

通用数据库管理系统(UDS)

子模式DDL 和 DML

航空工业部第六三一研究所

通用数据库管理系统(UDS)

子模式DDL和DML

罗秉松： 译 校

前 言

本手册是为系统分析员和程序员编写的，目的是使他们能够定义部分数据库和编写应用程序。

定义的部分数据库称为子模式，用来定义子模式的是子模式数据定义语言（子模式DDL）。为了编写COBOL应用程序，必须在COBOL中增加数据库部分，这一部分称为数据库操纵语言（DML）。

通常引用两种给定的语言，本手册后面将对每种语言给予详细描述。

本手册的最后一章，叙述CALL DML，即DBHC数据库处理程序）和不同的程序设计语言之间的接口。

附录I和附录II，叙述语言符号的规则。附录III给出各种状态代码和语句代码以及它们的组合。

子模式和COBOL-DML程序编译时产生的错误和警告信息在UDS系统和操作手册中给出。

模式DDL和子模式DDL是根据CODASYL DBTG 1971年的报告。DML是根据CODASYL COBOL 数据库研究1973年的提议。

当从子模式到DML中，读者应该注意到，用到的术语域（REALM）已改变为区（area），术语OPTIONAL和MANDATORY已由TRANSIENT和PERMANENT来代替。

目 录

1	子模式 DDL 引言	9
1.1	模式 / 子模式概念	9
1.2	一个子模式的范围和内容	10
1.3	子模式和模式之间的相似性和差别	11
1.4	在子模式中的 PRIVACY (保密) 锁和 PRIVACY 码	13
1.4.1	PRIVACY KEY FOR COPY 子句	13
1.4.2	PRIVACY LOCK FOR COMPILER 子句	13
2	DML 引言	14
2.1	FIND 语句	18
2.1.1	存取数据库中的记录	18
2.1.2	直接检索数据库中的记录	19
2.1.2.1	利用 CALC 码的直接存取 (FIND-2)	20
2.1.2.2	利用数据库码的直接存取 (FIND-1)	22
2.1.2.3	直接存取一个特定记录型的记录	23
2.1.2.4	直接存取一个象值内的记录 (FIND-7)	25
2.1.3	当前位置	29
2.1.4	利用当前位置的直接存取	35
2.1.4.1	FIND-5 语句	35
2.1.4.2	FIND-6 语句	37
2.1.5	顺序存取数据库内的记录	39
2.1.5.1	利用象的当前记录, 域的当前记录或者记录型的 当前记录的顺序存取 (FIND-4)	39
2.1.5.2	顺序存取重复 CALC 码的记录 (FIND-2)	43
2.1.5.3	顺序存取重复记录 (FIND-3)	44

2.1.6	具有析索表达式的 FIND-7 语句	47
2.1.6.1	FIND-7 语句的一般描述	47
2.1.6.2	在 FIND-7 语句中的析索表达式	55
2.1.7	FIND-7 语句的综述	61
2.2	系值的自动选择	64
2.2.1	THRU LOCATION MODE OF OWNER 选择方法	66
2.2.2	THRU CURRENT OF SET 选择方法	70
2.2.3	利用 SET OCCURRENCE SELECTION 子句中的 ALIAS 选择	72
2.3	GET 语句	78
2.4	FETCH 语句	79
2.5	STORE 语句	79
2.6	CONNECT 语句	81
2.7	ERASE 语句	84
2.8	DISCONNECT 语句	88
2.9	STORE, CONNECT, DISCONNECT 和 ERASE 语句的 综述	89
2.10	MODIFY 语句	91
2.10.1	改变数据项	92
2.10.2	改变系内的位置	93
2.10.3	改变系间的位置	93
2.10.4	MODIFY 语句的综述	95
2.11	数据的完全性：并行使用数据库	97
2.11.1	引言	97
2.11.2	在域层中的保护 (READY 和 FINISH 语句)	99
2.11.3	在块层中的数据保护	102
2.11.4	在记录层中的保护 (KEEP 和 FREE 语句)	103

