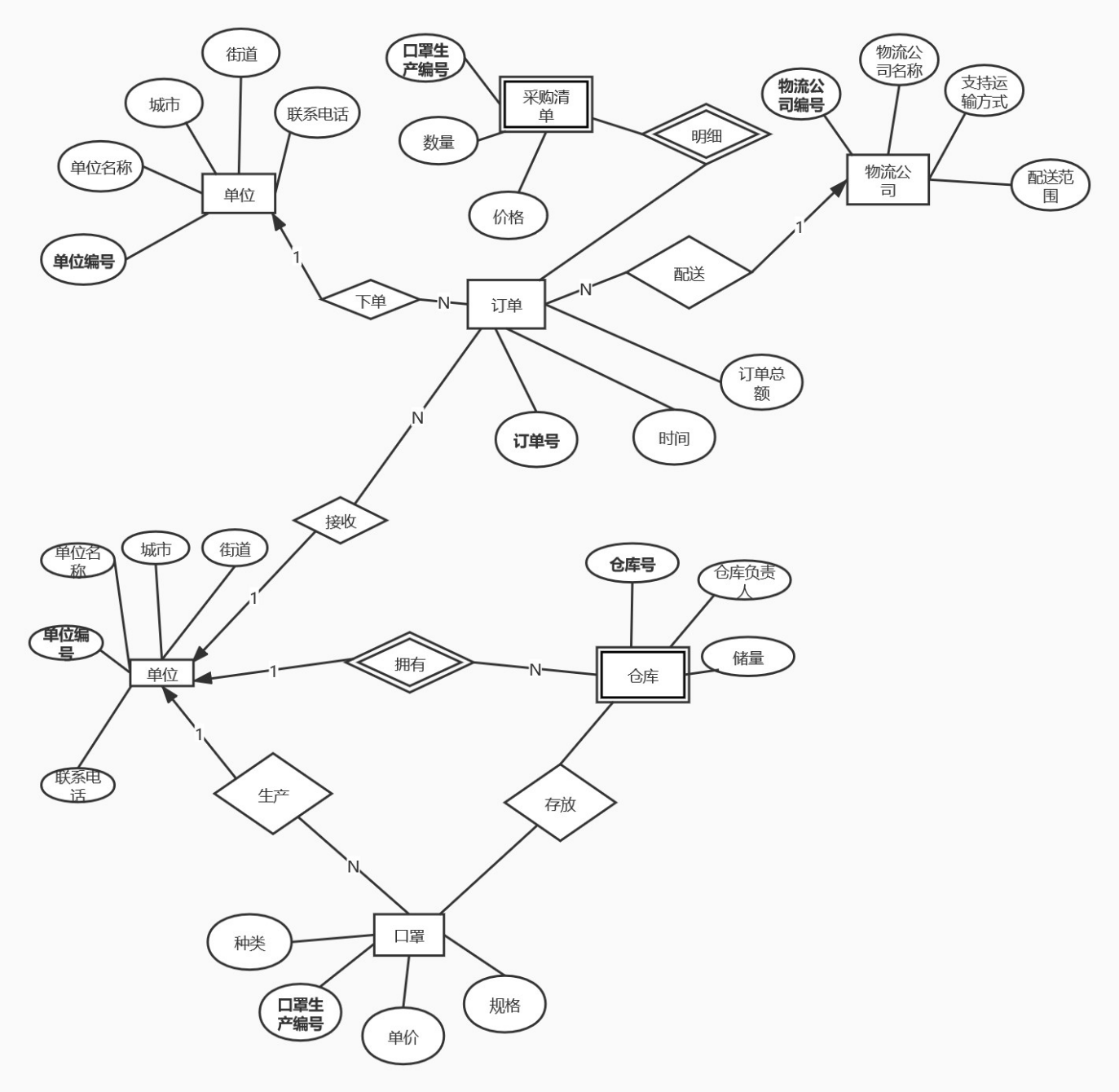
参考解答

E-R图



From ER to RM

第1步：转化

购买单位（<u>单位编号</u>, 单位名称，城市，街道，联系电话）

工厂（<u>单位编号</u>, 单位名称，城市，街道，联系电话）

物流公司（<u>物流公司编号</u>，物流公司名称，支持运输方式，配送范围）

订单（<u>订单号</u>，订单总额，时间）

采购清单（订单号，口罩生产编号，数量，价格）

口罩（口罩生产编号，种类，规格，单价）

仓库（工厂编号，仓库号，仓库负责人，储量）

下单（订单号，单位编号）

配送（订单号，物流公司编号）

接收（订单号，工厂编号）

生产（口罩生产编号，工厂编号）

存放（口罩生产编号，仓库号）

第2步：规范化

采购清单中***口罩生产编号→价格，数量***，为3NF违例，

分解为***口罩（订单号，口罩生产编号）*** 和 ***单价（口罩生产编号，数量，价格）***

第3步：消除冗余

利用函数依赖的合并律，将：

下单（订单号，单位编号）

配送（订单号，物流公司编号）

接收（订单号，工厂编号）

订单（订单号，订单总额，时间）

合并为：

订单（订单号，订单总额，购买单位编号，时间，物流公司编号，工厂编号）

最终结果

购买单位（单位编号, 单位名称，城市，街道，联系电话）

工厂（单位编号, 单位名称，城市，街道，联系电话）

物流公司（物流公司编号，物流公司名称，支持运输方式，配送范围）

采购清单（订单号，口罩生产编号，数量，价格）

口罩（订单号，口罩生产编号）

单价（口罩生产编号，数量，价格）

仓库（工厂编号，仓库号，仓库负责人，储量）

订单（订单号，订单总额，购买单位编号，时间，物流公司编号，工厂编号）

生产（口罩生产编号，工厂编号）

存放（口罩生产编号，仓库号）

ODL

```
//基本单位
Interface Unit(key code)
{
    attribute string code;
    attribute string name;
    attribute Struct    Addr
    { string street,string city } address;
    attribute string telephoneNumber;
}
//口罩
Interface Mask(key code)
{
    attribute string code;
    attribute string price;
    attribute string spec; //规格
    attribute string type; //型号

    relationship Factory producedBy
        inverse Factory::produce;
    relationship Warehouse storedBy
        inverse Warehouse::store;
}
//仓库
Interface Warehouse(key number)
{
    attribute int number;
    attribute string manager;
    attribute string capacity; //容量

    relationship Factory ownedBy
        inverse Factory::own;
    relationship Set<Mask> store
        inverse Mask::storedBy;
}
//工厂
Interface Factory : Unit
```

```
{
    relationship Set<Mask> produce
        inverse Factory::producedBy;
    relationship Set<Warehouse> own
        inverse Factory::ownedBy;
    //处理订单
    relationship List<Order> process
        inverse Order::processedBy;
}
//物流
Interface Logistic(key code)
{
    attribute string code;
    attribute string name;
