編年時間参照系データの RDB 実装の試行

村尾 吉章*・清野 陽一 ・藤本 悠 ・玉置 三紀夫

Experimental RDB Implementation of Chronological Reference System Data

Yoshiaki MURAO*, Yoichi SEINO, Yu FUJIMOTO and Mikio TAMAKI

"Chronology" is a classification of time periods based on the transitions of characteristics, such as archaeological pottery types or dynastic changes. Historic eras are also a kind of chronology. The chronology can be regarded as an identifier that distinguishes periods in a specific region or under specific conditions. This characteristic of chronology makes it possible to use chronology as a measure of time axis. The authors have defined the Data Model for Chronological Reference System conformed with JIS X7108 "Temporal Schema" to enable the expression of temporal attributes of features using the name of chronology. In this paper, we introduce how to implement this model by using UDT and UDF of RDB, and show how to express temporal attributes of events, such as instants or periods, by using chronology at the implementation level. In addition, we have confirmed that it is possible to search for events expressed by chronology using various search conditions. In the future, when a large amount of feature information with temporal attributes is accumulated and archived, it will be necessary to classify them along the time axis for categorizing or understanding. This classification makes the chronology. It suggests that the chronology will also become important for the classification of mass information produced in modern societies, and its standardization will be meaningful at that time, too.

Keywords: 編年（Chronology），編年時間参照系（Chronological Reference System），地理情報標準 (the standards for geographic information), ISO 19108, 時間スキーマ(temporal schema)

１．はじめに

地物は空間と時間とを前提として存在している．時間軸を少し長い範囲で捉えた時，地物群がもつ特性の時間的な変化を分類しグループ化することが必要となる．そのようにして区別されたグループのことを「編年」と呼ぶ. 歴史上や考古学上の地物には，時間属性を西暦年で表現できないものも多数あり，編年が唯一の時間属性となる場合も少なくない．そこで筆者らは, ISO 19108「地理情報－時間スキーマ」に準拠した編年時間参照系モデル（奈良文化財研究所 2011，以降「本モデル」と呼ぶ）を開発し，地物のもつ時間属性に対して編年での表現を可能にしてきた.

これまで筆者らは，主に本モデルを理論的な検証や部分的な実証確認する研究を積み重ねてきたが，他方では具体的な実装の検討の必要性を感じていた. そこで本稿では，本モデルをリレーショナルデータベース（RDB）上に構築してシステム実装に近づけた検証を行った結果を報告し，その成果から見える本モデルの有用性を論じる.

２．編年時間参照系モデルの概要

地物の持つ時間属性の分類や多様な表現方法については ISO 19108 で規定されている. 本モデルは ISO 規格に準拠しつつその一部を拡張定義することによって編年を ISO 規格のもとで取扱うことを可能にしている.

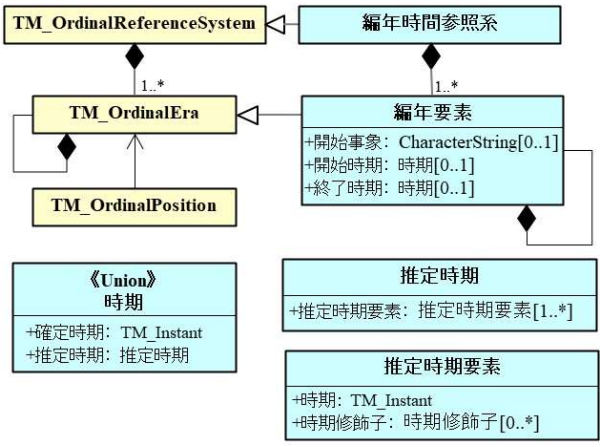

図 1 編年時間参照系モデル（一部）

* 正会員 ESRI ジャパン株式会社（ESRI Japan Corp.）
〒**532-0003** 大阪市淀川区宮原 2-14-14 Tel：**070-2493-[2050** E-mail：**yoshiaki_murao@esrij.com**

本モデルの中で，本稿において議論の対象としている部分を図 1 に示す.

