数据仓库与数据质量

第一章 概述

数据仓库

数据仓库（DataWarehouse），一般缩写成DW、DWH。数据仓库是一个面向主题的（Subject Oriented）、集成的（Integrate）、相对稳定的（Non-Volatile）、反映历史变化（Time Variant）的数据集合，用于支持管理决策

--
Bill Inmon

数据库 VS 数据仓库

1.数据偏重数据的业务处理（transaction），数据仓库着重于分析，可能会重点面向某个行业；所以数据库一般和OLTP（Online transaction processing）相联系，数据仓库一般和OLAP（Online analytical processing）相联系。

2.所以数据仓库又被称为“分析型数据库”（相对于“业务型数据库“而言）。它的数据结构有利于查询和分析的便利。数据库常采用行式存储，而数据仓库常采用列式存储。

3.数据库的用户数量大（主要是业务人员），既要执行“读”操作也要执行“写”操作，每次写的量不大，但是对时间不敏感。/ 数据仓库的用户数量小（主要是决策人员），一般只需要执行读操作，每次读取的数据量很大，对反应时间不那么敏感。

4.把所需要的数据从业务型数据库导入分析型数据仓库的过程，称为ETL（Extract-Transform-Load，“抽取-转换-加载”）。
数据仓库对分布在企业中的多个异构数据源集成，按照决策主题选择数据并以新的数据模型存储，一般不能修改。
数据仓库的目的是通过分析企业过去一段时间业务的经营状况，挖掘其中隐藏的模式。
数据仓库主要包括数据的提取、转换与装载（ETL ）、元数据、数据集市和操作数据存储等部分。

5.数据库用到的工具主要有MySQL，Oracle，MS SQLServer等，数据仓库用到的工具主要有Hive，GreenPlum等。

联机分析处理 VS 联机事务处理

联机事务处理【OLTP Online Transaction Processing】
　　联机事务处理，表示事务性非常高的系统，一般都是高可用的在线系统，以小的事务以及小的查询为主，以传统的关系型数据库为主要应用，主要是基本的、日常的事务处理，主要为业务数据，例如银行交易

　　OLTP系统最容易出现瓶颈的地方就是CPU与磁盘子系统。 OLTP比较常用的设计与优化方式为Cache技术与B-tree索引技术.OLTP 系统是一个数据块变化非常频繁，SQL 语句提交非常频繁的系统.

联机分析处理【OLAP Online Analytical Processing】：
　　联机分析处理，有的时候也叫DSS决策支持系统，就是我们说的数据仓库，重点主要是面向分析，会产生大量的查询，一般很少涉及增删改。在这样的系统中，语句的执行量不是考核标准，因为一条语句的执行时间可能会非常长，读取的数据也非常多。所以，在这样的系统中，考核的标准往往是磁盘子系统的吞吐量（带宽），如能达到多少MB/s的流量。

　　在OLAP系统中，常使用分区技术、并行技术。 分区技术在OLAP系统中的重要性主要体现在数据库管理上，分区技术对性能上的影响，它可以使得一些大表的扫描变得很快（只扫描单个分区）。另外，如果分区结合并行的话，也可以使得整个表的扫描会变得很快。总之，分区主要的功能是管理上的方便性，它并不能绝对保证查询性能的提高，有时候分区会带来性能上的提高，有时候会降低

总结：
* 联机分析处理（OLAP），数据量大，DML少。使用数据仓库模板

* 联机事务处理（OLTP），数据量少，DML频繁，并行事务处理多，但是一般都很短。使用一般用途或事务处理模板

	OLAP	**OLTP**
服务用户	决策者、高级管理者	操作者、底层管理者
功能	分析、决策	日常操作
DB设计	面向主题	面向应用
数据	历史的、聚合的、多维度集成	即时的，明细的
DB大小	GB-TB-PB	MB-GB
时间要求	实时性要求不高，一般为离线场景为主	实时性要求高
主要应用	数据仓库	数据库

第二章 OLAP

第一节 OLAP分类

分类	**描述**
ROLAP	表示基于关系数据库的OLAP实现（Relational OLAP）
MOLAP	表示基于多维数据组织的OLAP实现（Multidimensional OLAP）
HOLAP	表示基于多维数据组织的OLAP实现（Multidimensional OLAP）

ROLAP

ROLAP表示基于关系数据库的OLAP实现（Relational OLAP）

以关系数据库为核心,以关系型结构进行多维数据的表示和存储。

ROLAP将多维数据库的多维结构划分为两类表:

一类是事实表,用来存储数据和维关键字;

另一类是维表,即对每个维至少使用一个表来存放维的层次、成员类别等维的描述信息。

维表和事实表通过主关键字和外关键字联系在一起,形成了"星型模式"。

对于层次复杂的维,为避免冗余数据占用过大的存储空间,可以使用多个表来描述,这种星型模式的扩展称为"雪花模式"。特点是将细节数据保留在关系型数据库的事实表中，聚合后的数据也保存在关系型的数据库中。这种方式查询效率最低，不推荐使用

MOLAP

MOLAP表示基于多维数据组织的OLAP实现（Multidimensional OLAP）。

以多维数据组织方式为核心,也就是说,MOLAP使用多维数组存储数据。多维数据在存储中将形成"立方块（Cube）"的结构.

