> 导读：数据仓库的建设实施和落地需要团队中不同成员的参与和配合，需要各种各样的规范，规范的分层定义和表命名能让使用者轻而易举地明白该表的作用和含义。因此本文档重点介绍分层规范和可落地的表命名规范。

## 01 数据分层

### 一、数据运营层：ODS(Operational Data Store)

ODS层，是最接近数据源中数据的一层，为了考虑后续可能需要追溯数据问题，因此对于这一层就不建议做过多的数据清洗工作，原封不动地接入原始数据即可，至于数据的去噪、去重、异常值处理等过程可以放在后面的DWD层来做。

### 二、数据仓库层：DW (Data Warehouse)

数据仓库层是我们在做数据仓库时要核心设计的一层，在这里，从 ODS 层中 获得的数据按照主题建立各种数据模型 。DW 层又细分为DWD(Data Warehouse Detail )层、DWM(Data WareHouse Middle )层和DWS(Data WareHouse Servce )。

#### 1. 数据明细层：DWD（Data Warehouse Detail)

该层一般保持和ODS层一样的数据粒度 ， 并且提供一定的数据质量保证 。

DWD层要做的就是将数据清理、整合、规范化、脏数据、垃圾数据、规范不一致的、状态定义不一致的、命名不规范的数据都会被处理。

同时 ， 为了提高数据明细层的易用性 ， 该层会采用一些维度退化手法 ， 将维度退化至事实表中， 减少事实表和维表的关联。另外，在该层也会做一部分的数据聚合，将相同主题的数据汇集到一 张表中 ， 提高数据的可用性 。

#### 2. 数据中间层：DWM（Data WareHouse Middle)

该层会在 DWD 层的数据基础上，对数据做轻度的聚合操作，生成一系列的中间表，提升公共指标的复用性， 减少重复加工。直观来讲 ， 就是对通用的核心维度进行聚合操作， 算出相应的统计指标。

在实际计算中 ， 如果直接从 DWD 或者 ODS 计算出宽表的统计指标 ， 会存在 计算量太大并且维度太少的问题，因此一般的做法是，在 DWM 层先计算出多个小的 中间表 ， 然后再拼接成一 张 DWS 的宽表 。 由于宽和窄的界限不易界定，也可以去掉 DWM 这一层， 只留 DWS 层，将所有的数据再放在DWS 亦可。

#### 3.数据服务层：DWS（Data WareHouse Servce)

DWS 层为公共汇总层， 会进行轻度汇总 ， 粒度比明细数据稍粗，基于DWD层上的基础数据， 整合汇总成分析

某一个主题域的服务数据，一般是宽表。DWS层应覆盖 80%的应用场景 。又称数据集市或宽表。

按照业务划分，如主题域流量、订单、用户等 ，生成字段比较多的宽表 ， 用 于提供后续的业务查询，OLAP分析，数据分发等。一般来讲，该层的数据表会相对比较少，一张表会涵盖比较多的业务内容，由于其字段较多，因此一般也会称该层的表为宽表。

**三、数据应用层：APP（Application)**

在这里，主要是提供给数据产品和数据分析使用的数据 ，一般会存放在ES 、 PostgreSql、Redis 等系统中供线上系统使用 ，也可能会存在Hive或者Druid中供数据分析和数据挖掘使用 。比如我们经常说的报表数据，一般就放在这里。

