# 大数据数据仓库

## 数据仓库基本概念

数据仓库即Data Warehouse，简称DW，主要研究和解决从数据中获取信息的问题，为企业所有级别的决策制定过程，提供所有类型数据支持的战略集合。本质上，数据仓库试图提供一种从操作型系统到决策支持系统的数据流架构模型。主要是解决多重数据复制带来的高成本问题。在有数仓之前，需要大量的冗余数据来支撑多个决策支持系统，尽管每个系统服务于不同的用户，但是这些系统经常需要大量相同的数据。决策支持系统Decision support System，即DDS，是用于支持业务或组织决策活动的信息系统，服务于组织管理、运营和规划管理层（通常是中层或高级管理层），帮助人们对可能快速变化并且不容易预测结果的问题做出决策。数据仓库就是为决策系统提供数据支持的。

### 1.1、数据仓库的概念

数据仓库之父Bill Inmon提出了被广泛认可的数据仓库定义,把数据仓库定义为是一个面向主题的、集成的、非易失的和时变的数据集合，用于支持管理者的决策过程。

### 1.2、数据仓库的特性

面向主题

传统的操作型系统是围绕功能性应用来组织数据的，各个业务系统可能是相互隔离的，而数据仓库是面向主题的。主题是一个抽象概念，简单的来说就是用户使用数据仓库进行决策时所关心的重点方面，一个主题通常与多个操作型系统相关。例如一个保险公司要分析销售数据，就可以建立一个专是一个数据主题，同时数据仓库设计时要排除对于决策无用的数据，上述场景决策关注销售，理赔不是这个主题的，因此理赔的数据就不需要在这个数据仓库中存储。注销售数据的数据仓库，通过这个数据仓库就可以得到“过去半年销售保险总额以及各险种的占比”，这个场景下销售就

集成

集成是与面向主题密切相关。我们上面的保险的例子中，建立一个保险公司的销售数据仓库，不同的险种由不同的部门负责，他们有各自独立的销售数据库，此时要想从公司层面整体分析销售数据，必须将各分散的数据源统一成一致的、无歧义的数据格式后，再存储在数据仓库中，因此就需要解决各数据源的矛盾之处，例如字段的同名异意、异名同意、字段数据类型不一致，长度不一致，计量单位不一致等等。另外数据还需要进行综合和计算等处理。当完成了这些数据的整合工作后，数据仓库就是集成的，这个过程就是数据仓库中最复杂、最关键的过程：ETL。集成也就是把多个数据源的数据进行统一存储到数据仓库中。

非易失性

非易失性指的是数据一旦进入数据仓库，数据就不应该再有改变，操作型系统中的数据一般会频繁更新，而数据仓库中的数据一般不进行更新，当操作型系统的改变的数据进入数据仓库后会产生新的记录，这样就保留了数据变化的历史轨迹，因此，数据仓库中的数据基本是静态的，因为数据仓库的目的就是要根据曾经发生的事件进行分析，如果数据是可修改的，那么历史数据分析就没有意义了。

时变

为了发现业务变化的趋势、存在的问题以及新的机会，需要分析大量的历史数据。数据仓库中包含各种粒度的历史数据，虽然数据仓库具有非易失性，但数据仓库中的数据并不是不变的，是随着时间进行变化的，也就是说数据仓库中的数据反映的是某一历史时间点的数据快照，当业务发生变化，挖掘出的模式会失去时效性，并且任何一个存储介质都不可能无限扩展，数据也不可能只入不出永久驻留在数据仓库中，因此，数据仓库中的数据也有自己的生命期限，过期会从数据仓库中移除。移除的方式可以是将细节数据进行汇总后删除，将老的数据转储到大容量介质后删除和直接物理删除等。数据仓库是联机分析处理（OLAP），与之对应的是操作型系统是联机事务处理（OLTP），OLTP反映的是当前时间点的数据情况，要求高性能、高并发和极短的响应时间,因此从提高性能考虑,OLTP系统一般会根据数据的活跃程度分级,将活跃度低的历史数据迁移到归档数据库中,也就是我们通常所说的”热库”和”冷库”。

粒度

数据仓库还有一个重要的概念就是粒度，粒度是指数据的细化和综合的程度，细化程度越高，粒度级别就越低，例如，单个事务是低粒度级别的，而一个月事务的汇总就是高粒度级别。粒度之所以是数据仓库环境的关键设计问题，是因为它极大的影响数据仓库的数据量和可以进行的查询类型，粒度级别越低，数据量越大，查询的细化程度就越高，查询的范围越广泛，反之亦然。大多数情况下，数据会以很低粒度级别进入数据仓库，比如日志类型的数据或者点击流数据等，此时应对数据进行汇总和过滤，使其适应数据仓库的粒度级别，以减少数据存储量。因此在设计数据仓库的时候，要在数据量大小和查询的详细程度之间做出权衡。

### 建立数据仓库的必要性

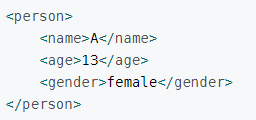
数据仓库的数据通常是来自各个业务应用系统。业务系统的数据形式多种多样，可能是Oracle、MySQL、DB2等关系型数据库里的结构化数据，也可能是文本、csv等平面文件或者word、excel等非结构化数据以及JSON、XML等半结构化数据。下面详细解释一下数据仓库的数据类型。

数据仓库数据源类型有：结构化数据、半结构化数据、非结构化数据

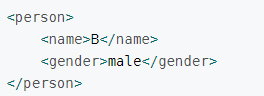
结构化数据：结构化数据是指可以使用关系型数据库进行表示和存储的数据，一般特点为数据以行单位，一行数据表示一个实体的信息，每一行的数的属性是相同的（即数据的列），结构化的数据的存储和排列是有规律的，这对查询和修改数据很有帮助，但扩展性不好，如果要增加一个属性（列），对已有数据的影响就很大。

C:\Users\nie\AppData\Local\Temp\1542613437(1).png

半结构化数据：半结构化数据是结构化数据的一种形式，它并不符合关系型数据库或其他数据表的形式的数据模型结构，但包含相关标记，用来分隔语义元素以及对记录和字段进行分层，因此，它也被称为自描述的结构。半结构化数据，属于同一类实体可以有不同的属性，即使他们被组合在一起，这些属性的顺序也并不重要。常用的半结构化数据有JSON、XML等，对于两个XML文件来说，第一个文件可能是



第二个XML可能是



从上面的例子中，属性的个数不一定相同，但他们有属性的描述和属性值，可以通过属性格式的定义，自动的表达很多有用的信息，因此，半结构化数据的可扩展性很好。

非结构化数据

非结构化数据没有固定的格式，各种文档，图片、音视频等都属于非结构化数据，一般存储为二进制的数据文件。

上面我们讲了数据源的数据的各种各样的类型，所有这些业务数据就需要经过一系列的操作：抽取、转换、清洗，最终以一种统一的格式装载到数据仓库中，作为数据分析的数据源，提供给后面的即席查询（用户根据自己的需求，灵活的设置自定义查询条件，普通应用查询的查询条件是定制开发的，是固化的）、分析系统、数据集市、报表系统、数据挖掘系统等。

从数据存储的角度看，数据仓库里的数据实际上已经存在于业务应用系统中了，那么为什么不能直接操作业务系统中的数据进行数据分析，而要使用数据仓库呢？原因有多个方面包括：

* 由于安全以及权限的因素，有些业务数据不能直接访问
* 业务系统变更频繁，每次变更都要重写分析系统，字段的增删更改等
* 很难生成和维护跨业务系统的汇总报表
* 业务系统的列名通常是硬编码，有时仅仅是无意义的字符串（拼音缩写等，很多保险公司的字段用大写的拼音缩写，基本上看不出含义），当业务系统很多，命名规范不一致的问题，给分析人员带来很大的困难
* 业务系统的表结构为事务处理性能而优化，有时并不适合查询分析，有时候要取个汇总的数据要关联很多表
* 没有适当的方式将有价值的数据合并进特定应用的数据库
* 没有适当的位置存储元数据
* 用户需要的字段，有时在业务数据库中并不存在
* 分析查询往往很耗性能，这给业务系统增加额外性能压力，会影响业务系统的正常运行
* 通常事务处理的优先级要高于分析系统，如果分析系统和事务处理运行在一个服务器上，分析系统往往性能很差并且性能的消耗，如果宕机直接就影响到了事务处理，没有在业务系统运行的服务器上去分析数据的
* 背锅，这个是我碰到的最多的情况，如果跟业务系统一起，只要业务系统出了问题，开发人员一定会说是分析数据造成的

