1、时态关系的创建

按照时态特性不同，TSQL2 中相应的时态关系分为如下四种类型。

① 快照时态关系，无显式的时间标签，即为常规的关系表。

② 有效时间时态关系。分为“状态”和“事件”两种情形。

- 有效时间状态关系：表示状态（STATE）,具有描述“状态”有效的有效时间，通常时间标签为时间期间，其说明语句为 AS VALID[STATE],此时 STATE 为可选项，也是默认值，实际应用中的 STATE 选与不选效果相同。
- 有效时间事件关系：表示事件（EVENT）,描述事件发生的确定时刻，通常有效时间为时刻的集合，其说明语句为 AS VALID EVENT。

③ 事物时间时态关系。只有事物和时间标签，其说明语句为 AS TRANSACTION

④ 双时态关系。分为“状态”和“事件”两种情形。

- 双时态状态关系。时间标签含事务时间和有效时间，其中有效时间描述关系表示的状态有效的期间，其说明语句为 AS VALID [STATE] AND TRANSACTION。
- 双时态事件关系。时间标签含事物时间和有效时间，但时间标签中的有效时间描述关系表示的事件发生时刻的集合，其说明语句为 AS VALID EVENT AND TRANSATION。

例 1 双时态关系“处方”的创建语句如下。

```
CREATE TABLE 处方(病人姓名 char(10),
                  医生姓名 char(10),
                  药名 char(30),
                  剂量 char(30),
                  服药间隔 INTERVAL MINUTE)
AS VALID [STATE] DAY AND TRANSACTION;
```

说明：语句中以 AS 开头的字句表明所建立关系是双时态关系，包含有效时

间和事物时间两个时态属性。有效时间用时间期间表示，粒度为天；事物时间粒度由系统决定。属性“服药间隔”表示每隔多少分钟服药一次。

2、不带时间条件的查询

按照关系查询中 WHERE 子句是否出现时间条件，TSQL2 中的查询可以分为不带时间条件和基于时间条件查询两种情形。

如果查询中 WHERE 子句中不显示出现时间条件，则称其为不带时间条件的查询，但此时并不意味着查询结果中不出现相应的时间标签。

(1)SNAPSHOP 运用

一般时态关系数据库需要兼容常规关系数据库。在 TSQL2 中，通过在 SELECT 子句之后添加关键词 SNAPSHOT 实现这一机制。当查询语句中出现关键字 SNAPSHOTS 时，系统就将转换到常规关系数据库中查询，输出没有时间标签的查询结果。

例 2 查询所有服用过红霉素药物的病人姓名。

```
SELECT SNAPSHOT 病人姓名
FROM 处方
WHERE 药名=’红霉素’;
```

说明：查询语句中出现 SNAPSHOP,表明查询结果不需要时态标签，查询输出为病人姓名列表。

例 3 查询曾服用过红霉素的病人姓名及服药期间。

```
SELECT 病人姓名
FROM 处方
WHERE 药名=’红霉素’;
```

说明：查询结果为所有服过红霉素的病人姓名及其服药期间的集合。如果病人连续服用此药，即集合中有若干个服药期间首尾相连，TSQL2 将这些服药期间自动归并为一个更大的时间区间，即实现”时间归并”。

例 3 查询与红霉素同时服用的其他药品、服用此药病人姓名以及同时服药

的期间。

SELECT P1.病人姓名，P2.药名

FROM 处方 AS P1,处方 AS P2

WHERE P1.药名='红霉素'

AND P2.药名<>'红霉素'

AND P1.病人姓名=P2.病人姓名;

说明：在 FROM 子句中，为时态关系“处方”定义了两个元组变量 P1 和 P2。这些元组变量名通常称为关联名。关联名 P1、P2 除满足 WHERE 子句要求外，还需满足默认条件：P1、P2 所表示元组在有效时间上应有重叠，查询结果中有效时间就是 P1、P2 的重叠部分。

（2）时态关系重构

对时间期间进行运算通常得到的是一般的时间元素，即时间期间关于时态运算并不封闭，所以 TSQL2 中时态元祖的时间标签通常设定为一般时间元素，即由时间期间、时刻点等组成的集合。TSQL2 需要兼容常规关系，常规关系中原子性要求不能存在两个或两个以上的时态标签不同而非时态属性值相同的元祖。为此引入时态关系重构机制。

设有时态关系模式 TR（A1,A2,...,Ak），其中 A1,...,Ak 为 TR 中相应的 K 个非时态属性。TR 中每个元祖后面带有时间标签，但为了表述方便可隐去。若对 TR 进行关于属性 A1,A2 的投影操作，相应查询语句如下：

SELECT A1,A2 FROM TR;

