10

A Second Example

Richard T. Snodgrass, Christian S. Jensen, Curtis E. Dyreson, Wolfgang Käfer, Nick Kline, and John F. Roddick

1 Introduction

This chapter expresses the schema and some of the queries of the extensive consensus test suite for temporal relational query languages [1]. The test suite is not related to performance issues, but has a *semantic* focus and is intended to be an aid in evaluating the user-friendliness of temporal query languages. An attempt is made to express the schema and queries in TSQL2 in a convenient and natural fashion, as an informal evaluation of that language's user-friendliness.

The test suite consists of a database schema, an instance for the schema, and a set of queries on this database. The queries are classified according to a taxonomy.

Within the test suite, no existing temporal data models are used or mentioned. The database schema of the test suite is described using the ER model. With the exception of attributes illustrating user-defined time, the underlying temporal aspects are implicit in the description of the database schema. The contents of the tables are described in natural language. The actual queries are also given only in natural language.

The test suite was designed to provide a "dense" coverage of a restricted range of queries rather than a "sparse" coverage of wide range of queries. A number of restrictions were imposed on which types of queries are intended to be included in the current test suite. These restrictions are listed in the test suite [1].

It should be emphasized that the test suite is not intended to constitute a metric for query language completeness, and hence this evaluation is not a substitute for a rigorous *theoretical* study of the expressive powers of TSQL2. Such a study is still needed. It must be emphasized that using the test suite as an advanced, quantitative scoring system for comparing languages makes little sense. Thus, one language is not necessarily superior to another just because one is capable of expressing more

queries than the other.

In the next section we show how the Entity-Relationship schema of the test suite can be mapped into TSQL2 relational schemas. We provide the TSQL2 `CREATE TABLE` statements that define this schema. Section 3 lists the modification statements necessary to populate these tables. The body of the chapter is Section 4, in which a selected few of the test suite queries are given, both in English and in TSQL2. A total of 43 queries are provided, or some 30% of those given in the full evaluation commentary [2].

2 A Database Schema for the Test Suite

2.1 Overview of the Schema for the Test Suite Database

The database schema is defined by the ER schema in Figure 1, containing three entity sets, namely `Emp`, `Skill`, and `Dept`, describing employees, skills, and departments, respectively. The attributes of the entity sets, and their interrelationships, are described next.

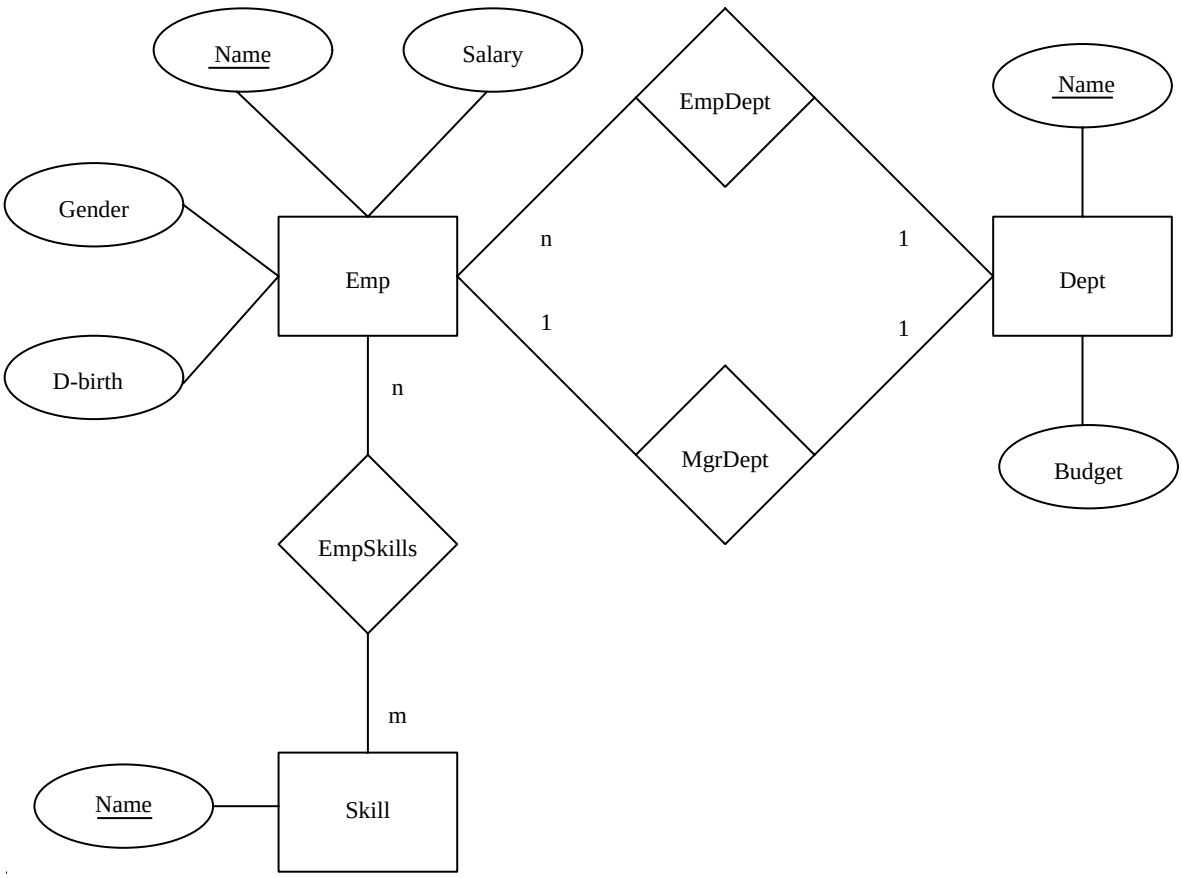

Figure 1: ER diagram of database design

Entities in entity set `Emp` are represented by the attributes `Name` and `Salary`

which record the names and salaries of employees. In addition, attributes `Gender` and `D-birth` indicate the gender and date of birth of employees. While the name and salary of an employee vary over time, both the gender and the date of birth are assumed to be time-invariant. The `Gender` attribute of `Emp` is one of `'F'` (female) and `'M'` (male).