在MOLAP中对"立方块"的"旋转"、"切块"、"切片"是产生多维数据报表的主要技术。特点是将细节数据和聚合后的数据均保存在cube中，所以以空间换效率，查询时效率高，但生成cube时需要大量的时间和空间

HOLAP

HOLAP表示基于混合数据组织的OLAP实现（Hybrid OLAP）。如低层是关系型的，高层是多维矩阵型的。这种方式具有更好的灵活性。特点是将细节数据保留在关系型数据库的事实表中，但是聚合后的数据保存在cube中,聚合时需要比ROLAP更多的时间,查询效率比ROLAP高，但低于MOLAP

第二节 维度与度量

概述

维度和度量是数据分析中的两个基本概念。

* 维度是指审视数据的角度，它通常是数据记录的一个属性，例如时间、地点等。

* 度量是基于数据所计算出来的考量值；它通常是一个数值，如总销售额、不同的用户数等。 分析人员往往要结合若干个维度来审查度量值，以便在其中找到变化规律

例如 统计公司员工，各个部门不同年龄段、性别的人数

```
select
    deptment,
    age_range,
    gender,
    count(distinct dep_num) as total_count
from ods_hr.employee
group by
    deptment,
    age_range,
    gender
```

维度

1 维度是度量的环境，用来反映业务的一类属性，这类属性的集合构成一个维度，也可以称为实体对象。
2 维度属于一个数据域，如地理维度(其中包括国家、地区、 省以及城市等级别的内容)、时间维度(其中包括年、季、月、周、日等级别的内容),维度所包含的表示维度的列信息为维度属性，维度属性常用来进行数据过滤、数据分类、维度描述信息（报表中title中的文字描述）
3 维度是指可指定不同值的对象的描述性属性或特征，一般是一种离散数据。比如时间维度上的每一个独立的日期， 或者商品维度上的每一件独立的商品。 因此统计时可以把维度值相同的记录聚合在一起， 然后应用聚合函数做累加、 平均、 去重复计数等聚合计算.
例如
城市名称：北京、上海、广州
人名：张三、李四
班级：1班、2班
性别：男、女

维度基数

维度的基数（Cardinality）指的是这个维度在数据集中出现不同值得个数。 比如上表中city这个维度，有湖北、广东、湖南、北京等34个值，则该维度的基数就是34

超高基数列

超高基数列是指基数超过一百万的维度，这种维度的设计需要格外谨慎。常见的比如 “userid”、“timestamp”、“production_id”等等。维度的基数可以通过hive 的count distinct函数来进行查询获得

指标

指标是指可以按总数或比值衡量的具体维度元素。例如，维度“城市”可以关联指标“人口”，其值为具体城市的居民总数

维度与指标

虽然维度和指标可以独立使用，但常见的还是相互结合使用。维度和指标的值以及这些值之间的关系，使您的数据具有了意义。为了挖掘尽可能多的深层次信息，维度通常与一个或多个指标关联在一起。
例如，维度“城市”可以与指标“人口”和“面积”相关联。有了这些数据，系统还可以创建“人口密度”等比值指标，带来有关这些城市的更详细的深入信息

度量

度量就是被聚合的统计值，也是聚合运算的结果，一般是连续的值

度量和维度构成OLAP的主要概念，这里面对于在事实表或者一个多维立方体里面存放的数值型的、连续的字段，就是度量。

度量主要用于分析或者评估，比如对趋势的判断，对业绩或者效果的判定等等。比如在一般的大数据分析应用里面就有总PV，总UV等度量用于评判一个网站或者APP的活跃度

指标与度量

表述为"它是表示某种相对程度的值"。区别于上面的度量概念，那是一种绝对值，尺子量出来的结果，汇总出来的数量等。而指标至少需要两个度量之间的计算才能得到，例如收入增长率，用本月收入比上上月收入。当然可能指标的计算还需要两个以上的度量。

维度变化

在现实世界中，维度的属性不是静态不变的，而是会随着时间的流逝可能发生缓慢变化
例如
变化前：

商品ID	商品描述	商品所属分类	其他
1000	XX电子产品	A	...

变化后：

商品ID	商品描述	商品所属分类	其他
1000	XX电子产品	AA	...

本质上讲经过数据变化导致了可能无法进行历史数据回溯等相关计算
解决方案：
1 重写变化维度值
就是覆盖方式，即不对历史数据进行保留操作，如上例子中的【商品所属分类】更新为AA后，更新前的所有数据如果要进行相关计算存在问题

2 追加维度记录方式
对变化的维度信息进行追加维度记录的方式保存，这样同时也保留了历史数据，变化前后的主题数据分别和其对应的维度信息关联并进行相关计算即可

3 追加维度列方式
对变化的维度信息进行追加维度列的方式保存，这样同时也保留了历史维度列数据，所有主题数据可根据需要关联维度列并进行相关计算即可

第三节 主题域、主题、实体

主题

主题是与传统数据库的面向应用相对应的，是一个抽象概念，是在较高层次上将企业信息系统中的数据综合、归类并进行分析利用的抽象。

每一个主题对应一个宏观的分析领域。在逻辑意义上，它是对应企业中某一宏观分析领域所涉及的分析对象。面向主题的数据组织方式，就是在较高层次上对分析对象数据的一个完整并且一致的描述，能刻画各个分析对象所涉及的企业各项数据，以及数据之间的联系。所谓较高层次是相对面向应用的数据组织方式而言的，是指按照主题进行数据组织的方式具有更高的数据抽象级别。

与传统数据库面向应用进行数据组织的特点相对应，数据仓库中的数据是面向主题进行组织的。主题是根据分析的要求来确定的。这与按照数据处理或应用的要求来组织数据是不同的。

例如，一个生产企业的数据仓库所组织的主题可能有产品订货分析和货物发货发运分析等。而按应用来组织则可能为财务子系统，销售子系统，供应子系统，人力资源子系统和生产调度子系统。

主题是根据分析的要求来确定的。如在生产企业中，同样是材料供应，在操作型数据库系统中，人们所关心的是怎样更方便和更快捷地进行材料供应的业务处理；而在进行分析处理时，人们就更关心材料的不同采购渠道和材料供应是否及时/材料质量情况等。

举例：我们可能有供应商主题、商品主题、客户主题和仓库主题。其中商品主题的内容包括记录商品的采购、商品的销售和存储；客户主题包括客户的购买商品情况；仓库主题包括仓库中商品的存储情况和仓库的管理情况。