**四、维表层 (Dimension)**

最后补充一个维表层，维表层主要包含两部分数据：高基数维度数据：一般是用户资料表、商品资料表类似的资料表。数据量可能是千万级或者上亿级别。

低基数维度数据：一般是配置表，比如枚举值对应的中文含义，或者日期维表。

数据量可能是个位数或者几千几万。


## 02 表规范

**关于词根**

词根属于数仓建设中的规范 ，属于元数据管理的范畴，现在把这个划到数据治理的一部分。完整的数仓建设是包

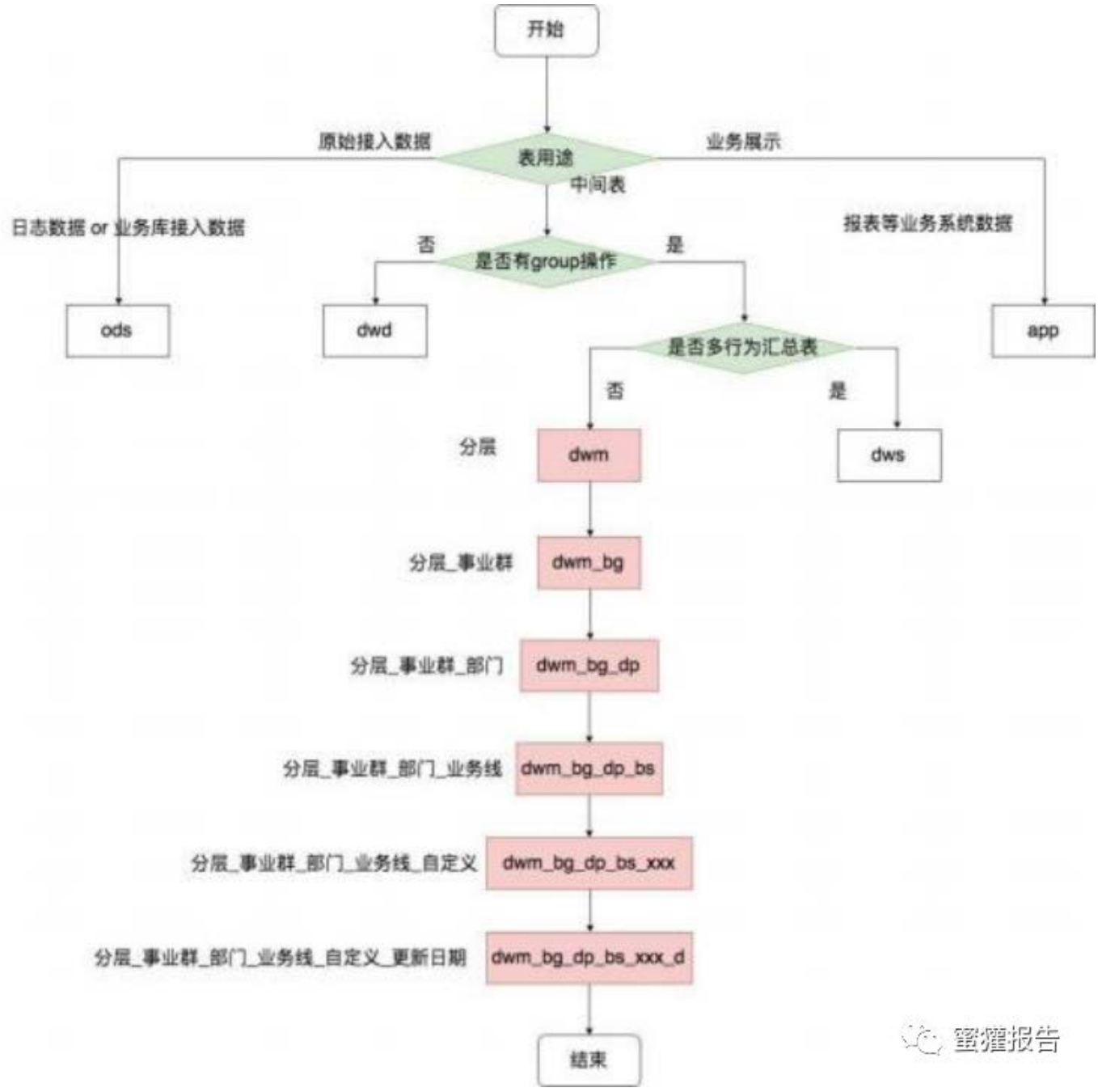
含数据治理的，只是现在谈到数仓偏向于数据建模， 而谈到数据治理，更多的是关于数据规范、数据管理。

