# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/094~103资料/102\_新零售业务下的采购主题表设计/笔记.docx**

《102\_新零售业务下的采购主题表设计》

ods，dwd，dim

ods：从新零售公司的业务系统里采集过来的数据，canal监听人家的库的binlog，把binlog同步过来，塞到ods层的表里去，追加的过程，每天凌晨都得有一个spark的作业，专门对业务表的binlog进行一下整合

增删改的日志，在ods层面给你做出来一份完整的今天零点以前的这个业务表的完整的数据，dwd的数据就从这里来

ods -> dwd，ETL做一些数据清洗和字段名称的转化，标准化的工作

把我刚才讲的一系列的事实表都做了，dim维度表也是一样的

dwt，两个粒度的采购主题表，一个是采购主题表，以采购订单no为核心，关联所有的发货通知、仓库收据、供应商支付交易，关联所有的维度表，做出来一张超级大宽表，一个是采购条目主题表

采购订单条目为核心粒度，关联到商品级别，关联其他的发货通知、仓库收据、供应商支付交易、所有的维度，都在里面

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/094~103资料/094\_新零售业务下的采购部门的数据需求分析/笔记.docx**

《094\_新零售业务下的采购订单事实表的设计》

订单、库存，采购

商品、采购、仓储、库存、订单、营销、门店

数据需求：采购部门的人，库存的数据分析报表，决定在什么时候去采购什么商品，采购多少件，配合库存的数据分析，大概可以估算出来他的库存周转的周期是多久，采购商品的时候，选择哪个供应商，哪个供应商历史上发货的时间最快，商品质量最好，而且商品的价格相对较低一些的

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/094~103资料/100\_新零售业务下的供应商支付交易事实表设计/笔记.docx**

《100\_新零售业务下的供应商支付交易事实表设计》

采购一批商品，发货速度最快，次品率最低，价格最低，回款周期最长

各自有各自的优缺点：有的供应商发货速度比较慢一些，价格高一些，汇款周期比较短，次品率最低，商品质量比较高；有的供应商，发货速度快，价格低一些，回款周期比较长，次品率略高一些，100件商品总有5件是次品

数据分析的时候，考虑有的时候有一些供应商是否会给我们一些折扣

供应商支付交易

vendor\_payment\_no：供应商支付交易编号

vendor\_payment\_datekey：供应商支付交易日期

vendor\_no：供应商编号

warehosue\_no：仓库编号

contract\_terms：合同条款

contract\_no：合同编号

vendor\_invoice\_amount：供应商发票金额

vendor\_discount\_amount：供应商折扣金额

vendor\_net\_payment\_amount：供应商实际支付金额

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/094~103资料/103\_新零售业务下的采购轻度聚合表设计/笔记.docx**

《103\_新零售业务下的采购轻度聚合表设计》

按照两种粒度来进行轻度聚合，每个供应商、每个仓库、每个物流公司，进行一定的轻度的聚合，算出来一堆的轻度聚合的指标；每个供应商、每个物流公司、每个仓库、每个商品，进行一定的轻度的聚合

可以大幅度的减少数据

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/094~103资料/101\_新零售业务下的采购累积快照事实表的设计/笔记.docx**

《100\_新零售业务下的采购累积快照事实表的设计》

库存周期快照事实表，每天的库存情况都存一行，在一行里对库存的变动的关键事件进行一个维护和统计分析，事务事实表，用户行为日志

purchase\_order\_no：采购订单编号

purchase\_order\_datekey：采购下单日期

shipping\_notification\_no：发货通知编号

shipping\_notification\_datekey：发货通知日期

warehouse\_receipt\_no：仓库收据编号

warehouse\_receipt\_datekey：仓库收据开具日期

vendor\_payment\_no：供应商支付交易编号

vendor\_payment\_datekey：供应商支付交易日期

vendor\_invoice\_datekey：供应商开发票日期

vendor\_no：供应商编号

contract\_terms：合同条款

contract\_no：合同编号

employee：下单员工编号

warehouse\_no：仓库编号

carrier\_no：运输商编号

purchase\_order\_amount：采购总金额

purchase\_order\_quantity：采购商品总数量

shipped\_quantity：发后商品总数量

received\_quantity：仓库收获总数量

return\_of\_goods\_quantity：仓库退货的次品数量

vendor\_invoice\_amount：供应商发票总金额

vendor\_discount\_amount：供应商折扣总金额

vendor\_net\_payment\_amount：供应商实际付款金额

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/094~103资料/095\_新零售业务下的采购业务流程分析/笔记.docx**

《095\_新零售业务下的采购业务流程分析》

采购部门，采购专员，平时会盯着库存数据分析报表，要看着每个商品的剩余库存还有多少，一旦说库存快要消耗完毕了，此时应该有一个系统的库存预警，他会知道说一个商品的库存周转周期是多久

决定，要对一个商品采购多少件

看一下采购数据分析报表，他需要分析一下，历史上对这个商品的不同的供应商的发货速度、次品退货的次数、商品报价，选择一个最合适的供应商，他其实应该去跟供应商进行一下商谈，商品有多少库存，商品报价

下一个采购订单，分成很多的采购订单条目，我从这个供应商这里，要采购什么商品，采购多少件，发采购订单给供应商

供应商会发货，此时他会委托运输商，物流公司，把商品运送到我指定的仓库那里去，这个运输是需要一段时间的，此时他一旦发货，供应商会发一个发货通知单给我，告诉我他已经发货了

过了一点时间，仓库管理员就会收货和验货，然后商品会入库，对库存会有增加 -> 库存那套东西了

对供应商进行打款，是最后的一个步骤

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/094~103资料/098\_新零售业务下的发货通知事实表单的设计/笔记.docx**

《098\_新零售业务下的发货通知事实表单的设计》

发货通知

shipping\_notification\_no：发货通知编号

shipping\_notification\_datekey：发货日期

purchase\_order\_no：采购订单编号

vendor\_no：供应商编号

warehouse\_no：仓库编号

carrier\_no：物流公司编号

employee\_no：下单员工编号

shipped\_quantity：发货总数量

发货通知条目

shipping\_notification\_item\_no：发货通知条目编号

shipping\_notification\_no：发货通知编号

product\_no：商品编号

shipped\_quantity：发货数量

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/094~103资料/099\_新零售业务下的仓库收据事实表的设计/笔记.docx**

《099\_新零售业务下的仓库收据事实表的设计》

仓库收据

warehouse\_receipt\_no：仓库收据编号

warehouse\_receipt\_datekey：仓库收货日期

vendor\_no：供应商编号

warehouse\_no：仓库编号

carrier\_no：物流公司编号

purchase\_order\_no：采购订单编号

employee\_no：下单员工编号

shipping\_notification\_no：发货通知编号

received\_quantity：收货总数量

return\_of\_goods\_quantity：退货总数量

warehouse\_employee\_no：仓储管理员编号

仓库收据条目

warehouse\_receipt\_item\_no：仓库收据条目编号

warehouse\_receipt\_no：仓库收据编号

product\_no：商品编号

received\_quantity：收货数量

return\_of\_goods\_quantity：退货数量

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/094~103资料/096\_新零售业务下的采购数据维度建模的4个步骤/笔记.docx**

《096\_新零售业务下的采购数据维度建模的4个步骤》

大概知道了采购部门的数据需求，采购业务流程，要对采购相关的数据进行维度建模，照例还是走维度建模4部曲

（1）选择业务流程

从供应商那里采购商品的一个全套业务流程，下采购订单开始，供应商发出发货通知，仓库收货验货，对供应商进行打款结算

（2）明确数据粒度

采购订单，采购订单的每个条目

发货通知

仓库收据

供应商支付交易

（3）确定维度数据

供应商、物流公司、仓库、日期、商品

（4）确定事实数据

采购商品总数量，采购商品总金额

每个采购订单条目，就对应了一个商品采购数量，一个商品采购金额

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/094~103资料/097\_新零售业务下的采购订单事实表的设计/笔记.docx**

《097\_新零售业务下的采购订单事实表的设计》

采购订单、发货通知、仓库收据、供应商支付交易，在你的仓储系统里都会有的

采购订单

purchase\_order\_no：采购订单编号

purchase\_order\_datekey：下采购订购的日期

vendor\_no：供应商编号

contract\_terms：合同条款

contract\_no：合同编号

warehouse\_no：仓库编号

carrier\_no：物流公司编号

employee\_no：员工编号

purchase\_agency\_no：采购代理编号

purchase\_order\_amount：本次采购订单总金额

purchase\_order\_quantity：采购总数量

采购订单条目

purchase\_order\_item\_no：采购订单条目编号

purchase\_order\_no：采购订单编号

product\_no：商品编号

purchase\_order\_item\_quantity：采购事务中的商品采购数量

purchase\_order\_item\_amount：采购事务涉及到的金额

供应商维度

vendor\_no：供应商编号

vendor\_name：供应商名称

vendor\_street\_address：供应商地址的街道名称，具体到门牌号的

vendor\_city：供应商地址所在市

vendor\_province：供应商地址所在省

vendor\_status：供应商状态，是否有效

shipping\_notification\_no：发货通知编号

shipping\_notification\_datekey：发货日期

purchase\_order\_no：采购订单编号

vendor\_no：供应商编号

warehouse\_no：仓库编号

carrier\_no：运输车队编号

employee\_no：下单员工编号

shipped\_quantity：发货总数量

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/010\_课后作业：梳理出来你们公司可能需要的所有埋点/笔记.docx**

《010\_课后作业：梳理出来你们公司可能需要的所有埋点》

你们数据需求，数据业务你要了解；核心的产品和系统的功能和流程，完整的一个体验；你需要从你们的产品或者系统里采集哪些数据出来，才能满足你们的数据需求，产出你们对应的数据报表

在你需要采集的那些数据里，哪些是需要通过埋点的方式，去采集用户行为日志出来，去进行报表的计算，有些公司的数据，可能压根儿就不是互联网公司，要采集的数据可能根本不是走用户行为日志埋点的方案的

写一些文档出来

（1）数据需求文档：整理一下，哪些报表，指标，每个指标的含义，看了以后用来干什么的，这些东西都搞清楚，那么你对你们公司的数据业务就很了解了

（2）产品功能和使用文档：整理一下，产品和系统，功能和运转的流程是什么

（3）数据采集文档：你需要从产品和系统里，采集哪些数据回来，其中哪些数据是通过日志埋点的方案来采集的

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/017\_业务数据同步方案：如何根据delete事件在hive中删除数据/笔记.docx**

《017\_业务数据同步方案：如何根据delete事件在hive中删除数据》

全量表里 增量表

id value id value event

1 100 1 120 update

2 200 2 250 update

3 300 3 null delete

4 400 insert

1 110 update

id value id value event

1 100 null null null

2 200 null null null

3 300 3 null delete

最终的结果如下：

1 120

2 250

4 400

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/001\_对未来2年的真实项目和生产环境下的实战安排和规划/笔记.docx**

《001\_对企业级数据采集平台的落地实战的安排和规划》

我不打算在三四个月的面试突击中进行落地实战的，按照项目的级别，梳理一些数据报表的需求，然后去梳理数据采集的方案，日志埋点方案，mysql binlog同步方案 ，构造出来大量的模拟数据，把数据导入到hdfs上去，建好对应的原始的hive表

起码1个多月才能搞出来

面试的时候对数据采集这块也不是重点，如果你能把我讲解的思路都理解了，然后把我留下的课后作业都搞定了

三四个月的大数据面试突击之后，会进入正式的为期2年的大数据架构师的大型项目实战的环节，三四个月，我会部署和做好真实的生产环境，我会在阿里云搞一堆真实的生产服务器，落地和部署数据采集平台，做一些源码的二次开发

把大量的数据通过数据采集平台，进入到我们的hdfs、hive存储层去，用户行为日志，业务库的数据，会完全基于真实的复杂的大型电商的APP+网站+小程序的业务，根据那套业务，采集真实的用户行为日志和业务数据，进入大数据平台

大型的实战项目

完全真实的环境，可以让大家一起在上面进行操作和实战的，我就是部署了一套跟公司里一样的大数据的环境，有真实的机器，你们每个人就是咱们这个虚拟公司里的一个工程师，在真实的大型项目背景之下，在真实的生产环境里，做各种开发、测试和部署

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/012\_课后作业：梳理一下基于业务数据你们公司需要哪些报表？/笔记.docx**

《012\_课后作业：梳理一下基于业务数据你们公司需要哪些报表？》

据我所知，大数据同学，有一部分是在互联网公司里，一部分是在传统IT厂商里，电信，是在传统公司的大数据团队里，某个产业/行业的公司的大数据团队里，五花八门，什么都有

作业：把你们基于业务数据的报表，补充到你们的数据需求文档里去，有哪些报表，有哪些指标，每个指标的含义是什么

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/014\_业务数据同步方案：拉取全量数据到hive中/笔记.docx**

《014\_业务数据同步方案：拉取全量数据到hive中》

刚开始你准备采集一个业务库的数据了，全量+增量，两个环节，你可以用一个程序基于select一下子在凌晨的时候，业务库负载很低的时候，几乎没人用的时候，把业务库里的数据都查询出来，慢慢的灌入到hdfs里面去

基于hdfs数据构建最原始的hive表

mysql里有一张表，有13581条记录；hive里有一张表，有13581条记录

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/009\_课后作业：摸透你们公司的APP、系统、网站等产品的功能流程/笔记.docx**

《009\_课后作业：摸透你们公司的APP、系统、网站等产品的功能流程》

数据需求，报表需求，你们必须对你们的数据的来源，APP、系统、网站、第三方购买的数据、工厂的机器产生的数据，了解的很清楚，另外对你们数据的来源，你要去熟悉和了解那个里面的一些功能和流程

是给人用的，体验他使用过程中的功能和流程；是给系统使用的，了解你们的系统的功能和使用的流程

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/020\_下周课程预告：离线数仓建模理论讲解/笔记.docx**

《020\_下周课程预告：离线数仓建模理论讲解》

假设我们有一些数据报表的需求，假设我们已经采集了一些数据到hive里去了，有一些原始数据的表了，此时我们如何基于一些用户行为和业务数据的原始数据，一步一步的去进行数据仓库建模，ETL开发，数据治理，生产问题和一些技术方案，生产调优

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/006\_后端埋点：精准采集用户与系统的交互行为/笔记.docx**

《006\_后端埋点：精准采集用户与系统的交互行为》

比如下订单，加购物车，支付，退款，评论，晒图，社交，用户与系统的交互行为，你可以让后台的系统发送用户行为日志到你的数据采集系统里去，一般就不用这么来用了，都会反映在业务数据库里

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/003\_为什么要通过埋点采集用户行为日志？/笔记.docx**

《003\_为什么要通过埋点采集用户行为日志？》

接入数据，老板要看数据报表了解公司的经营情况，财务部门需要看数据报表对公司的财务情况有一个了解，产品经理需要看APP上的用户使用的情况，对自己的APP产品设计有一定的了解，运营专员需要了解用户的点击、购买、留存、活跃，辅助自己的运营工作

任何一个公司内部的大数据团队，他们服务的对象就是公司里的各种人，老板，产品，运营，销售，财务，中层管理。第三方大数据厂商，开发一些大数据的平台和产品，你们公司不需要有完整的大数据团队，直接用我开发好的一套系统，把你们的数据接入到系统里去，就直接可以看数据了

