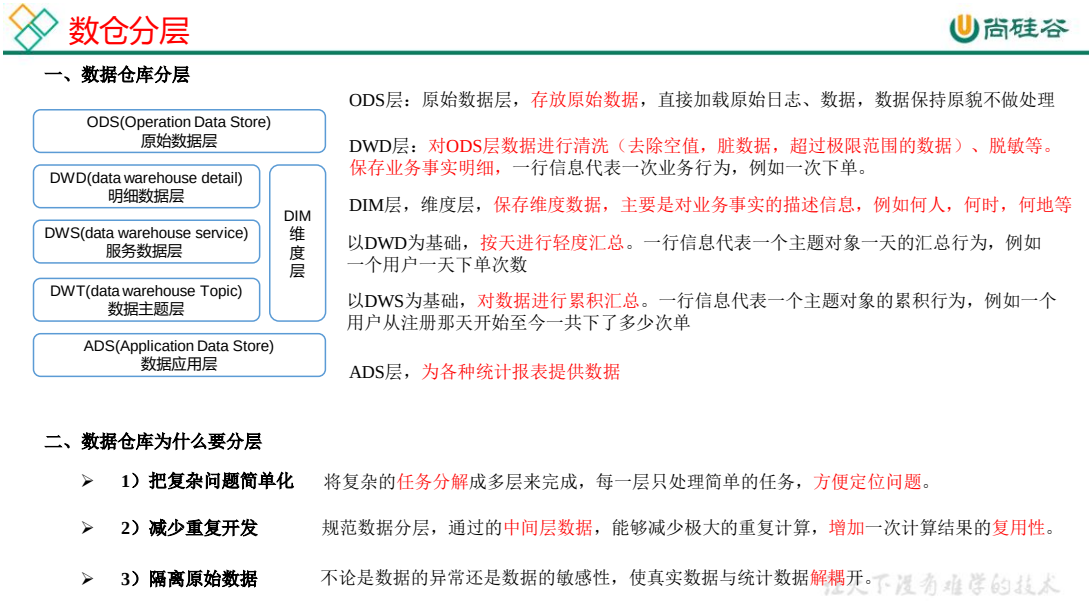
尚硅谷大数据项目之尚品汇（数据仓库系统）

(作者：尚硅谷研究院)

