A General Method For The Specification Of Time Frames For Chronologically Structured Databases.

1. Introduction.

One of the simplest ways in which to record real time information/events, processed by computer systems, is chronologically . Very simply, the information captured and processed is date/time stamped and appended to some file contained upon appropriate storage media (usually disc or magnetic tape) . In this way a chronologically (serially) ordered database can be constructed . The indexing of such databases using keys within records may also form part of the wider database schema, but for the purposes of this discussion these considerations are superfluous.

Such chronologically ordered databases are, by virtue of their nature, open ended and can potentially become very large It therefore follows that user requests for information from such databases will include a time scope (time frame) constraint which limits interest to a portion of the database.

A typical user request for information may be considered in three parts:

<time frame>
<selection criteria>
<report style>, where

<time frame> defines the portion of the database against which the <selection criteria> is to be applied, and
<selection criteria> defines the fields of interest. Note that if <selection criteria>=<NULL>, then all records within the <time frame> would be selected, and
<report style> defines the way in which results are to be presented, eg in tabular form, histogram, etc.

This last element, <report style>, is of no relevance for the discussions considered herein and is only identified here for completeness.

Example:
Q.1. "What events were recorded between 2100 hours on 12/12/86 and 2300 hours on 18/12/87 which involved burglaries in the area of Newtown?"

Clearly this user request contains both time frame and selection criteria . That is,

<time frame> is 2100 on 12/12/86 to 2300 on 18/12/86, and
<selection criteria> is burglaries in Newtown.

The mechanics necessary to carry out this database search would be

1. locate that part of the database which contains the first record falling within the specified time frame,
2. apply the selection criteria to the record . If a match is made then select the record, otherwise discard it,
3. read the next record . If the record still falls with the specified time frame repeat from step 2, otherwise stop.

This continuous time frame approach to searching has been used for many years and, for the most part, adequately copes with most user search requests . However, occasions do arise when a user will pose a question such as:

Q.2. "3What events were recorded between 2100 hours and 2300 hours on 12,13,14,15,16,17,18/12/86 which involved burglaries in the area of Newtown?"

Although this question is very similar to the one asked above, it is immediately clear that the request
does not cover single continuous period . The periods of interest are:

from 2100 hours on 12/12/86 to 2300 hours on 12/12/86, and
from 2100 hours on 13/12/86 to 2300 hours on 13/12/86, and
from 2100 hours on 14/12/86 to 2300 hours on 14/12/86, and
from 2100 hours on 15/12/86 to 2300 hours on 15/12/86, and
from 2100 hours on 16/12/86 to 2300 hours on 16/12/86, and
from 2100 hours on 17/12/86 to 2300 hours on 17/12/86, and
from 2100 hours on 18/12/86 to 2300 hours on 18/12/86

Now that the request is expanded it may be seen that the user has specified 7 separate continuous time frames, although his selection criteria is the same . Therefore, for this request to be serviced it would be necessary to consider the time frames individually and set up a run for each . This is not so much of a problem when the database to be searched resides on a fast access medium such as disc . However, where the database is contained on a slow medium such as magnetic tape then each run can take a substantial time to complete.

It would be a straight forward exercise to modify the format of the time frame structure so that it would be possible to specify a list of intervals . This, at first sight, seems like a sensible improvement to make. However, consider the following user request:

Q.3. "What is the breakdown of road traffic accidents in the Newtown area over each hour of the day from 1/12/86 to 31/12/8694?"

The user here is asking for a daily breakdown for every hour of the day for each day of December in 1986 . This equates to the following discrete time frames:

from 0000 hours on 01/12/86 to 0059 hours on 01/12/86, and
from 0100 hours on 01/12/86 to 0159 hours on 01/12/86, and
from 2300 hours on 31/12/86 to 2359 hours on 31/12/86

In total this represents 24x31 = 744 separate time frames! It is clear that this solution would prove very cumbersome and be prone to typographical errors.

This last example demonstrates that a more flexible and concise approach is needed if any worthwhile improvement is to be introduced . Before describing such an improved approach to the specification of time frame data it will be appropriate to identify three very important requirements:

1. the time frame syntax must be easy to understand,
2. the time frame syntax must be simply to use,
3. the syntax must allow complex user defined time frames to be specified and in a concise way.