    attribute string transportationType;//运输类型
    attribute string deliveryRange; //配送范围
    //配送订单
    relationship List<Order> deliver
        inverse Order::deliveredBy;
}
//购买单位
Interface buyerUnit : Unit
{
    //下单
    relationship List<Order> place
        inverse Order::placedBy;
}
//订单
Interface Order(key code)
{
    attribute string code;
    attribute string generateTime; //生成时间
    attribute string totalPrice; //总价

    attrubute struct List
    {
        string maskCode, string price, string quantity
    } list;//订单中口罩的订购信息

    relationship buyerUnit placedBy
        inverse Unit::place;
    relationship Factory processedBy
        inverse Factory::process;
```

```
80     relationship Logistic deliveredBy
81         inverse Logistic::deliver;
82 }
```

From ODL to RM

step 1: convert

Unit (<u>code</u>, name, street, city, telephoneNumber)

Mask (<u>code</u>, price, spec, type, factoryCode, warehouseNumber)

Factory (<u>code</u>, name, street, city, telephoneNumber)

Logistic (<u>code</u>, name, transportType, deliveryRange)

buyerUnit (<u>code</u>, name, street, city, telephoneNumber)

Order (<u>code</u>, generateTime, totalPrice, maskCode, price, quantity, buyerCode, factoryCode, logisticCode)

step 2：decompose

in relation Order, **m*askCode->price，quantity*** is 3NF violations.

decompose Order into:

Order (<u>code</u>, generateTime, totalPrice, maskCode, buyerCode, factoryCode, logisticCode)

and

UnitPrice (<u>maskCode</u>, price, quantity)

final result

Unit (<u>code</u>, name, street, city, telephoneNumber)

Mask (<u>code</u>, price, spec, type, factoryCode, warehouseNumber)

Factory (<u>code</u>, name, street, city, telephoneNumber)

Logistic (<u>code</u>, name, transportType, deliveryRange)

buyerUnit (<u>code</u>, name, street, city, telephoneNumber)

Order (<u>code</u>, generateTime, totalPrice, maskCode, buyerCode, factoryCode, logisticCode)

UnitPrice (<u>maskCode</u>, price, quantity)