**Design Inventory system**

**场景：  
你需要设计一个库存管理系统（Inventory System），用于管理商品库存和订单操作，保证库存在高并发环境下的正确性，同时考虑系统可扩展性和安全性。**

**核心功能要求**

1. **Add stock（增加库存）**
   * **支持补货或增加仓库库存数量。**
2. **Reserve stock（预留库存）**
   * **当用户下单时，将库存临时锁定，防止其他订单抢占（防止超卖）。**
3. **Fulfill stock（完成订单）**
   * **当订单完成支付或发货时，真正扣减库存。**
4. **Clear reservation（清理预留）**
   * **订单取消或超时未支付时，将预留库存释放回可用库存。**

**关键非功能点**

1. **并发控制 / race condition**
   * **多个用户同时下单时，需要防止库存超卖。**
   * **面试官可能希望你讨论事务、锁、乐观/悲观并发控制或 Redis 原子操作。**
2. **扩展性 / Scaling**
   * **系统需要支持高并发访问。**
   * **可考虑缓存、分库分表、异步消息队列等。**
3. **安全性**
   * **防止非法库存操作。**
   * **API 鉴权、权限控制、加密传输等。**
4. **Production Deployment Best Practices**
   * **高可用 DB、负载均衡、日志监控、备份、CI/CD 部署。**

**面试考察重点**

* **业务流程理解：Add → Reserve → Fulfill / Clear**
* **事务与并发处理：防止库存被重复预留或扣减**
* **系统可扩展性：缓存、异步处理、分库分表**
* **安全性与运维：权限控制、监控、高可用部署**

系统要准确地反映每个 product的real-time availability，保证customer下单的话 就能拿到他们选购的商品。两个role：customer 下单；shopper 接单-》去对应商店取货-》送货

Instacart, we have customers that place orders on our website, and personal shoppers we hire, whose job it is to go to grocery stores to fulfill those orders. To increase shopper efficiency we want to set up "dark stores" for some of our retail partners. These are special stores that carry a full range of groceries, are closed to the public, and are only available to Instacart shoppers.

We want to design a system that allows for:

Stocking inventory. The store gets periodic shipments from the retailer. The system needs to update its inventory tally and respond appropriately.

Reserving items. When customers add items to their baskets, we’ll make an API call to confirm availability. The system needs to respond appropriately and ensure that it honors these reservations and prevents other customers from purchasing them.

Shopping for customers. Remove items from inventory when they are collected by the personal shopper at the dark store.

Design inventory system (add stock, reserve stock, fulfill stock, clear reservation) focus 在 transaction, racing conditions. 还问了security 和 production deployment best practice, 如何scale

**Design Category system**

**场景**

* **设计一个 商品分类展示系统（Catalog / Category System），用户通过网页浏览商品。**
* **商品可以属于 多个分类（多对多关系）。**
* **分类可以有 多级层次（父分类、子分类），但前端每次只需要展示 直属子类。**

**用户需求**

1. **用户打开网页时：**
   * **显示所有顶层分类（例如 Food、Clothes）。**
   * **显示部分商品（热门商品或顶层分类商品）。**
2. **用户点击某个分类：**
   * **显示该分类的 直属子分类。**
   * **显示该分类下的 商品。**
   * **不需要递归显示所有子孙分类（简化查询和缓存）。**
3. **商品可能属于 多个分类：**
   * **一个商品可能在不同分类页面同时出现。**

**面试考察点**

1. **数据库设计**
   * **如何存储分类（Category 表自关联实现树形结构）。**
   * **如何存储商品与分类的多对多关系（Product / ProductCategory 表）。**
2. **API 设计**
   * **如何设计请求与返回格式。**
   * **如何让前端方便地 fetch 分类和商品信息。**
3. **数据访问模式**
   * **如何查询顶层分类。**
   * **如何查询某分类下的直属子分类和商品。**
   * **如何处理商品属于多个分类的情况。**
4. **系统扩展性 / 高可用性（follow-up）**
   * **如何缓存分类和商品信息。**
   * **如何在大量用户访问时扩展系统。**
   * **如何保证系统稳定运行。**