Skills, in entity set `Skill`, are represented by a single attribute, `Name`, which records the names of individual skills. The name of a skill is time-invariant. Entity sets `Skill` and `Emp` are connected via an *n*–*m* relationship set, `EmpSkills`. The skills of an employee may vary over time. For example, employees are considered to have the skill "driving" only during those period(s) when they hold valid licenses.

The entity set `Dept` represents departments and is described by the attributes `Name` and `Budget` which record the names and budgets of departments, respectively. While the budget of a department varies over time, the name is assumed to be time-invariant. Employees are associated with departments by means of two relationship sets. First `EmpDept` records which employees work in which departments. This is a time-varying *n*–1 relationship set. Second `MgrDept`, also time-varying, is a 1–1 relationship set associating those employees that are managers with the departments they manage.

The entity sets obey the following *snapshot* functional dependencies:

```
For Emp:
    Name → Salary
    Name → Gender
    Name → D-birth
For Dept:
    Name → Budget
```

Note that `Name` is the primary key of `Emp` (it is the only candidate key). For `Skill`, `Name` is the only attribute and is thus the key. For `Dept`, `Name` is the primary key.

It is emphasized that the notion of key does not capture correspondence between attribute values and the real-world objects they represent. As one consequence, it is possible in this ER schema, e.g., for an employee to change `Name` attribute value over time.

2.2 The Schema in TSQL2

We now map this schema to a TSQL2 schema.

```
SET SCALE AS INTERVAL '1' DAY

CREATE TABLE Emp
    (ID SURROGATE NOT NULL,
    Name CHARACTER ( 30 ) NOT NULL,
        PRIMARY KEY (Name),
```

```
    Salary DECIMAL ( 8,2 ) ,
    Gender CHARACTER ( 1 ) ,
    D-birth DATE ,
    DeptName CHARACTER ( 30 )
        FOREIGN KEY (DeptName) REFERENCES Dept (Name))
AS VALID STATE ;

CREATE TABLE Skill
    (EmpID SURROGATE NOT NULL,
        FOREIGN KEY (EmpID) REFERENCES Emp (ID),
    Name CHARACTER ( 30 ) NOT NULL)
AS VALID STATE ;

CREATE TABLE Dept
    (Name CHARACTER ( 30 ) NOT NULL,
        PRIMARY KEY (Name),
    Budget DECIMAL (9,0 ),
    MgrID SURROGATE NOT NULL,
        FOREIGN KEY (MgrID) REFERENCES Emp (ID))
AS VALID STATE ;
```

The scale is set to one day. All timestamps are determinate.

Table `Emp` models the entity set `Emp` and the relationship set `EmpDept`. The attribute `ID` identifies employees; since the `Name` is not time-invariant, we used a `SURROGATE` attribute to uniquely identify the employee. Table `Skill` models the relationship set `EmpSkills` and the entity set `Skill`. Values of attribute `EmpID` in `Skill`, as well as `MgrID` in `Dept`, identify employees. Finally, table `Dept` models the relationship set `MgrDept` and the entity set `Dept`.

In this design, `Name` is the snapshot primary key of `Emp` (it is the only candidate key). For `Skill`, there is no non-trivial key. For `Dept`, each of `Name` and `MgrID` are snapshot (as well as time-invariant) candidate keys; `Name` is selected as the primary key.

Each of the table schemas are in temporal BCNF [3].

3 The Test Suite Data

There are two employees, identified by *ED* and *DI* in the following.

ED worked in the Toy department from 2/1/82 to 1/31/87, and in the Book department from 4/1/87 to the present. From 4/1/87 to the present, he managed the Book department. The budget of that department has been $50K since *ED* became its manager. His name was Ed from 2/1/82 to 12/31/87, and Edward from 1/1/88 to the present. His salary was $20K from 2/1/82 to 5/31/82, then $30K from 6/1/82 to 1/31/85, then $40K from 2/1/85 to 1/31/87 and 4/1/87 to the present. *ED* is male

and was born on 7/1/55. Several skills are recorded for *ED*. He has been qualified for typing since 4/1/82 and qualified for filing since 1/1/85. He was qualified for driving from 1/1/82 to 5/1/82 and from 6/1/84 to 5/31/88.

