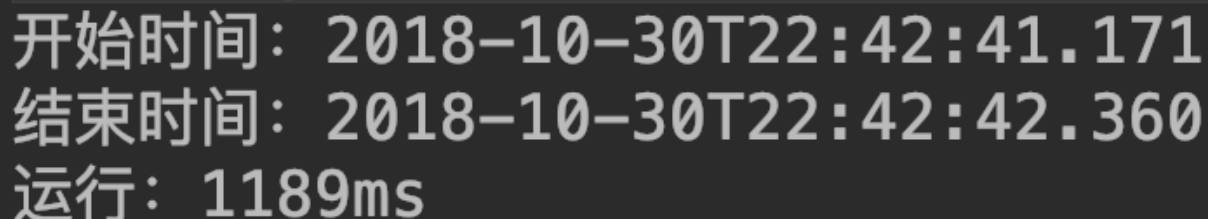
insert into orders(uid, pid, orderTime, startTime) values ({uid}, #{pid},
#{orderTime}, #{startTime})

```

对应方法：

```
int orderPlan(int uid, int pid, LocalDateTime nowTime, boolean  
immediately);
```

截图：



开始时间：2018-10-30T22:42:41.171  
结束时间：2018-10-30T22:42:42.360  
运行：1189ms

## 6.1 用户1立即订购本地流量套餐（通过历史记录和月账单查看）

[

套餐名：话费套餐

套餐费用：20.00

订购时间：2018-10-30T21:13:18

生效时间：2018-10-30T21:13:18

到期时间：

,

套餐名：本地流量套餐

套餐费用：20.00

订购时间：2018-10-30T21:14:18

生效时间：2018-11-01T00:00

到期时间：

,

套餐名：大王卡套餐

套餐费用：68.00

订购时间：2018-10-30T21:15:18

生效时间：2018-10-30T21:15:18

到期时间：

,

套餐名：本地流量套餐

套餐费用：20.00

订购时间：2018-10-30T22:42:41

生效时间：2018-10-30T22:42:41

到期时间：

2018 年 10 月  
您好! 17118860822  
本月账单 (单位: 元)  
合计: 108.00  
套餐费用:  
话费套餐—20.00  
本地流量套餐—20.00  
大王卡套餐—68.00  
基础费用:  
无

## 5.7 立即退订

设计:

```
update orders set endTime=#{endTime} where id=#{oid}
```

对应方法:

```
void cancelPlan(int oid, LocalDateTime nowTime, boolean immediately);
```

截图:

开始时间: 2018-10-30T23:04:08.305  
结束时间: 2018-10-30T23:04:10.084  
运行: 1779ms

## 7 用户1立即退订套餐5（通过历史记录和月账单查看）

[

套餐名：话费套餐

套餐费用：20.00

订购时间：2018-10-30T21:13:18

生效时间：2018-10-30T21:13:18

到期时间：2018-11-01T00:00

,

套餐名：本地流量套餐

套餐费用：20.00

订购时间：2018-10-30T21:14:18

生效时间：2018-11-01T00:00

到期时间：

,

套餐名：大王卡套餐

套餐费用：68.00

订购时间：2018-10-30T21:15:18

生效时间：2018-10-30T21:15:18

到期时间：2018-10-30T23:04:08

,

套餐名：本地流量套餐

套餐费用：20.00

订购时间：2018-10-30T22:56:56

生效时间：2018-10-30T22:56:56

到期时间：

2018 年 10 月  
您好! 17118860822  
本月账单 (单位: 元)  
合计: 6184.00  
套餐费用:  
话费套餐—20.00  
本地流量套餐—20.00  
基础费用:  
本地流量费用—6144.00

## 5.8 下月退订

设计:

```
update orders set endTime=#{endTime} where id=#{oid}
```

对应方法:

```
void cancelPlan(int oid, LocalDateTime nowTime, boolean immediately);
```

截图:

开始时间: 2018-10-30T22:56:57.707  
结束时间: 2018-10-30T22:56:57.861  
运行: 154ms

## 8 用户1下月退订套餐1（通过历史记录查看结果）

[

套餐名：话费套餐

套餐费用：20.00

订购时间：2018-10-30T21:13:18

生效时间：2018-10-30T21:13:18

到期时间：2018-11-01T00:00

套餐名：本地流量套餐

套餐费用：20.00

订购时间：2018-10-30T21:14:18

生效时间：2018-11-01T00:00

到期时间：

套餐名：大王卡套餐

套餐费用：68.00

订购时间：2018-10-30T21:15:18

生效时间：2018-10-30T21:15:18

到期时间：2018-10-30T23:04:08

套餐名：本地流量套餐

套餐费用：20.00

订购时间：2018-10-30T22:56:56

生效时间：2018-10-30T22:56:56

到期时间：

## 6. 注意事项

1. 系统中单位均简化，通话基本单位为分钟，流量为MB
2. 为准确计算时间，关闭了Mybatis自带的缓存功能，配置文件为resource文件加下的mybatis-config.xml，可自行修改

## 7. 优化

- 
1. 使用情况均记录在一个表中，实际情况各有不同。应该将其分开，不仅可以增加信息字段，而且可以提高对应数据查询的速度，减少冗余
  2. 订单表合并了订购退订记录，造成部分列存在到期时间字段为null，应该将订购和退订分开表存，在退订表中保留对应订购id字段