図 1 で，TM_ の接頭辞をもつ英語表記の各クラスは ISO 19108 で規定されており，日本語表記の各クラスは本モデルで定義している.

編年要素クラスは，TM_OrdinalEra クラスを継承していることから，TM_Instant で時点を示す際に直接指定することができ，これにより地物の時間属性値として編年を利用可能にしている.

３．ＲＤＢ実装の概要

3.1. 基本方針

本稿では，本モデルを RDB 上で実装して基本的な検索操作が機能的に可能になることを確認するものであり，操作インターフェースにはこだわらないこととした.

DBMS としては PostgreSQL 13.2 を使用した.

本モデルを RDB 上に実装するためには，(1)本モデルの各クラスに対応したテーブル定義，(2)必要なユーザー定義型の作成，(3)ユーザー定義等を取り扱うためのユーザー定義関数の作成，が必要となる. (1)のテーブル設計は UML クラス図で実施し，後述する実験データを各テーブルに投入した．(2)および(3)におけるユーザー定義型やユーザー定義関数は，PL/pqSQL を利用して作成し，SQL の SELECT 文から関数を呼び出すことにより実装することとした．特に(3)では，実装におけるパフォーマンスは考慮しないため C 言語等での実装は行わず，開発・テストが柔軟に可能な PL/pgSQL を使用することとした.

また，テーブル名，ユーザー定義型名，ユーザー定義関数名には TC_の接頭辞を付けることとした.

実験データについては，筆者らが昨年の研究（村尾ら 2020)の際に利用した古代中国王朝の変遷情報をもとにして，本稿の実験用にデータを作成した.

3.2. 物理モデルとユーザー定義型

図 1 で示したクラス図は本モデルの論理モデルであり，RDB 実装のために，図 2 に示す物理モデルを導出し定義した.

図 2 で，TC_TRS は定義された編年時間参照系を保持するテーブルであり，キーとして編年時間参照系 ID（TRSID）を持つ．TC_Element は編年参照系内で定義された編年要素であり，TRSID と編年時間参照系内でユニークキーである編年要素 ID（CHID）との組み合わせによって識別される.

TC_Position は図 1 の論理モデルでは定義されなかったクラスだが，ISO 19108 で順序時間用に定義された TM_OdinalPosition を継承したユーザー定義型を編年表現用に定義することにより実装の利便性を図っている.

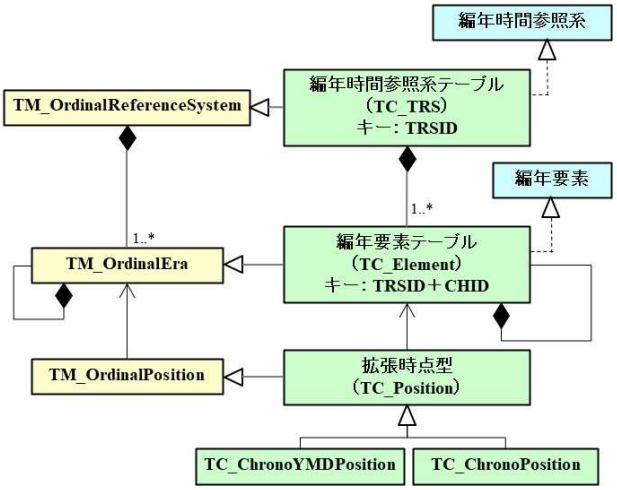

図 2 編年時間参照系（一部）の物理モデル

図 2 のクラス図では，TC_Position を継承する複数のクラスを定義しているが，実際に存在するのは TC_Position 型であり，その適用要件を明確化するために子クラスで表現している．その点の詳細は図 3 およびその説明で記述する.

図 3 では，図 2 の各クラスのもつ属性情報を詳細に記述している．なお，ここでは読者の理解を助けるために属性名は日本語で表記している.

図 3 で，TC_TRS（編年時間参照系テーブル）は，編年時間参照系そのものをテーブルレコードとして保持している．空間属性のフィールドはこの編年時間参照系が適用可能な空間範囲を示すものだが，今回の実装では時間属性に焦点を当てたため，空間属性値は設定していない.

