# 数据仓库

## 学习目标

理解OLTP和OLAP的区别

理解数据仓库的特点

理解数据仓库系统架构

理解指标与维度

理解下钻与上卷

理解事实表与维度表

理解星型模型和雪花模型

理解缓慢渐变维

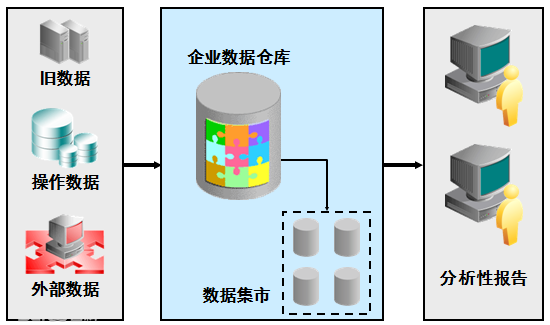
掌握数据仓库的分层方法

## 数据仓库介绍

### 数据仓库概念

数据仓库概念创始人在《建立数据仓库》一书中对数据仓库的定义是：数据仓库（Data Warehouse）是一个面向主题的（Subject Oriented）、数据集成的（Integrated）、相对稳定(非易失)的（Non-Volatile）、反映历史变化(时变)（Time Variant）的数据集合，用于支持管理决策(Decision Making Support)。

数据仓库是决策支持系统（dss）的结构化数据环境，如下图，决策支持系统基于数据仓库进行联机分析处理 ( OLAP ) 。常用的技术有，HDFS、HBase、Hive、SparkSql等。



1. 数据采集, 将源数据采集到数据仓库
2. 基于数据仓库进行数据分析
3. 生成报表

### OLTP和OLAP区别

OLTP（On-Line **Transaction** Processing）即联机事务处理，也称为面向交易的处理过程，其基本特征是前台接收的用户数据可以立即传送到计算中心进行处理，并在很短的时间内给出处理结果，是对用户操作快速响应的方式之一，比如ERP系统，CRM系统，互联网电商系统等，这类系统的特点是事务操作频繁，数据量小。

OLAP（On-Line **Analytical** Processing）即联机分析处理，有时也称为决策支持系统（DSS），支持复杂的分析操作，侧重决策支持，并且提供直观易懂的查询结果。这类系统的特点是没有事务性操作，主要是查询操作，数据量大。

详细的区别如下：

| 对比项目 | OLTP | OLAP |
| --- | --- | --- |
| 功能 | 面向交易的事务处理 | 面向分析查询 |
| 设计 | 面向业务 | 面向主题 |
| 数据 | 最新数据，二维数据 | 历史数据，多维数据 |
| 存储 | M,G ( 存储单位 ) | T、P、E |
| 响应时间 | 快 | 慢 |
| 用户 | 业务操作人员 | 管理决策人员 |

### 数据仓库的特点

#### 面向主题

理解主题的概念可以和数据库应用系统对比理解。

数据库应用是以业务流程来划分应用程序和数据库，比如ERP（Ent\*erp\*rise Resource Planning）包括：进销存系统、人力资源管理系统、财务管理系统、仓库管理系统等，进销存系统管理了进货、销售、存储等业务流程，人力资源系统管理了员工的信息、待遇等相关信息。

数据仓库是以数据分析需求来对数据进行组织划分若干主题，比如销售主题、员工主题、产品主题，主题是一个抽象的概念，可以理解为相关数据的分类、目录等，通过销售主题可以进行销售相关的分析，如年度销量排行、月度订单量统计等。

总之，主题是以分析需求为导向来组织数据，数据库应用系统是以业务流程为导向来组织数据，注意：主题中的数据是跨应用系统的。

#### 数据集成

主题中的数据是跨应用系统的，也就是说数据是分散在各各应用系统，比如销售数据在进销存系统中有，财务系统中也有，为了进行销售分析需要将销售数据进行集成，集成在销售主题中，就可以从销售主题来进行数据分析。

#### 非易失

数据库应用系统是根据业务需求进行数据处理和存储，而数据仓库是根据数据分析需求来进行数据存储，数据仓库中的数据用于查询和分析，为了保证数据分析的准确性和稳定性，数据仓库中的数据一般是很少更新的，会将历史快照保存下来。

