***3.3*** ***ETL***

数据由各种系统数据库、文件、网页等存储方式进入数据仓库需要经过抽取



**商** **业** **智** **能** **在** **电** **子** **商** **务** **中** **的** **实** **践** **与** **应** **用**

(Extraction) 、转换(Transform) 和装载(Load) 这样一个过程，简称 ETL, 这个过程 非常具有挑战性，但是只要能认识ETL 的本质和核心问题，还是能把 ETL 简单 完成。

焦点科技的 ETL 完全基于数据库存储过程进行开发，在这个过程中非常重视 完整的 ETL 设计思想、方案、架构及技术标准，下面我们从各个角度来介绍焦点科 技BI 人员对ETL 的设计及开发的认识，从中大家或许可以领略一点技术的魅力。

**3.3.1** **ETL** **目标和基本要求**

简单点理解 ETL 的目标就是保证数据仓库、数据集市数据的及时性和准确 性，从而保证我们各种分析需求的数据要求。基本要求包括：

(1)数据完全准确。数据完全准确是 ETL 的第一要求，数据不准的系统没人 敢用，不准的数据甚至会带来巨大危害，毕竟高层领导的分析和战略决策都需要 BI 数据来支撑。

(2)更新及时。及时更新数据会给 BI 带来很好的用户体验，但不太现实，因 为 ETL 运算量过于庞大，所有 ETL 都实施或者频繁更新难于实现，但是对于实 时性要求比较高的分析，提高更新频率是有必要的；对不同需求有不同的更新策 略就非常有必要了。

22

(3)系统稳定。区别一般系统稳定性这里有两层含义，除了服务不能停之外 还要求 ETL 在更新数据的时候不会经常中断，如果中断也能快速恢复。

(4)对其他系统影响小。虽然每天要抽取的数据很多，但是任何系统都不能 容忍对自身影响太大的请求，快速从业务系统获取数据是 ETL 第一步要解决的 问题。

当然如果能做到如下两条，那更为完美，焦点科技这方面已经达到。

(5)代码简单并且能够标准化。

(6)维护简单。

**3.3.2** **ETL 的开发**

**1)企业级** **ETL开发基本思路**

· ETL 包的分解：企业级 ETL 开发首先要做的事情是 BI 项目和企业业务 主题规划，然后按不同主题来划分ETL 包，最后得到分主题的设计ETL 包，包之 间数据可以单向依赖，不能互相依赖，这样我们在后面就可以按照依赖关系进行 调度。

· 分包分项目开发 ETL: 根据项目进度我们安排包的开发，先上项目先开 发，后上后开发，如果项目为同一主题，我们可以共用一个包，这样在数据统一性方

**第** **三** **章** **商** **业** **智** **能** **在** **电** **子** **商** **务** **的** **解** **决** **方** **案**

面会更好。

· ETL 更新顺序：根据表之间的数据传递关系，建立起表之间的依赖关系， 根据表之间的依赖关系建立起包之间的依赖关系，如表 A 有数据来源于表 B,那么 我们认为表 A依赖表 B, 对包同理。原则上表和表、包和包之间不互相依赖。

· ETL 中断纠正机制：ETI, 中断是商业智能系统常见的问题，建立起简单有 效的 ETL 中断处理机制是必不可少的工作之一。

· ETL 统一调度：确定更新时间、更新频率、更新对象。

**2)ETL 的挑战及应对思路**

· ETL 技术架构：技术架构只有合不合适，没有说哪个最好，前面我们在系 统架构方面基本了解到现在我们针对电子商务行业采用了 ODS 、DW 、DM 这样 三层架构，不管什么工具都要符合这样三层架构思想进行设计，这样我们在架构层 面就有明确的目标定位，相当于统一了思想，从而为后面的整体设计，分步、分工实 施提供了可能。系统架构也是我们后面制定技术标准的大框架。

· ETL 更新机制：ETL 开发除了数据质量之外，最具挑战的就是数据更新机 制的设计，ODS 、DW 、DM的每个表都面临数据怎么更新的问题。只有最合适，没 有说哪个方法最优，不管哪个层级的表数据更新策略可以简单分为：