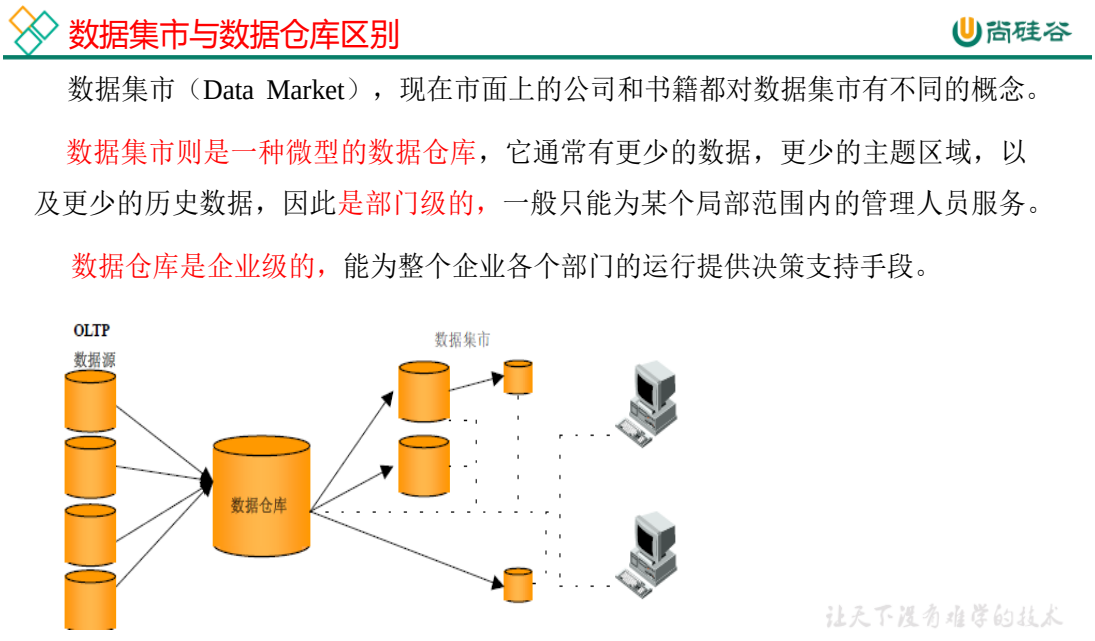
版本：V4.2.0

第 1 章 数仓分层

1.1 为什么要分层



1.2 数据集市与数据仓库概念



1.3 数仓命名规范

1.3.1 表命名

- ODS层命名为ods_表名
- DIM层命名为dim_表名
- DWD层命名为dwd_表名
- DWS层命名为dws_表名
- DWT层命名为dwt_表名
- ADS层命名为ads_表名
- 临时表命名为tmp_表名

1.3.2 脚本命名

- 数据源_to_目标_db/log.sh
- 用户行为脚本以log为后缀；业务数据脚本以db为后缀。

1.3.3 表字段类型

- 数量类型为bigint
- 金额类型为decimal(16, 2)，表示：16 位有效数字，其中小数部分 2 位
- 字符串(名字，描述信息等)类型为string
- 主键外键类型为string
- 时间戳类型为bigint