2.11.5	独立数据库处理程序	105
2.11.6	内部连接数据库处理程序	106
2.12	DB条目	107
2.13	ACCEPT 语句	110
2.14	SET 语句	114
2.15	USE 语句	114
2.16	IF 语句	119
2.16.1	TENANCY 条件	120
2.16.2	MEMBER 条件	121
2.17	PRIVACY KEY 子句	121

3	子模式数据定义语言	122
3.1	概述	122
3.2	IDENTIFICATION DIVISION	124
3.2.1	SUB-SCHEMA 子句	125
3.2.2	PRIVACY LOCK 子句	125
3.2.3	PRIVACY KEY 子句	126
3.3	DATA DIVISION	127
3.3.1	REALM SECTION	127
3.3.2	RECORD SECTION	128
3.3.2.1	RECORD SECTION 条目	134
3.3.2.2	记录控制子条目	136
3.3.2.2.1	RECORD 名字句	136
3.3.2.3	数据描述子条目	137
3.3.2.3.1	数据场名字句	137
3.3.2.3.2	OCCURS 子句	138
4 — 3.3.2.3.3	PICTURE 子句	139

3.3.2.3.4	USAGE 子句 - - - - -	142
3.3.2.4	条件名条目 - - - - -	145
3.3.2.4.1	条件名字句 - - - - -	145
3.3.2.4.2	VALUE 子句 - - - - -	145
3.3.3	SET SECTION - - - - -	147
4	数据操纵语言 (DDL) - - - - -	148
4.1	概述 - - - - -	148
4.2	DATA DIVISION - - - - -	149
4.2.1	DB 条目 - - - - -	150
4.3	PROCEDURE DIVISION - - - - -	151
4.3.1	ACCEPT - - - - -	151
4.3.2	CONNECT - - - - -	153
4.3.3	DISCONNECT - - - - -	155
4.3.4	ERASE - - - - -	157
4.3.5	FETCH - - - - -	160
4.3.6	FIND - - - - -	161
4.3.7	FINISH - - - - -	177
4.3.8	FREE - - - - -	177
4.3.9	GET - - - - -	178
4.3.10	IF - - - - -	179
4.3.11	KEEP - - - - -	181
4.3.12	MODIFY - - - - -	182
4.3.13	READY - - - - -	187
4.3.14	SET - - - - -	189
4.3.15	STORE - - - - -	190
4.3.16	USE - - - - -	195

4.4	一个COBOL DML 用户程序的生成 - - - - -	196
5	CALL DML - - - - -	198
5.1	综述 - - - - -	198
5.2	CALL DML和COBOL DML 的主要差别 - - - - -	198
5.3	CALL 接口 - - - - -	200
5.3.1	规则 - - - - -	203
5.4	DML CALL 的描述 - - - - -	203
5.4.1	CALL DML 操作数 - - - - -	204
5.4.2	接口表 - - - - -	229
5.4.3	格式规则 - - - - -	241
5.4.4	格式表 - - - - -	243
5.4.5	用户位恩(第4个操作数)和数据库系统缓冲 区 - - - - -	245
5.4.6	专用选择(第3个操作数) - - - - -	250
5.4.7	专用参数1(第10个操作数):保留 - - - - -	252
5.5	CALL DML 汇编程序宏指令 - - - - -	254
5.5.1	操作数方式 - - - - -	255
5.5.2	DSCAL - - - - -	256
5.5.3	DSCAP - - - - -	256
5.5.4	DSCDF - - - - -	257
5.5.5	DSCPA - - - - -	259
5.6	异步程序的要求(UDS BS1000) - - - - -	260
5.7	流程图 - - - - -	263
5.8	LOOKC - - - - -	267
5.8.1	功能描述 - - - - -	267
5.8.2	LOOK 操作数的描述 - - - - -	270

5.8.3	在记录区中建立 200K 块	271
5.8.4	功能选择和专用选择的可能组成	275
5.9	错误处理	287
5.9.1	错误信息	288
5.10	CALL DML 用户程序的生成	294

