**设计思路：**

道具和装备虽然有一些共性，但是服务器是把这两种类型分别存在item和equip两张表里面的，逻辑也是分开处理的。这样做的原因首先是提高数据库效率，因为装备需要有增益属性和宝石镶嵌信息，这些是以字符串形式存储在数据库中的，如果统一处理，mysql会为每一件道具预留这两个字段的空间，造成浪费。另外逻辑分开处理也会使服务器结构更加清晰。

但是考虑到配置文件大小有限，为了方便策划同学配置，道具和装备是配置在一起的。

**配置设计：**

配置的基础表是item表，道具和装备都配置在这里，下面说一下几个重要字段的含义：

**用于道具：**

targetItem：奖励列表ID，主要用于宝箱，奖励后面会详细介绍；

needItem: 前置装备列表，有一些道具的使用需要别的道具配合，比如有的宝箱需要钥匙，强制装备列表配在这里；

componentItem: 合成该装备所需要的道具列表；

targetItem/needCount: 合成兑换列表和所需本道具的数量；

gemId：宝石的基础属性配置ID，用于索引到baseAtrr配置表。

**用于装备：**

equipAttId ：用于索引到equipAttr配置表，equipAttr用于配置道具的不同品级和强化级别对应的附加属性和基础属性索引，基础属性索引和宝石的基础索引公用baseAttr配置表。

**奖励消耗：**

在道具和奖励配置表中经常出现3\_20001\_1\_1000,4\_70001\_1\_3\_2\_1000,1\_1\_88\_10000这种配置，这是奖励组中的一个配置， 初看可能会比较迷惑，这里详细分析一下这种结构的设计思路：

首先配置列表可能含有多项用逗号分隔的子项，每一个独立的子项都代表一种奖励，比如3\_20001\_1\_1000代表奖励类型为3（道具），道具配置id为20001，数量为1，奖励权重为1000。4\_70001\_1\_3\_2\_1000代表奖励类型为4（装备），装备配置id为70001，数量为1，品级为3，强化级别为2，权重为1000. 这里需要注意的是装备和其他奖励相比会多出两个字段分别代表品级和强化级别。之所以这样设计是为了简化奖励配置格式并且便于灵活扩展，根据第一列项奖励类型和第二列id就可以组装出不同的奖励类型。

目前的奖励类型主要有一下几种：

**itemType**

**{**

// 空属性

**NONE\_ITEM = 0;**

// 玩家属性 对应的id可能是金币或者经验的id

**PLAYER\_ATTR = 1;**

// 角色属性 对应的id目前只有怪物经验

**MONSTER\_ATTR = 2;**

// 道具 对应配置表中的道具

**ITEM = 3;**

// 装备 对应配置表中的装备

**EQUIP = 4;**

// 技能 （预留）

**SKILL = 5;**

// 奖励组 对应奖励组中的一个奖励

**GROUP = 6;**

// 怪物奖励 对应一个怪物的奖励

**MONSTER = 7;**

**}**

针对不同的类型又会有不同的id，比如道具或者装备对应配置里面的不同id，玩家属性类型则会对应玩家的不同增益id，比如金币，经验等。

服务器在处理玩家的经理或者消耗是会发送一个列表，列表中也包含了类型和id，值和配置的原则一致，另外为了安全起见服务器会把更改过的玩家和怪物属性（金币，经验，等级等）的当前状态连同列表一起同步给客户端。