A Language Coverage Test Suite for TSQL2

Giuseppe Di Mauro and Larry A. Huebel

January 26, 1998

TR-22

A TimeCenter Technical Report

Title	A Language Coverage Test Suite for TSQL2
Author(s)	Giuseppe Di Mauro and Larry A. Huebel
Publication History	February 1997. Initial version by Giuseppe Di Mauro. January 1998. A TIMECENTER Technical Report.

TIMECENTER Participants

Aalborg University, Denmark
Christian S. Jensen (codirector)
Michael H. Böhlen
Renato Busatto
Heidi Gregersen
Kristian Torp

University of Arizona, USA
Richard T. Snodgrass (codirector)
Anindya Datta
Sudha Ram

Individual participants
Curtis E. Dyreson, James Cook University, Australia
Kwang W. Nam, Chungbuk National University, Korea
Keun H. Ryu, Chungbuk National University, Korea
Michael D. Soo, University of South Florida, USA
Andreas Steiner, ETH Zurich, Switzerland
Vassilis Tsotras, University of California, Riverside, USA
Jef Wijsen, Vrije Universiteit Brussel, Belgium

For additional information, see The TIMECENTER Homepage:
URL: `<http://www.cs.auc.dk/general/DBS/tdb/TimeCenter/>`

The TIMECENTER icon on the cover combines two "arrows." These "arrows" are letters in the so-called *Rune* alphabet used one millennium ago by the Vikings, as well as by their precedessors and successors. The Rune alphabet (second phase) has 16 letters, all of which have angular shapes and lack horizontal lines because the primary storage medium was wood. Runes may also be found on jewelry, tools, and weapons and were perceived by many as having magic, hidden powers.

The two Rune arrows in the icon denote "T" and "C," respectively.

Abstract

TSQL2 is a temporal query language, designed to query and manipulate time-varying data stored in a relational database. TSQL2 is an upward-compatible extension of the international standard relational query language SQL-92. This document presents a TSQL2 test suite that utilizes many of the features of the TSQL2 temporal query language. The test suite consists of a set of table creation statements, a set of data manipulation statements, and a set of queries. This document includes the results of running the TSQL2 test suite on the 3.01 TSQL2 prototype DBMS from the University of Arizona.

1 Introduction

The SQL standard is a long and involved document; therefore, the temporal extension to SQL is necessarily a complex language. The TSQL2 temporal query language specification is 82 pages. As a result, determining whether a tool, such as a DBMS, conforms to the TSQL2 language specification is quite difficult. This test suite serves as a way to test such tools for complete support of the TSQL2 language.

This suite was designed to contain a comprehensive set of integrity constraints, queries, modification statements, and data definition statements that completely covers the TSQL2 language. In order for this TSQL2 test suite to be exhaustive, it was created using all aspects of the TSQL2 language. However, due to the complexity of the TSQL2 language specification, the test suite undoubtedly does not cover every facet of the language. For verification, the suite also contains the expected result of each statement when executed on known test data.

2 Structure

2.1 A Syntactic Classification

To better classify queries, the TSQL2 syntax has been divided into syntactical classes. These classes provide a framework for developing a comprehensive test suite.

The best way to view this syntax classification is as a tree. Figure 1 contains the TSQL2 syntax classification tree. Many of the queries of the test suite touch on more than one leaf of the tree.

The syntax tree is rooted in TSQL2, and this is the maximum abstraction. The leaves of the tree represent the most concrete view of a syntactical construct. Thus, walking the tree from the root to a leaf, different abstraction layers are reached, going from abstract to concrete.

2.2 Language Coverage Analysis

In this section, the queries of the TSQL2 test suite are presented as leaves of the syntax tree. Because of the great number of queries, the queries are not represented in figure 1.