20个人，平均每个人年薪20万，养这么一个团队一年光是基本工资就要400万，五险一金，工位，福利，五六百万，六七百万，这还只是一年！无非就是每年产出一大堆的数据报表，维护自己公司的数据，给公司各种人看数据

让公司上上下下可以根据数据来进行决策，调整高层战略，优化APP的用户使用，提升一些活动的效率，看到一些数据

把我们开发好的完整一套大数据系统和产品提供给你，每年只要20万就可以了，把你们数据接入进来，定制需求好商量

都是一些互联网公司，APP、网站一些东西，面向互联网上的人和你的用户，用户是你的核心资产，才能有接下来的一些商业模式，投放广告，电商卖产品，付费会员，老板，必然要关注一些用户相关的数据

总共有多少注册用户？每天有多少用户活跃在我们的平台上？平均每个用户在我们的APP每天使用多长时间？1000万注册用户，50万用户，30分钟，10万会下订单，平均每个订单大概多少金额，100块，每天大概就有1000万的营收，每个月大概是3亿的营收

财务统计出来的一些成本，商品成本，员工工资，经营成本，每个月以及每年的毛利润有多少，净利润有多少

战略有关系，运营总监，每天增长10万个注册用户，每天的广告投放的成本大概是10万块，12万，15万个注册用户呢？再多设计一些运营活动，比如搞一些噱头的营销，发一些广告邮件和短信，每天让100万用户来使用APP。产品总监，1小时，20万订单

金额往上涨，120块

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/004\_最low的一种埋点技术方案：基于SDK的代码埋点/笔记.docx**

《004\_最low的一种埋点技术方案：基于SDK的代码埋点》

比如说对于网站，你在里面嵌入一个js脚本，对页面上的各种组件（按钮、下拉框）加了一些事件监听器，比如说你点击按钮，js就会触发事件，把他封装成一条用户行为日志，通过ajax，异步的发送到你的后台服务器来了，http请求

APP，android，ios，嵌入一个sdk，负责进行android/ios开发的同学，你得跟人家交代一下，哥儿们，帮我采集一些用户行为日志，你能不能给你的android/ios上的界面的组件（按钮、下拉框、页面点击和浏览），加一些事件监听，比如触发一些事件的时候

你封装一个用户行为日志，用户id/android/app版本号/地理位置/做了一个什么操作/之前做了一个什么操作/操作对应的一些详细的数据（商品详情页id）/封装一个日志序列号和id，发送到后台来

增加埋点，修改埋点，开发的一段代码，必须等到app下一次发新版本，把新的埋点的代码带上去，让用户来更新的app版本，才能采集到最新的日志

埋点不会变更的那么的频繁，特别是对于一些中小公司而言，你只要刚开始采集到完善的日志之后，后续修修补补就可以了

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/011\_课后作业：自己写代码模拟埋点日志进入数据采集平台/笔记.docx**

《011\_课后作业：自己写代码模拟埋点日志进入数据采集平台》

数据需求、产品流程、数据采集，自己写一些代码，模拟自己是一个网站或者是一个android，把一些构造好的用户行为日志的数据，通过HttpClient这样的组件，直接http请求发送到你搭建好的数据采集平台里去

hdfs看过底层源码了，hdfs对你来说应该是透明的东西了，搞的很神秘，说是存储大数据的，无非就是一个大文件里面有10GB的数据，拆分成N多个256MB的小块，每个小块就是一个小文件而已

这些小文件就被分散在各个机器上而已

你不停的发送用户行为日志过去，通过flume就会不停的追加写入到hdfs的文件里去，就是在hdfs的机器上的block文件里不停的追加写入一行一行新的用户行为日志而已，没有什么神秘的

作业：把你自己想要采集的数据，你没有摸过你们公司的数据采集系统这块，模拟构造出来你要采集的各种用户行为日志，在不同的事件下发生的一些日志，发送到数据采集平台里去，观察hdfs和kafka里的数据不停的增多

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/019\_本周课程总结：量很少，但是作业完成量很大/笔记.docx**

《019\_本周课程总结：量很少，但是作业完成量很大》

承接日志采集平台的项目，你的数据需求有哪些，要从哪里采集数据，用户行为日志要做哪些埋点，业务数据库需要接入哪些，埋点的技术方案，业务数据同步方案，全量+增量+merge，技术方案都定下来

构造用户行为日志进入数据采集平台；构造一些业务数据进入数据采集平台，进行merge

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/007\_埋点之前：搞清楚用户行为数据用来出哪些报表？/笔记.docx**

《007\_埋点之前：搞清楚用户行为数据用来出哪些报表？》

现在采购第三方厂商的日志埋点的产品

需要去搞明白，自己的公司，有哪些报表的需求，搞明白报表的需求之后，埋点的方案也定了，自己搞，第三方厂商的产品，都可以根据报表的需求去制定要采集哪些用户行为的日志回来

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/008\_课后作业：想办法梳理清楚你们公司基于埋点数据要看哪些报表？/笔记.docx**

《008\_课后作业：想办法梳理清楚你们公司基于埋点数据要看哪些报表？》

每个大数据方向的同学，无论你当前在什么行业的公司里，你当前在大数据团队里是什么角色，负责的是一些平台开发，留一个作业

你需要去看看，你们公司现在就已经有数据需求 -> 数据报表 -> 结合你们公司的行业去搞清楚和理解每一张报表有哪些指标，每个指标背后的含义是什么，代表了什么，有什么作用

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/016\_业务数据同步方案：增量数据与存量数据merge/笔记.docx**

《016\_业务数据同步方案：增量数据与存量数据merge》

比如说我有一张表，第一次从mysql里全量同步到hive里，是有13581条记录；第二天的增量数据，就是对这张表进行的增删改的binlog，在hive里的一个增量表，比如说有1250条binlog记录

mysql，对其中800条数据进行了更新，对20条数据进行了删除，新增了500条数据

hive里的全量表和增量表进行merge，id value

对于增量表需要进行去重处理

id=1 110 update

id=1 120 update

增量表

id value timestamp

1 110

1 120

2 450

4 80

按照id进行group by，提取出来一个id下的记录里最新的那条数据，按照时间戳来排序，取出来里面时间最后的一条数据就可以了

全量表 增量表

id value id value

1 100 1 120

2 500 2 450

3 200 4 80

对两个表进行right join，右连接，就算右边表里有的记录join不上，也会保留他自己

id value id value

1 100 1 120

2 500 2 450

null null 4 80

连接以后的结果

id value

1 120

2 450

4 80

左外连接

id value id value

3 200 null null

他是没有变化的数据

union all

sql语法有很多种

id value

1 120

2 450

3 200

4 80

写入到hive全量表里去，今天就把数据进行了一次更新，此时的数据就代表了你昨天一整天的数据都合并到hive表里去了

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/013\_课后作业：梳理一下你们现在业务数据库有哪些表？/笔记.docx**

《013\_课后作业：梳理一下你们现在业务数据库有哪些表？》

业务系统有哪些核心的接口和功能，业务系统他对于各种核心接口和功能进行处理和运转的基本流程，了解一下，现在业务系统有哪些库和表，有哪些业务表，对于每个业务表里的字段的含义你了解吗

对于每个业务表里的一些数据的更新操作逻辑，你了解吗

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/005\_一种比较高逼格的埋点方案：可视化动态配置埋点/笔记.docx**

《005\_一种比较高逼格的埋点方案：可视化动态配置埋点》

大公司里，还有第三方日志埋点的厂商，无痕埋点，你不用说自己去为了埋点去做一些开发，在后台配置一些埋点信息，比如要在哪个按钮被点击的时候去发回来一个用户行为的日志，要采集哪些信息

网站/app，他们里面会嵌入一个sdk/js，他们会自动从后台拉取最新的埋点配置，然后他们会自动扫描网站/android/ios里的UI组件，加了一个新埋点，可以自动的根据新埋点，在UI组件里加一个埋点事件，自动把埋点就加上了

如果自己做

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/002\_课后作业：自己参照课程讲解搭建数据采集平台/笔记.docx**

《002\_课后作业：自己参照课程讲解搭建数据采集平台》

对于一些已经有几年经验的大数据工程师而言，关键的倒不是一些基础的技术带着慢慢做，欠缺了一些思路，架构的思路，技术方案的思路

数据零丢失方案，我有一个朋友，在狸猫技术窝公众号上出了一个专栏：《从0开始带你成为消息中间件实战高手》，基于RocketMQ完整的分析了MQ数据零丢失的方案，Kafka消息零丢失

要求他ISR列表里有超过1个副本存在，每次写入请求，必须是至少2个副本都同步成功，然后才能判定一次写入成功，kafka本地每次写入消息都同步刷到磁盘里去，消息一定是可以成功的投递到kafka去

如果上述条件不满足，不停的重试，重试多次之后，可以把消息写入本地磁盘，或者是写入db里去，后台线程随时可以搂出来再去进行重试

consumer消费数据，必须是你处理完一波数据之后，接着才能手动提交offset

架构设计的东西，你去考虑一下，如何基于他们的扩展机制，去实现一些高可用、高容错、数据不丢失的一些方案

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/015\_业务数据同步方案：基于canal采集增量数据/笔记.docx**

《015\_业务数据同步方案：基于canal采集增量数据》

23张表，每张表都直接全量导入到hive表里，记录数都是一样的，就开始做增量导入，通过canal接入表的增删改的binlog数据，对于一条数据，insert -> binlog，update -> binlog，delete -> binlog

我们需要在hive里对于每天收到的增量数据，做一张增量表，仅仅是存储每天数据增删改的binlog数据，insert、update、delete语句的binlog

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/001~020资料/018\_课后作业：自己实现基于canal的业务数据采集/笔记.docx**

《018\_课后作业：自己实现基于canal的业务数据采集》

你可以建一堆跟你们业务库一样的表，灌入进去一些初始化的数据，然后一次性把这些数据写代码查出来灌入hive里的全量表里去。然后写一些模拟的代码对这些表进行增删改，基于canal去采集增量的binlog，都写入到一个Hive的增量表里去

可以去写一个spark sql，去完成每天的全量表+增量表的merge的大SQL，里面会有很多的子查询，先过滤那些被删除的数据的，再merge update和insert

1月1号有一份全量数据导入hive，10000条就来，1月2号采集了1天的增量数据有1000条binlog，1月3号的凌晨，就把全量数据表 + 1月2号的增量binlog进行merge，拿到一份最新数据是10500条数据，灌入全量表

1月3号继续采集增量的binlog写入到1月3号的hive增量表里去，1月4号的凌晨，在把10500条数据的全量表和1月3号的增量表进行merge

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/073~082资料/078\_新零售场景下的dwt层的订单主题大宽表的设计/笔记.docx**

《069\_新零售场景下的dwt层的订单主题大宽表的设计》

设计每个维度表，都是从一大堆的数据需求来倒推，高层和中层会对你提出很多的数据需求，经营分析、商品分析、门店分析、促销分析，反应在一个数据系统里，应该是对应了多个菜单的，每个分析模块里，都有一大堆的报表

商品数据报表、品牌数据报表、类目数据报表，很多个报表

ods -> dwd + dim，全部完成了

dwt，dwd和dim做很多的join，就是一个超级大宽表，订单数据 join 日期维度 join 商品维度 join 门店维度 join 促销维度，每一个订单主题表里的每一行数据，可以看到这个订单的日期维度字段，门店维度的字段，订单条目主题表里也是一样，他可以里面包含很多订单字段、商品字段、促销字段

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/073~082资料/076\_新零售场景下的促销维度表应该如何设计？/笔记.docx**

《067\_新零售场景下的促销维度表应该如何设计？》

促销活动维度数据，新零售公司的业务系统里来的，商品系统、订单系统、促销系统、门店系统、仓储系统、采购系统、物流系统、配送系统、库存系统，正是因为现代的企业运行，都是需要一系列的系统来支持，企业运行过程中一切东西都是信息化的，会产生大量的数据，此时才会有数据需求

门店要搞一个促销活动，在促销系统里一定会有一条记录，就是代表了一次促销活动，或者是公司整体搞的一次促销活动，也会在促销系统里会有一条记录，从促销系统里抽取出来的所有的促销数据，就是促销维度表

门店维度表，促销维度表，这些数据都是可能会变化的，变化的比较缓慢，每天凌晨的时候把当天的门店表和促销表的数据，一次性同步到ods层的维度表来，insert overwrite，直接全量覆盖，有了最新的维度数据，才会去进行后续的ETL计算

promotion\_id：促销活动主键

promotion\_no：促销活动编号

promotion\_name：促销活动名称

price\_reduction\_type：促销降价类型，优惠券，满减，全场折扣，等等

promotion\_media\_type：促销活动的宣传途径，APP上广告，其他渠道广告投放，门店广告展示，门店外发传单，等等

advertisement\_type：广告投放类型，如果是投放了广告的，那么是通过什么方式投放的，比如说电梯广告，地铁广告，公交站台广告，电视广告，视频网站广告，等等

advertisement\_media\_name：如果是投放了渠道外广告，那么使用的渠道名称是什么，比如某个视频网站的名称，或者是某个城市的公交站，等等

display\_type：广告的展示类型，比如说是普通的产品介绍，或者是明星代言，或者是红包引流，或者是自编短剧

coupon\_type：如果是发放的优惠券，那么是什么样的券？满减？直接抵扣？赠送？

display\_provider：这个一般做一些渠道外广告投放，都有对应的第三方合作公司的，是哪个公司提供的这个服务

promotion\_cost：营销活动的花费

promotion\_begin\_date：活动开始时间

promotion\_end\_date：活动结束时间

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/073~082资料/077\_新零售场景下的其他维度表的介绍和讲解/笔记.docx**

《068\_新零售场景下的其他维度表的介绍和讲解》

dwd\_order：订单融合明细表

order\_no：订单编号

buy\_channel：购买渠道，线下门店/线上APP，退化维度，APP、STORE、无人便利店，中台思想，新零售业务，商品都是这些东西，下单、支付、促销，一系列的东西都是可以复用的

mobile\_phone\_os：手机操作系统，android/ios

app\_version：APP版本号

location：地理位置，经纬度，就可以了

user\_id：用户id，维度表，表设计，后续在其他的案例里给大家说一下

store\_no：门店编号，线上和线下都需要有一个门店编号

check\_out\_type：收银方式，收银台/自助收银机器

cashier\_no：收银员编号，也可以设计一个维度表，也应该从业务系统里抽取过来

self\_service\_cash\_register\_no： 自助收银机器编号

product\_count：总购买商品数量

total\_amount：总购买商品的金额

discount\_amount：总优惠金额

payable\_amount：总应付金额

pay\_type：支付方式，维度表，合适一些

pay\_ time：支付订单的时间，2020-01-01 23:56:31

create\_time：订单创建时间

pay\_datekey：20200101

create\_datekey：20200101

dwd\_order\_item：订单条目数据

order\_no：这个订单条目所属的订单编号

order\_item\_no：订单条目的编号

product\_no：商品编号

promotion\_no：促销活动编号

buy\_count：商品购买数量

price：商品单价

total\_amount：商品购买总金额

product\_discount\_amount：商品分摊折扣金额

discount\_amount：折扣总金额

payable\_amount：应付总金额

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/073~082资料/081\_作业：从公司需求反向驱动，完成数据仓库建模/笔记.docx**