2. Definition of a time frame syntax.

Consider the following definition:

<TIME FRAME LIST>::=<TIME FRAME>I<TIME FRAME LIST> <TIME FRAME>: :=<SD>KST ET>1,<FR>,<ED>, where

<SD> represents the start date,

<ST ET> represents the start time and end time respectively,
<FR> represents a frequency factor, and
<ED> represents the end date.

The <TIME FRAME LIST> definition should be interpreted as follows:

1. consider the first <TIME FRAME> from the <TIME FRAME LIST>,
2. select the starting date, <SD>, as the current date,
3. scan the database for records lying between the specified start time and end time, <ST ET>, for the current date,
4. when all such records have been considered increment the date by the given frequency, <FR>, and, If this new date lies on or before the end date, <ED>, repeat from step 3, otherwise continue from step 5,
5. if any further <TIME FRAME>s remain in the <TIME FRAME LIST> select the next one and repeat from step 2, otherwise terminate.

It will be seen that this structure will not only fulfil those requirements identified above but that it will also provide an additional benefit derived from the use of the <FR> (frequency) parameter.

Consider the example given in Q.3, above, where the user specification of the time frames extended to 744 separate intervals . Using the new syntax this will reduce to just 24 time frames:

```
01/12/86{0000 0059},1,31/12/86
01/12/86{0100 0159},1,31/12/86
                    :
                    :
01/12/98{2200 2259},1,31/12/86
01/12/86{2300 2359},1,1,31/12/86
```

Here the frequency <FR> parameter allows each day of December up to and including the 31st to be considered.

Example:
Q.4. "How many thefts from cars have taken place in Newtown between 1500 hours and 1630 hours on each Friday of March and April, 198794?"

This request would be coded as follows:

```
06/03/87{1500 1630},7,24/04/87
```

last Friday of April, 1987.

first Friday of March,1987.

The frequency parameter in this example is 7 (one week) thus enabling the dates for each successive Friday between the start date and end date to be automatically be calculated

The <TIME FRAME> definition described may also be used to specify continuous intervals ranging over two or more days:

Example:
Q.5. "Can I have a list of all events reported from 1030 hours on 22/01/85 to 2230 hours on 30/01/8594?"

The user here has requested a continuous time interval covering several days . The original way in which this would be specified would be:

0130 22/01/85 inclusive 2230 30/01/85 period.

Using the newly defined <TIME FRAME> representation this would equate to:

22/01/85{1030 2359},1,22/01/85
23/01/85{10000 2359},1,29/01/85
30/01/85{10000 2230},1,30/01/85

This example shows that special consideration must be taken of the start time and end time for the beginning and end of the interval.

2.1. Corollary:

The <TIME FRAME LIST>/<TIME FRAME> definition can be taken to represent the most general form for the specification of date/time intervals . It is possible, by relaxing the general syntax, to define subsets which will further improve its ease of use.

2.1.1. Derivative 1.

By omitting the final parameter, <ED>, thus:

<SD>{ST ET},<(FR>

the complete <TIME FRAME> can be inferred to be:

<SD>{<ST ET>},<FR>,<SD>+<FR>

That is: from <ST> to <ET> on <SD>, and from <ST> to <ET> on <SD>+<FR>.

2.1.2. Derivative 2.

By omitting the final two parameters, <FR>,<ED>, thus:
<SD>{ST ET>}

the complete <TIME FRAME> can be inferred to be:

<SD>{<ST ET>},0,<SD>

That is: from <ST> to <ET> on <SD>.

2.1.3. Derivative 3.

By specifying on the first parameter, <SD>, only, thus:

<SD>

the complete <TIME FRAME> cane be inferred to be:

<SD>{0000 2359},0,<SD>

That is: from 0000 hours to 2359 hours on <SD>.

2.1.4. Example.
Consider the following example:

```
<TIME FRAME LIST>::= 02/11/86
                     04/11/86{2100 2330}
                     08/11/86{0900 1454},2
                     23/11/86{1200 1945},1,26/11/86
```