CFDAB

国 家 食 品 药 品 监 管 信 息 化 标 准

CFDAB/T-0304-2013

第 10 部分：数据库设计规范

Part 10: Data interface standard

（征求意见稿 V1.0）

2013－xx-xx 发布　　20xx－xx-xx 实施

国家食品药品监督管理总局信息办　　发 布

目 次

前 言

《国家食品药品监管信息化标准》发布以下几个部分：

——第 1 部分：标准化指南；

——第 2 部分：基本术语；

——第 3 部分：标准编写规则；

——第 4 部分：信息分类与代码；

——第 5 部分：数据元与代码集；

——第 6 部分：数据集元数据；

——第 7 部分：软件开发过程规范；

——第 8 部分：应用支撑平台通用技术规范；

——第 9 部分：数据共享与交换接口；

——第 10 部分：数据库设计规范；

——第 11 部分：网络及通信；

——第 12 部分：信息安全；

——第 13 部分：工程管理。

本标准为第 10 部分。

本标准依据 GB/T1.1—2009 给出的规则起草。

本标准由国家食品药品监督管理总局信息中心提出。

本标准由国家食品药品监督管理总局信息办归口。

本标准起草单位：国家食品药品监督管理总局信息中心、广东省食品药品监督管理局、中科软科技股份有限公司。

第 10 部分：数据库设计规范

1. 范围

本规范规定了国家食品药品监管信息化工程各应用系统数据库的设计原则、数据库命名编写规范、数据库结构设计、数据库操作规范、数据存储规范、数据库运用、数据库性能设计、数据库外部设计、数据库版本控制、数据库安全性设计等方面内容，明确了应用系统数据库的安全性、完整性，并针对数据库进行外部设计、结构设计和运用设计，统一各应用系统的数据库设计和开发，为数据集成、数据共享和交互做好准备。

本规范适用于规范数据库的建设，并对国家食品药品监管信息化工程各应用系统的数据库建设起指导作用。

2. 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅所注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T 5271.1—2000 信息技术 词汇 第1部分 基本术语

GB/T 5271.4—2000 信息技术 词汇 第4部分 数据的组织

GB/T 5271.8—2001 信息技术 词汇 第8部分 安全

3. 术语和定义

CFDAB/T-0102-2013 收录的术语和定义适用于本标准。为了便于使用，以下重复列出了 CFDAB/T-0102-2013 中的部分术语和定义。

3.1

数据库 Database

按照概念结构组织的数据的汇集，它描述这些数据的特征及与数据对应的实体间的关系并支持一个或多个应用领域。

[CFDAB/T 0102-2013 定义 3.2.2.7]

3.2

表 Table

数据的一种排列，其中每一项可通过变元或关键字标识。

[CFDAB/T 0102-2013 定义 3.2.2.8]

3.3

关系模式 Relation Schema

关系的描述称为关系模式。一个关系模式应当是一个五元组，它可以形式化地表示为：R(U, D, DOM, F)。其中 R 为关系名，U 为组成该关系的属性名集合，D 为属性组 U 中属性所来自的域，DOM 为属性向域的映像集合，F 为属性间数据的依赖关系集合。

[CFDAB/T 0102-2013 定义 3.2.2.9]

3.4

范式 Paradigm

构造数据库必须遵循一定的规则，在关系数据库中，这种规则就是范式。范式是符合某一种级别的关系模式的集合。关系数据库中的关系必须满足一定的要求，即满足不同的范式。

[CFDAB/T 0102-2013 定义 3.2.2.10]

3.5

第一范式 1NF

如果一个关系模式 R 的所有属性都是不可分的基本数据项，则 R∈1NF。

第一范式包括下列指导原则：数据组的每个属性可以包含一个值；

关系中的每个数组必须包含相同数量的值；

关系中的每个数组一定不能相同。

[CFDAB/T 0102-2013 定义 3.2.2.11]

3.6

第二范式 2NF

若关系模式 R∈1NF，并且每一个非主属性都完全函数依赖于 R 的码，则 R∈2NF。

[CFDAB/T 0102-2013 定义 3.2.2.12]

3.7

第三范式 3NF

满足第三范式（3NF）必须先满足第二范式（2NF）。简而言之，第三范式（3NF）要求一个数据库表中不包含已在其它表中已包含的非主关键字信息。

[CFDAB/T 0102-2013 定义 3.2.2.13]

3.8

内聚 Cohesion

模块内部各部分的聚合程度。在数据库部分特指各个模块内部的数据库表之间的聚合程度。

[CFDAB/T 0102-2013 定义 3.2.2.14]

3.9

耦合 Coupled

模块之间的关联程度。在数据库部分特指各个模块之间的数据库表的关联程度。

[CFDAB/T 0102-2013 定义 3.2.2.15]

3.10

视图 View

视图是原始数据库数据的一种变换，是查看表中数据的另外一种方式。可以将视图看成是一个移动的窗口，通过它可以看到感兴趣的数据。

视图是从一个或多个实际表中获得的，这些表的数据存放在数据库中。那些用于产生视图的表叫做该视图的基表。一个视图也可以从另一个视图中产生。

[CFDAB/T 0102-2013 定义 3.2.2.16]

3.11

存储过程 Storage process

存储过程是由流控制和 SQL 语句书写的过程，这个过程经编译和优化后存储在数据库服务器中，使用时只要调用即可。其中，若干个有联系的过程可以组合在一起构成程序包。

[CFDAB/T 0102-2013 定义 3.2.2.17]

3.12

触发器 Trigger

触发器是一种特殊类型的存储过程，它不同于我们前面介绍过的存储过程。触发器主要是通过事件进行触发而被执行的，而存储过程可以通过存储过程名字而被直接调用。当对某一表进行诸如 UPDATE、INSERT、 DELETE 这些操作时，数据库会自动执行触发器所定义的 SQL 语句，从而确保对数据的处理必须符合由这些 SQL 语句所定义的规则。

[CFDAB/T 0102-2013 定义 3.2.2.18]

3.13

索引 Index

索引是一个单独的、物理的数据库结构，它是某个表中一列或若干列值的集合和相应的指向表中物理标识这些值的数据页的逻辑指针清单。

[CFDAB/T 0102-2013 定义 3.2.2.19]

3.14

主键 Primary key

主键，又称主关键字或主码，指能够唯一标识一条记录的字段或字段集。

[CFDAB/T 0102-2013 定义 3.2.2.20]

3.15

外键 Foreign key

如果一个属性集不是所在关系的关键字，但是是其他关系的关键字，则该属性集称为外部关键字。外部关键字也称为外码或外键。

[CFDAB/T 0102-2013 定义 3.2.2.21]

3.16

联机分析处理 On-Line Analysis Processing（OLAP）

联机分析处理是共享多维信息的、针对特定问题的联机数据访问和分析的快速软件技术。它通过对信息的多种可能的观察形式进行快速、稳定一致和交互性的存取，允许管理决策人员对数据进行深入观察。决策数据是多维数据，多维数据就是决策的主要内容。OLAP 专门设计用于支持复杂的分析操作，侧重对决策人员和高层管理人员的决策支持，可以根据分析人员的要求快速、灵活的进行大数据量的复杂查询处理，并且以一种直观而易懂的形式将查询结果提供给决策人员，以便他们准确掌握经营状况，了解对象的需求，制定正确的方案。