《072\_作业：从公司需求反向驱动，完成数据仓库建模》

也做过一些数据仓库，搞的非常的不正规，其实就是自己知道一些数据需求，自己野路子，想办法，去做一些数据仓库建模，分层，ETL理解，数据建模的理解，都比较浅显，而且没有一套完整的方法论去支撑

数据需求，维度建模4个步骤，把你数仓里的所有的ods -> dwd + dim，都设计出来，再考虑dwt主题宽表，dwa轻度聚合，app层的报表是如何通过dwt和dwa跑出来的，这套东西想清楚了

动态运行起来的效果，ETL内部的逻辑，ods层的各个表是如何抽取，增量，全量覆盖，从业务系统抽，一般都是全量抽数据，全量覆盖，全量重跑和计算

底层技术架构，数据采集的技术架构，数据仓库底层的技术架构（hadoop + hdfs + hive），ETL底层的技术架构（hive/spark sql -> flink sql），ETL作业调度的技术架构（自研，用的开源的），最终的报表数据存储的底层架构（mysql，hbase，es，clickhouse），数据可视化的技术架构（开源的、商业的、自研的可视化BI平台）

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/073~082资料/080\_新零售场景下的app层的数据需求报表的设计/笔记.docx**

《071\_新零售场景下的app层的数据需求报表的设计》

你只要是玩儿大数据的同学，在你公司里一定都有数据仓库，除非说你是专门做大数据工具的公司，智能BI系统的一些公司，你公司本身就自然而然的就是有对应的一个数据需求，你自然就知道你的app层的报表有哪些了

数据产品经理，数据分析师，他们是在业务和技术之间的桥梁，他们会深度理解业务，然后理解业务的需求，跟你的大数据的最终的公司内部的用户进行沟通，知道他们想要看什么样的数据，然后这些数据产品经理，未必懂技术，但是懂数据

理解你的ods层的每一张张表的字段的含义以及他计算出来的一些逻辑

提出需求，希望你基于你有的ods层的数据，产出一些他想要的数据报表，数据报表里的每个字段他都会定义清楚，包括每个字段的一个计算逻辑都会告诉你，只不过具体的底层技术的过程，包括数仓建模，ETL，性能优化，大数据上的一些技术栈的使用

数据仓库工程师，数据开发工程师去做

作为app层的各种报表的定义就可以了，也是非常简单的

真正在做数据仓库建模的时候，数据产品经理 / 数据分析师，提出一堆的数据报表的需求，此时你完全可以先设计app层的报表的模型，再去设计ods -> dwd + dim -> dwt -> dwa，内心要大概知道从dwa和dwt到你的app，ETL逻辑

接下来很简单，就是去写ETL，做ETL的测试，看看你跑出来的数据是否符合你的期望，正式部署ETL的定时调度，基于底层的hive/spark/flink去跑就可以了，每天定时去抽取数据到ods，跑一系列的ETL，最终数据产出到mysql/es/hbase/clickhouse，再有报表系统去可视化的展示这些报表出来就可以了

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/073~082资料/073\_为什么需要门店维度，需要去支撑哪些数据需求？/笔记.docx**

《064\_为什么需要门店维度，需要去支撑哪些数据需求？》

日期维度，商品维度

基于日期维度去统计各种不同的日期范围内的销售额之类的，这个东西仅仅是能够让高层看到整个公司的整体一个情况

商品维度，可以让中层管理人员，比如说商品部门的负责新品选品的人，可以去观察各种商品的在不同的日期维度内的销售额，可以适合总部的一些商品部门的人去分析这些数据

新零售，有线上APP可以卖，也有线下的门店，在线上的APP下订单，都是需要选择一个距离你最近的门店的，线上APP和线下门店下的订单几乎没什么区别，只不过说线下，自己选购，在人工收银台或者自助收银机器上，付钱

线上，也得选择一个门店，门店里的商品给他信息化了，放在APP里你可以看到，选购那个门店里的商品，选购了之后就可以下订单和支付，线下的自助收银机器，你自己把商品在机器那儿扫码，确认生成订单，按钮，此时就会生成一个订单了，在这个订单的界面，还可以对商品进行调整

对这个订单进行支付，新零售场景下，哪怕是线下，他也有订单和支付的概念

门店维度，圈定就只看我自己这个门店的数据

总部而言，门店就是一个地理位置，上升到市，某个区，某个省，区域，华北、华东，我就是想看看某个市的销售情况，某个省，每个省的销售额，每个区域，东北，西北，华北，华南，销售情况

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/073~082资料/082\_下周课程预告：高阶数据仓库建模技巧讲解/笔记.docx**

《073\_下周课程预告：高阶数据仓库建模技巧讲解》

高阶数据仓库建模，缓慢变化维，拉链表，雪花模型，星型模型，数据集市，数据治理（元数据的维护和管理，数据质量监控）

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/073~082资料/074\_新零售场景下的门店维度表应该如何设计？/笔记.docx**

《065\_新零售场景下的门店维度表应该如何设计？》

门店数据是从哪儿来的？必然是从业务系统里抽取出来的，新零售公司自己肯定有一套复杂的业务系统，包括了线上和线下，收银机器里面下订单和支付，都要走业务系统，收银员的POS枪扫码和支付，收银系统也得走业务系统，线上APP，全部都是走业务系统

对于业务系统里还应该包含了大量的专门给公司内部运营人员使用的功能，包括供应链，门店，商品，创建门店，可以在门店里维护商品，供应链的一些东西，包括下单、菜单、仓储、库存、物流

门店数据，都是从新零售公司的业务系统里抽取出来的

store\_id：门店主键

store\_no：门店编号

store\_name：门店名称

store\_street\_address：门店街道地址

store\_district：门店所在区

store\_city：门店所在市

store\_province：门店所在省

store\_region：门店所在国内区域，华东，华北之类的

store\_zip\_code：门店所在地址的邮编

store\_manager：门店店长

selling\_area：销售区域面积

total\_area：总区域面积

open\_date：开店日期

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/073~082资料/079\_新零售场景下的dwa层的订单相关指标轻度聚合表的设计/笔记.docx**

《070\_新零售场景下的dwa层的订单相关指标轻度聚合表的设计》

基于dwt去做一些轻度聚合，比如说对订单销售相关的一些常见指标做一些轻度聚合

日期维度、商品维度、门店维度、促销维度

group by 日期（每一天）、store\_no、product\_no、promotion\_no

看到每一天每个门店每个商品在每种促销活动下的各种销售指标，轻度的聚合和汇总，比如说本来订单条目主题表里，每天可能是100万订单，10个条目，1000万条数据，数据量其实比较大的

dwa层的轻度聚合表里，100个门店，1000个sku，10万条数据，5种，几十万条数据而已，100 \* 1000 \* 5 = 50万条数据

app层的报表的时候，优先基于dwa来出，已经聚合过了，包含了一些常用的维度，ETL的性能会很高，但是如果dwa层出不了你要的报表，就尽量从dwt，每天有1000万数据，但是提前都join过了，是大宽表，包含了大量的字段

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/073~082资料/075\_为什么需要促销维度？基于促销维度分析的大量数据需求举例/笔记.docx**

《066\_为什么需要促销维度？基于促销维度分析的大量数据需求举例》

（1）针对某个商品搞了一个促销活动，在促销期间，商品的销售额得到多少提升？

（2）促销前和促销后的商品销量对比，促销期间的销量提升是否冲抵了日常销量？也就是说，你促销的时候卖的很多，但是促销之后销量大幅度下跌，那么其实本质上卖的销量是一样的，结果促销的时候利润还更低，其实你亏了

（3）跟促销商品同类的商品是不是销量降低了，比如说大家都去买这个降价的薯片了，其他品牌的常价的薯片就没人买了，这个时候其实你搞促销也是没意义的，这叫做“销售侵蚀”

（4）有促销活动的商品，他的利润到底是否得到了增加，考虑促销前，促销后，促销期间，最后看一段时间里的总销量，计算他的利润，跟之前进行对比，看促销是否扩大了商品市场占有率以及利润

（5）同时考虑没有促销活动的商品，是否有销量和利润的下降，跟有促销活动的商品的利润提升一起考虑一下，门店整体的利润是上升还是下降

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/083~093资料/090\_库存数据的事务事实表能支持哪些数据需求呢？/笔记.docx**

《090\_库存数据的事务事实表能支持哪些数据需求呢？》

事务事实表，用户行为日志，一方面是可以做一些大的统计，对他的各种不同类型的用户行为进行一定的统计分析，每天有多少用户登录APP？有多少人对首页进行了下拉浏览呀？会选择搜索商品的有多少人呀？平均每个人会搜索几次商品呢？诸如此类，其实对他们的行为可以进行一定的统计和分析

平时进入APP，立马跳出的用户，或者在APP里停留时间不超过10分钟的用户，筛选出来，直接分析他们在APP里的浏览路径，第一步干了什么，第二部干了什么，就要把这个用户在这一天，他所有的用户行为日志都查出来

库存数据，事务事实表，通过监听仓储系统的mysql binlog变更，自己识别出来各种库存变动的事务事实，每天有多少商品会有进货到仓库，每天有多少商品会出现质检不通过退回，每天有多少商品会从仓库里发货到门店

对某个商品的库存有异议，追踪一下这个商品过往的事务事实，此时就可以把这个商品的在指定日期的事实事实都查出来，一点点分析

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/083~093资料/087\_布置作业：在自己公司业务里找到一个周期快照事实表场景/笔记.docx**

《087\_布置作业：在自己公司业务里找到一个周期快照事实表场景》

参考库存业务模型，每天都有一个库存快照，在自己公司的业务里找找类似库存业务的周期快照事实表的场景，有没有什么数据，你们是要去分析他每天的快照，就算是暂时没人提这个需求，你也得给他找一个场景，自己思考如何做，如何设计，支撑什么需求，底层的技术细节怎么实现

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/083~093资料/088\_库存数据的事务事实表如何设计？/笔记.docx**

《088\_库存数据的事务事实表如何设计？》

周期快照事实表，挺有用的，在数据分析的时候可以观察数据的一个变化，出一张报表，每个门店每个商品库存周转的周期

datekey：日期，某一天

product\_ no：商品编号

warehouse\_no：仓库编号

inventory\_transaction\_type：事务类型

inventory\_transaction\_no：这个事务的序号是第几个事务

类似的，其实就是用户行为日志，这都属于事务事实表

供应商把商品送入仓库，商品放入检验区，商品在检验区完成质检，将质检有问题的商品退回，质检通过的商品入库，根据门店的补货单从仓库里提取一批商品，对商品进行打包，把打包的商品发到门店去，门店检查商品，门店退回检查不过关的商品，门店对检查过关的商品正式入门店仓库

事务事实表，特别跟什么类似呢？特别类似于互联网行业的用户行为日志，可以把用户行为日志认为是一种事务事实表，登录APP、浏览首页、搜索商品、点击商品、加入购物车、下订单、完成支付、发表评论

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/083~093资料/083\_从新零售公司的库存数据分析需求出发/笔记.docx**

《083\_从新零售公司的库存数据分析需求出发》

两周的时间，给大家讲解一些工作的时候常用的数仓建模的高阶技巧，一个是这周讲解一些高阶的事实表设计的一些技巧，用新零售公司的库存数据分析来作为案例讲解，另外一个下周给大家讲解一些高阶一点的维度建模的技巧，缓慢变化维这一块

拉链表，高阶的建模技巧

一周，给大家讲解一些企业级的数据治理的思路和方法

离线数仓这块，在建模这块，就把很多人的短板给补齐了，正规军的数仓建模的思路，分层、包括需求反向驱动、数据采集、ods、每一层的设计、高阶的事实表设计、高阶的维度表设计、高阶的数据治理的思路和方法

跟Java的面试突击是一样的，不会过度的牵扯技术细节，主要是讲解一些高大上的一些方案，架构设计，自己跟项目融合，离线和实时两块的项目，就有亮点可以包装了，讲解的这些东西同时也面试高频常问的

kylin、clickhouse、druid、数据湖，你自己了解一下，不是面试必问的

从5月份开始，就是正儿八经的做大型的项目，一个大项目就做三四个月，课程量还会增加，每周的课程量是很少的，布置了很多作业，之前有大量的源码级的技术，还有很多的java课程

再做两年的项目，一边讲解对应你需要的技术

公司希望对库存数据进行分析，分析每个门店的每个商品每天的库存情况，比如说一个商品进货之后，库存在多少天会卖掉，会不会出现说脱销的情况，会不会出现说库存过多导致商品变质的情况，导致库存仓储成本过大

由此推断出每个门店每个商品的进货周期和进货数量，保证不会脱销，不会过期变质，同时避免出现过长时间的库存商品积压，导致库存仓储成本过大

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/083~093资料/092\_累积快照事实表的数据如何抽取？/笔记.docx**

《092\_累积快照事实表的数据如何抽取？》

我们要的数据，其实说白了，还是在仓储系统来进行抽取的，监听他的mysql binlog，有对应的一些库表更新的时候，你自己识别出来发生了一个什么事情，把日志发送到hive的ods层进去就可以了

在ods层的表里，只能看到的是事务事实数据

每一次事件是一行

每一次事件是一行

在dwd层可以建立一个库存的累积快照事实表，ETL

去识别每一行事务，如果发现有一个商品批次的一个关键事件，此时就得把这个事件，在ETL里走一个叫做多行转一行里的多个列，行转列，写ETL的时候有一些高阶的SQL技巧，百度一下，行转列hive语法技巧

一行里的多个列，转成多个行，列转行

product\_batch\_no：商品批次编号

warehouse\_no：仓库编号，在哪个仓库里的

vendor\_no：供应商编号，从哪个供应商那儿进货过来的

received\_datekey：这批商品到仓库的日期

inspected\_datekey：这批商品进行质检的日期

bin\_placement\_datekey：这批商品质检过后入库的日期

initial\_shipment\_datekey：这批商品第一次发货给门店的日期

last\_shipment\_datekey：这批商品最后一次发货给门店的日期

quantity\_received：这批商品到仓库的时候有多少件

quantity\_inspected：这批商品进行质检的时候有多少件

quantity\_return\_to\_vendor：这批商品质检完了以后有多少件退回给供应商了

quantity\_placed\_in\_bin：这批商品质检完以后有多少件入库了

quantity\_shipped\_to\_store：已经发货给门店的商品数量

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/083~093资料/093\_累积快照事实表可以用来满足什么样的数据需求？/笔记.docx**

《093\_累积快照事实表可以用来满足什么样的数据需求？》

如果对一个商品批次关键库存变动的数据，都可以在一行里追踪到的话，周期快照事实表，其实是可以知道的是他在门店里售卖的一个情况，多长时间可以周转完毕，累积快照事实表，他主要分析的是商品的库存在我们公司的总仓库里的变动情况

每个门店的店长而言，更多的是需要周期快照事实表，分析的是这个门店里的每个商品的每天的库存快照的变化，来决定一个商品的进货，每个门店都有自己的一个后面的小的储藏室，一般不会太大的

对总公司而言，他会统一采购商品，放入自己的仓库里，累积快照事实表，其实是从商品采购到仓库开始，包括后续不停的发货给各个门店，用一行数据追踪一批商品在总仓库里的库存变化的情况

一批商品采购到总仓库里，一般会积压多少时间，总仓库里的库存周转情况

三种高阶的事实表，周期快照事实表，事务事实表，累积快照事实表，结合新零售的库存数据分析的案例，周期快照事实表主要用于门店的库存周转周期的分析，事务事实表有两个功能，一个是对商品库存变动的各种事件进行分析，一个是对有异常库存的商品进行库存变动事件的追溯