查询结果为一个新关系 TRO(A1,A2),此时可能会产生时间标签不同而非时态属性完全相同的元祖。这种由投影操作带来的需要构造新的满足原子性要求的机制就是时态关系重构，也就是说，在 FROM 子句中，允许使用投影对“输入”的时态关系进行了时态归并，这相当于在 FROM 子句中重构新的时态关系。

例 4 查询服药必有红霉素的病人姓名。

SELECT SNAPSHOT P1.病人姓名

FROM 处方(病人姓名)AS P1,P1(药品)AS P2

WHERE P2.药名=‘红霉素’

AND VALID(P2)=VALID(P1);

说明：在 SELECT 子句中使用 SNAPSHOT,查询结果为只有病人姓名的快照关系。FROMZ 子句中创建两个重构时态关系 P1 和 P2。P1 是“处方”关系在“病人姓名”上投影，其时间标签是病人所有服药期间的集合，即不管处方是哪个医生开的，是什么药，以及剂量和服药间隔如何。P2 由 P1 加“药名”重构而成，相当于“SELECT 病人姓名，药名 FROM 处方”。通过 P1 而不是原有关系“处方”重构 P2 可减少归并时间开销，而在应用条件“VALID(P2)=VALID(P1)”时，隐含着“条件 P1.病人姓名=P2.病人姓名”。P2 中时间标签是指病人服某种药的期间集合。在 WHERE 子句中有限制 P2 的药名为“红霉素”的条件，因而 VALID(P2)是指病人服用“红霉素”的期间集合，而 VALID(P1)是病人服用任意药的集合。VALID(P2)=VALID(P1)说明该病人凡服药期间必有”红霉素”。

（3）时态关系划分

在 TSQL2 中，FROM 子句中关系的时间标签通常是一般时间元素，即由时间期间（状态有效期间）或者时间点（事件发生时刻）组成的集合，这样的时态关系元祖可表示为：

（a1,a2,…,an）,{p1,p2,…,pm}（3.1）

其中（a1,a2,…,an）为元组属性，{p1,p2,…,pm}为时间标签，p1,p2,…,pm 为时间期间或时刻点。在实际应用中为查询方便，需将（3.1）改为（3.2）的形式：

（a1,a2,…,an）， p1

（a1,a2,…,an）， p2

…… …

（a1,a2,…,an），　pm　　(3.2)

即将一个元组变为 m 个元组。这种变换机制即是时态关系划分（partition）。需要注意，上述时态关系划分违背常规关系的原子性要求，出现了一个以上除时间标签相异其它部分相同的元组。为此，TSQL2 只允许时态关系划分出现在 FROM 子句而不能出现在查询结果中。当将某关系进行时态划分时需在其后加关键词“（PERIOD）”。时态关系划分通常用于具连续时间条件的查询。

例 5 查询连续服用同一种药超过 6 个月的病人姓名、药名和服药期间。

```
SELECT SNAPSHOT 病人姓名，药名，VALID(P)
FROM 处方(病人姓名，药名)(PERIOD)AS P
WHERE CAST(VALID(P)AS INTERVAL MONTH)>INTERVAL’6’MONTH;
```

说明：处方关系表先对病人姓名、药名两个属性重构，对应着（病人姓名，药名）二元组，此时，每个二元组有一个形如（3.1）的时间标签。然后对重构后的关系进行划分，得到时态关系 P。P 中每个元组时间标签为单一时态元素——时间期间，而非时态元素的集合，以此时间标签判断是否连续服某药超过六个月。查询中出现的 CAST 是一个转换函数，用于将时间期间转换为以月为单位的间隔。SELECT 子句中出现关键词 SNAPSHOT，因此查询结果中无时间标签。为表示连续服药的实际时间. 以 VALID(P)作为查询结果的第三个属性。查询结果中，同一病人姓名和药名可能对应多个元组，每个元组有不同的期间。如果在 SELECT 子句中不用保留字 SNAPSHOT，也可得到类似的结果。

3.基于时间条件查询

如果查询中出现关于有效时间或事务时间的相关条件，就称这样的时态关系查询称为基于时间条件查询。

（1）有效时间查询 查询中，如需限制输出结果中有效期间的范围，可使用 VALID 子句。

例 6 在 2013 年，医生给 John 开了哪些药?