[CFDAB/T 0102-2013 定义 3.2.2.22]

3.17

联机事务处理系统 On-Line transaction processing（OLTP）

联机事务处理系统(OLTP)，也称为面向交易的处理系统，其基本特征是原始数据可以立即传送到计算中心进行处理，并在很短的时间内给出处理结果。这样做的最大优点是可以即时地处理输入的数据，及时地回答。也称为实时系统 (Real time System)。衡量联机事务处理系统的一个重要性能指标是系统性能，具体体现为实时响应时间(Response Time)，即用户在终端上送入数据之后，到计算机对这个请求给出答复所需要的时间。

[CFDAB/T 0102-2013 定义 3.2.2.23]

3.18

实体完整性 Physical integrity

实体完整性指表中行的完整性。要求表中的所有行都有唯一的标识符，称为主关键字。主关键字是否可以修改，或整个列是否可以被删除，取决于主关键字与其他表之间要求的完整性。

实体完整性规则规定基本关系的所有主关键字对应的主属性都不能取空值。

[CFDAB/T 0102-2013 定义 3.2.2.24]

3.19

参照完整性 Referential integrity

参照完整性，简单的说就是表间主键外键的关系。

参照完整性属于表间规则。对于永久关系的相关表，在更新、插入或删除记录时，如果只改其一不改其二，就会影响数据的完整性。例如：修改父表中关键字值后，子表关键字值未做相应改变；删除父表的某记录后，子表的相应记录未删除，致使这些记录成为孤立记录；对于子表插入的记录，父表中没有相应关键字值的记录，等等。对于这些设计表间数据的完整性，统称为参照完整性。

[CFDAB/T 0102-2013 定义 3.2.2.25]

3.20

数据完整性 Data Integrity

数据完整性是指数据的精确性（Accuracy）和可靠性（Reliability）。它是应防止数据库中存在不符合语义规定的数据和防止因错误信息的输入输出造成无效操作或错误信息而提出的。数据完整性分为四类：实体完整性（Entity Integrity）、域完整性（Domain Integrity）、参照完整性（Referential Integrity）、用户定义的完整性（User-defined Integrity）。

[CFDAB/T 0102-2013 定义 3.2.2.26]

3.21

概念模式（E-R 图）Conceptual model

E 就是实体，R 就是关系。

E-R 图就是实体关系图，形象的表示实体之间的联系。设计数据库时要区分出一个个的实体，它们之间由键进行关联。

[CFDAB/T 0102-2013 定义 3.2.2.27]

3.22

外模式 User Mode

外模式也称用户模式，它是数据库用户能够看见和使用的局部数据的逻辑结构和特征的描述，是数据库用户的数据视图，是与某一应用有关的数据的逻辑表示。外模式通常是模式的子集，一个数据库可以有多个外模式，应用程序都是和外模式打交道的。外模式是保证数据库安全性的一个有力措施，每个用户只能看见和访问所对应的外模式中的数据，但对数据库中的其余数据无权访问。

[CFDAB/T 0102-2013 定义 3.2.2.28]

3.23

内模式 Storage mode

内模式也称存储模式，一个数据库只有一个内模式。它是数据物理结构和存储方式的描述，是数据

在数据库内部的表示方式。例如，记录的存储方式是顺序结构存储还是 B 树结构存储；索引按什么方式组织；数据是否压缩，是否加密；数据的存储记录结构有何规定等。

[CFDAB/T 0102-2013 定义 3.2.2.29]

3.24

概念设计 Conceptual design

概念设计是由分析用户需求到生成概念产品的一系列有序的、可组织的、有目标的设计活动，它表现为一个由粗到精、由模糊到清晰、由具体到抽象的不断进化的过程。

[CFDAB/T 0102-2013 定义 3.2.2.30]

3.25

逻辑设计 Logical Design

逻辑设计是指把概念设计得到的概念数据库模式变为逻辑数据模式的过程，它依赖于 DBMS。

[CFDAB/T 0102-2013 定义 3.2.2.31]

3.26

物理设计 Physical design

物理设计是指设计数据库的存储结构和物理实现方法的过程。

[CFDAB/T 0102-2013 定义 3.2.2.32]

3.27

访问控制 Access control

一种保证手段，即数据处理系统的资源只能由被授权实体按授权方式进行访问。

[CFDAB/T 0102-2013 定义 3.2.2.33]

3.28

数据库管理系统 Database Management System （DBMS）

一种操纵和管理数据库的大型软件，用于建立、使用和维护数据库，简称 dbms。它对数据库进行统一的管理和控制，以保证数据库的安全性和完整性。用户通过 dbms 访问数据库中的数据，数据库管理员也通过 dbms 进行数据库的维护工作。它可使多个应用程序和用户用不同的方法在同时或不同时刻去建立，修改和询问数据库。

[CFDAB/T 0102-2013 定义 3.2.2.34]

4. 数据库架构

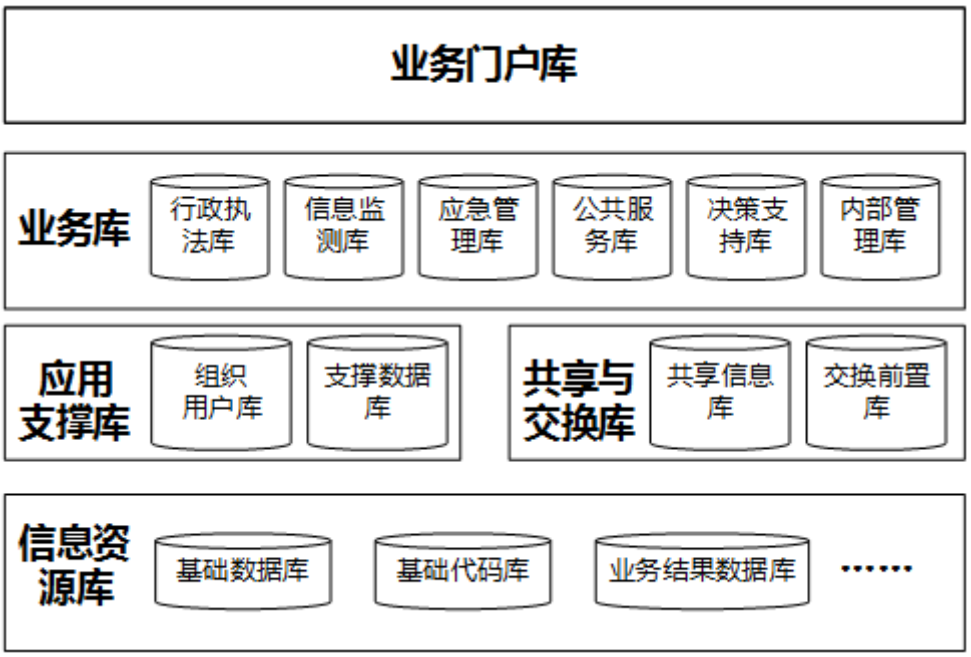

图 1：数据库逻辑结构图

图 1 为国家食品药品监管信息化工程数据库的逻辑结构图，将国家食品药品监管信息化工程数据库分为信息资源库、应用支撑库、共享与交换库、业务库、业务门户库五部分，应用系统数据库建设应符合以上结构的要求。其中：