累积快照事实表，其实针对的是公司的总仓库里的库存周转周期的分析

有没有类似库存的业务，场景可以做的，支撑的数据需求都有什么，想也要给像出来，包装到简历里去，必须要有一些稍微高端一点的数仓建模的技巧

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/083~093资料/091\_库存数据的累积快照事实表如何设计？/笔记.docx**

《091\_库存数据的累积快照事实表如何设计？》

高阶事实表，周期快照事实表，可以追踪商品的库存变动；事务事实表，追踪一个商品的库存变动的所有事件对应的统计分析，可以追踪事务发生的链条；累积快照事实表，对一个商品的库存出现变动的一些关键事件进行一定的追踪

product\_batch\_no：商品批次编号

warehouse\_no：仓库编号，在哪个仓库里的

vendor\_no：供应商编号，从哪个供应商那儿进货过来的

received\_datekey：这批商品到仓库的日期

inspected\_datekey：这批商品进行质检的日期

bin\_placement\_datekey：这批商品质检过后入库的日期

initial\_shipment\_datekey：这批商品第一次发货给门店的日期

last\_shipment\_datekey：这批商品最后一次发货给门店的日期

quantity\_received：这批商品到仓库的时候有多少件

quantity\_inspected：这批商品进行质检的时候有多少件

quantity\_return\_to\_vendor：这批商品质检完了以后有多少件退回给供应商了

quantity\_placed\_in\_bin：这批商品质检完以后有多少件入库了

quantity\_shipped\_to\_store：已经发货给门店的商品数量

通过这个东西，就可以完美的对一个仓库里某一天进货的一批商品，后续的所有的库存变动在这一行里进行追踪和更新

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/083~093资料/086\_从ods层到dwd层：如何实现库存周期快照事实表？/笔记.docx**

《086\_从ods层到dwd层：如何实现库存周期快照事实表？》

ods层，每天都是把库存系统里的每个门店每个商品的迄今为止剩余的库存数据，一次性加载到ods层来

从业务系统抽取数据到ods层表的时候，走的是insert overwrite，直接全量覆盖了

XX门店 XX商品 65

ods -> dwd的ETL里，就需要做一些逻辑，把ods层的库存表里的数据查询出来，每条数据都添加一个当日的日期，2020-01-01，加上日期之后，插入到dwd层的表里去，追加写入，不是insert orverwrite覆盖了

dwd层，每天的每个门店的每个商品的库存快照作为一行

2020-01-01 XX门店 XX商品 100

2020-01-02 XX门店 XX商品 80

2020-01-03 XX门店 XX商品 65

按照这样的一个思路，就可以做出来一个库存数据的周期快照事实表

从业务系统里抽取的数据，事实表还是维度表，每天都是全量insert overwrite覆盖，后续的每一层的ETL都是全量重跑，跑出来所有的数据

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/083~093资料/089\_库存数据的事务事实从哪里抽取？/笔记.docx**

《089\_库存数据的事务事实从哪里抽取？》

我们现在想要知道的是一个商品在仓库里的各种事务，供应商把一批商品运送到你的商品里去，这个事务你是怎么知道的？这个事情会在你的仓储系统里会发生，会有一个进货单一样的东西，完成一个更新

质检区里去进行检验，在仓储系统里本身就会有

质检不通过的商品退回，质检通过的商品正式入库，这些事情行为在仓储系统里都会有一定的体现的，WMS系统里都是会有体现的

仓储系统里一系列的行为，都是去更新他自己的表数据，不一定能让你直接抽取到库存的事务数据的，一般都需要进行日志埋点，也就是说，你需要专门跟后端工程师约定好，他们开发WMS系统的

发生一件事情的时候，约定好，让他们发送一条消息到MQ里去，通知你说自己的一批商品到仓库了，这种方式不太靠谱

基于canal监听仓储系统的表binlog的变更，根据每个核心表的binlog来判定是否发生了某件事情，一批商品到仓库了，会对应着仓储系统里的一个进货单，此时仓库管理员就应该在仓储系统里对这批商品的进货单，点击一个按钮，标识他是商品到仓库

此时必然对应仓储系统对他底层的表结构进行一定的更新，就有mysql binlog，你监听到这条binlog，就把这条binlog给写入到hive表里去，通过flume去做一个采集，在ods层面有一个表，放了一大堆的类似的这种库存事务日志

数据库的binlog，update purchase\_order set product\_arrived=1 where id=1155

id=1155的这个采购单对应的一批商品，商品编号是什么，此时已经到仓库了，你应该转移为这样的一条日志，通过采集系统，采集到hive的ods层面的表里去

dwd层的etl，直接一跑，把ods层的数据，给搞到dwd层里去，就出现上一讲看到的事务事实表

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/083~093资料/085\_库存数据的周期快照事实表需要保留多久数据？/笔记.docx**

《085\_库存数据的周期快照事实表需要保留多久数据？》

100门店，6万个sku，每天600万数据，每个月1.8亿，每年接近20亿条库存的周期快照数据

数据分析需求来说，是每个商品每次进货之后，在一段时间范围之内，他的库存快照的变化情况，XX门店的XX商品，一次性进货1000件，后续开始每天的库存变化情况，大致可以分析下来，平均每天在消耗50件左右，20天左右会卖完

这个XX门店对这个XX商品，每隔15天左右补一次货，每次补货之后，大致需要多久卖完，中间肯定会保证一直有存活在仓储里，保证不会缺货脱销，而且不会有太多的积压货在仓库里占用成本

一般来说，对于这种周期快照事实表的数据，一般都是存放1个月~2个月左右的周期，就是让你观察数据的变化的规律，观察出来一个规律之后，就可以做一些业务上的决策了，根本没必要是长周期的

业务系统里每天抽取一份全量数据到ods层，但是我们在dwd层需要设计一个库存周期快照事实表，在这个表里保存的是每个商品每天的库存快照数据，保留最近1个月~2个月的库存快照的数据

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/083~093资料/084\_为库存数据设计周期快照事实表模型/笔记.docx**

《084\_为库存数据设计周期快照事实表模型》

新零售业务系统 -> 库存系统，存储的是每个门店每个商品到目前为止的一个库存的数量

store\_no product\_no quantity\_on\_hand

XX门店 XX商品 100

从业务系统里采集的事实数据，每天就是把库存系统里的库存表里的数据全量导入一份到ods层，ods层每天都是全量覆盖，拿到的都是这个门店的这个商品的今天为止的库存数量，是这样的

从ods直接同步到dwd，拿到的只有说每天每个门店每个商品的迄今为止的库存数量

在dwd层的周期快照库存事实表

datekey：每天一个日期

store\_no：门店编号

produce\_no：商品编号

quantity\_on\_hand：周期库存快照

2020-01-01 XX门店 XX商品 100

2020-01-02 XX门店 XX商品 80

2020-01-03 XX门店 XX商品 75

100，60000个sku，每天就是600万条数据，每个月1.8亿条快照数据，20亿左右的快照数据了，其实这个就没有必要了

每天每个门店的每个商品的库存数量

保留最近几个月的数据

几个月之前的都做成3个月周期

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/114~122资料/114\_如果你不做数据治理，会呈现怎样混乱的局面？/笔记.docx**

《114\_如果你不做数据治理，会呈现怎样混乱的局面？》

混乱的局面，主要在于以下几点：

（1）指标口径不统一：哥儿们A负责了一个报表的需求，total\_amount，指标，当日总营收，当日理论上应该收的金额，哥儿们B负责了一个报表的需求，total\_amount，指标，当日总营收 – 当日折扣金额，当日实际收钱的金额

最大的问题就在于说，此时万一说有别的人要用这个数据仓库里的数据，比如说后来新加入的工程师，或者是运营、PM想要自己看数据，或者是隔了几个月以后，你自己也要去数据仓库里找数据

昏了头，发现说各种表里有一样的指标的名字，结果呢，发现指标背后的含义不同的

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/114~122资料/118\_建立团队的建模与开发规范，标准化研发流程/笔记.docx**

《118\_建立团队的建模与开发规范，标准化研发流程》

建立一套文档，把仓库里所有的字段和指标的定义，都放在里面

1 当日公司总营收 total\_amount 当日公司整体的营收金额 当日营收金额 – 当日折扣金额

01\_需求分析规范：数据需求评审会议，数据需求产出的可行性的分析

02\_项目管理规范：定排期，做数据项目管理

03\_数仓建模规范：必须参考统一的指标和字段的定义文档，必须严格按照分层来设计，把这个需求需要的每一层的表以及大致的ETL逻辑，写清楚，如果是你自己新定义的一套字段和指标，必须补充和完善到统一的指标文档里去

04\_方案评审规范：召开一个方案评审会议，别人来把关，看你的数仓建模的设计，看你是否复用了统一的指标和字段的定义，分层是否符合规范，按照你写的一套ETL逻辑，能否最终产出需要的数据指标和报表，是否出现重复计算

05\_ETL开发、测试以及上线规范：建表，写ETL，测试一下ETL，部署到线上调度系统去上线，产出需要的一些数据了

06\_数据质量规范：每个报表都需要有一个对应的python脚本，每天在报表的指标算出来字后，都把报表当日最新的指标查出来，按照一定的规则来进行统计和比对，指标A+指标B=指标C，如果不满足公式，用python发送报警邮件就可以了

指标A：当日应收总金额

指标B：当日折扣总金额

指标C：当日总营收

指标A – 指标B = 指标C，满足这个公式，否则当日数据就错误了

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/114~122资料/119\_数据治理的平台化体系建设思路/笔记.docx**

《119\_数据治理的平台化体系建设思路》

针对已有的数据仓库，此时可以做一套元数据系统，可以自动从已有的数据仓库里抽取你所有的表和字段，你可以统一的对表和字段的含义，做一个定义，统一所有的指标和字段的含义

元数据，业务元数据，技术元数据

业务元数据：分层、表、指标和字段的一些定义

技术元数据：你的每个ETL，血缘关系，每个ETL一般来说要耗费多长时间跑完，每个ETL一般来说需要耗费多少资源，每个表里对应的一些数据量，每天的增量数据是多少

数据地图系统：用可视化的界面给你展示出来你的各种元数据，你可以看到你的数仓的分层，每一层的表，每个表的指标定义和含义，每个表的ETL，ETL之间的依赖关系，每个ETL耗费多长时间跑，耗费多少资源，表的数据量

搜你想要查找的指标、表、ETL，ETL依赖关系

数据建模系统，在每一层里建模你需要的表，表的字段可以直接选用已有的字段，也可以建立全新的字段，可以发起审批，可以看到这里的表定义，审批通过，此时就会自动给你进行建表，元数据自动录入到元数据系统里去

ETL开发系统，写ETL，对ETL进行测试，ETL进行上线，配置一系列的数据质量监控规则，元数据系统会自动抽取技术元数据过去

数据质量监控系统，自动会去根据你配置的规则检查你每天跑出来的数据是否准确

指标绝对不会混乱，清晰的分层和审核机制，重复计算，数据质量都有监控，此时一般来说数据是不会错误的

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/114~122资料/116\_如何一步一步的迭代团队的数据治理能力？/笔记.docx**

《116\_如何一步一步的迭代团队的数据治理能力？》

指标混乱、重复计算、数据错误

从刚开始建立，刚开始做数据仓库，就应该开始同步建立数据治理的机制

就是完全基于在线协作的文档就可以了，wiki、腾讯文档、钉钉文档，在线文档协作的商用软件多的是了，就可以开始做数据治理了

统一化所有指标的口径和定义，就应该搞一套文档，把你数据仓库所有的每个表的字段、指标，都给一个清晰的定义，total\_amount，公司当日总营收，当日应收金额 – 当日折扣总金额，就是这个指标

就直接用指标定义的这套文档去找了

定义套完整的数据建模和ETL的流程，数仓建模（按照标准的维度建模方法和步骤去做） -> 数仓建模评审 -> 再是建表 -> ETL开发 -> ETL测试 -> ETL上线 -> 数据质量监控

这套东西也是用文档来定义就可以了

数据安全如何保障，ETL资源如何分配，数仓模型的可扩展性，重复计算的消除

平台化的东西，把一些文档里的东西用平台和系统来做，把数据仓库的元数据都抽取出来，做一个元数据系统，数据地图系统；数据建模系统，在里面建模，每一层需要用哪些表，每个表的定义，自动发起评审，自动建表；ETL开发系统，开发ETL，做测试，上线；数据质量监控系统，配置一些数据质量监控的规则

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/114~122资料/122\_下一段阶段课程预告：Flink实战开发以及源码剖析/笔记.docx**

《122\_下一段阶段课程预告：Flink实战开发以及源码剖析》

flink从0开始入门到实战开发，flink核心源码深度剖析

比如说在实时数据里会大量的运用到hbase，分布式海量数据NoSQL存储系统，也可能会先讲hbase，再讲一下实时数据平台的架构设计

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/114~122资料/121\_真实的大数据项目生产环境以及数据平台的准备/笔记.docx**

《121\_真实的大数据项目生产环境以及数据平台的准备》

两三个月的时间，在准备真实的大数据项目的业务环境，生产环境，几十亿级别的数据量，真实的大数据平台的部署，真实的机器在这里，研发适用于所有同学同时在上面进行实战的平台系统

会有大量的需求文档，大型电商网站的大数据平台，里面报表都有上百张，几亿真实的电商平台的数据，全套完整的大数据平台，可视化界面，都可以直接在可视化的界面上，开发ETL，提交你的flink作业去跑，可视化的监控和调优

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/114~122资料/117\_建立团队的指标规范，统一对外数据口径/笔记.docx**

《117\_建立团队的指标规范，统一对外数据口径》

在线文档来搞，对所有的指标，包括每个表的重要的字段的定义，全部都抽取出来放在一套文档里面去

业务需求的时候，数据需求的时候，人家会给你提需求，数据产品经理，主要是跟用户打交道，了解他们的业务流程和需求，考虑说，要做哪些数据报表给用户来使用，满足他们的需求，协助他们提高业务的处理效率， 辅助决策，人家怎么来用

了解业务流程，可以做出来哪些数据指标，做一些数据报表的需求，每个报表有一些指标，每个指标的中文解释，计算的公式，都要给清楚的

这些数据需求里的指标，都需要放在一套文档里，定义好，包括每个指标对应的背后的英文名字，都要是统一的，肯定是要去进行数仓建模的，然后才能在底层进行技术的落实，包括定义好要抽取哪些ods层的数据，每一层的表如何建模，每个表有哪些字段

设计很多表出来，每个表里都有很多的字段，每个字段都需要有一个定义，就需要放到一套统一的文档里去，你的整个数据仓库里，每个字段的含义都是一样的，哥儿们A设计一个dwt层的表，商品id，他叫做product\_id

另外一个哥儿们，也是一个dwa层的表，商品id，做成了product\_no

代表同一个含义的两个字段，名字不一样；两个字段名字一样，含义不同，区分开来

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/114~122资料/115\_如果你不做数据治理，会呈现怎样混乱的局面？（续）/笔记.docx**