SELECT 药名 VALID INTERSECT(VALID(处方),PERIORD

‘[2013-1-1, 2013-12-31]’)

FROM 处方

WHERE 病人姓名==‘John’；

说明 查询结果是 John 在 2013 年曾经服过的药名，每个药名后附有一个时间标签，它是王敏在 2007 年服用此药的期间集合。如果没有 VALID 子句，则查询的结果将是 John 曾经服过的所有药名，而每种药名所附的时间标签是 John 服用此药的期间的集合。

（2）事务时间查询 TSQL2 基于在双时态数据模型 BCDM，因此事务时间应是查询的条件之一。事务时间是系统时间，常规时态查询大多是基于数据对象的最新版本，设 D 为时态关系 TR 的元组，TSQL2 中以下述语句作为查询事务时间的缺省条件：

TRANSACTION(D) OVERLAPS（相当于 CONTAINS）
CURRENT-TIMESTAMP

例 7（缺省的事务时间查询）查询 John 的历次处方。

SELECT *

FROM 处方

WHERE 病人姓名＝‘John’；

说明 在此查询中，未显式地指明事务时间条件，因而按当前的事务时间查询。如果处方被修改过，则查询结果为最近修改过的处方版本。

如果在实际应用中需要查询数据对象的过往版本，就需要明确指定所限制的事务时间。

例 8 （指定了事务时间的查询条件）查询自 2007 年 10 月 1 日所看到的 John 的历次处方的版本

SELECT *

FROM 处方 AS P

WHERE 病人姓名＝‘王敏’ AND TRANSACTION(P) OVERLAPS DATE

‘2013—10-01’

例 9　查询王敏在 2013 年 3 月 2 日使用过的处方最近一次被修改的时间。

```
SELECT SNAPSHOT BEGIN(TRANSACTION(P2))
FROM 处方 AS P1. P2
WHERE P1. 病人姓名＝‘王敏’AND P2.病人姓名＝’ 王敏’
        AND VALID(P1) OVERLAPS  DATE ‘2013—03—02’
        AND VALID(P2) OVERLAPS  DATE，2013—03—02’
        AND TRANSACTION(P1)   MEETS  TRANSATION(P2)
```

说明 P1、P2 是两个关联名，分别代表处方中的两个元组。它们的病人姓名都是 John. 有效期间都覆盖 2013－03－02，都是 John 在 2013 年 3 月 2 日使用过的处方。但是 P1 与 P2 的事务时间是不同的。P2 事务时间紧挨在 P1 之后，因此，P1 是未修改过的处方，P2 是修改后的处方，P1 的修改时间在 TRANSATION(P2)开始之前。如果该处方仅修改一次，则此时间即 BEGIN(TRANSACTION(P2))就是所要求的答案；如果该处方多次修改，则查询结果将是该处方历次被修改的时间。而最后一次的修改时间才所要求的答案。

4.时态关系更新

TSQL2 数据更新包括插入、删除和修改，SQL 中的相应更新语句都可推广到 TSQL2 中。

（1）数据插入　在插入时，如果在时态关系“处方”中，已有属性相同的元组，则新插入元组需要与此元组归并，即在该元组的时间标签中，插入新元组有效期间标签集合。只有在被插入时态关系中没有任何元组的属性值与插入的新元组完全相同时，新元组才能作为一个单独元组插入到关系“处方”中。

例 10 插入一元组（处方），但其有效时间终点待定。

```
INSERT INTO 处方
VALUES(‘John’，‘Black’，‘Vitamin E’，’ 100mg’
```

INTERVAL ‘8:00’ MINUTE);

说明 服药间隔的数据类型为 INTERVAL,值为 8，时间元为分。在本例中未指明时间标签，则在插入时取时间标签的缺省值，即

VALID PERIOD (CURRENT-TIMESTAMP,NOBIND(CURRENT-STAMP))

CURRENT-TIMESTAMP 指插入时的当前时间，其粒度由系统决定。括号中的第一项指有效时间的开始时间，第二项指有效时间的终止时间。如果插入时终止时间明确那就直接填入即可，但有时终止时间在开处方时难以确定，要根据药效才能确定，TSQL2 用 NOBIND(CURRENT-STAMP)这样一个变量记录在数据库中，在查询时间，该变量就与查询的当前时间绑定。但这种形式变量的表示在 SQL 中是不允许的。

例 11 插入一处方，处方有效截止期已定。

INSERT INTO 处方

VALUES(‘John’，‘Black’，Vitamin E’，‘lOOmg’

INTERVAL‘8：00’MINUTE)

VALID PERIOD'[2013—02—0l，2013—08—31]’

说明：插入的这条处方的有效期间为 2013 年 2 月到 8 月。

（2）数据删除

例 12 删除 2013 年 10 月开给王敏的所有处方。

DELETE FROM 处方

WHERE 病人姓名=’ John’

VALID PERIOD‘[2013—10—0l，2013—10-31]’

说明：凡是在 2013 年 10 月份开给 John 的所有处方都将被删除。如果有些处方的有效期只有部分在 2013 年 10 月，则从有效期中删除 2013 年 10 月中的那部分，保留其余部分。

