Bases de Datos Temporales, Espaciales y Espacio-Temporales

Nicolás Del Piano

Índice general

Resúmen

Este trabajo presenta una introducción hacia las Bases de Datos Temporales, Espaciales y Espacio-Temporales. Las Bases de Datos Temporales (*Temporal Databases*) están diseñadas para la captura de información que varía en el tiempo. Las Bases de Datos Espaciales (*Spatial Databases*) fueron concebidas por la necesidad de registrar el cambio geográfico y físico de cierta información. Por último, las Bases de Datos Espacio-Temporales (*Spatio-Temporal Databases*) son el resultado de la unión de las capacidades y propiedades ofrecidas por ambos tipos de Bases de Datos. Primero se presentarán conceptos de Bases de Datos clásicas, para luego abordar más claramente los temas centrales de esta monografía. El segundo capítulo presenta de una manera detallada las Bases de Datos Temporales, el tercero lo hace para las Espaciales, y el cuarto para las Espacio-Temporales. Por último se abordan tópicos generales relacionados con estos tipos de Bases de Datos.

Introducción

Hoy en día, la cantidad de información que manejan las corporaciones y empresas es gigantesca. Por esta razón, es necesario el uso de una herramienta que provea una forma de gestionar adecuadamente esta información. Este es el propósito de las Bases de Datos: brindar al usuario una forma de controlar el acceso, almacenamiento y administración de los datos de la entidad en cuestión.
Con la aparición de nuevas tecnologías, el surgimiento de nuevas necesidades fue inevitable, implicando que las Bases de Datos Relacionales no sean una bala de plata (aunque sean las más usadas actualmente) para resolver todos los problemas de gestión de datos. Surgieron conceptos como Minería de Datos, Data Warehouse y Big Data: la información ya no tiene la misma dimensión que antes. Fue entonces cuando el Modelo Relacional clásico necesitaba extenderse para representar eficientemente datos que varíen en tiempo y espacio.
Las Bases de Datos Temporales se encargan del dominio del tiempo y su relación con los datos, permite analizar la historia y controlar la validez temporal de los mismos. Una gran variedad de aplicaciones del mundo real manejan datos variables en el tiempo: control de inventario, registros médicos, operaciones bancarias, sistemas de información geográfica, gestión de reservas, aplicaciones científicas, etcétera. Esta necesidad de referencias temporales justifica la creación de un modelo de datos temporal.
Las Bases de Datos Espaciales extienden el modelo para representar el dominio espacial, con estructuras que puedan identificar un objeto en el espacio. Deben permitir la descripción de objetos espaciales mediante tres características: atributos, localización y topología. Además deben proveer tipos de datos espaciales para estructurar entidades geométricas en el espacio. Existen diversas áreas donde la gestión de información geométrica, geográfica o espacial es crucial: Sistemas de Información Geográfica, Bases de Datos multimedia, imágenes satelitales, ciencias ambientales, astronomía.
El objetivo de las Bases de Datos Espacio-Temporales es extender los modelos de información espacial para incluír el tiempo y describir en forma más dinámica la realidad que se quiere representar. El modelo espacio-temporal abarca aplicaciones demográficas, ecológicas, relacionadas con marketing, militares, urbanísticas y de fenómenos naturales, entre otras.

Bases de Datos Relacionales

Bases de Datos Temporales

El tiempo es un aspecto importante para los fenómenos del mundo real: los eventos ocurren en momentos de tiempo específicos.
A veces nos interesa saber con cierta certeza cuándo ocurrió tal evento, y poder compararlo con otros para obtener información de interés.
Muchas de las áreas donde se aplican las Bases de Datos tienen naturaleza temporal:

- Control de inventario.
- Registros médicos.
- Sistemas de información geográfica.
- Operaciones bancarias.
- Data Warehousing.
- Sistemas de control de reservas (aerolíneas, hoteles, etc).
- Aplicaciones científicas.

