# 中央交易系统业务 要做的工作

D组 陈玮烨 孙克染 张梓欣 杨清杰

## 表格结构

### 1. 指令

#### 1.1 定义

- 一类具备以下属性的描述股票的买卖操作的对象
  - **编号**：唯一性的编号 作为指向该指令的索引
  - **时间**：日期YYYY-MM-DD 与 时间HH:MM:SS
    - （约束）交易时间应当处于系统允许进行交易的时间
  - **类别**：买指令 和 卖指令
  - **用户标识**：标记发出该指令的用户
    - （约束）在数据库中应当是一个连接用户列表关系的外键
  - **股票代码**：指定要被交易的股票
    - （约束）在数据库中应当是一个连接股票列表关系的外键
  - **交易数量**：单位为股，即该指令涉及的对应的股票代码的股票的在本指令的交易数量
    - （约束）该数目应当大于0，在卖出指令中应当小于等于该用户持股的数量，在买指令中应当小于等于该股票代码对应的企业的股票发行量）
  - **指令价格**：指令发出时用户指定的每股的买/卖价格。区别于最终的成交价格。
    - （约束）受限于对应日期的涨跌停限制价格
  - **状态**：待交易/存档/撤销
    - （约束）默认值：待交易
  - **未成交股数**：在该指令下没有成交的股票的数量（还需要继续撮合的股票数量）

#### 1.2 作为关系的内容

- 在数据库实现里，我们可以把指令分为不同的情况储存在不同的关系中
  - 待交易卖指令
    - 包括未全部卖出的卖指令
  - 待交易买指令
  - 存档买指令（Archived）
    - 即不需要继续交易的买指令

        - 包括全部买入的 未全部买入的 和 未买入的
    - 存档卖指令
- 问题
    - 对于用户可以看到的，应当是基于上述表中关于用户产生的指令的视图。
    - 根据需求可能会在一些表上添加更多的字段
    - 约束可能需要在前端和后端中都体现出来
        - 前端可以在确定的情况下限制用户操作
            - 例如规定在指定时间中进行交易 需要在前端中体现
        - 但结果应以后端相应的结果为准

**1.3 操作**

- 发出买卖指令
    1. 用户在前端指定了指令定义中的可由用户定义的部分（股票代码、交易数量、卖卖价格等参数）
    2. 向后段传输指示，生成SQL语句。
    3. 添加记录，触发触发器。
        - 每次添加记录时查询时，在对应的待交易的买卖指令表内查询是否有对应的指令可以形成交易。如果没有就不进行操作。
            - 形成交易的规则见第2节
        - 当成功添加指令时，向前端反馈指令发出成功。
- 撤销指令
    1. 用户在前端指示对指定的指令进行撤销
    2. 查询该指令，并检查其被交易的状态
        - 首先检查该交易是否还在待交易表中
            - 不在，则查询是否在存档表中，并给予反馈
        - 若还处于待交易状态，则获取完成交易的股数和价格
    3. 如果有未交易完成的
- 查询用户已发出的指令
    - 用户在前端指示查询已发出的指令
    - 向后端传输查询结果
- 存档流程
    - 过期自动存档（指令过期）
        - 交易日结束后（通常可以是交易日，我们将待待交易指令表中未完成的指令自动移动到存档指令表
    - 完成交易 进行存档

## 2. 交易

**2.1 定义**

- 交易是一类包含以下属性的对象
    - 交易编号

    - 买指令编号
    - 卖指令编号
    - 数量
    - 单价
    - 成交时间
- 交易关系中的记录由关于指令表的触发器生成
    - 每添加一个指令，都会查询是否可以生成交易
    - 具体的情况如下（撮合）
        - 每插入一个待交易的买指令时
            1. 查询待交易的卖指令：对应股票中 选择所有小于等于买指令价格的卖指令 按价格由高到低排序 再按时间由前至后排序 最后按卖指令中未交易的股票数量由大到小排序
            2. 将买指令和上述查询得到的待撮合指令一一排序撮合 每次撮合在对应的买卖指令关系中，减少“未成交股数”对应的数字
            3. 直到该买指令的未成交股数为0 或者是1中查询的卖指令全部撮合完毕为止
        - 每插入一个待交易的卖指令时
            1. 查询待交易的麦指令：对应股票中 选择所有大于等于卖指令价格的买指令 按价格由低到高排序 再按时间由前至后排序 最后按卖指令中未交易的股票数量由大到小排序
            2. 将买指令和上述查询得到的待撮合指令一一排序撮合 每次撮合在对应的买卖指令关系中，减少“未成交股数”对应的数字
            3. 直到该卖指令的未成交股数为0 或者是1中查询的买指令全部撮合完毕为止
        - 其他撮合规则
            - 每次价格为 $撮合价格 = \frac{买指令价格+卖指令价格}{2}$
    - 交易细节（和哪些买卖指令成交了）对用户隐藏，用户看到的是成交情况，见3.

## 3. 成交

- 定义
    - 成交是某个对应的指令参与的交易的统计集合，可以看作一个视图
        - 指令编号
        - 通过查询对应的指令所有的交易，形成价格的平均值和交易数量
            - 价格的平均值采用加权平均
        - 状态：未成交/部分成交/完全成交

# *系统结构*

- 数据库
    - 采用规定的DBMS（MySQL等）
- 后端代码
    - （拟）采用 Spring + SpringMVC + Mybatis(SSM)框架，主要涉及语言：Java

    - （拟）采用node.js架设后端，主要涉及插件：Sequelize，使用编程语言：JavaScript
- 与前端商讨，规定相同的数据接口