Data Definition

Table Definition

d1, d2, d3, d4, d5, d6

Table Alteration

Data Manipulation

Data Types

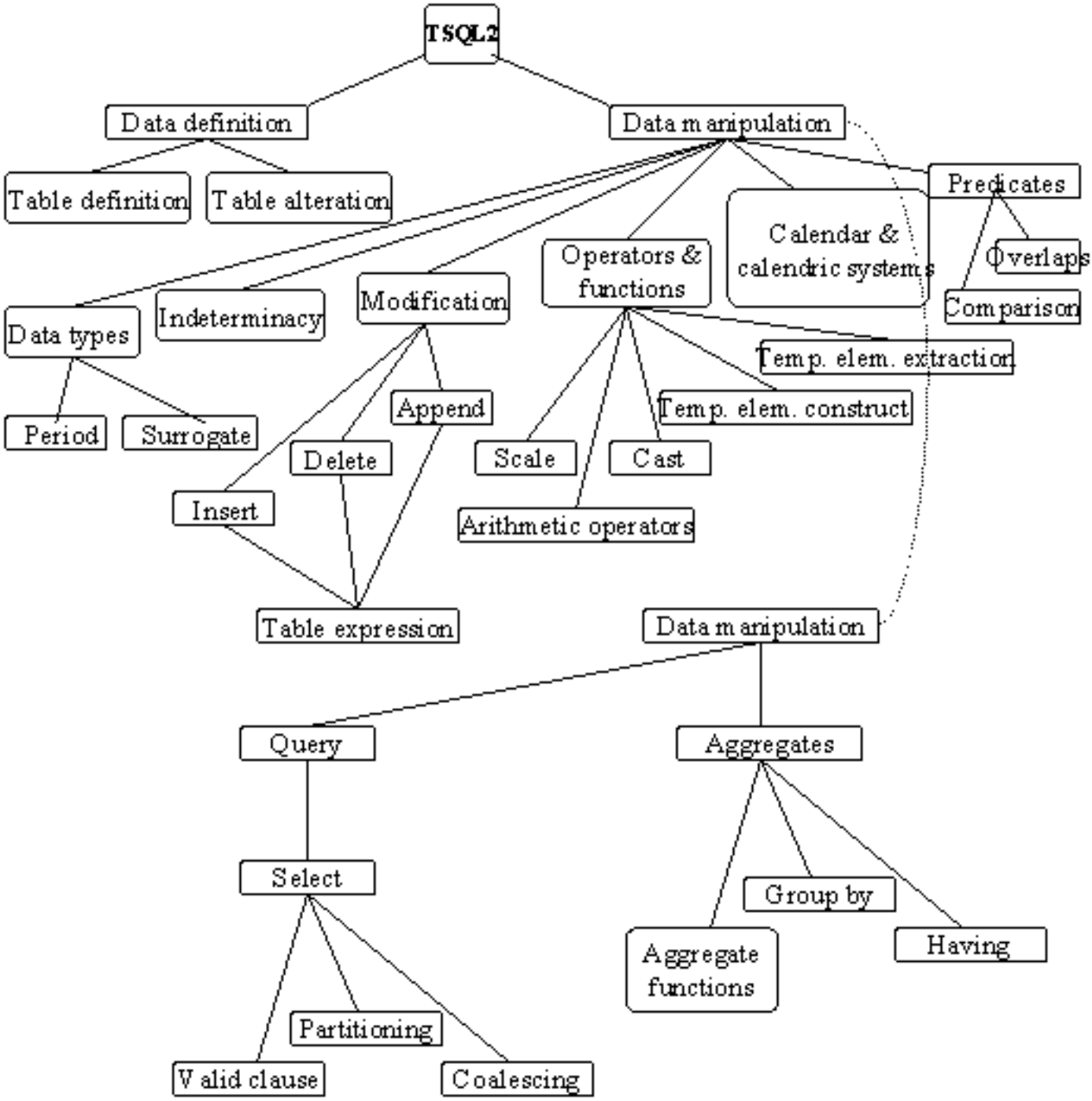

Figure 1: TSQL2 Syntax Classification

- Period
- Surrogate
  - **d2, d3, d4, d5, d6, m1, m10, m14, q6, q9, q14, q15, q23, q24, q31**
- Indeterminacy
  - **q45, q46, q47**
- Modification
  - Insert
    - **m1, m5, m6, m10, m14, q22**
  - Delete
    - **m17, m18, m19**
  - Append
- Query
  - Select
    - Valid Clause
      - **q3, q4, q23, q45**
    - Partitioning
      - **q1, q2, q8, q9, q10, q11, q14, q16, q17, q20, q25, q26**
    - Coalescing
      - **q1, q8, q9, q10, q17, q20, q23, q25, q26**
- Operators and Functions
  - Arithmetic Operators
    - **q12, q34, q35, q36, q37, q38, q39, q40, q41, q42, q43, q44**
  - Temporal Element Construction
    - **q3, q4, q6, q8, q10, q12, q16, q43, q44, m17**
  - Temporal Element Extraction
    - **q5, q6, q7, q8, q9, q10, q11, q12, q13, q14, q15, q16, q17, q18, q19, q21, q22, q25, q26, q28, q29, q31, q46, q47**
  - Cast
    - **q21, q23, q26, q33**
  - Scale
    - **q27, q28, q29**
- Predicates
  - Comparison
    - **q5, q11, q21, q25, q26, q30, q31, q33**
  - Overlaps
    - **q2, q3, q6, q7, q8, q9, q12, q13, q16, q17, q46, q47**
- Aggregates

Aggregate Functions

q20, q21, q22, q23, q31

Group By

q18, q19, q20, q21, q22, q23, q24

Having

q21

Calendar and Calendric Systems

q48

2.2.1 Uncovered Aspects

There are still some aspects uncovered in this test suite:

- *Bitemporal Tables*: This release does not cover bitemporal tables; transaction tables are not tested in this release.
- *Table Alteration*: Temporal Element Construction and Temporal Element Extraction.
- *Append*

3 Test Suite Statements

3.1 The Schema

The database schema for the TSQL2 test suite is basically the one shown by the entity-relationship graph in figure 2. Because of the non homogeneity of the queries and the need to cover as many statements as possible, sometimes the schema may change to test a particular aspect of the TSQL2 temporal query language.

3.1.1 Aspects of the Schema

The schema represented in figure 2 is comprised of six tables, named with letters from the English alphabet. Any table in the figure may be joined to a particular table in the temporal environment, yielding the following tables: *snapshot*, *state*, *transaction*, *event*, *bitemporal state and transaction* and *bitemporal event and transaction*. The last two tables are mirrored from equivalent non-temporal state and event tables, and they share data with the non-temporal state and event tables.

3.2 Create Table Statements

d1

```
CREATE TABLE a (
     g    CHARACTER (10) NOT NULL,
          PRIMARY KEY (g),
     h    DATE,
     i    CHARACTER,
          FOREIGN KEY(i) REFERENCES b (i)
);
```