《115\_如果你不做数据治理，会呈现怎样混乱的局面？（续）》

（1）指标口径不统一

大家用的英文单词做的一个字段是一样的名字，结果呢，背后的含义不同

大家明明有一个指标应该是一样的含义，在哥儿们A设计的一张表，total\_amount；哥儿们B设计的另外一个表里，daily\_received\_money

（2）冗余计算

每个人都是各自为政，DWT，DWA，把一些需要做的关联和聚合，放在两层里统一来做就可以了，你没有必要在每个报表的ETL里（APP层），都重复的进行关联和聚合，这是没有必要的

瞎搞，哥儿们A，根本就不管什么DWT、DWA，直接就是写一个报表，里面干出来800行的大SQL，把各种join、聚合都搞了一遍；哥儿们B，也不管什么DWT、DWA，直接干出来一个500行的大SQL，各种join、聚合都搞了一遍

做报表的时候，直接就是基于DWT/DWA搞就可以了，大家不需要做重复的关联和聚合

（3）数据质量

跑出来的报表，数据可能都是错的，不同的人对你写的ETL到底有没有bug，算出来的数据到底是不是对的，你心里根本就没底的

指标混乱、冗余计算、数据错误

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/114~122资料/120\_本周作业：梳理自己公司的指标统一、流程规范、数据治理/笔记.docx**

《120\_本周作业：梳理自己公司的指标统一、流程规范、数据治理》

能够把自己公司已有的数据仓库体系里的指标和字段进行一个梳理，如果有重复或者混乱的地方，你要考虑一下应该如何统一规范

数据仓库从建模、开发、测试和上线，一系列的流程规范

数据治理：就写一个python脚本，加入到你的调度系统里去，每次报表的ETL跑完，就执行一个数据质量检查的python脚本不就可以了吗

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/051~062资料/062\_维度建模4步骤的案例实战总结以及作业布置/笔记.docx**

《062\_维度建模4步骤的案例实战总结以及作业布置》

生鲜新零售的案例，实战演练了一遍维度建模4步骤的过程，你以后在做任何数据仓库的需求的时候，只要拿到数据需求，一定要按照维度建模4步骤的过程，去选择业务过程、确定明细数据的粒度、确定维度、确定事实，设计ods、dwd两层的表

ods其实就是你要选择从哪些业务系统/APP上的用户行为日志，去抽取数据过来，数据源里的数据原样同步到ods层就可以了，原始数据，可能很混乱的

设计dwd层的表

字段名称和含义的标准化：ods层里可能各种不同的表，同样含义的字段，名称是不同的，同样名称的字段，含义是不同的

对于一些字段的值进行标准化处理：有一些字段的空值是“”，有一些字段的空值是NULL，你要对字段的值进行标准化的处理；支付方式，123456，alipay、weixin\_pay、bank\_card

一些测试的数据要进行剔除处理

对于同样业务领域的不同的子业务的表，可以考虑是否做融合设计，dwd层的数据仓库的模型尽量要保持简便清爽

然后你的dwd层的ETL，就是干上面的事儿的，从ods层抽各种数据，进行清洗和转化，落入dwd层的表，让dwd层的表保持非常的干净的一个状态

各种各样的维度表如何来设计

作业：自己结合自己公司的数据分析的业务需求，去用维度建模4步骤去对不同的数据需求进行维度建模，考虑一下在维度建模的思想下，ods和dwd层的数据模型是如何出来的，你们现在的模型有没有问题

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/051~062资料/060\_完成生鲜新零售数据仓库的ods层订单条目表设计/笔记.docx**

《060\_完成生鲜新零售数据仓库的ods层订单条目表设计》

每个商品的购买数量、单价、总金额、促销活动、商品折扣、折扣总金额、应付总金额

ods\_offline\_order：线下从门店的收银系统里抽取

order\_no：订单编号

shop\_no：门店编号

check\_out\_type：收银方式，收银台/自助收银机器

cashier\_no：收银员编号

self\_service\_cash\_register\_no： 自助收银机器编号

product\_count：总购买商品数量

total\_amount：总购买商品的金额

discount\_amount：总优惠金额

payable\_amount：总应付金额

pay\_type：支付方式

pay\_ time：支付订单的时间

ods\_offline\_order\_item：线下门店的订单条目数据

order\_no：这个订单条目所属的订单编号

order\_item\_no：订单条目的编号

product\_no：商品编号

buy\_count：商品购买数量

price：商品单价

total\_amount：商品购买总金额

promotion\_no：促销活动编号

product\_discount\_amount：商品分摊折扣金额

discount\_amount：折扣总金额

payable\_amount：应付总金额

ods\_online\_order：线上从APP的订单系统里抽取

order\_no：订单编号

mobile\_phone\_os：手机操作系统，android/ios

app\_version：APP版本号

location：地理位置

user\_id：线上APP的用户id

product\_count：总购买商品数量

total\_ amount：总购买商品的金额

discount\_amount：总优惠金额

payable\_amount：总应付金额

create\_ time：创建订单的时间

pay\_type：支付方式

pay\_ time：支付订单的时间

ods\_online\_order\_item：线上门店的订单条目数据

order\_no：这个订单条目所属的订单编号

order\_item\_no：订单条目的编号

product\_no：商品编号

buy\_count：商品购买数量

price：商品单价

total\_amount：商品购买总金额

promotion\_no：促销活动编号

product\_discount\_amount：商品分摊折扣金额

discount\_amount：折扣总金额

payable\_amount：应付总金额

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/051~062资料/051\_BAT都在涉足的生鲜新零售的案例背景介绍/笔记.docx**

《051\_BAT都在涉足的生鲜新零售的案例背景介绍》

生鲜新零售，上百门店，每个门店有杂货、冷冻食品、日常生活用品、肉类、农产品、烘烤食品、花卉、保健品、海鲜、日日鲜蔬菜、生鲜新零售（线下+线上，APP，如果你住在附近，平时可以APP上下单买菜、米、油），等等，诸如此类的东西，SKU一共有几万个

核心数据来自两块：线下，收银台的POS机/自助收银机器，每次结账就是扫描你的商品二维码，然后付钱之后打印一张收据出来；线上，APP上选购商品，下订单，支付订单，电商的流程，每一个订单=线下的每一张收据

收据/线上订单，里面包含以下内容：门店编号、收银员编号以及姓名、每个商品的编号/名称/购买数量以及应付款价格、总应付价格、总商品数量、优惠折扣抵扣的价格、最终应付价格、收款事务编号、收款时间

大致包含上述内容

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/051~062资料/052\_生鲜新零售的线上+线下的核心数据分析/笔记.docx**

《052\_生鲜新零售的线上+线下的核心数据分析》

线下门店：收银台的POS机 + 自助收银机器，收据，在他们的内部其实就是一个订单

线上APP：走的生鲜电商的流程，选购商品、下单、支付，核心数据就是一个订单

生鲜新零售而言，最最核心的数据，就是订单数据

收据/线上订单，订单表里面包含以下内容：

购买渠道：线上/线下

线上购买的一些关键字段：手机操作系统、APP版本号、地理位置

门店编号：如果你要是线下的，那么肯定是在一家店里的

收银方式：收银台/自助收银机器

收银员编号以及姓名：如果走的是线下的收银台

收银机器编号

总应付价格、总商品数量、优惠方式、优惠价格、最终应付价格

订单编号

下单时间、付款时间

订单条目表：每个商品的编号/名称/购买数量以及应付款价格

大致包含上述内容

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/051~062资料/055\_维度建模第二步：确定生鲜新零售购买过程中的数据粒度/笔记.docx**

《055\_维度建模第二步：确定生鲜新零售购买过程中的数据粒度》

业务过程，在收银台/自助收银机器，购买商品，在APP上下单购买商品

订单粒度，订单表，每一行数据就是一个订单，粒度就是订单粒度

订单条目粒度，订单表，订单条目表，每一行数据就是个订单条目，购买的一个商品

管理层在进行数据分析的时候，肯定是要对订单 条目粒度进行分析的，搞的一个促销活动效果如何，考虑一下下个月对某个商品要进货进多少件

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/051~062资料/053\_生鲜新零售公司在数据分析上通常会有哪些数据需求？/笔记.docx**

《053\_生鲜新零售公司在数据分析上通常会有哪些数据需求？》

对于生鲜新零售公司的管理层，包括中层和高层，其实日常最大的数据需求，就是希望了解到每家门店的零售数据情况，比如说每家门店每天每个商品sku售卖的数量和价格，进而可以预测出来每家门店未来对各个商品sku应该进多少货，避免有生鲜食品卖不出去就坏掉折损的情况

另外其实对于每个生鲜新零售门店，最重要的决策就是各个商品sku的定价和促销，因为首先你的定价就决定了你的利润，定价越高那你的利润越多，但是定价太高，可能会导致你的销量下降

所以你的定价必须是一个合理的区间

另外就是你的促销活动，一般门店经常搞促销活动，搞促销活动往往降低商品sku的利润，但是会提高客流量和销量，所以管理层最需要思考的，就是每个商品sku到底定价在多少罪合适，每个月搞哪些促销活动

举个例子，假设一个商品成本是10块，售价是15块，利润是5块，但是正常每个月就售出1万件，那么利润是5万块。但是如果你搞一下促销活动，优惠价是13块，利润是3块，但是每个月售出2万件，那么利润是6万块

所以说，必须要依托大量的门店数据分析情况，才能决定各个商品的价格和促销活动

也就是说，需要看到每天每个商品在无促销活动的时候，销量多少，有促销活动的时候销量多少，每个商品价格在15块的时候销量多少，价格在12块的时候销量多少，管理层需要看到大量这样的报表，才能做精细化的管理

而且促销活动还分很多种，比如说商场里直接降价、在当地发传单说有促销活动、在商场外面发各种抵用券，管理层也得分析在各种不同的促销活动下，商品的销量如何

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/051~062资料/061\_在生鲜新零售数据仓库的dwd层设计融合订单表/笔记.docx**

《061\_在生鲜新零售数据仓库的dwd层设计融合订单表》

ods\_offline\_order：线下从门店的收银系统里抽取

order\_no：订单编号

shop\_no：门店编号

check\_out\_type：收银方式，收银台/自助收银机器

cashier\_no：收银员编号

self\_service\_cash\_register\_no： 自助收银机器编号

product\_count：总购买商品数量

total\_amount：总购买商品的金额

discount\_amount：总优惠金额

payable\_amount：总应付金额

pay\_type：支付方式

pay\_ time：支付订单的时间

ods\_offline\_order\_item：线下门店的订单条目数据

order\_no：这个订单条目所属的订单编号

order\_item\_no：订单条目的编号

product\_no：商品编号

buy\_count：商品购买数量

price：商品单价

total\_amount：商品购买总金额

promotion\_no：促销活动编号

product\_discount\_amount：商品分摊折扣金额

discount\_amount：折扣总金额

payable\_amount：应付总金额

ods\_online\_order：线上从APP的订单系统里抽取

order\_id：订单编号

mobile\_phone\_os：手机操作系统，android/ios

app\_version：APP版本号

location：地理位置

user\_id：线上APP的用户id

total\_purchase\_count：总购买商品数量

total\_product\_money：总购买商品的金额

discount\_amount：总优惠金额

payable\_amount：总应付金额

create\_ time：创建订单的时间

pay\_type：支付方式

pay\_ time：支付订单的时间

ods\_online\_order\_item：线上门店的订单条目数据

order\_no：这个订单条目所属的订单编号

order\_item\_no：订单条目的编号

product\_no：商品编号

buy\_count：商品购买数量

price：商品单价

total\_amount：商品购买总金额

promotion\_no：促销活动编号

product\_discount\_amount：商品分摊折扣金额

discount\_amount：折扣总金额

payable\_amount：应付总金额

设计dwd层的表，ods层的表是属于数据仓库里的原始数据，永远不能直接去使用ods层表里的数据进行统计分析的，做一些抽象和转化，抽过来的ods层的数据可能是非常的混乱的，所以必须要对ods层的数据进行一定的清洗和转化

在dwd层形成非常标准化的一些数据，然后才能供我们的数据仓库里的数据分析需求来使用，否则ods层一堆混乱不堪的数据，是无法使用的

如果是一些别的互联网公司，多个业务系统，不同的业务系统是做不同的业务的，比如有的业务系统是做保姆业务的，有的业务系统是做上门电器维修业务，也可能会搞出来两个订单系统，出来的两套订单数据

作为一个公司，公司层面可能最后要进行分析的时候，都是要站在公司全局的角度去进行分析的，要把多个业务的销售金额放在一起来进行统计，也有可能是分成两种不同的类别来进行统计和分析的

在dwd层里，看到各种不同业务的订单表，也许订单表的字段的名称你可以在数据清洗的时候给他统一化一下，但是你还是会看到很多不同的订单表，可能会让最终的数据仓库的开发人员感到有点迷惑

生鲜新零售，线下门店 + 线上APP，是属于你公司整体的业务，我们完全可以采取在dwd层做一个融合设计的订单表

ods -> dwd：ETL，一个是把不同订单表的各个字段的名称标准化，可以走一些融合，保证数据仓库里的dwd层的模型简便性

dwd\_order：订单融合明细表

order\_no：订单编号

buy\_channel：购买渠道，线下门店/线上APP

mobile\_phone\_os：手机操作系统，android/ios

app\_version：APP版本号

location：地理位置

user\_id：线上APP的用户id

shop\_no：门店编号

check\_out\_type：收银方式，收银台/自助收银机器

cashier\_no：收银员编号

self\_service\_cash\_register\_no： 自助收银机器编号

product\_count：总购买商品数量

total\_amount：总购买商品的金额

discount\_amount：总优惠金额

payable\_amount：总应付金额

pay\_type：支付方式

pay\_ time：支付订单的时间

create\_time：订单创建时间

dwd\_order\_item：订单条目数据

order\_no：这个订单条目所属的订单编号

order\_item\_no：订单条目的编号

product\_no：商品编号

buy\_count：商品购买数量

price：商品单价

total\_amount：商品购买总金额

promotion\_no：促销活动编号

product\_discount\_amount：商品分摊折扣金额

discount\_amount：折扣总金额

payable\_amount：应付总金额

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/051~062资料/057\_维度建模第四步：确定生鲜新零售核心数据的事实字段/笔记.docx**

《057\_维度建模第四步：确定生鲜新零售核心数据的事实字段》

收据/线上订单，订单表里面包含以下内容：

购买渠道：线上/线下

线上购买的一些关键字段：手机操作系统、APP版本号、地理位置

门店编号：如果你要是线下的，那么肯定是在一家店里的

收银方式：收银台/自助收银机器

收银员编号以及姓名：如果走的是线下的收银台

收银机器编号

总应付价格、总商品数量、优惠方式、优惠价格、最终应付价格

支付方式

订单编号

下单时间、付款时间

订单条目表：每个商品的编号/名称/购买数量、每个单价、总金额以及应付款价格

每个商品的购买数量、单价、总金额、商品折扣、折扣总金额、应付总金额

正规军的数据仓库的方法论，给完全吸收了

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/051~062资料/056\_维度建模第三步：确定生鲜新零售核心数据的维度/笔记.docx**

