The Data Organization

1251 Yosemite Way
Hayward, CA 94545
(510) 303-8868
rschoenrank@computer.org

Time Interval Processing

By Rainer Schoenrank

Data Warehouse Consultant

August 2012

Biography

Rainer Schoenrank is the senior data warehouse consultant for The Data Organization. He has degrees in physics from the University of Victoria and computer science from the University of Victoria and California State University Hayward. He has built data warehouses for clients such as Pacific Bell, Genentech, GE Leasing, SGI, PPFA, Brobeck, Bank of America, Clorox and Leapfrog. He can be reached at rschoenrank@computer.org.

Table of Contents

1. INTRODUCTION

When recording information in a database, we collect three distinct types of data

1. Data that is true for all time. For example, a person's birthdate.
2. Data that is true at a point in time (an event). For example, how much did you pay for that car?
3. Data that is true for an interval of time. For example, is your life insurance in force?

Dealing with the first two types of data is fairly straight forward, but processing time interval data correctly in a database is critical to avoiding the Type II processing complexity described by Kimball in "The Data Warehouse Toolkit, 2nd Edition: The Complete Guide to Dimensional Modeling".

The goal of time interval processing is to ensure that when the data is queried only one value of the data is returned for any point in time.

To process time interval data correctly, we need to understand that time interval is an abstract logical data type derived from the measurement of time. An abstract data type requires a definition and constraints, with functions and methods that operate on the data type. Since time interval is derived from time, we need to understand how we measure and use time. In reality, time appears to be continuous, without a beginning or an end. But, inside a computer, time is discrete, has an arbitrary lowest value and an arbitrary largest value.

1.1 Time Functions

The computer systems we use implement the universally agreed to definition of time. Time is used to record the occurrence of events. The computer also implements the functions we use to manipulate time. When using the time interval data type, we need the following additional functions for time:

- Beginning of time() – The smallest value of time in the computer's calendar
- End of time() – The largest value of time in the computer's calendar
- Today() – today's date in the computer's calendar
- Now() – the current value of time in the computer's calendar.

2. TIME INTERVAL DATA TYPE

As an abstract logical data type, a time interval consists of a pair of events [begin, end] where each event is a measurement of time. For a time interval:

- The time interval starts at begin
- The time interval finishes at end
- The time interval includes both begin and end

The constraints on a time interval are:

- begin <= end.
- Begin and end are expressed in the same units of measure.

The longest time interval is [beginning of time(), end of time()]. The shortest time interval has a length of 1. This is true regardless of the unit of measure that you choose for event

2.1 Functions

Time Interval Functions examine time intervals and events and give answers that are not time intervals.

- Equality(time interval1, time interval2),
    - answer is true or false
    - for true, time interval1.begin = time interval2.begin and
      time interval1.end = time interval2.end
- Contains(time interval, event),
    - answer is true or false
    - For true, event >= begin and event <= end
- Length(time interval)
    - Answer is an integer greater than zero
    - count of events (days, hour, minutes or seconds) in the time interval
- BeginEvent(time interval, event)
    - Answer is true or false
    - For true, event = begin

- EndEvent(time interval, event)
    - Answer is true or false
    - For true, event = end
- Contiguous(time interval1, time interval2)
    - Answer is true or false
    - For true, time interval1.end+1 = time interval2.begin

2.2 Methods

Time Interval Methods operate on time intervals and create new time intervals

Subdivide method – splits a time interval into two contiguous time intervals

- Subdivide(time interval1, date) = {time interval2, time interval3}
    - Input is a time interval and an event contained in the time interval
    - Output is a pair of time intervals
- The answer to Contains(time interval1, event) is true
- The output time intervals are contiguous
    - Time interval2.start = time interval1.start
    - Time interval2.end = event -1
    - time interval3.begin = event
    - time interval3.end = time interval1.end

Concatenate method – the inverse of subdivide

- Concatenate(time interval1, time interval2) = time interval3
    - Input is two contiguous time intervals
    - Output is a single time interval
- The input time intervals cannot overlap, the input time intervals are continuous
    - Time interval1.end +1 = time interval2.begin
- the output time interval is
    - Time interval3.begin = time interval1.begin
    - Time interval3.end = time interval2.end

3. TIME INTERVAL PROCESSING

Time intervals are used to describe properties of a data model entity that are valid over a period of time, such as the benefit history of an employee. When time intervals are implemented in a database it is as an attributive table to a parent table (data model entity).The attributive table contains a set of rows that hold the time interval data about an entity property which will answer questions such as for what period of time did the employee have a particular benefit?

3.1 Table Implementation

The goal of using a time interval table is to be able to determine the value of the other attributes on the time interval at any point in time.

The attributive time interval table is composed of the attributes:

- Parent table key
- Time interval
- Other attributes that describe the state of the parent entity during the time interval

Each row in the table describes a time interval for the attribute values. For time interval processing to apply to an attributive table, the time interval must be part of the table key.