基于以上几点，数据仓库的主要作用是辅助决策，从无规律的海量数据找到数据的规律，发现数据的价值，辅助决策，数据仓库是起着重大作用的。但如果业务量很小，业务比较单一，那么就没有建立数据仓库的必要，毕竟建立数据仓库也是需要人力、硬件等成本的。如果数据量庞大，像京东淘宝这样的大型电商，要搞清楚一个特定的商品到底适合不适合18-25岁的人群，以决定这个商品是否推荐给这个群体，这个商品应该使用什么样的销售策略，这个时候数据仓库的作用就得到了体现。我们需要去统计历史数据的销售量和点击量，分析这个商品的购买人群的年龄分布，看它的主要销售群体是哪些客户，有数据分析的支撑，就可以做到智能化的商品推荐。

### 操作型系统（OLTP）和分析系统(OLAP)

OLTP和OLAP的区别

联机事务处理OLTP（on-line transaction processing） 主要是执行基本日常的事务处理，比如数据库记录的增删查改。比如在银行的一笔交易记录，就是一个典型的事务。

OLTP的特点一般有：

1.实时性要求高。我记得之前上大学的时候，银行异地汇款，要隔天才能到账，而现在是分分钟到账的节奏，说明现在银行的实时处理能力大大增强。

2.数据量不是很大，生产库上的数据量一般不会太大，而且会及时做相应的数据处理与转移。

3.交易一般是确定的，比如银行存取款的金额肯定是确定的，所以OLTP是对确定性的数据进行存取

4.高并发，并且要求满足ACID原则。比如两人同时操作一个银行卡账户，比如大型的购物网站秒杀活动时上万的QPS请求。

联机分析处理OLAP（On-Line Analytical Processing） 是数据仓库系统的主要应用，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。典型的应用就是复杂的动态的报表系统。

OLAP的特点一般有：

1.实时性要求不是很高，比如最常见的应用就是天级更新数据，然后出对应的数据报表。

2.数据量大，因为OLAP支持的是动态查询，所以用户也许要通过将很多数据的统计后才能得到想要知道的信息，例如时间序列分析等等，所以处理的数据量很大;

3.OLAP系统的重点是通过数据提供决策支持，所以查询一般都是动态，自定义的。所以在OLAP中，维度的概念特别重要。一般会将用户所有关心的维度数据，存入对应数据平台。

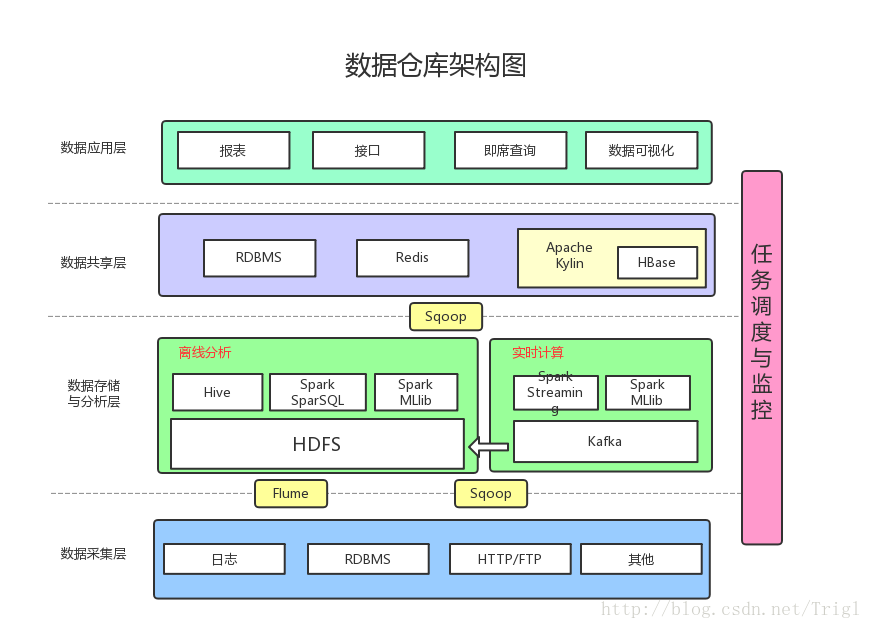
总结：

OLTP即联机事务处理，就是我们经常说的关系数据库，增删查改就是我们经常应用的东西，这是数据库的基础；TPCC(Transaction Processing Performance Council)属于此类。

OLAP即联机分析处理，是数据仓库的核心部心，所谓数据仓库是对于大量已经由OLTP形成的数据的一种分析型的数据库，用于处理商业智能、决策支持等重要的决策信息；数据仓库是在数据库应用到一定程序之后而对历史数据的加工与分析，读取较多，更新较少，TPCH属于此类。

随着大数据时代的到来，对于OLAP，列存储模式或者说nosql模式比传统意义的行存储模式可能更具优势。

### 数据仓库的架构



数据采集

数据采集层的任务就是把数据从各种数据源中采集和存储到数据存储上，期间有可能会做一些ETL操作。

数据采集，采用Flume收集日志，采用Sqoop将RDBMS以及NoSQL中的数据同步到HDFS上。

数据源种类可以有多种：

日志：所占份额最大，存储在备份服务器上

业务数据库：如Mysql、Oracle

来自HTTP/FTP的数据：合作伙伴提供的接口

其他数据源：如Excel等需要手工录入的数据

数据存储与分析

HDFS是大数据环境下数据仓库/数据平台最完美的数据存储解决方案。

离线数据分析与计算，也就是对实时性要求不高的部分，Hive是不错的选择。

使用Hadoop框架自然而然也提供了MapReduce接口，如果真的很乐意开发Java，或者对SQL不熟，那么也可以使用MapReduce来做分析与计算。

Spark性能比MapReduce好很多，同时使用SparkSQL操作Hive。

消息系统：可以加入Kafka防止数据丢失

实时计算：实时计算使用Spark Streaming消费Kafka中收集的日志数据，实时计算结果大多保存在Redis中

机器学习：使用了Spark MLlib提供的机器学习算法

多维分析OLAP：使用Kylin作为OLAP引擎

数据共享

前面使用Hive、MR、Spark、SparkSQL分析和计算的结果，还是在HDFS上，但大多业务和应用不可能直接从HDFS上获取数据，那么就需要一个数据共享的地方，使得各业务和产品能方便的获取数据。

这里的数据共享，其实指的是前面数据分析与计算后的结果存放的地方，其实就是关系型数据库和NOSQL数据库。

数据应用

报表：报表所使用的数据，一般也是已经统计汇总好的，存放于数据共享层。

接口：接口的数据都是直接查询数据共享层即可得到。

即席查询：即席查询通常是现有的报表和数据共享层的数据并不能满足需求，需要从数据存储层直接查询。一般都是通过直接操作SQL得到。

数据可视化：提供可视化前端页面，方便运营等非开发人员直接查询

### 数据存储和ETL

ETL是将业务系统的数据经过抽取、清洗转换之后加载到数据仓库的过程，目的是将企业中的分散、零乱、标准不统一的数据整合到一起，为企业的决策提供分析依据。 ETL是BI项目重要的一个环节。 通常情况下，在BI项目中ETL会花掉整个项目至少1/3的时间,ETL设计的好坏直接关接到BI项目的成败。

　　ETL的设计分三部分：数据抽取、数据的清洗转换、数据的加载。在设计ETL的时候我们也是从这三部分出发。数据的抽取是从各个不同的数据源抽取到ODS(Operational Data Store，操作型数据存储)中——这个过程也可以做一些数据的清洗和转换)，在抽取的过程中需要挑选不同的抽取方法，尽可能的提高ETL的运行效率。ETL三个部分中，花费时间最长的是“T”(Transform，清洗、转换)的部分，一般情况下这部分工作量是整个ETL的2/3。数据的加载一般在数据清洗完了之后直接写入DW(Data Warehousing，数据仓库)中去。

ETL的实现有多种方法，常用的有三种。一种是借助ETL工具(如Oracle的OWB、SQL Server 2000的DTS、SQL Server2005的SSIS服务、Informatic等)实现，一种是SQL方式实现，另外一种是ETL工具和SQL相结合。前两种方法各有各的优缺点，借助工具可以快速的建立起ETL工程，屏蔽了复杂的编码任务，提高了速度，降低了难度，但是缺少灵活性。SQL的方法优点是灵活，提高ETL运行效率，但是编码复杂，对技术要求比较高。第三种是综合了前面二种的优点，会极大地提高ETL的开发速度和效率。