表命名 ，其实在很大程度上是对元数据描述的一种体现 ，表命名规范越完善 ，我们能从表名获取到的信息就越多。比如：一部分业务是关于货架的，英文名是：rack ， rack 就是一个词根，那我们就在所有的表、字段等用到的地方都叫 rack， 不要叫成 别的什么 。 这就是词根的作用， 用来统一命名，表达同一个含义。

指标体系中有很多"率"的指标，都可以拆解成XXX+率 ，率可以叫 rate ，那我 们所有的指标都叫做 XXX+rate。

词根： 可以用来统一表名、字段名、主题域名等等。

举例： 以流程图的方式来展示，更加直观和易懂，本图侧重 dwm 层表的命名 规范， 其余命名是类似的道理：




第一个判断条件是该表的用途，是中间表、原始日志还是业务展示用的表如果该表被判断为中间表， 就会走入下一个判断条件：表是否有 group 操作 通过是否有 group 操作来判断该表该划分在 dwd 层还是 dwm 和 dws 层如

果不是 dwd 层，则需要判断该表是否是多个行为的汇总表(即宽表)。

最后再分别填上事业群 、部门、业务线、自定义名称和更新频率等信息即可。

分层 ： 表的使用范围
事业群和部门：生产该表或者该数据的团队
业务线 ： 表明该数据是哪个产品或者业务线相关
主题域 ： 分析问题的角度，对象实体
自定义 ： 一般会尽可能多描述该表的信息，比如活跃表、留存表等
更新周期 ： 比如说天级还是月级更新

**1、常规表**

常规表是我们需要固化的表，是正式使用的表，是目前一段时间内需要去维护去 完善的表。

规范：分层前缀[dwd|dws|ads]_部门_业务域_主题域_XXX_更新周期|数据范围。
业务域、主题域我们都可以用词根的方式枚举清楚，不断完善， 更新周期主要的是时间粒度、日、月、年、周等。

**2 、中间表**

中间表一般出现在 Job 中，是 Job 中临时存储的中间数据的表，中间表的作 用域只限于当前 Job 执行过程中，Job 一旦执行完成，该中间表的使命就完 成了，是可以删除的 (按照自己公司的场景自由选择，以前公司会保留几天 的中间表数据，用来排查问题) 。

规范：mid_table_name_[0~9|dim]
table_name 是我们任务中目标表的名字，通常来说一个任务只有一个目标表。 这里加上表名，是为了防止自由发挥的时候表名冲突，而末尾大家可以选择自 由发挥，起一些有意义的名字，或者简单粗暴，使用数字代替，各有优劣吧，谨慎选择。

