

基于 ORACLE 和 Notes 的大型商业企业管理信息系统的建设

浙江大学计算机系 杭州 胡朝晖 陈奇 俞瑞钊

摘要：企业的信息系统建设已经从独立的一个个子系统转向于建立集成的管理信息系统，所有的业务流程之间能够进行信息共享，结构化数据和非结构化数据能够进行很好的共享。本文根据实际的系统开发的情况，分析了大型的商业管理信息系统的软硬件结构，并且对系统建设中需要注意的问题给出了解决方案。

本系统是针对一家大型的商贸企业而开发的一套集业务管理、财务管理和办公自动化于一体的商业企业管理信息系统。该企业有一个业务运营中心、两个大型的集中式的仓储中心和十几家的连锁店和商场。其中商场和连锁店分布于浙江省各县市区。

系统的构成

1. 系统的硬件配置

在系统的硬件配置方面，服务器采用了两台 DEC 5000，并采用了 DEC 的 Cluster For Windows NT 双机集群结构。主要的应用业务数据都放在机柜上，包括 ORACLE 数据和 Notes 的数据。当一台服务器由于未知原因而 Down 机的时候，能在几分钟内把所有的相关的应用自动切换到另一台服务器上，采用的是 IP Switch 技术。两台服务器共享一个虚拟的 IP 地址。所以对终端用户而言，只要重新进入菜单就可以再次连接系统，使由于服务器 Down 机而对业务的影响减低到最小。由于两个仓储中心和业务运营中心之间的长度超过了双绞线所能连接的范围，用了 FDDI 技术把仓储中心和运营中心连接起来。业务运营中心和商场及连锁店之间采用了 Modem 拨号到业务运营中心的路由器(Cisco)的方法，每天批量的上传和下载数据。业务运营中心内部通过 Switch Hub(Bay) 进行连接。硬件的拓扑结构如图 1 所示：

2. 系统的软件配置和主要模块

在软件配置方面，我们采用了 Windows NT 作为服务器，Window9.X 作为客户端，数据库是 Oracle 公司的 8.0.4 For NT。业务流转和办公自动化则通过 Lotus 公司的 Domino Server 4.6，并采用了基于 Lotus Notes 的自动化的业务流转开发工具 Jetflow。开发工具采用了微软的快速开发工具 VB 6.0。

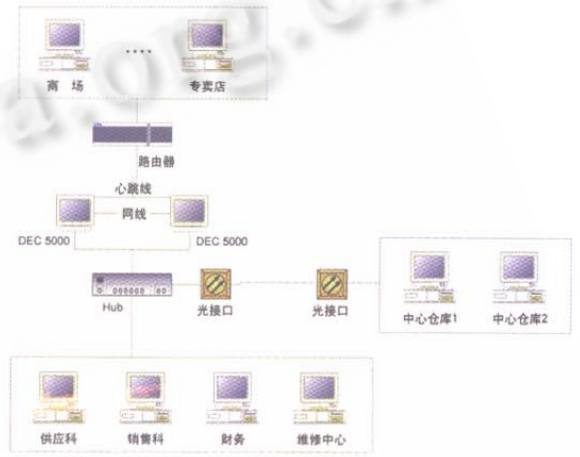


图 1

系统包括了进销存，财务和办公自动化等模块，由于业务流转是采用了 Notes 的工作流的机制，而 Notes 的数据库是非结构化的，非关系的数据库，所以在流转过程中所产生的数据是通过 Notes 提供的 ODBC 的接口和 Oracle 数据库进行通信，并把数据存储到 Oracle 数据库中。我们通过 Notes 进行方便的业务流转，通过 Oracle 数据库存储结构化数据以便查询和分析。软件的业务模块如图 2 所示。

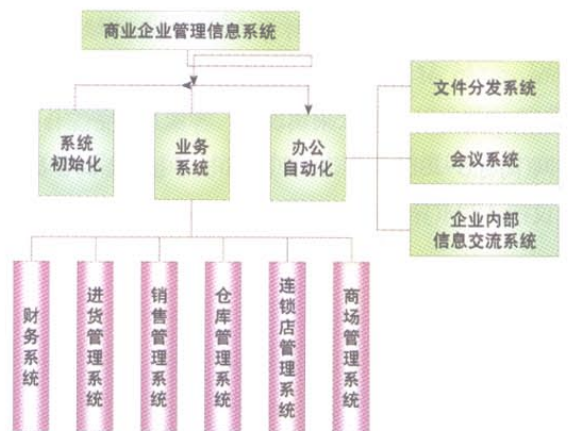


图 2

系统开发中的问题和解决方案

由于系统比较庞大,在系统的实现过程中,遇到了不少实际的问题,通过各种方法和技巧,我们解决了这些问题。以下是主要的问题和解决方法。

1. 多角度的数据字典查询和修改的实现

在我们的系统中,表的数目超过了50个,所以如何灵活的查询和修改表的结构就显得很重要。我们把表按照模块和功能进行了分类,既可以按照模块也可以按照功能查找相关的表,从模块的角度出发,主要有:全局/供应/销售/库存/固定资产/工资/商品维修/系统内部使用等。从功能的角度出发,主要有:静态配置/企业基础数据/业务动态数据/统计综合等。可以动态的修改对表和字段的注释,保证所有的表和字段能够被所有的开发人员所理解。