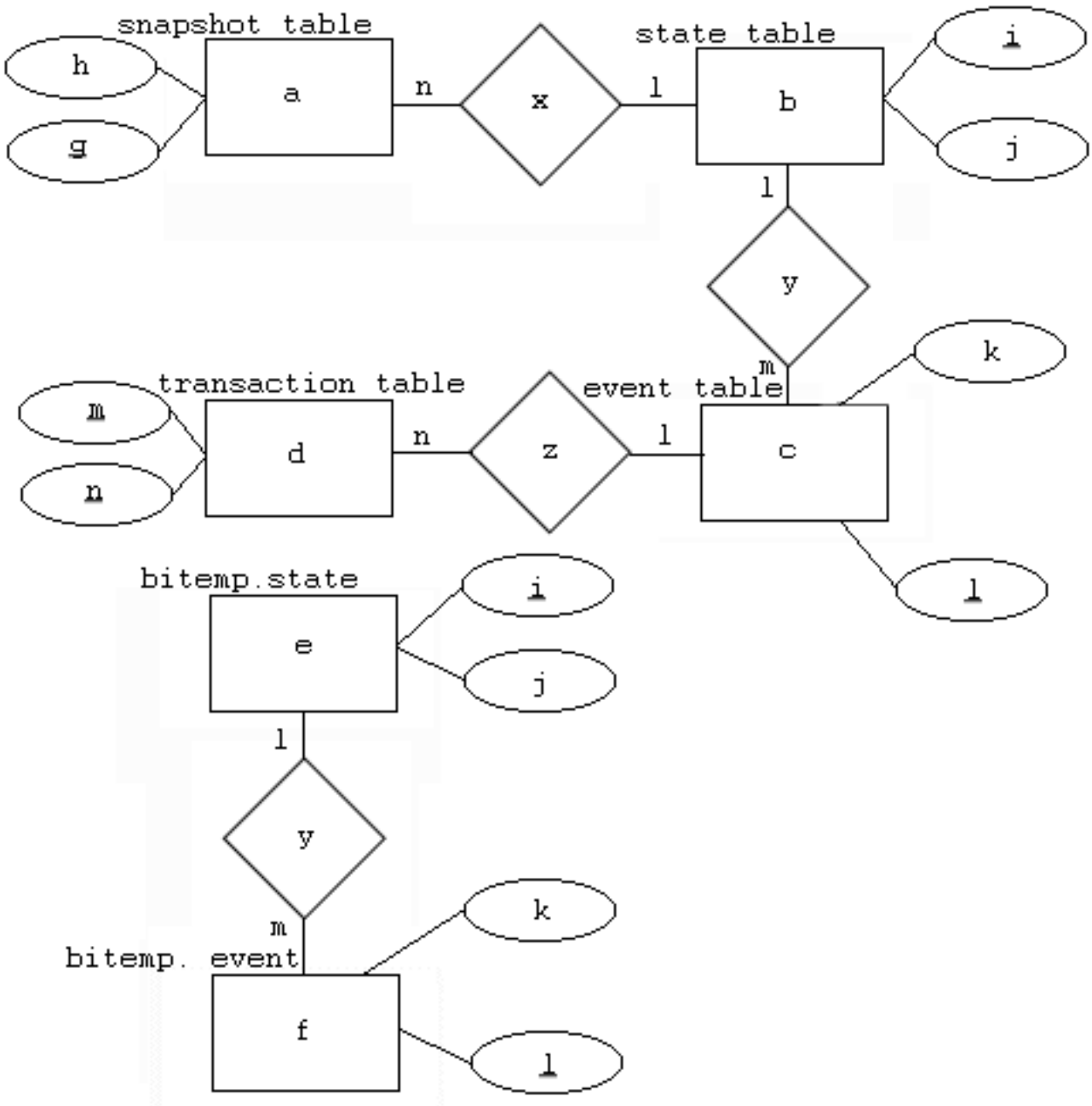

Figure 2: Entity-Relationship Graph for this Database Schema

d2

```
CREATE TABLE b (
     i    CHARACTER NOT NULL,
          PRIMARY KEY (i),
     j    TIMESTAMP,
     k    SURROGATE NOT NULL,
          FOREIGN KEY(k) REFERENCES c (k)
)
AS VALID STATE;
```

d3

```
CREATE TABLE c (
     k    SURROGATE NOT NULL,
     l    DECIMAL (8,0) NOT NULL,
          PRIMARY KEY (l)
)
AS VALID EVENT;
```

d4

```
CREATE TABLE d (
     m    CHARACTER (5) NOT NULL,
          PRIMARY KEY (m),
     n    TIME,
     k    SURROGATE NOT NULL,
          FOREIGN KEY(k) REFERENCES c (k)
)
AS TRANSACTION;
```

d5

```
CREATE TABLE e (
     i    CHARACTER NOT NULL,
          PRIMARY KEY(i),
     j    TIMESTAMP,
     k    SURROGATE NOT NULL,
          FOREIGN KEY(k) REFERENCES c(k)
)
AS VALID STATE AND TRANSACTION;
```

d6

```
CREATE TABLE f (
     k    SURROGATE NOT NULL,
     l    DECIMAL(8,0) NOT NULL,
          PRIMARY KEY(l)
)
AS VALID EVENT AND TRANSACTION;
```

3.3 Data Manipulation

This section shows queries which populate the schema with data. For almost all queries, the SQL92 calendric system is used.

m1

```
INSERT INTO b
     SELECT 't', TIMESTAMP '1994-11-04 10:00', k
          VALID PERIOD ( DATE '1995-01-01', TIMESTAMP 'now')
          FROM c
          WHERE l='10';
```

---

m2

```
INSERT INTO b
     SELECT 'f', TIMESTAMP '1994-11-03 10:20', k
          VALID PERIOD '[1993-01-01 - 1993-01-01]'
          FROM c
          WHERE l='20';
```

---

m3

```
INSERT INTO b
     SELECT 'i', TIMESTAMP '1994-12-03 13:00', k
          VALID PERIOD '[1993-01-01 - 1994-01-01]'
          FROM c
          WHERE l='10';
```

---

m4

```
INSERT INTO b
     SELECT 'h', TIMESTAMP '1990-08-07 11:00', k
          VALID PERIOD '[1996-01-07 - 1996-03-08]'
          FROM c
          WHERE l='30';
```

---

m5

```
UPDATE b
     SET j = TIMESTAMP '1994-12-12 19:00'
     VALID PERIOD '[1994-01-02 - 1994-08-10]'
     WHERE b.j=TIMPESTAMP '1994-12-03' AND b.i='t';
```

Resulting **b** table from statements **m1** - **m5**:

i	j	k	Valid Time
't'	'1994-11-04 10:00'	X	[1995-01-01 - now]
'f'	'1994-11-03 10:20'	Y	[1995-01-02 - now]
't'	'1994-12-03 13:00'	X	[1993-01-01 - 1994-01-01]
'h'	'1994-08-07 13:00'	Z	[1996-01-07 - 1996-03-08]
't'	'1994-12-08'	X	[1994-01-02 - 1994-08-10]

---

m6

```
INSERT INTO a
VALUES ('a', DATE '1970-11-21', 't');
```

---

m7

```
INSERT INTO a
VALUES ('b', DATE '1973-06-19', 't');
```

---

m8

```
INSERT INTO a
VALUES ('c', DATE '1973-04-14', 't');
```

---

m9

```
INSERT INTO a
VALUES ('h', DATE '1974-08-28', 'h');
```

Resulting a table from statements **m6 - m9**:

g	h	i
'a'	'1970-11-21'	't'
'b'	'1973-06-19'	't'
'c'	'1973-04-14'	't'
'h'	'1974-08-28'	'h'
'a'	'1972-05-10'	'h'

---

m10

```
INSERT INTO c
     VALUES (NEW, 10)
VALID TIMESTAMP '1996-01-01 10:00';
```

---

m11

```
INSERT INTO c
     VALUES (NEW, 20)
VALID TIMESTAMP '1996-01-02 11:00';
```

---

m12

```
INSERT INTO c
     VALUES (NEW, 30)
VALID TIMESTAMP '1996-01-03 12:00';
```

---

m13

```
INSERT INTO c
     VALUES (NEW, 40)
VALID TIMESTAMP '1996-01-04 13:00';
```

Resulting **c** table from statements **m10 - m13**:

k	l	Valid Timestamp
X	'10'	'1996-01-01 10:00'
Y	'20'	'1996-02-01 11:00'
Z	'30'	'1996-03-01 12:00'
T	'40'	'1996-04-01 13:00'

---

m14

transaction time: January 3, 1996

```
INSERT INTO d
SELECT SNAPSHOT 'a', TIME '10:20', x.k
     FROM b(i,k) AS x
     WHERE x.i='t';
```

---

m15

transaction time: April 5, 1996

```
INSERT INTO d
SELECT SNAPSHOT 'b', TIME '1:40 PM',x.k
     FROM b(i,k) AS x
     WHERE x.i='f';
```

m16

Transaction time: August 1, 1996

```
INSERT INTO d
SELECT SNAPSHOT 'c', TIME '12:20',x.k
     FROM b(i,k) AS x
     WHERE x.i='f';
```

Resulting **d** table from statements **m14 - m16**:

i	j	k	Transaction Time
'a'	'10:20'	X	'1996-01-03'
'b'	'1:40'	Y	'1996-04-05'
'c'	'12:20'	Y	'1996-08-01'

3.4 Other Data Manipulation Statements

m17

```
DELETE FROM b
WHERE i='t'
VALID PERIOD (DATE '1996-09-10', TIMESTAMP 'forever');
```

Resulting **b** table:

i	j	k	Valid Time
't'	'1994-11-04 10:00'	X	[1995-01-01 - 1996-09-10]
'f'	'1994-11-03 10:20'	Y	[1995-01-02 - now]
't'	'1994-12-03 13:00'	X	[1993-01-01 - 1994-01-01]
'h'	'1994-08-07 13:00'	Z	[1996-01-07 - 1996-03-08]
't'	'1994-12-08'	X	[1994-01-02 - 1994-08-10]

m18

```
DELETE FROM b
WHERE i='t'
VALID PERIOD '[1993-12-25 - 1994-04-14]';
```

Same result as the previous table.