TC_Element（編年要素テーブル）は編年時間参照系に含まれる個々の編年要素をテーブルレコードとして保持している.

編年要素には元号タイプのものがある．例えば，「令和 3 年 7 月 1 日」という時点を示す時，編年要素は「令和」であって，その編年要素内で最初の年を元年（1 年）として 3 年目がカウントされ，7 月 1 日が識別される．しかし，元号によっては元年で始まらないケースがあるため，属性値「元号開始年」が初年を何年とするかの情報を保持している．

TC_Position（拡張時点型）は，編年を用いて時点を表現するためのデータ型である．対象とする編年要素は，TRSID＋CHID により示すことができる．時点タイプは，図 2 で示した子クラスのいずれであるかを示している．TC_ChronoYMDPosition は先に述べた元号タイプにあたり，指定した編年要素にここで属性値として保持している年月日の情報をプラスして時点を示す．また，TC_ChronoPosition は元号タイプではない一般的な編年の取扱いであり，その際には年月日は設定不要である．

TC_Position クラスには，属性値「修飾子 ID」を定義している．これは編年に対する不確かな年代修飾子や曖昧な時間属性を設定可能にするための追加情報である．例えば，「古墳時代前期」という指定や「古墳時代前期～中期」などという表現である．不確かな年代修飾子および曖昧な時間属性については筆者らが過去に検討した手法（村尾ら 2014，2017）をもとにして，一部をここに実装しているが，本稿での議論の対象外なので説明は省略する．

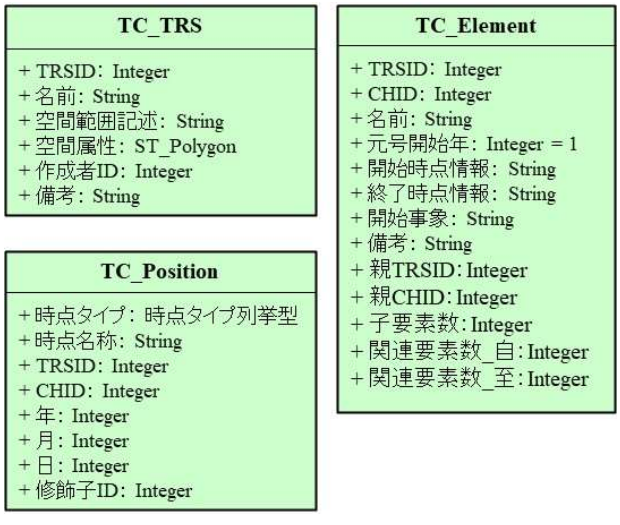

図 3 物理モデルの主要クラス

図 4 は，ISO 19108 で定義されている TM_Instant と TM_Period について，編年での表現の利便性を高めるため継承して定義したユーザー定義型である．これらも本稿の物理モデルにおいて定義している．