一、 数据的抽取（Extract）

这一部分需要在调研阶段做大量的工作，首先要搞清楚数据是从几个业务系统中来,各个业务系统的数据库服务器运行什么DBMS,是否存在手工数据，手工数据量有多大，是否存在非结构化的数据等等，当收集完这些信息之后才可以进行数据抽取的设计。

1、对于与存放DW的数据库系统相同的数据源处理方法

这一类数据源在设计上比较容易。一般情况下，DBMS(SQLServer、Oracle)都会提供数据库链接功能，在DW数据库服务器和原业务系统之间建立直接的链接关系就可以写Select 语句直接访问。

2、对于与DW数据库系统不同的数据源的处理方法

　　对于这一类数据源，一般情况下也可以通过ODBC的方式建立数据库链接——如SQL Server和Oracle之间。如果不能建立数据库链接，可以有两种方式完成，一种是通过工具将源数据导出成.txt或者是.xls文件，然后再将这些源系统文件导入到ODS中。另外一种方法是通过程序接口来完成。

3、对于文件类型数据源(.txt,.xls)，可以培训业务人员利用数据库工具将这些数据导入到指定的数据库，然后从指定的数据库中抽取。或者还可以借助工具实现

4、增量更新的问题

对于数据量大的系统，必须考虑增量抽取。一般情况下，业务系统会记录业务发生的时间，我们可以用来做增量的标志,每次抽取之前首先判断ODS中记录最大的时间，然后根据这个时间去业务系统取大于这个时间所有的记录。利用业务系统的时间戳，一般情况下，业务系统没有或者部分有时间戳。

二、数据的清洗转换（Cleaning、Transform）

　 一般情况下，数据仓库分为ODS、DW两部分。通常的做法是从业务系统到ODS做清洗，将脏数据和不完整数据过滤掉，在从ODS到DW的过程中转换，进行一些业务规则的计算和聚合。

1、 数据清洗

　　数据清洗的任务是过滤那些不符合要求的数据，将过滤的结果交给业务主管部门，确认是否过滤掉还是由业务单位修正之后再进行抽取。

不符合要求的数据主要是有不完整的数据、错误的数据、重复的数据三大类。

(1)不完整的数据：这一类数据主要是一些应该有的信息缺失，如供应商的名称、分公司的名称、客户的区域信息缺失、业务系统中主表与明细表不能匹配等。对于这一类数据过滤出来，按缺失的内容分别写入不同Excel文件向客户提交，要求在规定的时间内补全。补全后才写入数据仓库。

(2)错误的数据：这一类错误产生的原因是业务系统不够健全，在接收输入后没有进行判断直接写入后台数据库造成的，比如数值数据输成全角数字字符、字符串数据后面有一个回车操作、日期格式不正确、日期越界等。这一类数据也要分类，对于类似于全角字符、数据前后有不可见字符的问题，只能通过写SQL语句的方式找出来，然后要求客户在业务系统修正之后抽取。日期格式不正确的或者是日期越界的这一类错误会导致ETL运行失败，这一类错误需要去业务系统数据库用SQL的方式挑出来，交给业务主管部门要求限期修正，修正之后再抽取。

(3)重复的数据：对于这一类数据——特别是维表中会出现这种情况——将重复数据记录的所有字段导出来，让客户确认并整理。

　　数据清洗是一个反复的过程，不可能在几天内完成，只有不断的发现问题，解决问题。对于是否过滤，是否修正一般要求客户确认，对于过滤掉的数据，写入Excel文件或者将过滤数据写入数据表，在ETL开发的初期可以每天向业务单位发送过滤数据的邮件，促使他们尽快地修正错误,同时也可以做为将来验证数据的依据。数据清洗需要注意的是不要将有用的数据过滤掉，对于每个过滤规则认真进行验证，并要用户确认。

2、 数据转换

数据转换的任务主要进行不一致的数据转换、数据粒度的转换，以及一些商务规则的计算。

(1)不一致数据转换：这个过程是一个整合的过程，将不同业务系统的相同类型的数据统一，比如同一个供应商在结算系统的编码是XX0001,而在CRM中编码是YY0001，这样在抽取过来之后统一转换成一个编码。

(2)数据粒度的转换：业务系统一般存储非常明细的数据，而数据仓库中数据是用来分析的，不需要非常明细的数据。一般情况下，会将业务系统数据按照数据仓库粒度进行聚合。

(3)商务规则的计算：不同的企业有不同的业务规则、不同的数据指标，这些指标有的时候不是简单的加加减减就能完成，这个时候需要在ETL中将这些数据指标计算好了之后存储在数据仓库中，以供分析使用。

三、ETL日志、警告发送

1、 ETL日志

ETL日志分为三类。

一类是执行过程日志，这一部分日志是在ETL执行过程中每执行一步的记录，记录每次运行每一步骤的起始时间，影响了多少行数据，流水账形式。

一类是错误日志，当某个模块出错的时候写错误日志，记录每次出错的时间、出错的模块以及出错的信息等。

第三类日志是总体日志，只记录ETL开始时间、结束时间是否成功信息。如果使用ETL工具,ETL工具会自动产生一些日志，这一类日志也可以作为ETL日志的一部分。

记录日志的目的是随时可以知道ETL运行情况，如果出错了，可以知道哪里出错。

2、 警告发送

如果ETL出错了，不仅要形成ETL出错日志，而且要向系统管理员发送警告。发送警告的方式多种，一般常用的就是给系统管理员发送邮件，并附上出错的信息，方便管理员排查错误。

ETL是BI项目的关键部分，也是一个长期的过程，只有不断的发现问题并解决问题，才能使ETL运行效率更高，为BI项目后期开发提供准确与高效的数据。

做数据仓库系统，ETL是关键的一环。说大了，ETL是数据整合解决方案，说小了，就是倒数据的工具。回忆一下工作这么长时间以来，处理数据迁移、转换的工作倒还真的不少。但是那些工作基本上是一次性工作或者很小数据量。可是在数据仓库系统中，ETL上升到了一定的理论高度，和原来小打小闹的工具使用不同了。究竟什么不同，从名字上就可以看到，人家已经将倒数据的过程分成3个步骤，E、T、L分别代表抽取、转换和装载。

其实ETL过程就是数据流动的过程，从不同的数据源流向不同的目标数据。但在数据仓库中，

ETL有几个特点，

一是数据同步，它不是一次性倒完数据就拉到，它是经常性的活动，按照固定周期运行的，甚至现在还有人提出了实时ETL的概念。

二是数据量，一般都是巨大的，值得你将数据流动的过程拆分成E、T和L。

现在有很多成熟的工具提供ETL功能，且不说他们的好坏。从应用角度来说，ETL的过程其实不是非常复杂，这些工具给数据仓库工程带来和很大的便利性，特别是开发的便利和维护的便利。但另一方面，开发人员容易迷失在这些工具中。举个例子，VB是一种非常简单的语言并且也是非常易用的编程工具，上手特别快，但是真正VB的高手有多少？微软设计的产品通常有个原则是“将使用者当作傻瓜”，在这个原则下，微软的东西确实非常好用，但是对于开发者，如果你自己也将自己当作傻瓜，那就真的傻了。ETL工具也是一样，这些工具为我们提供图形化界面，让我们将主要的精力放在规则上，以期提高开发效率。从使用效果来说，确实使用这些工具能够非常快速地构建一个job来处理某个数据，不过从整体来看，并不见得他的整体效率会高多少。问题主要不是出在工具上，而是在设计、开发人员上。他们迷失在工具中，没有去探求ETL的本质。可以说这些工具应用了这么长时间，在这么多项目、环境中应用，它必然有它成功之处，它必定体现了ETL的本质。如果我们不透过表面这些工具的简单使用去看它背后蕴涵的思想，最终我们作出来的东西也就是一个个独立的job，将他们整合起来仍然有巨大的工作量。大家都知道“理论与实践相结合”，如果在一个领域有所超越，必须要在理论水平上达到一定的高度.