---

m19

```
DELETE FROM b
WHERE i='t';
```

Transaction time is September 26, 1996.

Resulting **b** table:

i	j	k	Valid Time
't'	'1994-11-04 10:00'	X	[1995-01-01 - 1996-10-26]
'f'	'1994-11-03 10:20'	Y	[1995-01-02 - now]
't'	'1994-12-03 13:00'	X	[1993-01-01 - 1994-01-01]
'h'	'1994-08-07 13:00'	Z	[1996-01-07 - 1996-03-08]
't'	'1994-12-08'	X	[1994-02-01 - 1994-08-10]

3.5 Query Statements

q1

```
SELECT SNAPSHOT a.h, x.i
FROM b(i)(PERIOD) AS x, a
WHERE x.i='t' AND x.i=a.i;
```

Resulting table:

h	i
't'	'1970-11-21'
't'	'1973-06-19'
't'	'1973-04-14'
't'	'1970-11-21'
't'	'1973-06-19'
't'	'1973-04-14'
'h'	'1974-08-28'
'h'	'1972-05-10'

---

q2

```
SELECT SNAPSHOT y.g
FROM a(i) AS x, x(g,h) AS y
WHERE y.h PRECEDES DATE '1973-08-01';
```

Resulting table:

j
'a'
'b'
'c'

---

q3

```
SELECT l VALID PERIOD (VALID(c), DATE '1996-01-31')
FROM c
WHERE VALID (c) PRECEDES TIMESTAMP '1996-02-28 12:00';
```

Resulting table:

l	Valid Time
'10'	[1996-01-01 10:00 - 1996-02-28 12:00]
'20'	[1996-02-02 11:00 - 1996-02-28 12:00]

---

q4

```
SELECT *
VALID INTERSECT PERIOD (DATE '1993-04-01',DATE '1995-04-01') FROM b;
```

Resulting table:

i	j	k	Valid Timestamp
't'	'1994-11-04 10:00'	X	[1993-04-01 - 1995-04-01]
'f'	'1994-11-03 10:20'	Y	[1995-02-01 - 1995-05-01]
't'	'1994-12-03 13:00'	X	[1995-05-01 - 1994-01-01]
't'	'1994-12-08 19:00'	X	[1994-02-02 - 1994-10-08]

---

q5

```
SELECT SNAPSHOT VALID(x) AS v
FROM c AS x
WHERE x.l < ALL(SELECT AVG(l) FROM c);
```

Resulting table:

v
'1996-01-01 10:00'
'1996-02-01 11:00'

---

q6

```
SELECT SNAPSHOT BEGIN(VALID(x)) AS beg
FROM b AS x, c AS y
WHERE x.k = y.k AND PERIOD(VALID(c), TIMESTAMP 'now') CONTAINS VALID(x);
```

Resulting table:

beg
'1994-08-07 13:00'

---

q7

```
SELECT SNAPSHOT m
FROM d
WHERE VALID(d) PRECEDES DATE '1996-05-05';
```

Resulting table:

m
'a'
'b'

---

q8

```
SELECT x.i, VALID (x) AS v
FROM b(i)(PERIOD) AS x
WHERE
     PERIOD (DATE '1995-01-01',DATE '1996-01-01')
     CONTAINS BEGIN(VALID(x));
```

Resulting table:

i	v
't'	[1995-01-01 - now]
'f'	[1995-02-01 - now]

---

q9

```
SELECT SNAPSHOT y.i, z.l
FROM b(k) AS x, x(i)(PERIOD) AS y,
     c(k) AS z, z(l)(PERIOD) AS t
WHERE FIRST(BEGIN((VALID(y))),VALID(z))
     PRECEDES DATE '1996-02-02' AND x.k = z.k;
```

Resulting table:

i	l
't'	'10'
'f'	'20'
't'	'10'

---

q10

```
SELECT SNAPSHOT x.i
FROM b(i)(PERIOD) AS x
WHERE INTERSECT(VALID(x),PERIOD(DATE '1994-05-01',
     DATE '1995-05-01')) IS NOT NULL;
```

Resulting table:

i
't'
'f'

---

q11

```
SELECT SNAPSHOT VALID(x) AS v
FROM c(l) AS x
WHERE VALID(x)<=(SELECT FIRST(VALID(c)) FROM c);
```

Resulting table:

v
'1996-01-01 10:00'

---

q12

```
SELECT SNAPSHOT i
FROM b
WHERE VALID(b) CONTAINS PERIOD '[1994-01-01 - 1995-01-01]' +
     PERIOD(DATE '1996-01-01', TIMESTAMP 'now');
```

Resulting table:

i
Empty

---

q13

```
SELECT SNAPSHOT l
FROM c
WHERE PERIOD '[1995-01-01 - 1996-03-01]' CONTAINS VALID(c);
```

Resulting table:

l
'10'
'20'

---

q14

```
SELECT SNAPSHOT x.i
FROM b(i,k) AS x, d AS y
WHERE x.k=y.k
     AND BEGIN(TRANSACTION(y)) = DATE '1996-04-04';
```

Resulting table:

i
'f'

---

q15

```
SELECT SNAPSHOT x.i
FROM b(i) AS x, x(period) as y
WHERE VALID(y)='[1996-01-07 - 1996-03-08]'
```

Resulting table:

i
'h'

---

q16

```
SELECT x.l
FROM c(l) AS x
WHERE PERIOD '[1995-05-01 - 1996-28-02]'
     OVERLAPS PERIOD (VALID(x), DATE 'now');
```

Resulting table:

l
'10'
'20'

---

q17

```
SELECT SNAPSHOT a.h
FROM b(i)(PERIOD) AS x, a
WHERE a.i=x.i AND PERIOD '[1973-04-14 - 1996-07-01]' MEETS VALID(x);
```

Resulting table:

h
'1974-08-28'
'1972-05-10'

---

q18

```
SELECT SNAPSHOT AVG(x.l) AS a
FROM c AS x
GROUP BY VALID(x) USING 1 YEAR;
```

Resulting table:

a
'25'

