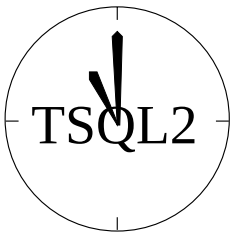

An Evaluation of TSQL2

October 9, 1994

A TSQL2 Commentary

The TSQL2 Language Design Committee

Title	An Evaluation of TSQL2
Primary Author(s)	Richard T. Snodgrass (editor), Ilsoo Ahn, Gad Ariav, Petra Bayer, James Clifford, Curtis Dyreson, Fabio Grandi, Luis Hermosilla, Christian S. Jensen, Wolfgang Käfer, Nick Kline, T.Y. Cliff Leung, Nikos Lorentzos, Y. Mitsopoulos, John F. Roddick, Michael Soo, and Suryanarayana M. Sripada
Publication History	October, 1994. TSQL2 Commentary.

TSQL2 Language Design Committee

Richard T. Snodgrass, Chair	University of Arizona
rts@cs.arizona.edu	Tucson, AZ
Ilsoo Ahn	Samsung Electronics Co.
isahn@trvax5.sait.samsung.co.kr	Seoul, Korea
Gad Ariav	Tel Aviv University
ariavg@ccmail.gsm.uci.edu	Tel Aviv, Israel
Don Batory	University of Texas
dsb@cs.utexas.edu	Austin, TX
James Clifford	New York University
jcliffor@is-4.stern.nyu.edu	New York, NY
Curtis E. Dyreson	University of Arizona
curtis@cs.arizona.edu	Tucson, AZ
Ramez Elmasri	University of Texas
elmasri@cse.uta.edu	Arlington, TX
Fabio Grandi	Universitá di Bologna
fabio@deis64.cineca.it	Bologna, Italy
Christian S. Jensen	Aalborg University
csj@iesd.auc.dk	Aalborg, Denmark
Wolfgang Käfer	Daimler Benz
kaefer%fuzi.uucp@germany.eu.net	Ulm, Germany
Nick Kline	University of Arizona
kline@cs.arizona.edu	Tucson, AZ
Krishna Kulkarni	Tandem Computers
kulkarni_krishna@tandem.com	Cupertino, CA
T. Y. Cliff Leung	Data Base Technology Institute, IBM
cleung@vnet.ibm.com	San Jose, CA
Nikos Lorentzos	Agricultural University of Athens
eliop@isosun.ariadne-t.gr	Athens, Greece
John F. Roddick	University of South Australia
roddick@unisa.edu.au	The Levels, South Australia
Arie Segev	University of California
segev@csr.lbl.gov	Berkeley, CA
Michael D. Soo	University of Arizona
soo@cs.arizona.edu	Tucson, AZ
Suryanarayana M. Sripada	European Computer-Industry Research Centre
sripada@ecrc.de	Munich, Germany

Contents

Abstract

This document evaluates the TSQL2 language against a sizable consensus test suite of temporal database queries. The test suite consists of a database schema, an instance for the schema, and a set of approximately 150 queries on this database. The reader is cautioned that the queries have not been independently validated nor tested on an implementation of TSQL2. Given the number and complexity of some of the queries, there are most certainly errors.

1 Introduction

The central goal of this document is to evaluate the TSQL2 language specification against an extensive consensus test suite for temporal relational query languages [Jensen et al. 1993]. 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 [Jensen et al. 1993].

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 document is Section 4, in which the test suite queries are given, both in English and in TSQL2.

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.

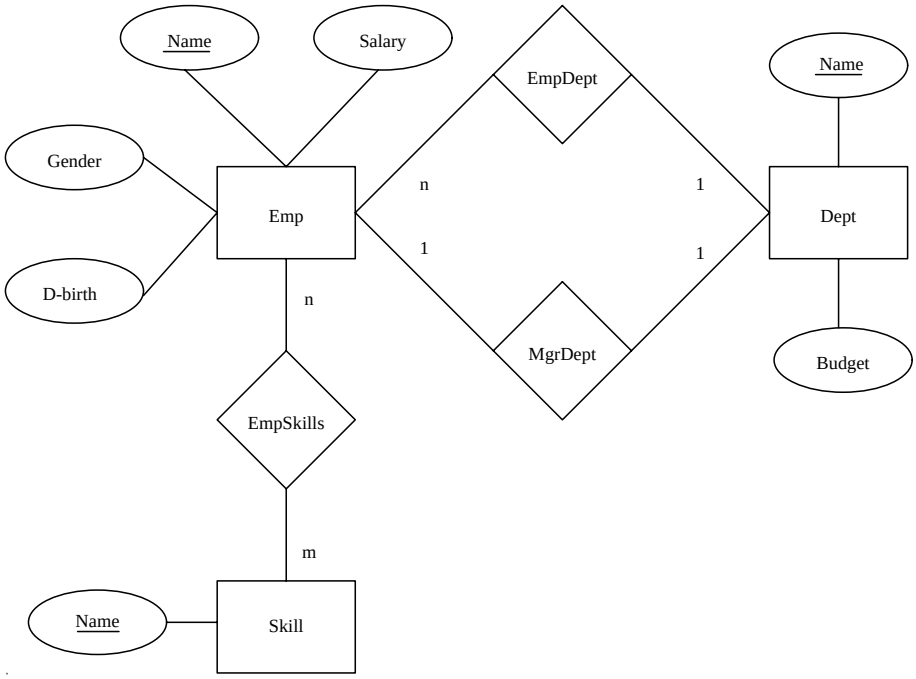

Figure 1: ER Diagram of Database Design

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 [Jensen et al. 1992].

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	D-birth	DeptName	V
ED	Ed	20	M	7/1/55	Toy	$\{[2/1/1982 - 5/31/1982]\}$
ED	Ed	30	M	7/1/55	Toy	$\{[6/1/1982 - 1/31/1985]\}$
ED	Ed	40	M	7/1/55	Toy	$\{[2/1/1985 - 1/31/1987]\}$
ED	Ed	40	M	7/1/55	Book	$\{[4/1/1987 - 12/31/1987]\}$
ED	Edward	40	M	7/1/55	Book	$\{[1/1/1988 - now]\}$
DI	Di	30	F	10/1/60	Toy	$\{[1/1/1982 - 7/31/1984]\}$
DI	Di	40	F	10/1/60	Toy	$\{[8/1/1984 - 8/31/1986]\}$
DI	Di	50	F	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.

4.1.1 Class O1.S1 (Duration, Interval, Computed)

QUERY Q 1.1.1: Which departments had managers who served for the shortest continuous period?
EXPECTED ANSWER: "Book"
CATEGORY: (Projected, None) / (Duration, Interval, Computed) /
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')}
COMMENTS: A "continuous period" is indicated with a restructuring partition of `PERIOD`. Shortest could be done with a correlation query, as above, or with an aggregate.

QUERY Q 1.1.2: Who worked continuously in the Book department for at least as long as Di did?
EXPECTED ANSWER: "*ED*"
CATEGORY: (Projected, None) / (Duration, Interval, Computed) / (=, Constant) (=, Constant)
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, because Di never worked in the book department. The expected answer is incorrect.
COMMENTS: `E3` and `E4` are required because the predicate didn't say 'continuously with the same name'.

QUERY Q 1.1.3: Who worked continuously in the Toy department for at least as long as Di did?
EXPECTED ANSWER: "None"
CATEGORY: (Projected, None) / (Duration, Interval, Computed) / (=, Constant) (=, Constant)
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')}
COMMENTS: Di did work in the Toy department, so this query returns the correct result.

QUERY Q 1.1.4: Who worked continuously in a department longer than their current manager worked in that department?

EXPECTED ANSWER: "None"
CATEGORY: (Projected, None) / (Duration, Interval, Computed) (Containment, Event, Explicit) /
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.
COMMENTS: Here the we compare the duration of an interval with the duration of a temporal element, as one is specified as 'continuously' and the other is not.

QUERY Q 1.1.5: Who had the same salary for the longest continuous time period?
EXPECTED ANSWER: "*DI*"
CATEGORY: (Projected, None) / (Duration, Interval, Computed) /
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')}
COMMENTS: As only `E2` has a `Name` attribute, there is no need to specify the correlation name in the select clause.

QUERY Q 1.1.6: Who worked for a manager in a department for a period as long as that manager managed the department?
EXPECTED ANSWER: "*DI*" and "*ED*"
CATEGORY: (Projected, None) / (Duration, Interval, Computed) /
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')}
COMMENTS: Since "`(PERIOD)`" was used, the `Dept` correlation variable was partitioned into maximal periods.

QUERY Q 1.1.7: Which managers served continuously longer than some other manager?
EXPECTED ANSWER: "*DI*"
CATEGORY: (Projected, None) / (Duration, Interval, Computed) /
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')}
COMMENTS: We don't need to know the name of 'some other manager.'

4.1.2 Class O1.S2 (Duration, Interval, Other)

QUERY Q 1.2.1: Which employees had the same salary for a single period of at least three years?
EXPECTED ANSWER: "*DI*"
CATEGORY: (Projected, None) / (Duration, Interval, Explicit) /
TSQL2 QUERY:

```
SELECT SNAPSHOT E2.Name
FROM Emp(ID, Salary)(PERIOD) AS E1, E1(Name) AS E2
WHERE CAST(VALID(E1)) AS INTERVAL YEAR) >= INTERVAL '3' YEAR
```

TSQL2 ANSWER: {('Di')}
COMMENTS: 'Single period' translates into a partitioning by maximal period.

QUERY Q 1.2.2: Who worked for the same manager for at least five years continuously?
EXPECTED ANSWER: "*ED*" and "*DI*"
CATEGORY: (Projected, None) / (Duration, Interval, Explicit) /
TSQL2 QUERY:

```
SELECT SNAPSHOT E2.Name
FROM (SELECT E3.ID FROM Emp(ID,DeptName) AS E3, Dept(Name,MgrID) AS D
           WHERE DeptName=Name)(PERIOD) AS E,
       Emp(ID, Name) AS E2
WHERE E.ID = E2.ID AND CAST(VALID(E)) AS INTERVAL YEAR) >= INTERVAL '5' YEAR
```

TSQL2 ANSWER: {('Ed'), ('Edward'), ('Di')}
COMMENTS: The result of the select in the from clause (with a default of `VALID INTERSECT(E3, D)`) is coalesced, then partitioned into maximal periods to test the 'for at least five years continuously' requirement.

QUERY Q 1.2.3: Which employees have stayed in the same department throughout the past 5 years?
EXPECTED ANSWER: "*DI*"
CATEGORY: (Projected, None) / (Duration, Interval, Explicit) (Containment, Event, Explicit) /
TSQL2 QUERY:

```
SELECT SNAPSHOT E2.Name
FROM Emp(ID,DeptName)(PERIOD) AS E1, E1(Name) AS E2
WHERE CAST(VALID(E1) AS INTERVAL DAY) >= INTERVAL '5' YEAR
    AND VALID(E1) MEETS CURRENT_DATE
```

TSQL2 ANSWER: {('Di')}

QUERY Q 1.2.4: For each department which has had the same managers and budget for the last 18 months, list its current name, manager, and budget.
EXPECTED ANSWER: "(Toy, *DI*, 100K)" and "(Book, *ED*, 50K)"
CATEGORY: (Complete, None) / (Duration, Interval, Explicit) (Containment, Event, Explicit) /
TSQL2 QUERY:

```
SELECT D2.Name, E.Name, D2.Budget
FROM Dept(Name, MgrID, Budget)(PERIOD) AS D1 D2, Emp(ID, Name) AS E
```

```
WHERE CAST(VALID(D1) AS INTERVAL MONTH) >= INTERVAL '18' MONTH
    AND VALID(D1) OVERLAPS CURRENT_DATE AND VALID(D2) OVERLAPS CURRENT_DATE
    AND D1.Name = D2.Name AND D2.MgrID = E.ID AND VALID(E) OVERLAPS CURRENT_DATE
```

TSQL2 Answer: {('Toy, 'Di', 100K), ('Book', 'Edward', 50K)}
Comments: We extracted the current name of the manager, here Edward instead of Ed.

Query Q 1.2.5: Who has worked in the Toy department and has earned at least 40K throughout the last two years?
Expected Answer: "*ED*" and "*DI*"
Category: (Projected, None) / (Duration, Interval, Explicit) / (=, Constant) (<>, Constant)
TSQL2 Query:

```
SELECT SNAPSHOT E2.Name
FROM Emp(ID,DeptName) AS E1, E1(Name) AS E2,
       (SELECT ID FROM Emp(ID,Salary) AS E
            WHERE ID=E1.ID AND SALARY >= 40000)(PERIOD) AS S
WHERE E1.DeptName = 'Toy' AND CAST(VALID(S) AS INTERVAL DAY) >= INTERVAL '2' YEAR
    AND VALID(S) MEETS CURRENT_DATE
```

TSQL2 Answer: {('Ed'), ('Di')}
Comments: Here we arbitrarily grab the employee's name when they worked in the Toy department.

Query Q 1.2.6: Who had at least three raises in a continuous five-year period?
Expected Answer: "None"
Category: (Projected, None) / (Duration, Interval, Explicit) /
TSQL2 Query:

```
SELECT SNAPSHOT E1.Name
FROM Emp(ID,Name) AS E1,
       (SELECT ID, CNT=COUNT(ID) FROM Emp(ID,A.SALARY) AS E
            GROUP BY ID, VALID(E) USING INSTANT LEADING INTERVAL '5' YEAR) AS Temp
WHERE Temp.Cnt >= 3 and E1.ID = Temp.ID
```

TSQL2 Answer: {('Di'), ('Edward')}
Comments: In the nested query, for each ID, we compute the number of salary changes in a 5 year period.

Query Q 1.2.7: Who had the most raises in a continuous five-year period?
Expected Answer: "*ED*" and "*DI*"
Category: (Projected, None) / (Duration, Interval, Explicit) /
TSQL2 Query:

```
SELECT SNAPSHOT E1.Name
FROM Emp(ID,Name) AS E1,
       (SELECT ID,COUNT(ID) FROM Emp(ID,A.SALARY) AS E
            GROUP BY ID, VALID(E) USING INSTANT LEADING INTERVAL '5' YEAR) AS Temp1(ID,Cnt)
       (SELECT MAX(Temp2.Cnt)
            From (SELECT COUNT(ID) FROM Emp(ID,A.SALARY) AS E
            GROUP ID, VALID(E) USING INSTANT LEADING INTERVAL '5' YEAR)
                AS Temp2(Cnt)) AS Temp3(SalMax)
WHERE Temp1.Cnt = Temp3.SalMax and E1.ID = Temp1.Id
```

TSQL2 Answer: {``Di'', ``Edward''}

4.1.3 Class O1.S3 (Duration, Element, Computed)

QUERY Q 1.3.1: Who worked in the Toy department for at least as long as *DI* worked there?
EXPECTED ANSWER: "*DI*"
CATEGORY: (Projected, None) / (Duration, Element, Computed) / (=, Constant) (=, Constant)
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')}
COMMENTS: Here the parallelism between the `ID` and the `Name` is more pronounced.