This information can be recorded in the database via the following modification operations.

```
INSERT INTO Emp
VALUES (NEW, 'Ed', 20000, 'M', DATE '7/1/1955', 'Toy')
VALID PERIOD '[2/1/1982 - 1/31/1987]' ;

INSERT INTO Emp
    SELECT (ID, 'Ed', 40000, 'M', DATE '7/1/1955',
            'Book')
        VALID PERIOD '[4/1/1987 - now]'
        FROM Emp
        WHERE Name = 'Ed' ;

INSERT INTO Dept
    SELECT ('Book', 50000, Emp.ID)
        VALID PERIOD '[4/1/1987 - now]'
        FROM Emp
        WHERE Emp.Name = 'Ed' ;

UPDATE Emp
    SET Name TO 'Edward'
        VALID PERIOD '[1/1/1988 - now]'
    WHERE ID = (SELECT DISTINCT ID
                FROM Emp WHERE Name = 'Ed') ;

UPDATE Emp
    SET SALARY TO 30000
        VALID PERIOD '[6/1/1982 - 1/31/1985]'
    WHERE ID = (SELECT DISTINCT ID
                FROM Emp WHERE Name = 'Ed') ;

UPDATE Emp
    SET SALARY TO 40000
        VALID PERIOD '[2/1/1985 - 1/31/1987]'
    WHERE ID = (SELECT DISTINCT ID
                FROM Emp WHERE Name = 'Ed') ;

INSERT INTO Skill
    SELECT (ID, 'Typing')
        VALID PERIOD '[4/1/1982 - now]'
        FROM Emp
        WHERE Name = 'Ed' ;
```

```
   INSERT INTO Skill
       SELECT (ID, 'Filing')
           VALID PERIOD '[1/1/1985 - now]'
           FROM Emp
           WHERE Name = 'Ed' ;
   INSERT INTO Skill
       SELECT (ID, 'Driving')
           VALID (PERIOD '[1/1/1982 - 5/1/1982]'
                   + PERIOD '[6/1/1984 - 5/31/1988]')
           FROM Emp
           WHERE Name = 'Ed' ;
```

DI worked in and managed the Toy department from 1/1/82 to the present. Her name is Di throughout her employment. The budget of the Toy department was $150K from 1/1/82 to 7/31/84, $200K from 8/1/84 to 12/31/86, and $100K from 1/1/87 to the present. Her salary was $30K from 1/1/82 to 7/31/84, $40K from 8/1/84 to 8/31/86, then $50K from 9/1/86 to the present. *DI* is female and was born on 10/1/60. *DI* has been qualified for directing from 1/1/82 to the present.

```
INSERT INTO Emp
VALUES (NEW, 'Di', 30000, 'F', DATE '10/1/1960', 'Toy')
VALID PERIOD '[1/1/1982 - now]' ;
INSERT INTO Dept
    SELECT ('Toy', 150000, Emp.ID)
        VALID PERIOD '[1/1/1982 - now]'
        FROM Emp
        WHERE Emp.Name = 'Di' ;
UPDATE Dept
    SET Budget TO 200000
        VALID PERIOD '[8/1/1984 - 12/31/1986]'
    WHERE Name = 'Toy' ;
UPDATE Dept
    SET Budget TO 100000
        VALID PERIOD '[1/1/1987 - now]'
    WHERE Name = 'Toy' ;
UPDATE Emp
    SET SALARY TO 40000
        VALID PERIOD '[8/1/1984 - 8/31/1986]'
    WHERE ID = (SELECT DISTINCT ID
             FROM Emp WHERE Name = 'Di') ;
UPDATE Emp
```

```
    SET SALARY TO 50000 VALID PERIOD '[9/1/1986 - now]'
    WHERE ID = (SELECT DISTINCT ID
            FROM Emp WHERE Name = 'Di') ;
INSERT INTO Skill
    SELECT (ID, 'Directing')
        VALID PERIOD '[1/1/1982 - now]'
        FROM Emp
        WHERE ID = (SELECT ID
            FROM Emp WHERE Name = 'Di') ;
```

These modification statements induce the following table instances. Table emp is as follows.

ID	Name	Salary	Gender	...
ED	Ed	20	M	
ED	Ed	30	M	
ED	Ed	40	M	
ED	Ed	40	M	...
ED	Edward	40	M	
DI	Di	30	F	
DI	Di	40	F	
DI	Di	50	F	

...	D-birth	DeptName	V
	7/1/55	Toy	{[2/1/1982 – 5/31/1982]}
	7/1/55	Toy	{[6/1/1982 – 1/31/1985]}
	7/1/55	Toy	{[2/1/1985 – 1/31/1987]}
...	7/1/55	Book	{[4/1/1987 – 12/31/1987]}
	7/1/55	Book	{[1/1/1988 – *now*]}
	10/1/60	Toy	{[1/1/1982 – 7/31/1984]}
	10/1/60	Toy	{[8/1/1984 – 8/31/1986]}
	10/1/60	Toy	{[9/1/1986 – *now*]}

Table skills is as follows.

EmpID	Skill	V
ED	Typing	{[4/1/1982 – *now*]}
ED	Filing	{[1/1/1985 – *now*]}
ED	Driving	{[1/1/1982 – 5/1/1982], [6/1/1984 – 5/31/1988]}
DI	Directing	{[1/1/1982 – *now*]}

Table `dept` is as follows.

Name	Budget	MgrID	V
Toy	150	DI	{[1/1/1982 – 7/31/1984]}
Toy	200	DI	{[8/1/1984 – 12/31/1986]}
Toy	100	DI	{[1/1/1987 – *now*]}
Book	50	ED	{[4/1/1987 – *now*]}

The present time (i.e., the value of `CURRENT_DATE` for the following queries is 1/1/90.

4 The Test Suite Queries

We adopt the structure of the test suite taxonomy in partitioning the queries into sections.

4.1 Explicit-attribute Output

This section involves queries which return only explicit-attribute values—no valid-time values are present in the result.

Class O1.S1 (Duration, Interval, Computed)

QUERY: Which departments had managers who served for the shortest continuous period?