第 2 章 数仓理论

2.1 范式理论

2.1.1 范式概念

1）定义

数据建模必须遵循一定的规则，在关系建模中，这种规则就是范式。

2）目的

采用范式，可以降低数据的冗余性。

为什么要降低数据冗余性？

（1）十几年前，磁盘很贵，为了减少磁盘存储。

（2）以前没有分布式系统，都是单机，只能增加磁盘，磁盘个数也是有限的

（3）一次修改，需要修改多个表，很难保证数据一致性

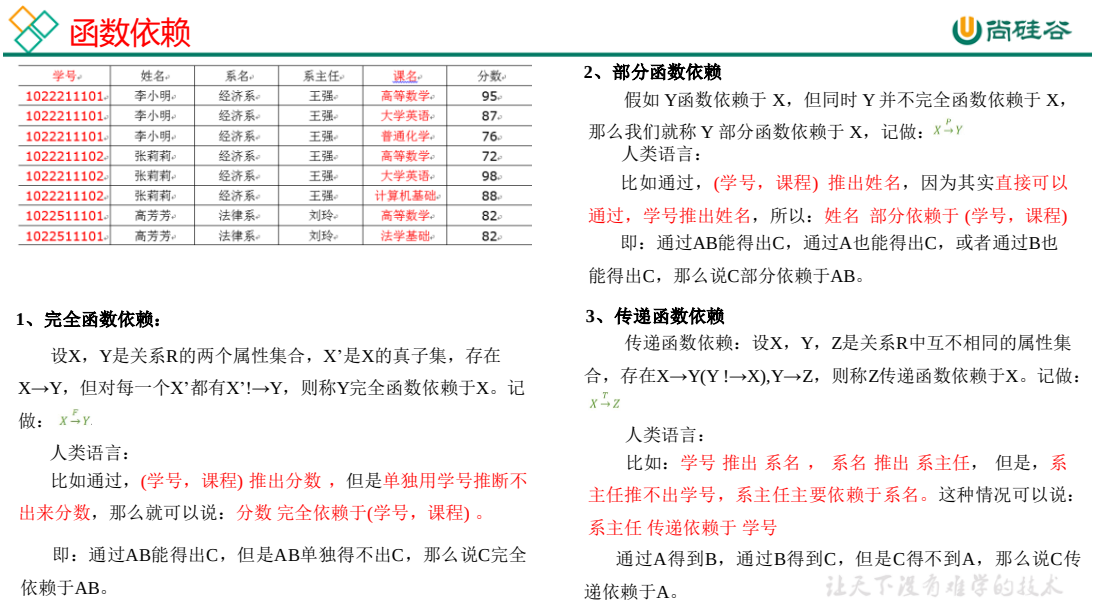
3）缺点

范式的缺点是获取数据时，需要通过 Join 拼接出最后的数据。

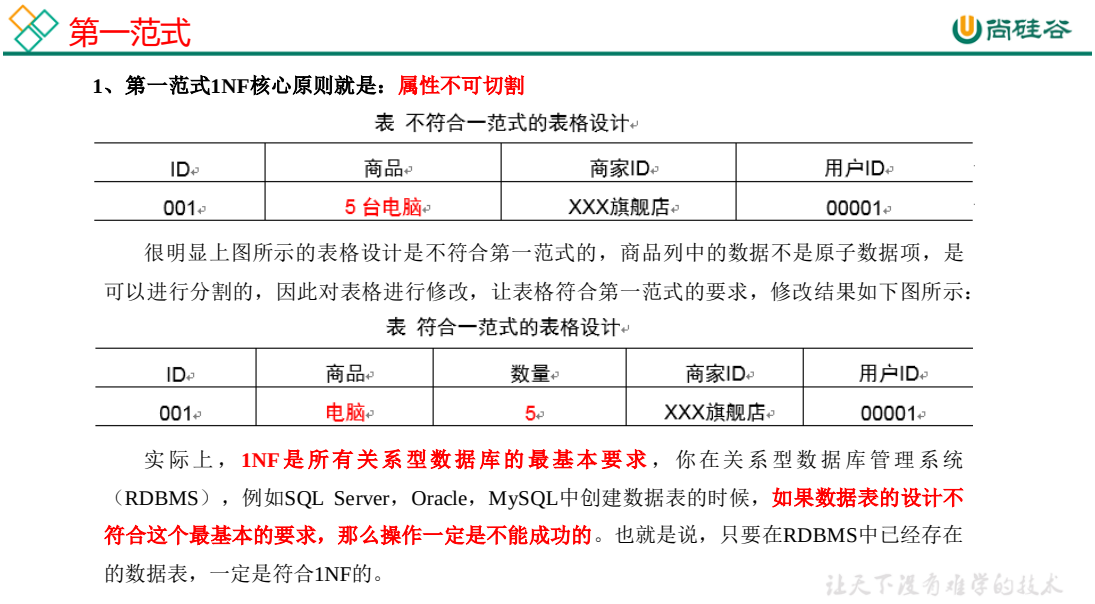
4）分类

目前业界范式有：第一范式(1NF)、第二范式(2NF)、第三范式(3NF)、巴斯-科德范式(BCNF)、第四范式(4NF)、第五范式(5NF)。

2.1.2 函数依赖



2.1.3 三范式区分



2、第二范式2NF核心原则：不能存在“部分函数依赖”

学号	姓名	系名	系主任	课名	分数
1022211101	李小明	经济系	王强	高等数学	95
1022211101	李小明	经济系	王强	大学英语	87
1022211101	李小明	经济系	王强	普通化学	76
1022211102	张莉莉	经济系	王强	高等数学	72
1022211102	张莉莉	经济系	王强	大学英语	98
1022211102	张莉莉	经济系	王强	计算机基础	88
1022511101	高芳芳	法律系	刘玲	高等数学	82
1022511101	高芳芳	法律系	刘玲	法学基础	82

以上表格明显存在，部分依赖。比如，这张表的主键是（学号，课名），分数确实完全依赖于(学号，课名），但是姓名并不完全依赖于(学号，课名）

学号	课名	分数
1022211101	高等数学	95
1022211101	大学英语	87
1022211101	普通化学	76
1022211102	高等数学	72
1022211102	大学英语	98