（3）数据修改

例 13　改变 John 在 2013 年 2 月至 8 月的 Vitamin E 的剂量为 50mg。

UPDATE 处方

SET 剂量 TO ‘50mg’

VALID　PERIOD　‘[2013-02-01. 2013-08-31]’

WHERE　病人姓名=’John’AND 药名= Vitamin E

例 14

1、创建表

```
CREATE TABLE NBCShows
(ShowName CHARACTER ( 30 ) NOT NULL,
InsertionLength INTERVAL SECOND,
Cost INTEGER)
AS VALID STATE YEAR ( 2 ) TO NBCSeason ;
```

说明：ShowName 是美国全国广播公司(NBC 电视台)一个项目的名称。InsertionLength 是一个商业的持续时间(称为一个插入)显示在项目期间，和成本的价格在美元广告。这个表述不同于 sql - 92 中 AS VALID 声明规则。这种构造识别 NBCShows 表作为一个有效时间状态表，记录现实的变化的信息。这些表包含时间戳的行，通过有效时间元素，集合的时间，自己锚定持续时间的时间。

这个声明中提到了三个粒度用来分划时间线。SECONG 和 YAER 是两个粒度可在 sql - 92 找到。NBCSeason 把每年分区为 3 个不同的季度，开始在不同的时间 years 不同。TSQL2 允许数据库管理员定义新的日历，提供一个或多个粒度。日历也可以提供 DBMS 供应商或第三方。

每个阶段(间隔，datetime)都有一个关联的范围内，最大时间可以表示，一个潜在的粒度。这里我们指定的粒度是 NBCSeason。我们指定一个范围的 100

年,通过语法”YEAR(2)”,这表明 100 年。

```
CREATE TABLE NBC_FB_Insertion
(GameName CHARACTER ( 30 ),
InsertionWindow INTERVAL FootballSegment,
InsertionLength INTERVAL SECOND ( 3, 0 ),
CommercialID CHARACTER ( 30 ) )
AS VALID EVENT YEAR ( 2 ) TO HOUR AND TRANSACTION ;
```

2、插入记录

```
INSERT INTO NBCShows

VALUES (’Roseanne’, INTERVAL ’30’ SECOND, 251000)

VALID TIMESTAMP ’Spring Season 1994’ ;
```

时态数据处理构件根据 ATSQL2 的语法引入了 Now、Beginning 和 Forever 三个变元。Beginning 和 Forever 分别表示时态数据处理构件所能表示的时间起点和终点。Now 表示当前时间。每次执行操作时必须使 Now 绑定到一个固定的值（操作执行的当前时间），这样后继操作才能正常进行。在下面的 ATSQL2 中表示的这几个单词是具有具体的含义的。

ATSQL2 的数据定义语句包括以下几项功能：创建表、创建视图、删除表、删除视图。

ATSQL2 语言和普通 SQL 语句一样，使用“create table”关键字创建数据库表，关键字“as validtime”用于指示系统创建具有有效时间支持的数据库表。

1. 创建员工表 Employee, 是在标准的创建表 SQL 结尾，加入 AS VALIDTIME 关键字即可。

```
CREATE TABLE Employee(
ID VARCHAR(30) NOT NULL,
NAME VARCHAR(50) NOT NULL)
AS VALIDTIME;
```

2. 插入一条记录的 ATSQL, 需要通过 VALIDTIME PERIOD 来指定一个时间范围的，其他部分和标准 SQL 一样

```
VALIDTIME PERIOD [1981-1985) Insert into Employee values ('112', 'Jack');
```

3. 记录查询: 查询所有记录，在标准 SQL 前加 VALIDTIME 关键字即可, 查询指定时间段的记录，可以通过 PERIOD 关键字指定时间区间：

 VALIDTIME SELECT * FROM Employee;

 VALIDTIME PERIOD [1983-1986) SELECT * FROM Employee;

 VALIDTIME PERIOD [1983-forever) SELECT * FROM Employee;

4. 向员工表里添加人员信息插入记录前后分别如表 3-1 和表 3-2 所示

 VALIDTIME period [date "2008-1-1" – now)

 Insert into Staff Values (132202, 'Zhang Jinning', '信息部');

5. 为 employee 表创建一个视图，该视图包含所有 1999-9-9 前离职的工人信息，其中 VALIDTIME(R)表示取元组 R 的有效时间区间：

```
Create view employee_v as
Validtime select * from employee
Where VALIDTIME(employee) before date "1999-9-9";
```

6. 删除员工表的 ATSQL 和一般的 SQL 语句一样，如下

```
DROP TABLE Employee;
```