Relaciones no temporales

En la Figura 3.1 puede observarse una tabla relacional no temporal. Cada tupla

Id	Nombre	Estado	Sueldo
1	Juan	Activo	5700
2	Manuel	Activo	2300

Figura 3.1: Tabla relacional no temporal.

representa un hecho verdadero *ahora*. Solo hay un estado representable de la Base de Datos: *el actual* (*current snapshot*). A medida que el tiempo transcurre, los datos se van actualizando y modificando. Con este modelo, perdemos información. Las Bases de Datos convencionales representan el estado de la información en un instante de tiempo dado. Aunque la Base de Datos es actualizada, estos cambios son vistos como modificaciones del estado actual y los datos obsoletos son borrados. Por lo tanto, solo podemos utilizar la información actual de la Base de Datos.
Esto genera un problema cuando queremos responder preguntas involucradas a intervalos de tiempo: *¿Cuáles empleados percibieron un aumento el mes pasado?*

Bases de Datos Temporales: Definición

Un *DBMS Temporal* es un Sistema de Gestión de Bases de Datos que proporciona herramientas para el manejo y control de Bases de Datos Temporales.
Una *Base de Datos Temporales* es una Base de Datos que tiene dimensión del tiempo a través del almacenamiento de datos temporales.
Proporcionan un marco que mantiene la historia de los cambios que se produjeron en la fuente de datos. Están diseñadas para la captura de la información que varía en el transcurso del tiempo (puede apreciarse esta relación en la Figura 3.2).

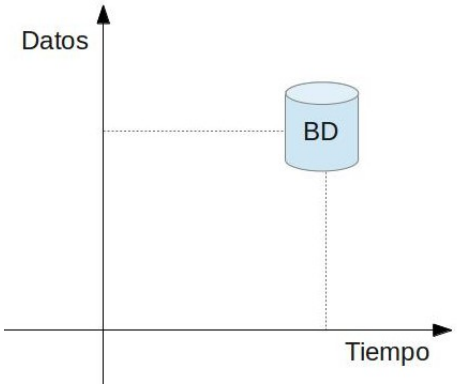


Figura 3.2: Relación tiempo y datos.

Datos Temporales

Un *Dato Temporal* es un dato convencional al que se le asocia un período de tiempo para expresar valores temporales en la Base de Datos.
Este agregado de información temporal se denomina *time-stamping*. Al asociar el tiempo con la información, es posible almacenar diferentes estados de una base de datos.

Dimensión del Tiempo

El tiempo es infinito, continuo y multidimensional [1]. Las computadoras no pueden representar información continua, de hecho, se aproximan discretamente. Así, para representar una noción del tiempo, se lo transforma en un conjunto discreto con una cierta granularidad. Por ejemplo, la sentencia *Homero Simpson nació el 12 de Mayo de 1956* tiene una granularidad de días.
Las Bases de Datos Temporales almacenan dos dimensiones de tiempo:

- Tiempo Válido
- Tiempo Transaccional

El *Tiempo Válido* representa cuándo un hecho tiene validez, es decir, es verdadero en el mundo real. El Tiempo Válido de un evento es el tiempo de un reloj en el que ese evento ocurrió [1]. Es independiente de si dicho evento fue registrado o no en la Base de Datos. Los Tiempos Válidos pueden encontrarse en el pasado, presente o futuro. Una de las características es que todos los eventos tienen asociado un Tiempo Válido, pero no necesariamente son registrados. Además brindan

la capacidad de gestionar la historia de la Base de Datos.
El *Tiempo Transaccional* registra el período de tiempo donde un hecho fue almacenado en la Base de Datos. Permiten realizar consultas que muestren el estado de la Base de Datos en un tiempo específico. Este tiempo está acotado en ambos extremos; la creación de la Base de Datos y el tiempo presente, es decir, los Datos Transaccionales viven solamente dentro de la vida de una Base de Datos. Una de las capacidades interesantes es que permiten volver hacia un estado anterior (*roll-back*), ya que almacenan datos de las operaciones que se fueron haciendo.
Estas dos dimensiones son ortogonales. Un Modelo de Datos que no soporte ninguno de estas dimensiones se denomina *snapshot*, ya que captura solamente una imagen de la Base de Datos. Si se brinda soporte para Tiempo Válido, entonces se cuenta con una Modelo de Datos *histórico*, mientras que uno que soporte Tiempo Transaccional solamente se denomina *rollback*. En caso de que ambas dimensiones estén soportadas, se denomina *bitemporal*.

