****

****

**数据库新技术课程报告**

|  |  |
| --- | --- |
| **论文题目** | 数据仓库技术报告 |
| **作者姓名** | 刘子凡 |
| **学 号** | DZ1533016 |
| **学科、专业名称** | 计算机科学与技术 |

**2018年5月18日**

# 目 录

[第一章 目 录 2](#_Toc516012870)

[第一章 数据仓库简介 3](#_Toc516012871)

[1.1 背景介绍 3](#_Toc516012872)

[1.2 诞生 3](#_Toc516012873)

[1.3 定义 4](#_Toc516012874)

[第二章 数据仓库体系结构与组成 6](#_Toc516012875)

[2.1 两层的体系结构 6](#_Toc516012876)

[2.2 三层的体系结构 8](#_Toc516012877)

[2.3 数据仓库的组成 9](#_Toc516012878)

[第三章 数据仓库数据模型 11](#_Toc516012879)

[3.1 元数据 11](#_Toc516012880)

[3.2 数据模型 12](#_Toc516012881)

[3.2.1 概念模型 13](#_Toc516012882)

[3.2.2 逻辑模型 13](#_Toc516012883)

[3.2.3 物理模型 13](#_Toc516012884)

[第四章 ETL过程 14](#_Toc516012885)

[4.1 主要流程 14](#_Toc516012886)

[4.2 数据抽取 15](#_Toc516012887)

[4.3 数据转换 17](#_Toc516012888)

[4.4 数据加载 17](#_Toc516012889)

[4.4.1 维表加载策略 18](#_Toc516012890)

[4.4.2 事实表加载策略 19](#_Toc516012891)

# 数据仓库简介

## 背景介绍

人类进入信息时代以来，特别是近20年来，数据规模日益扩大，数据量呈爆炸式增长。例如电商管理者需要从日积月累的交易信息中发现商品畅销的规律以及预测未来的趋势；医生需要从大量病人的电子病历中发现某些疾病的起因、症状；天文学家需要从天文望远镜获取的观测数据（其规模）中发现新的遥远天体及其运动规律等。这些数据的共同特点有：一是数据量巨大，一般都是TB级别以上；二是均以结构化的形式存储在数据库中，包含了大量潜在的、有价值的知识，有的已被发现，而有的还未被发现。这样的知识获取，仅仅依靠数据库管理系统的查询检索机制和统计分析方法，已经远远不能满足实际需求。为了有效地管理和利用这些海量数据、发现其中潜在的知识，数据仓库应运而生。

## 诞生

数据仓库是建立在传统事务型数据库基础之上，为企业决策支持系统以及数据挖掘系统提供数据源的技术。迄今为止，国外的数据仓库已经发展了二十几年，虽然国内起步较晚，但是发展十分迅速，目前已有众多的大型公司或企业正在建设或已经建设不同规模的数据仓库。

传统数据库和数据仓库的最根本区别在于其侧重点的不同。数据处理分为两大类：事务型处理，又称联机事务处理（Online Transaction Processing,OLTP）；分析型处理，又称联机分析处理（Online Analytical Processing,OLAP）。事务型处理以传统数据库为中心进行企业日常的业务处理；分析型处理以数据仓库为中心分析数据背后的关联和规律，为企业决策提供可靠、有效的科学依据。事务型处理和分析型处理的分离，划清了数据处理的分析型环境与事务型环境之间的界限，从而由原来以单一数据库为中心的数据环境，演变为以数据库为中心的事务处理系统和以数据仓库为基础的分析处理系统。企业的生产环境也从以数据库为中心演化为以数据库和数据仓库为中心。因此，在事务处理环境中直接构建分析处理应用是不合适的，要提高分析和决策的效率和有效性，分析性处理及其数据必须与事务型处理及其数据相分离，必须把分析型数据从事务型处理环境中提取出来，按照决策支持的需要重新组织，建立相对独立的分析处理环境，数据仓库正是为了构建这种新的分析处理环境而出现的一种数据存储、组织和处理技术。

## 定义

数据仓库的概念最早出现在20世纪80年代。1996年，被称为“数据仓库之父”的William H. Inmon系统地阐述了数据仓库的定义，即一个面向主题的、集成的、不可修改的且随时间变化的数据集合，用以支持管理人员的决策。