I. 全量更新，每次把表数据清空，重新生成，对 ETL 来讲小表我们可以每次 全量更新，而对于每天增量数据较多的表则不能全量更新，只能利用增量 更新办法。

23

Ⅱ.增量更新，其中增量更新分完全增量更新和相对增量更新，完全增量更新 是指加载到目标表的数据和目标表原数据既不出现重合也不出现缝隙， 这是最理想的一种更新方式，但是因为增量判断机制复杂，增量方式众多 (新增、更新、删除),每种增量方式的增量算法完全不一样，开发难度巨 大，特别是数据源如果不稳定，那将出现致命性错误，甚至导致数据仓库 必须重新初始化这样大的事件；相比较相对增量更新可以允许数据有一 定重合度，通过关键字等判断自动删除重合部分再简单加载新数据，不用 面临一个指标更新要几个算法应付各种更新机制的问题，这里只需要一 种更新算法就可以，设计简单，性能也不会有太大牺牲，如果大数据引入 分区技术，每次通过移除分区来删除重复数据那性能几乎没有损失。

· 异常中断纠正机制：很多企业在开发 BI 之初，面临 ETL 中断之后是重新 跑数据还是哪里断了哪里重启这样一个问题。 一般来讲重新跑风险小，但是时间 上要允许，而断点处开始执行则面临人工干预的问题，即使再有经验的人操作这个 风险也始终存在。事实上 ETL 开发到一定程度可以用点时间来解决该问题，现在 焦点科技通过合理设计在效果上实现了融合两者优点的方法：



**商** **业** **智** **能** **在** **电** **子** **商** **务** **中** **的** **实** **践** **与** **应** **用**

24

I. 建立表之间依赖关系。

Ⅱ.任何报错只影响对该 ETL 对象表有依赖关系的表，受影响的表不执行， 节省时间，因为跑了也是白跑，不受影响的表继续往下执行。

Ⅲ.跑完之后系统自我检查是否有报错，有报错再次重跑，重跑过程中系统自 动调度已发生错误的 ETL 进行重跑，其他成功的表不再重跑，依赖于该 ETL 的表其 ETL 都需要重跑。

IN. 如果再次重跑还是失败那么通过邮件预警到相关人员，检查错误排除错 误直到解决后再次重跑。

· 大数据的 ETL 技术：解决大数据，是电子商务行业必须面临的问题，现在 主流的办法还是分布式计算，其中 Hadoop 最为常用，当然 EMC 的 Greenplum 更 为成熟，其费用也不菲，有条件的建议大家应用后者。

· 数据质量：我们可以认为BI 产生的数据和业务部门需要的数据不一致问 题都是数据质量问题，所以在开发 ETL 过程中处处是风险：

I. 首先是需求问题，BI需求最大难点是难于统一各个独立业务部门的指标 定义，除非自上而下强行统一，但一般来讲 BI 实施之初不可能有这种力 度，比较可行的办法还是在数据仓库放基础数据，面向不同业务部门设计 独立的集市来满足其需求，当BI 在各个业务部门应用到一定程度，他们

会自动统一这些定义和需求，否则他们将无法沟通，当然 BI 团队可以加 快这个进程。

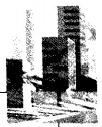
Ⅱ.其次就是架构问题，在数据仓库层面要放基础数据，基础数据不要有冗 余， 一个字段在多个表出现将会导致数据难于统一，这是最大忌讳，为了 提升仓库到集市的数据统一性，我们可以在仓库层生成结构性的中间表， 让大部分基础运算都统一到这层，这样为仓库进集市的数据统一性又降 低了难度。集市设计最好是分主题，在同一主题下，数据定义要求完全一 致，表尽量共用设计，这样不管是哪个项目在同一体系内数据都是一致 的，另外就是不同定义的数据哪怕在不同主题间也要区别命名。

Ⅲ.最直接的数据质量问题还是 ETL 开发和测试，ETL 测试不仅要看数据总 量，必要时候还得交叉维度和层级进行数据测试，简单测试指标包括汇 总、计数等。

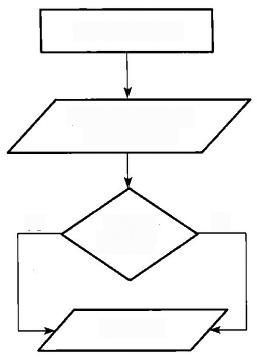
IN. 系统维护要能自动预警数据质量问题，这样可以让相关人员在第一时间 发现问题、解决问题。

**3)电子商务行业可采用的** **ETL** **开发**

一般的设计数据仓库都可分为 ODS 、DW 、DM三层，下面我们来看一下这三 层我们是怎么做的。



**ODS** **全量加载：**



清空ODS 目标表

将数据源的数据全 量插入目标表

插入是

否成功?