2. 建立以业务为中心的数据库表

我们通过在 Oracle 上另外建立了一个 schema,在这个 schema 下的表都是以业务为中心的表,比如:一系列面向客户的表,面向供应商的表,面向销售商的表,面向最终零售客户的表,面向商品的表等等。以通过对这些主题的分析,查询,更好的支持用户决策。

3. 业务流程的设计

根据系统的要求,我们建立了供应申请、销售批发、连锁店进货、库存管理等相应的业务流程,实际上根据 Notes 的机制,每一个业务流过程中的所有数据和资料都保存在一个 Notes 数据库中,由于 Notes 的表单可以根据不同用户的权限设置相关的属性,如只读、更改、不可见等,这样在表单的流过程中,不同的用户根据他的权限可以看到表单相应的部分,添入相应的内容。

4. 编码的设计

在编码设计方面主要考虑灵活性和方便性两个方面,当然一般来说编码本身有含义的话,可以使用户方便的记住编码。以前很多系统代码字段的长度都是固定的。这样在系统最开始,就必须定义好代码的层次和每个层次的长度。如果在系统设计完成后,需要修改代码体系的话,整个系统就必须做一个很大的调整。所以现在一般都是把代码字段设计成一个可变长的字段,只限制代码的最大长度,系统一开始并不确定代码的层次和每个层次的长度。可以让用户自己自由设定。最为灵活的方式为用户可以为每一条记录的相关代码定义层次和每层的长度。

5. 动态字段的加入

由于关系型数据库自身的限制,一个表中的字段在

设计阶段定下来以后,在使用阶段一般是不可能修改表结构的,所以如果用户想在一个表中的某条记录中增加一个字段事实上是不可能的。但是如果能够允许用户动态的在一些记录中增加信息,对用户无疑会有极大的方便。为了满足用户的需求,我们设计了如下的实现方法。

限制条件:用户新加的字段只能是字符型字段,而且有最大长度的限制。当然实质上,这个限制条件并不会给用户很大的不方便。

假设 T1 表为主记录表,我们建立一个 T2 表,该表的字段主要包括三部分: T1 表的主键相关的字段+用户附加字段的名称+用户附加字段的内容。以后用户自己添加的东西就放在这个表中,用户要查询某一记录是否有附加字段,也可以从该表中获得。如此,就实现了动态字段的加入的可能性。

6. 复杂查询条件的可视化

我们知道如果查询条件仅是简单的与查询和或查询,通过表格的形式就可以清晰的表现出来,但是当查询是由与或复杂关系的组合时,要可视化的表达比较困难,我们这里采用了树的形式来可视化表达复杂的查询。具体方法如下:

比如一个复杂的查询形式,用符号表示为: $((A \text{ and } B \text{ and } D) \text{ or } E) \text{ and } F$ 。其中 A, B, D, E, F 分别表示为一个查询条件,比如 A 可以等价于 $\text{age} > 20$ 这么一个条件。我们知道,表达式可以方便的转化为二叉树的形式,上式就可以表示为如图 3 所示:

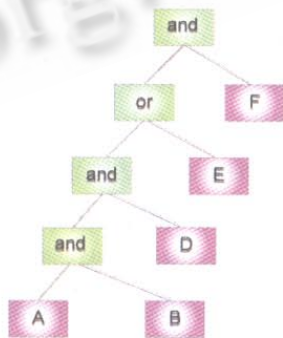


图 3

而二叉树很容易用树控件的形式表示出来,用户可以通过树控件方便的可视化的建立和维护复杂查询。当然,对使用这种查询的用户要求比较高,需要有一定的计算机基础,但灵活性也相当大。

7. 某些汉字字段内容拼音的自动生成

我们知道在很多查询中,我们必须输入汉字,而汉字

的输入相对比较慢,如果我们能够根据汉字自动生成拼音,这样,我们查询时只要输入拼音就可以,无疑加速了查询的输入灵活性。实现方法要基于有一个汉字拼音对照表,而该表可以从 Windows 的输入法中方便的得到。