TSQL2 QUERY:

```
SELECT SNAPSHOT Name
FROM Dept(Name, MgrID)(PERIOD) AS D
WHERE CAST(VALID(D) AS INTERVAL DAY) <=
    ALL (SELECT CAST(VALID(D2) AS INTERVAL DAY)
         FROM Dept(Name, MgrID)(PERIOD) AS D2
         WHERE D2.Name = D.Name)
```

TSQL2 ANSWER: {('Book')}

QUERY: Who worked continuously in the Book department for at least as long as Di did?

TSQL2 QUERY:

```
SELECT SNAPSHOT E4.Name
FROM Emp(ID, DeptName)(PERIOD) AS E1 E2,
    E1(Name) AS E3, E2(Name) AS E4
WHERE E3.Name = 'Di' AND E1.DeptName = 'Book'
    AND E2.DeptName = 'Book'
```

```
        AND CAST(VALID(E2) AS INTERVAL DAY) >
            CAST(VALID(E1) AS INTERVAL DAY)
```

TSQL2 ANSWER: The empty table.

QUERY: Who worked continuously in the Toy department for at least as long as Di did?

TSQL2 QUERY:

```
SELECT SNAPSHOT E4.Name
FROM Emp(ID, DeptName)(PERIOD) AS E1 E2,
    E1(Name) AS E3, E2(Name) AS E4
WHERE E3.Name = 'Di' AND E1.DeptName = 'Toy'
    AND E2.DeptName = 'Toy'
    AND CAST(VALID(E2) AS INTERVAL DAY) >
        CAST(VALID(E1) AS INTERVAL DAY)
```

TSQL2 ANSWER: {('Ed'), ('Edward')}

QUERY: Who worked continuously in a department longer than their current manager worked in that department?

TSQL2 QUERY:

```
SELECT SNAPSHOT E3.Name
FROM Dept(Name, MgrID) AS D, Emp(ID) AS EM,
    Emp(ID, DeptName)(PERIOD) AS E2, E2(Name) AS E3
WHERE D.MgrID = EM.ID
    AND CAST(VALID(E2) AS INTERVAL DAY) >
        CAST(VALID(EM) AS INTERVAL DAY)
```

TSQL2 ANSWER: The empty table.

QUERY: Who had the same salary for the longest continuous time period?

TSQL2 QUERY:

```
SELECT SNAPSHOT Name
FROM Emp(ID, Salary)(PERIOD) AS E, E(Name) AS E2
WHERE CAST(VALID(E) AS INTERVAL DAY) >
    ALL (SELECT CAST(VALID(E3) AS INTERVAL DAY)
         FROM Emp(ID, Salary)(PERIOD) AS E3
         WHERE E3.ID <> E.ID)
```

TSQL2 ANSWER: {('Di')}

QUERY: Who worked for a manager in a department for a period as long as that manager managed the department?

TSQL2 QUERY:

```
SELECT SNAPSHOT E2.Name
FROM Emp(ID,DeptName) AS E,
```

```
        Dept(Name, MgrID)(PERIOD) AS D, E(Name) AS E2
    WHERE E.DeptName = D.Name AND
        VALID(E) CONTAINS VALID(D)
```

TSQL2 ANSWER: {('Ed'), ('Edward'), ('Di')}

QUERY: Which managers served continuously longer than some other manager?
TSQL2 QUERY:

```
    SELECT SNAPSHOT E.Name
    FROM Dept(Name, MgrID)(PERIOD) AS D1 D2,
        Emp(ID, Name) AS E
    WHERE CAST(VALID(D1) AS INTERVAL DAY) >
            CAST(VALID(D2) AS INTERVAL DAY)
        AND D1.MgrID = E.ID
```

TSQL2 ANSWER: {('Di')}

Class O1.S3 (Duration, Element, Computed)

QUERY: Who worked in the Toy department for at least as long as *DI* worked there?
TSQL2 QUERY:

```
    SELECT SNAPSHOT E4.Name
    FROM Emp(ID, DeptName) AS E E2,
        Emp(ID, Name) AS E3 E4
    WHERE E.DeptName = 'Toy' AND E2.DeptName = 'Toy'
        AND E3.Name = 'Di' AND E.ID = E3.ID
        AND E2.ID = E4.ID
        AND CAST(VALID(E2) AS INTERVAL DAY) >=
            CAST(VALID(E) AS INTERVAL DAY)
```

TSQL2 ANSWER: {('Di')}

QUERY: Who worked in a department longer than their current manager worked in that department?
TSQL2 QUERY:

```
    SELECT SNAPSHOT E3.Name
    FROM Emp(ID, DeptName) AS E E2, Emp(ID, Name) AS E3,
        Dept(Name, MgrID) AS D
    WHERE E.DeptName = D.Name AND D.MgrID = E2.ID AND
        E.ID = E3.ID
        AND CAST(VALID(E) AS INTERVAL DAY) >
            CAST(VALID(E2) AS INTERVAL DAY)
```

TSQL2 ANSWER: The empty table.

QUERY: Which managers managed which departments, longer than Di managed the Toy department?

TSQL2 QUERY:

```
SELECT SNAPSHOT E2.Name, D2.Name
FROM Emp(ID, Name) AS E E2, Dept(Name, MgrID) AS D D2
WHERE D.MgrID = E.ID AND E.Name = 'Di'
    AND D.Name = 'Toy' AND D2.MgrID = E2.ID
    AND CAST(VALID(D2) AS INTERVAL DAY) >
        CAST(VALID(D) AS INTERVAL DAY)
```

TSQL2 ANSWER: The empty table.