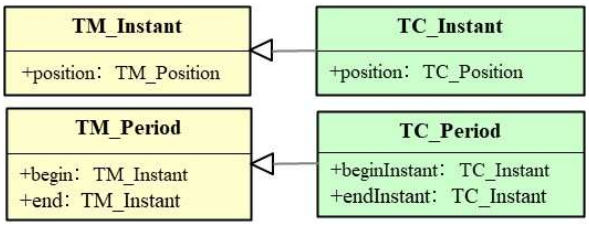

図 4 編年による Instant および Period の表現クラス

また，図 5 に示す事象データ（TC_TestEvent2）が存在する．これは，実験データとして作成し検索対象とする事象のデータをテーブルとして保持する．

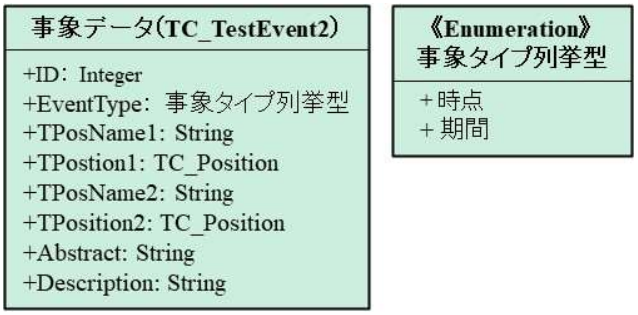

図 5 事象データ・クラス

図 5 で，EventType フィールドは，事象データが時点タイプか期間タイプかを示す．時点タイプの場合は TPosName1 フィールドにその時点の文字列表記を，TPosition1 フィールドに TC_Position 型で時点の情報を保持する．期間タイプの場合は TPosition1 と TPosition2 とで自至時点の情報を保持する．

事象データ・クラスについては，「3.5 実験データ」の節で再度説明する．

3.3. 編年インデックス値（CI 値）

編年には多様な定義がある．例えば，「江戸時代」を編年として定義し使用した場合，その開始年・終了年は西暦年で定義することができる．しかし，「弥生時代」の場合には，開始年も終了年も西暦年で定義することはできない．一方で，概ね縄文時代より

後であり，概ね古墳時代より前であると言える．

土器や瓦の型式で編年を定義した際にも同様のことが言えるが，このように順序のみ定義できる編年に対して先後関係を明確にするための指標値として編年インデックス値（CI 値と呼ぶ）を設定することとする．ある編年時間参照系に属する各編年要素の開始時期から終了時期までの期間を CI 値で数値化し，これをもとに他の編年要素との先後関係を解決する．

もし編年要素の開始年・終了年が，西暦での年や年月日，またユリウス日などで明らかな場合は，その値そのものやその値から算出した値を CI 値とするなど，さまざまな方法により CI 値を定義することができる．しかしながら，本稿の実装例では，CI 値の定義そのものが目的ではないので，単純な方法として，事象データの西暦年や西暦年月から，（年 x 10000 +月 x100 +日）を算出して CI 値とした．

3.4. ユーザー定義関数

今回の実装では多数のユーザー定義関数を作成した．主な関数の機能概要を以下に記述する．

(1) ユーザー定義型データ作成関数

表 1 に示す関数群は，SELECT 文中でユーザー定義型データを設定する際に使用できる．

表 1 ユーザー定義型データ作成関数群

関数名	作成するデータ型
TC_DefnieInstant	TC_Instant
TC_DefinePeriod	TC_Period
TC_DefineChronoPosition	TC_Position
TC_DefineChronoYMDPosition	TC_Position

表 1 で，TC_DefineChronoPosition 関数は，編年要素を指定することによる TC_Position の作成であり，例えば，魏王朝が存続した時期を指定することが可能となる．

また，TC_DefineChronoYMDPosition 関数は，編年要素として元号などを指定した際に，元号の年数や月日を指定することによる TC_Position データの作成を行う．例えば，魏王朝の元号である「景初二年六月」に対応した TC_Position データを SELECT 文中において設定することを可能にしている．

(2) ユーザー定義型データ文字列化関数群

表 2 に示す関数群は，ユーザー定義型データに対して SELECT 文による出力が可能となるようにする．

表 2 ユーザー定義型データ文字列化関数群

関数名	文字列化対象のデータ型
TC_PositionToString	TC_Position
TC_InstantToString	TC_Instant
TC_PeriodToString	TC_Period

(3) 編年の比較を行う関数群

表 3 に示す関数群は，編年で表現された時間属性値を比較して，その相対関係をもとに Allen が分類・定義した値（Allen 1983）を返す関数である．

表 3 編年による時間属性の比較関数群

関数名	比較対象データ
TC_ComparePositions	TC_Position どうし
TC_CompareEventByInstant	事象データと TC_Instant
TC_CompareEventByPeriod	事象データと TC_Period

表 3 で，「事象データ」とあるのは，実験データとして作成した事象データのことである．

この比較関数を用いることにより，例えば次の SELECT 文を使って，「魏の時代に発生した事象」を抽出して一覧を出力することが可能となる．

```
SELECT ID, TPosName1, Abstract, Description
FROM TC_TestEvent2 as T
WHERE TC_CompareEventByInstant (T.*,
            TC_DefineInstant(
                  TC_DefineChronoPosition(1000, 3, 0)) )
        = 'During' ;
```