学号	课名	分数
1022211102	计算机基础	88
1022511101	高等数学	82
1022511101	法学基础	82

学号	姓名	系名	系主任
1022211101	李小明	经济系	王强
1022211102	张莉莉	经济系	王强
1022511101	高芳芳	法律系	刘玲

以上符合第二范式，去掉部分函数依赖依赖

3、第三范式 3NF核心原则：不能存在传递函数依赖

在下面这张表中，存在传递函数依赖：学号->系名->系主任，但是系主任推不出学号。

学号	姓名	系名	系主任
1022211101	李小明	经济系	王强
1022211102	张莉莉	经济系	王强
1022511101	高芳芳	法律系	刘玲

上面表需要再次拆解：

学号	姓名	系名
1022211101	李小明	经济系
1022211102	张莉莉	经济系
1022511101	高芳芳	法律系

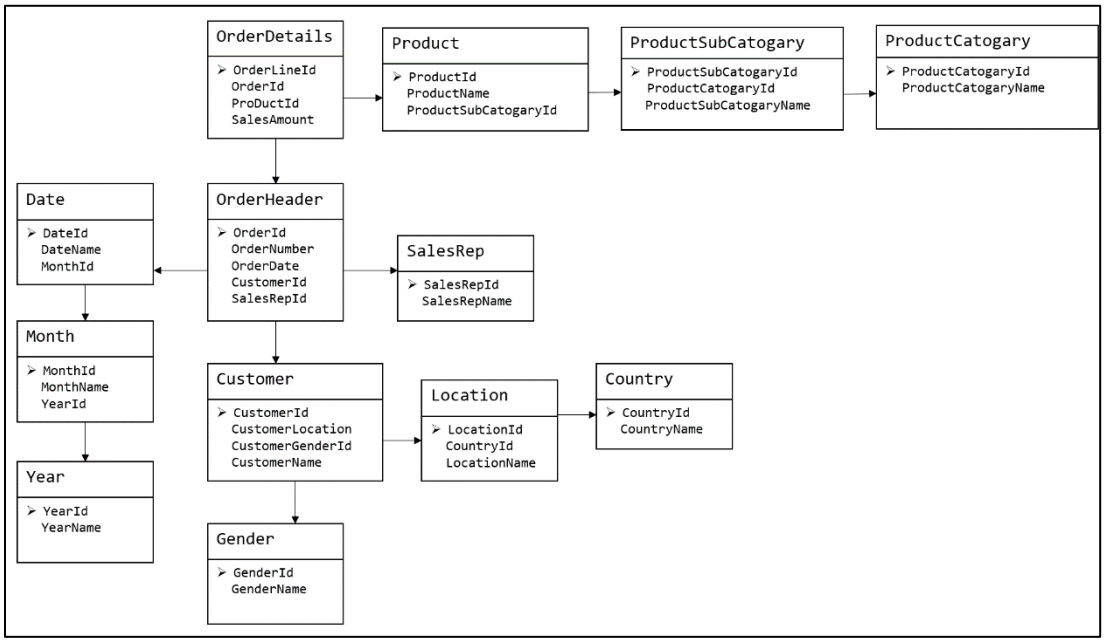
系名	系主任
经济系	王强
法律系	刘玲

2.2 关系建模与维度建模

关系建模和维度建模是两种数据仓库的建模技术。关系建模由 Bill Inmon 所倡导，维度建模由 Ralph Kimball 所倡导。

2.2.1 关系建模

关系建模将复杂的数据抽象为两个概念——实体和关系，并使用规范化的方式表示出来。关系模型如图所示，从图中可以看出，较为松散、零碎，物理表数量多。



关系模型严格遵循第三范式（3NF），数据冗余程度低，数据的一致性容易得到保证。由于数据分布于众多的表中，查询会相对复杂，在大数据的场景下，查询效率相对较低。