a）信息资源库存储信息资源的结果数据，应包括基础数据库、基础代码库、业务结果数据库等等，信息资源库应存储公共的基础数据和业务结果数据，作为数据仓库的数据源之一。

b）应用支撑库存储应用支撑平台的支撑数据和工程统一的组织和用户信息。

c）共享与交换库存储系统与内部各应用系统、与系统外部应用系统的共享与交换数据，分为共享信息库和交换前置库，建设应用系统需要进行信息交换时，应将数据共享与交换内容抽取出进行共享与交换。

d）业务库存储国家食品药品监管信息化工程各应用系统的业务数据，包括行政执法库、信息监测库、应急管理库、公共服务库、决策支持库、内部管理库等。

e）业务门户库存储国家食品药品监管信息化工程业务门户的数据，应用系统建设业务门户时，应将业务门户数据存储在业务门户库中。

5. 设计原则

5.1 概述

数据库设计（Database Design）是指根据用户的需求，在某一具体的数据库管理系统上，设计数据库的结构和建立数据库的过程，就是规划和结构化数据库中的数据对象以及这些数据对象之间关系的过程。数据库设计原则主要从规范化、数据完整性、分级设计、高内聚、低耦合、整

体性、高效性几个方面进行阐述，是各应用系统数据库建设的基本要求和基本原则。所有数据库的建设应在满足上述原则的基础上，进行个性化的扩展设计。

5.2 设计规范化原则

对数据库的设计，完全采用 E-R 模型设计思想，并遵循关系数据库设计第三范式（3NF）标准要求。

范式是数据库设计所需要满足的规范，满足这些规范的数据库是简洁的、结构明晰的，同时，不会发生插入（insert）、删除（delete）和更新（update）操作异常。

数据库设计应遵循以数据库三范式为主导的设计原则，尽量保证数据的一致性、减少数据冗余。对于有较高性能及其它特殊要求的情况，可以考虑以“空间换时间”的设计方式。第三范式（3NF）要求一个数据库表中不包含已在其它表中已包含的非主关键字信息。

5.3 数据完整性原则

数据完整性是为防止数据库中存在不符合语义规定的数据和防止因错误信息的输入输出造成无效操作或错误信息而提出的，数据完整性确保了数据库中的信息不会被篡改。如果数据具有完整性，则表示数据有效（正确并且准确），而且数据库的关系结构是完整的。参照完整性约束可以加强数据库的关系结构，这些规则使数据在各个表之间保持一致。

5.4 数据安全性原则

数据库的安全性是指保护数据，防止非法用户使用数据库或合法用户非法使用数据库造成数据泄露、更改或破坏。

系统在设计时要充分考虑到数据安全性，不同权限的用户看到的数据是存在差异的，通过制定对数据库的各类操作的标准，建立具有统一的管理和控制功能，例如：即使是系统管理员也无法知道用户的口令，因此无法以用户的身份进行登录。

5.5 分级设计原则

a）需求分析阶段：综合并分析各个用户的应用需求。

b）概念设计阶段：形成独立于机器特点，独立于各个 DBMS 产品的概念模式(E-R 图)。

c）逻辑设计阶段：首先将 E-R 图转换成具体的数据库产品支持的数据模型，如关系模型，形成数据库逻辑模式，然后根据用户处理的要求、安全性的考虑，在基本表的基础上再建立必要的视图(View)，形成数据的外模式。

d）物理设计阶段：根据 DBMS 特点和处理的需要，进行物理存储安排，建立索引，形成数据库内模式。

按照需求分析、概念设计、逻辑设计、物理设计四个过程进行设计，形成一个统一的、稳定的、可扩展、安全的数据库方案。

5.6 高内聚原则

将数据库中公共使用的基础数据元和数据集独立建立基础数据库，同一数据中心内以数据库授权等形式访问基础库中的数据，避免重复设计和存储。对基础数据进行统一管理，保证数据的一致性。

5.7 低耦合原则

应用系统在数据库的设计上应该避免直接调用其它应用系统的数据库，对数据库进行独立建库和独立维护，降低因为其它应用系统数据库变更而引起的威胁，但要遵循统一的数据库规范和命名规则。

基础库的独立设计，同一数据中心内以数据库授权等形式访问基础库中的数据，而不是重复设计、存储。

5.8 高效性原则

数据库设计时，在遵循标准规范的基础上要注意的提高数据库的效率。数据库的执行效率将直接影响系统的响应速度，过于迟缓的响应速度将直接降低用户的使用感受，所以在设计数据库时，应该注意提高数据库的效率。

6. 数据库命名

6.1 概述

使用统一的数据库命名规范，使数据库命名风格标准化，以便于阅读、理解和继承。本规范主要通过基本命名规范和数据库对象的命名规范两方面来阐述。

a）基本命名规范包括：字段命名规范、约束命名规范、序列命名规范、同义词命名规则、Db Link 命名规范等。

b）数据库对象的命名规范包括：表命名规范、视图命名规范、存储过程命名规范、触发器命名规范、索引命名规范等。

6.2 系统标识

国家食品药品监管信息化工程应对所建应用系统及数据库等进行系统标识，表 1 为部分应用系统及数据库的系统标识，供应用系统制定系统标识参考。

表 1

序号	系统名称/数据库名称	系统标识
1	药品监管码管理子系统	MDSC
2	药品监管码赋码子系统	ECS
3	药品流通监管子系统	GSP
4	药品认证查询子系统	IDC
5	药品真伪鉴别子系统	IDA
6	药品强制召回子系统	DMR
7	药品紧急调配子系统	DUA

8	药品安全评估作业子系统		DSA
9	医院合理用药（配药剂量）监管子系统		SHRP
10	药效禁忌告知子系统		NDC
11	广告监督子系统		ADS
12	GLP 监管子系统		GLP
13	GCP 监管子系统		GCP
14	GMP 监管子系统		GMP
15	内部管理作业平台		OA
16	行政许可作业子系统		AP
17	应用支撑平台		PUB
18	数据采集与资源整合平台		DCP
19	数据管理与维护平台		DM
20	数据共享与服务平台		DSS
21	互联网业务门户		IBP
22	专网业务门户		NBP
23	共享信息库		SDS
24	交换前置库		SDE
25	信息资源库	法律法规类	DS_LAW
26		基础数据类	DS_BAS
27		基础代码类	DS_COD
28		行政许可类	DS_AP
29		监管对象类	DS_OS
30		业务分析类	DS_BA
31		公众服务类	DS_PS