**重点注意**

* **前半部分面试会偏重 API + 数据库 schema。**
* **后半部分可能会问 整体系统架构、Cache、Load Balancer、数据同步策略 等细节。**
* **面试中需要清楚表达：**
  1. **数据在数据库中的存储形式。**
  2. **请求格式和响应格式。**
  3. **前端 fetch 的逻辑。**

design catalog, 设计一个self join的表就行了，主要focus在 API和db schema

题意就是一个货物信息的display，然后货物有各种标签 可以属于不同的类，然后题目就是设计一个web page来display这些货物信息。

我当时问了一些问题，比如用户分布是否是全球、是否有频繁的update webpage还是说就静态地展示。但是我的考官考察的要点是数据库，scheme是怎么样的。

system design：catalog system。很typical的system design。前半个小时api design， schema design。 后半个小时讨论整体的structure， 会比较详细的问每个structure中的每个component的细节， 比如cache中key value pair怎么设计， load balancer具体会干嘛

System Design: Catalog

楼主的列表每次只需要回复某标签的全部直属子嗣，包括标签以及物品，不需要包含子嗣的子嗣，也就省去了优化数据库或者缓存标签族谱的必要。可以理解为用户每次点一个某标签的按钮，后续标签按钮与货物出现，再点一个，再细分。

其他要注意的点是，清晰的表达数据在库中的储存形式，request的格式与response的格式。楼主还被着重问了用户刚打开网页的细节，网页需要在用户互动之前显示些商品那么商品从哪来，是否要request，用户们刷新会如何之类，建议提前思考一下。

后面楼主时间充裕，写了管理员视角，也写了一些补充的东西，图文会被保存到后续审核，所以建议聊天的途中把重要的东西全部写在板子上

SD: category system

主要focus on getCategories API 和 data model， request format

Design a catelog system

how to handle a product belongs to multiple catelog

API design

database schema

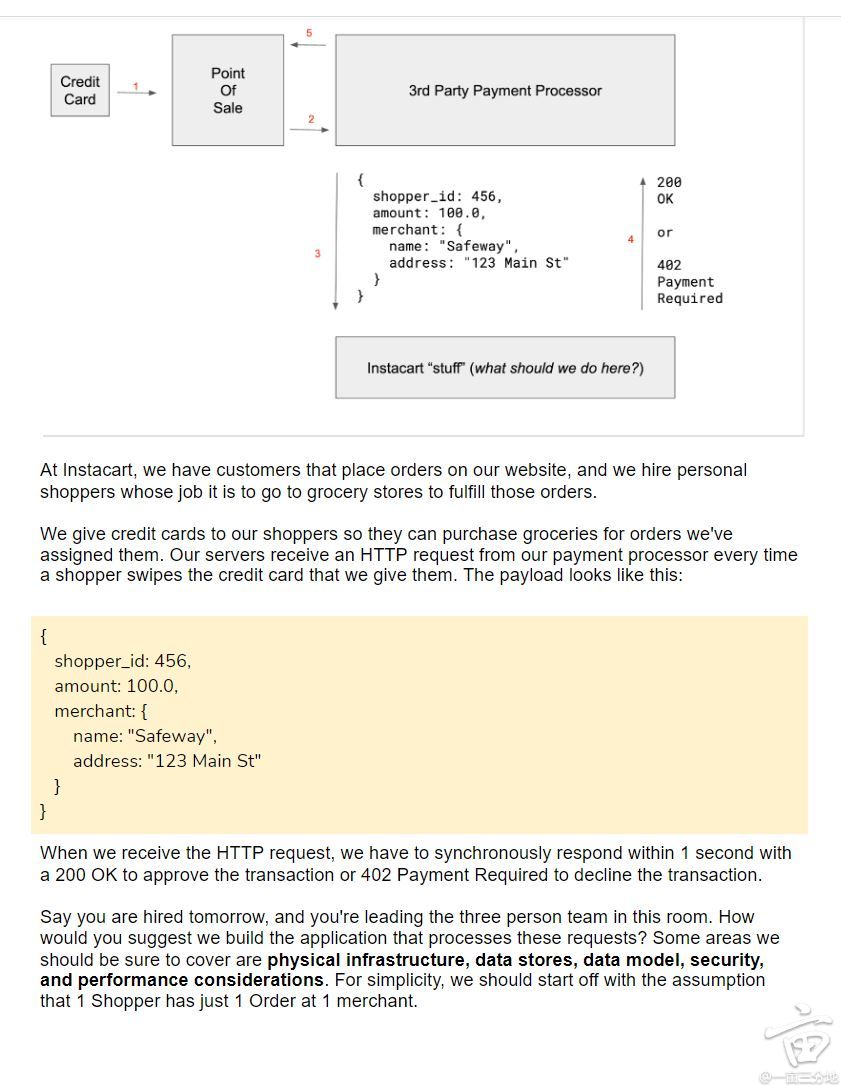
SD设计商店的catalog 系统， 主要focus 在DB schema， 讲清楚如何data access pattern 去实现前端fetch 类目以及每个类目中商品信息。