2.2.2 维度建模

维度模型如图所示，从图中可以看出，模型相对清晰、简洁。

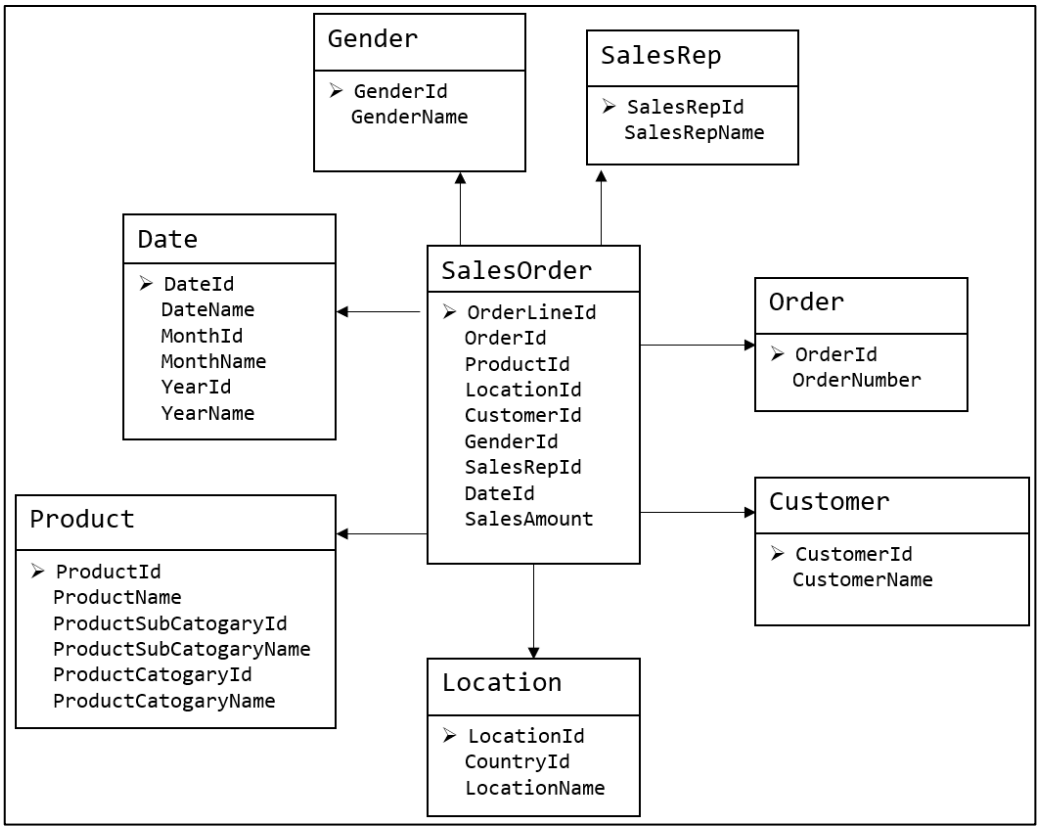


图 维度模型示意图

维度模型以数据分析作为出发点，不遵循三范式，故数据存在一定的冗余。维度模型面向业务，将业务用事实表和维度表呈现出来。表结构简单，故查询简单，查询效率较高。

2.3 维度表和事实表（重点）

2.3.1 维度表

维度表：一般是对事实的**描述信息**。每一张维表对应现实世界中的一个对象或者概念。例如：用户、商品、日期、地区等。

维表的特征：

- 维表的范围很宽（具有多个属性、列比较多）
- 跟事实表相比，行数相对较小：通常< 10 万条
- 内容相对固定：编码表

时间维度表：

日期 ID	day of week	day of year	季度	节假日
2020-01-01	2	1	1	元旦
2020-01-02	3	2	1	无
2020-01-03	4	3	1	无
2020-01-04	5	4	1	无
2020-01-05	6	5	1	无

2.3.2 事实表

事实表中的每行数据代表一个业务事件（下单、支付、退款、评价等）。“事实”这个术语表示的是业务事件的**度量值（可统计次数、个数、金额等）**，例如，2020 年 5 月 21 日，宋宋老师在京东花了 250 块钱买了一瓶海狗人参丸。维度表：时间、用户、商品、商家。事实表：250 块钱、一瓶