6.3 基本命名规范

a）语言

命名宜使用拼音简写，不宜使用中文或者特殊字符。

当出现对象名重名且不同类型对象时，宜加类型前缀或后缀以示区别。

b）大小写

名称宜大写，方便不同数据库移植，避免程序调用问题。

c）单词分隔

命名的各单词之间可以使用下划线进行分隔。

d）保留字

命名不宜使用 SQL 保留字。

e）命名长度

表名、字段名、视图名长度应限制在 20 个字符内(含前缀)。

f）字段名称

同一个字段名在一个数据库中只能代表一个意思。比如 telephone 在一个表中代表“电话号码”的意思，在另外一个表中就不能代表“手机号码”的意思。

不同的表用于相同内容的字段应该采用同样的名称，字段类型定义。

6.4 数据库对象命名规范

a）表

1）实体（表）

前缀为系统标识，<系统标识>_<表标识>。

表标识建议为拼音简称。

2）属性（列）

COMP_CODE，字段名称应用字母开头，采用有特征含义的单词或缩写，不宜用双引号包含。

b）视图

前缀为 V_，v_[<系统标识>]_<视图标识>。

视图标识建议为拼音简称。

c）存储过程

前缀为 P_，p_[<系统标识>]_<存储过程标识>。

存储过程标识建议为拼音简称。

d）函数

前缀为 F_，f_[<系统标识>]_<函数标识>。

函数标识建议为拼音简称。

e）触发器

tr_<表标识>_<i,u,d 的任意组合>(after)；

ti_<表标识>_<i,u,d 的任意组合>(instead)。

f）自定义数据类型

ud_<自定义数据类型标识>_<数据类型>。

g）Default

Df_<Default 标识>。

h）Rule

ru_<Rule 标识>。

i）主键

前缀为 PK_，pk_<表标识>_<主键标识>。

j）外键

前缀为 FK_，fk_<表标识>_<主表标识>_<外键标识>。

k）索引

前缀为 IDX_，IDX_<表标识>_<构成的字段名>。如果复合索引的构成字段较多，则只包含第一个字段，并添加序号。

7. 数据库结构设计

7.1 需求分析

设计一个性能良好的数据库系统，明确应用环境对系统的要求是首要的和基本的，因此，应该把对用户需求的收集和分析作为数据库设计的第一步。

需求分析的主要任务是通过详细调查要处理的对象，包括某个组织、某个部门的业务管理等，充分了解原手工或原计算机系统的工作概况及工作流程，明确用户的各种需求，产生数据流图和数据字典，然后在此基础上确定新系统的功能，并产生《软件需求规格说明书》。应该注意的是，新系统应充分考虑今后可能的扩充和改变，不宜仅仅按当前应用需求来设计数据库，应留有一定的扩展。

需求分析的重点是调查、收集和分析用户数据管理中的信息需求、处理需求、安全性与完整性要求。信息需求是指用户需要从数据库中获得的信息的内容和性质。由用户的信息需求可以导出数据需求，即在数据库中应该存储哪些数据。处理需求是指用户要求完成什么处理功能，对某种处理要求的响应时间，处理方式指是联机处理还是批处理等。明确用户的处理需求，将有利于后期应用程序模块的设计。

需求分析的方法：调查组织机构情况、各部门的业务活动情况、协助用户明确对新系统的各种要求、确定新系统的边界。

a）了解组织机构的情况，调查这个组织由哪些部门组成，各部门的职责是什么，为分析信息流程做准备。

b）了解各部门的业务活动情况，调查各部门输入和使用什么数据，如何加工处理这些数据。输出什么信息，输出到什么部门，输出的格式等。在调查活动的同时，要注意对各种资料的收集，如票证、单据、报表、档案、计划等，要特别注意了解这些报表之间的关系，各数据项的含义等。

c）确定新系统的边界。确定哪些功能由计算机完成或将来准备让计算机完成，哪些活动由人工完成。由计算机完成的功能就是新系统应该实现的功能。

常用的调查方法有： 跟班作业、开调查会、请专人介绍、询问、设计调查表请用户填写、查阅记录。

分析和表达用户需求的方法主要包括自顶向下和自底向上两类方法。自顶向下的结构化分析方法(Structured Analysis，简称 SA 方法)从最上层的系统组织机构入手，采用逐层分解的方式分析系统，并把每一层用数据流图和数据字典描述。

数据流图表达了数据处理过程的关系。系统中的数据则借助数据字典(Data Dictionary，简称 DD)来描述。数据流图是软件工程中专门描绘信息在系统中流动和处理过程的图形化工具，因为数据流图是逻辑系统的图形表示，即使不是专业的计算机技术人员也容易理解，所以是极好的交流工具。图 2 给出了数据流图中所使用的符号及其含义。

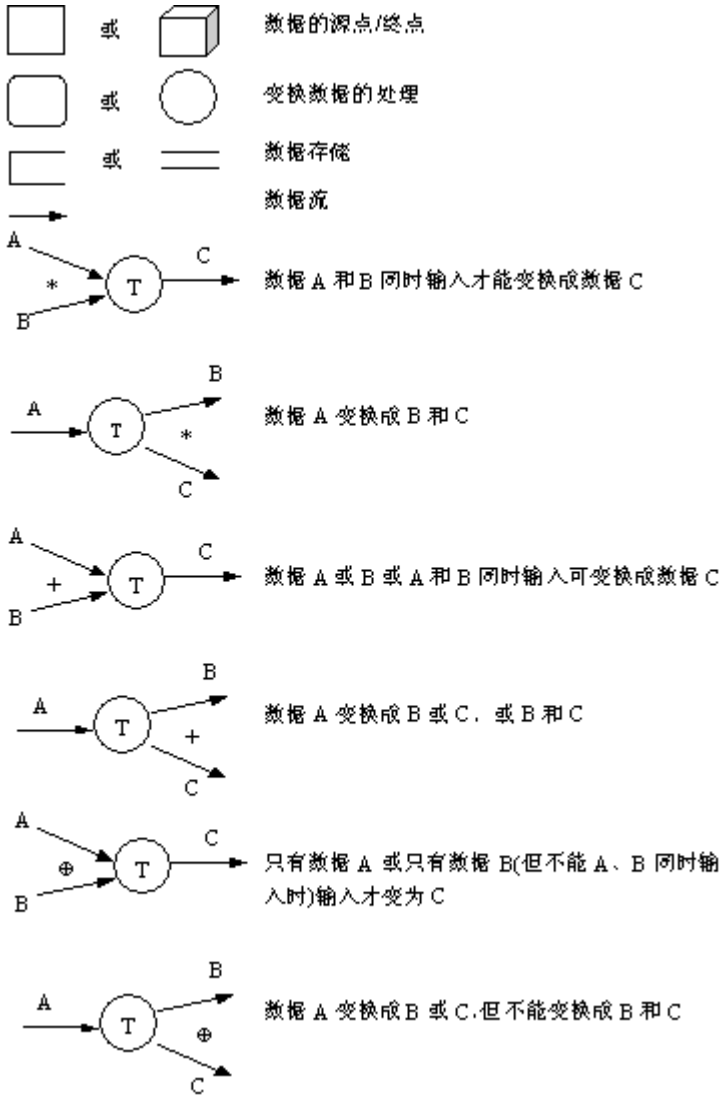

图 2：数据流图的符号图

7.2 概念设计

概念结构设计的任务是在需求分析阶段产生的《软件需求规格说明书》的基础上，按照特定的方法把它们抽象为一个不依赖于任何具体机器的数据模型，即概念模型。概念模型使设计者的注意力能够从复杂的实现细节中解脱出来，而只集中在最重要的信息的组织结构和处理模式上。