Ejemplo

Análizamos en forma no temporal y temporal el siguiente ejemplo:

- El Señor X nace en Springfield el 12 de Mayo de 1956.

- Su padre lo registra el 13 de Mayo de 1956.
- Se muda a Arroyos Cipreses el 3 de Agosto de 1980, pero olvida registrarse; lo hace el 16 de Agosto del mismo año.
- Muere el 20 de Abril de 2004.

Ejemplo: no temporal

Nombre	ViveEn
Señor X	Springfield

⇓ Update

Nombre	ViveEn
Señor X	Arroyos Cipreses

⇓ Delete

Nombre	ViveEn
~~Señor X~~	~~Arroyos Cipreses~~

Ejemplo: Tiempo Válido

Nombre	ViveEn	Valid-From	Valid-To
Señor X	Springfield	12-May-1956	∞

⇓ Update

Nombre	ViveEn	Valid-From	Valid-To
Señor X	Springfield	12-May-1956	2-Aug-1980

⇓ Insert

Nombre	ViveEn	Valid-From	Valid-To
Señor X	Springfield	12-May-1956	2-Aug-1980
Señor X	Arroyos Cipreses	3-Aug-1980	∞

⇓ Update

Nombre	ViveEn	Valid-From	Valid-To
Señor X	Springfield	12-May-1956	2-Aug-1980
Señor X	Arroyos Cipreses	3-Aug-1980	20-Apr-2004

Ejemplo: Tiempo Transaccional

Nombre	ViveEn	Transaction-From	Transaction-To
Señor X	Springfield	13-May-1956	∞

⇓ Insert

Nombre	ViveEn	Transaction-From	Transaction-To
Señor X	Springfield	13-May-1956	16-Aug-1980
Señor X	Arroyos Cipreses	16-Aug-1980	∞

⇓ Update

Nombre	ViveEn	Transaction-From	Transaction-To
Señor X	Springfield	13-May-1956	16-Aug-1980
Señor X	Arroyos Cipreses	16-Aug-1980	20-Apr-2004

Base de Datos Bitemporal

Incluyen ambos tiempos (Válido y Transaccional) lo que les permite proveer información histórica, a la vez que brindan la capacidad de hacer roll-back de los datos. En la siguiente figura se puede apreciar un ejemplo:

Nombre	ViveEn	Valid-From	Valid-To	Transaction-From	Transaction-To
Señor X	Springfield	12-May-1956	∞	13-May-1956	16-Aug-1980
Señor X	Springfield	12-May-1956	2-Aug-1980	16-Aug-1980	∞
Señor X	Arroyos Cipreses	3-Aug-1980	∞	16-Aug-1980	20-Apr-2004
Señor X	Arroyos Cipreses	3-Aug-1980	20-Apr-2004	20-Apr-2004	∞

Figura 3.3: Ejemplo de una relación bitemporal.