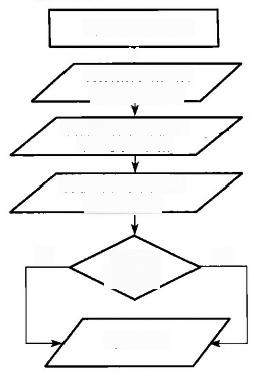
写入日志

否

是

第 三 章 商 业 智 能 在 电 子 商 务 的 解 决 方 案

(1)ODS 层全量加载、增量加载 ETL 开发流程(图3.8)



ODS 增量加载：

清空ODS临时表

王

将数据源增量数 据插入临时表

删除目标表与临时 表重复的数据

将临时表数据全量

插入目标表

是 插入是 否成功?

写入日志

否

**图3.8** **ODS\_ETL 开发流程图**

· 全量加载将清空原有数据，重新载入最新的源数据。

· 增量加载每次只更新历史数据中有更新的记录，并且载入历史数据中没有 的新数据。

25

(2)DW 层 ETL 开发

· 事实表先更新，再更新维表。

· DW 层事实表全量加载 ETL 开发：与 ODS 层全量加载 ETL 开发方法类 似，在此不再重回复。

· DW层事实表增量加载 ETL 开发：与 ODS 层增量加载ETL 开发方法基 本一致，但是需要把相关维表字段关联修改为维表的代理关键字。

· DW层维表增量加载 ETL 开发如图3.9所示。

(3)DM层

· DM层全量加载 ETL 开发：与ODS 层全量加载 ETL 开发方法类似，在此 不再重复。

· DM 层 增 量 加 载 ETL 开 发 如 图 3 . 1 0 所 示 。

**3.3.3** **ETL** **的测试**

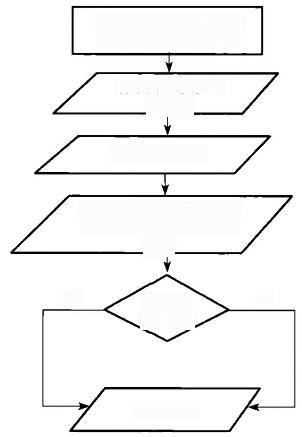
ETL 数据测试采用黑盒测试[8],包括 ETL 常规检查和业务逻辑测试。

ETL 常规检查包括①ETL 脚本是否有运行错误；②ETL 编码是否符合 ETL 技术规范。

ETL 业务逻辑测试是 ETL 测试的重点，主要包括：①业务逻辑测试：指标设

**商** **业** **智** **能** **在** **电** **子** **商** **务** **中** **的** **实** **践** **与** **应** **用**

DW 维表增量加载：



将增量数据按照历史数 据的ROWNUM 排序

将增量数据插入

临时表

清空DW 维表

将ODS 源数据表与临时 表关联，将数据插入维

表中

插入是

否成功?

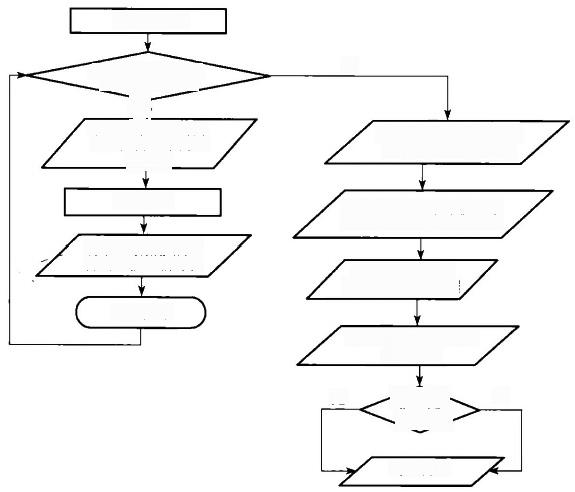
写入日志

是

否

**图3** **.9** **DW\_ 维表增量加载** **ETL 开发流程图**

26



否

取出数据表中时间截范 围内的不同业务时间

将临时表1的数据按照维 度字段汇总到临时表2中

删除目标表与临

时表2重复的数据

将临时表2.的数据插

入目标表

插 入 是

香成功?

写入日志

清空临时表

是否还有指标取百 不同的DW 表 ?

是

取出数据表中时间

戳范围内的不同业

务时间

循环处理

将增量数据按照业 务时间插入临时表

结束循环

是

否

**图3** **.** **10** **DM 层增量加载** **ETL 开发流程图**



**第三章** **商业智能在电子商务的解决方案**

计是否符合业务逻辑；②数据量测试：目标表数据量是否与数据源数据量一致； ③唯一性检查：检查 ETL 重复调度是否会出现重复数据；④性能测试：测试 ETL 的效率是否在可接受的范围内。

ETL 的测试流程包括：理解业务逻辑，开发测试用例，编写测试文档，通知开 发人员修改 BUG,BUG 验证等。