#### 时变

数据仓库中的数据存储的是历史数据，历史数据是随时间变化的，比如历年的销售数据都会存储到数据仓库中，即使数据仓库中的数据很少更新，但也不能保证没有变化，如下需求：

1）会不断添加新数据

每年的销售数据会逐渐添加到数据仓库。

2）删除过期数据

数据仓库中的数据会保存很长的时间（5--10年），但也有过期时间，到过期 时间会删除过期 数据。

3）对历史明细数据进行聚合

为了方便数据分析，根据分析需求会将比较细粒度的数据进行数据聚合存储，这也是时变的一种表现，比如：为了方便统计年度销售额会将销售记录按月进行统计，统计年度销售额时只需要针对月度销售结果进行统计即可。

## 数据仓库系统架构

### 系统结构图

数据仓库提供企业决策分析的数据环境，数据从哪里获取？数据如何存储到数据仓库？决策分析系统如何从数据仓库获取数据进行分析？我们可以把数据从获取、存储到数据仓库、数据分析的所有部分称为一个数据仓库系统，本节讲解数据仓库系统的工作流程和系统架构。

下图是数据仓库系统的结构图：



以下系统各部分的执行流程是：

1、确定分析所依赖的源数据。

2、通过ETL将源数据采集到数据仓库。

3、数据按照数据仓库提供的主题结构进行存储。

4、根据各部门的业务分析要求创建数据集市（数据仓库的子集）。

5、决策分析、报表等应用系统从数据仓库查询数据、分析数据。

6、用户通过应用系统查询分析结果、报表。

### 源数据

源数据是指用于分析的原始数据，这一步主要是根据分析需求确定源数据，这个数据分布在内部系统和外部分系统中，内部数据主要是企业ERP系统、外部数据是指企业外部分系统所产生的数据，通常是指行业数据。源数据最大的特点是格式不统一，如果要对源数据进行分析需要经过ETL对数据进行集中获取、过虑、转换等处理。

### ETL

ETL（Extra, Transfer, Load）包括数据抽取、数据转换、数据装载三个过程。

1、抽取

​ 数据抽取是从各各业务系统、外部系统等源数据处采集源数据。

2、转换

​ 采集过来的源数据如果要存储到数据仓库需要按照一定的数据格式对源数据进行转换，常见的转换方式有数据类型转换、格式转换、缺失值补充、数据综合等。

3、装载

​ 转换后的数据就可以存储到数据仓库中，这个过程要装载。数据装载通常是按一定的频率进行的，比如每天装载当天的订单数据、每星期装载客户信息等。

### 数据仓库与数据集市

**数据仓库是用于企业整体分析的数据集合，**比如分为：销售主题、客户主题、产品主题等。**数据集市是用于部门分析的数据集合，**从范围上来讲它属于数据仓库的子集，比如：销售部门的数据集市只有销售主题。

为什么会有数据集市的概念？

通常从企业整体出发去建数据仓库比较困难，所涉及到的业务及分析需求比较多，所以提出数据集市的概念，可以先从某个部门开始建设数据仓库，这样效率就比较高。

业界把从企业整体出发建设数据仓库的过程叫自顶向下，把从数据集市开始建设数据仓库再逐渐完善整个数据仓库的过程叫自下向上。通常建议自下向上建设数据仓库，不过这个在业界也存在争议。

数据仓库和数据集市具有什么区别？

1、范围的区别

数据仓库是针对企业整体分析数据的集合。

数据集市是针对部门级别分析的数据集合。

2、数据粒度不同

数据仓库通常包括粒度较细的数据明细。

数据集市则会在数据仓库的基础上进行数据聚合，这些聚合后的数据就会直接用于部门业务分析。

### 应用系统

这里的应用系统是指使用数据仓库完成数据分析、数据查询、数据报表等功能的系统。应用系统需要从数据仓库中查询数据、分析数据，比如：OLAP 系统、数据查询系统等。

### 用户

使用数据仓库系统的用户主要有数据分析人员、管理决策人员（公司高层）等。

## 维度分析

### 维度分析介绍

对数据进行分析通常采取维度分析，比如：用户提出分析课程访问量的指标，为了满足不同的分析需求可以从时间维度分析课程访问量，分析每天、每小时的课程访问量；也可以从课程维度来分析课程访问量，分析每个课程、每个课程分类的访问量。

### 指标与维度

要进行维度分析需要先理解两个术语：指标和维度。

指标是衡量事务发展的标准，也叫度量，如价格，销量等；指标可以求和、求平均值等计算。

指标分为绝对数值和相对数值，绝对数值反映具体的大小和多少，如价格、销量、分数等；相对数值反映一定的程度，如及格率、购买率、涨幅等。

维度是事务的特征，如颜色、区域、时间等，可以根据不同的维度来对指标进行分析对比。比如根据区域维度来分析不同区域的产品销量，根据时间来分析每个月产品的销量，同一个产品销量指标从不同的维度分析会得出不同的结果。