Extensiones Temporales

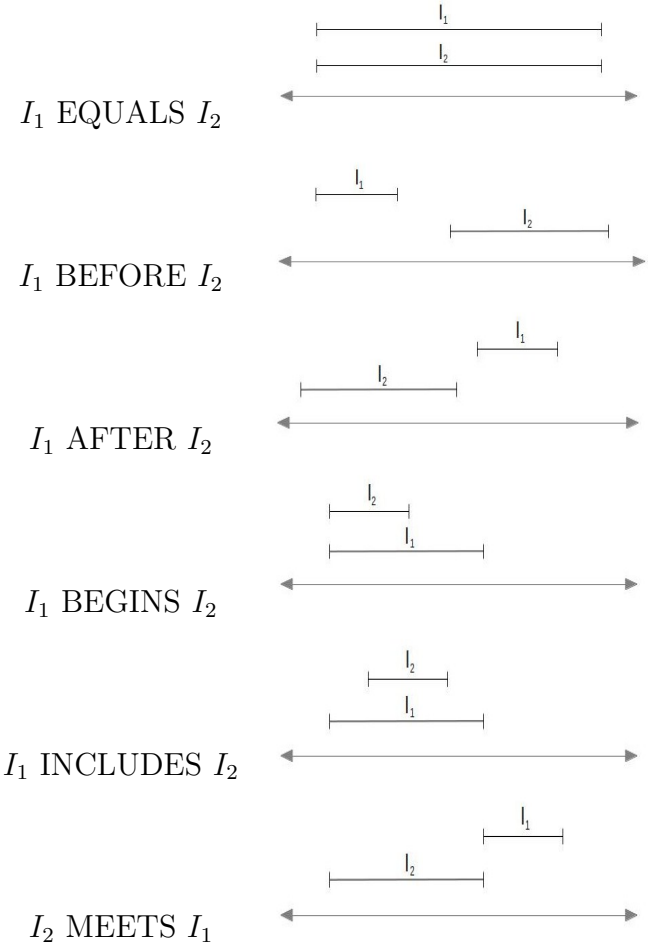
Hay dos formas de extender el modelo relacional para especificar requisitos temporales.
La forma mostrada en los ejemplos antes mencionados se denomina **marcaje de tuplas**. Este método es muy común en el modelo relacional. Se utiliza un atributo especial para indicar la validez de una tupla: se indica *desde* y *hasta* para representar intervalos de tiempo.

$(attr_1, ..., attr_n) \rightarrow (attr_1, ..., attr_n, temp\_attr_1, ..., temp\_attr_m)$

Una de las desventajas que tiene esta forma es que una entidad puede estar representada por varias tuplas, por lo que no puede lograrse una representación 1:1 de la realidad lo que podría generar información redundante.
La segunda forma se denomina **marcaje de atributos** y usa atributos multivaluados. La idea es que la marca de tiempo y la entrada referenciada se almacenen en el mismo atributo de forma anidada. Al mismo tiempo que permite la correspondencia 1:1 entre entidades y hechos reales, dificulta las actualizaciones y no cumple la 1NF.

Operadores de Allen

Necesitamos una forma de comparar datos temporales. Allen (1983) propone un conjunto de operadores temporales lógicos para comparar intervalos de tiempo. El operador es una función de tipo: $I_t \times I_t \rightarrow \{True, False\}$.
Algunos ejemplos de estos operadores:



Modelo Relacional Temporal

El modelo relacional temporal incorpora la semántica temporal en el modelo relacional utilizando marcaje de tuplas como extensión temporal.
Alguna de las características que debe ofrecer son:

- Un tipo de datos para períodos de tiempo, incluyendo la habilidad de representar períodos infinitos.
- Soporte para tiempo válido, transaccional y tablas bitemporales.
- Tiempo transaccional controlado por el sistema.
- Consultas temporales.
- Predicados y operadores que actúen sobre intervalos de tiempo.

Lenguaje de consulta temporales

Una Base de Datos Temporal es un repositorio de información temporal. Un lenguaje de consulta temporal es cualquier lenguaje de consulta para bases de datos temporales.
Las propiedades de un lenguaje de consulta temporal son:

- Semántica declarativa

- Implementación eficiente.
- Independencia en la representación.
- Expresividad en la consulta.

Algunos ejemplos importantes son: TSQL, TQuel, HRDM, Backlog. Muchos otros son derivados de éstos.

DBMS Temporal

Los DBMS Temporales deben respaldar un lenguaje de definición de datos temporales, un sistema de restricciones temporales, un lenguaje de manipulación de datos y un lenguaje de consulta temporal.
Algunos ejemplos de DBMS Temporales son: TimeDB, Oracle Workspace Manager, PostgreSQL, IBM DB2, entre otros.