---

q19

```
SELECT SNAPSHOT (x.l) AS a
FROM c AS x
GROUP BY VALID(x) USING PERIOD 'All of time' YEAR;
```

Resulting table:

a
'25'

---

q20

```
SELECT SNAPSHOT COUNT(DISTINCT x.i) AS c
FROM b(i)(PERIOD) AS x
GROUP BY VALID(x) USING 6 MONTH LEADING 6 MONTH TRAILING 6 MONTH;
```

Resulting table:

Count
'1'
'2'
'3'

---

q21

```
SELECT SNAPSHOT AVG(x.j) AS a, CAST(VALID(x) AS YEAR) AS y
FROM b AS x
GROUP BY VALID(x) USING 1 YEAR
HAVING CAST(VALID(x) AS INTERVAL DAY) > INTERVAL '180' DAY;
```

Resulting table:

a	y
'1994-12-03 13:00'	'1993'
'1994-12-08 19:00'	'1994'
'1994-12-03 10:10PM'	'1995'

---

q22

```
SELECT SNAPSHOT AVG(VALID(c)) AS avg
FROM c
GROUP BY VALID(c) USING PERIOD 'All of time' MINUTE;
```

Resulting table:

avg
'13.84'

---

q23

```
CREATE TABLE f (
     l    DECIMAL (8,0) NOT NULL,
     PRIMARY KEY(l)
);
INSERT INTO f
     SELECT x.l
     VALID y
     FROM b(PERIOD) AS y, c AS x
     WHERE y.k=x.k;

SELECT SNAPSHOT MAX(WEIGHTED f.l) AS max
FROM f
GROUP BY VALID(f) USING PERIOD 'All of time' MINUTE;
```

Resulting table:

max
'20'

---

q24

```
CREATE TABLE b2 (
     i    CHARACTER NOT NULL,
          PRIMARY KEY (i),
     j    TIMESTAMP
) AS VALID STATE;
```

Resulting table:

k	attr	Valid Time
'a'	'10'	[1994-01-03 - 1994-04-04]
'a'	'5'	[1994-04-05 - 1995-06-07]
'a'	'15'	[1995-06-08 - 1995-12-11]
'a'	'20'	[1995-12-12 - 1995-12-31]
'a'	'11'	[1996-01-01 - 1996-03-01]
'a'	'12'	[1996-03-02 - now]
'b'	'10'	[1994-01-03 - 1995-07-07]
'b'	'9'	[1995-07-07 - 1995-01-01]
'b'	'10'	[1996-01-02 - 1996-01-03]
'b'	'11'	[1996-01-04 - now]

```
SELECT SNAPSHOT RISING(attr) AS r, k
FROM b2
GROUP BY k;
```

Resulting table:

r	k
[1994-05-01 - 1995-11-12]	'a'
[1995-08-07 - now]	'b'

---

q25

```
SELECT SNAPSHOT x.i, VALID (x) AS v
FROM b(i)(PERIOD) AS x
WHERE CAST(VALID(x) AS INTERVAL DAY) > CAST(INTERVAL '6' MONTH AS INTERVAL DAY);
```

Resulting table:

i	v
't'	[1995-01-01 - now]
'f'	[1995-02-02 - now]
't'	[1993-01-01 - 1994-10-08]

---

q26

```
SELECT SNAPSHOT x.i, CAST(VALID(x) AS INTERVAL MONTH) AS v
FROM b(i)(PERIOD) AS x
WHERE CAST(VALID(x) AS INTERVAL DAY) > CAST(INTERVAL '6' MONTH AS INTERVAL DAY);
```

Resulting table:

i	v
't'	'23' MONTHS
'f'	'22' MONTHS
't'	'34' MONTHS

---

q27

```
SELECT SNAPSHOT m, SCALE (n AS DAY)
FROM d;
```

Resulting table:

m	n
'a'	'0 10:20'
'b'	'0 1:40'
'c'	'0 12:20'

---

q28

```
SELECT SNAPSHOT m, SCALE(TRANSACTION(d) AS HOUR) AS s
FROM d;
```

Resulting table:

m	s
'a'	'1996-01-01 0 1996-01-03 23'
'b'	'1996-04-04 0 1996-04-05 23'
'c'	'1996-08-01 1996-08-01 23'

---

q29

```
SELECT SNAPSHOT SCALE(VALID(b) AS INTERVAL HOUR) AS s
FROM b;
```

Resulting table:

s
'16656' HOURS
'16632' HOURS
'8760' HOURS
'1464' HOURS
'5304' HOURS

q30

```
SELECT SNAPSHOT b.i, b.j
FROM b AS x
WHERE EXTRACT(MONTH FROM x.j) = 'November' MONTH;
```

Resulting table:

i	j
't'	'1994-11-04 10:00'
'f'	'1994-11-03 10:20'

---

q31

```
SELECT SNAPSHOT PERIOD(b.j, VALID(c)) AS p
FROM b,c
WHERE b.k = c.k AND b.j < VALID(c);
```

Resulting table:

p
[1994-11-04 10:00 - 1996-01-01 10:00]
[1994-12-03 10:20 - 1996-01-01 10:00]
[1994-12-08 14:00 - 1996-01-01 10:00]
[1994-11-03 10:20 - 1996-02-01 11:00]
[1994-08-07 13:00 - 1996-03-01 12:00]

---

q32

```
CREATE TABLE a1 (
     g    CHARACTER (10) NOT NULL,
     h    DATE,
     i    CHARACTER,
     m    INTERVAL MONTH
);
```

Table contents are:

g	h	m	i
'a'	'1970-11-21'	'3' YEARS	't'
'b'	'1973-06-19'	'2' YEARS	't'
'c'	'1973-04-14'	'10' YEARS	'h'
'h'	'1974-04-28'	'9' YEARS	't'

```
SELECT SNAPSHOT a1.g, a1.m
FROM a1;
```

---

q33

```
SELECT SNAPSHOT a1.g
FROM a1
WHERE CAST(PERIOD(a1.h,DATE '1978-01-01') AS INTERVAL YEARS)
     < (SELECT AVG(a1.m) FROM a1);
```

Resulting table:

g
'b'
'c'
'h'

---

q34

```
SELECT SNAPSHOT a1.m-INTERVAL '6' YEARS AS dif
FROM a1;
```

Resulting table:

dif
'-1' YEARS
'-2' YEARS
'6' YEARS
'4' YEARS

---

q35

```
SELECT SNAPSHOT ABSOLUTE(a1.m-INTERVAL '6' MONTHS) AS abs
FROM a1;
```

Resulting table:

Interval
'1' YEARS
'2' YEARS
'6' YEARS
'4' YEARS

---

q36

```
SELECT SNAPSHOT -a1.m AS int
FROM a1;
```

Resulting table:

Interval
'-3' YEARS
'-2' YEARS
'-10' YEARS
'-9' YEARS

---

q37

```
SELECT SNAPSHOT a1.m - INTERVAL '1' MONTH AS dif
FROM a1;
```

Resulting table:

dif
'2' YEARS
'1' YEARS
'9' YEARS
'8' YEARS

---

q38

```
SELECT SNAPSHOT INTERVAL '3' MONTH + a1.m AS sum
FROM a1;
```

Resulting table:

sum
'4' YEARS
'3' YEARS
'11' YEARS
'9' YEARS

---

q39

```
SELECT SNAPSHOT a1.h, a1.m, a1.h+a1.m AS sum
FROM a1;
```

Resulting table:

h	m	sum
'1970-11-21'	'3' YEARS	'1973-11-21'
'1973-06-19'	'2' YEARS	'1975-06-19'
'1973-04-14'	'10' YEARS	'1983-04-14'
'1974-08-28'	'8' YEARS	'1982-08-28'

---

q40

```
SELECT SNAPSHOT a1.h, a1.m, a1.h-a1.m AS dif
FROM a1;
```

Resulting table:

h	m	dif
'1970-11-21'	'3' YEARS	'1967-11-21'
'1973-06-19'	'2' YEARS	'1971-06-19'
'1973-04-14'	'10' YEARS	'1963-04-14'
'1974-08-28'	'8' YEARS	'1966-08-28'

---

q41

```
SELECT SNAPSHOT a1.m, a1.m*2 AS mul
FROM a1;
```

Resulting table:

m	mul
'3' YEARS	'6' YEARS
'2' YEARS	'4' YEARS
'10' YEARS	'20' YEARS
'8' YEARS	'16' YEARS

---

q42

```
SELECT SNAPSHOT a1.m, a1.m/INTERVAL '2' YEAR AS div
FROM a1;
```

Resulting table:

m	div
'3' YEARS	'1.5'
'2' YEARS	'1'
'10' YEARS	'5'
'8' YEARS	'4'

---

q43

```
SELECT SNAPSHOT a1.m,
     a1.m+PERIOD(DATE '1995-01-01',DATE '1996-01-01') AS sum
FROM a1;
```

Resulting table:

m	sum
'3' YEARS	[1998-01-01 - 1999-01-01]
'2' YEARS	[1997-01-01 - 1998-01-01]
'10' YEARS	[2005-01-01 - 2006-01-01]
'8' YEARS	[2003-01-01 - 2004-01-01]

---

q44

```
SELECT SNAPSHOT a1.m,
     a1.m - PERIOD(DATE '1992-01-01',DATE '1997-01-02') AS dif
FROM a1;
```

Resulting table:

m	dif
'3' YEARS	[1989-01-01 - 1994-01-02]
'2' YEARS	[1990-01-01 - 1995-01-02]
'10' YEARS	[1982-01-01 - 1987-01-02]
'8' YEARS	[1984-01-01 - 1989-01-02]

---

q45

Due to the indetermination in this query, table **b** table is modified as follow:

i	j	k	Valid Time
't'	'1994-11-04 10:00'	X	[1994-12-25 1995-01-01 - now]
'f'	'1994-11-03 10:20'	Y	[1994-12-01 1995-01-02 - now]
't'	'1994-12-03 13:00'	X	[1992-12-21 1993-01-01 - 1994-12-01 1994-01-01]
'h'	'1994-08-07 13:00'	Z	[1996-01-04 1996-01-07 - 1996-03-08 1996-03-18]
't'	'1994-12-08'	X	[1994-01-30 1994-01-02 - 1994-08-10 1994-12-10]

```
SELECT *
VALID x
FROM b AS x WITH CREDIBILITY 50
WHERE i='t';
```

Resulting table:

i	j	k	Valid Time
't'	'1994-11-04 10:00'	X	[1994-12-28 1995-01-01 - now]
't'	'1994-12-03 13:00'	X	[1992-12-26 1993-01-01 - 1994-12-01 1994-06-01]
't'	'1994-12-08'	X	[1994-01-31 1994-01-02 - 1994-08-10 1994-10-10]

---

q46

```
SELECT SNAPSHOT VALID(x), x.i
FROM b AS x
WHERE INDETERMINATE DATE '1994-12-31 1995-02-01' OVERLAPS x WITH PLAUSIBILITY 25;
```

Resulting table:

i	j	k	Valid Time
't'	'1994-11-04 10:00'	X	[1994-12-25 1995-01-01 - now]

---

q47

```
SELECT SNAPSHOT VALID(x), x.i
FROM b AS x
WHERE INDETERMINATE DATE '12/31/1994 2/1/1995' OVERLAPS x WITH PLAUSIBILITY 75;
```

Resulting table:

i	j	k	Valid Time
't'	'1994-11-04 10:00'	X	[1994-12-25 1995-01-01 - now]
'f'	'1994-11-04 10:00'	X	[1995-10-01 1995-02-01 - now]

---

q48

```
DECLARE CALENDRIC SYSTEM AS russian;

SELECT SNAPSHOT i,j WITH CALENDRIC SQL92_calendric_system
FROM b;
WHERE BEGIN(VALID(b)) < TIMESTAMP '2 Jinvar 1996'
     AND CAST (VALID(b) AS INTERVAL DAY) > INTERVAL '800 days'
     WITH CALENDRIC SQL92_calendric_system;

DECLARE CALENDRIC SYSTEM AS SQL92_calendric_system;
```

Resulting table:

i	j
't'	'1994-12-03 13:00'
't'	'1994-12-08'

4 Applying the Test Suite

This section describes the results of running the TSQL2 test suite on the 3.01 TSQL2 prototype DBMS. The results are very interesting because they highlight unimplemented or incorrectly implemented portions of the prototype.

The tables below describe the results of running the TSQL2 test suite on the 3.01 TSQL2 temporal DBMS prototype. The tables display the results for three categories of TSQL2 statements: *data definition*, *data manipulation*, and *query* statements.

Table 1: Results of Test Suite on Prototype 3.01 - *(Data)*

Statement	Frontend	Semantic	Interpreter
d1	yes	yes	failure
d2	failure		
d3	failure		
d4	failure		
d5	failure		
d6	failure		

The data definition queries **d2** - **d6** fail frontend syntax analysis because *SURROGATE* is not implemented.

Similarly, the general queries **q3** - **q23**, **q25** - **q31**, and **q45** - **q48** fail frontend syntax analysis because *SURROGATE* is not implemented.

All other queries fail either *frontend*, *semantic*, or *interpreter* stages of the 3.01 TSQL2 prototype due to individual problems in the prototype implementation.