QUERY Q 1.3.2: Who worked in a department longer than their current manager worked in that department?
EXPECTED ANSWER: "None"
CATEGORY: (Projected, None) / (Duration, Element, Computed) (Containment, Event, Explicit) /
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 Q 1.3.3: Which managers managed which departments, longer than Di managed the Toy department?
EXPECTED ANSWER: "None"
CATEGORY: (Projected, None) / (Duration, Element, Computed) / (=, Constant) (=, Constant)
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.
COMMENTS: The symmetry indicates that we are interested in managers who were ever called "Di". If we were only interested in those who were called "Di" when they managed the Toy department, we would need the additional predicate `AND VALID(E) COVERS VALID(D)`.

QUERY Q 1.3.4: Who had the same salary for the longest total time?
EXPECTED ANSWER: "*ED*"
CATEGORY: (Projected, None) / (Duration, Element, Computed) /
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')}
Comments: The nested query selects the largest time during which an employee had the same salary.

Query Q 1.3.5: Which departments had managers who served for the shortest total time?
Expected Answer: Answer???
Category: (Projected, None) / (Duration, Element, Computed) /
TSQL2 Query:

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

TSQL2 Answer: {('Book)'}
Comments: The nested query selects the shortest amount of time a department had a single manager.

Query Q 1.3.6: List all employees currently in the Book department who received salaries of over 40K longer than *ED* did.
Expected Answer: "None"
Category: (Projected, None) / (Duration, Element, Computed) / (=, Constant) (<>, Constant) (=, Constant)
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 Q 1.3.7: Who worked in the Toy department for at least as long as the total time that the Toy department was *not* managed by *ED*?
Expected Answer: "*DI*"
Category: (Projected, None) / (Duration, Element, Computed) / (=, Constant) (<>, Constant)
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')}

COMMENTS: Here we take the temporal element when anyone managed the Toy department, and remove those times when Ed managed that department.

QUERY Q 1.3.8: Find the names of employees that have been in a department named Toy for a shorter period than has *DI*.
EXPECTED ANSWER: "Ed" and "Edward."
CATEGORY: (Projected, None) / (Duration, Element, Computed) / (=, Constant) (=, Constant)
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 Q 1.3.9: Find the current name and department for the employees which made $40K for a longer period than *DI* did.
EXPECTED ANSWER: "(Edward, Book)."
CATEGORY: (Projected, None) / (Duration, Element, Computed) (Containment, Event, Explicit) / (=, Constant)
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')`}

4.1.4 Class O1.S4 (Duration, Element, Other)

QUERY Q 1.4.1: Who managed the Book department for at least two years?
EXPECTED ANSWER: "*ED*"
CATEGORY: (Projected, None) / (Duration, Element, Explicit) / (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT Name
FROM Emp(ID, Name) AS E, Dept(Name,MgrID) AS D
WHERE E.ID = MgrID AND Dept.Name = 'Book'
    AND CAST(VALID(D) AS INTERVAL YEAR) >= INTERVAL '2' YEAR
```

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

QUERY Q 1.4.2: Which employees had the same salary for at least three years?

EXPECTED ANSWER: "*ED*" and "*DI*"
CATEGORY: (Projected, None) / (Duration, Element, Explicit) /
TSQL2 QUERY:

```
SELECT SNAPSHOT E.Name
FROM Emp(ID, Name) AS E, Emp(ID, Salary)(PERIOD) AS E1
WHERE E.ID = E1.ID AND CAST(VALID(E1) AS INTERVAL YEAR) >= INTERVAL '3' YEAR
```

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

QUERY Q 1.4.3: Who worked for the same manager for at least five years?
EXPECTED ANSWER: "*ED*" and "*DI*"
CATEGORY: (Projected, None) / (Duration, Element, Explicit) /
TSQL2 QUERY:

```
SELECT SNAPSHOT E.Name
FROM Emp(ID, Name) AS E,
    (SELECT E1.ID, D.MgrID
    FROM Emp(ID, DeptName) AS E1, Dept(Name, MgrID) AS D
    WHERE E1.DeptName = D.Name)(PERIOD) AS DT
WHERE E.ID = DT.ID AND CAST(VALID(DT) AS INTERVAL YEAR) >= INTERVAL ' 5' YEAR
```

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

QUERY Q 1.4.4: Who worked in a department for less than 6 months total?
EXPECTED ANSWER: "None"
CATEGORY: (Projected, None) / (Duration, Element, Explicit) /
TSQL2 QUERY:

```
SELECT SNAPSHOT E1.Name
FROM Emp(ID, Name) AS E1, FROM Emp(ID, DeptName) AS E2
WHERE E1.ID = E2.ID AND CAST(VALID(E2) AS INTERVAL MONTH) < INTERVAL '6' MONTH
```

TSQL2 ANSWER: The empty table.

4.1.5 Class O1.S5 (Other, Event, Computed)

QUERY Q 1.5.1: Find *ED*'s skills when he joined the Book department
EXPECTED ANSWER: "typing," "filing" and "driving."
CATEGORY: (Projected, None) / (Containment, Event, Computed) / (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT S.Skill
FROM Emp(ID, Name, DeptName) AS E, Skills AS S
WHERE E.Name = 'Ed' AND E.DeptName = 'Book' AND E.ID = S.EmpID
    AND VALID(S) CONTAINS FIRST(VALID(E))
```

TSQL2 ANSWER: {(`typing'), (`filing'), (`driving')}

COMMENTS: We assume that "*ED*" really means "*Ed*" because we cannot directly compare the employee's surrogate field against "*ED*".

QUERY Q 1.5.2: Find the name and the budget of *ED*'s departments when he joined them.
EXPECTED ANSWER: "(Toy, $150K)" and "(Book, $50K)."
CATEGORY: (Projected, None) / (Containment, Event, Computed) / (=, Constant) (=, Foreign)
TSQL2 QUERY:

```
SELECT SNAPSHOT E.DeptName, D.Budget
FROM Emp(Name, DeptName) AS E, Dept(Name, Budget) AS D
WHERE E.Name = 'Ed' AND E.DeptName = D.Name AND VALID(D) CONTAINS FIRST(VALID(E))
```

TSQL2 ANSWER: {('Toy', $150K), ('Book', $50K)}
COMMENTS: We assume that "*ED*" really means "*Ed*".

QUERY Q 1.5.3: For each employee who was in the Toy department when it opened, find all data and skills that were valid at the time.
EXPECTED ANSWER: ''
CATEGORY: (Complete, None) / (Containment, Event, Computed) / (=, Constant) (=, Foreign)
TSQL2 QUERY:

```
SELECT SNAPSHOT E.*, S.*
FROM Emp AS E, Skills(EmpID, Skill) AS S, Dept(Name) AS D
WHERE S.EmpID = E.ID AND VAILD(E) CONTAINS FIRST(VALID(D))
    AND E.DeptName = 'Toy' AND D.Name = 'Toy'
```

TSQL2 ANSWER: {('Di', $30K, 'Toy', 'F', 10/1/1960, 'directing')}

QUERY Q 1.5.4: Find the names valid when the budget of the Toy department was decreased of the employees who had been working in the Toy department before the budget was decreased.
EXPECTED ANSWER: "Ed" and "Di"
CATEGORY: (Projected, None) / (Ordering, Event, Computed) (Ordering, Interval, Computed) / (=, Constant) (<>, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT E3.Name
FROM Emp(ID, Dept) AS E E2, Emp(ID,Name) AS E3, Dept(Name, Budget)(PERIOD) AS D1 D2
WHERE VALID(D1) MEETS VALID(D2) AND VALID(D1) PRECEDES VALID(D2)
    AND D1.Budget > D2.Budget AND E.DeptName = 'Toy'
    AND E2.Name = 'Toy' AND E1.ID = E2.ID AND E2.ID = E3.ID
    AND VALID(E) OVERLAPS VALID(D1) AND VALID(E) OVERLAPS VALID(D2)
```

TSQL2 ANSWER: {('Ed'), ('Di')}
COMMENTS: This assumes that integrity of budget data is maintained in the sense that there is always a prevailing Budget value. The `OVERLAPS` enforce the requirement for selected employees to be in the Toy department *before* the budget decrease.

QUERY Q 1.5.5: Find *ED*'s skills when his salary increased from $30K to $40K.
EXPECTED ANSWER: "typing," "filing" and "driving."

CATEGORY: (Projected, None) / (Containment, Event, Computed) (Ordering, Interval, Computed) / (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT Skill
FROM Skills, Emp(Salary,ID)(PERIOD) AS E1 E2,
WHERE E1 MEETS E2 AND E1 PRECEDES E2 AND E1.ID = E2.ID AND E1.ID = S.EmpID
    AND E1.Salary = 30 AND E2.Salary = 40
    AND VALID(S) CONTAINS BEGIN(VALID(E2))
```

TSQL2 ANSWER: {('typing'), ('filing'), ('driving')}
COMMENTS: The dual `CONTAINS` statements is another way to operationalize the requirement to identify skills when the salary increase occured, but not necessarilly *before* it.

4.1.6 Class O1.S6 (Other, Event, Other)

QUERY Q 1.6.1: Find the name, current budget, and manager of the Toy department.
EXPECTED ANSWER: "(Toy, *DI*, $100K)."
CATEGORY: (Complete, None) / (Containment, Event, Explicit) / (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT *
FROM Dept AS D
WHERE Name = 'Toy' AND VALID(D) CONTAINS CURRENT_DATE
```

TSQL2 ANSWER: {('Toy', 100000, 'Di')}

QUERY Q 1.6.2: Find the skills for which *ED* became qualified after 1/1/83.
EXPECTED ANSWER: "filing" and "driving."
CATEGORY: (Projected, None) / (Ordering, Event, Explicit) / (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT Skill
FROM Skills(PERIOD) AS S
WHERE S.EmpID = 'Ed' AND DATE '1/1/1983' PRECEDES VALID(S)
```

TSQL2 ANSWER: {('Filing'), ('Driving')}
COMMENTS: ED became qualified for driving twice. The two times are distinguished with a restructuring partition of `PERIOD`.

QUERY Q 1.6.3: Find *DI*'s salary on her 25^{th} birthday.
EXPECTED ANSWER: "$40K."
CATEGORY: (Projected, None) / (Containment, Event, User-defined) / (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT E2.Salary
FROM Emp AS E1 E2
WHERE E1.Name = 'Di' AND E1.ID = E2.ID
    AND VALID(E1) CONTAINS (E2.D-birth + INTERVAL '25' YEAR)
```

TSQL2 ANSWER: {(40)}

COMMENTS: No join is required as `D-birth` is a time-invariant attribute.

QUERY Q 1.6.4: Find the departments an employee named (sometime) Ed worked in before and not after 1/1/88.
EXPECTED ANSWER: "Toy."
CATEGORY: (Projected, None) / (Ordering, Event, Explicit) / (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT DeptName
FROM Emp(ID, DeptName) AS E1, Emp(ID,Name) AS E2
WHERE E1.ID = E2.ID AND E2.Name = 'Ed' AND VALID(E1) PRECEDES DATE '1/1/1988'
```

TSQL2 ANSWER: {('Toy')}

QUERY Q 1.6.5: Find the date of birth and current name of the women who were working in the Toy department on 1/1/83.
EXPECTED ANSWER: "(Di, 10/1/60)."
CATEGORY: (Projected, None) / (Containment, Event, Explicit) / (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT E2.Name, E1.D-birth
FROM Emp(ID, Gender, D-birth, DeptName) AS E1, E(Name)(PERIOD) AS E2
WHERE E1.Gender = 'F' AND E1.DeptName = 'Toy'
    AND VALID(E1) CONTAINS DATE '1/1/1983' AND E1.ID = E2.ID
    AND VALID(E2) CONTAINS CURRENT_DATE
```

TSQL2 ANSWER: {('Di', 10/1/1960))}
COMMENTS: `E1` and `E2` represent two "versions" of the same employee.

QUERY Q 1.6.6: Who worked in their current department for a longer time than their current manager worked in that department?
EXPECTED ANSWER: "None."
CATEGORY: (Projected, None) / (Containment, Event, Explicit) (Duration, Element, Computed) / (=, Foreign)
TSQL2 QUERY:

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

4.1.7 Class O1.S7 (Other, Interval, Computed)

QUERY Q 1.7.1: Find the names of all employees that changed departmenst while *DI* was working in a department called Toy.
EXPECTED ANSWER: "Ed" and "Edward."
CATEGORY: (Projected, None) / (Containment, Interval, Computed) (Duration, Element, Computed) / (=, Constant) (=, Constant)

TSQL2 QUERY:

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

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

QUERY Q 1.7.2: Which of all skills ever recorded did *ED* not acquire while working in the Book department.
EXPECTED ANSWER: "Driving," "Directing," "Filing," and "Typing."
CATEGORY: (Projected, None) / (Containment, Interval, Computed) / (<>, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT Skill
FROM Skills
MINUS
SELECT SNAPSHOT Skill
FROM Skills(EmpID, Skill) AS S1, Emp(ID, DeptName) AS E1 E2
WHERE E1.ID = E2.ID AND E2.Name = 'Ed' AND E1.DeptName = 'Book' AND E1.ID = S1.EmpID
    AND VALID(E1) OVERLAPS BEGIN(VALID(S1))
```

QUERY Q 1.7.3: Of the skills at some time possessed by *ED*, list those he did not acquire while he was working in the Book department.
EXPECTED ANSWER: "Driving," "Filing," and "Typing."
CATEGORY: (Projected, None) / (Containment, Interval, Explicit) (Containment, Interval, Explicit) / (<>, Constant)
TSQL2 QUERY:

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

QUERY Q 1.7.4: For any employee who reacquired a skill, find the name of the employee when a skill was reacquired.
EXPECTED ANSWER: "Ed."
CATEGORY: (Projected, None) / (Ordering, Interval, Computed) (Ordering, Interval, Computed) / (<>, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT Name
FROM Emp(ID, Name), Skills(EmpID, Skill) (PERIOD) AS S1 S2
WHERE S1.EmpID = S2.EmpID AND ID = S1.EmpID AND S1.Skill = S2.Skill
    AND VALID(S1) PRECEDES VALID(S2)
```

QUERY Q 1.7.5: Find the gender and name at the time of all employees that started working in some department before *ED* acquired the skill Driving for the second time.

EXPECTED ANSWER: "(Ed, M) and (Di, F)."
CATEGORY: (Projected, None) / (Ordering, Interval, Computed) (Ordering, Event, Computed) / (Sequence, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT E3.Name, E3.Gender
FROM Emp(ID, Name) AS E1, Emp(DeptName)(PERIOD) AS E2, E2(Name, Gender) AS E3
    Skills(EmpID, Skill) (PERIOD) AS S1 S2
WHERE E1.ID = 'ED' AND E1.ID = S1.EmpID AND
    S1.EmpID = S2.EmpID AND S1.Skill = S2.Skill AND VALID(S1) PRECEDES VALID(S2)
    AND NOT EXISTS ( SELECT SNAPSHOT S3.EmpID
                FROM Skills(EmpID, Skill) (PERIOD) AS S3
                WHERE S3.EmpID = S1.EmpID AND S3.Skill = S1.Skill
                    AND VALID(S3) PRECEDES VALID(S1) )
                    AND BEGIN(VALID(E2)) PRECEDES VALID(S2)
```

QUERY Q 1.7.6: Who worked in a department for a given manager for at least the period when that manager managed that department?
EXPECTED ANSWER: "*ED*" and "*DI*."
CATEGORY: (Projected, None) / (Containment, Interval, Computed) /
TSQL2 QUERY:

```
SELECT SNAPSHOT M1.ID
FROM ( SELECT E.ID, E.DeptName, D.MgrID FROM Emp AS E, Dept AS D
            WHERE E.DeptName = D.Name)(PERIOD) AS M,
       M(E.ID, E.DeptName)(PERIOD) AS M1, M(E.DeptName, D.MgrID)(PERIOD) AS M2
WHERE M1.ID = M2.ID AND VALID(M1) CONTAINS VALID(M2)
```

QUERY Q 1.7.7: What was the highest salary earned by *ED* before changing his name to Edward?
EXPECTED ANSWER: "40K."
CATEGORY: (Projected, None) / (Ordering, Interval, Computed) / (=, Constant) (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT MAX(E3.Salary)
FROM Emp(ID, Name) (PERIOD) AS E1 E2, E1(Salary) AS E3
WHERE E1.Name = 'Ed' AND E2.Name = 'Edward' AND E1.ID = E2.ID AND VALID(E1) MEETS VALID(E2)
```

4.1.8 Class O1.S8 (Other, Interval, Other)

QUERY Q 1.8.1: Find the name and skill pairs sometime possessed by people who worked in the Book or Toy department last year.
EXPECTED ANSWER: "(Edward, Typing), (Edward, Filing), (Edward, Driving), (Ed, Typing), (Ed, Filing), (Ed, Driving), and (Di, Directing)."
CATEGORY: (Complete, None) / (Containment, Interval, Explicit) / (=, Constant) (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT E1.Name, S.Skill
FROM Employee(Name, DeptName)(PERIOD) AS E1, Skills(PERIOD) AS S
WHERE E1.ID = S.EmpId
       AND E1.ID = ANY (SELECT SNAPSHOT E2.ID
```

```
                    FROM Employee(ID, DeptName) AS E2,
                    WHERE (E2.DeptName = 'Toy' OR E2.DeptName = 'Book') AND
                        VALID(E2) OVERLAPS PERIOD '[now-1 - now]' YEAR
```

TSQL2 Answer: {('Edward', 'Typing'), ('Edward', 'Filing'), ('Edward', 'Driving'), ('Ed', 'Typing'), ('Ed', 'Filing'), ('Ed', 'Driving'), ('Di', 'Directing')}
Comments: The nested SELECT computes the surrogates of all employess who worked in the Toy department the previous year. These surrogates are then matched to any skills ever possessed by that employee. An equivalent non-nested query is possible.

Query Q 1.8.2: Find the current name and skills of all people who worked for the Book or Toy department last year.
Expected Answer: "(Edward, Typing), (Edward, Filing), and (Di, Directing)."
Category: (Complete, None) / (Containment, Interval, Computed) (Containment, Interval, Explicit) / (=, Constant) (=, Constant)
TSQL2 Query:

```
SELECT SNAPSHOT E1.Name, S.Skill
FROM Employee(Name, DeptName)(PERIOD) AS E1, Skills(PERIOD) AS S
WHERE E1.ID = S1.EmpId AND E1 OVERLAPS CURRENT_DATE AND VALID(S) OVERLAPS CURRENT_DATE
       AND E1.ID = ANY (SELECT SNAPSHOT E2.ID
                    FROM Employee(ID, DeptName) AS E2,
                    WHERE (E2.DeptName = 'Toy' OR E2.DeptName = 'Book')
                        AND VALID(E2) OVERLAPS PERIOD '[now-1 - now]' YEAR
```

TSQL2 Answer: {('Edward', 'Typing'), ('Edward', 'Filing'), ('Di', 'Directing')}
Comments: The nested SELECT computes the surrogates of all employess who worked in the Toy department the previous year. The outer query matches these surrogates to the current tuples in the Employee and Skills tables for the same employees. An equivalent non-nested query is possible.

Query Q 1.8.3: Find their names (when they reported to *DI*) for all people who reported to *DI* before last year.
Expected Answer: "Ed" and "Di."
Category: (Complete, None) / (Ordering, Interval, Explicit) / (<, Constant)
TSQL2 Query:

```
SELECT SNAPSHOT E.Name
FROM Employee(Name, DeptName)(PERIOD) AS E, Dept(Name, MgrID)(PERIOD) AS D,
WHERE E.DeptName = D.Name AND D.MgrID = 'DI'
       AND VALID(E) OVERLAPS VALID(D) AND
       BEGIN(INTERSECT(VALID(E), VALID(D))) PRECEDES DATE 'now-1' YEAR
```

TSQL2 Answer: {('Ed'), ('Di')}

Query Q 1.8.4: Find the current manager of anyone who acquired a skill between 1983 and 1987 inclusive.
Expected Answer: "*ED*."
Category: (Projected, None) / (Containment, Interval, Computed) / (=, Constant)
TSQL2 Query:

```
SELECT SNAPSHOT E2.Name
```

```
FROM Employee(Name, DeptName)(PERIOD) AS E1, Dept(Name, MgrID)(PERIOD) AS D
WHERE E1.DeptName = D.Name AND VALID(E1) OVERLAPS CURRENT_DATE
       AND VALID(D) OVERLAPS CURRENT_DATE
       AND E1.ID = ANY (SELECT SNAPSHOT S.EmpID
                    FROM Skills(PERIOD) AS S,
                    WHERE BEGIN(VALID(S)) OVERLAPS PERIOD '[now-7 - now-2]' YEAR)
```

TSQL2 Answer: {('Ed')}
Comments: The nested SELECT computes the surrogates of all employess who acquired a skill during the specified period. The outer query matches these surrogates to the current employee tuple representing the manager of the employee's department. As surrogates can not be printed, the query returns the manager's name.

Query Q 1.8.5: Find the current name of anyone who lost a skill in the last four years.
Expected Answer: "Edward."
Category: (Projected, None) / (Containment, Interval, Explicit) / (=, Constant)
TSQL2 Query:

```
SELECT SNAPSHOT E.Name
FROM Employee(Name)(PERIOD) AS E
WHERE VALID(E) OVERLAPS CURRENT_DATE
       AND E.ID = ANY (SELECT SNAPSHOT S.EmpID
                    FROM Skills(PERIOD) AS S,
                    WHERE END(VALID(S)) OVERLAPS PERIOD '[now-4 - now]' YEAR
```

TSQL2 Answer: {('Edward')}
Comments: The nested SELECT computes the surrogates of all employess who lost a skill during the specified period. The outer query matches these surrogates to the current tuple for the employee in the Employee table, and projects out the Name attribute.

Query Q 1.8.6: Find the current name and department of anyone who changed their name or salary between July 1987 and June 1988 inclusive.
Expected Answer: "(Edward, Book)."
Category: (Projected, None) / (Containment, Interval, Computed) / (=, Constant)
TSQL2 Query:

```
SELECT SNAPSHOT E1.Name, E1.DeptName
FROM Employee(Name, DeptName)(PERIOD) AS E1
WHERE VALID(E1) OVERLAPS CURRENT_DATE
       AND E1.ID = ANY (SELECT SNAPSHOT E2.ID
                    FROM Employee(ID, Name) AS E3 E4,
                        Employee(ID, Salary) AS E5 E6
                    WHERE (E3.ID = E4.ID AND E3.Name <> E4.Name
                        AND VALID(E3) OVERLAPS PERIOD '[7/1/1987 - 6/1/1988]'
                        AND VALID(E4) OVERLAPS PERIOD '[7/1/1987 - 6/1/1988]')
                        OR (E5.ID = E6.ID AND E5.Name <> E6.Name
                            AND VALID(E5) OVERLAPS PERIOD '[7/1/1987 - 6/1/1988]'
                            AND VALID(E6) OVERLAPS PERIOD '[7/1/1987 - 6/1/1988]')
```

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

COMMENTS: The nested SELECT computes the surrogates of all employess who either changed their names (given by the first disjunct in the WHERE clause) or changed their salary (given by the second disjunct in the WHERE clause). These surrogates are then matched to the employee's current Employee tuple by the outer query.

QUERY Q 1.8.7: List the names, and current managers and budgets of all departments with budgets of less than 200K during any period between January 1, 1985 and December 31, 1989.
EXPECTED ANSWER: "(Toy, *DI*, 100K)" and "(Book, *ED*, 50K)."
CATEGORY: (Complete, None) / (Containment, Interval, Explicit) / (<>, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT D1.Name, D1.Budget, E.Name
FROM Dept(Name, MgrID)(PERIOD) AS D1, Employee(Name)(PERIOD) AS E
WHERE E.ID = D1.MgrID AND VALID(D) OVERLAPS CURRENT_DATE AND VALID(E) OVERLAPS CURRENT_DATE
    AND D1.Name = ANY (SELECT SNAPSHOT D2.Name
                       FROM Dept(Name, Budget) AS D2
                       WHERE D2.Budget < 200000
                           AND VALID(D2) OVERLAPS PERIOD '[1/1/1985 - 1/31/1989]'
```

TSQL2 ANSWER: {('Toy', 'Di', 100K), ('Book', 'Ed', 50K)}
COMMENTS: The nested SELECT computes the names of all departments satisfying the budget restriction during the given time period. The outer query matches these department names to the current department tuples, and uses the MgrID attribute of the matched tuples to find the name of the current department manager. The manager's name, rather than surrogate, is projected.

QUERY Q 1.8.8: Who worked in the Toy department at some point and earned at least 40K throughout the last two years?
EXPECTED ANSWER: "*ED*" and "*DI*"
CATEGORY: (Projected, None) / (Containment, Element, Explicit) / (=, Constant) (<>, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT E1.Name
FROM Employee(ID, DeptName)(PERIOD) AS E1, Employee(ID, Salary) AS E2
WHERE E1.DeptName = 'Toy' AND E2.Salary >= 40000 AND E1.ID = E2.ID
    AND SELECT SUM(VALID(E2)) CONTAINS PERIOD '[now-2 - now]' YEAR
```

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

4.1.9 Class O1.S9 (Other, Element, Computed)

QUERY Q 1.9.1: Find the names of departments that always had a budget greater than $90K during the times when managed by someone named Di.
EXPECTED ANSWER: "Toy."
CATEGORY: (Projected, None) / (Containment, Element, Computed) / (=, Constant) (=, Constant) (=, Foreign)
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)
```

QUERY Q 1.9.2: Find *ED*'s salaries when he worked in the same department as *DI*.
EXPECTED ANSWER: "$20K, $30K, $40K."
CATEGORY: (Projected, None) / (Containment, Element, Computed) / (=, Constant) (=, Single) (=, Constant)
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
```

QUERY Q 1.9.3: Find the names of the departments that *ED* worked in while earning $40K.
EXPECTED ANSWER: "Toy" and "Book."
CATEGORY: (Projected, None) / (Containment, Element, Computed) / (=, Constant) (=, Constant)
TSQL2 QUERY:

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

QUERY Q 1.9.4: Find *ED*'s names after he left the Toy department.
EXPECTED ANSWER: "Ed" and "Edward."
CATEGORY: (Projected, None) / (Ordering, Element, Computed) / (=, Constant) (=, Constant)
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)
```

QUERY Q 1.9.5: Find the skills that *ED* possessed sometime when he worked in the Toy department.
EXPECTED ANSWER: "Driving," "Filling," and "Typing."
CATEGORY: (Projected, None) / (Containment, Element, Computed) / (=, Constant) (=, Foreign) (=, Constant)
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
```

QUERY Q 1.9.6: What new skills did *ED* obtain after he had changed his name to Edward?
EXPECTED ANSWER: "None."
CATEGORY: (Projected, None) / (Ordering, Element, Computed) / (=, Constant) (=, Constant)
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)
```

QUERY Q 1.9.7: What were Toy's departmental budgets when it had a manager named Di?
EXPECTED ANSWER: "150K," "200K," and "100K."
CATEGORY: (Projected, None) / (Containment, Element, Computed) / (=, Constant) (=, Constant)
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: {}

4.1.10 Class O1.S10 (Other, Element, Other)

QUERY Q 1.10.1: Which managers managed which departments between January 1, 1982 and December 31, 1989?
EXPECTED ANSWER: "(DI, Toy)" and "(*ED*, Book)."
CATEGORY: (Projected, None) / (Containment, Element, Explicit) /
TSQL2 QUERY:

```
SELECT SNAPSHOT M.Name, D.Name
FROM Dept(Name, MgrID) AS D, Emp(ID, Name) As M
WHERE D.MgrID= M.ID AND VALID(D) CONTAINS PERIOD '[1/1/1982 - 12/31/1989]'
```

TSQL2 ANSWER: {(Di, Toy), (Ed, Book), (Edward, Book)}

4.2 Valid-time Output

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

4.2.1 Class O2.S1 (Duration, Interval, Computed)

QUERY Q 2.1.1: Find the times when persons with a shorter employment in the Toy department than *DI* were employed in the Book department.
EXPECTED ANSWER: "{ [4/1/87 - now]}"
CATEGORY: (None, Element, Union) / (Duration, Element, Computed) / (=, Constant)
TSQL2 QUERY:

```
SELECT VALID(E1)
FROM Emp (Id, DeptName) E1 E2 E3, E3 (Name) E4
```

```
WHERE E1.DeptName = 'Book' AND E1.Id = E2.Id
    AND E2.DeptName = 'Toy' AND E3.DeptName = 'Toy'
    AND CAST(VALID(E2) AS INTERVAL DAY) < CAST(VALID(E3) AS INTERVAL DAY)
    AND E4.Name = 'Di'
```

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

QUERY Q 2.1.2: Find the employment periods of persons that made 40K for a longer time than *DI* made 40K.
EXPECTED ANSWER: "{ [2/1/82 - 1/31/87], [4/1/87 - now]}"
CATEGORY: (None, Element, Union) / (Duration, Element, Computed) / (=, Constant)
TSQL2 QUERY:

```
SELECT VALID(E1)
FROM Emp (Id, Salary) E1 E2, E2 (Name) E3
WHERE E1.Salary = 40000 AND E2.Salary = 40000 AND E3.Name = 'Di'
    AND CAST(VALID(E1) AS INTERVAL DAY) > CAST(VALID(E2) AS INTERVAL DAY)
```

TSQL2 ANSWER: {([2/1/1982 - 1/31/1987]), ([4/1/1987 - now])}

QUERY Q 2.1.3: Find the starting times in the Book department of persons which possessed the Filing skill for a longer time than *DI*.
EXPECTED ANSWER: "{ 4/1/87}"
CATEGORY: (None, Element, Derived) / (Duration, Element, Computed) / (=, Constant)
TSQL2 QUERY:

```
SELECT BEGIN(E1)
FROM Emp (Id, DeptName) E1, Skill (EmpId, Skill) E2 E3, E3 (Name) E4
WHERE E1.DeptName = 'Book' AND E1.Id = E2.EmpId
    AND E2.Skill = 'Filing' AND E4.Name = 'Di' AND E3.Skill = 'Filing'
    AND CAST(VALID(E2) AS INTERVAL DAY) > CAST(VALID(E3) AS INTERVAL DAY)
```

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

QUERY Q 2.1.4: Return the times when persons employed a shorter time than *DI* acquired a skill.
EXPECTED ANSWER: "{ 1/1/82, 4/1/82, 6/1/84, 1/1/85, }"
CATEGORY: (None, Element, Derived) / (Duration, Element, Computed) / (=, Constant)
TSQL2 QUERY:

```
SELECT BEGIN(S4)
FROM Emp (Id) E1 E2, E2 (Name) E3, Skill (EmpId, Skill) S4
WHERE E1.Id = S4.EmpId AND E3.Name = 'Di'
    AND CAST(VALID(E1) AS INTERVAL DAY) < CAST(VALID(E2) AS INTERVAL DAY)
```

TSQL2 ANSWER: {(1/1/1982), (4/1/1982), (6/1/1984), (1/1/1985)}

QUERY Q 2.1.5: Find the employment periods of persons employed shorter time than *DI*.
EXPECTED ANSWER: "{ [2/1/82 - 1/31/87], [4/1/87 - now]}"
CATEGORY: (None, Element, Union) / (Duration, Element, Computed) /(=, Constant)

TSQL2 Query:

```
SELECT VALID(E1)
FROM Emp (Id) E1 E2, E2 (Name) E3
WHERE E3.Name = 'Di' AND CAST(VALID(E1) AS INTERVAL DAY) < CAST(VALID(E2) AS INTERVAL DAY)
```

TSQL2 Answer: {([2/1/1982 - 1/31/1987]), ([4/1/1987 - now]))}

Query Q 2.1.6: When did someone get a raise more quickly than *DI* got her first raise?
Expected Answer: "06/01/82" and "09/01/86."
Category: (None, Event, Derived) / (Duration, Interval, Computed) (Ordering, Interval, Computed) / (= ,Constant)
TSQL2 Query:

```
SELECT END(E1)
FROM Emp (Id, Salary) (PERIOD) AS E1 E3, E3 (Name) AS E4
WHERE EXISTS (SELECT E2.Id FROM Emp (Id, Salary) (PERIOD) AS E2
                WHERE E1.Id = E2.Id AND E1.Salary < E2.Salary
                AND END(VALID(E1)) = BEGIN(VALID(E2)))
    AND E4.Name = 'Di' AND CAST(VALID(E1) AS INTERVAL DAY) < CAST(VALID(E3) AS INTERVAL DAY)
    AND E3.Salary <= ALL (SELECT E5.Id FROM Emp (Id, Salary) (PERIOD) AS E5
                        WHERE E3.Id = E5.Id AND E3.Salary < E5.Salary)
```

TSQL2 Answer: {(06/01/1982), (09/01/1986)}

Query Q 2.1.7: What was the longest period when no one was hired or left employment?
Expected Answer: "02/01/82–01/31/87"
Category: (None, Interval, Union) / (Duration, Interval, Computed) /
TSQL2 Query:

```
CREATE VIEW VT (C1) AS
       SELECT BEGIN (VALID(E1)) FROM Emp (Id) (PERIOD) AS E1
       UNION
       SELECT END(VALID(E2)) FROM Emp (Id) (PERIOD) AS E2;
SELECT MAX(V2.C1 - V1.C1)
FROM VT V1 V2
WHERE V2.C1 in (SELECT MIN(C1) from VT WHERE VT.C1 > V1.C1)
```

TSQL2 Answer: {([02/01/1982 - 01/31/1987])}

Query Q 2.1.8: What was the longest period when no one received a raise?
Expected Answer: "09/01/86–now"
Category: (None, Interval, Union) / (Duration, Interval, Computed) /
TSQL2 Query:

```
CREATE VIEW VT (C1) AS
       SELECT END (E1) FROM Emp (Id, Salary) (PERIOD) AS E1
       WHERE EXISTS (SELECT E2.Id FROM Emp (Id, Salary) (PERIOD) AS E2
                WHERE E1.Id = E2.Id and E1.Salary < E2.Salary
                AND END(VALID(E1)) = BEGIN(VALID(E2)))
```

```
SELECT MAX(V2.C1 - V1.C1) FROM VT AS V1 V2
WHERE V2.C1 in (SELECT MAX(CAST(VALID(C1) AS INTERVAL DAY) FROM VT WHERE VT.C1 > V1.C1)
```

TSQL2 ANSWER: {([09/01/1986 - now])}

QUERY Q 2.1.9: When was the longest period when a department was without a manager?
EXPECTED ANSWER: "Never."
CATEGORY: (None, Interval, Union) / (Duration, Interval, Computed) /
TSQL2 QUERY:

```
SELECT MAX (CAST(VALID(NoMgr) AS INTERVAL DAY)
       FROM (SELECT * FROM Dept
             MINUS
             SELECT D.* FROM Dept (Name, MgrId) AS D) AS NoMgr
```

TSQL2 ANSWER: The empty table.

4.2.2 Class O2.S2 (Duration, Element, Other)

QUERY Q 2.2.1: Find employment periods in the Toy department for persons that have worked there for at least 8 years.
EXPECTED ANSWER: "{ [1/1/82 - now]}"
CATEGORY: (None, Element, Union) / (Duration, Element, Explicit) /(=, Constant)
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 Q 2.2.2: Find the starting times of managers which managed a department for at least 5 years.
EXPECTED ANSWER: "{ 1/1/82}"
CATEGORY: (None, Element, Derived) / (Duration, Element, Explicit) /(=, Constant)
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)}
COMMENTS: This answer assumes that the manager managed only one department for 5 years. Omitting the coalescing attribute `Name` gives the starting times of managers who managed any combination of departments for 5 years.

QUERY Q 2.2.3: Find the rehiring dates of employees with a gap in employment that exceeds 1 month.
EXPECTED ANSWER: "{ 4/1/87}"
CATEGORY: (None, Element, Union) / (Duration, Element, Explicit) /(=, Constant)

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 Q 2.2.4: Find the times when persons possessed skills that they lost and regained more than 1 year later.
Expected Answer: "{ [1/1/82 - 5/1/82], [6/1/84 - 5/31/88]}"
Category: (None, Element, Derived) / (Duration, Element, Explicit) /(=, Constant)
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 Q 2.2.5: Find budget periods that exceed 2 years.
Expected Answer: "{ [1/1/87 - now], [4/1/87 - now]}"
Category: (None, Element, Derived) / (Duration, Element, Explicit) /(=, Constant)
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 Q 2.2.6: When did no one's salary change for at least six months?
Expected Answer: "06/01/82–07/31/84," "08/01/84–01/31/85," "02/01/85–08/31/86," "09/01/86–now."
Category: (None, Interval, Derived / (Duration, Interval, Explicit) (temporal aggregate) /
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])}
Comments: Restructuring on Emp(ID, Salary) gives an answer based on individuals - "When did someone's salary not change for 6 months"

4.2.3 Class O2.S3 (Duration, Element, Computed)

QUERY Q 2.3.1: When did somebody have the same salary for the longest continuous period?
EXPECTED ANSWER: "09/01/86–01/01/90"
CATEGORY: (None, Element, Derived) / (Duration, Element, Computed) (temporal aggregate) /
TSQL2 QUERY: None.

QUERY Q 2.3.2: When did anybody work for a manager in a department for as long as that manager managed that department?
EXPECTED ANSWER: "1/1/82–1/1/90" and "4/1/87–1/1/90."
CATEGORY: (None, Element, Union) / (Duration, Element, Computed) /
TSQL2 QUERY: None.

QUERY Q 2.3.3: When did someone manage the Toy department for longer than *DI* did?
EXPECTED ANSWER: "No time."
CATEGORY: (None, Element, Union) / (Duration, Element, Computed) / (=, Constant) (=, Constant)
TSQL2 QUERY: None.

QUERY Q 2.3.4: When did anyone have a skill longer than *ED* had Driving?
EXPECTED ANSWER: "{ [1/1/82 - now], [4/1/82 - now], [1/1/85 - now] }"
CATEGORY: (None, Element, Union) / (Duration, Element, Computed) / (=, Constant) (=, Constant)
TSQL2 QUERY: None.

4.2.4 Class O2.S4 (Duration, Element, Other)

4.2.5 Class O2.S5 (Other, Event, Computed)

4.2.6 Class O2.S6 (Other, Event, Other)

QUERY Q 2.6.1: When did anybody have at least the skills that *DI* currently has?
EXPECTED ANSWER: "No time."
CATEGORY: (None, Element, Intersection) / (Containment, Event, Explicit) / (=, Constant)
TSQL2 QUERY:

```
SELECT CAST(VALID(s5) AS INTERVAL DAY)
FROM Skill(EmpId) s5,
    (SELECT SNAPSHOT s4.EmpId
    FROM Skill s4
    MINUS
    (SELECT EmpId
    FROM (SELECT DT1.EmpId, DT2.Skill
       FROM (SELECT SNAPSHOT s1.EmpId
            FROM Skill s1) AS DT1,
            (SELECT SNAPSHOT s2.Skill
            FROM Skill s2, EMPLOYEE(Id, Name) e
            WHERE e.Id = s2.Empid AND e.Name = 'Di' AND
                VALID(s2) CONTAINS CURRENT_DATE)) AS DT2)
    MINUS
```

```
      (SELECT SNAPSHOT s3.EmpId, s3.Skill
      FROM Skill AS s3)))(EmpID) AS dt
  WHERE s5.EmpId = dt.EmpId
```

TSQL2 Answer: {([1/1/82 - now])}
Comments: The table expression DT returns all employee surrogate that have at least the skills that Di currently has. Di should be in the answer set.

4.2.7 Class O2.S7 (Other, Interval, Computed)

Query Q 2.7.1: When did the Toy budget decrease?
Expected Answer: "1/1/87."
Category: (None, Event, Derived) / (Ordering, Interval, Computed) / (=, Constant) (<>, Single)
TSQL2 Query:

```
SELECT SNAPSHOT BEGIN(VALID(D2))
FROM Dept(Name) AS D, D(Budget)(PERIOD) AS D1 D2
WHERE D1 meets D2 AND D1.Budget > D2.Budget
```

TSQL2 Answer: {(1/1/87)}
Comments: D1 and D2 denote consecutive data versions of the same department.

Query Q 2.7.2: When did an employee change name?
Expected Answer: "1/1/88."
Category: (None, Event, Derived) / (Ordering, Interval, Computed) / (=, Single) (<>, Single)
TSQL2 Query:

```
SELECT SNAPSHOT BEGIN(VALID(N2))
FROM Emp(Id) AS E, E(Name)(PERIOD) AS N1 N2
WHERE VALID(N1) MEETS VALID(N2) AND N1.Name <> N2.Name
```

TSQL2 Answer: {(1/1/1988)}
Comments: The nested query retrieves the IDs of the managers during the name change.

Query Q 2.7.3: When did the salary of an employee increase while the employee was a manager?
Expected Answer: "8/1/84" and "9/1/86."
Category: (None, Event, Derived) / (Ordering, Interval, Computed) (Containment, Element, Computed) / (=, Foreign) (=, Const) (<>, Single)
TSQL2 Query:

```
SELECT SNAPSHOT BEGIN(VALID(Sal2))
FROM Emp(Id) AS Mgr, Mgr(Salary)(PERIOD) AS Sal1 Sal2
WHERE VALID(Sal1) MEETS VALID(Sal2) AND Sal1.Salary < Sal2.Salary
    AND Mgr.Id IN (SELECT EmpId FROM Dept
                   WHERE VALID(Dept) CONTAINS BEGIN(VALID(Sal2)) )
```

TSQL2 Answer: {(8/1/1984), (9/1/1986)}

Query Q 2.7.4: Find the periods during which *DI* earned $40K and was a manager of the Toy department.

EXPECTED ANSWER: "8/1/84 – 8/31/86."
CATEGORY: (None, Element, Intersection) / (Containment, Interval, Computed) / (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT INTERSECT(VALID(Emp), VALID(D))
FROM Emp, Dept(EmpId) AS D
WHERE Emp.Id = 'DI' AND Emp.Salary = 40 AND D.EmpId = 'DI'
```

TSQL2 ANSWER: {([8/1/84 - 8/31/84])}
COMMENTS: If DI's name, gender and D-birth had been changed within the retrieved period, more consecutive periods were returned in different tuples.