Historia y estandarización

Hacia un modelo de datos temporal

La comunidad de las Bases de Datos Temporales fue prolífica en la producción de modelos de datos temporales y lenguajes de consulta.
En los últimos 20 años, docenas de modelos relacionales de datos temporales fueron propuestos. *Richard Snodgrass* propuso en 1992 que la comunidad de Bases de Datos Temporales realice extensiones a SQL.

Proceso de estandarización de SQL

El responsable del estándar SQL en Estados Unidos es INCITS(ANSI) DM32.2. Internacionalmente, es responsable el comité ISO/IEC JTC 1/SC 32 Data Management and Interchange/WG3. Muchas de las capacidades del estándar SQL fueron originadas en Estados Unidos. Usualmente la aprobación de nuevos estándares tiene un ciclo de 3 a 5 años. Actualmente, hay 7 versiones del Standard SQL: 86(SQL-87), 89, 92(SQL2), SQL:1999(SQL3), SQL:2003, SQL:2008, SQL:2011.

SQL Temporal: primer intento (1995-2001)

X3H2(ahora conocido como DM32.2) y WG 3 aprobaron el trabajo SQL/Temporal en 1995. Estados Unidos fue el primero en hacer la propuesta, añadiendo nuevas extensiones a SQL, basadas en el trabajo de Richard Snodgrass. La propuesta de USA estaba basada en TSQL2, una extensión de SQL-92. En relación al trabajo realizado por Estados Unidos, el Reino Unido lanzó una propuesta muy similar, pero debido a conflictos entre estas propuestas, ANSI e ISO decidieron cancelar en el año 2001 el proyecto SQL/Temporal.

SQL Temporal: segundo intento (2008-2011)

En 2008, un segundo intento se hizo presente. Empezó con la aceptación de la propuesta *system-versioned tables* llevada a cabo por INCITS DM32.2 y ISO/IEC JTC1 SC32 WG3. La idea no fue resucitar SQL/Temporal, sino que se añadieron

las ideas a SQL/Foundation. Una de las características interesantes son las *application-time period tables*.
Las características temporales en SQL:2011 están inspiradas en las anteriores propuestas hechas en SQL/Temporal, pero con una sintaxis un poco diferente.

Características temporales en SQL-92

Inclusión de tipos de datos temporales:

- DATE (10 posiciones) = YEAR, MONTH, DAY (yyyy-mm-dd)
- TIME (8 posiciones) = HOUR, MINUTE, SECOND (hh:mm:ss)
- TIMESTAMP (DATE, TIME, fracciones de segundo y desplazamiento de acuerdo al huso horario estándar)
- INTERVAL: período de tiempo. Para incrementar/decrementar el valor actual

TSQL2

TSQL2 [2] es un lenguaje de consulta temporal consensuado por un comité de grupos de investigación en bases de datos temporales. Originalmente apareció como una extensión para SQL-92. La especificación incluye las ideas y conceptos de Tiempo Válido, Tiempo Transaccional y tablas bitemporales. Este lenguaje es conocido también como extensiones ANSI X3.135-1992 y ISO/IEC 9075:1992.
Algunas de las características principales:

- Incluye cuatro tipos de marcas de tiempo válido: intervalos, instantes, períodos y elementos.
- Asociadas a estas marcas, existen tres categorías de operadores: extractores, constructores y comparadores.
- Tiene una variable denominada NOW que indica el momento actual (es usada como tiempo de referencia).

Extractores

Operación	Operador
Extractor de eventos	BEGIN(event) END(period) BEGIN(element) END(event) END(period) END(element)
Extractor de períodos	FIRST(period FIRST(element) LAST(period) LAST(element)

Constructores

Operación	Operador
Constructor de eventos	FIRST(event, event) LAST(event, event)
Constructor de períodos	PERIOD(event, event) INTERSECT(period, period)
Constructor de elementos	INTERSECT(element, element) element + element element − element

Comparadores

Operación	Operador
Comparador de eventos	event PRECEDES event event = event event OVERLAPS event event MEETS event event CONTAINS event
Comparador de períodos	period PRECEDES period period = period period OVERLAPS period period MEETS period period CONTAINS period
Comparador de elementos	element PRECEDES element element = element element OVERLAPS element element CONTAINS element