面向主题是相对于传统数据库的面向应用而言。所谓面向应用，是指系统实现过程中主要围绕着一些应用或功能，而面向主题则是考虑一个个的问题域，对问题域涉及的数据和分析数据所采用的功能给予同样的重视。数据仓库是面向在数据模型中已定义业务的主要主题域的，例如在“商场”这样一个领域中典型的主题域包括商品、供应商、顾客等。

集成是指数据仓库中的数据来自不同的数据源。由于历史的原因，个数据源的组织结构往往不同，在这些异构的数据导入到数据仓库之前，需经历一个集成过程。在数据仓库的所有特点中，这是最重要的。应用系统的设计人员经历多年制定出来的不同的设计策略有很多种不同的表示方法，在编码、命名习惯、属性和属性度量等方面往往是不一致的。例如“客户性别”的编码，在数据仓库中的编码为“男/女”还是“m/f”并不重要，重要的是无论是什么原始应用系统，在数据仓库中应该编码一致。如果应用系统中编码为“X/Y”，则在其导入数据仓库时就应进行转换。对所有应用都要考虑一致性，如命名习惯、键码结构、属性度量以及数据特点等。

不可修改是指与面向应用的书屋数据库需要对数据进行频繁的插入、更新操作不同，数据仓库中对数据的操作仅限于数据的初始导入和查询。操作型数据一般是一次访问和处理一条记录，也可以对操作型数据进行更新。但数据仓库中的数据则不然，通常是一起载入与访问，在数据仓库中并不进行一般意义上的数据更新。

随时间变化是指数据仓库以维的形式对数据进行组织，时间维是数据仓库中很重要的一个维度，并且数据仓库中的数据的时间跨度较大，从几年甚至到几十年，称为历史数据。数据仓库中数据随时间变化的特性表现在以下几个方面：

①数据仓库中数据的时间期限要远远长于操作型数据库中数据的时间期限。操作型数据库中数据的时间期限一般是60~90天，而数据仓库中数据的时间期限通常是5~10年。

②操作型数据库中含有“当前值”的数据，这些数据的准确性在访问时是有效的，同样当前值的数据可被更新。而数据仓库中的数据仅仅是一系列某一时刻生成的复杂快照。

③操作型数据的键码结构可能包含也可能不包含时间元素，如年、月和日，而数据仓库的键码结构总是包含某一时间元素。

数据仓库是企业决策系统的基础，因为在数据仓库中只有单一集成的数据源，并且数据是可以访问的，所以与传统数据库比，在数据仓库中分析决策人员的工作将容易很多。

# 数据仓库体系结构与组成

## 两层的体系结构

有数据仓库的定义可知，它是将企业各个业务系统中与分析有关的数据集成在一起，同时数据仓库面向的应用是分析型数据，因此形成了DB-DW两层数据仓库体系结构，如图2-1所示。



图2-1 DB-DW两层体系结构

其中，业务系统作为主要的分析数据来源，其数据格式主要是表的形式。实际中，由于要保证业务系统的正常运行，一般不直接在业务系统中进行数据的查询和抽取，而是采取备份库或者文件传输的方式进行数据仓库的数据抽取。外部数据源是指数据来源与企业的外部，描述企业运营的外部环境与企业经营分析有关的数据，如各个企业的市场份额等，外部数据作为经营分析的补充，对企业经营决策的正确性起着十分重要的作用。因此应保证外部数据的实时性和准确性。外部数据源具有多样性的特点，如年报等都可以作为外部数据源，同时外部数据源的格式也不统一，如文本、表格和图像等。因此对外部数据源及其数据格式等都要在数据仓库的元数据中进行记录，同时元数据中还应对外部数据的可信程度有一定的评价。