维度分为定性维度和定量维度两种，定性维度就是字符类型的特征，比如区域维度包括全国各省份；定量维度就是数值类型的特征，如价格区间、销量区间等，如价格区间维度分为0--100、100-1000两个区间，可以按价格区间维度来对指标进行分析，说到这里，其实指标是可以转成维度的，所转成的维度就是定量维度。

用具体的指标数值, 来度量不同的维度。x轴和y轴的关系。

### 业务需求

#### 课程访问量

说明：统计指定时间段内，访问课程的客户数量。以天为单位进行展示。

展现：线状图

条件：年、季度、月

展示粒度：天

数据来源：咨询系统的web\_chat\_ems\_2019\_12等月表

按年：显示指定年范围内每天的客户访问量

按季度：显示指定季度范围内每天的客户访问量

按月：显示指定月份范围内每天的客户访问量

#### 课程购买量

说明：统计指定时间段内，用户购买课程的数量。以天为单位进行展示。

展现：线状图

条件：年、季度、月

展示粒度：天

数据来源：咨询系统的web\_chat\_ems\_2019\_12等月表

按年：显示指定年范围内每天的客户访问量

按季度：显示指定季度范围内每天的客户访问量

按月：显示指定月份范围内每天的客户访问量

### 识别维度案例

在日常生活中，我们从不同的角度看待事务会有不同的体会，数据分析也如此，比如：一个在线教育的平台，作为运营方会关注按时间段分析课程的访问量，作为教育机构则关注单个课程的访问量，都是课程访问量指标根据不同的维度去分析得到结果不同，这就是维度分析。

比如：按时间分析课程访问量，时间维度是课程访问量的分析依据，时间维度和业务中的课程访问量是对应的，下表列出了课程访问量明细记录：

| **IP** | **访问时间** | **课程ID** |
| --- | --- | --- |
| 127.0.0.1 | 2019-10-01 00:00:00 | 10010 |
| 127.0.0.1 | 2019-10-01 01:00:00 | 10011 |
| 127.0.0.1 | 2019-10-01 02:00:00 | 10011 |

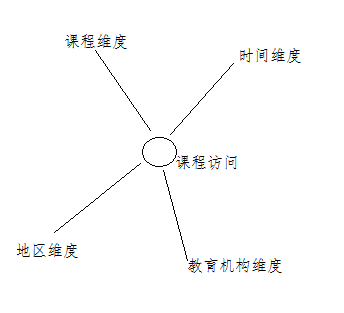
上表中显示了部分课程访问的记录，每条记录表示一次课程访问，记录内容包括：IP，访问时间、课程ID，根据上边的记录可以按时间统计每天所有课程的访问量，时间就是一个维度，如下是按时间维度分析的课程访问量：

**时间维度（天）**

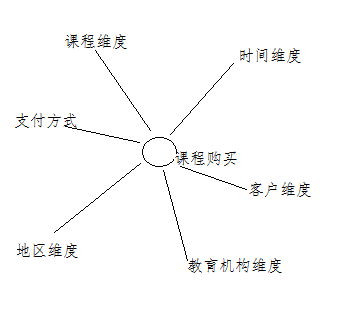
| **时间（天）** | **课程访问量** |
| --- | --- |
| 2019-10-01 | 10000 |
| 2019-10-02 | 9000 |
| 2019-10-03 | 11000 |
| ... | ... |

维度是数据仓库建模的基础，维度是在分析时从多个方面来进行分析，根据上边的例子，将**课程访问量**作为度量，识别的维度主要是时间。

在真实的统计需求中，识别的维度除了时间外，还可能包括：课程、机构（校区）、学科（课程分类）等，如下图：



将**课程购买量**作为度量的维度可能包括：

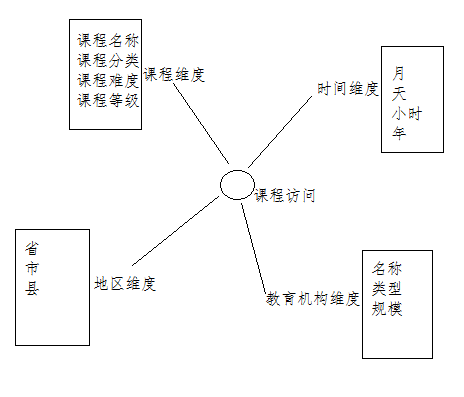


### 维度分层与分级

通常在分析结果中首先看到的是一个总数，比如全年课程购买量，然后会详细去看每个季度、每个月的课程购买量，全年、季度、月这些属于时间维度的一个层次，年、季度、月是这个层次的三个级别；再比如按地区分析课程购买量，全国、省、市、县属于地区维度的一个层次，层次中共有四个级别。