Operadores de Allen

Operador de Allen	Operador TSQL2
a BEFORE b	a PRECEDES b
a EQUAL b	a = b
a OVERLAPS b	a OVERLAPS b AND END(a) PRECEDES END(b)
a MEETS b	END(a) = BEGIN(b)
a DURING b	BEGIN(b) PRECEDES BEGIN(a) AND END(a) PRECEDES END(b)
a START b	BEGIN(a) = BEGIN(b) AND END(a) PRECEDES END(b)
a FINISH b	BEGIN(b) PRECEDES BEGIN(a) AND END(a) = END(b)

Ejemplos de consulta en TSQL2: Tiempo Válido

> CREATE TABLE empleado(ename VARCHAR(12), eno INTEGER PRIMARY KEY, cumple DATE);
> CREATE TABLE salario(eno INTEGER REFERENCES empleado(eno), sueldo INTEGER);

> INSERT INTO empleado
VALUES('Homero', 1, DATE '1955-03-21');
> INSERT INTO empleado
VALUES('Lenny', 2, '1956-09-18');

> INSERT INTO salario VALUES(1, 2000);
> INSERT INTO salario VALUES(2, 4000);
> ALTER TABLE salario ADD VALIDTIME PERIOD(DATE);
> ALTER TABLE empleado ADD VALIDTIME PERIOD(DATE);

> INSERT INTO empleado
VALUES('Carl', 3, DATE '1955-08-10');
> INSERT INTO salario VALUES(3, 3500);
> COMMIT;

ename	eno	cumple	Válido
Homero	1	1955-03-21	[1995-02-01 - 9999-12-31)
Lenny	2	1956-09-18	[1995-02-01 - 9999-12-31)
Carl	3	1955-08-10	[1995-02-01 - 9999-12-31)

eno	sueldo	Válido
1	2000	[1995-02-01 - 9999-12-31)
2	4000	[1995-02-01 - 9999-12-31)
3	3500	[1995-02-01 - 9999-12-31)

Cambiar el nombre de 'Carl' por 'Carlos'.

> VALIDTIME UPDATE empleado
SET ename = 'Carlos'
WHERE ename = 'Carl';
> COMMIT;

ename	eno	cumple	Válido
Homero	1	1955-03-21	[1995-02-01 - 9999-12-31)
Lenny	2	1956-09-18	[1995-02-01 - 9999-12-31)
Carlos	3	1955-08-10	[1995-02-01 - 9999-12-31)

¿Quién percibió un aumento de sueldo?

> UPDATE salario
SET sueldo = 1.05 * amount
WHERE eno = 3;
> COMMIT;

> NONSEQUENCED VALIDTIME SELECT ename
FROM empleado AS E, salario AS S1, salario AS S2
WHERE E.eno = S1.eno AND E.eno = S2.eno
AND S1.sueldo < S2.sueldo AND
VALIDTIME(S1) MEETS VALIDTIME(S2);

ename
Carlos

Ejemplos de consulta: Tiempo Transaccional

ename	eno	cumple	Válido
Homero	1	1955-03-21	[1995-02-01 - 9999-12-31)
Lenny	2	1956-09-18	[1995-02-01 - 9999-12-31)
Carlos	3	1955-08-10	[1995-02-01 - 9999-12-31)

> ALTER TABLE empleado ADD TRANSACTIONTIME;
> COMMIT;

ename	eno	cumple	Válido	Transacción
Homero	1	1955-03-21	[1995-02-01 - 9999-12-31)	[1995-07-01 - 9999-12-31)
Lenny	2	1956-09-18	[1995-02-01 - 9999-12-31)	[1995-07-01 - 9999-12-31)
Carlos	3	1955-08-10	[1995-02-01 - 9999-12-31)	[1995-07-01 - 9999-12-31)