概念模型具有以下的特点：

a）概念模型是对现实世界的抽象和概括，它真实、充分地反映了现实世界中事物和事物之间的联系，能满足用户对数据的处理要求。

b）由于概念模型简洁、明晰、独立于计算机，很容易理解，因此可以用概念模型和不熟悉计算机的用户交换意见，使用户能积极参与数据库的设计工作，保证设计工作顺利进行。

c）概念模型易于更新，当应用环境和应用要求改变时，容易对概念模型修改和扩充。

d）概念模型很容易向关系、网状、层次等各种数据模型转换。

描述概念模型的有力工具是 E-R 图。E-R 模型是一个面向问题的概念模型，即用简单的图形方式(E-R 图)描述现实世界中的数据。这种描述不涉及数据在数据库中表示和存取方法，非常接近人的思维方式。后来又提出了扩展实体联系模型(Extend Entity-Relationship Model)，简称为“EER 模型”。EER 模型目前已经成为一种使用广泛的概念模型，为面向对象的数据库设计提供了有效的工具。

7.3 逻辑设计

7.3.1 概述

概念结构设计所得的 E-R 模型是对用户需求的一种抽象的表达形式，它独立于任何一种具体的数据模型，因而也不能为任何一个具体的 DBMS 所支持。为了能够建立起最终的物理系统，还需要将概念结构进一步转化为某一 DBMS 所支持的数据模型，然后根据逻辑设计的准则、数据的语义约束、规范化理论等对数据模型进行适当的调整和优化，形成合理的全局逻辑结构，并设计出用户子模式。这就是数据库逻辑设计所要完成的任务。

7.3.2 Tablespace

每个表在创建时候，应指定所在的表空间，不宜采用默认表空间以防止表建立在系统表空间上导致性能问题。对于事务比较繁忙的数据表，应存放在该表的专用表空间中。

7.3.3 数据项

业务数据项设计内容应从“CFDAB/T 0202-2013 第 5 部分：数据元与代码集”标准中获取相应的数据项，如业务数据项在“CFDAB/T 0202-2013 第 5 部分：数据元与代码集”标准中查询不到，应向标准管理维护机构提出增项申请，待“CFDAB/T 0202-2013 第 5 部分：数据元与代码集”标准中完善后，再获取相应的数据项。

数据项的命名参照基本命名规范。

7.3.4 主键约束

关联表的父表应有主健，主健字段或组合字段应满足非空属性和唯一性要求。对于数据量比较大的父表，应指定索引段。

7.3.5 外键关联

对于关联两个表的字段，应该分别建立主键、外键。实际是否建立外键，根据对数据完整性的要求决定。为了提高性能，对于数据量比较大的表应对外键建立索引。

7.3.6 NULL 值

对于字段能否 NULL，应该在 SQL 建表脚本中明确指明，不应使用缺省。由于 NULL 值在参加任何运算中，结果均为 NULL。所以在应用程序中应利用 null()函数把可能为 NULL 值得字段或变量转换为非 NULL 的默认值。

7.3.7 Check 条件

对于字段有检查性约束，应指定 check 规则。

7.3.8 触发器

触发器是一种特殊的存储过程，通过数据表的 DML 操作而触发执行，起作用是为确保数据的完整性和一致性不被破坏而创建，实现数据的完整约束。

触发器的 before 或 after 事务属性的选择时候，对表操作的事务属性应与应用程序事务属性保持一致，以避免死锁发生。在大型导入表中，尽量避免使用触发器。

7.3.9 注释

表、字段等应该有中文名称注释，以及需要说明的内容。

7.4 物理设计

7.4.1 概述

为逻辑数据模型选取一个最适合应用环境的物理结构（包括存储结构和存取方法）。根据 DBMS 特点和处理的需要，进行物理存储安排，设计索引，形成数据库内模式。

7.4.2 数据库服务名称

数据库服务名称是唯一标志数据库的符号，命名长度不应超过 5 个字符。对于单节点数据库，以字符开头的 5 个长度以内字串作为服务的命名。对于集群数据库，当定义服务名称后，各节点服务名称命名增加递增的序号，例如节点的序号:1，2，…,64。例如 rac1、rac2、rac24。

7.4.3 数据库类型选择

对于海量数据库系统，采用 data warehouse 的类型。对于小型数据库或 OLTP 类型的数据库，采用 Transaction Processing 类型。

7.4.4 数据库连接类型选择

数据库有专用服务器连接类型和多线程服务器 MTS 连接类型。对于批处理服务，需要专用服务器连接方式，对于 OLTP 服务则 MTS 的连接方式比较合适。

7.4.5 数据库字符集选择

为了使数据库能够正确支持多国语言，应配置合适的数据库字符集，采用 UTF8 字符集。

8. 数据库操作规范

8.1 表

a）避免使用业务主属性作为表的主键。

b）多表连接时，使用表的别名来引用列。

示例：

X SELECT abc002,abd003
FROM ab001 ,ab020
WHERE ab001.col2=ab020.col3

.....

O SELECT t1.abc002,t2.abd003
FROM ab001 t1,ab020 t2
WHERE t1.col2=t2.col3

.....

8.2 视图

a）使用静态视图，不宜使用动态视图。

b）视图中尽量避免使用 Order By 语句，Order By 语句会影响视图的效率。

c）视图中尽量避免使用 union 或 union all 语法（Sybase 不支持）。

d）视图编写宜使用英文字母并且大写。

e）经常使用的字段放到属于该表的那组字段的前面，减少数据库对视图的操作次数。

8.3 触发器

a）编写过程中不宜使用自动缩排，也不宜使用 TAB 键缩进。

b）变量名称命名方式同存储过程。

c）游标 （Cursor）以 ‘Cur_’ + 名称 命名。

d）所有变量应加注释。

e）每个功能段之间应加注释。

8.4 存储过程

存储过程编写时应注意函数名字不能使用数据库系统的保留字、内置函数、预定义类型以及 SQL 关键字，表 2 为保留字、内置函数、预定义类型、SQL 关键字的举例。

表 2

类型	约定	举例
保留字	大写	BEGIN、DECLARE、ELSIF

内置函数	大写	SUBSTR、COUNT、TO_NUMBER
预定义类型	大写	NUMBER(7,2)、BOOLEAN
SQL 关键字	大写	SELECT、INTO、WHERE

8.5 索引

索引是从数据库中获取数据的最高效方式之一。95%的数据库性能问题都可以采用索引技术得到解决。在使用索引时，应注意以下几点：

a）逻辑主键使用唯一的成组索引，对系统键（作为存储过程）采用唯一的非成组索引，对任何外键列采用非成组索引。考虑数据库的空间有多大，表如何进行访问，还有这些访问是否主要用作读写。

b）大多数数据库都索引自动创建的主键字段，也应索引外键，它们也是经常使用的键，比如运行查询显示主表和所有关联表的某条记录就用得上。

c）不宜索引 memo/note 字段，不宜索引大型字段（有很多字符），这样作会让索引占用太多的存储空间。

d）不宜索引常用的小型表

不宜为小型数据表设置任何键，对这些进行插入和删除操作的索引进行维护会比扫描表空间消耗更多的时间。

9. 数据存储规范

9.1 数据库存放规则

9.1.1 操作系统环境

针对不同类型数据库系统，应采用不同的操作系统以及不同的数据库部署方式，选择操作系统环境应考虑数据库系统的需求。

示例：

对于中小型数据库系统，采用 Linux 操作系统比较合适，对于数据库冗余要求负载均衡能力要求较高的系统，可以采用数据库的集群方法，集群节点数范围在 2—64 个。对于大型数据库系统，可以采用小型机系统。RAD5 适合只读操作的数据库，RAD1 适合 OLTP 数据库。