主题域

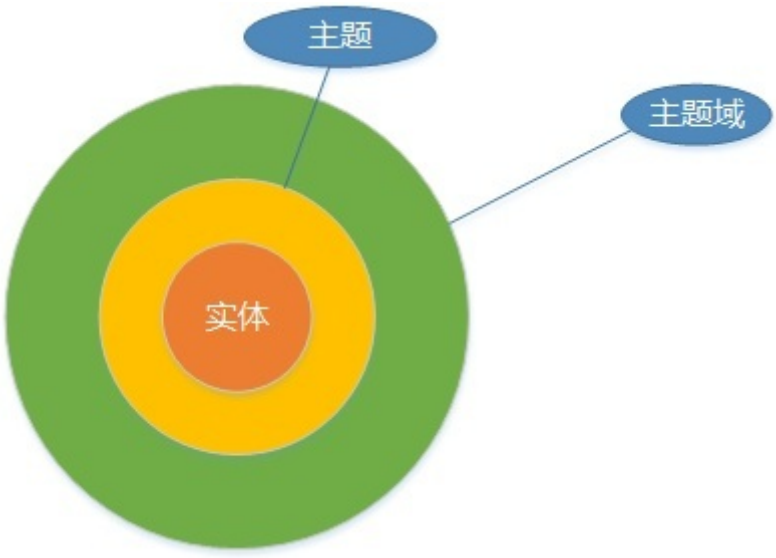
主题域通常是联系较为紧密的数据主题的集合。可以根据业务的关注点，将这些数据主题划分到不同的主题域。主题域的确定必须由最终用户和数据仓库的设计人员共同完成

主题域、主题、实体间关系

主题设计是对主题域进一步分解，细化的过程。主题域下面可以有多个主题，主题还可以划分成更多的子主题，而实体则是不可划分的最小单位。主题域、主题、实体的关系如下图所示

示例：

用户主题、商品主题、货流主题、仓库管理主题



第四节 数据模型

3NF

第一范式（1NF）：原子性
（1NF是对属性的原子性约束，要求属性具有原子性，不可再分解）

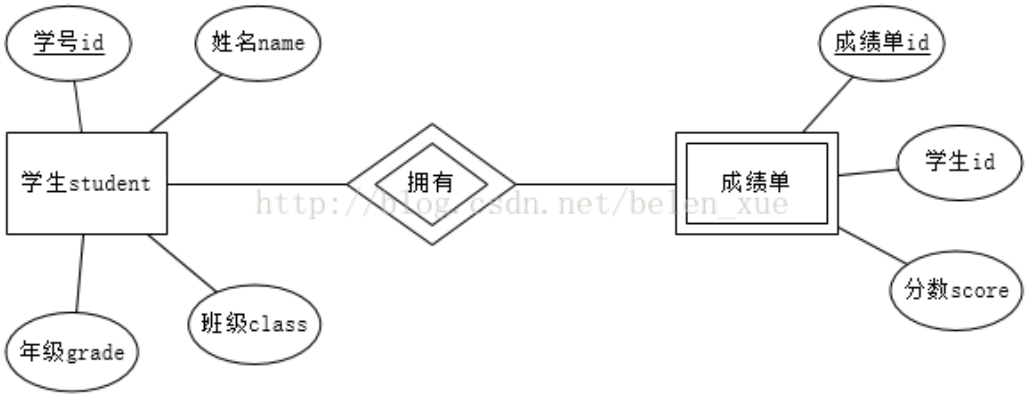
第二范式（2NF）：符合1NF，惟一性约束。
（2NF是对记录的惟一性约束，要求记录有惟一标识，即实体的惟一性，更通俗说有主键ID）

第三范式（3NF）：符合2NF，消除字段冗余。
（3NF是对字段冗余性的约束，即任何字段不能由其他字段派生出来，它要求字段没有冗余）

数据模型种类

ER模型

概念模型是对信息世界的建模，所以概念模型应该能够方便、准确地表示信息世界中的常用概念。
通常用E-R模型（Entity-Relationship，实体 - 关系）来描述现实世界的概念模型，在范式理论上符合3NF，而关系型数据库也即是二维表。
数据仓库的3NF和OLTP关系中的3NF区别在于，数据仓库站在企业角度面向主题的抽象，而非针对具体的业务流程的实体对象关系的抽象

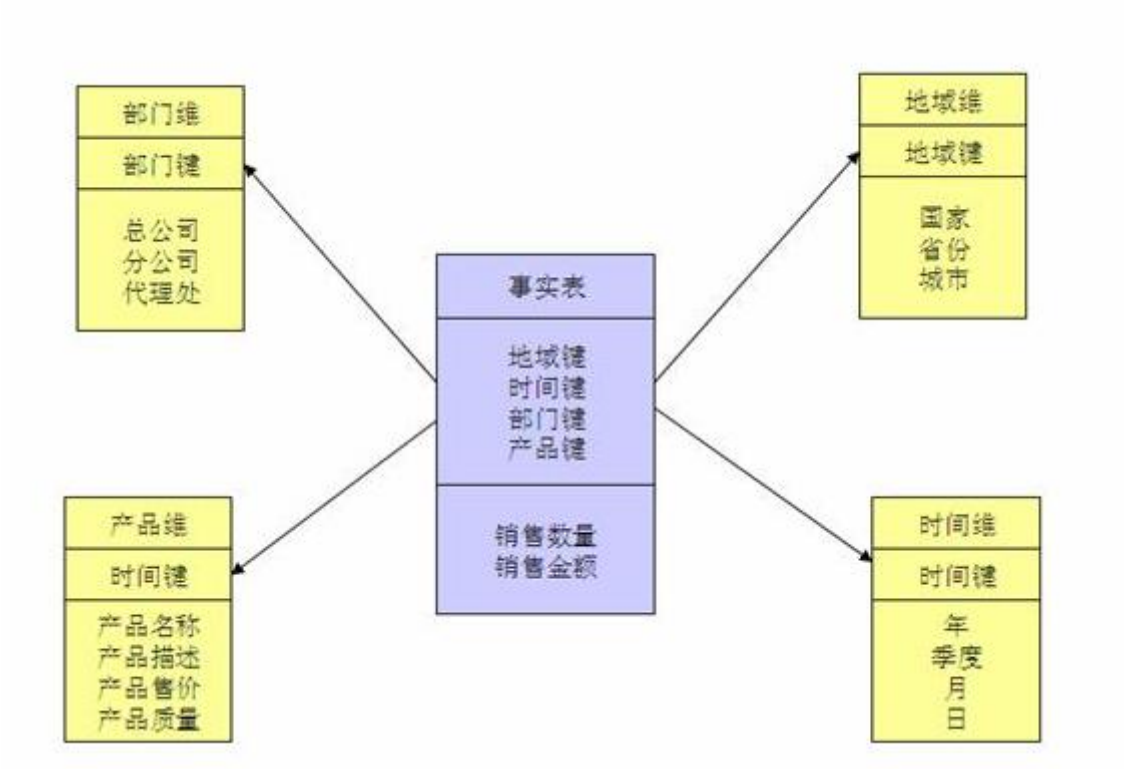


维度模型

维度模型从分析决策的需求出发构建模型，关注重点为如何快速地完成需求分析，同时具有较好的大规模复杂查询的响应性能，典型的如星型模型，另外在一些特殊场景下使用的雪花模型。