## 数据仓库设计基础、维表和事实表技术

专业数据仓库面临的一个问题是数据仓库中数据库设计的基本模型选取问题。广泛采用的数据库设计模型有两种，关系型和多维型。

下面介绍两种模型，及其两种方法的区别和在数据仓库中的应用，两种方法的优缺点。在建立数据仓库过程中，对于数据库设计而言，建立关系型数据库是最佳的长期的方法，并且这种情况需要真正的企业方法。多维模型有利于短期数据仓库，但这种方法使用的数据仓库的范围有限。

一、关系模型

二、多维模型

三、两种模型的区别

一、关系模型

关系型数据库设计首先要创建一张数据表，表中每一行包含不同的列。关系表可以包含不同的属性，每一数据列表示不同的物理特征。不同的列可以索引并作为标识符。部分列在执行过程中可以为空。所有列都是根据数据定义语言（DDL）标准定义的。关系型数据以一种称为“标准化”的形式存在。数据标准化是指数据库设计会使数据分解成非常低的粒度级。标准化的数据以一种孤立模式存在，这种情况下对数据表里的数据关系要求很严格。当进行标准化的时候，表中的数据只能与这张表里的其他数据关联。

通过关系模型产生的数据仓库的设计是很灵活的。基于设计的数据库起初可以是一种方式，当根据关系模型设计后又形成另一种形式。数据元可以以多种方式重新赋值。灵活性是关系模型最大的优势，其次是功能性。因为细节数据需要被收集到一起并且能够结合，因此给予关系模型的数据仓库的设计可以支持数据的多种视图。

二、多维模型

星形连接：

多维模型方法也叫星形连接。

数据库设计多维模型方法的中心是星形连接，之所以称为星形连接，是因为它的表示方法是以一颗“星”为中心，周围围绕着其他数据结构。即星形连接的中心是一张事实表。事实表是包含大量数据值的一种结构。事实表的周围是维表，用来描述事实表的某个重要方面。

雪花结构：

通常，星形连接只包含一张事实表。但是在数据库设计中要创建一种雪花结构的复合结构需要多张事实表结合。在雪花结构中，不同的事实表通过共享一个或多个公共维表连接起来。雪花结构隐含的另外一个想法是将事实表和维表结合起来，形成一个类似于雪花结构的形式。

多维模型设计的最大优点在于访问的高效性。当设计适当时，通过星形连接将数据传递给最终用户是非常高效的。为了提高传递信息的效率，必须收集并吸收最终用户的请求。最终用户使用数据的过程是要定义什么样的多维结构的核心。一旦清楚了最终用户的请求，这些请求就可以用来最终确定星形连接，形成最理想的结构。

三、两种模型的区别

最重要的区别：灵活性和性能。

灵活性和性能：

关系模型粒度低，具有高灵活性，但是对用户来说在性能方面却不是理想的。

多维模型在满足用户需求方面是非常高效的，但灵活性不好。

设计的范围不同：

多维设计通过用户需求建立模型，只能在有限范围内进行，适合于小范围数据（如一个部门甚至子部门），仅服务于一组用户的需求；

关系型设计以最低粒度进行存储，那么就可以无限制的添加新数据。很显然，添加数据到关系模型永远也不会停止。所以关系模型适合于大范围数据(如一个企业模型），能满足多个企业的信息需求

关系模型对数据仓库的直接访问是最佳的，而多维模型用于服务数据仓库的直接用户的需求时最理想的。

四、独立数据集市和从属数据集市

数据集市：是用来表示服务一组特定群体的分析需求的数据结构。

独立数据集市：是指直接通过历史应用创建的数据集市。独立数据集市可以由单一的部门创建，而不考虑其他部门或组织。建立独立数据集市也不需要有“全局思想”考虑。独立数据集市表示企业全部DSS请求的一个子集，建立独立数据集市的费用不高，并且允许企业掌控子集的信息。

从属数据集市：与独立数据集市对应。从属数据集市是利用来自数据仓库的数据建立的。它的数据源不依赖于历史数据或操作型数据，只依赖于数据仓库。从属数据集市要求预先计划和投资，并需要“全局考虑”。此外，从属数据集市要求多个用户共享他们创建数据仓库时的信息需求。总之，从属数据集市要求有预先的计划、长期的观察、全局的分析和企业各不同部门对需求分析的合作与协调。

多维模型的另一特点是在数据集市中进行分析时，能够直接访问历史和操作型数据。数据集市分为两类：独立数据集市和从属数据集市。

独立数据集市存在很多问题，而这些问题只有当建立起一定数目的独立数据集市后才会变得明显。

独立数据集市存在以下问题：

不提供数据重用平台；

不提供数据一致性的基础；

不提供单一历史接口程序基础；

需要每一个独立数据集市创建自己的细节数据集，其他的独立数据集市会建立大量的冗余数据。

从属数据集市可以从数据仓库提起数据，而不存在独立数据集市存在的体系结构的问题。

维度表

在星型模式中，每个维度表都分配有一个代理键（surrogate key，SK）。该列是维度表的唯一标识符，是维度表的事实主键（这里事实主键是指事实意义上的主键，能唯一标识维度表中的一行），只在数据仓库中创建。代理键在星型模式的加载过程中分配和维护。代理键没有内在的含义，通常表现为一个整数。代理键有时指的是warehouse key，是维度表的主键。

维度表中也包含类似操作型系统中存在的用于唯一区分实体的键列。这些操作型系统中的键通常称为自然键（natural key，NK）。NK在维度表中未必唯一标识一条记录，即并非维度表的事实主键。

在数据仓库中，区分代理键和自然键的目的是跟踪在操作性系统中无须考虑的数据变化情况。例如，假定客户A在操作型系统中以customer\_id 10711标识。如果客户的位置发生变化，操作型系统中只需对customer\_id为10711的记录修改；而从分析角度考虑，可能需要根据地区统计，因此不能直接覆盖维度表中相关记录。因为星型模式的客户维度表中不以customer\_id作为事实主键，仅作为NK，因此可以存储多个版本的客户A的信息，这些版本都具有相同的customer\_id。不同版本的信息可以通过不同的代理键加以区分。这样就通过增加带有序号的自然键的方式对变化进行跟踪。代理键可以基于单一的列实现事实表和维度表之间的连接操作。

这与缓慢变化维有关。

维度表中包含的列应该尽可能全面。如对于操作型系统中为代码（如，使用0和1代表男和女）的列，维度表中应该包含该代码（0或1）和代码描述（男或女）。

2 事实表

事实表由紧凑的包含引用维度和事实的外键构成。

事实表应该包含所有与过程有关的事实，即使某些事实可以由其他事实计算得来。类似比率等非可加事实应该分解为完全可加的组成部分，其计算应该在创建报表时执行。

事实表是稀疏的，只有当某些事实发生时才产生相应的记录行。

对事实表粒度的声明非常重要，要么以维度术语声明，要么以业务术语声明。

存储在事实表中的维度被称为退化维度。这种技术通常用于具有较高基数（cardinality）的事务标识符中。

记录在事实表中的行表示业务活动的发生情况，这意味着事实表中的行没有包含所有可能的维度组合。出现在事实表中的组合数量远远小于可能存在的组合数量。事实表的这项特性称为稀疏性。例如，某客户在某天未从某销售商处购买特定产品，则不会有此项记录。

多数情况下，模式设计者会避免在将数据加载到事实表之前聚合数据。尽可能保持最细粒度的数据，星型模式就能解决范围更加宽泛的分析型需求（即扩展性更好）。无论采用何种数据仓库结构（多维数据仓库或企业信息化工厂，CIF），这一指导原则都普遍适用。对多维数据仓库来说，事实表包含细粒度数据至关重要，因为多维数据仓库是细节数据的集中存储仓库。对于CIF，这一指导原则可以宽松一些，因为数据仓库中包含细粒度数据，因此数据集市的事实表可以聚集数据，而不用担心丢失信息。不过这些数据集市可能无法满足新需求。

存储到事实表中的维度列被称为退化维度，简称退化维。虽然被存储在事实表中，但该列仍被视为维度。与其他表中的维度列一样，其值仍然可以用于过滤查询、控制聚合层次、排序数据、定义主从关系等。应该改谨慎使用退化维度，因为事实表累计记录的速度很快，包含退化维度可能会造成存储空间的过度消耗，特别是当退化维度为文本元素时。多数情况下，适合选作退化维度的维度最好放置到杂项维度中，事务标识除外。

事务标识通常作为退化维度存储。它也可以作为事实表中行的唯一标识，并用于定义事实表的粒度。

3 缓慢变化维

维度表中的数据来源于操作型系统。在多维数据仓库（Kimball）或独立型数据集市中，数据直接来源于操作型系统。在企业信息化工厂（Inmon）中，来自于操作型系统的数据首先移到企业数据仓库中，然后进入多维数据集市。进入到维度表中的信息，在操作型系统中可能发生变化。因此维度设计中需要确定维度表如何处理数据源的发生变化的情况，这种维度表称为缓慢变化的维度，简称缓慢变维。