《056\_维度建模第三步：确定生鲜新零售核心数据的维度》

收据/线上订单，订单表里面包含以下内容：

购买渠道：线上/线下

线上购买的一些关键字段：手机操作系统、APP版本号、地理位置

门店编号：如果你要是线下的，那么肯定是在一家店里的

收银方式：收银台/自助收银机器

收银员编号以及姓名：如果走的是线下的收银台

收银POS机器编号

总应付价格、总商品数量、优惠方式、优惠价格、最终应付价格

支付方式

订单编号

下单时间、付款时间

订单条目表：订单编号，每个商品的编号/名称/购买数量以及应付款价格

第一个财务季度的销售金额的统计

日期维度表、门店维度表、促销维度表、收银员维度表、支付方式维度表、商品维度

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/051~062资料/054\_维度建模第一步：选择生鲜新零售场景下的业务过程/笔记.docx**

《054\_维度建模第一步：选择生鲜新零售场景下的业务过程》

数仓建模的第一步，维度建模 -> ods + dwd + dim，就出来了 -> 数仓分层，dwt + dwa + app -> 自然而然的出来了

商品售卖的数据来源于最核心的一个业务流程，商品售卖这个流程，线上和线下，线下来说，就是客户在收银台/自助收银机器上，扫码一大堆商品，然后付款，就这个过程；线上来说，在APP里，选购一堆商品，然后支付

用户付款去购买商品的这个过程，我们要选择的这个业务过程

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/051~062资料/059\_完成生鲜新零售数据仓库的ods层订单表设计/笔记.docx**

《059\_完成生鲜新零售数仓的ods层表设计》

每个商品的购买数量、单价、促销活动、总金额、商品折扣、折扣总金额、应付总金额

ods\_offline\_order：线下从门店的收银系统里抽取

order\_no：订单编号

shop\_no：门店编号

check\_out\_type：收银方式，收银台/自助收银机器

cashier\_no：收银员编号

self\_service\_cash\_register\_no： 自助收银机器编号

product\_count：总购买商品数量

total\_amount：总购买商品的金额

discount\_amount：总优惠金额

payable\_amount：总应付金额

pay\_type：支付方式

pay\_ time：支付订单的时间

ods\_online\_order：线上从APP的订单系统里抽取

order\_no：订单编号

mobile\_phone\_os：手机操作系统，android/ios

app\_version：APP版本号

location：地理位置

user\_id：线上APP的用户id

product\_count：总购买商品数量

total\_ amount：总购买商品的金额

discount\_amount：总优惠金额

payable\_amount：总应付金额

create\_ time：创建订单的时间

pay\_type：支付方式

pay\_ time：支付订单的时间

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/051~062资料/058\_最终确定生鲜新零售核心数据的明细数据的字段/笔记.docx**

《058\_最终确定生鲜新零售核心数据的明细数据的字段》

收据/线上订单，订单表里面包含以下内容：

购买渠道：线上/线下

线上购买的一些关键字段：手机操作系统、APP版本号、地理位置

门店编号：如果你要是线下的，那么肯定是在一家店里的

收银方式：收银台/自助收银机器

收银员编号以及姓名：如果走的是线下的收银台

自助收银机器编号

总应付价格、总商品数量、优惠方式、优惠价格、最终应付价格

支付方式

订单编号

付款日期、付款时间

订单条目表：每个商品的编号/名称/购买数量、每个单价、总金额以及应付款价格

每个商品的购买数量、单价、总金额、商品折扣、折扣总金额、应付总金额

ods\_order：线下从门店的收银系统里抽取，线上从APP的订单系统里抽取

order\_no：订单编号

buy\_channel：购买渠道，线上/线下

mobile\_phone\_os：手机操作系统，android/ios

app\_version：APP版本号

location：地理位置

shop\_no：门店编号

check\_out\_type：收银方式，收银台/自助收银机器

cashier\_no：收银员编号

self\_service\_cash\_register\_no： 自助收银机器编号

promotion\_no：促销活动编号

total\_product\_count：总购买商品数量

total\_product\_amount：总购买商品的金额

total\_discount\_amount：总优惠金额

total\_payable\_amount：总应付金额

create\_order\_time：创建订单的时间

pay\_order\_time：支付订单的时间

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/041~050资料/042\_基于BAT的各种创新业务案例来讲解数据仓库建模的方法论/笔记.docx**

《042\_基于BAT的各种创新业务案例来讲解数据仓库建模的方法论》

大体上的一个数据仓库建模的方法论，大型互联网公司来的，简便实用的这套方法论，很多同学对数仓这块了解的还是太简单了，数仓，里面有很多的东西，理论和实践的东西，把他随便做做，很简单，往往最后会有一大堆的问题

数据不准确，语义不清晰，别人看不懂，开发起来经常出错，开发的效率很低下

手头一个数据报表的需求，ods -> dwd的数据清晰的ETL，dwt，dwa，app，报表出来了

BAT大互联网公司，抽取出来各种新业务、新行业、传统行业互联网化的一些案例，基于不同的案例来给大家讲解一整套的数据仓库建模的方法论

数仓，不专业；实时数据，技术不专业；数据中台、数据湖、clickhouse、kylin、druid，这些新技术，简单潦草的介绍一下，他们的一些运用的场景，针对面试的指导，自己学习一下的原理和使用，在你自己的业务里落一下地

每一块都得用一两个月去讲解，一大堆的案例，告诉你生产项目里，他是怎么玩儿的，结合自己的工作情况，结合自己公司的项目和业务，绝对可以把我的案例里的一大堆的生产经验、设计经验、架构经验，落地到自己的业务里去，包装很好的项目

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/041~050资料/048\_维度建模的经典4步骤：一行数据里包含哪些事实？/笔记.docx**

《048\_维度建模的经典4步骤：一行数据里包含哪些事实？》

现在我们假设明确了一个业务流程，就是医生开药方，然后数据粒度每一行数据是一个药方里的一个药品，然后描述每一行数据的维度包括了分院id、科室id、医生id、病人id、日期、处方id、药品id，这些东西 + 药品数量 + 药品金额

dwd ->

（1）处方表：每一个处方自己是一行明细数据，这里包含了每一个处方的数据

（2）处方药品表：就有一个处方里的每个药品的明细数据表，每一行数据就是上面看到的那些东西

dim -> 分院维度表、科室维度表、医生维度表、病人维度表、药品维度表

其实这些维度就描述了一行数据，是哪个分院的哪个科室的哪个医生，给哪个病人，在哪一天，开了哪个药方，里面开了哪个药

接着确定这些东西之后，就需要确定这行数据里包含哪些事实，所谓的事实，其实就是可以用于计算的一些数据指标，比如说在上面吧，那就是说这个药开了几盒？然后开这个药的价格是多少钱？是否有折扣，有多少折扣？是否有医保抵扣，抵扣了多少钱？

就是这些数字类的东西，可以用于计算的，其实就是所谓的事实数据了

所以现在，每一行数据，大家会发现，已经涵盖了完整的一行数据，通过这些数据，你完全可以给医院开发一个数据仓库，通过这些数据算出来医院里的一些营业数据报表、药品销售报表，通过各种不同的维度的汇总计算，可以算出来不同的数据报表

对于每一个数据报表的需求，都应该走上面的4个维度建模的步骤：从哪儿抽什么数据，数据的粒度，数据的维度，数据的事实 -> ods、dwd、dim，每一个数据报表在做他的需求的时候，都是要走上面的4个维度建模的步骤

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/041~050资料/049\_维度建模之后：数仓分层、数据采集、ETL以及报表展示/笔记.docx**

《049\_维度建模之后：数仓分层、数据采集、ETL以及报表展示》

针对一些数据报表的需求，维度建模搞完了之后，ods -> dwd + dim，就可以确定下来了，包括你需要的ods层的表结构，dwd和dim层的表结构 -> 数仓分层，dwt，把dwd和dim先进行join，做成一个大宽表 ->

处方药品表，里面有很多维度的id -> 在dwt层，跟dim层的各个维度表里的各种字段进行关联，做成一个大宽表，进入dwt层

医生id -> join一下医生维度表 -> 在dwt层你的这个表里，就可以有医生的姓名，年龄，别的一些信息 -> 病人、科室等等一些维度，都可以在dwt层做一个join，在表里就可以有完善的一些信息，dwt层的表可能有10万行记录

dwa这一层，你是不是可以先基于一些维度做一个轻度的汇总和聚合

做一个轻度的聚合，select sum(sale\_count),sum(sale\_price) from dwt\_xxx group by 分院、科室、医生、药品，100个医生，每个医生每个月会开出去100种药品，1万行记录

就可以计算出来，每个分院的每个科室的每个医生开出的每个药品，有多少件，有多少金额，这就是一个典型的轻度汇总聚合表

dwa\_xxx，有了这样的一个表，聚合过的，已经没有明细数据了，某一个处方的某一个药品的情况，已经不可能在这个表里看到了

出一些app层的报表，根据科室来计算

select sum(sale\_price) from dwa\_xxx group by 科室 -> 科室报表，每个科室这个月开的药品的金额是多少，这个app层报表的ETL的性能就会很好

select sum(sale\_count), sum(sale\_price) from dwa\_xxx group by 药品 -> 药品报表，这个医院每个月对每种药品开出来的金额一共是多少，数量是多少

进行数据采集，采集你需要的数据到ods；开发对应的ETL，ETL调度，把每一层的数据跑出来，一直到把报表数据算出来；报表系统的展示

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/041~050资料/045\_维度建模的经典4步骤：选择你公司的业务流程/笔记.docx**

《045\_维度建模的经典4步骤：选择你公司的业务流程》

手头有一堆数据报表需求，第一件事情，不是瞎搞，是有一个很专业的步骤，维度建模。在一个多月的时间里，你在数仓这块已经从杂牌军，变成正规军的玩儿法

电商公司：看一些订单的数据

传统材料公司：客户在我这里采购了很多东西，开发票，发票相关的数据

贸易公司：经销一些商品，跟客户完成了一笔大额的交易，交易的数据

旅游公司：客服，处理一些咨询和投诉的电话，客服的处理咨询的数据

学校：校长，看一下今年已经有多少新生完成了注册

保险公司：看一下最近这个月完成了多少笔理赔，理赔了多少钱

医院：院长，本月一共完成多少次病人就诊

维度建模 -> dwd + dim，明细数据和维度数据 -> ETL把各种报表计算出来

选择业务过程

先理解公司的完整业务流程，然后从里面选择一个或者几个跟你的数据报表需求相关的业务流程，比如说用户支付订单、开发票、与客户完成支付交易、客服处理一个咨询电话、学校完成一个学生的注册、保险公司完成一笔索赔、医院完成一次病人就诊，等等。

记住，你的数据报表的需求一定是来源于你的公司的业务流程中，在执行业务流程的过程中，必然会生产一些数据，然后利用这些数据去出你的数据报表

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/041~050资料/047\_维度建模的经典4步骤：一行数据有哪些维度描述？/笔记.docx**

《047\_维度建模的经典4步骤：一行数据有哪些维度描述？》

互联网医疗的一个场景举例子，线上找医生问诊，医生会给你开一个电子处方，你甚至接着就可以在医药电商网站上，立马就可以通过你的电子处方去买药，我之前就体验过这个完整的流程

在线问诊，和药品销售，是两回事儿

要分析的数据报表，是在线问诊的环节，看的是本月在线问诊的环节中，医生开方子的时候，里面的用药的一些数据情况，每种药在这个月里被医生开了多少次，对应的金额是多少，你其实就可以计算出来，你这个互联网医疗公司，对社会产生的价值有多大

确定了业务流程中生成出来的一些数据，我们要采集什么样的数据粒度，然后应该定义好对这些数据的维度描述，其实维度特别的简单，就是说这一行数据必然是你干了一个什么事儿，无论是医生开了一个药方、跟客户完成一笔交易、当日最后一秒的商品库存情况，他都是一个事儿

第一步：医生开处方或者是推荐药品列表的业务流程，是我们选择的业务过程

第二步：每一个处方里的每一个药，就是一行数据，作为粒度

第三步：医生、科室、分院、日期、病人、是否使用医保、是否异地就医，是维度数据

维度建模是什么意思呢，对你的数据报表的需求，你必须要通过维度建模，选择好你要抽取数据的业务过程（可能就对应了线上某个业务系统里的一个业务功能），抽取的业务数据的粒度，确定描述数据的维度

数据分层，ods（业务系统里抽取出来的你需要的粒度的数据） -> dwd（明细数据，代表科室的字段，就是一个科室id字段，医生id字段） + dim（维度数据，从业务系统里抽取出来很多的维度表，科室维度表，医生维度表）

然后对这个事儿你得有一些维度来描述他，比如说医生开药方吧，他是哪个医生开的？哪个科室开的？哪个分院开的？哪一天开的？给哪个病人开的？有没有用医保？是不是异地病人在本地就医？

其实类似上面这些东西，就是对一件事情一连串的描述，上面说的每一个药方的医生、科室、分院、日期、病人、是否使用医保、是否异地就医，这些东西就是医生开药方这个事情的维度，他们是描述你开药方这个事儿的

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/041~050资料/043\_从一个数据工程师接到一大堆数据需求开始讲起/笔记.docx**

《043\_维度建模的经典4步骤：选择你公司的业务流程》

手头有一堆数据报表需求，第一件事情，不是瞎搞，是有一个很专业的步骤，维度建模。在一个多月的时间里，你在数仓这块已经从杂牌军，变成正规军的玩儿法

电商公司：看一些订单的数据

传统材料公司：客户在我这里采购了很多东西，开发票，发票相关的数据

贸易公司：经销一些商品，跟客户完成了一笔大额的交易，交易的数据

旅游公司：客服，处理一些咨询和投诉的电话，客服的处理咨询的数据

学校：校长，看一下今年已经有多少新生完成了注册

保险公司：看一下最近这个月完成了多少笔理赔，理赔了多少钱

医院：院长，本月一共完成多少次病人就诊

选择业务过程

先理解公司的完整业务流程，然后从里面选择一个或者几个跟你的数据报表需求相关的业务流程，比如说用户支付订单、开发票、与客户完成支付交易、客服处理一个咨询电话、学校完成一个学生的注册、保险公司完成一笔索赔、医院完成一次病人就诊，等等。

记住，你的数据报表的需求一定是来源于你的公司的业务流程中，在执行业务流程的过程中，必然会生产一些数据，然后利用这些数据去出你的数据报表

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/041~050资料/041\_回顾一下之前讲解的整套互联网公司数仓建设方法论/笔记.docx**

《041\_基于BAT的各种创新业务案例来讲解数据仓库建模的方法论》

初步的讲解了一下数据仓库这个东西是什么，包括一些基本的互联网公司，数据仓库的分层办法，数据仓库构建过程中的全流程，数仓建模（分层、各层的模型设计、最终报表的结构设计），数据采集，写一堆的ETL（SQL），做作业调度，原始数据（ods）-> dwd + dim -> dwt -> dwa -> app（出各种各样的报表）-> mysql、es -> 报表系统展示

从0到1的过程，搭建上面那套架构

从1到10的过程，就是在上面的这套架构下，不停的开发各种各样的报表，数据需求

从10到100，玩儿深了，数据治理，元数据，数据质量，数据生命周期，数据权限，数据中台

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/041~050资料/050\_因为大病初愈，所以本周恢复讲课不能讲太多的原因解释/笔记.docx**

《050\_因为大病初愈，所以本周恢复讲课不能讲太多的原因解释》

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/041~050资料/046\_维度建模的经典4步骤：确定业务流程中的明细数据粒度/笔记.docx**