Table 2: Results of Test Suite on Prototype 3.01 - *(Manipulation)*

Statement	Frontend	Semantic	Interpreter
m1	yes	seg. fault	
m2	yes	seg. fault	
m3	yes	seg. fault	
m4	yes	seg. fault	
m5	failure		
m6	yes	Error 38	
m7	yes	Error 38	
m8	yes	Error 38	
m9	yes	Error 38	
m10	yes	Error 38	
m11	yes	Error 38	
m12	yes	Error 38	
m13	yes	Error 38	
m14	yes	seg. fault	
m15	yes	seg. fault	
m16	yes	seg. fault	
m17	yes	yes	seg. fault
m18	failure		
m19	yes	yes	seg. fault

5 Acknowledgements

Giuseppe Di Mauro
Being that this work is a result of research at the University of Arizona to write my bachelor thesis, many people have helped me in this. Particular greetings to Prof. Richard T. Snodgrass who permitted me to have this experience. Prof. Snodgrass also lead and helped in the research itself.

Larry A. Huebel
I would also like to thank Professor Snodgrass for giving me the opportunity to work with him. This project was quite challenging.

Table 3: Results of Test Suite on Prototype 3.01 - *(Queries)*

Statement	Frontend	Semantic	Interpreter
q1	yes	yes	lacks grouping
q2	yes	Error 38	
q3	failure		
q4	failure		
q5	failure		
q6	failure		
q7	failure		
q8	failure		
q9	failure		
q10	failure		
q11	failure		
q12	failure		
q13	failure		
q14	failure		
q15	failure		
q16	failure		
q17	failure		
q18	failure		
q19	failure		
q20	failure		
q21	failure		
q22	failure		
q23	failure		
q24	failure		
q25	failure		
q26	failure		
q27	failure		
q28	failure		
q29	failure		
q30	failure		
q31	failure		
q32	yes	yes	passes
q33	yes	Error 38	
q34	yes	yes	passes
q35	yes	yes	passes
q36	yes	yes	passes
q37	yes	yes	passes
q38	yes	yes	passes
q39	yes	yes	passes
q40	yes	yes	passes

Table 4: Results of Test Suite on Prototype 3.01 - *(Queries Continued)*

Statement	Frontend	Semantic	Interpreter
q41	yes	yes	passes
q42	yes	yes	passes
q43	yes	yes	passes
q44	yes	yes	failure
q45	failure		
q46	failure		
q47	failure		
q48	failure		

A Appendix : References to the TSQL2 Book

A.1 Syntactic Coverage

This appendix represents the relationship between the syntax specification in the book "The TSQL2 Temporal Query Language" and the numbered queries of the TSQL2 test suite.

The table below is divided by the pages of the book. For every page there may be many productions of the form p*x*-*y*, where *x* represents the particular point of the grammar in the order it appears in the section leaded on the page; the number *y* represents the right side of the point *x*.

Because of the complexity of the syntax, there are many points (or subpoints) that are blank. Blank points do not necessarily represent uncovered parts of the grammar which are absent from the test suite. In many cases, only one of many similar grammar constructs are tested. For example, only a few arithmetic operators are tested.

A.2 Coverage Analysis

Page 551

p2-1
p2-2

q6, q8, q12, q13, q48

p2-3

q45

p2-4
p2-5

d3, d6

p2-6
p2-7
p2-8

q46, q47

p2-9
p2-10

q17

p2-11

m10, m11, m12, m13

p2-12
p2-13
p2-14

q1, q3, q4, q6, q8, q9, q10, q12, q13, q16, q17, q19, q20, q22, q23, q25, q31, q33, q43, q44, m17, m18, m1, d8, m3, m4, m5

p2-15

q45, q46, q47

p2-16

q2, q3, q7, q9

p2-17
p2-18
p2-19

q24

p2-20

q27, q28, q29

p2-21

q30, q31, q32, q33, q34, q35, q36, q37, q38, q39, q40, q41, q42, q43, q44, q45, q46, q1, q2, q5, q6, q7, q9, q10, q11, q12, q13, q14, q15, q17, q18, q19, q20, q21, q22, q23, q24, q25, q26, q27, q28, q29, m14, m15, m16

p2-22

q24

p2-23

d2, d3, d4, d5, d6

p2-24
p2-25

q3, q4, q5, q6, q7, q8, q9, q10, q11, q12, q13, q16, q17, q18, q19, q20, q21, q22, q23, q25, q26, q29, q31, q45, q46, q47, m17, m18

p2-26

q23

Page 552–553

p2

d4, q7, q10, q16, q46, q47

p3
p4

q4, q6, q9, q10, q12, q14, q15, q23, q48, m17

p5

q34, q35, q37, q38, q42

p12

q12, q13, q16, q17, q19, q22, q23, m18

Page 561

q48

Page 563

p2-1
p5-1
p6-1

d2, d3, d4, d5, d6

p7-1

d1

p7-2

d4

p7-3

d2, d5

p10-1

Page 568

p1-1
p2-1

q45

p3-1
p3-2
p4-1
p5-1

q1, q2, q8, q9, q10, q11, q14, q15, q16, q17, q20, q25, q26

p6-1
p6-2
p7-1
p7-2

q1, q8, q9, q10, q17, q20, q23, q25, q26

Page 570

p1-1
p2-1

q24

Page 573

p1-1

q6, q8, q9, q14, q15

p1-2
p1-3

q9

p1-4
p1-5

q11

p1-6
p1-7

q3, q5, q6, q7, q8, q9, q10, q11, q12, q13, q16, q17, q21, q22, q25, q26, q29, q31, q46, q47

p1-8

q27, q28

p1-9
p1-10

Page 574

p1-1

q21, q25, q26, q33, q48

Page 579

p1-1
p1-2
p1-3

Page 580

p1-1

Page 582

p1-1

Page 583

p1-1
p1-2

q35

p1-3
p1-4
p1-5

Page 584

p1-1
p1-2
p1-3
p2-1
p2-2
p2-3
p2-4
p2-5
p2-6

Page 585

p1-1

q3, q5, q6, q7, q8, q9, q10, q11, q12, q13, q16, q17, q21, q22, q25, q26, q29, q31, q46, q47

p1-2

q14, q15, q28

p1-3

q8, q12, q16, q31, q33, q43, q44, m17

p1-4

q10

p1-5

q9, q11

p1-6
p1-7

q29

p1-8
p1-9

Page 587

p1-1
p1-2
p2-1
p3-1
p4-1
p5-1

Page 588

p1-1

q3, q5, q6, q7, q8, q9, q10, q11, q12, q13, q16, q17, q21, q22, q25, q26, q29, q31, q46, q47