通常会遇到需要补全维度的表，这里使用dim 结尾。

中间表在创建时，请加上 ,如果要保留历史的中间表，可以加上日期或者时间戳。

**3 、临时表**

临时表是临时测试的表，是临时使用一次的表，就是暂时保存下数据看看，后续一般不再使用的表，是可以随时删除的表。

规范：tmp_xxx

只要加上 tmp 开头即可，其他名字随意，注意 tmp 开头的表不要用来实际使用，只是测试验证而已。

### 4 、维度表

维度表是基于底层数据，抽象出来的描述类的表。维度表可以自动从底层表抽象
出来，也可以手工来维护。

规范：dim_xxx

维度表，统一以 dim 开头，后面加上，对该指标的描述，可以自由发挥。

### 5 、手工表

手工表是手工维护的表，手工初始化一次之后，一般不会自动改变，后面变更，
也是手工来维护。

一般来说，手工的数据粒度是偏细的，所以，暂时我们统一放在dwd层，后面　如果有目标值或者其他类型手工数据，再根据实际情况分层。

规范：dwd_业务域_manual_xxx

手工表，增加特殊的主题域，manual ，表示手工维护表。

## 03 指标规范

### 3.1 命名

小写
下划线分割
可读性优于长度(词根，避免出现同一个指标，命名一致性)数量字段后缀_cnt等标识...
金额字段后缀_price标识
禁止使用sql关键字

### 3.2 字段格式

浮点数使用decimal(28,6)控制精度等

### 3.3 NULL 字段处理

对于维度字段，需设置为-1
对于指标字段，需设置为0

## 04 口径规范

保证主题域内，指标口径一致，无歧义。另外，还需要注意同一口径的指标数据来源要一致。

## 05 数据处理方式

### 1.增量表：

新增数据，增量数据是上次导出之后的新数据。

(1) 记录每次增加的量，而不是总量；
(2) 增量表，只报变化量，无变化不用报
(3) 每天一个分区

| 增量 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 日期 | A1 | A2 | A3 | | 日期 | A1 | A2 | A3 |
| 2019-04-01 | A | A | A | | 2019-04-01 | A | A | A |
| 2019-04-02 | B | B | B | | 2019-04-02 | B | B | B |
| 2019-04-03 | C | C | C | | 2019-04-03 | C | C | C |



### 2 、全量表

每天的所有的最新状态的数据。

(1) 全量表，有无变化，都要报
(2) 每次上报的数据都是所有的数据 (变化的 + 没有变化的)
(3) 只有一个分区

| 全量 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 日期 | A1 | A2 | A3 | | 日期 | A1 | A2 | A3 |
| 2019-04-01 | A | A | A | | 每天一个全部 | A | A | A |
| 2019-04-02 | B | B | B | | | B | B | B |
| 2019-04-03 | C | C | C | | | C | C | C |



### 3、快照表

按日分区，记录截止数据日期的全量数据。

(1) 快照表，有无变化，都要报
(2) 每次上报的数据都是所有的数据(变化的+没有变化的)
(3) 一天一个分区

快照

| 日期 | A1 | A2 | A3 |
|---|---|---|---|
| 2019-04-01 | A | A | A |
| 2019-04-02 | B | B | B |
| 2019-04-03 | C | C | C |

| 日期 | A1 | A2 | A3 |
|---|---|---|---|
| 2019-04-01 | A | A | A |
| 2019-04-02 | A | A | A |
| 2019-04-02 | B | B | B |
| 2019-04-03 | A | A | A |
| 2019-04-03 | B | B | B |
| 2019-04-03 | C | C | C |



**4、拉链表**

记录截止数据日期的全量数据。

(1) 记录一个事物从开始，一直到当前状态的所有变化的信息；
(2) 拉链表每次上报的都是历史记录的最终状态，是记录在当前时刻的历史总量；
(3) 当前记录存的是当前时间之前的所有历史记录的最后变化量(总量)；
(4) 只有一个分区

拉链

2019-04-01

| key | A1 | A2 | A3 |
|---|---|---|---|
| 1 | A | A | A |
| 2 | B | B | B |
| 3 | C | C | C |

2019-04-02

| key | A1 | A2 | A3 |
|---|---|---|---|
| 1 | A | A | AA |
| 2 | B | B | B |
| 3 | C | C | C |
| 4 | D | D | D |

| key | A1 | A2 | A3 | start_date | end_date | dp |
|---|---|---|---|---|---|---|
| 1 | A | A | A | 2019-04-01 | 2019-04-02 | EXPRIED |
| 1 | A | A | AA | 2019-04-02 | 2999-12-31 | ACTIVE |
| 2 | B | B | B | 2019-04-01 | 2999-12-31 | ACTIVE |
| 3 | C | C | C | 2019-04-01 | 2999-12-31 | ACTIVE |
| 4 | D | D | D | 2019-04-02 | 2999-12-31 | ACTIVE |



数据治理体系
持续完善数据治理实战体系，数据仓库、标签、指标体系，实现业务数字化，数字资产化，资产业务化，资产资
31篇原创内容

公众号

收录于合集 #数据仓库
15

上一篇
有了数据湖，数据仓库究竟能不能被取代？他们又有什么样的区别呢？