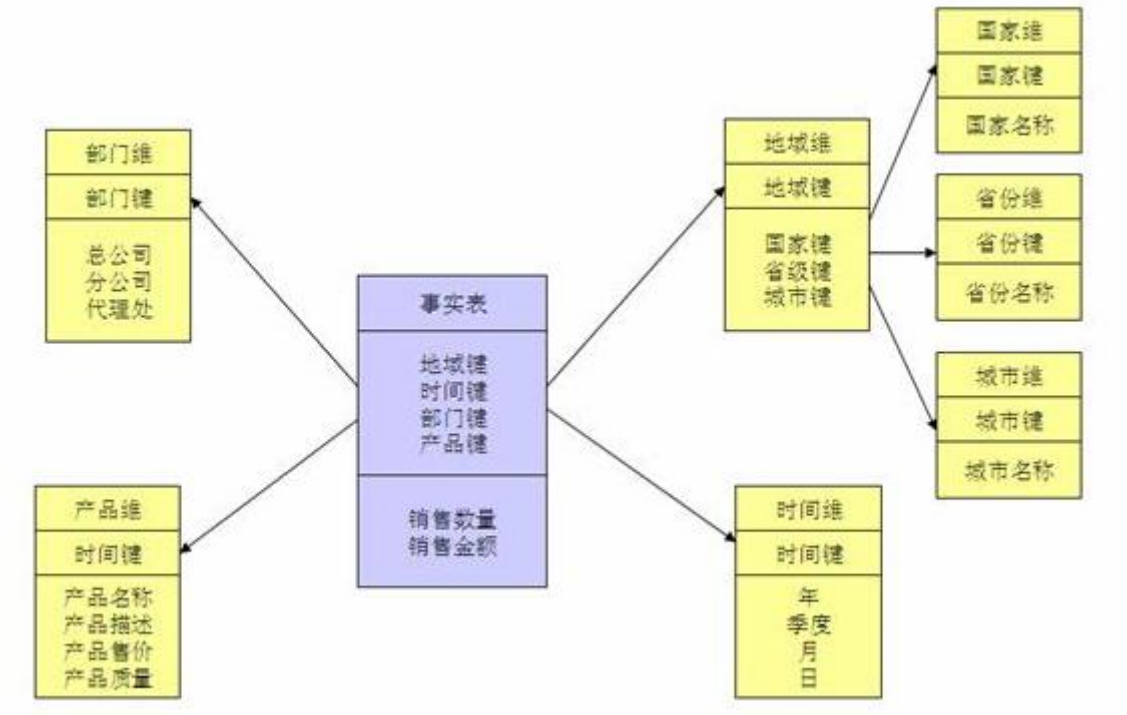
星型模型

星型模是一种多维的数据关系，它由一个事实表和一组维表组成。每个维表都有一个维作为主键，所有这些维的主键组合成事实表的主键。强调的是对维度进行预处理，将多个维度集合到一个事实表，形成一个宽表。这也是我们在使用hive时，经常会看到一些大宽表的原因，大宽表一般都是事实表，包含了维度关联的主键和一些度量信息，而维度表则是事实表里面维度的具体信息，使用时候一般通过join来组合数据，相对来说对OLAP的分析比较方便。



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，其图解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。如图 2，将地域维表又分解为国家，省份，城市等维表。它的优点是 ： 通过最大限度地减少数据存储量以及联合较小的维表来改善查询性能。雪花型结构去除了数据冗余。



Killball建模

1 收集业务需求

在建模工作前，项目组需要跟使用数据的业务人员进行沟通调研，理解业务过程的KPI，数据分析的目标，利用数据进行哪些决策制定。可以收集业务人员，高层决策者经常看的报表，了解他们观察数据的维度跟指标。

2 协作维度建模研讨

维度建模不应该只由那些不懂业务以及业务需求的技术人员来负责，还需要企业数据管理者与使用者的参与共同制定数据分析主题。与业务代表开展一系列高级别交流讨论可以帮助技术人员和需求分析人员对数据有更深入的了解，寻找不同部门使用数据的分析维度与指标的异同。

3 维度建模的设计过程

维度模型设计主要设计四个步骤：

(1) 选择业务过程

业务过程是组织完成的操作型活动，例如：获得订单，处理保险索赔、学生课程注册或每个月每个账单的快照等等。过程的选择很重要，因为我们要从业务过程中得出事实的指标度量，以及事实表的粒度选取，维度划分等等。

(2) 声明粒度

声明粒度是维度设计的重要步骤。粒度用于确定某一事实表中的行表示什么。选择维度或事实前必须声明粒度，因为每个候选维度或事实必须与定义的粒度保持一致。在所有维度设计中强制实行一致性是保证BI应用性能和易用性能的关键。在从给定的业务过程获取数据时，原子粒度是最低级别的粒度。粒度越小，描述的业务过程越详细。建议从设计最小粒度的数据开始，这样可以保证比较大的灵活性，满足无法预期的业务用户的查询需求。这对不同的事实表粒度要建立不同的物理表，在同一事实表中不要混用多种不同的粒度。