相当于将维度进行细分。细分两层，则维度包含一个层次，多个级别。 细分三层，则维度包含多个层次，多个级别。

每个维度至少有一个层次且该层次至少有一个级别。下边将课程访问的各各维度定义层次和级别，如下：



时间维度：

一个层次四个级别：年、月、天、小时

课程维度：

课程名称：只有一个级别，每门课程的名称

课程分类：两个级别，大类和小类

课程难度：只有三个级别，简单、一般、难

课程等级：只有三个级别，初、中、高

地区维度：

一个层次三个级别：省、市、县

### 下钻与上卷

维度中有不同的层次，每个层次可以有多个级别，这样就可以根据多个维护层次和级别进行分析，可以灵活获取高级别的汇总信息，获取低级别的明细信息。

把获取高级别的汇总信息的过程叫上卷，把获取低级别的明细信息的过程叫下钻，比如：课程访问量分析，时间维度有四个级别，分别是年、月、天、小时，现在我们某个级别分析每天的课程访问量，比如按天分析课程访问量，此时我们可以按小时下钻分析，得出一天内每小时的课程访问量，也可以按月上卷，得到月度的课程访问量。

下钻维度：

天、小时

上卷维度：

年、月

## 数仓建模

### 概述

数据仓库建模的方法常用的有两种：三范式建模法、维度建模法，三范式建模法主要是应用于传统的企业级数据仓库，这类数据仓库通常使用关系型数据库实现，是由Inmon提出的，应用于自顶向下的数据仓库架构; 维度数据模型就是基于维度分析来创建模型，是由Kimball提出，应用于自下向上的数据仓库架构。本课程采用维度建模的方法。

维度建模，简称DM（Dimensional modeling），数据仓库大师Kimball的观点：维度数据模型是一种趋向于支持最终用户对数据仓库进行查询的设计技术，是围绕性能和易理解性构建的。维度模型是按照用户看待或分析数据的角度来组织数据。

维度建模的两个核心概念：事实表和维度表。

### 事实表

#### 概述

事实表记录了特定事件的数字化信息，一般由数值型数字和指向维度表的外键组成。

事实表的设计依赖于业务系统，事实表的数据就是业务系统的指标数据。数据分析的实质就是基于事实表开展的计算操作。

#### 分类

##### 事务事实表

Transaction fact table，事务事实表与周期快照事实表、累积快照事实表使用相同的维度，但是它们在描述业务事实方面是有着非常大的差异的。

事务事实表记录的事务层面的事实，保存的是最原子的数据，也称“原子事实表”或“交易事实表”。事务事实表中的数据在事务事件发生后产生，数据的粒度通常是每个事务一条记录。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。

事务事实表的日期维度记录的是事务发生的日期，它记录的事实是事务活动的内容。用户可以通过事务事实表对事务行为进行特别详细的分析。

沟通中常说的事实表，大多指的是事务事实表。

##### 周期快照事实表

Periodicsnapshot fact table，周期快照事实表以具有规律性的、可预见的时间间隔来记录事实，时间间隔如每天、每月、每年等等。典型的例子如销售日快照表、库存日快照表等。

想象以下场景，需要统计一个季度的商品成交量怎么取做？如果用一个季度内的事实事务表进行计算，虽然可以得出结果但是效率太低，在实际生产中并不可行，因此，需要定期对指定的度量进行整合，作为周期快照表用于下游应用。一般设计事实表时，事务事实表和周期快照表是成对设计的，大部分的周期表由事务表加工产生，还有部分特殊数据是直接应用系统产生（如订单评价）。

周期快照事实表的粒度是每个时间段一条记录，通常比事务事实表的粒度要粗，是在事务事实表之上建立的聚集表。比如说时间周期是1周，那么这个周期快照事实表的一条记录就是这一周的对于某个度量的统计值。周期快照事实表的维度个数比事务事实表要少。