由于维度表引入了代理键作为其主键，因此不需要与源系统采用相同的处理方式。操作型系统可以跟踪数据变化的历史情况，也可以简单地采用重写变化值的方式。对于任何一种情况，星型模式都可以采用两种响应方式：变化类型1、变化类型2。

变化类型1在响应数据源变化时，重写维度属性。这样维度表不能反映历史情况，已存在事实的历史环境被改变了。

变化类型2在源数据值发生改变时，创建一个新版本的维度行。变化类型2保存了变化的历史事实。描述变化前的事件的事实与过去的值关联，描述变化后的事件的事实将与新值关联。多数操作型系统的变化采用变化类型2处理。

## 基于大数据的数据仓库

数据仓库概念最早可追溯到20世纪70年代，希望提供一种架构将业务处理系统和分析处理分为不同的层次。

20世纪80年代，建立TA2(Technical Architecture2)规范，该明确定义了分析系统的四个组成部分：数据获取、数据访问、目录、用户服务

1988年，IBM第一次提出信息仓库的概念：一个结构化的环境，能支持最终用户管理其全部的业务，并支持信息技术部门保证数据质量；抽象出基本组件：数据抽取、转换、有效性验证、加载、cube开发等，基本明确了数据仓库的基本原理、框架结构，以及分析系统的主要原则

1991年，Bill Inmon出版《 Building the Data Warehouse 》提出了更具体的数据仓库原则：

1.数据仓库是面向主题的

2.集成的

3.包含历史的

4.不可更新的

5.面向决策支持的

6.面向全企业的

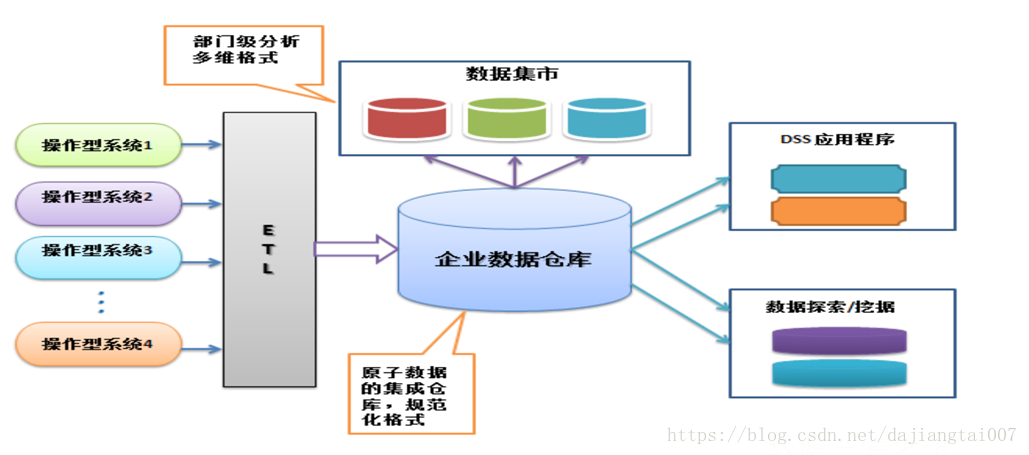
7.最明细的数据存储

8.数据快照式的数据获取

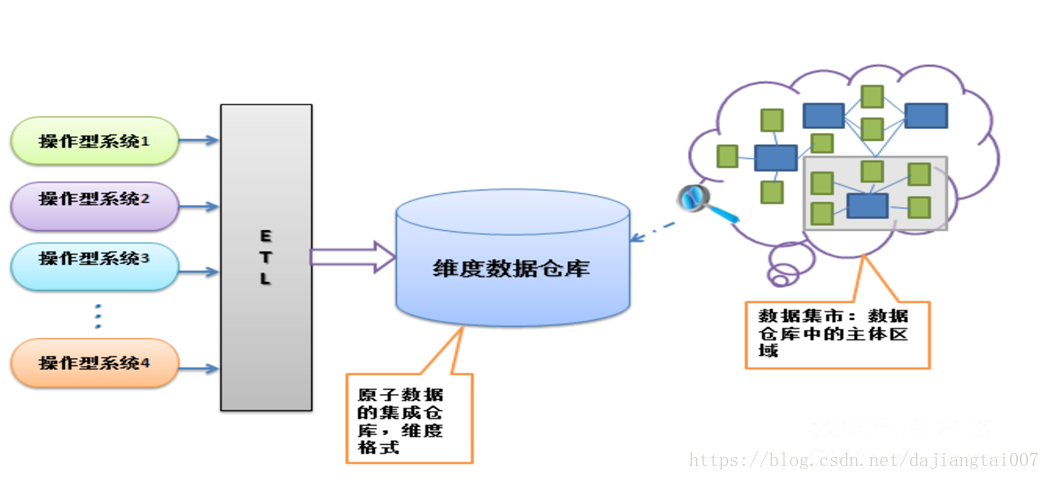
尽管有些理论目前仍有争议，但凭借此书获得“数据仓库之父”的殊荣

Bill Inmon主张自上而下的建设企业数据仓库，认为数据仓库是一个整体的商业智能系统的一部分。

一家企业只有一个数据仓库，数据集市的信息来源出自数据仓库，在数据仓库中，信息存储符合第三范式，大致架构：



Ralph Kimball 出版《The Data Warehouse Toolkit》，其主张自下而上的建立数据仓库，极力推崇建立数据集市，认为数据仓库是企业内所有数据集市的集合，信息总是被存储在多维模型当中，其思路：



两种思路和观点在实际的操作中都很难成功的完成项目交付，直至最终Bill Inmon提出了新的BI架构CIF(Corporation information factory),把数据集市包含了进来。CIF的核心是将数仓架构划分为不同的层次以满足不同场景的需求，比如常见的ODS、DW、DM等，每层根据实际场景采用不同的建设方案，改思路也是目前数据仓库建设的架构指南，但自上而下还是自下而上的进行数据仓库建设，并未统一。

基于大数据数仓构建特点

随着我们从IT时代步入DT时代，数据从积累量也与日俱增，同时伴随着互联网的发展，越来越多的应用场景产生，传统的数据处理、存储方式已经不能满足日益增长的需求。而互联网行业相比传统行业对新生事物的接受度更高、应用场景更复杂，因此基于大数据构建的数据仓库最先在互联网行业得到了尝试。

尽管数据仓库建模方法论是一致的，但由于所面临的行业、场景的不同，在互联网领域，基于大数据的数据仓库建设无法按照原有的项目流程、开发模式进行，更多的是需要结合新的技术体系、业务场景进行灵活的调整，以快速响应需求为导向。

应用场景广泛

1）传统的数仓建设周期长，需求稳定，面向DSS、CRM、BI等系统，时效性要求不高。

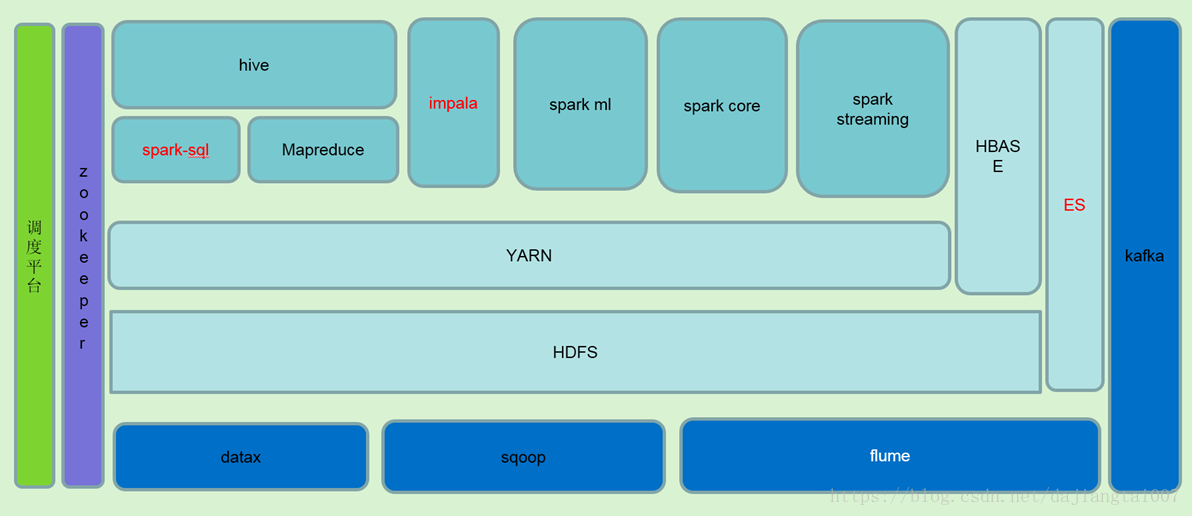
2）基于大数据的数据仓库建设要求快速响应需求，同时需求灵活、多变，对实时性有不同程度的要求，除了面向DSS、BI等传统应用外，还要响应用户画像、个性化推荐、机器学习、数据分析等各种复杂的应用场景。