比如对于一个中文词组,我们可以查到每一个汉字对应的拼音,然后取每一个拼音的首字母,这个拼音的组合就是该中文词组的拼音。当然如果一个汉字有多个读法,而且首字母不一样的话,就会有多种组合,要把这些拼音组合和相应的中文词组都放到一个表中,以后用户输入拼音时,就去该表中找出相应的中文词组,再到相关表中找到和该中文词组相匹配的所有记录。查全率没有问题,就是有时可能查到不相关的记录(即存在查准率的问题)。但是实际应用发现,查准率一般在 97% 以上。

8. 用户输入错误的实时检测

用户的输入错误一般可以在两个地方进行检测,一个是在数据库服务器端,通过规则和触发器可以对用户的输入错误进行检测,另外,可以在应用程序端对用户的错误进行检测。

在应用程序端的数据检测可以有几种类型

(1)简单的数据输入类型的检测。比如:有的字段只能是数字型,就必须避免用户输入字母型字符,有的字段必须要有具体的值,则必须提示用户输入具体的内容。

(2)相关性数据检测。比如有些字段的值,必须从另外一些表中取出,则不允许用户自己随意输入,必须从相关的表中得到。

(3)通常易犯错误检测。比如:在记帐凭证中,必须是借方和贷方相对应,不允许都是借方或者都是贷方。这样系统在数据入库之前,就可以检查用户的单子,看是否单子中的科目都是只有一个方向的。另外,用户可能经常输错一些商品的价格,可以对这些商品的价格设定一个通常的上下限(可以进行不同的配置),如果用户输入的价格超过了这个范围,就自动给出警告。同时对用户经常容易输错的模式进行归纳,把用户的输入和这些模式相比较,如果相匹配的话,就给出警告。

通过上述这些错误预防的方法,保证了用户数据输入的准确性,对系统很有好处,因为某些数据是在流程中进行操作的,一旦发生错误,很难进行回滚。所以这些错误预防对于现在很多对计算机操作还不是很熟练的企业员工来说显得尤为重要。

9. 信息输出的多种模式设计

一般现在的 4GL 的开发工具都自带报表工具,但是这些报表工具一般都比较死板,格式都是在设计时候已

经确定的,用户既不可能对格式做修改也不可能对数据进行自定义的汇总和其他的编排。所以灵活性不是很多,我们考虑到现有系统的客户端一般都是在 Window 操作系统下实现的,而 Windows 中有一个 OLE Automation 这个很好的技术,很多流行的表格和字处理系统都支持 OLE Automation 技术。所以我们可以通过 OLE Automation 技术,可以让用户选择把信息输出到流行的表格和字处理系统中,比如 EXCEL 和 WORD 等。而且 EXCEL 和 WORD 都可以通过 VBA 进行方便的编程,这样用户可以通过操作 EXCEL 和 WORD 软件对信息进行二次处理,使用户能够更灵活的处理信息。

10. 用户权限的设定

用户权限的设定一般有三种方式:

(1)充分利用数据库的用户权限管理。

(2)在应用程序级设定权限。

(3)数据库用户管理和应用程序级权限的管理相结合。

这种方法是针对每一个用户,他的权限是一个数据库权限和应用程序权限的交集,即他的操作权限受到数据库权限和应用程序权限的双重制约。这样既保证数据库的安全性,同时又保证应用程序的安全性。不过在实现的时候相对比较复杂一点。

我们在实现的时候采用了第三种方案,保证了数据库级和应用级的安全性。

11. 分布式系统的数据复制策略制定

一般来说,复制机制有下面几种:

(1)单一的发布者,多个订阅者

(2)多个发布者,单个订阅者

(3)多个发布者,多个订阅者

(4)远程分发

最常见的是第一种模式,即所有的数据的修改和变化都在单一的服务器上发生,然后该服务器把这些修改的信息发布到其他的数据库服务器上,比较适合于只需要进行简单的数据分发要求的小型的企业组织。

第二种模式允许从多个服务器中得到数据,并统一存放在一个中心的订阅者那里。这种模式比较适合于有多个分公司的企业,每个分公司有自己的维护的数据库,总部有所有分公司的数据的汇总,以便了解整个企业的运作情况。

在第三种模式下,数据库服务器既是数据的发布者也可以是数据的订阅者,这种结构一般是对应于分布式的企业,各个分公司有很大的自主权,它们不但需要发布数(下至 17 页)

(上接 16 页)

据给其他的分公司，也需要从其他的分公司得到数据。

第四种模式适应于更大的企业的组织形式，在这种情况下，有一个订阅者服务器组群和发布者服务器组群，它们之间通过一对发布服务器对所有的发布和订阅消息进

行调度和处理。

根据我们系统的特点，有多个连锁店和商场，它们相对业务对立运行，但向总部要求相关的商品，并且在关键业务上需要总部审批，所以我们采取了第二种方案。■