9.1.2 内存要求

不同的操作系统管理内存的能力不同，不同的数据库系统对内存的要求也不同，建议根据操作系统环境和数据库系统规模选择适当的数据库内存。

示例：

对于 Linux 操作系统下的数据库，由于在正常情况下对数据库内存的管理能力不超过 1.7GB。所以总的物理内存在 4GB 以下。数据库内存的大小为物理内存的 50%—75%。对于 64 位的小型系统，数据库内存的管理超过 2GB 的限制，数据库内存设计在一个合适的范围内：物理内存的 50%— 70%，当数据

库内存过大的时候会导致内存分页，影响系统性能。

9.1.3 交换区设计

当物理内存在 2GB 以下的情况下，交换分区 swap 为物理内存的 3 倍，当物理内存>2GB 的情况下，swap 大小为物理内存的 1—2 倍，当物理内存>8GB 时，swap 交换区仅设置为 8GB 即可。

9.2 数据库存取方法

数据存取层介于语言处理层和数据存储层之间。它向上提供单元组接口，即导航式的一次一个元组的存取操作，向下则以系统缓冲区的存储器接口作为实现基础。

数据存取层的任务是：

a）提供一次一个元组的查找、插入、删除、修改等基本操作。

b）提供元组查找所循的存取路径以及对存取路径的维护操作。如对索引记录的查找、插入、删除、修改。

c）对记录和存取路径的封锁、解锁操作。

d）日志文件的登记和读取操作。

e）辅助操作。如扫描、合并/排序，其操作对象有关系、有序表、索引等。

为了完成上述功能，通常把存取层又划分为若干功能子系统加以实现。

9.3 数据类型

9.3.1 字符型

固定长度的字串类型采用 char，长度不固定的字串类型采用 varchar。避免在长度不固定的情况下采用 char 类型。如果在数据迁移等出现以上情况，则应使用 trim()函数截去字串后的空格。

9.3.2 数字型

在存储数字数据时，应该充分考虑数据的长度选择合适的类型进行存储，同时为数据的扩展保留一定的空间。

在 SQL 中使用 Small Int 类型需要注意数据项的范围。例如，想看月销售总额，总额字段类型是 Small Int，如果总额超过 32767，不宜进行计算操作。

9.3.3 日期和时间

a）系统时间

由数据库产生的系统时间首选数据库的日期型，如 DATE 类型。

b）外部时间

由数据导入或外部应用程序产生的日期时间类型采用 varchar 类型，数据格式采用：YYYYMMDDHH24MISS。

9.3.4 大字段

如无特别需要，避免使用大字段（blob，clob，long，text，image 等）。

9.3.5 唯一键

对于数字型唯一键值，尽可能用系列 sequence 产生。

10. 数据库运用

10.1 数据处理过程

10.1.1 概述

对数据库设计来讲，数据字典是进行数据收集和数据分析所获得的主要成果。数据字典是各类数据描述的集合。

数据字典通常包括数据项、数据结构、数据流、数据存储和处理过程五个部分。

10.1.2 数据项

数据项是不可再分的数据单位。对数据项的描述通常包括以下内容：

数据项描述＝｛数据项名，数据项含义说明，别名，数据类型，长度，取值范围，取值含义，与其他数据项的逻辑关系｝。

其中取值范围、与其他数据项的逻辑关系定义了数据的完整性约束条件，是设计数据检验功能的依据。

10.1.3 数据结构

数据结构反映了数据之间的组合关系。一个数据结构可以由若干个数据项组成，也可以由若干个数据结构组成，或由若干个数据项和数据结构混合组成。对数据结构的描述通常包括以下内容：

数据结构描述＝｛数据结构名，含义说明，组成:｛数据项或数据结构｝｝。

10.1.4 数据流

数据流是数据结构在系统内传输的路径。对数据流的描述通常包括以下内容：

数据流描述＝｛数据流名，说明，数据流来源，数据流去向，组成:｛数据结构｝，平均流量，高峰期流量｝。

其中数据流来源是说明该数据流来自哪个过程，数据流去向是说明该数据流将到哪个过程去，平均流量是指在单位时间（每天、每周、每月等）里的传输次数，高峰期流量则是指在高峰时期的数据流量。

10.1.5 数据存储

数据存储是数据结构停留或保存的地方，也是数据流的来源和去向之一。对数据存储的描述通常包括以下内容：

数据存储描述＝｛数据存储名，说明，编号，流入的数据流，流出的数据流，组成:｛数据结构｝,数据量，存取方式｝。

其中数据量是指每次存取多少数据，每天（或每小时、每周等）存取几次等信息。存取方法包括批处理、联机处理、检索、更新、顺序检索、随机检索等。另外，流入的数据流要指出其来源，流出的数据流要指出其去向。

10.1.6 处理过程

数据字典中只需要描述处理过程的说明性信息，通常包括以下内容：

处理过程描述＝｛处理过程名，说明，输入:｛数据流｝，输出:｛数据流｝,处理：｛简要说明｝｝。

其中简要说明中主要说明该处理过程的功能及处理要求。功能是指该处理过程用来做什么（而不是怎么做），处理要求包括处理频度要求，如单位时间里处理多少事务，多少数据量；响应时间要求等。这些处理要求是后面物理设计的输入及性能评价的标准。

10.2 安全保密设计

10.2.1 概述

说明在数据库的设计中，将如何通过区分不同的访问者、不同的访问类型和不同的数据对象，进行分别对待而获得的数据库安全保密的设计考虑，具体数据库设计安全设计要求请参照“信息安全标准”。

10.2.2 访问安全性设计

访问安全性可以从以下几点设计：

——锁定或者失效默认用户；

——修改可用用户的默认密码；

——限制操作系统存取权限；

——定期更新厂家推出的安全性补丁；

——采用多种用户访问审计（如标准审计、系统级触发器审计、细精度审计等）。

通过角色（Role），权限（Privilege）等的一系列授予（Grant）和回收（Revoke）功能控制用户的权限。

10.2.3 数据安全性设计

数据安全性可以从以下几点设计：

——采用 RAID 磁盘阵列，保证数据的安全性；

——数据库归档（ARCHIVE）模式；

——定期的数据全量备份策略；

——本地增量备份策略；

——异地远程实时容灾备份策略。

11. 数据库性能优化

11.1 概述

通过集群、分库、分区设计，增加数据库的容错和恢复能力，提高数据库的信息，在最大程度上保证系统的正常工作和崩溃后的迅速恢复。

11.2 数据集群

采用并行数据库集群技术，提高性能和容错功能，保证系统最大的正常工作时间。

11.3 表分区设计

对于海量的数据表，采用表分区（PARTITION）方式，可以：

a）增强可用性：如果表的一个分区由于系统故障而不能使用，表的其余好的分区仍然可以使用。

b）减少关闭时间：如果系统故障只影响表的一部分分区，那么只有这部分分区需要修复，故能比整个大表修复花的时间更少。

c）维护轻松：如果需要重建表，独立管理每个分区比管理单个大表要轻松得多。

d）均衡 I/O：可以把表的不同分区分配到不同的磁盘来平衡 I / O 改善性能。

e）改善性能：对大表的查询、增加、修改等操作可以分解到表的不同分区来并行执行，可使运行速度更快。

f）分区对用户透明，最终用户感觉不到分区的存在。

11.4 索引的使用

根据业务系统应用的实际情况创建相应的索引，对于 OLTP 系统，可采用 B-Tree 索引，对于 OLAP 系统，可采用 Bit Map 索引，对于海量数据，可对索引进行分区。