技术栈更全面、复杂

传统数仓建设更多的基于成熟的商业数据集成平台，比如Teradata、Oracle、Informatica等，技术体系比较成熟完善，但相对比较封闭，对实施者技术面要求也相对专业且单一，一般更多应用于银行、保险、电信等“有钱”行业。

基于大数据的数仓建设一般是基于非商业、开源的技术，常见的是基于hadoop生态构建，涉及技术较广泛、复杂，同时相对于商业产品，稳定性、服务支撑较弱，需要自己维护更多的技术框架。

技术栈转变



数仓模型设计更灵活

1.传统数仓有较为稳定的业务场景和相对可靠的数据质量，同时也有较为稳定的需求，对数仓的建设有较为完善的项目流程管控，数仓模型设计有严格的、稳定的建设标准。

2.在互联网行业：

1）行业变化快、业务灵活，同时互联网又是个靠速度存活的行业

2）源数据种类繁多：数据库、Nginx log、用户浏览轨迹等结构化、非结构化、半结构化数据

3）数据质量相对差，层次不齐

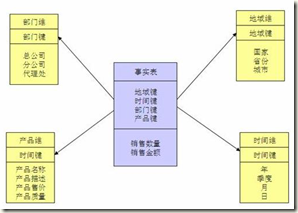
所以，在互联网领域，数仓模型的设计更关注灵活、快速响应和应对多变的市场环境，更加以快速解决业务、运营问题为导向，快速数据接入、快速业务接入，更不存在一劳永逸。

## 数据仓库模型

数据仓库中常见的模型有：范式建模，雪花模型，星型建模，事实星座模型.

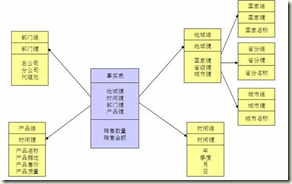
星型模型

星型模型是数据集市维度建模中推荐的建模方法。星型模型是以事实表为中心，所有的维度表直接连接在事实表上，像星星一样。星型模型的特点是数据组织直观，执行效率高。因为在数据集市的建设过程中，数据经过了预处理，比如按照维度进行了汇总，排序等等，数据量减少，执行的效率就比较高。

[](http://images2015.cnblogs.com/blog/1008827/201610/1008827-20161029202446265-1606931840.png)

雪花模型

雪花模型也是维度建模中的一种选择。雪花模型的维度表可以拥有其他维度表的，虽然这种模型相比星型模型更规范一些，但是由于这种模型不太容易理解，维护成本比较高，而且性能方面需要关联多层维表，性能也比星型模型要低。所以一般不是很常用。

[](http://images2015.cnblogs.com/blog/1008827/201610/1008827-20161029202447796-91263897.png)

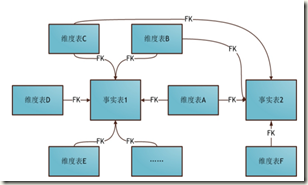
范式建模

第三范式建模是在数据库建模中使用的建模方法，特点是体系化，扩展性好，避免冗余，避免更新异常。所以，在数据仓库的EDW层建模中，我们也提倡使用第三范式建模。但是数据仓库的集成和反映历史变化的特征意味着数据量非常之大，表和表之间的关联效率比较低，所以有些时候完全规范的范式建模并不是最好的选择，通常我们会选择非规范化处理，增加一些冗余的字段来避免表之间关联的次数，这样会节约大量的时间。

雪花模型是介于星型模型和范式建模之间的。个人理解，范式建模和雪花模型的区别在于雪花模型在维度上也是有冗余的。例如雪花模型例图的地域维度不符合第三范式，因为地域维度中存在传递依赖，城市-省级-国家-地域。

星座模型

星座模型是星型模型延伸而来，星型模型是基于一张事实表的，而星座模型是基于多张事实表的，而且共享维度信息。 通过构建一致性维度，来建设星座模型，也是很好的选择。比如同一主题的细节表和汇总表共享维度，不同主题的事实表，可以通过在维度上互相补充来生成可以共享的维度。

[](http://images2015.cnblogs.com/blog/1008827/201610/1008827-20161029202449281-126910885.png)

## 数据的抽取、转换、装载和自动化ETL

ETL(Extract-Transform-Load的缩写，即[数据抽取](http://baike.baidu.com/view/709638.htm" \t "_blank)、转换、装载的过程)作为BI/DW（Business Intelligence）的核心和灵魂，能够按照统一的规则集成并提高数据的价值，是负责完成数据从[数据源](http://baike.baidu.com/view/286828.htm" \t "_blank)向目标[数据仓库](http://baike.baidu.com/view/19711.htm" \t "_blank)转化的过程，是实施数据仓库的重要步骤。如果说[数据仓库](http://baike.baidu.com/view/19711.htm" \t "_blank)的模型设计是一座大厦的设计蓝图，数据是砖瓦的话，那么ETL就是建设大厦的过程。在整个项目中最难部分是用户需求分析和模型设计，而ETL规则设计和实施则是工作量最大的，约占整个项目的60%～80%，这是国内外从众多实践中得到的普遍共识。

ETL是[数据抽取](http://baike.baidu.com/view/709638.htm" \t "_blank)（Extract）、清洗（Cleaning）、转换（Transform）、装载（Load）的过程。是构建[数据仓库](http://baike.baidu.com/view/19711.htm" \t "_blank)的重要一环，用户从[数据源](http://baike.baidu.com/view/286828.htm" \t "_blank)抽取出所需的数据，经过[数据清洗](http://baike.baidu.com/view/1739747.htm" \t "_blank),最终按照预先定义好的数据仓库模型，将数据加载到数据仓库中去。

信息是现代企业的重要资源，是企业运用科学管理、决策分析的基础。目前，大多数企业花费大量的资金和时间来构建[联机事务处理](http://baike.baidu.com/view/8028.htm" \t "_blank)OLTP的业务系统和[办公自动化系统](http://baike.baidu.com/view/45194.htm" \t "_blank)，用来记录事务处理的各种相关数据。据统计，数据量每2～3年时间就会成倍增长，这些数据蕴含着巨大的商业价值，而企业所关注的通常只占在总数据量的2%～4%左右。因此，企业仍然没有最大化地利用已存在的数据资源，以致于浪费了更多的时间和资金，也失去制定关键商业决策的最佳契机。于是，企业如何通过各种技术手段，并把数据转换为信息、知识，已经成了提高其核心竞争力的主要瓶颈。而ETL则是主要的一个技术手段。如何正确选择ETL工具？如何正确应用ETL？

ETL工具的典型代表有:Informatica、Datastage、OWB、微软DTS……

开源的工具有eclips的etl插件。http://www.cloveretl.org/clover/

数据集成：快速实现ETL

　　ETL的质量问题具体表现为正确性、完整性、一致性、完备性、有效性、时效性和可获取性等几个特性。而影响质量问题的原因有很多，由系统集成和历史数据造成的原因主要包括:业务系统不同时期系统之间数据模型不一致；业务系统不同时期业务过程有变化；旧系统模块在运营、人事、财务、办公系统等相关信息的不一致；遗留系统和新业务、管理系统数据集成不完备带来的不一致性。

　　实现ETL，首先要实现ETL转换的过程。它可以集中地体现为以下几个方面：

　　空值处理 可捕获字段空值，进行加载或替换为其他含义数据，并可根据字段空值实现分流加载到不同目标库。

　　规范化数据格式 可实现字段格式约束定义，对于数据源中时间、数值、字符等数据，可自定义加载格式。

　　拆分数据 依据业务需求对字段可进行分解。例，主叫号 861084613409，可进行区域码和电话号码分解。

　　验证数据正确性 可利用Lookup及拆分功能进行数据验证。例如，主叫号861084613409，进行区域码和电话号码分解后，可利用Lookup返回主叫网关或交换机记载的主叫地区，进行数据验证。