QUERY: Who had the same salary for the longest total time?

TSQL2 QUERY:

```
SELECT SNAPSHOT E2.NAME
FROM Emp(ID,Salary) as E1, Emp(ID,Name) as E2
    (SELECT MAX(CAST(VALID(E) AS INTERVAL DAY)
     FROM Emp(ID,Salary)(PERIOD) AS E) AS E3
WHERE E2.ID = E1.ID
```

TSQL2 ANSWER: {('Ed')}

QUERY: Which departments had managers who served for the shortest total time?

TSQL2 QUERY:

```
SELECT SNAPSHOT D1.Name
FROM Dept(Name,MgrID) AS D1,
    (SELECT MgrID,
       Duration = MIN(CAST(VALID(D2) AS INTERVAL DAY))
    FROM DeptName(Name, MgrId) AS D2) AS D
WHERE CAST(VALID(D1) AS INTERVAL DAY) = D.Duration
    AND D.MgrID = D1.MgrID
```

TSQL2 ANSWER: {('Book)'}

QUERY: List all employees currently in the Book department who received salaries of over 40K longer than *ED* did.

TSQL2 QUERY:

```
SELECT SNAPSHOT E3.Name
FROM Emp(ID, DeptName)(PERIOD) E,
    Emp(ID, Salary) AS E1 E2, Emp(ID, Name) AS E3 E4
WHERE E.DeptName = 'Book' AND E OVERLAPS CURRENT_DATE
    AND E.ID = E3.ID
    AND E1.Salary > 40000 AND E2.Salary > 40000
```

```
    AND E3.ID = E1.ID AND E4.ID = E2.ID
    AND E4.Name = 'Ed'
    AND CAST(VALID(E1) AS INTERVAL DAY) >
        CAST(VALID(E2) AS INTERVAL DAY)
```

TSQL2 ANSWER: The empty table.

QUERY: Who worked in the Toy department for at least as long as the total time that the Toy department was *not* managed by *ED*?

TSQL2 QUERY:

```
SELECT SNAPSHOT E2.Name
FROM Emp(ID, DeptName) AS E, Emp(ID, Name) AS E2 E3,
    Dept(Name) AS D, Dept(Name, MgrID) AS D2
WHERE E.DeptName= 'Toy' AND E2.ID = E.ID
    AND D.Name = 'Toy' AND D2.Name = 'Toy'
    AND D2.MgrID = E3.ID AND E3.Name = 'Ed'
    AND CAST(VALID(E) AS INTERVAL DAY) >=
        (VALID(D) - VALID(D2)) DAY
```

TSQL2 ANSWER: {('Di')}

QUERY: Find the names of employees that have been in a department named Toy for a shorter period than has *DI*.

TSQL2 QUERY:

```
SELECT SNAPSHOT E.Name
FROM Emp(ID, Name) AS E E2,
    Emp(ID, DeptName) AS E3 E4
WHERE E.ID = E3.ID AND E2.ID = E4.ID
    AND E2.Name = 'Di' AND E3.DeptName = 'Toy'
    AND E4.DeptName = 'Toy'
    AND CAST(VALID(E3) AS INTERVAL DAY) <
        CAST(VALID(E4) AS INTERVAL DAY)
```

TSQL2 ANSWER: ('Ed') and ('Edward')

QUERY: Find the current name and department for the employees which made $40K for a longer period than *DI* did.

TSQL2 QUERY:

```
SELECT SNAPSHOT E.Name, E.DeptName
FROM Emp(ID, Name, DeptName) AS E,
    Emp(ID, Salary) AS E2 E3, Emp(ID, Name) AS E4
WHERE E.ID = E2.ID AND VALID(E) OVERLAPS CURRENT_DATE
    AND E3.ID = E4.ID AND E4.Name = 'Di'
    AND E2.Salary = 40000 and E3.Salary = 40000
```

```
    AND CAST(VALID(E2) AS INTERVAL DAY) >
        CAST(VALID(E3) AS INTERVAL DAY)
```

TSQL2 ANSWER: {('Edward', 'Book')}

Class O1.S9 (Other, Element, Computed)

QUERY: Find *ED*'s salaries when he worked in the same department as *DI*.
TSQL2 QUERY:

```
SELECT SNAPSHOT E3.Salary
FROM Emp(ID, DeptName) AS E1 E2, E1(Salary) AS E3
WHERE E1.ID = 'ED' AND E2.ID = 'DI'
    AND E1.DeptName = E2.DeptName
```

TSQL2 ANSWER: {(20000), (30000), (40000)}

QUERY: Find the names of departments that always had a budget greater than $90K during the times when managed by someone named Di.
TSQL2 QUERY:

```
SELECT SNAPSHOT D2.Name
FROM Dept(ID, Name, MgrID) AS D1,
    Dept(ID, Budget)(PERIOD) AS D2, Emp(ID, Name) AS M
WHERE D1.MgrID = M.ID AND M.Name = 'Di'
    AND D2.Budget > 90 AND D1.ID = D2.ID
    AND VALID(D2) CONTAINS VALID(D1)
```

TSQL2 ANSWER: {('Toy')}

QUERY: Find the names of the departments that *ED* worked in while earning $40K.
TSQL2 QUERY:

```
SELECT SNAPSHOT DeptName
FROM Emp(ID, Salary, DeptName) AS E
WHERE ID = 'Ed' AND Salary = 40000
```