follow up： 1 一个商品可能会属于多个category 怎么解决 2 如何scale 系统 3 如何确保系统在正常运行

System design

问如果做一个能显示各种category （food，clothes..）的页面, 然后点开每个category 的话还能显示sub- category，应该怎么设计database

**Design Payment Service**



At Instacart, we have customers that place orders on our website, and we hire personal shoppers whose job it is to go to grocery stores to fulfill those orders.

We give credit cards to our shoppers so they can purchase groceries for orders we've assigned them. Our servers receive an HTTP request from our payment processor every time a shopper swipes the credit card that we give them. The payload looks like this:

{

    shopper\_id:  123，

    amount: 300,

    merchant: {

        name: "Target",

        address: "123 Main St"

    }

}

When we receive the HTTP request, we have to synchronously respond within 1 second with a 200 OK to approve the transaction or 402 Payment Required to decline the transaction.

Say you are hired tomorrow, and you're leading the three person team in this room.  How would you suggest we build the application that processes these requests? Some areas we should be sure to cover are \*physical infrastructure, data stores, data model, security, and performance considerations\*.  For simplicity, we should start off with the assumption that 1 Shopper has just 1 Order at 1 merchant.

Payment Service (着重在API). follow up: DB的选择, 怎么scale DB.

设计一个payment verification的服务器，接受的request是他们的第三方支付服务，形如{shipper\_id:1123, amount: 13.34, merchant: {addr: 123 ave, city:new york}} ，SLA 1s

design shopper payment verification service。问了api，security，data model， scale。

设计一个payment verification的服务器，接受的request是他们的第三方支付服务，形如{shipper\_id:1123, amount: 13.34, merchant: {addr: 123 ave, city: san jose}} ，问如何在1sec之内，完成这笔交易的验证，并返回。

    主要考察以下几点

     1）performance

     2) security

     3) data model

     4) data storage

     5) stability/ fallover

     问到一个小问题，shopper在刷信用卡时的金额和用户下订单的不是完全一致，比方说有10刀左右的误差，为什么。原因有二：1）instacart系统里面的价格和商店的标价没有实时一致，有滞后的情况。2）有些东西比如一斤苹果，买的时候重量有误差。然后追问，如何判断这个误差是正常的误差，还是shopper有欺诈。 我给的方案是，用历史数据作为判断依据。可以加一个async的服务，一旦交易完成，就记录下每个商品的误差，并且存到db里面。

还有问到db是被不同的service共享的，如何提高性能。加cache，怎么加cache，cache里面存放的啥，dump的策略是什么。我这个地方没有回答好。我最后还专门问了一下面试官。他说用write through的策略会简单有效，而且只用存放每个shopper最新的一个order的信息就好了。

感觉这道题的难点是API design 以及 如何处理transaction error handling/rollback 和ACID compliance

应该主要focus on merchant和amount的verification service，而不用考虑其他像是Payment gateway或者PSP。感觉我刚开始说的时候有点偏题了。

被问的还是很细的，比如说什么是index，如何优化transactions的读，如何实现idempotent，如果同时收到两个request with different amount如何决定

设计 shopper swipe credit card 的时候的 risk evaluation

这题在论坛里被提到过几次。面试官会先描述一下这个场景，然后说 issuer bank 会来 call instacart 的 api, 并给了一个 sample payload, 然后要面试者描述背后怎么实现。楼主没领悟到其中的奥秘，就设计了几张 db table, 然后根据 input dollar amount 去尝试猜一个 open order, 然后结合一些别的信息来判断是否 reject transaction. 面试官在过程中并没有问太多的问题。

問了需要哪些table schema 怎麼設計 用sql or nosql, why?