6	连接 DML 用户程	295
6.1	UDSLI (UDS 连接宏指令)	295
6.1.1	可能允许的操作数的组合	298
6.2	执行	299
6.3	例子	299

附录 I 子模式 DDL 的符号和规则

1	字符集	302
2	字	302
3	直接旁	302
4	名字	304
5	标点符号	304
6	子模式 DDL 的一般格式的符号	305
7	子模式 DDL 的源记录格式	306
8	子模式 DDL 的错误信息和警告信息	306

附录 II DML 的符号和规则

1	字符集	307
2	字	307
3	名字	308
4	标点符号	308
5	DML 的一般格式的符号	308

6	DML源记录格式	309
7	错误信息和警告信息	309

附录Ⅲ 语句代码和状态代码

1	数据库专用寄存器	310
1.1	一般描述	310
1.2	DATABASE—STATUS	310
1.3	DATABASE—REALM—NAME	310
1.4	DATABASE—SET—NAME	311
1.5	DATABASE—RECORD—NAME	311
2	数据库状态指示器	312
2.1	一般描述	312
2.2	语句代码和状态代码	312
2.3	语句代码和状态代码的组合	312
2.4	用于数据库异常条件的专用寄存器	312
2.5	DATABASE—REALM—NAME	313
2.6	DATABASE—SET—NAME	313
2.7	DATABASE—RECORD—NAME	313
2.8	状态代码的分类	313

1 子模式DDL引言

1.1 模式/子模式的概念

在模式中，利用模式DDL对一个特定的环境定义完整的逻辑数据结构。但是，程序一般使用的数据只是与这个环境的一部分数据有关，而且，这部分数据也只是涉及数据库中的某一部分。在UDS中，实现数据描述的第二级就是描述数据库的某一部分，这一部分数据是应用程序实际使用的。这种描述是利用子模式数据定义语言（子模式DDL）来实现。通过这种语言描述的数据库的这一部分，称为子模式。

事实上，程序员的复杂情况（作业）在总的数据库的描述中实际上不是现有的，但这不是要引入第二级的数据描述的主要原因。

首先，UDS是一个主要语言系统和在一个现有的程序设计语言中嵌入把记录存入数据库中和修改或者检索数据库中的记录的语句。在UDS V2中，这种语言是COBOL语言，这种主语言的数据处理的能力必须适用于模式DDL所描述的记录的数据结构。当实现二级描述时，总的描述（模式DDL）与主语言无关，并且下一级的描述起着主语言和总的描述之间的接口的作用。

第二，当一个特定的程序仅仅对数据库的某一部分起作用时（这一部分由子模式来定义），则数据库的其余部分被保护起来。只有调用的子模式所描述的数据被存取，而其余的数据自动的保护起来。除了这些保护措施以外，也有提高效率的作用，只有实际存取的域才需要联机。这个数据库（数据库的某部分）比存放总的数据库的磁盘组要小。

第三是模式/子模式的设计容许它们之间的差别。例如，在模式中定义的某些数据项在一个子模式中可以省略。这

样，每个记录型的数据可以保护起来。当某些数据项描述改变时，则不包含这些数据项的子模式不受影响，因而调用这些子模式的程序也不受影响。

由于子模式概念的引进，使它在一定程度上与逻辑数据定义的改变无关。

1.2 子模式的范围和内容

通过一个子模式的描述，模式的一部分被分离出来。一个子模式将为调用它的所有应用程序形成共有的最大工作域。确定一个子模式的规模是由物理上的限制和逻辑上的考虑这两方面来决定的。在一个特定的计算机中，能够同时联机的磁盘组数是由磁盘机的个数来确定。这样就限制了一个程序能够存取的磁盘空间的大小。

在子模式中所描述数据库中的一部分的大小归根到底依赖于这些物理上的限制。子模式的描述是使用户程序实际需要数据库适应于这些限制的一种方法。实际上，应用程序一般起沉于一个特定的子系统（如开发票，发放工资，物资管理等），因此，只利用到数据库中的部分数据。