(3) 确认维度

维度表又是被称为数据仓库的“灵魂”，因为维度包含确保DW/BI系统能够被用作业务分析的入口和描述性标识。维度提供围绕某一业务过程事件所涉及的“谁、什么、何处、何时、为什么、如何”等背景。维度表包含BI应用所需要的用于过滤及分类事实的描述性属性。牢牢掌握事实表的粒度，就能够将所有可能存在的维度区分开。当与给定事实表行关联是，任何情况下都应使维度保持但一值。

(4) 确认事实

事实基本上都是以数量值表示，涉及来自业务过程事件的度量，例如销售量，销售额等。一个事实表行与按照事实表粒度描述的度量事件之间存在一对一关系，因此事实表对应一个物理可观察的事件。

星型模型与OLAP多维数据库

星型模式是部署在关系数据库管理系统(RDBMS)之上的多维结构。典型地，主要包含事实表，以及通过主键/外键关系与之关联的维度表。联机分析处理(OLAP)多维数据库是实现在多维数据库之上的数据结构，它来源于关系型星型模式。OLAP多维数据库包含维度属性和事实表，但它能够比SQL语言具有更强的分析能力和访问，比如XMLA和MDX等。

事实表

事务事实表（Transaction fact table)

事务事实表用来描述业务过程，记录的事务层面的事实，保存的是最原子的数据，也称为“原子事实表”.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。

通过事务事实表，还可以建立聚集事实表，为用户提供高性能的分析。

示例：
销售业务中的订单销售表、商品货物库存表

周期快照事实表(Periodic snapshot fact)

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。

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。周期快照事实表的维度个数比事务事实表要少，但是记录的事实要比事务事实表多。

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

累计快照事实表（Accumulating snapshot fact table)

累积快照事实表用于定义业务过程开始、结束以及期间的可区分的里程碑事件。通常在此类事实表中针对过程中的关键步骤都包含日期外键，并包含每个步骤的度量，这些度量的产生一般都会滞后于数据行的创建时间。累积快照事实表中的一行，对应某一具体业务的多个状态。例如，当订单产生时会插入一行。当该订单的状态改变时，累积事实表行被访问并修改。这种对累积快照事实表行的一致性修改在三种类型的事实表（事务、周期快照、累积快照）中具有独特性，对于前面两类事实表只追加数据，不会对已经存在的行进行更新操作。除了日期外键与每个关键过程步骤关联外，累积快照事实表中还可以包含其它维度和可选退化维度的外键。

示例1：
假设希望跟踪以下五个销售订单的里程碑：下订单、分配库房、打包、配送和收货，分别用状态N、A、P、S、R表示。这五个里程碑的日期及其各自的数量来自源数据库的销售订单表。一个订单完整的生命周期由五行数据描述：下订单时生成一条销售订单记录；订单商品被分配到相应库房时，新增一条记录，存储分配时间和分配数量；产品打包时新增一条记录，存储打包时间和数量；类似的，订单配送和订单客户收货时也都分别新增一条记录，保存各自的时间戳与数量。为了简化示例，不考虑每种状态出现多条记录的情况（例如，一条订单中的产品可能是在不同时间点分多次出库），并且假设这五个里程碑是以严格的时间顺序正向进行的

示例2：
累积快照事实表在库存、采购、销售、电商等业务领域都有广泛应用。比如在电商订单里面，下单的时候只有下单时间，但是在支付的时候，又会有支付时间，同理，还有发货时间，完成时间等。下面以销售订单数据仓库为例，讨论累积快照事实表的实现

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

在这个累积快照事实表中，记录的是购买货物的整个生命周期的数据，记录第一次产生时，实际发货日期和实际交货日期是不确定的，需要用表示未知的代理关键字来代替。等实际发货后，需要对数据仓库中的这条记录进行更新操作，将实际发货日期补上

特点	事务事实	周期快照事实	累计快照事实
时间	数据产生时间	数据产生时间基础上的时间周期	多个过程环节的产生时间
粒度	每行代表一个事务事件	每行代表一个时间周期	每行代表一个业务过程
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
时间维度	业务日期	周期结束时间	多个业务环节完成时间
事实	事务信息	周期内的统计信息	多个业务环节内分别统计信息

多维度模型

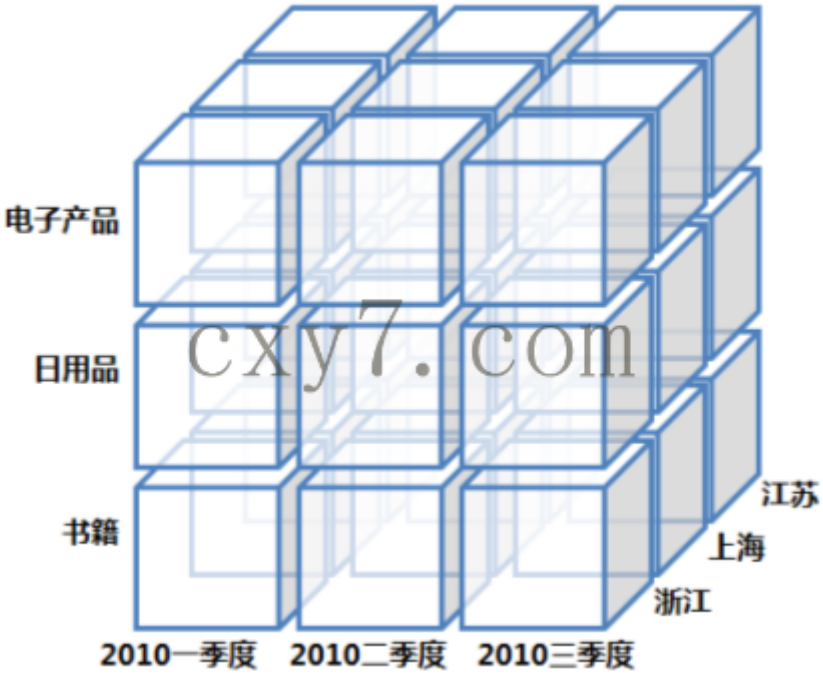
多维数据模型提供了多角度多层次的分析应用，比如基于时间维、地域维等构建的星形模型、雪花模型，可以实现在各时间维度和地域维度的交叉查询，以及基于时间维和地域维的细分。所以多维数据模型的应用一般都是基于联机分析处理（Online Analytical Process, OLAP）的，而面向特定需求群体的数据集市也会基于多维数据模型进行构建

数据立方体

概念和形式

数据立方体，是一种常用于数据分析与索引的技术；它可以对原始数据建立多维度索引。通过Cube对数据进行分析，可以大大加快数据的查询效率.