每一个事实表的行包括：具有可加性的数值型的度量值、与维表相连接的外键，通常具有两个和两个以上的外键。

事实表的特征：

- 非常的大
- 内容相对的窄：列数较少（主要是外键 id 和度量值）
- 经常发生变化，每天会新增加很多。

1）事务型事实表

以**每个事务或事件为单位**，例如一个销售订单记录，一笔支付记录等，作为事实表里的一行数据。一旦事务被提交，事实表数据被插入，数据就不再进行更改，其更新方式为增量

更新。

2）周期型快照事实表

周期型快照事实表中**不会保留所有数据**，**只保留固定时间间隔的数据**，例如每天或者每月的销售额，或每月的账户余额等。

例如购物车，有加减商品，随时都有可能变化，但是我们更关心每天结束时这里面有多少商品，方便我们后期统计分析。

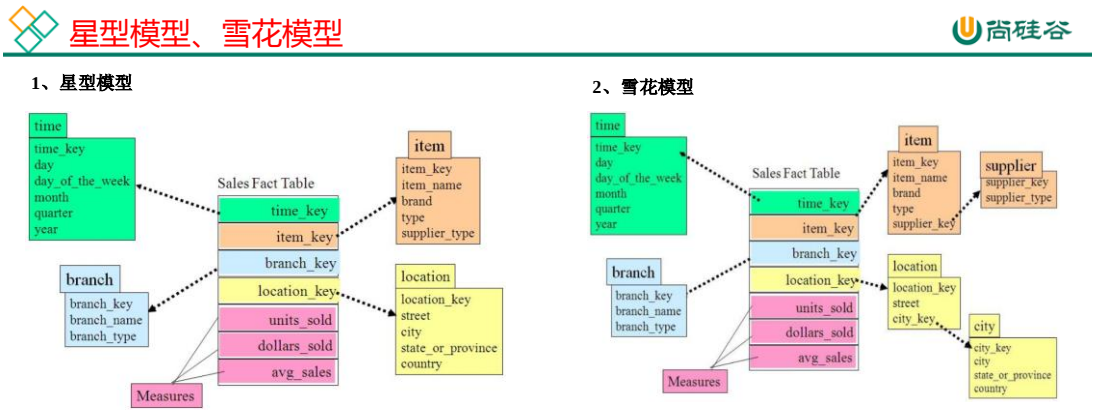
3）累积型快照事实表

累计快照事实表用于跟踪业务事实的变化。例如，数据仓库中可能需要累积或者存储订单从下订单开始，到订单商品被打包、运输、和签收的各个业务阶段的时间点数据来跟踪订单声明周期的进展情况。当这个业务过程进行时，事实表的记录也要不断更新。