資料很多怎麼增加performace

cache, LB 基本問題

斷電 怎麼保證資料都有存好

怎麼確保刷兩次卡 只扣一次錢

怎麼確定shopper 買對的東西

- 如果shopper在不合作的店家買東西 怎麼handle

- 如果shopper分很多次完成訂單怎麼辦, order 10瓶牛奶,太重了 得分兩次刷卡 一次買五瓶

- 商場東西不夠了怎麼辦,order 要五個蘋果, shopper買3個, 買10個分別怎麼處理

- 如果 價格有誤差怎麼處理, costco說蘋果一個一塊 現場卻一個兩塊

**-------------------------------------------------------------------------------------------------------------------------------**

design: inventory system，聊了一下race condition，如何scale

SD： design catalog, 地理有好多参考，设计一个self join的表就行了，主要focus在 API和db schema

system design -> catalogs (面试官在中期比较喜欢打断我说话，也是一个经验吧）

题意就是一个货物信息的display，然后货物有各种标签 可以属于不同的类，然后题目就是设计一个web page来display这些货物信息。

我当时问了一些问题，比如用户分布是否是全球、是否有频繁的update webpage还是说就静态地展示。但是我的考官考察的要点是数据库，scheme是怎么样的。

system design：catalog system。很typical的system design。前半个小时apidesign， schema design。 后半个小时讨论整体的structure， 会比较详细的问每个structure中的每个component的细节， 比如cache中key value pair怎么设计， load balancer具体会干嘛

System Design: Catalog

题目本身不过多描述了，也是老题。不知道题目是否每次都完全一样，楼主的列表每次只需要回复某标签的全部直属子嗣，包括标签以及物品，不需要包含子嗣的子嗣，也就省去了优化数据库或者缓存标签族谱的必要。可以理解为用户每次点一个某标签的按钮，后续标签按钮与货物出现，再点一个，再细分。

其他要注意的点是，清晰的表达数据在库中的储存形式，request的格式与response的格式。楼主还被着重问了用户刚打开网页的细节，网页需要在用户互动之前显示些商品那么商品从哪来，是否要request，用户们刷新会如何之类，建议提前思考一下。

后面楼主时间充裕，写了管理员视角，也写了一些补充的东西，图文会被保存到后续审核，所以建议聊天的途中把重要的东西全部写在板子上

系统设计: 老题Payment Service (着重在API). follow up: DB的选择, 怎么scale DB. 感觉我说什么都回馈OK. 但HR后来反馈是这轮拿的No. 这是原贴: 🔗 www.1point3acres.com (题目在评论区里)

验证payment那道 https://www.1point3acres.com/bbs/thread-777511-1-1.html

SD - Design inventory system (add stock, reserve stock, fulfill stock)

SD: category system

主要focus on getCategories API 和 data model， request format

System Design： inventory system. focus 在 transaction, racing conditions. 还问了security 和 production deployment best practice.

Design a catelog system

how to handle a product belongs to multiple catelog, API design, database schema

Design an inventory system: 要支持reserve stock和clear reservation

SD设计商店的catalog 系统， 主要focus 在DB schema， 讲清楚如何data access pattern 去实现前端fetch 类目以及每个类目中商品信息。

follow up： 1 一个商品可能会属于多个category 怎么解决 2 如何scale 系统 3 如何确保系统在正常运行

System design

问如果做一个能显示各种category （food，clothes..）的页面, 然后点开每个category 的话还能显示sub- category，应该怎么设计database

以下是我聊到或者面试官问到的点，大家可以参考着准备

physical infrastructure: server, storage, network

data stores: SQL vs NoSQL

data model: merchant, shopper, order, transaction (includes order id in transaction table)

security: API token, pre-shared secret

performance considerations: load balancer, data partition, write through cache of order table

Monitoring: how to ensure the system performs expected?

Testing & Deployment: Load testing etc

Research & Analytics: how can data scientists leverage data for research?

尤其有两个特匪夷所思的点：

如果database down了，如何确保在sla一秒内respond？这时直接approve，因为他们trust shoppers，认为fraud rate低

database不需要replica，面试官认为这个只会增加latency或者inconsistency，不值得有这个complexity