数据立方体只是多维模型的一个形象的说法。立方体其本身只有三维，但多维模型不仅限于三维模型，可以组合更多的维度.一方面是出于更方便地解释和描述，同时也是给思维成像和想象的空间；另一方面是为了与传统关系型数据库的二维表区别开来



cxy7.com

基本操作

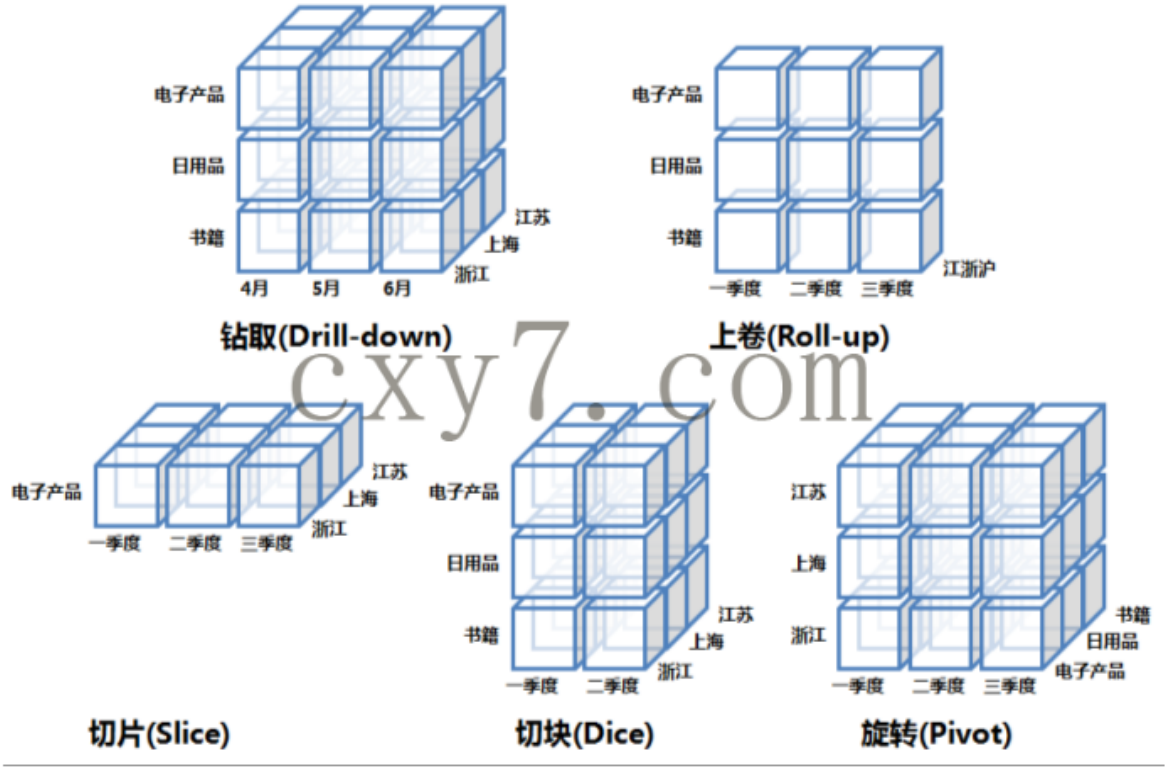
1 钻取（Drill-down）：在维的不同层次间的变化，从上层降到下一层，或者说是将汇总数据拆分到更细节的数据，比如通过对2010年第二度的总销售数据进行钻取来查看2010年第二季度4、5、6每个月的消费数据，如上图； 当然也可以钻取浙江省来查看杭州市、波市、温州市……这些城市的销售数据。

2 上卷（Roll-up）：钻取的逆操作，即从细粒度数据向高层的聚合，如将江苏省、上海市和浙江省的销售数据进行汇总来查看江浙沪地区的销售数据，如上图。

3 （Slice）：选择维中特定的值进行分析，比如只选择电子产品的销售数据，或者2010年第二季度的数据。

4 块（Dice）：选择维中特定区间的数据或者某批特定值进行分析，比如选择2010年第一季度到2010年第二季度的销售数据，或者是电子产品和日用品的销售数据。

5 旋转（Pivot）：即维的位置的互换，就像是二维表的行列转换，如图中通过旋转实现产品维和地域维的互换



cxy7.com

第三章 数据仓库

第一节 数据类型

结构化数据

结构化数据的格式非常规范，典型的代表就是关系数据库中的数据，这些数据可以用二维表来存储，有固定的字段数，每个字段有固定的数据类型（数字、字符、日期等），并且每个字段的字节长度也相对固定。
例如：
订单记录表
用户表

半结构化数据

半结构化数据的格式较为规范，一般都是纯文本数据，可以通过某种方式解析得到每项的数据。最常见的就是日志数据、XML、JSON等格式的数据，它们每条记录可能会有预定义的规范，但是可能每条记录包含的信息不尽相同，也可能会有不同的字段数，包含不同的字段名或字段类型，或者包含着嵌套的格式。这类数据一般都是以纯文本的形式输出，管理维护也较为方便，但在需要使用这些数据时，如获取、查询或分析数据时，可能需要先对这些数据格式进行相应的解析。

半结构化的数据通常是指网站的日志数据，或者因为某些需求以XML或JSON格式输出的数据。最常见的就是网站的Apache日志，它根据预定义的字段顺序打出相应的值

例如：

日志数据

```
{time: 1234567890, action: “comment”, respond: true, user: {userid: 1, username: “abc”}}
```