TSQL2 ANSWER: {('Toy'), ('Book')}

QUERY: Find *ED*'s names after he left the Toy department.
TSQL2 QUERY:

```
SELECT SNAPSHOT E3.Name
FROM Emp(ID, DeptName) (PERIOD) AS E1 E2,
    E2(Name) AS E3
WHERE E1.DeptName = 'Toy' AND E2.DeptName <> 'Toy'
    AND E1.ID = E2.ID AND VALID(E1) MEETS VALID(E2)
```

TSQL2 ANSWER: {('Ed'), ('Edward')}

QUERY: Find the skills that *ED* possessed sometime when he worked in the Toy department.
TSQL2 QUERY:

```
SELECT SNAPSHOT S1.Skill
FROM Emp(ID, DeptName) AS E1, Skills(EmpID, Skill) S1
WHERE E1.ID = 'ED' AND E1.DeptName = 'Toy'
    AND E1.ID = S1.EmpID
```

TSQL2 ANSWER: {('Driving'), ('Filing'), ('Typing')}

QUERY: What new skills did *ED* obtain after he had changed his name to Edward?
TSQL2 QUERY:

```
SELECT SNAPSHOT S1.Skill
FROM Emp(ID, Name) (PERIOD) AS E1 E2,
    Skills(EmpID, Skill) S1
WHERE E1.Name = 'Ed' AND E2.Name = 'Edward'
    AND E1.ID = E2.ID AND VALID(E1) MEETS VALID(E2)
    AND E2.ID = S1.EmpID
MINUS
SELECT SNAPSHOT S1.Skill
FROM Emp(ID, Name) (PERIOD) AS E1 E2,
    Skills(EmpID, Skill) S1
WHERE E1.ID = 'ED' AND E1.ID = S1.EmpID AND
    E1.Name <> 'Edward' AND E2.Name = 'Edward'
    AND E1.ID = E2.ID AND VALID(E1) MEETS VALID(E2)
```

TSQL2 ANSWER: The empty table.

QUERY: What were Toy's departmental budgets when it had a manager named Di?
TSQL2 QUERY:

```
SELECT SNAPSHOT D1.Budget
FROM Dept(Name, MgrID, Budget) AS D1,
    Emp(ID, Name) AS M
WHERE D1.MgrID = M.ID AND M.Name = 'Di' AND
    D1.Name = 'Toy'
```

TSQL2 ANSWER: {(150000), (200000), (100000)}

Class O2.S2 (Duration, Element, Other)

QUERY: Find employment periods in the Toy department for persons that have worked there for at least 8 years.

TSQL2 QUERY:

```
SELECT VALID(E)
FROM Emp(ID)(PERIOD) AS E
WHERE DeptName = 'Toy'
    AND CAST(VALID(Emp) AS INTERVAL YEAR) >
        INTERVAL '8' YEAR
```

TSQL2 ANSWER: {([1/1/1982 - now])}

QUERY: Find the starting times of managers which managed a department for at least 5 years.

TSQL2 QUERY:

```
SELECT BEGIN(VALID(D))
FROM Dept(MgrID, Name) (PERIOD) AS D
WHERE CAST(VALID(D) AS INTERVAL YEAR) >
    INTERVAL '5' YEAR
```

TSQL2 ANSWER: {(1/1/1982)}

QUERY: Find the rehiring dates of employees with a gap in employment that exceeds 1 month.

TSQL2 QUERY:

```
SELECT BEGIN(VALID(E2))
FROM Emp(ID) (PERIOD) AS E1 E2
WHERE (E1.ID = E2.ID)
    AND VALID(E1) PRECEDES VALID(E2)
    AND ((END(VALID(E2)) - BEGIN(VALID(E1))) MONTH) >
        INTERVAL '1' MONTH
```

TSQL2 ANSWER: {(4/1/1987)}

QUERY: Find the times when persons possessed skills that they lost and regained more than 1 year later.

TSQL2 QUERY:

```
SELECT VALID(S1), VALID(S2)
FROM Skills(ID, Name)(INTERVAL) AS S1 S2
WHERE S1.ID = S2.ID AND S1.Name = S2.Name
    AND VALID(S1) PRECEDES VALID(S2)
    AND ((END(VALID(S2)) - BEGIN(VALID(S1)) YEAR) >
```

```
          INTERVAL '1' YEAR
```

TSQL2 ANSWER:

```
{( [1/1/1982-5/1/1982],[6/1/1984-5/31/1988] )}
```

QUERY: Find budget periods that exceed 2 years.

TSQL2 QUERY:

```
SELECT VALID(D)
FROM Dept(Name, Budget) (PERIOD) AS D
WHERE CAST(VALID(D) AS INTERVAL YEAR) >
      INTERVAL '2' YEAR
```

TSQL2 ANSWER: {([1/1/1987 - now], [4/1/1987 - now])}

QUERY: When did no one's salary change for at least six months?

TSQL2 QUERY:

```
SELECT VALID(E)
FROM Emp(Salary) (PERIOD) AS E
WHERE CAST(VALID(E) AS INTERVAL MONTH) =>
      INTERVAL '6' MONTH
```

TSQL2 ANSWER:

```
{([6/1/1992 - 7/31/1984]), ([8/1/1984 - 1/31/1985]),
([2/1/1985 - 8/31/1986]), ([9/1/1986 - now])}
```

Class O2.S9 (Other, Element, Computed)

QUERY: At what times did an employee simultaneously posses at least the same skills that *DI* possessed?