《046\_维度建模的经典4步骤：确定业务流程中的明细数据粒度》

选择好了一个或者几个业务流程之后，接着你就要想，你要从这些业务流程中提取的明细数据的粒度是什么

比如说对于一个贸易公司，你跟客户之间完成交易的过程，你对每个客户的每一笔订单数据，是你的一行明细数据粒度？还是说订单中的每一个商品订购条目，是一行明细数据粒度？确定好业务流程里采集出来的数据的粒度

贸易公司的老板，我主要看到这个月完成了多少笔交易订单，整个的交易金额是多少，就可以了，数据报表需求是这样的话 -> 跟客户完成交易的过程 -> 一个交易订单是一行数据，作为数据粒度 -> 订单的数量和金额

比如说对于一个医院而言，是医生给你开的每一个药方是一行明细数据粒度？还是说药方里的每一个药品是一行明细数据粒度？院长，看的数据报表，这个月医院里面对每种药的售出的金额是多少钱

比如说对于一个机场而言，你在登机的时候，是不是要给人家扫描你的机票，然后才能登机？那么登机口每次扫描一次你的机票，就这一次扫描数据是不是就应该是一行明细数据粒度？一行数据就是一次登机

对于一个仓库而言，是不是每一个商品的库存在每天晚上23:59:59秒的时候，他在这一天里最后一秒的那个库存数值快照，就应该是你的一行明细数据粒度？在你的数据仓库里，dwd明细数据里，一个商品会对应多条库存数据，每天都有一个库存快照数据

iphone手机 2020-01-01 5987

iphone手机 2020-01-02 4456

对于一个银行而言，是不是每个银行账户每天在23:59:59最后一秒的时候，他的一个账户余额整体情况，就是一行明细数据粒度？

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/063~072资料/064\_企业级日期维度表设计：字段含义详解/笔记.docx**

《064\_企业级日期维度表设计：字段含义详解》

现在就是想要看看过去的几个月内，工作日的商品销售额，非工作日的商品销售额，两种情况下的销售额，我都想看一看

select xxx xxx sum(xxx)

from xxx

where datekey>= and datekey<=

and 需要有一个运行逻辑较为复杂的你自己写的hive udf函数(daykey) = non\_weekday

select xxx xxx sum(xxx)

from dwt\_order join dim\_date on dwt\_order.pay\_datekey=dim\_date.datekey

where dwt\_order.datekey>= and dwt\_order.datekey<=

and dim.is\_weekday=yes

从你公司创建以来，有数据以来，一直到未来的10年为止，每一天都是一行数据，每一天是否是工作日，在这张表里就已经提前计算好了，有一个标识这一天是否是工作日，直接把事实表和日期维度表

datekey：日期键，比如20200101

date：另外一种格式的日期，2020-01-01

full\_date\_description：日期的详细描述，2020年01月01日

day\_of\_week：这一天是在一周中的周几，Monday，Tueseday

day\_number\_in\_calendar\_month：这一天是在一个月中的哪一天

day\_number\_in\_calendar\_year：这一天是在一年中的哪一天

day\_number\_in\_fiscal\_month：这一天是在财务月中的哪一天

day\_number\_in\_fiscal\_year：这一天是在财务年中的哪一天，2020-03-25~2021-03-24

calendar\_week\_number\_in\_year：这一周是在一年中的第几周

calendar\_month：日历月，January

calendar\_month\_number\_in\_year：这一个月是在一年中的第几个月，5

calendar\_quarter：日历季度

calendar\_year：日历年，2020

fiscal\_week\_number\_in\_year：这一周是在财务年中的第几周

fiscal \_month：财务月，January

fiscal \_month\_number\_in\_year：这一个月是在财务年中的第几个月

fiscal \_quarter：财务年中的第几个季度

fiscal \_year：财务年，2020

is\_holiday：是否为节假日，yes or no

is\_weekday：是否为工作日，yes or no

2020年第一个月，2020-01-01~2020-01-31

财务结算的周期跟日历的周期是不同的，2020年第一个财务上的月份，可能是2020-01-21~2020-02-21，2019-12-21~2020-01-20

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/063~072资料/069\_企业级的商品维度表都包含哪些字段？/笔记.docx**

《069\_企业级的商品维度表都包含哪些字段？》

order，到订单这个层面，配合上一些日期维度表，综合性的分析一下新零售公司的整体的营业额，基本还算是ok的，很多时候，做一些具体的分析的时候，还要看到商品维度，过去一个月里，我在全国的门店引入了新的商品，新奥尔烤鸡

报表需求：过去一个月里的所有门店的新奥尔良烤鸡这个商品sku的销售额是多少

报表需求：过去一个月里引入了两种新品牌的薯片，分析一下这两种新品牌的薯片销售额是多少

order\_item表，才有每个订单的每一个商品的销售数据，dim\_product

product\_no：商品编号

product\_sku\_no：商品的sku号码

product\_name：商品名称

brand：品牌

sub\_category：子类目，薯片

category：类目，零食

department\_no：部门编号，商品部门是一个大部门，会包含很多子部门

department\_name：部门名称

package\_type：包装类型

package\_size：包装大小

fat\_content：食品热量

is\_diet\_food：是否为减肥食品

weight：商品重量

weight\_unit\_of\_measure：商品重量的单位

storage\_type：存储类型

shelf\_lift\_type：货架类型

shelf\_width：货架宽度

shelf\_height：货架高度

shelf\_lenth：货架深度

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/063~072资料/072\_本周作业：梳理一下自己公司的数仓业务可以有哪些维度表？/笔记.docx**

《v 072\_本周作业：梳理一下自己公司的数仓业务可以有哪些维度表？》

两个维度表的设计，没讲什么设计技巧，缓慢变化维，拉链表，高阶的设计技巧还是有一些的，学全了，几十种甚至上百种之多，应付各种各样的奇葩的情况，如果没有那些维度表会有什么损失，有了这些维度表可以做哪些复杂分析，维度表里包含哪些字段，是否从数据源抽取之后做过清洗

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/063~072资料/070\_基于企业级的商品维度表，可以实现哪些数据需求？/笔记.docx**

《070\_基于企业级的商品维度表，可以实现哪些数据需求？》

order，到订单这个层面，配合上一些日期维度表，综合性的分析一下新零售公司的整体的营业额，基本还算是ok的，很多时候，做一些具体的分析的时候，还要看到商品维度，过去一个月里，我在全国的门店引入了新的商品，新奥尔烤鸡

报表需求：过去一个月里的所有门店的新奥尔良烤鸡这个商品sku的销售额是多少

报表需求：过去一个月里引入了两种新品牌的薯片，分析一下这两种新品牌的薯片销售额是多少

order\_item表，才有每个订单的每一个商品的销售数据，dim\_product

数据仓库里的所谓的下钻

每个品牌的商品的销售额

品牌AAA 销售额

品牌BBB 销售额

每个品牌的每个类目的商品的销售额

品牌AAA 饮料 销售额

品牌AAA 速食 销售额

品牌AAA 卤味 销售额

品牌BBB 蔬菜 销售额

品牌BBB 荤菜 销售额

品牌AAA 饮料 碳酸饮料 销售额

品牌AAA 饮料 牛奶饮料 销售额

品牌AAA 饮料 果汁饮料 销售额

rollup，上卷

product\_no：商品编号

product\_sku\_no：商品的sku号码

product\_name：商品名称

brand：品牌

sub\_category：子类目，薯片

category：类目，零食

department\_no：部门编号，商品部门是一个大部门，会包含很多子部门

department\_name：部门名称

package\_type：包装类型

package\_size：包装大小

fat\_content：食品热量

is\_diet\_food：是否为减肥食品

weight：商品重量

weight\_unit\_of\_measure：商品重量的单位

storage\_type：存储类型

shelf\_lift\_type：货架类型

shelf\_width：货架宽度

shelf\_height：货架高度

shelf\_lenth：货架深度

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/063~072资料/066\_企业级日期维度表的数据生成的实现思路的提示/笔记.docx**

《066\_企业级日期维度表的数据生成的实现思路的提示》

就是两个字：逻辑

如果你要用java来生成这些数据，也是可以的，python、shell都可以

java里面本身就有一个日历包，或者到网上去找一个日历包，通过日历包，就可以遍历未来20年的每一天的数据

days

while(days.hasNext()) {

day = days.next();

String datekey = getDatekey(day);

int dayOfWeek = getDayOfWeek(day);

}

public String getDatekey(Day day) {

String date = day.getData(); // 2020-01-01

return date.replaceAll(“-”, “”);

}

public String getDayOfWeek(Day day) {

int weekKey = day.getWeekKey(); // Tueseday

switch weekKey :

case “Tueseday”: return 2

….

}

现在就是想要看看过去的几个月内，工作日的商品销售额，非工作日的商品销售额，两种情况下的销售额，我都想看一看

select xxx xxx sum(xxx)

from xxx

where datekey>= and datekey<=

and 需要有一个运行逻辑较为复杂的你自己写的hive udf函数(daykey) = non\_weekday

select xxx xxx sum(xxx)

from dwt\_order join dim\_date on dwt\_order.pay\_datekey=dim\_date.datekey

where dwt\_order.datekey>= and dwt\_order.datekey<=

and dim.is\_weekday=yes

从你公司创建以来，有数据以来，一直到未来的10年为止，每一天都是一行数据，每一天是否是工作日，在这张表里就已经提前计算好了，有一个标识这一天是否是工作日，直接把事实表和日期维度表

报表需求1：我现在想看看，过去几个月内，周五这一天的销售额和周四这一天的销售额的对比，周日这一天的销售额和周六这一天的销售额的对比

报表需求2：过去几个月里，每个月的第一天的销售额，加起来是多少，每个月的最后一天额销售额加起来是多少

报表需求3：看一下过去的一个财务年里的第三个财务季度的销售情况

如果未来有一天你发现一个复杂的日期维度的报表需求，日期维度表无法支持，你要做的是不是先去写一个复杂的日期函数呢？应该找团队里负责维护这张日期维度表的同学，让他在脚本里加入一个逻辑，把这一天的你需要的一个字段计算出来

hive里这张表没多少数据，几千条数据，是绝对的一个小表，在hdfs上就是一个小文件，几十MB的一个小文件，此时你重新算一下所有日期的数据，再次往hive表insert overwrite导入覆盖一下

datekey：日期键，比如20200101

date：另外一种格式的日期，2020-01-01

full\_date\_description：日期的详细描述，2020年01月01日

day\_of\_week：这一天是在一周中的周几，Monday，Tueseday，Friday

day\_number\_in\_calendar\_month：这一天是在一个月中的哪一天，1，2，3

is\_last\_day\_in\_calendar\_month：是否是当月的最后一天，yes or no

day\_number\_in\_calendar\_year：这一天是在一年中的哪一天

day\_number\_in\_fiscal\_month：这一天是在财务月中的哪一天

day\_number\_in\_fiscal\_year：这一天是在财务年中的哪一天，2020-03-25~2021-03-24

calendar\_week\_number\_in\_year：这一周是在一年中的第几周

calendar\_month：日历月，January

calendar\_month\_number\_in\_year：这一个月是在一年中的第几个月，5

calendar\_quarter：日历季度

calendar\_year：日历年，2020

fiscal\_week\_number\_in\_year：这一周是在财务年中的第几周

fiscal \_month：财务月，January

fiscal \_month\_number\_in\_year：这一个月是在财务年中的第几个月

fiscal \_quarter：财务年中的第几个季度，3

fiscal \_year：财务年，2020

is\_holiday：是否为节假日，yes or no

is\_weekday：是否为工作日，yes or no

2020年第一个月，2020-01-01~2020-01-31

财务结算的周期跟日历的周期是不同的，2020年第一个财务上的月份，可能是2020-01-21~2020-02-21，2019-12-21~2020-01-20

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/063~072资料/067\_从企业级日期维度表的设计，理解什么是维度建模？/笔记.docx**

《067\_从企业级日期维度表的设计，理解什么是维度建模？》

对维度建模的4部曲，接到数据需求的时候，如何像个正规军一样，别像个杂牌军，有步骤的去进行维度建模，设计ods、dwd，ods和dwd里需要哪些字段就已经知道了，维度建模，真的是国外的数仓的大师，积累了二三十年的一些经验，总结出来

dim层需要哪些维度表，也知道了，接着就是进行维度表的设计就可以了，日期维度表

维度建模4步曲，比较简单一些，ods和dwd，一般来说都是从数据源抽取过来的，其实一些字段包含什么，本身业务系统的表里都有了，采集的日志一般来说也就那么几个字段而已，就可以了

要做各种各样的数据报表，最核心的其实是维度，任何数据报表一定都会有一个或者几个维度，比如按照日期维度，产品维度，门店维度，支付维度，很多的维度，在这些维度之下，可能会基于维度做一些复杂的需求

数据仓库建设里，ods和dwd，ods其实就是原样从业务系统里抽取数据或者是用户行为日志采集，机器监控数据采集，dwd明确一些事实字段和维度字段，ods->dwd，你要设计一些数据清洗的逻辑

维度表，可能里面蕴含了很多复杂的设计技巧，离线这块，3月份一个月，都是在讲解离线数仓的设计技巧，结合一些案例来讲解，突击课程，数仓设计里，维度表这块，最核心最核心，最高频，最常用的一些技巧，拿出来给大家讲解

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/063~072资料/063\_在企业级数据仓库中，日期维度表通常如何设计？/笔记.docx**

《063\_在企业级数据仓库中，日期维度表通常如何设计？》

现在就是想要看看过去的几个月内，工作日的商品销售额，非工作日的商品销售额，两种情况下的销售额，我都想看一看

select xxx xxx sum(xxx)

from xxx

where datekey>= and datekey<=

and 需要有一个运行逻辑较为复杂的你自己写的hive udf函数(daykey) = non\_weekday

select xxx xxx sum(xxx)

from dwt\_order join dim\_date on dwt\_order.pay\_datekey=dim\_date.datekey

where dwt\_order.datekey>= and dwt\_order.datekey<=

and dim.is\_weekday=yes

从你公司创建以来，有数据以来，一直到未来的10年为止，每一天都是一行数据，每一天是否是工作日，在这张表里就已经提前计算好了，有一个标识这一天是否是工作日，直接把事实表和日期维度表

datekey：日期键，比如20200101

date：另外一种格式的日期，01/01/2020

full\_date\_description：日期的详细描述，2020年01月01日

day\_of\_week：这一天是在一周中的周几，Monday，Tueseday

day\_number\_in\_calendar\_month：这一天是在一个月中的哪一天

day\_number\_in\_calendar\_year：这一天是在一年中的哪一天

day\_number\_in\_fiscal\_month：这一天是在财务月中的哪一天

day\_number\_in\_fiscal\_year：这一天是在财务年中的哪一天

calendar\_week\_number\_in\_year：这一周是在一年中的第几周

calendar\_month：日历月，January

calendar\_month\_number\_in\_year：这一个月是在一年中的第几个月

calendar\_quarter：日历季度

calendar\_year：日历年，2020

fiscal\_week\_number\_in\_year：这一周是在财务年中的第几周

fiscal \_month：财务月，January

fiscal \_month\_number\_in\_year：这一个月是在财务年中的第几个月

fiscal \_quarter：财务年中的第几个季度

fiscal \_year：财务年，2020

is\_holiday：是否为节假日

is\_weekday：是否为工作日

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/063~072资料/071\_商品维度表的数据是从哪儿抽取过来的？/笔记.docx**