非结构化数据

非结构化数据指的是那些非纯文本类数据，没有标准格式，无法直接地解析出相应的值。常见的非结构化数据有富文本文档、网页、多媒体（图像、声音、视频等）。这类数据不易收集管理，也无法直接查询和分析，所以对这类数据需要使用一些不同的处理方式

第二节 数仓构建

ETL过程

数据仓库从各数据源获取数据及在数据仓库内的数据转换和流动都可以认为是ETL（抽取Extra，转化Transfer，装载Load）的过程，ETL是数据仓库的流水线

维度确认

源数据中的某些维度信息对于分析而言，没有价值或者其可能产生的价值远低于储存这些数据所需要的数据仓库的实现和性能上的成本

面向主题

面向主题是数据仓库的第一特性，主要是指合理地组织数据用以实现分析。对于源数据而言，其数据组织形式是多样的，像点击流的数据格式是未经优化的，前台数据库的数据是基于OLTP操作组织优化的，这些可能都不适合分析，而整理成面向主题的组织形式才是真正地利于分析的。

例如：

点击流日志整理成页面（Page）、访问（Visit或Session）、用户（Visitor）三个主题，这样可以明显提升分析的效率

全量表

对于业务数据、同步数据等进行全量数据保存，触发周期为每天凌晨N点开始执行，记录截止到之前一天的所有数据

增量表

一般按照日期进行时间分区，记录之前一天的变化的业务数据，如使用mysql binlog采集仅仅前一天的增量(变化)数据，往往可以通过增量数据与历史数据合并行程“快照的全量数据”

快照表

按日期分区，对所关注的数据，如事实数据、维度数据进行截止到前一天的全量数据记录
优点：
　　可以根据日期对不同业务进行回溯计算或数据变化分析
缺点：
　　占用存储空间较大

拉链表

拉链表是针对数据仓库设计中表存储数据的方式而定义的，顾名思义，所谓拉链，就是记录历史。记录一个事物从开始，一直到当前状态的所有变化的信息

使用场景

1 数据量很大的数据集，如大型电商用户表、社交APP（微信用户）
2 数据集有数据值变化，如用户手机、维度值变更、过程中的环节变化
3 需要进行历史数据快照信息，如12306用户的手机号、关系人

解决方案

1 只在数据同步时，抓取全量数据覆盖式同步，在数据量级较小时可应用，这样不能反映某些维度的变化趋势无法进行与此维度相关的数据挖掘和分析工作
2 按日期进行分区处理，每天一份全量数据，存储成本太高，数据有冗余存储
3 数据计量较大，只保存增量数据，通过数据处理形成新数据集使用(不进行物化)
4 使用拉链表

拉链表示例

原始数据

用户注册表，存储了用户编号、手机、注册时间

注册日期	用户编号	手机号码
2017-01-01	001	111111
2017-01-01	002	222222
2017-01-01	003	333333
2017-01-01	004	444444

数据变更

```
(1) 20170102这天数据变更，用户002和004资料进行了修改，005是新增用户
(2) 20170103这天数据变更，用户004和005资料进行了修改，006是新增用户
```

注册日期	用户编号	手机号码	备注
2017-01-01	001	111111	
2017-01-01	002	233333	（由222222变成233333）
2017-01-01	003	333333	
2017-01-01	004	432432	（由444444变成432432）
2017-01-02	005	555555	（2017-01-02新增）

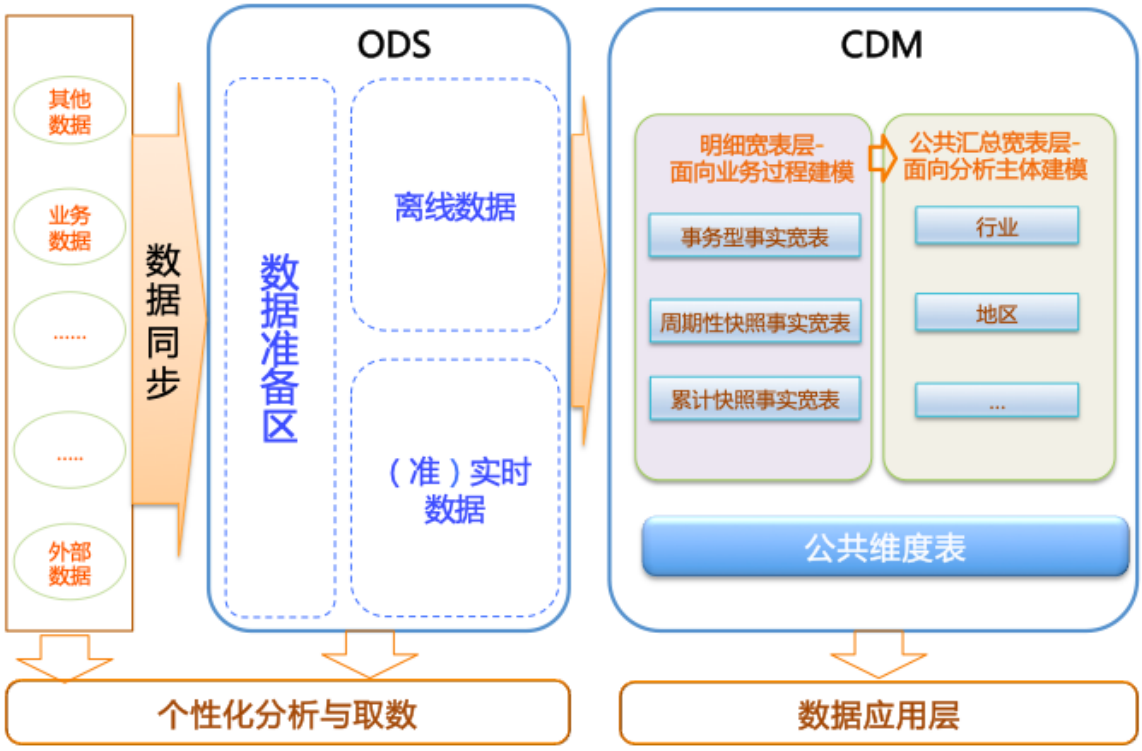
注册日期	用户编号	手机号码	备注
2017-01-01	001	111111	
2017-01-01	002	233333	
2017-01-01	003	333333	
2017-01-01	004	654321	（由432432变成654321）
2017-01-02	005	115115	（由555555变成115115）
2017-01-03	006	666666	（2017-01-03新增）