周期快照事实表的日期维度通常是记录时间段的终止日，记录的事实是这个时间段内一些聚集事实值。事实表的数据一旦插入即不能更改，其更新方式为增量更新。

##### 累积快照事实表

Accumulatingsnapshot fact table，累积快照事实表和周期快照事实表有些相似之处，它们存储的都是事务数据的快照信息。但是它们之间也有着很大的不同，周期快照事实表记录的确定的周期的数据，而累积快照事实表记录的不确定的周期的数据。

累积快照事实表代表的是完全覆盖一个事务或产品的生命周期的时间跨度，它通常具有多个日期字段，用来记录整个生命周期中的关键时间点。例如订单累计快照事实表会有付款日期，发货日期，收货日期等时间点。

事务事实表中一个完整的交易记录会有一系列不同状态的数据来记录整个交易过程；而累积快照事实表只会有一条记录，数据会一直更新直到过程结束。

累积快照事实表代表的是完全覆盖一个事务或产品的生命周期的时间跨度，它通常具有多个日期字段，用来记录整个生命周期中的关键时间点。另外，它还会有一个用于指示最后更新日期的附加日期字段。

由于事实表中许多日期在首次加载时是不知道的，所以必须使用代理关键字来处理未定义的日期，而且这类事实表在数据加载完后，是可以对它进行更新的，来补充随后知道的日期信息。

举例来说：订货日期、预定交货日期、实际发货日期、实际交货日期、数量、金额、运费。

##### 总结

|  |  |  |  |
| --- | --- | --- | --- |
| 特点 | 事务事实 | 周期快照事实 | 累积快照事实 |
| 时间/时期 | 时间 | 时期 | 时间跨度较短的多个时点 |
| 粒度 | 每行代表一个事务事件 | 每行代表一个时间周期 | 每行代表一个业务周期 |
| 事实表加载 | 新增 | 新增 | 新增和修改 |
| 事实表更新 | 不更新 | 不更新 | 新事件产生时更新 |
| 时间维 | 业务日期 | 时期末 | 多个业务过程的完成日期 |
| 事实 | 事务活动 | 时间周期内的绩效 | 限定的多个业务阶段内的绩效 |

### 维度表

#### 概述

维度是指观察数据的角度，一般是一个名词，比如对于销售金额这个事实，我们可以从销售时间、销售产品、销售店铺、购买顾客等多个维度来观察分析。

维度表的记录数比事实表少，但是每条记录可能会包含很多字段。

#### 分类

主要包含两大类数据：

1. 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。
2. 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、地理维表等。数据量可能是个位数或者几千条几万条。