　　数据替换 对于因业务因素，可实现无效数据、缺失数据的替换。

　　Lookup 查获丢失数据 Lookup实现子查询，并返回用其他手段获取的缺失字段，保证字段完整性。

　　建立ETL过程的主外键约束 对无依赖性的非法数据，可替换或导出到错误数据文件中，保证主键惟一记录的加载。

　　为了能更好地实现ETL，笔者建议用户在实施ETL过程中应注意以下几点：

　　第一，如果条件允许，可利用数据中转区对运营数据进行预处理，保证集成与加载的高效性；

　　第二，如果ETL的过程是主动“拉取”，而不是从内部“推送”，其可控性将大为增强；

　　第三，ETL之前应制定流程化的配置管理和标准协议；

　　第四，关键数据标准至关重要。目前，ETL面临的最大挑战是当接收数据时其各源数据的异构性和低质量。以电信为例，A系统按照统计代码管理数据，B系统按照账目数字管理，C系统按照语音ID管理。当ETL需要对这三个系统进行集成以获得对客户的全面视角时，这一过程需要复杂的匹配规则、名称/地址正常化与标准化。而ETL在处理过程中会定义一个关键数据标准，并在此基础上，制定相应的数据接口标准。

　　ETL过程在很大程度上受企业对源数据的理解程度的影响，也就是说从业务的角度看数据集成非常重要。一个优秀的ETL设计应该具有如下功能：

　　管理简单；采用元数据方法，集中进行管理；接口、数据格式、传输有严格的规范；尽量不在外部数据源安装软件；数据抽取系统流程自动化，并有自动调度功能；抽取的数据及时、准确、完整；可以提供同各种数据系统的接口，系统适应性强；提供软件框架系统，系统功能改变时，应用程序很少改变便可适应变化；可扩展性强。

　　数据模型：标准定义数据

　　合理的业务模型设计对ETL至关重要。数据仓库是企业惟一、真实、可靠的综合数据平台。数据仓库的设计建模一般都依照三范式、星型模型、雪花模型，无论哪种设计思想，都应该最大化地涵盖关键业务数据，把运营环境中杂乱无序的数据结构统一成为合理的、关联的、分析型的新结构，而ETL则会依照模型的定义去提取数据源，进行转换、清洗，并最终加载到目标数据仓库中。

　　模型的重要之处在于对数据做标准化定义，实现统一的编码、统一的分类和组织。标准化定义的内容包括：标准代码统一、业务术语统一。ETL依照模型进行初始加载、增量加载、缓慢增长维、慢速变化维、事实表加载等数据集成，并根据业务需求制定相应的加载策略、刷新策略、汇总策略、维护策略。

　　元数据：拓展新型应用

　　对业务数据本身及其运行环境的描述与定义的数据，称之为元数据（metadata）。元数据是描述数据的数据。从某种意义上说，业务数据主要用于支持业务系统应用的数据，而元数据则是企业信息门户、客户关系管理、数据仓库、决策支持和B2B等新型应用所不可或缺的内容。

　　元数据的典型表现为对象的描述，即对数据库、表、列、列属性（类型、格式、约束等）以及主键/外部键关联等等的描述。特别是现行应用的异构性与分布性越来越普遍的情况下，统一的元数据就愈发重要了。“信息孤岛”曾经是很多企业对其应用现状的一种抱怨和概括，而合理的元数据则会有效地描绘出信息的关联性。

　　而元数据对于ETL的集中表现为：定义数据源的位置及数据源的属性、确定从源数据到目标数据的对应规则、确定相关的业务逻辑、在数据实际加载前的其他必要的准备工作，等等，它一般贯穿整个数据仓库项目，而ETL的所有过程必须最大化地参照元数据，这样才能快速实现ETL。

ETL是指从源系统中提取数据，转换数据为一个标准的格式，并加载数据到目标数据存储区，通常是数据仓库。

ETL体系结构图

　　Design manager 提供一个图形化的映射环境，让开发者定义从源到目标的映射关系、转换、处理流程。设计过程的各对象的逻辑定义存储在一个元数据资料库中。

　　Meta data management 提供一个关于ETL设计和运行处理等相关定义、管理信息的元数据资料库。ETL引擎在运行时和其它应用都可参考此资料库中的元数据。

　　Extract 通过接口提取源数据，例如ODBC、专用数据库接口和平面文件提取器，并参照元数据来决定数据的提取及其提取方式。

　　Transform 开发者将提取的数据，按照业务需要转换为目标数据结构，并实现汇总。

　　Load 加载经转换和汇总的数据到目标数据仓库中，可实现SQL或批量加载。

　　Transport services 利用网络协议或文件协议，在源和目标系统之间移动数据，利用内存在ETL处理的各组件中移动数据。

　　Administration and operation 可让管理员基于事件和时间进行调度、运行、监测ETL作业、管理错误信息、从失败中恢复和调节从源系统的输出。

## 数据分析

数据分析是把隐没在一大批看来杂乱无章的数据中的信息，集中、萃取和提炼出来，以找出所研究对象的内在规律，并提供决策支持的一系列分析过程。数据统计分析方法已经应用到各行各业，在互联网电商的运用也非常广泛。

在做网络推广时，一定要用数据分析作为基础。没有数据做支撑的推广是没有意义的。在这里介绍一些方法和工具。

1. 比较分析法

是统计分析中最常用的方法。是通过有关的指标对比来反映事物数量上差异和变化的方法。指标分析对比分析方法可分为静态比较和动态比较分析。静态比较是同一时间条件下不同总体指标比较，如不同部门、不同地区、不同国家的比较，也叫横向比较；动态比较是同一总体条件不同时期指标数值的比较，也叫纵向比较。这两种方法既可单独使用，也可结合使用。

1. 分组分析法

统计分析不仅要对总体数量特征和数量关系进行分析，还要深入总体的内部进行分组分析。分组分析法就是根据统计分析的目的要求，把所研究的总体按照一个或者几个标志划分为若干个部分，加以整理，进行观察、分析，以揭示其内在的联系和规律性。

统计分组法的关键问题在于正确选择分组标值和划分各组界限。

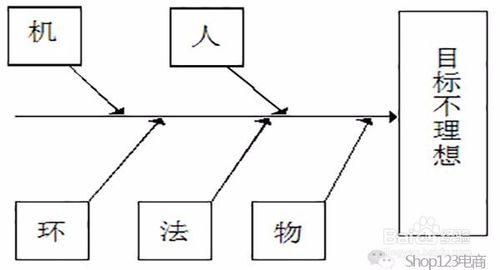
1. 回归分析法

回归分析法是依据事物发展变化的因果关系来预测事物未来的发展走势，它是研究变量间相互关系的一种定量预测方法，回归分析中，当研究的因果关系只涉及因变量和一个自变量时，叫做一元回归分析；当研究的因果关系涉及因变量和两个或两个以上自变量时，叫做多元回归分析。此外，回归分析中，又依据描述自变量与因变量之间因果关系的函数表达式是线性的还是非线性的，分为线性回归分析和非线性回归分析。

[](http://jingyan.baidu.com/album/86fae346fdbf2a3c49121a27.html?picindex=2)

1. 因素分析法

因素分析法的最大功用，就是运用数学方法对可观测的事物在发展中所表现出的外部特征和联系进行由表及里、由此及彼、去粗取精、去伪存真的处理，从而得出客观事物普遍本质的概括。其次，使用因素分析法可以使复杂的研究课题大为简化，并保持其基本的信息量。

[](http://jingyan.baidu.com/album/86fae346fdbf2a3c49121a27.html?picindex=3)

工具：

除了各种科学分析法，在过网络推广时，我们还会用到各种工具：

一、 各种数据分析工具。其实大部分数据分析可以用EXCEL解决，再高阶一点可以用SPSS、SAS等软件。《谁说菜鸟不会数据分析》一书就详细分析了各种工具和实用方法，公众号<shop123电商>里有一些关于这本书的研究，有兴趣可以关注下。

二、  关键词提取。如何从一大推杂乱的信息中提取出关键信息？如何利用这些关键信息去推广自己的产品/网站？光年有一款简单好用的关键词提取工具，可以通过分析文本内容提取出关键信息，从而应用到SEO做关键词研究、优化文章标题或文案、PPC关键词选择等各种应用场景。

## 数据报表及数据可视化

数据可视化，可以增强数据的呈现效果，方便用户以更加直观的方式观察数据，进而发现数据中隐藏的信息。可视化应用领域十分广泛，主要涉及网络数据可视化、交通数据可视化、文本数据可视化、数据挖掘可视化、生物医药可视化、社交可视化等领域。依照CARD可视化模型，将数据可视化过程分为：数据预处理、绘制、显示和交互这几个阶段。依照SHNEIDERMAN分类，可视化的数据分为：一维数据、二维数据、三维数据、高维数据、时态数据、层次数据和网络数据。其中高维数据、层次数据、网络数据、时态数据是当前可视化的研究热点。