由于数据仓库中的数据源不统一，同时元数据的存储形式也不相同，因此有必要在数据进入数据仓库前先将数据存放在一个统一的暂存区中，引入数据暂存区的主要作用在于：

①统一不同数据源的数据格式，将不同数据源中不同的数据格式转换成同一的数据格式，供数据仓库统一处理。

②进行数据的初步检查，在数据进入数据仓库之前，先对数据进行初步检查，鉴于不影响数据仓库的处理时间，这里的检查将仅涉及比较粗略的数据检查，如记录数量、关键字段是否丢失等，对于错误的数据暂不导入数据仓库，这样对进入数据仓库的数据质量有一定的保证，但是更复杂的数据清洁工作，如字段格式的统一以及数据内容的清洗这种单一记录级的处理工作则应该在数据抽取中完成。

数据暂存区可以多种存储形式实现，如文件目录或者数据库表。

数据仓库中保存了大量的历史数据，同时数据仓库面向的是整个企业的分析应用，但在实际应用中不同部门的用户可能只使用其中的一部分数据，从处理速度和效率的角度出发，可以将这部分数据在逻辑或物理上进行分离，使用户无需到数据仓库的海量数据中查询，只在与本部门有关的数据子集上操作，这样就形成了数据集市（Data Mart）的概念，它是指面向企业中的某个部门（主题）在逻辑上或物理上划分出来的数据仓库的数据子集。将数据仓库按照数据的应用划分为多个数据集市，有利于数据仓库的负载均衡，保证应用的执行效率。同时，由于数据集市具有统一的数据来源——数据仓库，遵循统一的数据模型，保证了各个不同数据集市中的数据统一。

可以看出DB-DW两层的数据仓库体系结构是一种管道过滤器的结构，数据从数据源进入数据仓库到展示给最终用户，都有一定的关联关系，因此要保证数据仓库中数据处理的合理调度，则需要通过数据仓库的元数据完成。

## 三层的体系结构

数据仓库的提出使得操作型处理和分析型处理得以分离，从而形成了DB-DW两层的体系结构，但是在企业的业务处理中存在介于操作型和分析型之间的需求，需要对短期的历史数据进行分析，同时要求较快的响应速度，这种分析无法在操作型数据库中完成，因为其保存的是数据的瞬态信息，但是如果通过数据仓库完成，由于数据仓库保存了大量的历史数据，在响应时间上无法满足需求，因此提出了操作型数据存储（Operational Data Store，ODS）的概念，ODS数据可以概括为面向主题的、集成的、当前或接近当前的和不断变化的数据。其中，面向主题和集成的特点与数据仓库的概念相似；“当前”是指数据在存取时刻是最新的；“接近当前”是指存取的数据是最近一段时间得到的；而“不断变化的”是指ODS数据可以联机改变，包括增加、删除和更新等操作。



图2-2 DB-ODS-DW三层体系结构

面向主题和集成的特点使得ODS数据在部分特征上很接近数据仓库的数据，但是ODS和数据仓库之间也存在重要的差别，主要体现在：

①数据的内容不同。数据仓库中历史数据是指长期保存并可重复查询的数据，既保存细节数据，也保存综合数据，而且一般是静态数据（延迟>24小时）。而ODS一般只保存细节数据，而且ODS数据是可以更新的，即变化的，ODS中保存的历史数据也是近期的（延迟>1秒）。

②数据量不同。ODS保存的数据量要远远小于数据仓库的数据量。

③面向的应用不同。数据仓库用于长期的趋势分析或决策支持，而ODS主要支持企业的全局OLTP和即时决策分析应用。

引入ODS后，原来的DB-DW两层体系结构将被拓展为DB-ODS-DW三层体系结构，如图2-2所示。

在DB-ODS-DW三层的体系结构中，ODS的作用可以概括为：

①为数据仓库提供数据，减少数据仓库数据抽取的复杂性。由ODS的定义可知，它具有面向主题和集成两个特点，因此来自各业务系统的源数据首先进入ODS，在进入ODS时完成数据清洁和集成的工作，这样再向数据仓库提供的数据就是清洁和统一的，减轻了数据仓库中数据抽取的工作。

②即时的OLAP分析。由于在业务系统中需要对近期或当前的数据进行分析，如果该应用放在数据仓库中完成，由于数据仓库相应的处理环节较多，同时数据仓库保存了大量的历史数据，如果要完成这些需求势必造成留给数据仓库的数据处理时间减少，不利于数据仓库的建设，所以将这部分任务分配给ODS，由于ODS保存了近期的数据，可以完成用户的即时分析需求。