基数指的是一个字段中不同值的个数，比如主键列具有唯一值，所以具有最高的基数，而性别枚举值（日期、地区等）这样的列的基数就很低。

***栗子：***

时间维度表

描述事件发生的时间，数据仓库就是一个随时间变化的数据集合，因此可能需要一个时间维度表。年月日时分秒。

地理维度表

描述地理位置信息数据，国家、省市县镇村、邮编等。

产品维度表

描述产品属性。比如书的分类，有科技、教育、小说等分类属性。

人员维度表

描述人员相关信息，销售人员、市场人员、开发人员等。

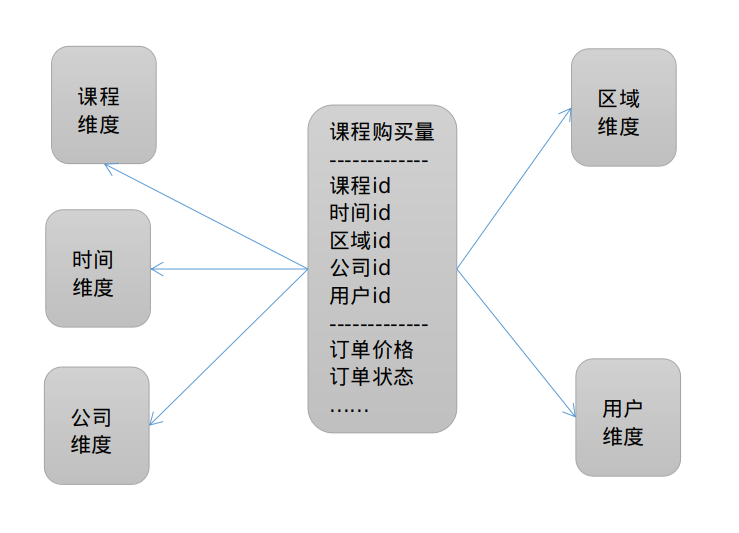
### 常见的建模方法

#### 星型模型

是一种多维的数据关系。一个事实表为中心，多个维度表环绕周围。

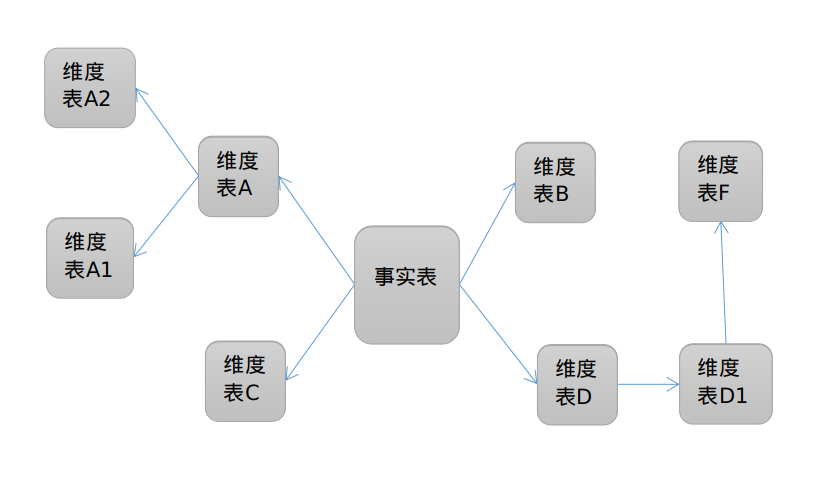
一个星型模型中可以有一个或多个事实表，每个事实表可以引用任意数量的维度表。

星型模型将业务流程分为事实和维度。事实是对业务的度量，是定量的数据，比如价格、销售数量、距离、速度、质量等。维度是对事实数据属性的描述，比如日期、产品、客户、地理位置等。



#### 雪花模型

当有一个或多个维表没有直接连接到事实表上，而是通过其他维表连接到事实表上时，就像多个雪花连接在一起，故称雪花模型。雪花模型是对星型模型的扩展，它对星型模型的维表进一步层次化，原有的各维表可能被扩展为小的事实表，形成一些局部的 "层次 " 区域，这些被分解的表都连接到主维度表而不是事实表。



**如何将维度表进行层次化处理呢？**

即把低基数（重复比较多、辨识度比较低、维度数据少，比如性别）的属性从维度表中移除并形成单独的表。

比如之前讲到的案例，购买量指标有课程维度，课程维度又可以将课程分类进行层次化扩展为新的维度表。

**层次化的影响**

层次化的过程是将维度表中重复度比较高的字段组成一个新表，所以层次化不可避免增加了表的数量，减少了数据的存储空间，提高了数据更新的效率。但是查询时就需要连接更多的表。

总结，雪花模型中，一个维度被规范化成多个关联的表，星型模型中，每个维度由一个单一的维度表所表示。

### 渐变维（SCD）

#### 什么是渐变维

维度可以根据变化剧烈程度主要分为无变化维度和变化维度。例如一个人的相关信息，身份证号、姓名和性别等信息数据属于不变的部分；而婚姻状态、工作经历、工作单位和培训经历等属于可能会变化的字段。

大多数维度数据随时间的迁移是缓慢变化的。比如增加了新的产品，或者产品的ID号码修改了，或者产品增加了一个新的属性，此时，维度表就会被修改或者增加新的记录行。这样，在设计维度和使用维度的过程中，就要考虑到缓慢变化维度数据的处理。

缓慢渐变维，即维度中的属性可能会随着时间发生改变，比如包含用户住址Address的DimCustomer维度，用户的住址可能会发生改变，进而影响业务统计精度，DimCustomer维度就是缓慢渐变维（SCD）。

SCD有三种分类，我们这里以顾客表为例来进行说明：

假设在第一次从业务数据库中加载了一批数据到数据仓库中，当时业务数据库有这样的一条顾客的信息。

IMG_256

顾客 BIWORK ，居住在北京，目前是一名 BI 的开发工程师。假设 BIWORK 因为北京空气质量 PM2.5 等原因从北京搬到了三亚。那么这条信息在业务数据库中应该被更新了。