  高维数据目前已经成为了计算机领域的研究热点，所谓高维数据是指每一个样本数据包含p(p≥4)维空间特征。人类对于数据的理解主要集中在低维度的空间表示上，如果单从高维数据的抽象数据值上进行分析很难得到有用的信息。相对于对数据的高维模拟，低维空间的可视化技术显得更简单、直截。而且高维空间包含的元素相对于低维空间来说更加更复杂，容易造成人们的分析混乱。将高维数据信息映射到二三维空间上，方便高维数据进行人与数据的交互，有助于对数据进行聚类以及分类。高维数据可视化的研究主要包含数据变化、数据呈现两个方面。

层次数据具有等级或层级关系。层次数据的可视化方法主要包括节点链接图和树图２种方式。其中树图(treemap)由一系列的嵌套环、块来展示层次数据。

为了能展示更多的节点内容，一些基于“焦点＋上下文”技术的交互方法被开发出来。包括“鱼眼”技术、几何变形、语义缩放、远离焦点的节点聚类技术等。

网络数据表现为更加自由、更加复杂的关系网络。分析网络数据的核心是挖掘关系网络中的重要结构性质，如节点相似性、关系传递性、网络中心性等，网络数据可视化方法应清晰表达个体间关系以及个体的聚类关系。主要布局策略包含结点链接法和相邻矩阵法。

时间序列数据是指具有时间属性的数据集，针对时间序列数据的可视化方法包含：线形图、动画、堆积图、时间线、地平线图。

数据可视化伴随着大数据时代的到来而兴起，可视化分析是大数据分析不可或缺的一种重要手段和工具，只有在真正理解可视化概念本质后，才能更好的研究并应用其方法和原理，获得数据背后隐藏的价值。

数据可视化的定义

    数据可视化，是关于数据视觉表现形式的科学技术研究。可视化技术是利用计算机图形学及图像处理技术，将数据转换为图形或图像形式显示到屏幕上，并进行交互处理的理论、方法和技术。它涉及计算机视觉、图像处理、计算机辅助设计、计算机图形学等多个领域，成为一项研究数据表示、数据处理、决策分析等问题的综合技术。

数据可视化的基本概念

1）数据空间。由n维属性、m个元素共同组成的数据集构成的多维信息空间。

2）数据开发。利用一定的工具及算法对数据进行定量推演及计算。

3）数据分析。对多维数据进行切片、块、旋转等动作剖析数据，从而可以多角度多侧面的观察数据。

4）数据可视化。将大型数据集中的数据通过图形图像方式表示，并利用数据分析和开发工具发现其中未知信息。

数据可视化的标准

为实现信息的有效传达，数据可视化应兼顾美学与功能，直观的传达出关键的特征，便于挖掘数据背后隐藏的价值。

可视化技术应用标准应该包含以下4个方面：

1）直观化。将数据直观、形象的呈现出来。

2）关联化。突出的呈现出数据之间的关联性。

3）艺术性。使数据的呈现更具有艺术性，更加符合审美规则。

4）交互性。实现用户与数据的交互，方便用户控制数据。

常用的数据可视化工具

目前常用的数据可视化工具有很多。

1、EXCEL是常用的入门级的数据可视化工具。

可视化应用：报表、统计图表等方面。

优点：快速、方便。

不足：样式选择范围有限

2、Google Chart API

可视化应用：动态图表、丰富的现成的图表类型。

优点：丰富的图表选择、SVG、CANVAS、VML浏览器。

不足：客户端动态图生成会引发问题。

3、Flot

可视化应用：jQuery JavaScript绘图库。

优点：操作简单、定制、灵活。

不足：在展现不同效果时，难度会增加。

4、RaphaёL

可视化应用：在线输出图表、图形等。

优点：SVG/VML矢量输出格式，分辨率高。

不足：速度比画布创建栅格化图像慢。

5、D3(Data Driven Documents)

可视化应用：复杂的可视化图形。

优点：复杂的交互、展现效果好。

不足：不够简洁。

6、Visual.ly

可视化应用：信息可视化图形、信息图设计师的在线集市。

优点：大量的信息图模板。

不足：功能有一定限制。

7、Crossfilter是常用的GUI工具。

可视化应用：交互式GUI图形图表。

优点：方便快速查看、操作有交互性。

不足：操作复杂性增加。

8、Modest Maps

可视化应用：基本的地图功能。

优点：小型、拓展性好。

不足：基本形式非常有限。

9、Leaflet

可视化应用：移动端平面地图。

特点：小巧轻便、灵活、备份。

10、Polymaps

可视化应用：网络地图功能。

优点：强大的资源库、全方位信息可视化。

11、OpenLayers

可视化应用：地图库。

优点：强大的地图库，可靠性高。

不足：文档注释不完善，操作难度高。

12、Kartogragh

可视化应用：区域地图绘制。

优点：标记线、定义，更多的选择。

不足：处理世界范围的数据有一定的困难。

13、CartoDB

可视化应用：地图库。

优点：轻易结合表格数据与地图。

不足：需要按月付费。

14、Processing是一款适合于编程进阶的常用可视化工具。

可视化应用：开源的编程语言；

优点：语法简易，大量实例和代码。

15、R

可视化应用：分析大数据集的统计组件包。

优点：强大社区和组件库。

不足：复杂、学习难度大。

16、Weka

可视化应用：机器学习、数据挖掘。

优点：免费。

17、Gephi

可视化应用：社交图谱数据可视化。

其中，Gephi是一款开源的跨平台的基于JVM的复杂网络分析软件，可用作：链接分析、探索性数据分析、生物网络分析、社交网络分析。Gephi支持的数据输入格式包含：GEXF、GDF、GML、GraphML、Pajek NET、GraphViz DOT、CSV、UCINET DL、Tulip TPL、Netdraw VNA、Spreadsheet等。

ArcGIS产品线为用户提供一个可伸缩的，全面的GIS平台。具有强大的地图制作、空间数据管理、空间分析、空间信息整合、发布与共享的能力。

当然，Tableau、SAS、IBM、SAP、ORACLE、Qlik、Microsoft等是目前商业主流的数据可视化工具。

数据可视化面临的挑战

伴随着大数据时代的到来，数据可视化日益受到关注，可视化技术也日益成熟。然而，数据可视化仍存在许多问题，且面临着巨大的挑战。

大数据可视化存在以下问题：

1）视觉噪声。在数据集中，大多数数据具有极强的相关性，无法将其分离作为独立的对象显示。

2）信息丢失。减少可视数据集的方法可行，但会导致信息的丢失。

3）大型图像感知。数据可视化不单单受限于设备的长度比及分辨率，也受限于现实世界的感受。

4）高速图像变换。用户虽然能够观察数据，却不能对数据强度变化做出反应。

5）高性能要求。对于静态可视化对性能要求不高，因为可视化速度较低，性能要求不高，然而动态可视化对性能要求会比较高。

数据可视化面临的挑战主要指可视化分析过程中数据的呈现方式，包括可视化技术和信息可视化显示。目前，数据简约可视化研究中，高清晰显示、大屏幕显示、高可扩展数据投影、维度降解等技术都试着从不同角度解决这个难题。

可感知的交互的扩展性是大数据可视化面临的挑战之一。从大规模数据库中查询数据可能导致高延迟，使交互率降低。

在大数据应用程序中，大规模数据及高维数据使数据可视化变得十分困难。

在超大规模的数据可视化分析中，我们可以构建更大、更清晰的视觉显示设备，但是人类的敏锐度制约了大屏幕显示的有效性。

由于人和机器的限制，在可预见的未来，大数据的可视化问题会是一个重要的挑战。

数据可视化技术的发展方向

1）可视化技术与数据挖掘有着紧密的联系。数据可视化可以帮助人们洞察出数据背后隐藏的潜在信息，提高了数据挖掘的效率，因此，可视化与数据挖掘紧密结合是可视化研究的一个重要发展方向。

2）可视化技术与人机交互拥有着紧密的联系。实现用户与数据的交互，方便用户控制数据，更好地实现人机交互这是我们一直追求的目标。因此，可视化与人机交互相结合是可视化研究的一个重要发展方向。

3）可视化与大规模、高维度、非结构化数据有着紧密的联系。目前，我们身处于大数据时代，大数据时代，大规模、高纬度、非结构化数据层出不穷，要将这样的数据以可视化形式完美的展示出来，并非易事。因此，可视化与大规模、高维度、非结构化数据结合是可视化研究的一个重要发展方向。

## 8、大数据数据仓库实例