一个应用程序用到的记录型和系将形成逻辑上和部分的数据库相关，这一部分将由子模式来描述。因而，一个子模式不仅必须包含记录型，系和域，而且还必须涉及到应用程序。某些 DML 语句的执行与某些记录、系和域不是直接有关，而是间接地得到某些结果。例如，当一个记录从数据库中删去时，有时，由于这种操作而引起它的若干成员记录也被删去，等等。

删除某一个记录时，则与它相关的记录型，系和域也必须包含在这个程序调用的子模式内。

用户按照模式可以根据需要定义许多子模式。子模式之间允许有重叠部分，它们的说明彼此不相互影响。一般地说，子模式必须逻辑上与模式的子集相一致。在记录型层内，这意味着在子模式中对记录的描述必须是模式中对记录描述的一个子集，并且必须至少包含模式中定义的一个初等数据项。

对于每个记录型，子模式必须至少包含存放这个特定记录型的记录的一个域。模式中对系的描述在子模式中不能改变。

在一个程序中，显式地或者隐含地涉及到的所有系必须复制到由这个程序调用的子模式中。

子模式用作对程序的一个数据描述是对所涉及数据库中的数据说的。

在一个程序中涉及到的其他数据成分，例如打印机文件，磁带文件，卡片文件以及它们的记录是由标准方式的COBOL—DML程序的DATA DIVISION来定义。

1.3 一个子模式和模式之间的相似性和差别

在讨论模式和由其导出的子模式之间的相似性和差别之前，先简单地叙述一下子模式DDL中采用的各术语。

初等项：是记录中有命名的数据的最小单位。等价对象在模式DDL中是数据项。

组项：一个或者多个数据项或数据集的一个命名序列。在模式DDL中没有严格等价的术语。只出现一次的重复组，有相同的特性，但是，在模式中，OCCURS子句中不允许只出现一次。

阵列：所有初等项的一维，二维或者三维的集合都有相同特性。对于一维阵列在模式DDL中的等同符号是向号。
重复组：出现任意次数的数据元素的集合。模式DDL中的等价的也称重复组。

除了和模式DDL有相同含义的术语记录型以外，用来表示数据成分上面那些术语也是相当重要的。

为了模式和子模式的记录描述之间等效性，在模式中定义的分层结构的一般规则在子模式中必须保留。这意味着子模式中说明模式中的一个初等项，则关于这个初等项的分层结构必须复制。从另一方面来说，当在子模式中复制一个分层元素时，则这个分层元素必须至少有一个下一层的初等项的一个元素。模式和子模式之间的相同性和差别的其他规则是：

- 在一个子模式中，可以省略模式中定义的某些数据项和数据集。在模式中的一个数据项必须在子模式中作为一个初等项来定义。子模式中定义初等项的次序必须对应于子模式中定义数据项的次序。
- 在子模式描述中，必须至少涉及到一个初等项。
- 在子模式中可以增加组项。初等项，阵列，重复组和组项可以组合起来，形成新的组项。
- 子模式中的一个一维阵列必须对应于模式中的一个向号。在子模式中，向号元素的个数可以比模式中的向号元素的个数少。
- 在子模式中，二维和三维阵列分别对应于模式中的一次或者二次重复组的从属向号或者分别对应于一次或二次重复组的从属数据项。这意味着子模式中的一个阵列的维数必须对应于模式中从属的数据项的OCCURS子句的数目（包括数据项本身可能有的OCCURS子句）。

在 OCCURS 子句中可以给三层，而每层上的元素，其出现的次数可以减少，甚至可以减少到 1。

——在子模式中的重复组对应于模式中的重复组。在子模式中的 OCCURS 子句可以减少出现的次数，最少的次数是 1。如果一个重复组从模式中省略，则在子模式中对包含的从属元素可以不用说明。

利用子模式 DDL 的子句来描述数据元素的方式完全类似于通常的 COBOL 对数据元素的说明。

1.4 在子模式中的保密锁和保密码

1.4.1 PRIVACY KEY FOR COPY 子句

在 UDS 中，一种保密措施是用来保护一个模式，避免未经许可的一个子模式对它进行编辑。如果一个模式用这种方式来保护，则在子模式中必须给出 PRIVACY KEY FOR COPY 子句。