IMG_256

那么当下次从业务数据库中抽取这类信息的时候，数据仓库又应该如何处理呢？

我们假设在数据仓库中实现了与业务数据库之间的同步，数据仓库中也直接将词条数据修改更新。后来我们创建报表做一些简单的数据统计分析，这时在数据仓库中所有对顾客 BIWORK 的销售都指向了 BIWORK 新的所在地 - 城市三亚，但是实际上 BIWORK 在之前所有的购买都发生在 BIWORK 居住在北京的时候。

通过这个简单的例子，描述了因一些基本信息的更改可能会引起数据归纳和分析出现的问题。

#### SCD1（缓慢渐变类型1）

通过更新维度记录直接覆盖已存在的值。不维护记录的历史。一般用于修改错误的数据，即历史数据就是错误数据，除此没有他用。

在数据仓库中，我们可以保持业务数据和数据仓库中的数据始终处于一致。可以在 Customer 维度中使用来自业务数据库中的 Business Key - CustomerID 来追踪业务数据的变化，一旦发生变化那么就将旧的业务数据覆盖重写。

DW 中的记录根据业务数据库中的 CustomerID 获取了最新的 City 信息，直接更新到 DW 中。

IMG_256

#### SCD2（缓慢渐变类型2）

在源数据发生变化时，给维度记录建立一个新的“版本”记录，从而维护维度历史。SCD2不删除、不修改已存在的数据。SCD2也叫拉链表。

在数据仓库中有很多需求场景会对历史数据进行汇总和分析，因此会尽可能的维护来自业务系统中的历史数据，使系统能够真正捕获到这种历史数据的变化。

以上面的例子来说，可能需要分析的结果是 BIWORK 在 2012年的时候购买额度整体平稳，但是从2013年开始购买额度减少了。出现的原因可能与所在的城市有关系，在北京的门店可能比在三亚的门店相对要多一些。

像这种情况，就不能很简单在数据仓库中将 BIWORK 当前所在城市直接更新，否则此用户所有的购买额度都会归于三亚。

通过起始时间来标识，Valid To（封链时间）为 NULL 的标识当前数据，也可以用2999，3000，9999等等比较大的年份。数仓内部需要保持统一。每个版本都会产生一行新的数据。

IMG_256

#### SCD3（缓慢渐变类型3）

实际上SCD1 and 2 可以满足大多数需求了，但是仍然有其它的解决方案，比如说 SCD3。 SCD3希望只维护更少的历史记录。

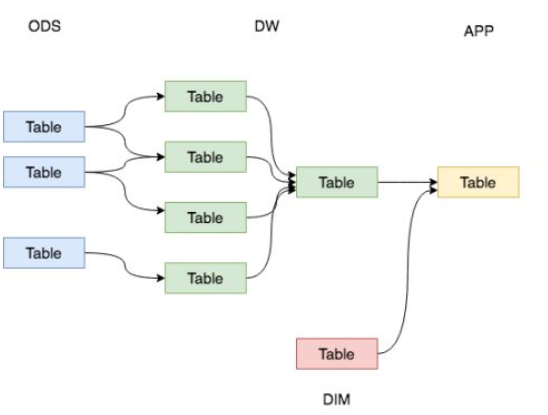
比如说把要维护的历史字段新增一列，然后每次只更新 Current Column 和 Previous Column。这样，只保存了最近两次的历史记录，历史数据都在同一行数据中。但是如果要维护的字段比较多，就比较麻烦，因为要更多的 Current 和 Previous 字段。所以 SCD3 用的还是没有 SCD1 和 SCD2 那么普遍。它只适用于数据的存储空间不足并且用户接受有限历史数据的情况。

IMG_256

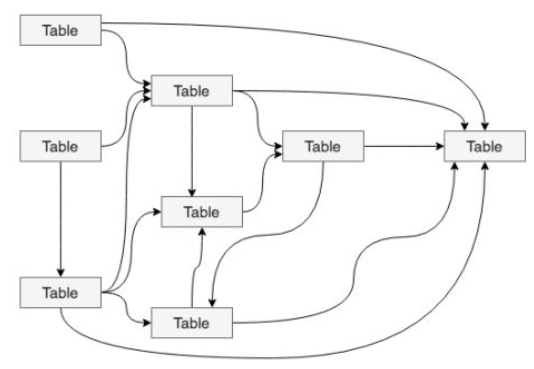
## 数据仓库分层

### 为什么要分层？

作为一名数据的规划者，我们肯定希望自己的数据能够有秩序地流转，数据的整个生命周期能够清晰明确被设计者和使用者感知到。直观来讲就是如图这般层次清晰、依赖关系直观。



但是，大多数情况下，我们完成的数据体系却是依赖复杂、层级混乱的。如下的右图，在不知不觉的情况下，我们可能会做出一套表依赖结构混乱，甚至出现循环依赖的数据体系。



因此，我们需要一套行之有效的数据组织和管理方法来让我们的数据体系更有序，这就是谈到的数据分层。数据分层并不能解决所有的数据问题，但是，数据分层却可以给我们带来如下的好处：