OLAP 与 OLTP 型业务尽可能分时或分库，以避免 OLAP 型业务影响 OLTP 型业务的处理效率。

a）在经常进行连接，但是没有指定为外键的列上建立索引，而不经常连接的字段则由优化器自动生成索引。

b）在频繁进行排序或分组（即进行 group by 或 order by 操作）的列上建立索引。

c）在条件表达式中经常用到的不同值较多的列上建立检索，在不同值少的列上不要建立索引。比如在雇员表的“性别”列上只有“男”与“女”两个不同值，因此就无必要建立索引，因为建立索引不但不会提高查询效率，反而会严重降低更新速度。

d）如果待排序的列有多个，可以在这些列上建立复合索引（compound index）。

11.5 存储过程

存储过程则不需要编译就能直接执行，因此速度可以更快。

建立存储过程的原则：

a）在频繁使用的 SQL 语句建议使用存储过程。

b）存储过程宜使用 SQL 自带的返回参数，不宜自定义的返回参数。

c）减少不必要的参数，避免数据冗余。

11.6 游标

想要依次遍历一个记录集的唯一方法就是使用系统游标，在使用完成之后应及时关闭和销毁游标对象释放用到的资源。并且不在万不得已的情况下，不宜随意使用游标，因为游标会占用较多的系统资源，尤其是对于大并发量的情况下，很容易使得系统资源耗尽而崩溃。

11.7 事务处理

在存储过程中如果遇到需要同时操作多个表的情况，宜避免在操作的过程中由于意外而造成的数据的不一致性。要将操作多个表的操作放入到事务中进行处理。

a）为避免引发事务的非正常错误，保障保证数据的一致性，不宜在事务中使用 return 语句强行退出。

b）一旦将多个处理放入事务当中，系统的处理速度会有所降低，所以应当将频繁操作的多个可分割的处理过程放入到多个存储过程当中，这样会大大提高系统的响应速度，但前提是不违背数据的一致性。

11.8 活动库与历史库的使用

对于数据量随时间不断增长的海量数据库，如查询日志、话单等信息，应考虑采用活动库与历史库分开设计，提高系统访问即时数据的效率。

12. 数据库外部设计

12.1 标识符和状态

联系用途，详细说明用于唯一地标识该数据库的代码、名称或标识符，附加的描述性信息亦要给出。如果该数据库属于尚在实验中、尚在测试中或是暂时使用的，则要说明这一特点及其有效时间范围。

12.2 使用它的程序

列出将要使用或访问数据库的所有应用程序，对于这些应用程序的每一个，给出它的名称和版本号。

12.3 约定

程序员或系统分析员使用数据库时，需要对如何建立标号、标识等约定内容做出陈述，例如用于标识数据库的不同版本的约定和用于标识库内各个文卷、记录、数据项的命名约定等。

12.4 专门指导

向准备从事此数据库的生成、测试、维护人员提供专门的指导，例如将被送入数据库的数据的格式

和标准、送入数据库的操作规程和步骤，用于产生、修改、更新或使用这些数据文卷的操作指导。

12.5 版本控制

12.5.1 概述

系统的数据库会因为需求的变更而不断的迭代，形成不同的版本，通过标准化的版本管理，可以有效的提高系统的开发和维护效率，降低系统维护风险。

介绍同此数据库直接有关的支持软件，如数据库管理系统、存储定位程序和用于装入、生成、修改、更新数据库的程序等。说明这些软件的名称、版本号和主要功能特性，如所用数据模型的类型、允许的数据容量等，列出这些支持软件的技术文件的标题、编号及来源。

12.5.2 设计方法

采用标准化的开发工具对数据库的结构进行设计，并以此为依据，每一次的数据库结构变更都要对设计文件进行维护，标明版本并对更新部分作具体说明。

12.5.3 版本控制

将每一个里程碑和一次重大修改作为一个数据库版本的确定标准，版本号的命名方式如下：XXXXX_时间_版本号。

一个确定发布的数据库版本由数据库设计文件、数据库文件（包含数据表信息和数据信息的文件）以及版本说明文件构成，统一压缩成以标准命名方式命名的压缩文件。

对数据库版本要采用规范的版本控制软件（如 CVS 等）进行版本控制，避免的数据库更新丢失和覆盖的情况发生。

13. 数据接口

13.1 概述

数据库访问服务中的数据操作接口是用来实现对于数据库中数据资源的各项操作，包含数据操作请求（Manipulation Request）与数据操作响应（DAI_Manipulation Response）。通过该接口将各种操作命令传递到信息资源数据库，在数据库中完成查询数据操作、插入数据操作、更新数据操作、删除数据操作和用户自定义的其它操作。

当数据库访问服务接收该数据操作请求后，连接相应的数据库，然后让数据库存储执行这个数据操作。数据操作执行完毕，数据库访问服务会生成 XML 报告文档（指明数据操作完成状态）并返回相应的操作结果。一旦数据操作发生异常时，通过异常响应返回具体的异常情况报告。

13.2 基本框架

13.2.1 概念模型

异构数据库访问的概念模型如图 3 所示，访问过程由数据访问客户端和数据访问服务器间交互操作实现，客户端向服务器发送规范的数据访问请求，服务器接收数据访问请求并通过调用底层数据库访问接口完成服务请求处理，并将处理结果返回客户端。