订单 id	用户 id	下单时间	打包时间	发货时间	签收时间	订单金额
		3-8	3-8	3-9	3-10	

2.4 维度模型分类

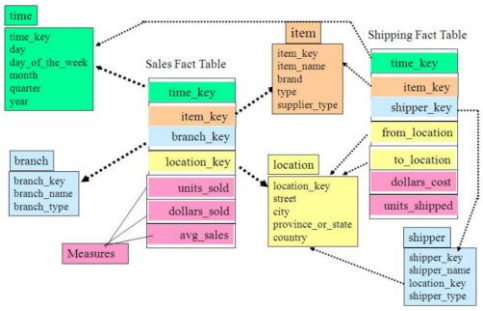
在维度建模的基础上又分为三种模型：星型模型、雪花模型、星座模型。



雪花模型与星型模型的区别主要在于维度的层级，标准的星型模型维度只有一层，而雪花模型可能会涉及多级。

雪花模型，比较靠近3NF，但是无法完全遵守，因为遵循3NF的性能成本太高。

3、星座模型



星座模型与前两种情况的区别是事实表的数量，星座模型是基于多个事实表。

基本上是很多数据仓库的常态，因为很多数据仓库都是多个事实表的。所以星座不星座只反映是否有多个事实表，他们之间是否共享一些维度表。

所以星座模型并不和前两个模型冲突。

4、模型的选择

首先就是星座不星座这个只跟数据和需求有关系，跟设计没关系，不用选择。

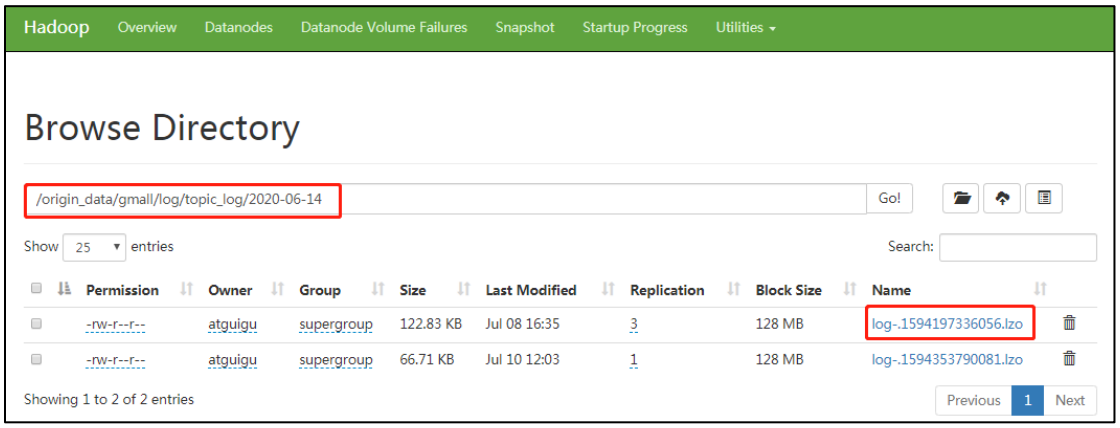
星型还是雪花，取决于性能优先，还是灵活更优先。

目前实际企业开发中，不会绝对选择一种，根据情况灵活组合，甚至并存（一层维度和多层维度都保存）。但是整体来看，更倾向于维度更少的星型模型。尤其是Hadoop体系，减少Join就是减少Shuffle，性能差距很大。（关系型数据可以依靠强大的主键索引）