> UPDATE empleado
SET ename = 'Homero J.'
WHERE ename = 'Homero';
> COMMIT;

ename	eno	cumple	Válido	Transacción
Homero	1	1955-03-21	[1995-02-01 - 9999-12-31)	[1995-07-01 - 2014-04-23)
Homero J.	1	1955-03-21	[1995-02-01 - 9999-12-31)	[2014-04-23 - 9999-12-31)
Lenny	2	1956-09-18	[1995-02-01 - 9999-12-31)	[1995-07-01 - 9999-12-31)
Carlos	3	1955-08-10	[1995-02-01 - 9999-12-31)	[1995-07-01 - 9999-12-31)

¿Cuándo trabajó algún empleado por más de seis meses?

ename	eno	cumple	Válido	Transacción
Homero	1	1955-03-21	[1995-02-01 - 9999-12-31)	[1995-07-01 - 2014-04-23)
Homero J.	1	1955-03-21	[1995-02-01 - 9999-12-31)	[2014-04-23 - 9999-12-31)
Lenny	2	1956-09-18	[1995-02-01 - 9999-12-31)	[1995-07-01 - 9999-12-31)
Carlos	3	1955-08-10	[1995-02-01 - 1995-03-31)	[1995-07-01 - 9999-12-31)

> VALIDTIME AND TRANSACTIONTIME SELECT ename, eno
FROM empleado
WHERE INTERVAL(VALIDTIME(empleado) MONTH) >
INTERVAL '6' MONTH;

ename	eno	cumple	Válido	Transacción
Homero	1	1955-03-21	[1995-02-01 - 9999-12-31)	[1995-07-01 - 2014-04-23)
Homero J.	1	1955-03-21	[1995-02-01 - 9999-12-31)	[2014-04-23 - 9999-12-31)
Lenny	2	1956-09-18	[1995-02-01 - 9999-12-31)	[1995-07-01 - 9999-12-31)

Estado del Arte: Las BDT en la actualidad

Actualmente se pueden manipular los datos temporales de las siguientes formas:

- Usar un tipo de datos temporal integrado al DBMS y brindar soporte temporal con aplicaciones.
- Implementar un tipo de dato abstracto para el tiempo.
- Extender el modelo de datos no temporal a uno temporal.
- Generalizar un modelo de datos no temporal en uno temporal.

SQL:2011

El tiempo transaccional es manejado con *system-versioned tables*, que contienen el período de tiempo del sistema, y el tiempo válido es manejado con tablas que contienen *application-time period* (en la Figura 3.4 se puede apreciar esta relación). Las técnicas e ideas de TSQL se tuvieron en cuenta por el comité, pero las exten-

transaction time → system time
valid time → application time

Figura 3.4: Relación entre los diferentes tiempos.

siones sintácticas que se hicieron difirieron considerablemente de aquellas propuestas en TSQL.

SQL:2011 Application-Time tables

El tipo de datos PERIOD para intervalos de tiempo, sigue no disponible. Es simulado usando pares de instantes (con la semántica [cerrado,abierto)). Application-time period tables son tablas que contienen una cláusula PERIOD, con un nombre del período definido por el usuario. Estas tablas contienen también dos columnas adicionales para almacenar el tiempo de inicio y fin de un período de un dato temporal.

SQL:2011 System-Versioned tables

Son tablas que contienen una cláusula PERIOD con un nombre de período predefinido (SYSTEM_TIME). Contienen dos columnas adicionales que se refieren a el inicio y fin de una transacción. Ambos valores son seteados por el sistema. También preservan las versiones antiguas de las filas.

SQL:2011 Ejemplo

Se tiene la siguiente tabla:

emp_name	dept_id	start_date	end_date
John	M24	1998-01-31	9999-12-31
John	J13	1995-11-15	1998-01-31
Tracy	K25	1996-01-01	2000-03-31

¿En cuál departamento estuvo John el 1 de Diciembre de 1997?

SELECT dept_id
FROM empleados
WHERE emp_name = 'John' AND start_date $\leq$ DATE '1997-12-01' AND end_date $>$ DATE '1997-12-01';

¿En qué departamento está John ahora?