《071\_商品维度表的数据是从哪儿抽取过来的？》

从业务系统里抽取过来，商品表一般来说都是在业务系统里，表里的数据量一般不会太多，你可以让他每天都抽一份ods层的维度表，直接insert overwrite直接覆盖导入到ods层的一个dim表里去，覆盖掉全量数据

对ods层的dim表的数据，你自己写一个ETL，做一些清洗，把ods层的dim表弄到dim层的dim表里去

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/063~072资料/068\_企业级的商品维度表，应该如何设计？/笔记.docx**

《068\_企业级的商品维度表，应该如何设计？》

order，到订单这个层面，配合上一些日期维度表，综合性的分析一下新零售公司的整体的营业额，基本还算是ok的，很多时候，做一些具体的分析的时候，还要看到商品维度，过去一个月里，我在全国的门店引入了新的商品，新奥尔烤鸡

报表需求：过去一个月里的所有门店的新奥尔良烤鸡这个商品sku的销售额是多少

报表需求：过去一个月里引入了两种新品牌的薯片，分析一下这两种新品牌的薯片销售额是多少

order\_item表，才有每个订单的每一个商品的销售数据，dim\_product

product\_no：商品编号

sku\_number：商品的sku号码

product\_name：商品名称

brand：品牌

sub\_category：子类目

category：类目

department\_number：部门编号

department\_name：部门名称

package\_type：包装类型

package\_size：包装大小

fat\_content：食品热量

diet\_type：减肥食品类型

weight：商品重量

weight\_unit\_of\_measure：商品重量的单位

storage\_type：存储类型

shelf\_lift\_type：货架类型

shelf\_width：货架宽度

shelf\_height：货架高度

shelf\_depth：货架深度

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/063~072资料/065\_基于企业级日期维度表，可以实现哪些复杂需求？/笔记.docx**

《065\_基于企业级日期维度表，可以实现哪些复杂需求？》

现在就是想要看看过去的几个月内，工作日的商品销售额，非工作日的商品销售额，两种情况下的销售额，我都想看一看

select xxx xxx sum(xxx)

from xxx

where datekey>= and datekey<=

and 需要有一个运行逻辑较为复杂的你自己写的hive udf函数(daykey) = non\_weekday

select xxx xxx sum(xxx)

from dwt\_order join dim\_date on dwt\_order.pay\_datekey=dim\_date.datekey

where dwt\_order.datekey>= and dwt\_order.datekey<=

and dim.is\_weekday=yes

从你公司创建以来，有数据以来，一直到未来的10年为止，每一天都是一行数据，每一天是否是工作日，在这张表里就已经提前计算好了，有一个标识这一天是否是工作日，直接把事实表和日期维度表

报表需求1：我现在想看看，过去几个月内，周五这一天的销售额和周四这一天的销售额的对比，周日这一天的销售额和周六这一天的销售额的对比

报表需求2：过去几个月里，每个月的第一天的销售额，加起来是多少，每个月的最后一天额销售额加起来是多少

报表需求3：看一下过去的一个财务年里的第三个财务季度的销售情况

如果未来有一天你发现一个复杂的日期维度的报表需求，日期维度表无法支持，你要做的是不是先去写一个复杂的日期函数呢？应该找团队里负责维护这张日期维度表的同学，让他在脚本里加入一个逻辑，把这一天的你需要的一个字段计算出来

hive里这张表没多少数据，几千条数据，是绝对的一个小表，在hdfs上就是一个小文件，几十MB的一个小文件，此时你重新算一下所有日期的数据，再次往hive表insert overwrite导入覆盖一下

datekey：日期键，比如20200101

date：另外一种格式的日期，2020-01-01

full\_date\_description：日期的详细描述，2020年01月01日

day\_of\_week：这一天是在一周中的周几，Monday，Tueseday，Friday

day\_number\_in\_calendar\_month：这一天是在一个月中的哪一天，1，2，3

is\_last\_day\_in\_calendar\_month：是否是当月的最后一天，yes or no

day\_number\_in\_calendar\_year：这一天是在一年中的哪一天

day\_number\_in\_fiscal\_month：这一天是在财务月中的哪一天

day\_number\_in\_fiscal\_year：这一天是在财务年中的哪一天，2020-03-25~2021-03-24

calendar\_week\_number\_in\_year：这一周是在一年中的第几周

calendar\_month：日历月，January

calendar\_month\_number\_in\_year：这一个月是在一年中的第几个月，5

calendar\_quarter：日历季度

calendar\_year：日历年，2020

fiscal\_week\_number\_in\_year：这一周是在财务年中的第几周

fiscal \_month：财务月，January

fiscal \_month\_number\_in\_year：这一个月是在财务年中的第几个月

fiscal \_quarter：财务年中的第几个季度，3

fiscal \_year：财务年，2020

is\_holiday：是否为节假日，yes or no

is\_weekday：是否为工作日，yes or no

2020年第一个月，2020-01-01~2020-01-31

财务结算的周期跟日历的周期是不同的，2020年第一个财务上的月份，可能是2020-01-21~2020-02-21，2019-12-21~2020-01-20

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/104~113资料/109\_还有几种复杂的缓慢变化维要学习吗？/笔记.docx**

《109\_还有几种复杂的缓慢变化维要学习吗？》

应对维度表的属性值缓慢变化的情况，还有几种高阶技巧，微型维度，支架表的变化维度，几乎很少用，数据仓库架构师，对大数据技术这块不是特别的精通，主要是会用，比较熟悉而已，专门玩儿各种复杂业务的数仓建模

不变的维度属性值，变化的维度属性值但是每天都覆盖旧值，后续的所有数据全量重跑，这种是用的比较多的

拉链表，把一个维度的属性值变化的快照记录下来，可以针对不同的快照去进行一个统计，一般来说很多人都停留在覆盖旧值的水平，拉链表都没玩儿过，拉链表都算是一种高阶一点的技巧吧

基于拉链表的增量更新，就是你的表里每天新增出来的拉链数据，start\_date是今天的数据，就是最新的一些变更过的数据，针对这些变更过的数据，你此时可以在你的ETL里专门针对相关联的数据就把变更过的数据的指标重新计算一下

每天就是在增量更新

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/104~113资料/110\_拉链表实战场景引入：新零售场景下的订单退款问题/笔记.docx**

《110\_拉链表实战场景引入：新零售场景下的订单退款问题》

真实经历过的案例，新零售的订单状态跨天变更

有一个订单，在2020-01-01创建了，此时在数据仓库里了，他的状态此时是已支付，1月2号的凌晨会算1月1号的数据，此时会把这个订单支付的金额，算到一些指标里去，当日销售额

但是呢，到了2020-01-03号的时候，这个订单的状态变更为了已退款了，在业务系统里，订单的状态就是已退款，此时这个订单既然退款了，那么就必须要把那一天的数据指标全部重新计算一遍了，否则那一天的数据就都是错的了！

1月3号会收集到一个变更binlog，对于订单表肯定有一个update语句的binlog

1月4号的凌晨，会发现这个订单有一个变更，此时如果你简单的采取merge的办法，直接把已退款的状态merge到那个订单里去，此时可能每天都有一些未知的订单会状态变更为已退款

但是其实你很难鉴别出来是哪些订单变成已退款了，在通用的处理逻辑里，无非就是根据binlog更新一下订单的状态，update\_time，此时这个时间会变更一下，就是订单最近一次更新的时间

从dwd，dwt，dwa，app，所有层的ETL全部基于全量数据重跑一遍

所以这个问题就导致了几乎这种数据指标每天都得全量重新计算一遍，非常的麻烦

此时就需要引入拉链表了，基于拉链表来解决这个问题

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/104~113资料/112\_新零售场景下的订单拉链表的ETL如何开发？/笔记.docx**

《112\_新零售场景下的订单拉链表的ETL如何开发？》

INSERT OVERWRITE TABLE ods\_order

SELECT \* FROM

(

SELECT A.order\_no

A.status,

// 省略一大堆字段

A.pay\_time,

A.start\_date,

CASE

WHEN A.end\_date = '9999-12-31'

AND B.order\_no IS NOT NULL

THEN yesterday()

ELSE A.end\_date

END AS end\_date

FROM ods\_order AS A

LEFT JOIN ods\_order\_binlog AS B

ON A.order\_no = B.order\_no

WHERE B.timestamp=today()

AND B.operation=’update’

UNION

SELECT C.order\_no,

C.order\_status,

C.pay\_time,

// 省略一大堆字段

today() AS start\_date,

'9999-12-31' AS end\_date

FROM ods\_order\_binlog AS C

WHERE C.timestamp=today()

AND C.operation=’update’

) AS T

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/104~113资料/107\_第三种缓慢变化维：类型2，维度拉链表/笔记.docx**

《107\_第三种缓慢变化维：类型2，维度拉链表》

product\_no product\_id name department

111223 110 iphone手机 智能终端部

提了一个需求：就是我想看看2020-01-31之前的所有的数据里面，电子产品部的一些销售的情况，好此时完蛋了，因为你的这个商品的部门已经变成了智能终端部门，此时你无法把这个商品在2020-01-31之前的销售归纳到老的电子产品部里去了

product\_no product\_id name department start\_date end\_date status

111223 110 iphone手机 电子产品部 2020-01-01 2020-01-31 expired

223546 110 iphone手机 智能终端部 2020-02-01 9999-12-31 valid

假设你每个商品对应了一些交易记录，如果你要分开统计，那么就是这样的

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/104~113资料/111\_新零售场景下的订单拉链表应该如何设计？/笔记.docx**

《111\_新零售场景下的订单拉链表应该如何设计？》

order\_no

中间省略一大堆的字段

order\_status：待支付，已取消，已支付，已退款

pay\_time：2020-01-01

start\_date

end\_date

大家可以看到，所谓的拉链表完全没什么神秘的，无非就是对于订单这种数据变更的时候，不是直接覆盖原来的旧值，而是多加一行，然后用start\_date和end\_date标记每一行数据的开始日期和截止日子

order\_no order\_status pay\_time start\_date end\_date

11355 已支付 2020-01-01 2020-01-01 2020-01-02

11355 已退款 2020-01-01 2020-01-03 9999-12-31

对于dwd、dwt、dwa、app，后续所有的etl，全部可以做增量计算了，只要把order表里的start\_date为当天的数据，也就是说当天有变化的数据拿出来，跟相关联数据做一个重新的计算就可以了

然后只要你做好拉链表，那么你只要根据start\_date把每天多出来的拉链数据来做一下计算就可以了，比如说每天新增的支付的订单，拿出来可以计算为当天的各种交易指标；每天的退款订单，可以单独拎出来，把之前那一天的交易指标减去今天的退款金额，就可以了

这样每天只要计算增量的数据就可以了，完全没必要说去全量重新计算一下

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/104~113资料/104\_缓慢变化维指的是什么？新零售场景下有缓慢变化维吗？/笔记.docx**

《104\_缓慢变化维指的是什么？新零售场景下有缓慢变化维吗？》

商品维度表

product\_no name description price department

11355 魔法薯片 墨西哥烤肉 48.50 油炸食品部门

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/104~113资料/106\_第二种缓慢变化维：类型1，直接覆盖旧值/笔记.docx**

《106\_第二种缓慢变化维：类型1，直接覆盖旧值》

抽取业务系统的数据，抽的是他的变更binlog，拿到每天的最新binlog之后，需要跟ods层的全量数据进行一个merge，比如有数据新增了，那么就在etl里给弄到全量表新增一条数据，如果有数据修改了，那么就修改一下数据，如果有数据删除了，那么就把全量表里的数据删除了

在ods层里一定会维护一份跟业务系统0点之前的数据快照一模一样的一份全量数据

基于spark做一些逻辑出来

缓慢变化了，一定会拿到对应的binlog，直接基于binlog跟ods层的维度表进行一个merge，此时比如说维度的一个属性值变化了，此时就直接在ods层的全量维度表里，把那个属性值变更一下就可以了

在这种情况下，你的维度表是随时可有可能会变化的，如何变化你是不知道的，从业务系统里抽取数据出来，一般考虑到事实表和维度表每天随时可能会变化的问题，每天都是抽取增量binlog，merge到ods层的全量表里去

ods层的有全量的事实表和维度表，用binlog更新完之后，此时就可以基于全量的事实表和维度表，把dwd、dwt、dwa、app层所有层的数据表全部基于全量数据重跑一遍，第一天的数据是10万条，基于10万条数据把整个数据仓库全量重跑一遍

ods层的事实表有全量数据300万条，维度表有5000条数据，此时还是要基于300万+5000把全量的数据全部重跑一遍

这就是之前我们讲的那种，如果你的维度表的属性值变化了，那么收集来binlog日志，接着就会通过spark直接进行merge，维度表里仅仅保留最新的值，同时从ods层开始，所有的etl全部全量重算，所有报表重新计算一遍

这个是比较常见的一种做法

比如说某个商品所属的部门，变化了，那么就肯定对应的报表重新计算

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/104~113资料/108\_第四种缓慢变化维：类型3，增加属性列/笔记.docx**

《108\_第四种缓慢变化维：类型3，增加属性列》

product\_no product\_id name department start\_date end\_date status

111223 110 iphone手机 电子产品部 2020-01-01 2020-01-31 expired

223546 110 iphone手机 智能终端部 2020-02-01 9999-12-31 valid

product\_no

product\_id

name

current\_department ：智能终端部

expired\_department：电子产品部

expired\_department\_start\_date

expired\_department\_end\_date

产品部 -> 产品第一分部 -> 电子产品部 -> 智能终端部

这样你就可以用最新部门来查看历史所有数据，也可以根据过期部门的截止日期，看到最新部门变更后的数据，比较灵活，但是涉及到更新表字段了，这个就会比较麻烦一些，除非有这种需求，再这么干

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/104~113资料/105\_第一种缓慢变化维：类型0，值永远不变/笔记.docx**

《105\_第一种缓慢变化维：类型0，值永远不变》

员工维度表：姓名、身份证号、性别、出生日期，这些属性值其实都属于永远不会变化的

比如说类似于员工姓名之类的维度属性值，是永远不会变的

玩儿数仓的时候，你呢对于每个事实表和维度表，必须建立起来一个动态的思维意识，如果你看到一张维度表，必须脑子里要有一个区分，就是这个维度表，他的各个属性的值是否会变化

一个维度表里的属性值，哪些是不会变的，哪些是会变的，针对于会变的那些值，假设有一天他变化了，你应该如何来处理呢

从业务系统里抽取出来的事实数据，要考虑一下这个事实数据的事实字段，比如说订单的金额，折扣金额，这些东西一般都是不变的，但是呢，一些简单的退化维度，比如说订单的状态，可能是会变化的

# **File Path: /Users/xiaotingting/Downloads/学习资料/10\_大数据面试突击之离线数仓unzip/104~113资料/113\_本周作业：从自己公司的业务场景里找一个适合拉链表的场景出来/笔记.docx**

《113\_本周作业：从自己公司的业务场景里找一个适合拉链表的场景出来》

拉链表，必须在自己公司里去找，找一个场景，把你的一些维度表，事实表，做成拉链表，start\_date，end\_date，拉链表的数据如何生成，参考性的ETL，在拉链表里，如果每天有新增那么就插入数据，如果是删除，那就删除数据，如果是更新，直接把更新的快照合并到全量表里去，作为拉链数据

拉链表有了，后续的每一层的ETL如何基于拉链表里的增量数据进行计算，避免全量计算，这个也需要自己去考虑一下