TSQL2 QUERY:

```
SELECT SNAPSHOT INTERSECT(VALID(E),VALID(S))
FROM Emp(ID) AS E, Skills(EmpID, Skill) AS S
WHERE E.ID = Skills.EmpID AND Skills.Skill CONTAINS
    (SELECT SDi.Skill
     FROM Emp(ID,Name) AS EDi,
       Skills(EmpID, Skill) AS SDi
     WHERE EDi.Name = 'Di' AND EDi.ID = SDi.EmpID)
```

TSQL2 ANSWER: An empty table.

QUERY: When was the budget for Toy department more than 100K?
TSQL2 QUERY:

```
SELECT VALID(D)
FROM Dept(Name, Budget) AS D
WHERE D.Budget > 100 AND D.Name = 'Toy'
```

TSQL2 ANSWER: {([1/1/82-12/31/86])}

QUERY: What was the last continuous period when *ED* was named Edward and had Driving, Filing and Typing as skills simultaneously?
TSQL2 QUERY:

```
SELECT SNAPSHOT
    MAX(INTERSECT(VALID(E), INTERSECT(VALID(SDrive),
       INTERSECT(VALID(SFile),VALID(SType)))))
FROM Emp(ID,Name)(PERIOD) AS E,
    Skills(EmpID,Skill)(PERIOD) AS SDrive SFile SType
WHERE E.Name = 'Edward' AND E.ID = SDrive.EmpID
    AND E.ID = SFile.EmpID AND E.ID = SType.EmpID
    AND VALID(E) OVERLAPS INTERSECT(VALID(SFile),
            INTERSECT(VALID(SDrive),VALID(SType)))
```

TSQL2 ANSWER: {([1/1/1988 - 5/31/1988])}

QUERY: When did *DI* earn less than *ED*?
TSQL2 QUERY:

```
SELECT SNAPSHOT VALID(E1)
FROM Emp(Name, Salary)(PERIOD) AS E1 E2
WHERE E1.Name = 'Di' AND E2.Name = 'Ed'
    AND E1.Salary < E2.Salary
```

TSQL2 ANSWER: An empty table.

QUERY: When did *ED* work in Toy department while the department was managed by *DI*?
TSQL2 QUERY:

```
SELECT SNAPSHOT
    INTERSECT(VALID(VALID(E1)),VALID(VALID(E2)))
FROM Emp(ID, Name, Salary, DeptName) AS E1 E2,
    Dept(MgrID, Name) AS D1
WHERE E1.Name = 'Di' AND E2.Name = 'Ed'
    AND E1.ID = D1.MgrID AND D1.Name = 'Toy'
    AND E2.DeptName = 'Toy'
    AND VALID(D1) OVERLAPS VALID(E2)
```

TSQL2 ANSWER: {([2/1/82 - 1/31/87])}

QUERY: When did an employee currently named Edward have driving skills?
TSQL2 QUERY:

```
SELECT VALID(S)
FROM Emp(ID, Name) AS E,
    SKills(EmpID, Skill)(PERIOD) AS S
WHERE E.Name = 'Edward' AND E.ID = S.EmpID
    AND S.Skill = 'Driving'
```

TSQL2 ANSWER:

```
{([1/1/1982 - 5/1/1982]), ([6/1/1984 - 5/31/1988])}
```

4.2 Explicit-attribute and Valid-time Output

The output from queries in this section contains explicit attribute values with associated valid times.

Class O3.S1 (Duration, Interval, Computed)

QUERY: Who, and when, were continuously employed in the Toy department shorter than *DI* was continuously employed in the Toy department?
TSQL2 QUERY:

```
SELECT e2.Name
FROM Emp(ID, DeptName)(PERIOD) AS e1, e1(Name) AS e2,
    Emp(ID, DeptName)(PERIOD) AS e3
WHERE e3.ID =
    ( SELECT ID
      FROM Emp
      WHERE Name = 'Di'
        AND VALID(Emp) CONTAINS CURRENT_DATE )
    AND e1.DeptName = 'Toy' AND e3.DeptName = 'Toy'
    AND CAST(VALID(e1) AS INTERVAL DAY) <
        CAST(VALID(e3) AS INTERVAL DAY)
```

TSQL2 ANSWER: `{( 'Ed' | [2/1/82 - 1/31/87]) }`

QUERY: Who, and when, were continuously employed in the Toy department shorter than *DI* was continuously employed in the Toy department, and what are their gender and date of birth?
TSQL2 QUERY:

```
SELECT e2.Name, e2.Gender, e2.D-birth
FROM Emp(ID, DeptName)(PERIOD) AS e1,
    e1(Name, Gender, D-birth) AS e2,
```

```
    Emp(ID, DeptName)(PERIOD) AS e3
WHERE e3.ID =
    ( SELECT ID
      FROM Emp
      WHERE Name = 'Di'
        AND VALID(Emp) CONTAINS CURRENT_DATE )
    AND e1.DeptName = 'Toy' AND e3.DeptName = 'Toy'
    AND CAST(VALID(e1) AS INTERVAL DAY) <
        CAST(VALID(e3) AS INTERVAL DAY)
```

TSQL2 ANSWER:

```
{( 'Ed', 'M', 7/1/1955 | [2/1/82 - 1/31/87]) }
```

QUERY: Who were continuously employed in the Toy department shorter than *DI* was continuously employed in the Toy department, and when did this employment start?