SELECT dept_id
FROM empleados
WHERE emp_name = 'John' AND start_date $\leq$ CURRENT_DATE AND end_date $>$ CURRENT_DATE;

Se borra la 3 fila el 16/4/2014:

DELETE FROM empleados
WHERE emp_name = 'Tracy';

La tabla queda:

emp_name	dept_id	system_start	system_end
John	M24	1998-01-31	9999-12-31
John	J13	1995-11-15	1998-01-31
Tracy	K25	1995-11-15	2014-04-16

SQL:2011 vs TSQL

La siguiente tabla comparativa muestra las diferencias entre ambos estándares:

TSQL	SQL:2011
valid time **transaction** time	**application** time **system** time
timestamping	versioning
validtime table **transactiontime** table **bitemporal table**	**application time period** table **system-versioned** table **system-versioned** **application time period** table

TimeDB

TimeDB es un DBMS Bitemporal basado en SQL [**?**]. Soporta un lenguaje de consulta, un lenguaje de manipulación de datos y un lenguaje de definición de datos. TimeDB brinda ATSQL2, un lenguaje de consulta basado en TSQL2, ChronoLog [**?**] y Bitemporal ChronoLog. Las características principales son que manipula los datos temporales buscando extender el modelo de datos relacional a uno temporal y traduce sentencias TSQL en sentencias SQL estándar para luego ser ejecutadas en DBMS como Oracle, Sybase, etcétera.

TimeDB: TDDL

En TimeDB, una tabla bitemporal puede crearse de esta manera:

CREATE TABLE empleados (EmpID INTEGER, Name CHAR(30), Department CHAR(40),
Salary INTEGER)
AS VALIDTIME AND TRANSACTIONTIME;

TimeDB: TDML

Las siguientes sentencias insertan datos temporales a una tabla:

VALIDTIME PERIOD '1985-1990'
INSERT INTO empleados VALUES (10,'John','Research',11000);

VALIDTIME PERIOD '1990-1993'
INSERT INTO empleados VALUES (10,'John','Sales',11000);

VALIDTIME PERIOD '1993-forever'
INSERT INTO empleados VALUES (10,'John','Sales',12000);

TimeDB: TQL

Para hacer consultas en TimeDB:

VALIDTIME
SELECT * FROM empleados;

TRANSACTIONTIME
SELECT * FROM empleados;

VALIDTIME AND TRANSACTIONTIME
SELECT * FROM empleados;

Oracle Workspace Manager

Es una herramienta de Oracle Database [**?**] que brinda a los desarrolladores la capacidad de manejar distintas versiones temporales de la base de datos.

Se puede habilitar el soporte de Tiempo Válido al momento de creación de una tabla:

- Crear la tabla de empleados y sus salarios:
  CREATE TABLE empleados (
  name VARCHAR(16) PRIMARY KEY,
  salary NUMBER
  );

- Versionar la tabla. Especificar TRUE para el soporte de tiempo válido:
  EXECUTE DBMS_WM.EnableVersioning('empleados','VIEW_WO_OVERWRITE',FALSE,T

- Insertar las filas:
  INSERT INTO empleados VALUES(
  'Adams',
  30000,
  WMSYS.WM_PERIOD(TO_DATE('01-01-1990','MM-DD-YYYY'),
  TO_DATE('01-01-2005','MM-DD-YYYY'))

El tipo de datos WM_PERIOD es usado para especificar el rango del tiempo válido.

Las constantes del tiempo válido son:
DBMS_WM.MIN_TIME $\approx$ 01-Jan-(-4712)
DBMS_WM.MAX_TIME $\approx$ 31-Dec-9999
DBMS_WM.UNTIL_CHANGED es un TIMESTAMP que se comporta como MAX_TIME hasta que es modificado.

Algunos operadores:
WM_OVERLAPS, WM_CONTAINS, WM_MEETS, WM_EQUALS, WM_INTERSECTION, etcétera.

Bases de Datos Espaciales

Bases de Datos Espacio-Temporales

Bibliografía

[1] "Snodgrass R. (1986). Temporal Databases."

[2] "TSQL2 Official Webpage." `http://digital.cs.usu.edu/~cdyreson/pub/temporal/tsql2.htm`.