QUERY Q 2.7.5: Find the acquisition dates of the skills *ED* acquired before or during the year he joined the Toy department.
EXPECTED ANSWER: "1/1/82" and "4/1/82."
CATEGORY: (None, Event, Derived) / (Containment, Interval, Computed) / (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT BEGIN(VALID(S))
FROM Skills(PERIOD) AS S, Emp(Id, DeptName) AS E
WHERE E.Id = 'ED' AND E.DeptName = 'Toy' AND S.EmpId = 'ED'
    AND BEGIN(VALID(S)) <= CAST(CAST(BEGIN(E) AS INTERVAL YEAR)
       + INTERVAL '1' YEAR) AS INTERVAL DAY) - INTERVAL '1' DAY
```

TSQL2 ANSWER: {(1/1/1982), (4/1/1982)}

4.2.8 Class O2.S8 (Other, Interval, Other)

QUERY Q 2.8.1: Find the beginning of a continuous period in which *ED* was named Edward, in which he had a constant salary, and which includes the year 1989.
EXPECTED ANSWER: "1/1/88."
CATEGORY: (None, Event, Derived) / (Containment, Interval, Explicit) / (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT BEGIN(INTERSECT(E1, E2))
FROM Employee(ID, Name)(PERIOD) AS E1, Employee(ID, Salary)(PERIOD) AS E2
WHERE E1.Name = 'Edward' AND INTERSECT(VALID(E1), VALID(E2)) OVERLAP PERIOD '[1989]'
    AND E1.ID = E2.ID
```

TSQL2 ANSWER: {(1/1/1988)}

QUERY Q 2.8.2: Find the dates, before or after the years 1984 and 1985, when *ED* acquired a skill.
EXPECTED ANSWER: "1/1/82" and "4/1/82."
CATEGORY: (None, Event, Derived) / (Ordering, Interval, Explicit) / (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT BEGIN(VALID(S))
FROM Skills(PERIOD) AS S
WHERE S.EmpID = 'ED' AND (BEGIN(VALID(S)) PRECEDES DATE '1/1/84'
```

```
      OR DATE '12/31/1985' PRECEDES BEGIN(VALID(S)))
```

TSQL2 Answer: {(1/1/1982), 4/1/1982)}

Query Q 2.8.3: Find *ED*'s unemployment periods when he was not exactly 30 years old.
Expected Answer: "2/1/87 – 3/31/87."
Category: (None, Element, Union) / (Ordering, Interval, User-defined) (Containment, Interval, User-defined) / (=, Constant)
TSQL2 Query:

```
SELECT SNAPSHOT PERIOD(END(VALID(E1))+INTERVAL '1' DAY,
       BEGIN(VALID(E2))-INTERVAL '1' DAY) -
       PERIOD(E1.D-birth + INTERVAL '30' YEAR,
            E1.D-birth + INTERVAL '31' YEAR - INTERVAL '1' DAY))
FROM Employee(ID)(STATE) AS E1 E2, Employee(ID) AS E3
WHERE E1.ID = 'ED' AND E2.ID = E1.ID and E3.ID = E1.ID
    AND NOT (VALID(E3) CONTAINS PERIOD(END(VALID(E1)) + INTERVAL '1' DAY,
       BEGIN(VALID(E2)) - INTERVAL '1' DAY))
```

TSQL2 Answer: The empty table.
Comments: This query is incorrect. The WHERE clause correctly computes the periods when *ED* was unemployed. It is intended that the SELECT clause subtract the period when *ED* was thirty years old from each of the unemployment periods, producing the desired result. However, subtraction is not defined between periods. An alternative approach, having the WHERE clause return valid-time elements, and performing set difference in the SELECT clause, has the difficulty of not transforming the result into periods.

Query Q 2.8.4: Find the time periods when *DI* worked in the department in which she has been working during all of 1987.
Expected Answer: "1/1/82 – |now|."
Category: (None, Element, Union) / (Containment, Interval, Explicit) / (=, Constant)
TSQL2 Query:

```
SELECT SNAPSHOT PERIOD(BEGIN(VALID(E1)), END(VALID(E1)))
FROM Employee(Name, Dept)(PERIOD) AS E1 E2
WHERE E1.ID = 'DI' AND E2.ID = 'DI' AND E1.DeptName = E2.DeptName
    AND VALID(E2) CONTAINS PERIOD '[1987]'
```

TSQL2 Answer: {([1/1/1982 - 7/31/1984]), ([8/1/1984 - 8/31/1986]), ([9/1/1986 - now])}
Comments: The computed periods are not coalesced.

Query Q 2.8.5: Find all the dates, between 1/1/83 and 12/31/85, when the Toy department budget changed.
Expected Answer: "8/1/84."
Category: (None, Event, Derived) / (Containment, Interval, Explicit) (Ordering, Interval, Computed) / (=, Single) (<>, Single)
TSQL2 Query:

```
SELECT SNAPSHOT BEGIN(VALID(D2))
FROM Dept(Name, Budget) AS D1 D2
```

```
WHERE D1.Name = 'Toy' AND D2.Name = 'Toy' AND END(VALID(D1)) PRECEDES BEGIN(VALID(D2))
    AND PERIOD '[1/1/83, 12/31/1985]' CONTAINS PERIOD(END(VALID(D1)),BEGIN(VALID(D2)))
    AND D1.Budget <> D2.Budget
```

TSQL2 ANSWER: {(8/1/1984)}

4.2.9 Class O2.S9 (Other, Element, Computed)

QUERY Q 2.9.1: At what times did an employee simultaneously posses at least the same skills that *DI* possessed?
EXPECTED ANSWER: "No times."
CATEGORY: (None, Element, Intersection) / (Containment, Element, Computed) / (=, Constant)
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.
COMMENTS: This should be an extension of query Q 2.6.1, however, that query appears to be incorrect (it yields the wrong answer!?). The above query is wrong since it does not find the correct times. What is needed is some way to intersect the times of *DI*'s skills with those of the employee.

QUERY Q 2.9.2: When was the budget for Toy department more than 100K?
EXPECTED ANSWER: "01/01/82–12/31/86."
CATEGORY: (None, Element, Derived) / (Containment, Element, Computed) / (=, Constant) (<>, Constant)
TSQL2 QUERY:

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

TSQL2 ANSWER: {([01/01/82-12/31/86])}

QUERY Q 2.9.3: What was the last continuous period when *ED* was named Edward and had Driving, Filing and Typing as skills simultaneously?
EXPECTED ANSWER: "01/01/88–05/31/88."
CATEGORY: (None, Interval, Intersection) / (Containment, Event, Computed) (temporal aggregate) / (=, Constant) (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT MAX(INTERSECT(VALID(E),INTERSECT(VALID(SDrive),
            INTERSECT(VALID(SFile),VALID(SType)))))
FROM Emp(ID,Name) AS E (PERIOD), Skills(EmpID,Skill) AS SDrive SFile SType (PERIOD)
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: {([01/01/1988 - 05/31/1988])}
Comments: This query partitions into periods to find "continuous periods" and then uses the MAX aggregate to find the last such period.

Query Q 2.9.4: When did *DI* earn less than *ED*?
Expected Answer: "Never."
Category: (None, Element, Intersection) / (Duration, Element, Computed) / (=, Constant) (=, Constant)
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 Q 2.9.5: When did *ED* work in Toy department while the department was managed by *DI*?
Expected Answer: "02/01/82–01/31/87"
Category: (None, Element, Union) / (Duration, Element, Computed) / (=, Constant) / (=, Constant) (=, Constant)
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: {([02/01/82 - 01/31/87])}

Query Q 2.9.6: When did an employee currently named Edward have driving skills?
Expected Answer: "01/01/82–05/01/82" and "06/01/84–05/31/88."
Category: (None, Element, Derived) / (Containment, Element, Computed) / (=, Constant) (=, Constant)
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: {([01/01/1982 - 05/01/1982]), ([06/01/1984 - 05/31/1988])}

4.2.10 Class O2.S10 (Other, Element, Other)

4.3 Explicit-attribute and Valid-time Output

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

4.3.1 Class O3.S1 (Duration, Interval, Computed)

QUERY Q 3.1.1: Who, and when, were continuously employed in the Toy department shorter than *DI* was continuously employed in the Toy department?
EXPECTED ANSWER: "*ED* from 01-Feb-1982 to 31-Jan-1987."
CATEGORY: (Projected, Interval, Derived) / (Duration, Interval, Computed) / (=, Constant) (=, Constant) (=, Constant)
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]) }
COMMENTS: Variable `e3` is used for ranging over continuous periods where DI was in the Toy department. This variable groups tuples on department names (we are interested in when DI was in a specific department, namely Toy), on ID's (we are interested in a specific individual, namely DI), and on "period" because we are interested in continuous periods. We know that DI is currently named Di—this helps us establish the ID of DI. Next, variable `e3` is restricted to the Toy department.

Variable `e1` is used for ranging over continuous periods where individual employees were in the Toy department. This variable has the same declaration as has `e3`. It is restricted to the Toy department. If the duration of a timestamp for an `e1` tuple exceeds the duration for some `e3` tuple then all names for the `e1` tuple are output. This is accomplished using the nested tuple variable `e2`. The output is a valid-time table, and each name in the result has thus an associated time period.

QUERY Q 3.1.2: 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?
EXPECTED ANSWER: "*ED* from 01-Feb-1982 to 31-Jan-1987, born 1-Jul-1955 and Male."
CATEGORY: (Complete, Interval, Derived) / (Duration, Interval, Computed) / (=, Constant) (=, Constant) (=, Constant)
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]) }

COMMENTS: This query resembles the previous query. To accomodate the additional attributes in the output, variable `e2` is grouped on those additional attributes.

QUERY Q 3.1.3: Who were continuously employed in the Toy department shorter than *DI* was continuously employed in the Toy department, and when did this employment start?
EXPECTED ANSWER: "*ED*, on 01-Feb-1982."
CATEGORY: (Projected, Event, Derived) / (Duration, Interval, Computed) / (=, Constant) (=, Constant) (=, Constant)
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) }
COMMENTS: This query resembles the previous queries. The decision to return a snapshot table has been made because the natural-language query is interpreted to not request a valid time, but a user-defined time. To obtain an equivalent valid-time table the `BEGIN(e2)` is simply moved from the `SELECT` clause (`SNAPSHOT` is omitted) and made the body of a `VALID` clause.

QUERY Q 3.1.4: 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.
EXPECTED ANSWER: "Ed, 01-Jan-1984."
CATEGORY: (Projected, Event, Imposed) / (Duration, Interval, Computed) / (=, Constant) (=, Constant) (=, Constant)
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) }
COMMENTS: This query resembles the previous query.

4.3.2 Class O3.S2 (Duration, Interval, Other)

QUERY Q 3.2.1: When was the Toy department's budget constant and greater than $175K for more than one year, and what was the budget?

EXPECTED ANSWER: "$200K from 01-Aug-1984 to 31-Dec-1986."
CATEGORY: (Projected, Interval, Derived) / (Duration, Interval, Supplied) / ($\neq$, Constant) (=, Constant)
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]) }
COMMENTS: Clearly what is requested here is a valid-time table. In order to consider only continuous periods, grouping on `PERIOD` is used.

QUERY Q 3.2.2: When was the Toy department's budget constant and greater than $175K for more than one year, and who was the manager for that time?
EXPECTED ANSWER: "From 01-Aug-1984 to 31-Dec-1986, managed by *DI*."
CATEGORY: (Projected, Interval, Derived) / (Duration, Interval, Supplied) / ($\neq$, Constant) (=, Constant)
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]) }
COMMENTS: Variable `d1` is used for ranging over those Toy department tuples with a constant budget, over 175K and valid for more than one year. It may be that there are several managers associated with the Toy department for a period where the remains over 175K. Thus, variable `d2` is used for accessing each manager associated with Toy sometime during a qualifying period. The names of the manager are subsequently found in the `Employee` table, using variable `e1`.

QUERY Q 3.2.3: Who managed a department with a budget that exceeded $175K and then held constant for one year, and when did that occur?
EXPECTED ANSWER: "*DI*, from 01-Aug-1984 to 31-Dec-1986."
CATEGORY: (Projected, Interval, Intersection) / (Duration, Interval, Supplied) / (=, Constant) (=, Foreign)
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]) }
COMMENTS: This query resembles the previous query, the difference being that we are now not only interested in the Toy department, but in any department.

QUERY Q 3.2.4: What departments were in continuous operation (and when) longer than the duration between *ED*'s and *DI*'s birth dates?

EXPECTED ANSWER: "Toy, from 01-Jan-1982 to present."
CATEGORY: (Projected, Interval, Derived) / (Duration, Interval, Computed) / (=, Constant) (=, Constant)
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]) }
COMMENTS: The two subqueries in the `WHERE` clause return instant timestamps. We can make no assumption about the relative order of ED's and DI's birth days, and the `CAST` operator cannot be applied to two instant timestamps, but only to an interval. Thus, we must construct a period argument to the `INTERVAL` operator, as shown.

QUERY Q 3.2.5: Who worked in one department for at least two years continuously and what were the periods of employment in that department?
EXPECTED ANSWER: "{ (*ED*, [2/1/82 - 1/31/87]), (*ED*, [4/1/87 - 1/1/90]), (*DI*, [1/1/82 - 1/1/90]) }"
CATEGORY: (Projected, Element, Derived) / (Duration, Interval, Explicit) /
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])}
COMMENTS: Tuple variable `e1` is used for ranging over tuples that represent continuous periods longer than two years where the same person works in the same department. ED works in Toy and Book, and DI works in Toy. Tuple variable `e2` ranges over tuples in `Emp` and is used for finding the names and employment periods of persons (i.e., ED and DI) in departments where those persons were employed continuously for more than two years. The employment periods are all times when the qualifying persons were in the qualifying departments.

4.3.3 Class O3.S3 (Duration, Element, Computed)

QUERY Q 3.3.1: Who, when, and for which department did anybody work for as long as the length of time that department's budget was below 200K?
EXPECTED ANSWER: "(*DI*, Toy, 01/01/82–01/01/90)" and "(*ED*, Book, 4/1/87–01/01/90)"
CATEGORY: (Projected, Element, Intersection) / (Duration, Element, Computed) / (<>, Constant)
TSQL2 QUERY:

```
SELECT e1.Name, e1.DeptName
FROM EMP(Name,DeptName) AS e1, DEPT(Name,Budget) AS d1
WHERE d1.Name=e1.DeptName AND d1.Budget<200K
```

```
    AND CAST(VALID(e1) AS INTERVAL DAY)>=CAST(VALID(d1) AS INTERVAL DAY)
```

TSQL2 Answer: {('Di', 'Toy' | [1/1/1982-1/1/1990]), ('Ed', 'Book' | [4/1/1987-1/1/1990])}

Comments: It is assumed that "for as long" in the natural-language formulation of the query is intended to mean "for at least as long" (as opposed to "for exactly as long"). Also, not that the query does not consider whether the peiod that the employee worked in the department contained the period that budget was below 200K. Although one might not expect this, it is possible that these periods are in fact disjoint!

Query Q 3.3.2: Who and when did anybody work in a department longer than their current manager worked in that department?

Expected Answer: "None"

Category: (Projected, Element, Derived) / (Duration, Element, Computed) / (Containment, Event, Explicit) /

TSQL2 Query:

```
SELECT e4.Name
FROM Emp AS e1, Dept AS d1
    Emp(ID, DeptName) AS e2, Emp(ID, DeptName) AS e3, e3(Name) AS e4,
WHERE VALID(e1) OVERLAPS CURRENT_DATE ANDVALID( d)1 OVERLAPS CURRENT_DATE
    AND e1.DeptName = d1.Name AND d1.MgrID = e2.ID AND e3.ID = e1.ID
    AND e2.DeptName = e3.DeptName
    AND CAST(VALID(e3) AS INTERVAL DAY) > CAST(VALID(e2) AS INTERVAL DAY)
```

TSQL2 Answer: `Empty result`

Comments: In this query, we first find a currently valid employee tuple (using `e1`). We will proceed to check whether the employee represented by this tuple should be in the result. To do so, we locate the current manager of the employee. This is done using `d1`. Tuple variable `e2` then contains a tuple, grouped on `ID` and `DeptName`, for the manager. A new tuple variable, `e3`, is created for the employee in question. The durations of `e2` and `e3` are now compared when `e2` and `e3` are restricted to the same department. Employee names are returned using e4 for each qualifying employee.

Query Q 3.3.3: For all employees who managed any departments at least as long as *DI* managed the Toy department, list their names, their gender, their departments and their salary histories during that time.

Expected Answer: "(*DI*, (Di, 01/01/82–01/01/90), F, (Toy, 01/01/82–01/01/90), ((30K, 01/01/82–07/31/84), (40K, 08/01/84–08/31/86), (50K, 09/01/86–01/01/90)))"

Category: (Projected, Element, Derived) / (Duration, Element, Computed) / (=, Constant) (=, Constant)

TSQL2 Query:

```
SELECT e1.Name, e1.Gender, e1.DeptName, e1.Salary
FROM Emp AS e1, Dept(Name, MgrID) AS d1
WHERE e1.ID = d1.MgrID AND CAST(VALID(d1) AS INTERVAL DAY) >=
    ( SELECT SNAPSHOT CAST(VALID(d2) AS INTERVAL DAY)
    FROM d2 AS Dept(Name, MgrID)
    WHERE d2.Name = 'Toy' AND d2.MgrID = (SELECT ID
       FROM Emp
       WHERE Emp.Name = 'Di' and VALID(Emp) OVERLAPS CURRENT_DATE))
```

TSQL2 Answer: {('Di', 'F', 'Toy', 30 | [1/1/1982 - 7/1/1984]), ('Di', 'F', 'Toy', 40 | [8/1/1984 - 8/31/1986]), ('Di', 'F', 'Toy', 50 | [9/1/1986 - 1/1/1990])}

QUERY Q 3.3.4: For departments that had a manager that served for the shortest total time, list the department name, the shortest-serving manager(s), and the times when those managers served the department.
EXPECTED ANSWER: "Book, *ED*, 04/01/87–01/01/90"
CATEGORY: (Projected, Element, Intersection) / (Duration, Element, Computed) /
TSQL2 QUERY:

```
SELECT d1.name, e1.name
FROM DEPT(Name,MgrID) AS d1, EMP AS e1
WHERE e1.ID = d1.MgrID AND
CAST(VALID(d1) AS INTERVAL DAY) = (SELECT MIN(CAST(VALID(d3) AS INTERVAL DAY))
    FROM DEPT(MgrID) AS d3)
```

TSQL2 ANSWER: {('Book', 'Ed', [4/1/1987 - 12/31/1987]) and ('Book', 'Edward', [1/1/1988 - 1/1/90])}
COMMENTS: The inner-most query returns the minimum duration of service for any manager, here [4/1/198 -, 1/1/1990].

QUERY Q 3.3.5: For all departments which had budgets of at least 200K for a longer total time than budgets of less than 200K, list their names, budgets, and times when the budgets were not below 200K.
EXPECTED ANSWER: "None."
CATEGORY: (Projected, Element, Derived) / (Duration, Element, Computed) / (<>, Constant) (<>, Constant)
TSQL2 QUERY:

```
SELECT d1.Name, d1.Budget
FROM Dept(Name, Budget) AS d1
WHERE d1.Budget >= 200000 AND
       ( SELECT SUM ( CAST(VALID(d2) AS INTERVAL DAY))
       FROM Dept(Name, Budget) AS d2
       WHERE d2.Budget >= 200000 AND d2.Name = d1.Name )
    >
       ( SELECT SUM ( CAST(VALID(d3) AS INTERVAL DAY))
       FROM Dept(Name, Budget) AS d3
       WHERE d3.Budget < 200000 AND d3.Name = d1.Name )
```

TSQL2 ANSWER: The empty table.
COMMENTS: Where are the times when the budgets were not below 200K?

QUERY Q 3.3.6: What skills did *ED* hold for at least as long as the total time during his employment that he did not have the Driving skill, and when did he have those skills?
EXPECTED ANSWER: "Typing, 04/01/82–01/01/90," "Filing, 01/01/85–01/01/90," and "Driving, (01/01/82–05/01/82, 06/01/84–05/31/88)."
CATEGORY: (Projected, Element, Derived) / (Duration, Element, Computed) / (=, Constant) (<>, Constant)
TSQL2 QUERY:

```
SELECT s1.Name
FROM Skill AS s1, Emp AS e1
WHERE s1.EmpID = e1.ID AND e1.Name = 'Edward' AND
```

```
CAST(VALID(s1) AS INTERVAL DAY) >= ( SELECT SNAPSHOT (VALID(e2) - VALID(s2)) DAY)
   FROM Emp(ID) AS e2, Skill AS s2
   WHERE e2.ID = e1.ID AND s2.Name = 'Driving' AND s2.EmpID = e1.ID )
```

TSQL2 ANSWER: {('Typing' | [4/1/1982 - 1/1/1990]), ('Filing' | [1/1/1985 - 1/1/1990]), ('Driving' | [1/1/1982 - 5/1/1982] ∪ [6/1/1984 - 5/31/1988])}

4.3.4 Class O3.S4 (Duration, Element, Other)

QUERY Q 3.4.1: Find the oldest employee who was a Typist on 12/31/85, and the times when that employee had been a Typist so far.
EXPECTED ANSWER: "(*ED*, 4/1/82 – 12/31/85)"
CATEGORY: (Projected, Element, Duration) / (Duration, Element, Explicit) / (=, Constant), (=, Foreign)
TSQL2 QUERY:

```
SELECT e1.Name
VALID s1 INTERSECT PERIOD '[beginning - 12/31/1985]')
FROM EMP AS e1
WHERE e1.D-Birth = (
    SELECT MIN(e2.D-Birth)
    FROM SKILLS AS s1, EMP AS e2
    WHERE VALID(s1) OVERLAPS DATE '12/31/1985' AND
       s1.Skill = 'Typing' AND s1.EmpID = e2.ID
    ) AND VALID(e1) INTERSECT VALID(s1)
```

TSQL2 ANSWER: {('Ed' | [4/1/1982 - 12/31/1985]), ('Edward' | [4/1/1982 - 12/31/1985])}
COMMENTS: This formulation of the query is not correct. First of all, I am not sure that it is valid to refer to the variable s1 in the outermost query, since it is only defined in the nested query. However, it is necessary to link the valid-times of these two tuple, both to eliminate the possibility of retrieving ('Edward' — [4/1/1982, 12/31/85]) as another answer to the query, and also to restrict the valid-time of the result (derived from an EMP tuple) based upon the valid time of the Skills tuple.

QUERY Q 3.4.2: For employees in the Toy department who had worked less than *DI* in that department as of 1/1/85, find their names on 1/1/85 and the time difference on 1/1/85.
EXPECTED ANSWER: "(Ed, 1 month less)"
CATEGORY: (Projected, —) / (Duration, Element, Explicit) / (=, Constant), (<>, Single)
TSQL2 QUERY:

```
SELECT e1.Name, e3.Name
FROM (SELECT e1.Name VALID INTERSECT PERIOD '[beginning - 1/1/1985]'
            FROM EMP AS e1 WHERE e1.Dept="Toy") AS e1 e2,
    EMP AS e3
WHERE e2.Name = 'Di' AND e1.ID = e3.ID
AND CAST(VALID(e1) AS INTERVAL DAY) < CAST(VALID(e2) AS INTERVAL DAY)
AND VALID(e3) CONTAINS DATE '1/1/1985'
```

TSQL2 ANSWER: ('Ed', -26 MONTH

QUERY Q 3.4.3: Find the current employees who worked the least during 1987, and the times in 1987 during which they worked.

EXPECTED ANSWER: "(*ED*, ((1/1/87 – 1/31/87), (4/1/87 – 12/31/87)))"
CATEGORY: (Projected, Element, Intersection) / (Duration, Element, Explicit) / (Nothing)
TSQL2 QUERY:

```
SELECT e.Name
FROM EMP AS e, (SELECT Name VALID INTERSECT PERIOD '[1987]' FROM EMP) AS e2
WHERE VALID(e) CONTAINS CURRENT_DATE AND e.ID = e2.ID
AND CAST(VALID(e2) AS INTERVAL DAY) = (SELECT MIN (CAST(VALID(e3) AS INTERVAL DAY))
    FROM (SELECT Name VALID INTERSECT PERIOD '[1987]' FROM EMP) AS e3)
```

TSQL2 ANSWER: {('Ed' | [1/1/1987 - 1/31/1987]), ('Ed' | [4/1/1987 - 12/31/1987]) }

4.3.5 Class O3.S5 (Other, Event, Computed)

QUERY Q 3.5.1: List the names and ages of all employees at the time they received their first salary increment.
EXPECTED ANSWER: "(Ed, 26 years & 11 months), (Di, 23 years & 10 months)"
CATEGORY: (Projected, —) / (Containment, Event, Computed) /
TSQL2 QUERY:

```
SELECT a.Name, MIN(a.Age)
FROM (SELECT e1.Name, BEGIN(e2)-E1.DBirth AS Age
VALID BEGIN(e2)
FROM EMP e1, EMP e2
WHERE e1.Id=e2.Id AND e1.Salary <> e2.Salary AND VALID(e1) MEETS VALID(e2)) AS a
GROUP BY a.Name
```

COMMENTS: A changes operator and temporal ordering would be appropriate for formulating this query.

QUERY Q 3.5.2: List the name and salary histories up until their 25th birthday of all female employees.
EXPECTED ANSWER: "((Di, (1/1/82–10/1/85), ((30K, 01/01/82–07/31/84), (40K, 08/01/84–10/01/85)))"
CATEGORY: (Projected, Element, Derived) / (Containment, Event, Computed) (Duration, Interval, explicit) / (=, Constant)
TSQL2 QUERY:

```
SELECT Emp.Name, Emp.Salary
VALID INTERSECT PERIOD(DATE 'BEGINNING', Emp.DBirth + INTERVAL '25' YEAR)
FROM Emp
WHERE Emp.Gender = 'F' AND
    (Emp PRECEDES (Emp.DBirth + INTERVAL '25' YEAR)
       OR VALID (Emp) CONTAINS (Emp.DBirth + INTERVAL '25' YEAR))
```

TSQL2 ANSWER: {('Di', 30000 | [1/1/1982-7/31/1984]), ('Di', 40000 | [8/1/1984-10/1/1985])}

QUERY Q 3.5.3: When and who ever changed their names?
EXPECTED ANSWER: "(*ED*, 01/01/88)"
CATEGORY: (Projected, Event, Derived) / (Containment, Event, Computed) /
TSQL2 QUERY:

```
SELECT E1.Name
```

```
VALID BEGIN(E2)
FROM Emp E1, Emp E2
WHERE E1.Id = E2.Id AND E1.Name <> E2.Name AND E1 MEETS E2
```

TSQL2 Answer: {('Ed' | 1/1/1988)}
Comments: Unfortunately no special query construct for changes.

Query Q 3.5.4: How old was *ED* and what skills did he have at the time he changed his name to Edward?
Expected Answer: "(32 years & 6 months, {Driving, Filing, Typing})"
Category: (Projected, —) / (Containment, Event, Computed) / (=, Constant) (=, Constant)
TSQL2 Query:

```
SELECT SNAPSHOT BEGIN(Emp2)-Emp1.DBirth AS Age, Skills.Name
FROM Emp Emp1, Emp Emp2, Skills
WHERE Emp1.Name = 'Ed' AND Emp2.Name = 'Edward'
    AND Emp1.Id = Emp2.Id AND VALID(Emp1) MEETS VALID(Emp2)
       AND Emp2.Id = Skills.Id AND VALID(Skills) CONTAINS BEGIN(VALID(Emp2))
```

TSQL2 Answer: {('32.5' YEAR, 'Driving'), ('32.5' YEAR, 'Filing'), ('32.5' YEAR, 'Typing')}

Query Q 3.5.5: When did *ED* acquire the Driving skill, and what other skills did he have at the time?
Expected Answer: "(01/01/82, None), (06/01/84, Typing)"
Category: (Projected, Event, Intersection) / (Containment, Event, Computed) / (=, Constant) (=, Constant)
TSQL2 Query:

```
SELECT S1.Name
VALID BEGIN(S2)
FROM Emp, Skills S1, Skills S2
WHERE Emp.Name = 'ED' AND Emp.Id = S1.ID
    S1.Id = S2.Id AND S2.Name = 'Driving' AND VALID(Emp) OVERLAPS S1
       AND VALID(S1) CONTAINS BEGIN(VALID(S2))
```

TSQL2 Answer: {('Typing' | 6/1/1984)}
Comments: This result is incorrect, as it doesn't include None. Instead, a one-sided outer join should be used.

Query Q 3.5.6: Who was the first female manager of the Toy department, and when did she become manager of that department for the first time?
Expected Answer: "(*DI*, 01/01/82)"
Category: (Projected, Event, Derived) / (Containment, Event, Computed) (temporal aggregate) / (=, Constant) (=, Constant)
TSQL2 Query:

```
SELECT BEGIN(f), f.Name
FROM (SELECT * FROM EMP AS e1 WHERE e1.Gender=F AND e1.DeptName = "Toy") AS f
WHERE BEGIN(VALID(f)) = (SELECT MIN(BEGIN t) FROM FM AS t)
```

4.3.6 Class O3.S6 (Other, Event, Other)

QUERY Q 3.6.1: Find the name and salary histories of employees whose date-of-birth was after 1/1/56.
EXPECTED ANSWER: "((Di, (1/1/82–1/1/90), ((30K, 01/01/82–07/31/84), (40K, 08/01/84–08/31/86), (50K, 09/01/86–01/01/90)))"
CATEGORY: (Projected, Element, Derived) / (Ordering, Event, User-defined) /
TSQL2 QUERY:

```
SELECT e1.Name, e1.Salary
FROM EMPLOYEE AS e1
WHERE e1.D-Birth > DATE '1/1/1956'
```

TSQL2 ANSWER: {{ ('Di', 30 | [1/1/1982 - 7/31/1984]),('Di', 40 | [8/1/1984 - 8/31/1986]), ('Di', 50 | [9/1/1986 - 1/1/1990])}
COMMENTS: The "expected answer" shows this information grouped in a more compact way, since the single name has a single valid-time lifespan, whereas there are three different salaries, each with its own valid-time lifespan. I do not know how or if it is possible to get the answer in this representation via TSQL2.

QUERY Q 3.6.2: Find the name and salary histories of employees who were called "Ed" after 1/1/88.
EXPECTED ANSWER: "Empty Answer."
CATEGORY: (Projected, Element, Derived) / (Ordering, Event, User-Defined) /
TSQL2 QUERY:

```
SELECT e1.Name, e1.Salary
FROM EMPLOYEE AS e1
WHERE DATE '1/1/1988' PRECEDES VALID(e1) AND e1.Name = 'Ed'
```

TSQL2 ANSWER: The empty table, because no one was called "Ed" after this date.

QUERY Q 3.6.3: Find the name and salary histories since their latest pay raise of employees whose latest pay raise was in 1985.
EXPECTED ANSWER: "(((Ed, (2/1/85 – 12/31/87)), (Edward, (1/1/88 – 1/1/90))), (40K, ((2/1/85 – 1/31/87), (4/1/87 – 1/1/90))))"
CATEGORY: (Projected, Element, Derived) / (Containment, Event, Explicit) /
TSQL2 QUERY:

It is not clear how to do this.

COMMENTS: This should be doable using the following steps: (1) Compute payraises (can be done using "meet") (2) Computing most recent payraise and seeing if this is in 85.

QUERY Q 3.6.4: Find the name and salary histories of employees whose latest pay raise occurred after the date-of-birth of every other employee.
EXPECTED ANSWER: The answer contains Both *DI*'s and *ED*'s entire Name and Salary history, see Section 3.
CATEGORY: (Projected, Element, Derived) / (Ordering, Event, User-Defined) /
TSQL2 QUERY:

It is not clear how to do this.

COMMENTS: (1) For each employee, compute the min date of birth of all other employees. `select e1.Name, D`

```
from emp e1
where D = (select min(d-Birth) from emp e2 where e2.ID <> e1.ID)
```
(2) As for previous query, compute most recent payraise and check that this is after the employee's D-value computed in the above step.

QUERY Q 3.6.5: Find the name and salary histories of employees whose latest pay raise occurred on the date-of-birth of some other employee.
EXPECTED ANSWER: "Empty answer."
CATEGORY: (Projected, Element, Derived) / (Containment, Event, User-Defined) /
TSQL2 QUERY:

It is not clear how to do this.

QUERY Q 3.6.6: Who and when had at least the skills that *DI* currently has?
EXPECTED ANSWER: "(*DI*, [1/1/82–now])"
CATEGORY: (Projected, Element, Derived) / (Containment, Event, Explicit) / (=, Constant)
TSQL2 QUERY:

```
SELECT e.Name
FROM EMP AS e
WHERE ( SELECT s.Name FROM SKILLS s WHERE e.ID = s.EmpID ) CONTAINS
( SELECT INTERSECT(?, CURRENT_DATE) FROM SKILLS AS s WHERE s.EmpID = DI )
```

COMMENTS: Remove DI.

4.3.7 Class O3.S7 (Other, Interval, Computed)

QUERY Q 3.7.1: For the time before *ED* first worked in the Toy department, find the salary paid, and the time it was paid, of any employee who was in the Toy department at a time before Ed worked there.
EXPECTED ANSWER: "($30K, 1/1/82 – 1/31/82)"
CATEGORY: (Projected, Not Empty) / (Ordering, Interval, Computed) / (<>, Constant)
TSQL2 QUERY:

```
SELECT Emp1.Salary
VALID E1 - PERIOD(BEGIN(E2), DATE 'forever')
FROM AS Emp Emp1, Emp(Name, Department) AS Emp2
WHERE Emp2.Name = 'ED' AND Emp2.Dept = 'Toy' AND
    Emp1.Dept = 'Toy' AND BEGIN(VALID(Emp1)) PRECEDES BEGIN(VALID((Emp2))
```

TSQL2 ANSWER: {(30000 | [1/1/1982 - 1/31/1982])}

QUERY Q 3.7.2: Find the greatest salary under $50K paid to *ED* and the times during which it was paid.
EXPECTED ANSWER: "($40K, 2/1/85 – 1/31/87)," "($40k, 4/1/87 – present)"
CATEGORY: (Projected, Not Empty) / (Containment, Interval, Computed) / (=, Constant)
TSQL2 QUERY:

```
SELECT MAX(Emp1.Salary)
```

```
FROM Emp AS Emp1, Emp1 Emp2
WHERE Emp2.Name = 'ED' AND Emp1.Id = Emp2.Id
GROUP BY Emp1.Id
HAVING MAX(Emp1.Salary) < 50000
```

TSQL2 ANSWER: {(40000 | [2/1/1985 - 1/31/1987]) and (40000 | [4/1/1987 - 1/1/1990])}
COMMENTS: An aggregate is required here.

QUERY Q 3.7.3: Find *ED*'s salary history.
EXPECTED ANSWER: "($20K, 2/1/82 – 5/31/82)," "($30K, 6/1/82 – 1/31/85)," and "($40K, (2/1/85 – 1/31/87), (4/1/87 – present))."
CATEGORY: (Projected, Not Empty) / (Containment, Interval, Computed) / (=, Constant)
TSQL2 QUERY:

```
SELECT E1.Salary
FROM Emp AS E1, Emp AS E2
WHERE E2.Name = 'ED' AND E1.Id = E2.Id AND VALID(E1) OVERLAPS VALID(E2)
```

TSQL2 ANSWER: {(20000 | [2/1/1982 - 5/31/1982]), (30000 | [6/1/1982 - 1/31/1985]), (40000 | [2/1/1985 - 1/31/1987] ∪ [4/1/1987 - 1/1/1990])}

QUERY Q 3.7.4: For employees that were drivers and simultaneously made less than $40K, find the names, salaries, and times during which this occurred.
EXPECTED ANSWER: "(Ed, $20K, 2/1/82 – 5/1/82)" and "(Ed, $30K, 6/1/84 – 1/31/85)."
CATEGORY: (Projected, Not Empty) / (Containment, Interval, Computed) / (<>, Constant)
TSQL2 QUERY:

```
SELECT Emp.Name, Emp.Salary
FROM Emp, Skills
WHERE Emp.Id = Skills.Id AND Skills.Name = 'Driving' AND
    Emp.Salary < 40000 AND VALID(Emp) OVERLAPS VALID(Skills)
```

TSQL2 ANSWER: {('Ed', 20000 | [2/1/1982 - 5/1/1982]), ('Ed', 30000 | [6/1/1984 - 1/31/1985])}

QUERY Q 3.7.5: For the Toy department when *ED* worked there, find the budgets and associated times.
EXPECTED ANSWER: "(2/1/82 – 7/31/84, $150K)," "(8/1/84 – 12/31/86, $200K)," and "(1/1/87 – 1/31/87, $100K)"
CATEGORY: (Projected, Not Empty) / (Containment, Interval, Computed) / (=, Constant), (=, Foreign
TSQL2 QUERY:

```
SELECT Dept.Budget
FROM Emp, Dept
WHERE Dept.Name = 'Toy' AND Dept.Name = Emp.Department
    AND Emp.Name = 'ED' AND VALID(Emp) OVERLAPS VALID(Dept)
```

TSQL2 ANSWER: {(150000 | [2/1/1982 - 7/31/1984]), (200000 | [8/1/1984 - 12/31/1986]), (100000 | [1/1/1987 - 1/31/1987])}

4.3.8 Class O3.S8 (Other, Interval, Other)

QUERY Q 3.8.1: For all employees that have been in the Book or Toy departments sometime during the last two years, find their current names and their skills together with the times when they were valid.
EXPECTED ANSWER: "(Edward, Typing, 1-Apr-1982 to date), (Edward, Filing, 1-Jan-1985 to date), (Edward, Driving, 1-Jan-1982 to 1-May-1982), (Edward, Driving, 1-Jun-1984 to 31-May-1988) and (Di, Directing, 1-Jan-1982 to date)."
CATEGORY: (Projected, Element) / (Containment, Interval, Explicit) / (=, Constant) (=, Constant)
TSQL2 QUERY:

```
SELECT E2.Name, S.Name
FROM EMP(ID, DeptName) As E1, EMP(ID, Name) As E2, SKILL(EmpID, Name) As S
WHERE E1.ID = S.EmpID AND E1.ID = E2.ID
    AND VALID(E1) OVERLAPS PERIOD '[now-2 - now]' YEAR)
    AND VALID(E2) CONTAINS CURRENT_DATE
```

TSQL2 ANSWER: {('Edward', 'Typing' | [4/1/1982 - 1/1/1990]), ('Edward', 'Filing' | [1/1/1985 - 1/1/1990]'), ('Edward', 'Driving' | [1/1/1982 - 5/1/1982]), ('Edward', 'Driving', [6/1/1984 - 5/31/1988]), ('Di', 'Directing', [1/1/1982 - 1/1/1990])}

QUERY Q 3.8.2: Find the current names of all people who reported to *DI* before last year along with the time when they reported to her.
EXPECTED ANSWER: "(Edward, 1-Feb-1982 to 31-Jan-1987)"
CATEGORY: (Projected, None) / (Containment, Interval, Computed) / (<, Constant)
TSQL2 QUERY:

```
SELECT E2.Name
FROM EMP(ID, DeptName) As E1, EMP(ID, Name) As E2, EMP(ID, Name) As E3, DEPT(Name, MgrID) As D
WHERE E1.ID = E2.ID ANDVALID (E2) CONTAINS CURRENT_DATE AND E1.DeptName = D.Name
AND E1.ID <> D.MgrID AND E3.ID = D.MgrID AND E3.Name = 'Di'
AND VALID(E1) PRECEDES PERIOD '[now - 1 year - now]'
```

TSQL2 ANSWER: {('Edward' | [2/1/198 - 1/31/1987])}

QUERY Q 3.8.3: Find the manager and time when a skill was acquired between 1983 and 1987 (inclusive) by anyone who acquired a skill between these times.
EXPECTED ANSWER: "(*DI*, 1-Jun-1984) and (*DI*, 1-Jan-1985)."
CATEGORY: (Projected, None) / (Containment, Interval, Computed) / (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT E1.Name, BEGIN(S)
FROM EMP(ID, Name) As E1, EMP(ID, DeptNname) As E2, SKILL(EmpID, Name)(PERIOD) As S,
    DEPT(Name, MgrID) As D
WHERE DATE '1/1/1983' PRECEDES BEGIN(VALID(S))
    AND BEGIN(VALID(S)) PRECEDES DATE '1/1/1987' AND S.EmpID = E2.ID
    AND E2.DeptName = D.Name AND D.MgrID = E1.ID
```

TSQL2 ANSWER: {('Di', 6/1/1984), ('Di', 1/1/1985)}

QUERY Q 3.8.4: For anyone who had two raises in the period March 1982 to March 1985 (inclusive), find the current name and the dates when raises occurred during the aforementioned times.

EXPECTED ANSWER: "(Edward, 1-Jun-1982, 1-Feb-1985)."
CATEGORY: (Projected, None) / (Containment, Interval, Computed) / (=, Constant)
TSQL2 QUERY:

```
SELECT SNAPSHOT E2.Name, BEGIN(E1)
FROM EMP(ID, Salary) PERIOD As E1, EMP(ID, Name) As E2
WHERE E1.ID = E2.ID AND VALID(E2) CONTAINS CURRENT_DATE
    AND DATE '3/1/1982' PRECEDES BEGIN(VALID(E1))
    AND BEGIN(VALID(E1)) PRECEDES DATE '3/1/1985' AND 2 <= (SELECT SNAPSHOT count(*)
            FROM EMP(ID, Salary) PERIOD As E3
            WHERE E3.ID = E1.ID
            AND E3.Salary > E1.Salary
            AND VALID(E1) PRECEDES VALID(E3)
            AND BEGIN(VALID(E3)) PRECEDES DATE '3/1/1985')
```

TSQL2 ANSWER: {('Edward', 6/1/1982), ('Edward', 2/1/1985)}

4.3.9 Class O3.S9 (Other, Element, Computed)

QUERY Q 3.9.1: Find the salary history associated with the name Ed when it was associated with a person that had the Driving skill.
EXPECTED ANSWER: "($20K, 2/1/82 – 5/1/82)," "($30K, 6/1/84 – 1/31/85)" and "($40K, 2/1/85 – 1/31/87 ∪ 4/1/87 – 12/31/87)"
CATEGORY: (Projected, Not Empty) / (Containment, Element, Computed) / (=, Constant)(=, Foreign)
TSQL2 QUERY:

```
SELECT Salary
VALID INTERSECT(Emp, Skills)
FROM Emp, Skills
WHERE Emp.Name='Ed' AND Skills.EmpId = Emp.Id AND Skills.Skill = 'Driving'
    AND VALID(Emp) OVERLAPS VALID(Skills)
```

TSQL2 ANSWER: {(20 | [2/1/1982 - 5/1/1982]), (30 | [6/1/1984 - 1/31/1985]), (40 | [2/1/1985 - 1/31/1987] ∪ [4/1/1987 - 12/31/1987])}

QUERY Q 3.9.2: Find the salary history, during the periods in which *ED* had a driving skill, of the employees who earned less than $50K throughout 1989.
EXPECTED ANSWER: "(—, 1/1/82 – 1/31/82)," "($20K, 2/1/82 – 5/1/82)," "($30K, 6/1/84 – 1/31/85)," "($40K, 2/1/85 – 1/31/87)," "(—, 1/31/87 – 3/30/87)" and "($40K, 4/1/87 – 5/31/88)."
CATEGORY: (Projected, Not Empty) / (Containment, Element, Computed) (Containment, Interval, Explicit) / (=, Constant) (<>, Constant)
TSQL2 QUERY:

```
SELECT E1.Salary
VALID INTERSECT(E1, Skills)
FROM Emp(Id,Salary)(PERIOD) AS E1 E2, Skills
WHERE E1.Id = E2.Id AND E2.Salary < 50 AND VALID(E2) CONTAINS PERIOD '[1/1/89 - 12/31/89]'
    AND Skills.Skill = 'Driving' AND Skills.EmpId = 'ED' AND E1 overlaps Skills
```