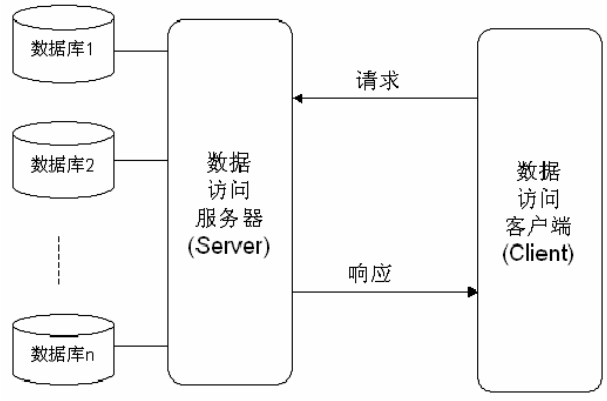

图 3：异构数据库接口概念模型

本规范定义了接口服务机制，为客户端实现对各种分布式异构数据库进行透明访问提供了保证。

13.2.2 结构框架

结构框架说明异构数据库访问接口在政务信息资源访问过程中的作用。结构框架主要由数据库、数据访问接口、异构数据库服务接口组成，如图 4 所示。

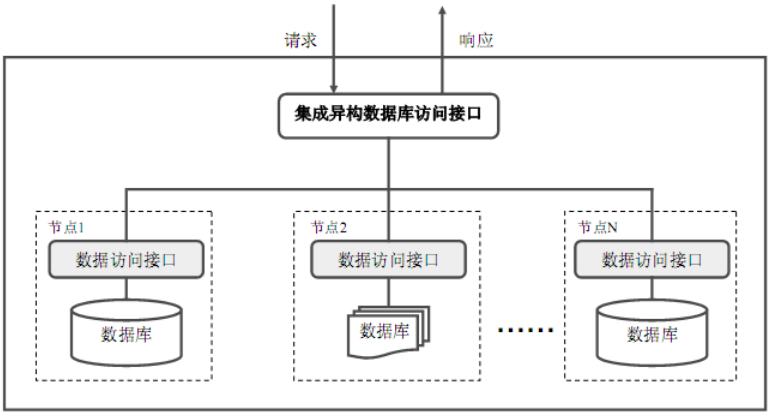

图 4：异构数据库访问接口结构框架

a）异构数据库服务接口：规定和描述访问异构数据库的一组服务，服务包括服务自描述、数据类型描述、服务操作及服务数据元素。

b）数据访问接口：根据数据库的异构特征，将上层服务转换为信息访问与调用服务的各种访问驱动及服务接口映射实现。

c）数据库：包括关系型数据库、文件数据库，这些数据库可以是集中的，也可以是分散的。

13.3 接口定义规范

13.3.1 基本结构概述

异构数据库接口包括数据操作、数据描述规范两个部分。数据操作接口定义数据库访问的四种主要行为及数据库操作的响应结果，包括获取数据、新增数据、更新数据、删除数据以及操作响应。数据描述规范用来规范数据库访问行为及数据操作响应中数据描述的格式。

13.3.2 数据描述规范

数据描述规范通过 XML 描述结构化交换信息数据集的结果以及具体的时间内容。

数据描述通过数据包来表示。数据包可以包含多个数据，每一个数据由数据集进行描述，每一个数据都有属性 context 用于描述结构的上下文环境，一个数据集可以包含多个记录数据，一个数据记录可以包含多个单元数据，每一个数据单元由数据名称、数据值、数据类型组成。

13.3.3 数据操作接口

13.3.3.1 简介

数据操作分成数据请求与数据反馈。数据请求分为查询数据、插入数据、更新数据、删除数据四种行为，分别用 Query、Insert、Update、Delete 表示。数据反馈用 Response 表示，当操作发生异常时，通过异常反馈。

13.3.3.2 请求

a）查询数据

查询数据操作（Query）允许用户从数据库中获取指定的数据，获取操作中的 Operate 应指定操作的属性 ID，以便于在响应操作中进行关联。

1）获取操作中应指定 From：从什么对象中获取。

2）获取操作中应指定 Return：指定返回的数据结构以及返回的数据最大个数，最大个数通过指定属性 maxCount 值来标识，数据结构通过 Structure 来说明，数据结构可以引用一个定义的 Schema，或者可以进行自定义数据结构。

3）获取操作可以指定 Conditions：获取的条件，条件可以多个，如果不指定则选取全部数据。

——第一个条件应指定属性 isFirst 为“True”，Relate 应为空；

——非第一个条件应指定属性 isFirst 为“False”，Relate 应为“AND”或者“OR”；

——条件应指定 Op 操作类型（“EQUAL”等于、“GREATE”大于、“LESS”小于、“GREATE_EQIAL”大于等于、“LESS_EQUAL”小于等于）；

——条件应指定 DataName，数据的名称，根据具体的情况定义；

——条件应指定 DataValue 数据的值，数据的值来源于下面定义的数据集的值或者是一个固定值，如果来源于数据集的值，应用 XPath 来进行表达。

b）新增数据

新增数据操作（insert）允许用户向数据库中存入指定的数据，存入操作中的 Operate 应指定操作的属性 ID，以便于在响应操作中进行关联。

1）新增操作中应指定 To：存入的对象。

2）新增操作中应指定 Data：新增的数据，数据来源于下面定义的数据集，应用 Xpath 来进行表达。

c）更新数据

更新数据操作（Update）允许用户更新数据库指定的数据，更新操作中的 Operate 应指定操作的属性 ID，以便于在响应操作中进行关联。

1）更新操作中应指定 To：存入的对象。

2）更新操作中应指定 Data：新增的数据，数据来源于下面定义的数据集，应用 Xpath 来进行表达。

3）更新操作可以指定 Conditions：获取的条件，条件可以多个。

——第一个条件应指定属性 isFirst 为“False”，Relate 应为“AND”或者“OR”；

——条件应指定 Op 操作类型（“EQUAL”等于、“GREATE”大于、“LESS”小于、“GREATE_EQIAL”大于等于、“LESS_EQUAL”小于等于）；

——条件应指定 DataName，数据的名称，根据具体的情况定义；

——条件应指定 DataValue 数据的值，数据的值来源于下面定义的数据集的值或者是一个固定值，如果来源于数据集的值，应用 XPath 来进行表达。

d）删除数据

删除数据操作（Delete）允许用户删除数据库指定的数据，删除操作中的 Operate 应指定操作的属性 ID，以便于在响应操作中进行关联。

1）删除操作中应指定 To：存入的对象。

2）删除操作中应指定 Data：新增的数据，数据来源于下面定义的数据集，应用 Xpath 来进行表达。

3）删除操作可以指定 Conditions：获取的条件，条件可以多个。

——第一个条件应指定属性 isFirst 为“False”，Relate 应为“AND”或者“OR”；

——条件应指定 Op 操作类型（“EQUAL”等于、“GREATE”大于、“LESS”小于、“GREATE_EQIAL”大于等于、“LESS_EQUAL”小于等于）；

——条件应指定 DataName，数据的名称，根据具体的情况定义；

——条件应指定 DataValue 数据的值，数据的值来源于下面定义的数据集的值或者是一个固定值，如果来源于数据集的值，应用 XPath 来进行表达。

13.3.4 响应

响应（Response）是指在用户进行请求操作后，服务返回操作结果给用户，响应分为成功响应和异常响应。

响应指定属性 RefToOperate 为请求操作的 Operate 的 ID 属性，用以关联请求服务。

响应中数据直接在响应内容之内，对获取数据请求直接内容为获取的数据，对于存入、更新、删除数据的请求，响应数据为操作成功与否已经成功的个数。

如果在对请求进行操作时有错误发生，则产生一个异常响应，异常响应应指定 Faults。

13.3.5 异常

当接收请求操作后，在实际操作数据的过程中出现了异常情况需要告知请求者时，可以通过构建一个异常响应的应用，用于描述异常发生的细节，以便于请求者做出判断。当出现网络或上层（非数据操作层）的异常时在此不作定义，由其他交换体系的其它层进行处理。

异常构建通过响应的 Faults 进行描述，Faults 主要由多个 Cause 进行描述，一个异常响应可以包括多个发生异常的原因。

当 Cause 描述无法清楚的表达时，用户可以根据情况进行相应的扩展。