上記 SELECT 文において，WHERE 句にある比較関数 TC_CompareEventByInstant の第 1 パラメータ

で，T.＊ とあるのは，検索対象の事象データ，第 2 パラメータの TC_DefineInstant 関数は, この SELECT 文中で TC_Instant を作成し, その内容が TRSID=1000, CHID=3 すなわち，魏王朝を示す編年要素であることを要求している．そして比較関数の結果として生成される相対関係が‘During’であるような事象データ，すなわち魏王朝の時点（編年として見た際の時点であり，実際には期間になる）で起こった事象を抽出することを可能にしている．

比較関数が Allen の分類値を返すことから，複雑な条件も SELECT 文の WHERE 句に書き込んで処理可能となる．次の SELECT 文は，「魏の景初 2 年から景初 4 年の間に発生した事象を抽出する」.

```
SELECT ID, TPosName1, Abstract, Description
FROM TC_TestEvent2 as T
WHERE (   TC_CompareEventByInstant(T.*,
          < 景初 2 年初の TC_Instant > ) = 'Equals'
 OR   TC_CompareEventByInstant(T.*,
          < 景初 2 年初の TC_Instant > ) = 'After'   )
AND ( TC_CompareEventByInstant(T.*,
          < 景初 4 年末の TC_Instant > ) = 'Equals'
 OR   TC_CompareEventByInstant(T.*,
          < 景初 4 年末の TC_Instant > ) = 'Before' ) ;
```

なお，ここでは WHERE 句の全体像が把握しやすいように，比較関数の第 2 パラメータは簡略化して記述している．

3.5. 実験データ

RDB 実装用データとしては, 編年時間参照系のデータ一式を準備する必要がある．そして，事前に定義された編年を用いて事象の時間属性を表現した実験データを用意する必要がある．

(1) 編年時間参照系データの準備

本稿での実装では，古代中国王朝の変遷と各王朝において制定された元号とを対象として編年時間参照系のデータを準備した．

まず，１つの編年時間参照系として「中国王朝編年」（TRSID=1000）を定義し，その配下に魏王朝が開かれた西暦 265 年から唐王朝が滅亡した西暦 907 年までの間に興った 63 の王朝を編年要素として定義した．例えば，魏（CHID=1），蜀（CHID=2），呉（CHID=3），隋（CHID=40），唐（CHID=53）などが定義されている．

そして,「中国王朝編年」の下位に各王朝における元号の編年時間参照系を定義し，それぞれの元号を編年要素として定義した. 例えば,「魏王朝元号編年」（TRSID=1001）では, 黄初（CHID＝1）, 太和（CHID=2）などが定義され,「蜀王朝元号編年」（TRSID=1002）では，章武（CHID=1），建興（CHID=2）などが定義されている．

今回準備した編年要素は開始年・終了年が西暦年で明確に設定でき，それらの値から CI 値を算出して時間軸上の相対関係を導いている．仮に設定した西暦年の値が学術的に正確な値でなくても，本稿における議論や考察には影響はない．

編年時間参照系の定義は，図 2 の編年時間参照系テーブルに 64 レコードが格納され，編年要素の定義は同図の編年要素テーブルに 461 レコードが格納された．

(2) 事象データの準備

事象データはアプリケーションデータであって，RDB 実装において検索条件に応じて適切なレコードが抽出されることを確認するためのデータである．編年を用いた時間属性を保持していることが必要だが，レコード形式は任意であることから，本稿では図 5 に示した事象データ・クラスの形式を定義し，そのデータを事象データテーブルに格納した．

事象データには時点タイプのデータと期間タイプのデータがあり，時点タイプのデータでは図 5 の TPosName1 フィールドに時間属性をテキスト形式で，TPosition1 フィールドには同じ時間属性値を TC_Position の形式で保持している．また，期間タイプのデータの場合は，時間属性として範囲を示す２つの時点データが必要となるため，さらに TPosName2 フィールドと TPosition2 フィールドを使用する．時間属性値としては編年による表現だけであり，西暦年の情報は保持していない．