1. 清晰数据结构：每一个数据分层都有它的作用域和职责，在使用表的时候能更方便地定位和理解。
2. 复杂问题简单化：将一个复杂的任务分解成多个步骤来完成，每一层解决特定的问题。
3. 便于维护：当数据出现问题之后，可以不用修复所有的数据，只需要从有问题的步骤开始修复。
4. 减少重复开发：规范数据分层，开发一些通用的中间层数据，能够减少重复开发的工作量。
5. 高性能：数据仓库的构建将大大缩短获取信息的时间，数据仓库作为数据的集合，所有的信息都可以从数据仓库直接获取，尤其对于海量数据的关联查询和复杂查询，所以数据仓库分层有利于实现复杂的统计需求，提高数据统计的效率。

通常将数据模型分为三层：数据运营层（ ODS ）、数据仓库层（DW）和数据应用层（APP）。简单来讲，我们可以理解为：ODS层存放的是接入的原始数据，DW层是存放我们要重点设计的数据仓库中间层数据，APP是面向业务定制的应用数据。下面详细介绍这三层的设计。

### 分层方法

#### 源数据层（ODS）

此层数据无任何更改，直接沿用外围系统数据结构和数据，不对外开放；为临时存储层，是接口数据的临时存储区域，为后一步的数据处理做准备。

#### 数据仓库层（DW）

DW 层的数据应该是一致的、准确的、干净的数据，即对源系统数据进行了清洗（去除了杂质）后的数据。

此层可以细分为三层:

明细层DWD（Data Warehouse Detail）：存储明细数据，此数据是最细粒度的事实数据。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。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。

中间层DWM（Data WareHouse Middle）：存储中间数据，为数据统计需要创建的中间表数据，此数据一般是对多个维度的聚合数据，此层数据通常来源于DWD层的数据。

业务层DWS（Data WareHouse Service）：存储宽表数据，此层数据是针对某个业务领域的聚合数据，应用层的数据通常来源与此层，为什么叫宽表，主要是为了应用层的需要在这一层将业务相关的所有数据统一汇集起来进行存储，方便业务层获取。此层数据通常来源与DWD和DWM层的数据。

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，将所有的数据在放在DWS亦可。

#### 数据应用层（DA 或 APP）

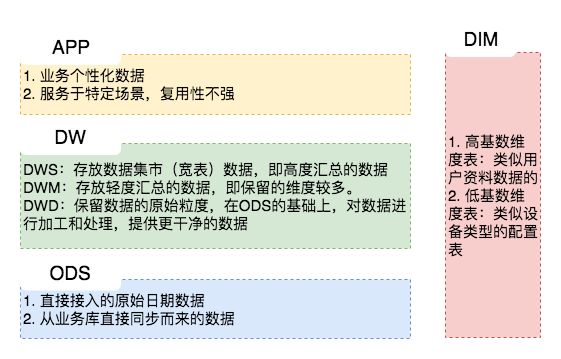
前端应用直接读取的数据源；根据报表、专题分析的需求而计算生成的数据。

#### 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：

1.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。

2.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。



## 数据仓库设计案例

这里我们以电商网站的数据仓库为例，针对用户访问日志这一部分数据进行举例说明。

在ODS层中，由于各端的开发团队不同或者各种其它问题，用户的访问日志被分成了好几张表上报到了我们的ODS层。

为了方便大家的使用，我们在DWD层做了一张用户访问行为天表，在这里，我们将PC网页、H5、小程序和原生APP访问日志汇聚到一张表里面，统一字段名，提升数据质量，这样就有了一张可供大家方便使用的明细表了。

在DWM层，我们会从DWD层中选取业务关注的核心维度来做聚合操作，比如只保留人、商品、设备和页面区域维度。类似的，我们这样做了很多个DWM的中间表。

然后在DWS层，我们将一个人在整个网站中的行为数据放到一张表中，这就是我们的宽表了，有了这张表，就可以快速满足大部分的通用型业务需求了。

最后，在APP应用层，根据需求从DWS层的一张或者多张表取出数据拼接成一张应用表即可。