③全局的OLTP操作。由于ODS数据的集成性，整合了企业中不同业务系统的数据，同时ODS数据是可更新的，因此ODS可以提供面向企业全局的OLTP操作。

## 数据仓库的组成

数据仓库的大小一般都在TB级以上。传统的关系型数据库是针对OLTP的，并不适用于数据量大且复杂度高的数据仓库。因此，数据仓库系统必须能够满足：

①扩充数据仓库的数据。

②每日对数据仓库系统的管理和维护。

③允许用户增加需求。

要充分满足上述三点并不容易，尤其是数据仓库最终以自动的数据分析处理为目标。扩充数据仓库数据是非常重要的，其重点是抽取、整理并转换数据以及以适当的方式展现给用户分析使用。数据仓库并不是一个只读系统，虽然说事实数据导入数据仓库之后就不会再更新，但是如果用户需求改变，例如希望以不同的方式浏览相同的分析结果，则仍然会修订索引数据。每日对数据仓库系统的管理和维护与传统的OLTP系统完全不同，因为数据仓库的数据量远比OLTP系统大得多，所以需要更积极的管理方式。例如添加或删除数据、将数据仓库数据存入备存介质、由备存介质加载数据等。因此可以这样理解，数据仓库是一个持续更新的系统，以满足用户新的决策需求。允许用户增加需求的能力似乎是设计数据仓库系统时最困难的工作，因为每个用户都有各种不同的需求。数据仓库除了能够允许更新现有的需求之外，还应该可以增加新的分析主题。

数据仓库系统应该具有以下功能：

①抽取数据与加载数据。

②整理并转换数据为数据仓库适用的格式。

③备份与备存数据。

④管理所有的查询，将他们导向适当的数据源。

具体来说，数据仓库系统的管理器可细分为三种，即：

①加载管理器，负责抽取并加载数据，在加载数据之前与过程中执行简单地转换。

②仓库管理器，负责转换并管理数据仓库数据、备份与备存数据。

③查询管理器，负责引导并管理数据仓库的查询。

# 数据仓库数据模型

## 元数据

元数据（Metadata）是关于数据的数据。在数据仓库系统中，元数据可以帮助数据仓库管理员和数据仓库开发人员非常方便地找到他们所需的数据；元数据是描述数据仓库中数据结构和构建方法的数据。

随着计算机技术的应用日益广泛，元数据得到人们越来越多的关注，这是由多方面的需求决定的。

其一是管理数据的需求。当系统数据量越来越大时，检索和使用这些数据的效率就会降低，通过存储关于系统和数据的内容、组织、特性等细节可以帮助有效地管理，从而提高效率。

其二是系统分布、互通和重用的要求。目前信息系统一个共同的趋势是信息共享，要实现异构系统的信息共享，则需要描述数据语义以及软件开发过程的元数据，而且这些元数据必须标准化，以充分实现分布、互通和重用。

其三是元数据重用、综合的需求。目前，很少有单一工具能满足大型商业应用的需求，用户常常需要使用多种工具的组合，不同工具之间数据交换的途径之一就是通过标准的元数据。

在过去十年里，元数据的概念在现实中大量使用，又是为了支持信息检索，又是为了软件配置，有时为了不同系统之间的数据交互。对于不同的领域专家，元数据有着不同的应用，但至少有两点是共同的，即元数据是对数据的描述；元数据的存在是为了更有效地使用数据。

对于元数据，根据观察角度的不同，可以划分为不同的类别。

按照与特定领域是否相关可划分为：

①与特定领域相关的元数据——描述数据在此特定领域内的公共属性。

②与特定领域无关的元数据——描述所有数据的公共属性。

③与模型相关的元数据——描述信息和元信息建模过程的数据。此类元数据又可分为两类：（1）横向模型关联元数据，综合现有的两个或多个信息模型，例如两个不同数据库之间的交互、从多个数据源中提取数据时，就需要这种横向模型关联元数据。（2）纵向模型关联元数据，即模型信息层与元信息层之间的关联元数据，上下层之间的对应关联。