3.2 Processing Assumptions

1. The set of time interval rows can be ordered by begin.
2. The set of time interval rows is continuous for all time
3. No two time interval rows will overlap
4. When a time interval is subdivided, a set of time intervals is created. The concatenation of this set of time intervals would create the original time interval.

3.3 Processing Conditions

There are three conditions used to choose which update process to use. The conditions are:

1. The number of existing time interval rows affected by the new time interval. The answer can be zero or more.

2. Does the beginning of the new time interval match the beginning of an existing time interval row?
3. Does the end of the new time interval match the end of an existing time interval row?

All the possible combinations of the answers the above questions are shown in Table 1.

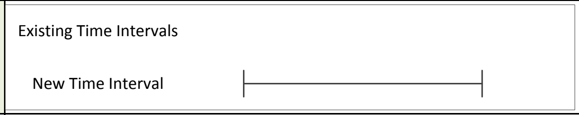
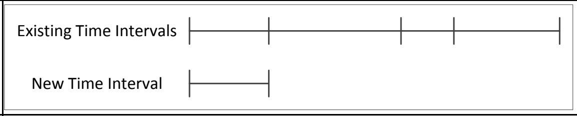
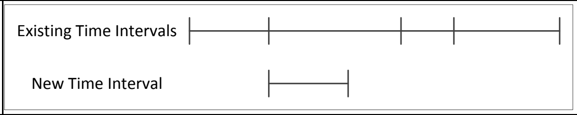
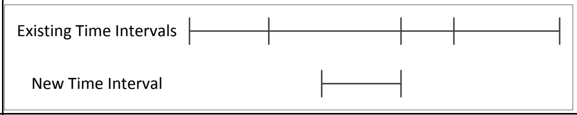
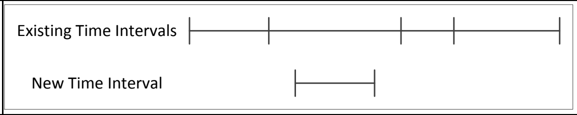
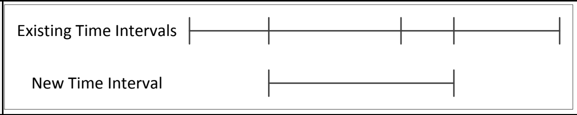
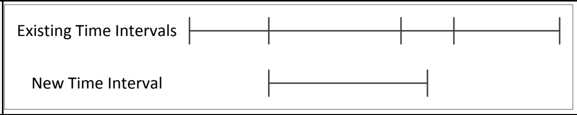
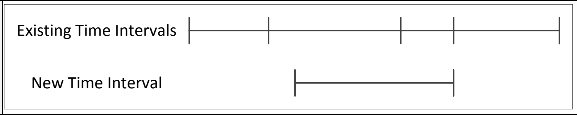
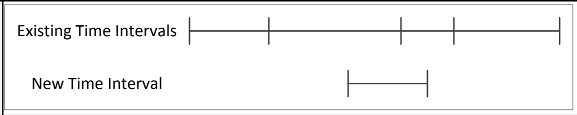
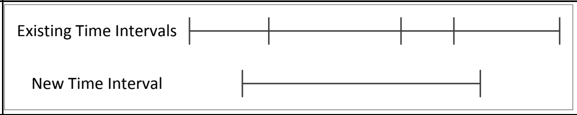
Case	Number of Time Intervals	new.begin = row.begin	new.end = row.end	Condition	Diagram
1	0			• The time interval row was not found	
2	1	TRUE	TRUE	• New time interval completely overlaps the existing time interval row	
3	1	TRUE	FALSE	• New time interval has same begin date as the existing time interval row	
4	1	FALSE	TRUE	• New time interval has same end date as the existing time interval row	
5	1	FALSE	FALSE	• New time interval overlaps center of the existing time interval row	
6	2	TRUE	TRUE	• New time interval completely overlaps two existing time interval rows	
7	2	TRUE	FALSE	• New time interval completely overlaps the first existing time interval row	
8	2	FALSE	TRUE	• New time interval completely overlaps the second existing time interval row	
9	2	FALSE	FALSE	• New time interval partly overlaps two existing time interval rows	
10	> 2			• New time interval overlaps more than two existing time interval rows	

Table 1. Time Interval Processing Conditions

3.4 Table Constraints

The constraints on the data contained in a time interval table are:

- A time interval is represented by a row in the table
- One time interval row will start at the beginning of time()
- One time interval row will end at the end of time()
- A time interval of length 1 cannot be subdivided

3.5 Processing Options

The time interval processing conditions restrict the normal database CRUD processes.

- The create process is Case 1 in the conditions table (Table 1).
- The read process is typical for a DBMS and is not described.
- The update process has to fulfill the requirements of Case 2 through 10 in the conditions table (Table 1).
- The delete process is not allowed because deleting a time interval row would create a gap in the set of time interval rows and we could not achieve our goal.

Described below are four different ways to implement time interval processing. Depending on your requirements, the processing for the time interval conditions can be modified to suit, but the ability to achieve the time interval processing goal should not be compromised.