拉链数据

注册日期	用户编号	手机号码	t_start_date	t_end_date
2017-01-01	001	111111	2017-01-01	9999-12-31
2017-01-01	002	222222	2017-01-01	2017-01-01
2017-01-01	002	233333	2017-01-02	9999-12-31
2017-01-01	003	333333	2017-01-01	9999-12-31
2017-01-01	004	444444	2017-01-01	2017-01-01
2017-01-01	004	432432	2017-01-02	2017-01-02
2017-01-01	004	654321	2017-01-03	9999-12-31
2017-01-02	005	555555	2017-01-02	2017-01-02
2017-01-02	005	115115	2017-01-03	9999-12-31
2017-01-03	006	666666	2017-01-03	9999-12-31

通过日期时间区分存储了维度信息的变化，在使用时就可以通过日期时间回溯到之前历史数据时刻

第三节 模型层次



ODS

ODS：Operational Data Store，操作数据层，在结构上其与源系统的增量或者全量数据基本保持一致。它相当于DW数据的一个数据准备区，同时又承担着基础数据的记录以及历史变化

DIM

DIM:Dictionary Data Layer 也叫数据字典DICT，比如存储一些诸如时间、地区(省、市、县区域表)、渠道列表、商品类目等等表数据

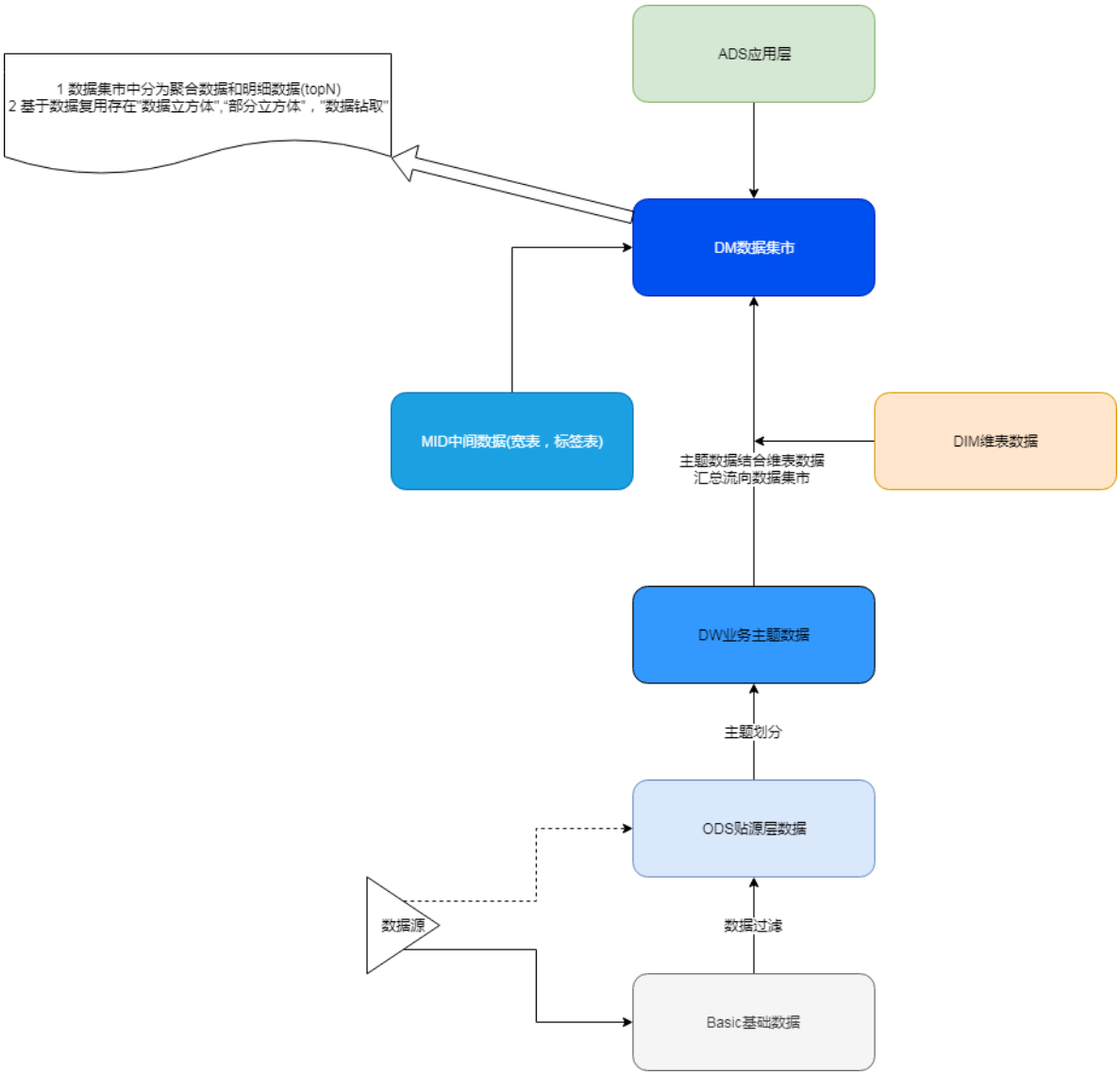
CDM（DWD+DWS)

Common Data Model，公共维度模型层，又细分为DWD和DWS。它的主要作用是完成数据加工与整合，建立一致性的维度，构建可复用的面向分析和统计的明细事实表，以及汇总公共粒度的指标

DWD：明细数据层
DWS：汇总数据层

ADS

Application Data Service，应用数据层



参考示例：

1 数据仓库 https://www.jianshu.com/p/72e395d8cb33

2 数据质量 https://dbaplus.cn/news-73-2003-1.html

3 griffin https://blog.csdn.net/ebay/article/details/52537285

4 数据质量洞察 https://insights.thoughtworks.cn/data-quality-management/

5 案例分析 https://www.oceanengine.com/case