④其他元数据，例如系统硬件、软件描述和系统配置描述等。

按照元数据的应用场合可划分为：

①数据元数据，又称为信息系统元数据。信息系统使用元数据描述信息源，以按照用户需求检索、存取和理解源信息。因此，数据元数据保证了在新的应用环境中使用信息，支持了整个信息结构的演进。

②过程元数据，又称为软件结构元数据。过程元数据是关于应用系统的信息，它帮助用户查找、评估、存取和管理数据。大型软件结构中包括描述各个组件接口、功能和依赖关系的元数据，这些元数据保证了软件组件的灵活、动态配置。

按照元数据的具体内容可划分为：

①内容（Content）——识别、定义、描述基本数据元素，包括数据单元、合法值域等。

②结构（Structure）——在相关范围内定义数据元素和逻辑概念集合。

③表示（Representation）——描述每个值域（多位技术相关）的物理表示，以及数据元素集合的物理存储结构。

④文法（Context）——提供基础数据的族系和属性评估，包括所有与基础数据的收集、处理和使用相关的信息。

此外，按照用途的不同可划分为技术元数据和业务元数据两大类。技术元数据存储关于数据仓库系统技术细节的数据，是用于开发和管理数据仓库的数据，它保证了数据仓库系统的正常运行；业务元数据是从业务角度描述数据仓库中的数据，它提供介于使用者和实际系统之间的语义层，使得数据仓库使用人员能够“读懂”数据仓库中的数据。

## 数据模型

数据模型是对现实世界的一种抽象，根据抽象程度的不同，形成了不同抽象层次上的数据模型。与关系数据库的数据模型类似，数据仓库的数据模型也分为如下三个层次。

### 概念模型

概念模型是客观世界到计算机系统的一个中间层次，最常用的表示方法是E-R图。目前，数据仓库一般建立在数据库基础上，所以其概念模型与一般关系型数据库采用的概念模型一致。

### 逻辑模型

逻辑模型是数据的逻辑结构，如多为模型、关系模型和层次模型等。数据仓库的逻辑模型描述了数据仓库主题的逻辑实现，即每个主题对应的模式定义。

目前，对数据仓库数据模型的讨论大多集中在逻辑模型，其中对常用的是多维模型。在多维模型中，涉及以下一些基本概念：

①维是指人们观察数据的特定角度。例如企业常常关心不同销售数据随时间的变化情况，所以时间就是一个维度。

②维度层次是指人们观察数据的某个特定角度还可以存在细节程度不同的多个描述。一个维度往往有多个层次，例如描述时间维度时，可以有年、季度、月和日等不同层次，年、季度、月和日就是时间维度的层次。

③维成员是指维的一个取值。如果一个维是多层次的，则该维度的成员就是在不同层次上取值的组合。例如时间维有年、月和日三个层次，则分别在三个层次上各取一个值组合起来就等到了时间维度的一个成员，即“某年某月某日”。

④度量描述了要分析的数值，例如销售额等。

⑤粒度是指数据仓库所保存数据的细化或综合程度的区别。细化程度越高，粒度越小；反之，细化程度越低，粒度越大。

### 物理模型

物理模型是逻辑模型的具体实现，如物理存取方式、数据存储结构、数据存放位置以及存储分配等。在设计数据仓库的物理模型时，需要考虑一些提高性能的技术，如表分区和建立索引等。

# ETL过程

业务系统的数据经过抽取、转换并加载到数据仓库所在的中心存储库的过程称为ETL（Extract，Transform and Load）过程，制定这一过程的策略称为ETL策略，而完成ETL过程的工具则是ETL工具。相对于数据仓库的表而言，业务系统数据库中的表称为源表，业务系统数据库称为源数据库，数据仓库中绝大部分数据都来自用于业务系统数据库。在构建数据仓库的过程中，ETL的实施是一项烦琐、冗长而艰巨的任务，因为它关系到数据仓库的数据质量问题。如果导入的数据漏洞百出，那么对于决策者而言无疑会造成困扰。

## 主要流程

随着应用和系统环境的不同，数据的抽取、转换和加载具有不同的特点。

（1）预处理是正式开始作业之前的准备工作，包括清空工作区、检查过渡、准备区。在需要直接访问操作型数据源系统时，应检查远程数据库服务器状态，并核对目标数据加载状态，以核算出加载作业的参数，如加载数据的时间间隔和范围（是24小时的数据，还是前3天的数据）。