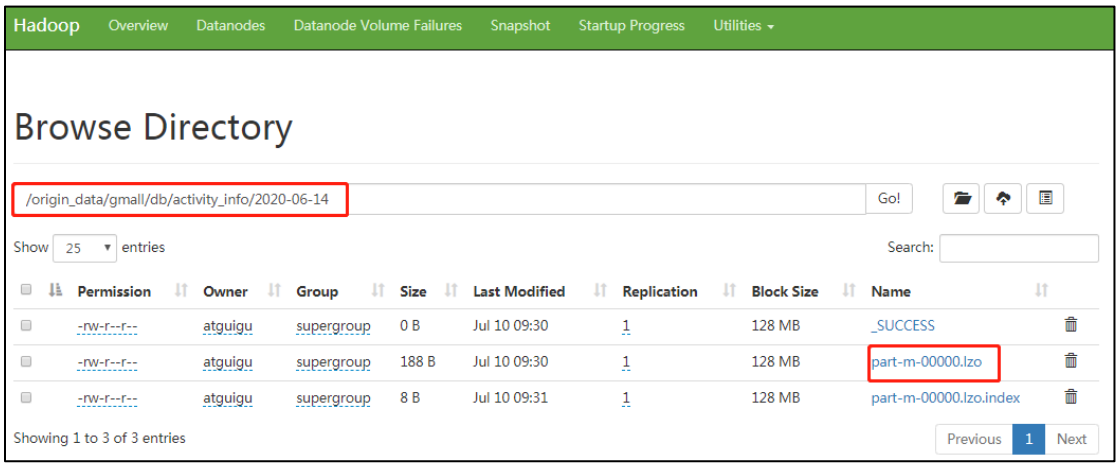
2.5 数据仓库建模（绝对重点）

2.5.1 ODS 层

1）HDFS 用户行为数据



2）HDFS 业务数据



3）针对 HDFS 上的用户行为数据和业务数据，我们如何规划处理？

（1）保持数据原貌不做任何修改，起到备份数据的作用。

（2）数据采用压缩，减少磁盘存储空间（例如：原始数据 100G，可以压缩到 10G 左右）

（3）创建分区表，防止后续的全表扫描

2.5.2 DIM 层和 DWD 层

DIM 层 DWD 层需构建维度模型，一般采用星型模型，呈现的状态一般为星座模型。

维度建模一般按照以下四个步骤：

选择业务过程→声明粒度→确认维度→确认事实

（1）选择业务过程

在业务系统中，挑选我们感兴趣的业务线，比如下单业务，支付业务，退款业务，物流业务，一条业务线对应一张事实表。

（2）声明粒度

数据粒度指数据仓库的数据中保存数据的细化程度或综合程度的级别。

声明粒度意味着精确定义事实表中的一行数据表示什么，应该尽可能选择**最小粒度**，以此来应各种各样的需求。

典型的粒度声明如下：

订单事实表中一行数据表示的是一个订单中的一个商品项。

支付事实表中一行数据表示的是一个支付记录。

（3）确定维度

维度的主要作用是描述业务是事实，主要表示的是“谁，何处，何时”等信息。

确定维度的原则是：后续需求中是否要分析相关维度的指标。例如，需要统计，什么时间下的订单多，哪个地区下的订单多，哪个用户下的订单多。需要确定的维度就包括：时间维度、地区维度、用户维度。

（4）确定事实

此处的“事实”一词，指的是业务中的度量值（次数、个数、件数、金额，可以进行累加），例如订单金额、下单次数等。

在 DWD 层，以**业务过程**为建模驱动，基于每个具体业务过程的特点，构建**最细粒度**的明细层事实表。事实表可做适当的宽表化处理。

事实表和维度表的关联比较灵活，但是为了应对更复杂的业务需求，可以将能关联上的表尽量关联上。

	时间	用户	地区	商品	优惠券	活动	度量值
订单	√	√	√				运费/优惠金额/原始金额/最终金额
订单详情	√	√	√	√	√	√	件数/优惠金额/原始金额/最终金额
支付	√	√	√				支付金额
加购	√	√		√			件数/金额
收藏	√	√		√			次数
评价	√	√		√			次数
退单	√	√	√	√			件数/金额
退款	√	√	√	√			件数/金额
优惠券领用	√	√			√		次数

至此，数据仓库的维度建模已经完毕，DWD 层是以业务过程为驱动。

DWS 层、DWT 层和 ADS 层都是以需求为驱动，和维度建模已经没有关系了。

DWS 和 DWT 都是建宽表，按照主题去建表。主题相当于观察问题的角度。对应着维度表。

2.5.3 DWS 层与 DWT 层

DWS 层和 DWT 层统称宽表层，这两层的设计思想大致相同，通过以下案例进行阐述。

1）问题引出：两个需求，统计每个省份订单的个数、统计每个省份订单的总金额

2）处理办法：都是将省份表和订单表进行 join，group by 省份，然后计算。同样数据被计算了两次，实际上类似的场景还会更多。

那怎么设计能避免重复计算呢？

针对上述场景，可以设计一张地区宽表，其主键为地区 ID，字段包含为：下单次数、下单金额、支付次数、支付金额等。上述所有指标都统一进行计算，并将结果保存在该宽表

中，这样就能有效避免数据的重复计算。

3）总结：

（1）需要建哪些宽表：以维度为基准。

（2）宽表里面的字段：是站在不同维度的角度去看事实表，重点关注事实表聚合后的度量值。

（3）DWS 和 DWT 层的区别：DWS 层存放的所有主题对象当天的汇总行为，例如每个地区当天的下单次数，下单金额等，DWT 层存放的是所有主题对象的累积行为，例如每个地区最近 7 天（15 天、30 天、60 天）的下单次数、下单金额等。

2.5.4 ADS 层

对电商系统各大主题指标分别进行分析。