TSQL2 QUERY:

```
SELECT SNAPSHOT e2.Name, BEGIN(e2)
FROM Emp(ID, DeptName)(PERIOD) AS e1, e1(Name) AS e2,
    Emp(ID, DeptName)(PERIOD) AS e3
WHERE e3.ID =
    ( SELECT ID
      FROM Emp
      WHERE Name = 'Di'
        AND VALID(Emp) CONTAINS CURRENT_DATE )
    AND e1.DeptName = 'Toy' AND e3.DeptName = 'Toy'
    AND CAST(VALID(e1) AS INTERVAL DAY) <
        CAST(VALID(e3) AS INTERVAL DAY)
```

TSQL2 ANSWER: `{( 'Ed', 2/1/1982 ) }`

QUERY: Return the name on 01-Jan-1984 along with the date 01-Jan-1984 for each employee who was continuously employed in the Toy department shorter than *DI* was continuously employed in that department.

TSQL2 QUERY:

```
SELECT SNAPSHOT e2.Name, DATE '1/1/84'
FROM Emp(ID, DeptName)(PERIOD) AS e1, e1(Name) AS e2,
    Emp(ID, DeptName)(PERIOD) AS e3
WHERE e3.ID =
    ( SELECT ID
      FROM Emp
      WHERE Name = 'Di'
        AND VALID(Emp) CONTAINS CURRENT_DATE)
```

```
        AND e1.DeptName = 'Toy' AND e3.DeptName = 'Toy'
        AND CAST(VALID(e1) AS INTERVAL DAY) <
            CAST(VALID(e3) AS INTERVAL DAY)
        AND VALID(e2) OVERLAPS DATE '1/1/84'
```

TSQL2 ANSWER: {('Ed', 1/1/1984) }

Class O3.S2 (Duration, Interval, Other)

QUERY: When was the Toy department's budget constant and greater than $175K for more than one year, and what was the budget?

TSQL2 QUERY:

```
SELECT d1.Budget
FROM Dept(Name, Budget)(PERIOD) AS d1
WHERE d1.Name = 'Toy'
    AND CAST(VALID(d1) AS INTERVAL YEAR) >
        INTERVAL '1' YEAR
    AND d1.Budget > 175000
```

TSQL2 ANSWER: {(200000 | [8/1/1984 - 12/31/1986]) }

QUERY: When was the Toy department's budget constant and greater than $175K for more than one year, and who was the manager for that time?

TSQL2 QUERY:

```
SELECT e1.Name
VALID d1
FROM Dept(Name, Budget)(PERIOD) AS d1,
    d1(MgrID) AS d2, Emp AS e1
WHERE d1.Name = 'Toy'
    AND CAST(VALID(d1) AS INTERVAL DAY) >
        INTERVAL '1' YEAR
    AND d1.Budget > 175000 AND d2.MgrID = e1.ID
```

TSQL2 ANSWER: {('Di' | [8/1/1984 - 12/31/1986]) }

QUERY: Who managed a department with a budget that exceeded $175K and then held constant for one year, and when did that occur?

TSQL2 QUERY:

```
SELECT e1.Name
VALID d1
FROM Dept(Name, Budget)(PERIOD) AS d1,
    d1(MgrID) AS d2, Emp AS e1
WHERE CAST(VALID(d1) AS INTERVAL DAY) >
    INTERVAL '1' YEAR AND d1.Budget > 175000
```

```
      AND d2.MgrID = e1.ID
```
TSQL2 ANSWER: {('Di' | [8/1/1984 - 12/31/1986]) }

QUERY: What departments were in continuous operation (and when) longer than the duration between *ED*'s and *DI*'s birth dates?

TSQL2 QUERY:
```
SELECT d1.Name
FROM Dept(Name)(PERIOD) AS d1
WHERE CAST(VALID(d1) AS INTERVAL DAY) >
    (LAST
       (SELECT D-birth FROM Emp WHERE Name = 'Edward',
        SELECT D-birth FROM Emp WHERE Name = 'Di') -
    FIRST
       (SELECT D-birth FROM Emp WHERE Name = 'Edward',
        SELECT D-birth FROM Emp WHERE Name = 'Di')) DAY
```
TSQL2 ANSWER: {('Toy' | [1/1/1982 - 1/1/1990]) }

QUERY: Who worked in one department for at least two years continuously and what were the periods of employment in that department?

TSQL2 QUERY:
```
SELECT E2.Name
FROM Emp(ID, DeptName)(PERIOD) AS E1, Emp AS E2
WHERE CAST(VALID(E1) AS INTERVAL YEAR) >
      INTERVAL '2' YEAR
    AND E1.ID = E2.ID AND E1.DeptName = E2.DeptName
```
TSQL2 ANSWER: {('Ed' | [1/1/1982 - 1/31/1987]),
('Ed' | [4/1/1987- 12/31/1987]),
('Edward' | [1/1/1988 - 1/1/1990]),
('Di' | [1/1/1982 - 1/1/1990])}

References

[1] Jensen (ed.), C. S. "A Consensus Test Suite of Temporal Database Queries." Technical Report R 93-2034. Department of Mathematics and Computer, Institute for Electronic Systems. Nov. 1993.

[2] Snodgrass, R. T., (editor) "An Evaluation of TSQL2." Commentary. TSQL2 Design Committee. Sep. 1994.

[3] Jensen, C. S., R. T. Snodgrass and M. D. Soo. "Extending Normal Forms to Temporal Relations." TR 92-17. Computer Science Department, University of Arizona. July 1992.