（2）启动数据抽取的批处理，执行数据提取、清洁和转换等过程。

（3）因为维表有事实表所参照的主键，所以需要先完成对维表的加载，生成维表主键，并作为以后加载事实表所需的外键。在加载维表时，有时需要处理好缓慢变化的维，并可能涉及版本号的处理问题。

（4）加载事实表。这中间也涉及键查找的问题，即从有关维表中找到相应的主键，并以此作为事实表的外键。

（5）事实表加载完成后，再对数据立方体体系进行刷新，以保障数据立方体与其基础数据同步。

（6）设计具有完善的出错处理机制和作业控制日志系统，以监测和协调整个加载过程。

## 数据抽取

数据抽取是ETL处理的首要任务，解决的主要问题是确定需要抽取的数据，并采用适当的方式抽取。

源数据进入数据仓库是通过数据抽取完成的，从一个或多个源数据库中通过记录选取进行数据复制的过程。抽取过程将记录写入ODS或临时存储区以备进一步处理。

数据抽取的主要功能包括：

①数据提取。主要是确定要导入数据仓库的数据。

②数据清洁。检查数据源中存在矛盾的数据，按照用户确认的清洁规则对数据进行修改。

③数据转换。主要是将数据源的数据转换成数据仓库要求的格式，其中包括数据格式的转换，例如将数据源中的日期字段转换成数据仓库要求的字符形式；数据内容的转换主要是将同一含义的字段用统一的形式表达；数据模式的转换是由于数据仓库系统和业务系统面向的数据操作不同，所以在数据模式上也存在不同，例如电信业务的出账表的主键包括用户标识和费用项，但是在数据仓库中用户主体的用户账务信息采用用户标识作为主键，将不同费用项的费用作为字段，这样就需要在数据抽取时进行不同数据模式的转换。

④生成衍生数据。由于数据仓库保存了大量的历史数据同时需要保证查询的效率，需要对用户经常进行的查询通过预处理操作来提高查询效率，生成衍生数据。衍生数据既包括某些数值数据的预运算，例如平均值和汇总等，也包括某些分类字段的生成，例如对用户费用的分档信息等。

数据抽取的重要组成部分是变化数据捕获（Change Data Capture，CDC）。实现CDC的方法包括时间戳、读取RDBMS系统的日志文件、使用源系统中的触发器或者自行开发CDC程序来检查日志文件等。时间戳是最简单的，也是相当普遍的方法，前提是所有的源表都有时间戳。例如超市的业务系统数据库中几乎每个表都有“插入记录日期”和“更新记录日期”两个字段，这是规范建库的一个基本标准。

在多数情况下，数据源与数据仓库并不处于同一个数据服务器中，它们往往是独立的，并处在远程系统中。数据抽取往往是以远程、分布式的方式进行，并涉及各种各样的方法和手段，其中包括：

①应用SQL \* PLUS提取到文本文件。

②应用OCI或Pro \* C程序，或者Oracle UTIL\_FILE提取到文本文件。

③应用Oracle Export Utility程序提取到Oracle Export Files文件。

④远程数据复制。

⑤信息流。

如果来自操作型数据源的数据含有不洁的成分和不规范的格式，将对数据仓库的构建和维护，特别是对联机分析处理带来很多问题和麻烦。因此，必须在ETL中加以解决，通常包括以下的处理方法：

①拼写检查、分类并与标准值对照检查。

②处理名字和地址。

③为名字和地址建立辅助表或联机字典，据此进行检查修正。

④数据转换函数以及子程序库。

保障数据清洁顺利进行的原则是优先对数据清洁流程进行分析和系统化的设计，针对数据的主要问题和特征，设计一系列数据对照表和数据清洁程序库的有效组合，以便应对不断变化的、形形色色的数据清洁问题。