p1-2

q10

p1-3

q27, q28, q29

Page 589

p1-1
p1-2
p2-1

Page 590

p1-1

q3, q5, q6, q7, q8, q9, q10, q11, q12, q13, q16, q17, q18, q19, q20, q21, q22, q23, q25, q26, q29, q31, q46, q47

p1-2

q10

Page 591

p1-1
p2-1

m1, d8, m3, m6, m7, m2, m9, m10, m11, m12, m13

Page 592

p1-1
p2-1

q3, q4, q5

Page 593

p1-1

q46, q47

Page 594

p1-1

q18, q19, q20, q21, q22, q23, q24

p2-1

q18, q19, q20, q22, q23

p2-2
p2-3

q18, q19, q20, q21, q22, q23

p2-4

q18

p2-5

q18

p3-1
p3-2
p3-3

q18, q20, q21

p3-4

q19, q22, q23

Page 597

q43

Page 598

p1-1
p2-1

Page 599

p1-1

q2, q3, q7, q9

p1-2

q17

p1-3

q16, q46, q47

p1-4

q6

Page 600

q20, q25, q26, q33

Page 601

q31, q25, q26, q5, q6, q11, q30, q31, q33

Page 602

p1-1

q16, q46, q47

p2-1

q2, q3, q4, q9

p3-1

q17

p4-1

q6, q8, q12, q13

Page 605

p1-1
p1-2
p2-1
p3-1
p4-1
p4-2
p5-1

Page 607

p1-1

d1, d2, d3, d4, d5, d6

Page 608

p1-1

d2, d3, d4, d5, d6

p1-2
p2-1

d2, d3, d5, d6

p2-2

d4

p3-1

Page 610

p1-1

Page 611

p1-1
p1-2

Page 613–614

empty

Page 616

empty

Page 617

empty

Page 618–619

empty

Page 621

empty

Page 623

empty

Page 624

p1-1

m17, m18

Page 625

p1-1

m1, d8, m3, m4, m10, m11, m12, m13

p2-1
p3-1
p3-2

m10, m11, m12, m13

Page 624

p1-1

m17, m18

Page 625

p1-1

m1, d8, m3 , m10, m11, m12, m13

p2-1
p3-1
p3-2

m10

Page 626

m5

Page 627

m5

Page 629–630

empty

B Appendix : MULTICAL 3.01 Prototype DBMS

B.1 MULTICAL Files

The `multical3.01/prototype` directory has the following directory structure:

```
total 208
drwxr-x--- 13 login timecenter 4096 Oct 26 13:16 .
drwxr-x--- 13 login timecenter 4096 Oct 6 16:35 ..
-rwxr-x--- 1 login timecenter 2023 Oct 26 12:40 Makefile
-rwxr-x--- 1 login timecenter 31831 Nov 9 17:19 Makefile.multical
drwxr-x--- 2 login timecenter 4096 Oct 6 16:34 bin
drwxr-x--- 3 login timecenter 12288 Jan 7 18:43 coverage
drwxr-x--- 2 login timecenter 4096 Oct 6 16:34 errors
drwxr-x--- 7 login timecenter 4096 Jan 7 18:29 frontend
drwxr-x--- 3 login timecenter 8192 Jan 7 18:43 initest
drwxr-x--- 4 login timecenter 4096 Jan 7 18:32 interpret
drwxr-x--- 2 login timecenter 4096 Oct 26 11:05 lextest
drwxr-x--- 2 login timecenter 4096 Jan 7 18:35 lib
drwxr-x--- 3 login timecenter 4096 Jan 7 18:35 oneprocess
drwxr-x--- 6 login timecenter 4096 Jan 7 18:32 semantic
drwxr-x--- 3 login timecenter 4096 Jan 7 18:27 specs
```

The `bin` directory contains the MULTICAL executables.

The `coverage` directory contains the TSQL2 test suite described in this document. The data definition statements, data manipulation statements, and query statements adhere to the naming described in Section 3.2 through Section 3.5. However, all statements end in the `.sql` suffix. For example, the first data definition statement is named `d1.sql`.

The `errors` directory contains diagnostic output.

The `frontend` directory contains the code for the frontend of the prototype DBMS. Some of the files in this directory are:

- `doparse.c` : IDL frontend rules.
- `frontend.idl` : contains IDL code concerning the frontend.
- `frontendInv.idl` : contains IDL invariant code.
- `lex.l` : `lex` lexical analyzer code.
- `parser.y` : `yacc` parser code.

The `initest` directory contains the regression test suite.

The `interpret` directory contains the code for the interpreter of the prototype DBMS. Some of the files in this directory are:

- `dointerpret.c` : IDL interpreter rules.
- `interpret.idl` : contains IDL code concerning the interpreter.
- `interpretInv.idl` : contains IDL invariant code.

The `lextest` directory contains some SQL queries that are used to test the lexical analyzer of the frontend.

The `lib` directory contains MULTICAL component executables that are used to create the final MULTICAL executable.

The `oneprocess` directory contains code from the frontend, semantic, and interpreter directories. `Oneprocess` refers to the fact that 3 disjoint MULTICAL components - frontend analysis, semantic analysis, and interpretation - are combined to create one executable. Thus, SQL statements can be run through oneprocess to test all components of MULTICAL. Diagnostic messages from MULTICAL indicate which component failed if queries do not pass.

The `semantic` directory contains code for the semantic analyzer of the prototype DBMS. Some of the files in this directory are:

- `doanalysis.c` : IDL semantic rules.
- `semantic.idl` : contains IDL code concerning the analyzer.
- `semanticInv.idl` : contains IDL invariant code.

The `specs` directory contains IDL code for parsing and semantic analysis.

B.2 Building MULTICAL

Before building the MULTICAL 3.01 prototype DBMS, edit the `Makefile.include` file found in the `multical3.01` directory. Verify that the information found in `Makefile.include` is correct for your system. The file contains many variables which point to UNIX system tools. Make certain the variables are set correctly.

Lastly, verify that the `TOPDIR` variable points to the `multical3.01` directory. For example, if you installed multical3.01 in your home directory, then `TOPDIR` should be set as:

```
TOPDIR=/home/yourhome/multical3.01
```

where `yourhome` is your home directory.

Once the `Makefile.include` file is configured properly, change directories to the `multical3.01/prototype` directory. At the UNIX command prompt enter:

```
make full_clean
```

This make command will remove any MULTICAL 3.01 executable files that will be built during a full install.

Next enter:

```
make install
```

This make command installs all the MULTICAL 3.01 binaries needed.

B.3 Running the Test Suite on MULTICAL

Change directories to the `multical3.01/prototype` directory. At the UNIX command prompt enter:

```
make coveragetest
```

This make command will run the entire TSQL2 test suite on the MULTICAL 3.01 prototype DBMS. It will also report any unexpected behavior which differs from the output listed in Table 1, Table 2, Table 3, and Table 4 of Section 4.