3.5.1 Attribute Time Interval Processing

Treating the time interval as an attribute the changing of the time interval by:

- Only one time interval row exists in the table
- The time interval row is never subdivided
- For case 1, the time interval is created
- For case 2 through 5, the time interval is replaced

The process to update the time interval attribute is:

Case	Number of Time Intervals	new.begin = row.begin	new.end = row.end	Attribute Time Interval Processing
1	0			• insert new time interval row
2	1	TRUE	TRUE	• update the existing time interval row
3	1	TRUE	FALSE	• update the existing time interval row
4	1	FALSE	TRUE	• update the existing time interval row
5	1	FALSE	FALSE	• update the existing time interval row
6	2	TRUE	TRUE	• Processing not allowed • return error message
7	2	TRUE	FALSE	• Processing not allowed • return error message
8	2	FALSE	TRUE	• Processing not allowed • return error message
9	2	FALSE	FALSE	• Processing not allowed • return error message
10	> 2			• Processing not allowed • return error message

Table 2. Attribute Time Interval Create Update Processing

3.5.2 Current Time Interval Processing

The current time interval process restricts the changing of the time interval rows by:

- Only the current time interval can be subdivided, i.e., Contains(time interval, today()) is true
- No other time intervals are allowed to be subdivided, i.e., Contains(time interval, today()) is false
- The current time interval can only be subdivide at today(), i.e., only Subdivide(current time interval, today()) is allowed

The process to update the current time interval is:

Case	Number of Time Intervals	new.begin = row.begin	new.end = row.end	Current Time Interval Processing (new.begin = today()) (new.end = end of time())
1	0			• create beginning time interval row • insert new time interval row
2	1	TRUE	TRUE	• update the existing time interval row
3	1	TRUE	FALSE	• Processing not allowed • return error message
4	1	FALSE	TRUE	• adjust existing time interval row end date • insert new time interval row
5	1	FALSE	FALSE	• Processing not allowed • return error message
6	2	TRUE	TRUE	• Processing not allowed • return error message
7	2	TRUE	FALSE	• Processing not allowed • return error message
8	2	FALSE	TRUE	• Processing not allowed • return error message
9	2	FALSE	FALSE	• Processing not allowed • return error message
10	> 2			• Processing not allowed • return error message

Table 3. Current Time Interval Create Update Processing

3.5.3 Simple Time Interval Processing

In the simple update process, any time interval (past or future) row is allowed to be subdivided but

- Concatenating two time intervals is not allowed
- No existing time interval is allowed to be replaced

The process to for simple time interval updating is:

Case	Number of Time Intervals	new.begin = row.begin	new.end = row.end	Simple Time Interval Processing
1	0			• create beginning time interval row • insert new time interval row • create ending time interval row
2	1	TRUE	TRUE	• Not allowed to replace the existing time interval row • return error message
3	1	TRUE	FALSE	• adjust existing time interval row begin date • insert new time interval row
4	1	FALSE	TRUE	• adjust existing time interval row end date • insert new time interval row
5	1	FALSE	FALSE	• copy the existing time interval row • adjust the existing time interval row begin date • insert the existing time interval row with new end date • insert new time interval row
6	2	TRUE	TRUE	• Not allowed to replace the existing time interval row • return error message
7	2	TRUE	FALSE	• Not allowed to replace the existing time interval row • return error message
8	2	FALSE	TRUE	• Not allowed to replace the existing time interval row • return error message
9	2	FALSE	FALSE	• adjust existing time interval row1 with new end date • adjust existing time interval row2 with new begin date • insert new time interval record
10	> 2			• Processing not allowed • return error message

Table 4. Simple Time Interval Create Update Processing

3.5.4 General Time Interval Processing

In the general update process, any time interval (past or future) row is allowed to be subdivided or replaced, two adjacent time intervals may be subdivided or concatenated, but more than two adjacent intervals cannot be operated on.

The process to update the time interval in general is:

Case	Number of Time Intervals	new.begin = row.begin	new.end = row.end	General Time Interval Processing
1	0			• create beginning time interval row • insert new time interval row • create ending time interval row
2	1	TRUE	TRUE	• update the existing time interval row
3	1	TRUE	FALSE	• adjust existing time interval row begin date • insert new time interval row
4	1	FALSE	TRUE	• adjust existing time interval row end date • insert new time interval row
5	1	FALSE	FALSE	• copy the existing time interval row • adjust the existing time interval row begin date • insert the existing time interval row with new end date • insert new time interval row
6	2	TRUE	TRUE	• delete existing time interval row1 • delete existing time interval row2 • insert new time interval record
7	2	TRUE	FALSE	• delete existing time interval row1 • adjust existing time interval row2 with new begin date • insert new time interval record
8	2	FALSE	TRUE	• adjust existing time interval row1 with new end date • delete existing time interval row2 • insert new time interval record
9	2	FALSE	FALSE	• adjust existing time interval row1 with new end date • adjust existing time interval row2 with new begin date • insert new time interval record
10	> 2			• Processing not allowed • return error message

Table 5. General Time Interval Create Update Processing