数据清洁通常包括：

①预处理——对于新的数据加载文件（特别是新的文件和数据集）要进行预先诊断和检测，不能贸然加载。有时需要临时编写判断小程序。

②标准化处理——应用数据仓库内部的标准字典，对地名、人名、公司名、产品名、品类名进行标准化处理。

③查重——应用各种数据查询手段，避免引入重复数据。

④出错处理和修正——将出错的记录和数据写入日志文件，留待进一步处理。

数据仓库必须存放“优质数据”，即符合一致性的、大家公认或经过验证是有价值的，并符合元数据定义的。通过数据清洁能够检测出那些违反规则的数据，一旦发现要么抛弃，要么将其转换成“清洁”数据，使其符合规则，然后再装载到数据仓库。

## 数据转换

数据转换是将抽取出的数据进行过滤、合并、解码和翻译等，为数据仓库创建有效数据的过程。一旦数据抽取完成，就需要设计并确定转换规则应用于已抽取出的数据。数据转换需要理解业务侧重点、信息需求和目前可用的数据源。

常用的转换规则包括：

①字段级的转换，主要是指数据类型的转换；增加“上下文”数据，如时间戳；解码，即将数值型的地域编码替换成地域名称等。

②清洁和净化，主要是保留字段具有特定值或特定范围的记录；引用完整性检查；去除重复记录等。

③多数据源整合，主要是字段映射；代码变换，即将不同数据源中的数据值标准化为数据仓库数据值。例如，将原系统非英文编码转换成数据仓库英文编码；将原系统信息编码转换成数据仓库信息编码等；合并，即将两个或更多源系统记录合并为一个输出或“目标”记录；派生，即根据源数据，利用数学公式产生数据仓库需要的数据。例如，由身份证号码计算出出生日期、性别、年龄等。

④聚合和汇总。

虽然数据转换较为繁琐，但却是ETL三个步骤中最简单的。许多ETL工具都提供了强大的转换功能，例如DTS中有复制字段转换、小写字符串转换，大写字符串转换、中间字符串转换、剪裁字符串转换、日期/时间字符串转换、读取文件转换、写入文件转换和ActiveX脚本转换等，其中最常用的是ActiveX脚本转换，因为它允许自行撰写VBScript或Jscript将原始字段中的数据转换至目标字段中的数据。事实上，如果上述的各种转换类型都无法满足需求时，则ActiveX脚本转换将是最终的选择。

## 数据加载

数据加载是将转换/清洁后的数据装载到数据仓库，实现数据加载可选用的实用程序和工具很多，例如最基本的Import、SQL Loader和SQL语言等。为了提高程序和过程的复用性，编写和设计数据加载的函数库/子程序库是十分必要的。

数据加载包括维表和事实表的加载，两者具有不同的加载策略。

### 维表加载策略

本质上，有三种维表加载策略。每种策略按不同的方式处理维表中的变化以及更新维表数据或捕获属性的变化历史。这三种加载策略统称为慢速变化维表策略（Slowly Changing Dimension，SCD）。在这三种策略中，所有输入数据都与现存的数据进行比较，如果在自然键上没有发现匹配的记录，那么输入的记录就被插入维表中。自然键由维表中的一些列（数据项）组成，能唯一确定维表中一条记录的代理键不包括在这些列中。

（1）慢速变化维表类型1（SCD-1）

在SCD-1策略中不需要保存历史记录。如果一条输入记录已经在目标维表中（根据自然键值进行判断），则可根据输入记录的数据对该记录进行更新。

（2）慢速变化维表类型2（SCD-2）

有时记录中有一些重要值（即维表中一个或多个列组成的自然键值）需要保留。在SCD-2策略中，当相关事实发生时，就可保存一条维表记录。因此，如果一条输入记录中的某个字段或列值属于“重要值”，并且它与目标表中相应的列值不同，则现存的记录就已经“过期”了，则需要根据输入的记录，在维表中插入一条新纪录并指派一个新的代理键值。如果输入记录中没有一个重要值与目标表中相应的列有所差别，则对现存记录进行更新，而不是使它过期。

（3）慢速变化维表类型3（SCD-3）

SCD-3与SCD-2非常相似，用于跟踪重要值的变化，然而这种策略不是为每个变化添加一条不同的记录，而是在现存的记录上使用不同的列来保存当前值和任意的前*n*个值。当探查到一个重要值发生变化时，该字段所有以前的列需要向下一列移动，第*n*个以前的值将被丢弃。但是这种策略不建议使用，因为这将加大ETL难度。