每当一个子模式从一个被保护的模式中得到编辑的时候，则必须提供匹配的密钥。为了给这个密钥，在子模式 DDL 的 IDENTIFICATION DIVISION 中必须包括一条 PRIVACY KEY FOR COPY 子句。

1.4.2 PRIVACY LOCK FOR COMPILE 子句

和防止一个模式的非法编辑来保护一个模式一样，一个子模式在 COBOL-DML 程序编译期间进行保护。在子模式中给一个保密锁。为了在一个 COBOL-DML 程序的编译期间允许利用这个子模式，这个程序必须有一个密钥值和这个保密锁相匹配。

——为了描述这个保密锁，在子模式的 IDENTIFICATION DIVISION 中使用 PRIVACY LOCK FOR COMPILE 子句。

2 DML 引言

利用模式数据库定义语言 (DDL) 定义的模式，给出数据库的所有用户共同需要的所有记录型和象。而子模式数据库定义语言是用来定义模式的一部分数据，用这种方式得到的数据定义，即子模式，可以由 COBOL 书写的应用程序来使用。为了编制应用程序，除了 COBOL 以外，程序员必须配置一种语言，用这种语言能够把记录值存入数据库或者从数据库检索记录值，这种语言也允许程序员利用记录之间的关系式（例如，当检索数据库内的记录时）。这种语言必须是一种独立的语言。根据 CODASYL 的数据库任务组的报告，首先则上，一个数据库管理系统不能面向任何一种程序设计语言，但是，它必须容许采用几种语言。

在 UDS 中，实现这些功能的语言称为数据操纵语言 (DATA MANIPULATION LANGUAGE, 编写为 DML)。DML 本身并不是一个完整的语言。这种语言的语句允许用户从数据库中检索记录，存入新的记录和修改或者删去已有的记录。这种操作考虑到记录之间的现有的关系和记录之间的已有的定义。而且 DML 语句不仅适用于把应用程序的记录转换到数据库中的记录，并且反过来也一样。通过一个现有的程序设计语言的功能实现对数据的更进一步的处理。因此，DML 认为是通用的程序设计语言的一种有力的扩充，特别是在数据库内，记录的存入和记录的检索。

在 UDS 中，DML 语句是程序设计语言 COBOL 的一种扩充。一个应用程序包含有 COBOL 语言和 DML 语句。

在一种通常的 COBOL 程序中，程序员利用 READ 和 WRITE 语言来检索文件内的记录并把记录存入文件。为了把记录从数据库转换到应用程序或者把应用程序中的记录转换到数

数据库，则应用程序不是包含 READ 和 WRITE 语句，而是包含 DML 语句。由于这些 DML 语句，记录能够在数据库和用户程序通伎区之间进行转换，这个用户通伎区称为用户工作区 (USER WORK AREA, 缩写为 UWA)。

这个 UWA 是在一个应用程序编译期间形成的。它由 COBOL-DML 的编译程序利用这个应用程序调用的子模式的伎恩来建立。UWA 包含在子模式中定义的每个记录型的一个记录值的存储空间。UWA 的这一部分称为记录区。

当通过一条 DML 语句把一个记录从数据库中转换到应用程序时，则这个记录存放在它的记录型的 UWA 中保留的空间上。在用户程序执行期间，UWA 是这个程序的一部分。

在 UWA 中的记录和它的初等项由 COBOL 语句来存取，例如，准备打印取面初等项的值或者修改初等项的值。在 UWA 中的记录和初等项是子模式中定义的名字所产生的。

如果把一个记录存入数据库，则首先利用 COBOL 语句用适当的值充满 UWA 的记录区中的数据库数据项。然后利用 DML 语句 (STORE 语句) 把 UWA 中的这个记录转换到数据库中。在一个应用程序中，DML 语句和 COBOL 语句之间的不同在图 21 中说明。