TSQL2 ANSWER: {(20 | [2/1/1982 - 5/1/1982]), (30K | [6/1/1984 - 1/31/1985]), (40K | [2/1/1985 - 1/31/1987]), (40K | [4/1/1987 - 5/31/1988])}

QUERY Q 3.9.3: Find the name and salary histories of male employees when they were directed by a woman.
EXPECTED ANSWER: "(Ed, 2/1/82 – 1/31/87)" and "(($20K, 2/1/82 – 5/31/82), ($30K, 6/1/82 – 1/31/85), ($40K, 2/1/85 – 1/31/87))."
CATEGORY: (Projected, Not Empty) / (Containment, Element, Computed) / (=, Constant) (=, Foreign)
TSQL2 QUERY:

```
SELECT E.Name, E.Salary
VALID INTERSECT(E, Dept)
FROM Emp(Id, Name, Salary) as E, Emp AS Mgr, Dept(Name, EmpId) AS D
WHERE E.Gender = 'M' AND E.DeptName = Dept.Name
    AND D.EmpId = Mgr.Id AND Mgr.Gender = 'F' AND VALID(E) OVELAPS VALID(D)
```

TSQL2 ANSWER: {(Ed, 20 | [2/1/1982 - 5/31/1982]), (Ed, 30 | [6/1/1982 - 1/31/1985]), (Ed, 40 | [2/1/1985 - 1/31/1987])}

QUERY Q 3.9.4: Find the department, the current manager name, and the periods when a department manager earned more than one third of the departmental budget.
EXPECTED ANSWER: "(Toy, Di, 1/1/87 – now), (Book, Edward, 4/1/87 – now)."
CATEGORY: (Projected, Not Empty) / (Containment, Element, Computed) (Containment, Event, Explicit) / (=, Foreign) (<>, Arbitrary)
TSQL2 QUERY:

```
SELECT Dept.Name, CMgr.Name
VALID INTERSECT(Dept, Mgr)
FROM Dept, Emp(Id, Salary, DeptName) AS Mgr, Mgr(Name) AS CMgr
WHERE Dept.EmpId = Mgr.Id AND (Dept.Budget / 3) < Mgr.Salary
    AND VALID(Dept) OVERLAPS VALID(Mgr) AND VALID(CMgr) CONTAINS CURRENT_DATE
```

TSQL2 ANSWER: {(Toy, Di | [1/1/1987 - 1/1/1990]),(Book, Edward | [4/1/1987 - 1/1/1990])}

QUERY Q 3.9.5: Find the name and the salary history of the employees in the periods they earned as much as their managers (distinct from themselves).
EXPECTED ANSWER: "(Ed, $30K, 6/1/82 – 7/31/84)" and "(Ed, $40K, 2/1/85 – 8/31/86)."
CATEGORY: (Projected, Not Empty) / (Containment, Element, Computed) / (=, Foreign) (=, Single) (<>, Single)
TSQL2 QUERY:

```
SELECT E.Name, E.Salary
FROM Emp(Id, Name, Salary) AS E, Emp(Id, Salary) AS Mgr, Dept
WHERE E.DeptName = Dept.Name AND VALID(E) OVERLAPS VALID(Dept)
    AND Dept.EmpId = Mgr.Id AND VALID(Dept) OVERLAPS VALID(Mgr)
    AND E.Salary = Mgr.Salary AND E.Id <> Mgr.Id AND VALID(E) OVERLAPS VALID(Mgr)
```

TSQL2 ANSWER: {(Ed, 30 | [6/1/1982 - 7/31/1984]), (Ed, 40 | [2/1/1985 - 8/31/1986])}
COMMENTS: The default "`VALID INTERSECT(E, Mgr, Dept)`" works correctly. The last "overlaps" condition is required to avoid tuples with a "null" timestamp to be retrieved.

QUERY Q 3.9.6: When did one person earn a lower salary than another younger person, and who were those persons?

EXPECTED ANSWER: "*ED* and *DI*, 02/01/82–05/31/82,
08/01/84–01/31/85, 09/01/86–01/31/87,
04/01/87–01/01/90."
CATEGORY: (Projected, Element, Derived) / (Containment, Element, Computed) / (<>, Single)
TSQL2 QUERY:

```
SELECT Emp1.Id, Emp2.Id
FROM Emp(Id, Salary) AS Emp1 Emp2
WHERE Emp1.Salary < Emp2.Salary AND Emp1.D-birth < Emp2.D-birth
    AND VALID(Emp1) OVERLAPS VALID(Emp2)
```

TSQL2 ANSWER: {(ED, DI | [2/1/1982 - 7/31/1982]), (ED, DI | [8/1/1984 - 01/31/1985]),
(ED, DI | [9/1/1986 - 1/1/1990])}

QUERY Q 3.9.7: When and who had the same salary for the longest continuous period of time?
EXPECTED ANSWER: "09/01/86–01/01/90, *DI*"
CATEGORY: (Projected, Element, Derived) / (Containment, Element, Computed) / (<>, SIngle)
TSQL2 QUERY:

```
SELECT Id
FROM Emp(Id, Salary)(PERIOD) AS E
WHERE CAST(VALID(E) AS INTERVAL DAY) >= ALL (SELECT SNAPSHOT CAST(E AS INTERVAL DAY)
                            FROM Emp(Id, Salary)(PERIOD) AS E)
```

TSQL2 ANSWER: {(DI | [09/01/1986 - 01/01/1990])}
COMMENTS: Aggregates (e.g. MAX) could also be used.

QUERY Q 3.9.8: List *DI*'s skill and salary histories during the time she was a manager.
EXPECTED ANSWER: "(Directing, 01/01/82–01/01/90)" and "(30K, 01/01/82–07/31/84), (40K, 08/01/84–
08/31/86), (50K, 09/01/86–01/01/90)"
CATEGORY: (Projected, Element, Intersection) / (Containment, Element, Computed) / (=, Constant)
TSQL2 QUERY:

```
SELECT Skills.Skill, Emp.Salary
FROM Emp, Dept, Skills
WHERE Emp.Id = 'DI' AND Dept.EmpId = 'DI' AND VALID(Dept) OVERLAPS VALID(Emp)
    AND Skills.EmpId = 'DI' AND VALID(Skills) OVERLAPS VALID(Emp)
    AND VALID(Dept) OVERLAPS VALID(Skills)
```

TSQL2 ANSWER: {('Directing', 30 | [1/1/1982 - 7/31/1984]), ('Directing', 40 | [8/1/1984
- 8/31/1986]), ('Directing', 50 | [09/01/1986 - 01/01/1990])}

QUERY Q 3.9.9: List the names and salary histories of all employees when they were managers and
earned at least 36K.
EXPECTED ANSWER: "((Di, 08/01/84–01/01/90),
((40K, 08/01/84– 08/31/86), (50K, 09/01/86–01/01/90))), (((Ed, 04/01/87–12/31/87), (Edward,
01/01/88–01/01/90)), (40K, 04/01/87– 01/01/90))"
CATEGORY: (Projected, Element, Derived) / (Containment, Element, Computed) / (<>, Constant) (=,
Foreign)

TSQL2 QUERY:

```
SELECT Emp.Name, Emp.Salary
FROM Emp, Dept
WHERE Emp.Salary > 36 AND Emp.Id = Dept.EmpId AND VALID(Emp) OVERLAPS VALID(Dept)
```

TSQL2 ANSWER: {('Di', 40 | [8/1/1984 - 8/31/1986]), ('Di', 50 | [9/1/1986 - 1/1/1990]), ('Ed', 40 | [4/1/1987 - 12/31/1987]), ('Edward', 40 | [1/1/1988 - 1/1/1990])}

4.3.10 Class O3.S10 (Other, Element, Other)

QUERY Q 3.10.1: Find the budget history in the period from 1/1/82 to 12/31/84 and from 1/1/87 till now of all departments *ED* ever worked in.
EXPECTED ANSWER: "(Toy, $150K, 2/1/82 – 7/31/84)," "(Toy, $200K, 8/1/84 – 12/31/84)," "(Toy, $100K, 1/1/87 – now)," and "(Book, $50K, 4/1/87 – now)."
CATEGORY: (Projected, Not Empty) / (Containment, Element, Explicit) / (=, Foreign) (=, Constant)
TSQL2 QUERY:

```
SELECT D.Budget
VALID PERIOD '[1/1/1982 -12/31/1984]' + PERIOD '[1/1/1987 - now]'
FROM EMP(ID, Name, DeptName) As E1, DEPT(Name, Budget) As D
WHERE E1.Name = 'ED' AND E1.DeptName = D.Name
    AND VALID(D) CONTAINS PERIOD '[1/1/198 - 12/31/1984]'
    OR D CONTAINS PERIOD '[1/1/1987 - now]'
```

TSQL2 ANSWER: {('Toy', 150000 | [2/1/1982 - 7/31/1984]), ('Toy', 200000 | [8/1/1984 - 12/31/1984]), ('Toy', 100000 | [1/1/1987 - 1/1/1990]), ('Book', 50000 | [4/1/1987 - 1/1/1990])}

QUERY Q 3.10.2: Find the name and the budget history in 1984 and 1987 of the department being directed by Di
EXPECTED ANSWER: "(Toy, $150K, 1/1/84 – 7/31/84)," "(Toy, $200K, 8/1/84 – 12/31/84)" and "(Toy, $100K, 1/1/87 – 12/31/87)."
CATEGORY: (Projected, Not Empty) / (Containment, Element, Explicit) / (=, Constant)
TSQL2 QUERY:

```
SELECT D.Name, D.Budget
VALID PERIOD '[1/1/1994 - 12/31/1984]' + PERIOD '[1/1/1987 - 12/31/1987]'
FROM EMP(ID, Name) As E1, DEPT(Name, Budget, MgrID) As D
WHERE E1.ID = D.MgrID AND E1.Name = 'Di'
```

TSQL2 ANSWER: {('Toy', 150000 | [1/1/1984 - 7/31/1984]), ('Toy', 200000 | [8/1/1984 - 12/31/1984]), ('Toy', 100000 | [1/1/1987 - 12/31/1987])}

QUERY Q 3.10.3: Find the name of the department where *ED* working at the beginning of both of the years 1986 and 1987, and the periods *ED* worked there.
EXPECTED ANSWER: "(Toy, 2/1/82 – 1/31/87)."
CATEGORY: (Projected, Not Empty) / (Containment, Element, Explicit) / (=, Constant)
TSQL2 QUERY:

```
SELECT E1.DeptName
```

```
FROM EMP(Name, DeptName) As E1
WHERE E1.Name = 'Ed' AND VALID(E1) CONTAINS DATE '1/1/1986' AND EXISTS (SELECT *
            FROM EMP(Name, DeptName) As E2
            WHERE E1.Name = E2.Name AND E1.DeptName = E2.DeptName
                AND VALID(E2) CONTAINS DATE '1/1/1987')
```

TSQL2 Answer: {('Toy' | [2/1/1982, 1/31/1987])}

Query Q 3.10.4: Find the current name of the manager *ED* had on both 1984's Christmas and his 27^{th} birthday, and the dates the manager started as a manager.
Expected Answer: "(Di, 1/1/82)."
Category: (Projected, Not Empty) / (Containment, Element, User-defined) / (=, Constant) (=, Single) (=, Foreign)
TSQL2 Query:

```
SELECT SNAPSHOT E1.Name, BEGIN(D1)
FROM EMP(ID, Name) As E1, DEPT(Name, MgrID) As D1, EMP(ID, Name, DeptName) As E2
WHERE E1.ID = D1.MgrID AND E2.Name = 'Ed' AND E2.DeptName = D1.Name AND
    VALID(E2) CONTAINS DATE '12/25/1984'
    AND D1.MgrID = (SELECT D2.MgrID
            FROM DEPT(Name, MgrID) As D2,
            EMP(ID, D-birth, DeptName) As E3
            WHERE E3.ID = E2.ID AND D2.Name = E3.DeptName
            AND VALID(E3) CONTAINS E3.D-birth + INTERVAL '27' YEAR)
```

TSQL2 Answer: {('Di', 1/1/1982)}

Query Q 3.10.5: Find the department name, the then manager, the modification dates and the new values of the budget for every budget change that occurred in 1984, 1986 and 1988.
Expected Answer: "(Toy, Di, $200K, 8/1/84)."
Category: (Complete, Not Empty) / (Containment, Element, Explicit) (Ordering, Interval, Computed) / (=, Single) (<>, Single)
TSQL2 Query:

```
SELECT SNAPSHOT D.Name, E.Name, D.Budget, BEGIN(D)
VALID PERIOD '[1/1/1984 - 12/31/1984]' + PERIOD '[1/1/1986 - 12/31/1986]'
       + PERIOD '[1/1/1988 - 12/31/1988]'
FROM EMP(ID, Name) As E, DEPT(Name, Budget, MgrID) PERIOD As D1
WHERE E.ID = D1.MgrID
    AND D1.Budget <> (SELECT SNAPSHOT D2.Budget
            FROM DEPT(Name, Budget) PERIOD As D2
            WHERE D2.Name = D1.Name AND VALID(D1) MEETS VALID(D2))
```

TSQL2 Answer: {('Toy', 'Di', 200000, 8/1/1984)}

5 Acknowledgements

Raghu Ramakrishnan provided many helpful comments on this document, as well as several queries. Support was provided in part by the National Science Foundation under grant IRI-9302244 to the editor.

6 Bibliography

[Jensen et al. 1993] Jensen, C.S. (editor), J. Clifford, S.K. Gadia, F. Grandi, P.P. Kalua, N. Kline, N. Lorentzos, Y. Mitsopoulos, A. Montanari, S.S. Nair, E. Peressi, B. Pernici, E.L. Robertson, J.F. Roddick, N.L. Sarda, M.R. Scalas, A. Segev, R.T. Snodgrass, A. Tansel, R. Tiberio, A. Tuzhilin, and G.T.J. Wuu, "A Consensus Test Suite of Temporal Database Queries," Technical Report R 93-2034, Dept. of Mathematics and Computer Science, Aalborg University, Denmark, November, 1993, 45 pages.

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