所有维表加载都遵循上述三种策略，但有两个维表例外，即静态维表和完全由新数据替换的维表。静态维表不会变化，如果它发生变化，则发生的唯一变化是在其中添加一条新纪录，时间维表就是一个静态维表；完全由新数据替换的维表通常是小的代码列表，它们不需要代理键或变化历史。但需要注意的是表中任何重要值的丢失都是不可承受的。

### 事实表加载策略

事实表的加载是必须的也是非常重要的，他是后续数据分析的基础。事实表的加载就是持续不断地增加数据。事实表的加载不是简单的数据复制，必须首先将每个事实表与各自的维表的代理键相结合，每个源表（与事实表关联的表）必须有足够的信息用于查找维表中的自然键，以验证事实记录的完整性。

在数据加载过程中，经常涉及逐渐查找的问题。主要是对某些键查找函数程序进行修改补充，原因是这些键查找程序需要异常控制以返回两种不同的结果（找到的键值或返回空值）。

加载维表的过程中，在插入新的记录或修改已有记录之前，需要比较某些逻辑键来确定当前记录是否存在。在加载事实表的过程中，经常涉及更多的主键查找处理。数据仓库的事实表含有许多外键，并以此与有关维表的主键相关联。在进行事实表加载时，往往需要查找有关维表的主键值，并以此确定事实表的外键值。常用的方法是，通过从源数据提取的数据部分的逻辑键与数据仓库内有关维表的逻辑键进行比较和匹配，如果匹配，则取维表的主键值，并以此作为事实表的外键值加载到事实表中。例如对客户维度而言，可以通过客户的姓名、邮政编码与客户维度进行比较，如果匹配，则以客户标志符键作为事实表的外键。

逻辑上，实现ETL主要采用三种策略，即远程抽取（Remote Extraction）、推（Push）和拉（Pull）。其中，远程抽取是指ETL过程是在一个独立的远程平台上进行，这种方式受数据源现有系统或目标数据仓库的结构影响最小，几乎不影响源和目标系统的CPU和容量，但可能会提高成本，对网络带宽要求较高。推是指在现有系统环境中开展ETL过程。当现有系统具有相当充裕的容量，而且不会超出CPU处理限度时，经常采用这种方法。当现有系统时同构的，也经常采用这种方式。拉是指ETL过程在“仓库”端进行，当现有系统是异构时，经常采用这种方式。当“仓库”端平台具备必须的性能和容量时，将体现出一定优势。但是由于“仓库”的不断增长，可能需要制定长期策略。

# OLAP

## 定义

OLAP是基于数据仓库的一种数据分析和处理技术，也可以看成是基于数据仓库的一种软件工具。OLAP侧重于对决策者和高层人员的支持，可以根据分析人员的要求，快速、灵活地实现大量数据的复杂查询，并以一种简单、只管的形式展现查询结果。基于数据仓库实施OLAP，可以帮助企业管理者掌握企业经营状况，了解市场需求，制定科学决策，提高核心竞争力。OLAP的目标是满足决策支持或满足在多维环境下特定的查询和报表需求。

OLAP最早是由关系数据库之父E. F. Codd于1993年提出的。当时，Codd认为OLTP已不能满足终端客户对数据库查询分析的需要，SQL对大型数据库进行的简单查询也不能满足客户分析的需求。客户的决策分析需要对关系数据库进行大量计算才能完成，而查询的结果并不能满足决策者提出的需求。因此Codd提出多维数据库和多维分析的概念,即OLAP。

参考文献

**Chaudhuri, Surajit and Dayal, Umeshwar. 1997.** An overview of data warehousing and OLAP technology. *Acm Sigmod Record.* 1997, Vol. 26(1), pp. 65-74.

**Mathur, Nikhita. 2016.** An Overview of Data Warehouse and OLAP Technology Used in Banking System. *International Journal of Emerging Trends & Technology in Computer Science.* 2016, Vol. 33(2), pp. 76-79.

**郑岩. 2015.** 数据仓库与数据挖掘原理及应用. 北京 : 清华大学出版社, 2015. ISBN 978-7-302-37861-7.