RDB 実装のための事象データとして, 古代中国王

朝の建国年，滅亡年を「時点」の事象データとし，同時に各王朝の存続期間を「期間」の事象データとして作成した．これらは，編年時間参照系データを作成時に使用したデータである．加えて，１つの編年要素の中に含まれる状態の時点タイプの複数のデータと，複数の王朝（編年要素）をまたがって存在する期間タイプの複数のデータを別に準備しテーブルに格納した．それらの結果，事象データテーブルには 203 レコード格納された．

なお，後出する ID=9 のレコードは次のような内容となっている．TPosition1 および TPosition2 のデータは，図 3 で示した TC_Position の各属性項目をカンマ区切りデータとして保持している．

```
ID ： 9
EventType：期間
TPosName1：北魏の太延五年
TPosition1：(ChronologicalYMD,太延五年,1021,11,,5,,,,)
TPosName2：隋の開皇九年
TPosition2：(ChronologicalYMD,開皇九年,1040,1,,9,,,,)
Abstract：南北朝時代（北魏の華北統一から隋の統一まで）
Description： [記述なし]
```

４．ＲＤＢ実装による検索結果

本章では，RDB 実装の結果をテーブル別に見ていくこととする．各 SELECT 文は実装ケースであり，(a)以下の実装ケース番号を付記して参照可能にしている．

4.1. 編年時間参照系テーブル

(a) このテーブルに対して次の SELECT 文で検索することにより，定義されている編年時間参照系の一覧が表示でき，編年時間参照系名と対応する TRSID が分かる．

```
SELECT TRSID, Name FROM TC_TRS;
```

4.2. 編年要素テーブル

(b) このテーブルに対して，例えば次の SELECT 文で検索することにより，中国王朝編年に属する編年要素（実際には国名）の一覧が表示でき，各編年要素の自至年，存続期間などが分かる．

```
SELECT * FROM TC_TRS WHERE TRSID=1000;
```

(c) 次の SELECT 文で魏王朝における元号の一覧が表示できる．また，TRSID の指定により，どの王朝の元号でも一覧表示できる．

```
SELECT * FROM TC_TRS WHERE TRSID=1001;
```

(d) 次の SELECT 文で，２つの編年要素（魏と西晋）の相対関係を知ることができる．（結果は’Meets’）

```
SELECT TC_ComparePositions (
            TC_DefineChronoPosition(1000, 1, 0),
            TC_DefineChronoPosition(1000, 5, 0) );
```

(e) 次の SELECT 文により，唐の元号で「貞観」の次の元号を得ることができる．（結果は「永徽」）

```
SELECT * FROM TC_Element
  WHERE TRSID=1053 LIMIT 1
    OFFSET ( SELECT CHID from TC_Element
          WHERE TRSID=1053 AND Name='貞観' );
```

(f) 次の SELECT 文により，唐の元号で西暦 710 年にあたるものを表示できる．（同年に 2 回元号が変わっており，私年号もあることから，結果は「景龍」「唐隆」など 4 レコードが表示される．）

```
SELECT * FROM TC_Element as T
  WHERE TRSID=1053     AND (
TC_ComparePositions(
    TC_DefineChronoPosition(T.TRSID, T.CHID, 0),
    TC_DefineChronoYMDPosition(0, 0, 710, 0, 0, 0) )
 = 'Equals'     OR
TC_ComparePositions(
    TC_DefineChronoPosition(T.TRSID, T.CHID, 0),
    TC_DefineChronoYMDPosition(0, 0, 710, 0, 0, 0) )
  = 'Meets'     OR
TC_ComparePositions(
    TC_DefineChronoPosition(T.TRSID, T.CHID, 0),
    TC_DefineChronoYMDPosition(0, 0, 710, 0, 0, 0) )
  = 'MetBy'     OR
```

```
TC_ComparePositions(
    TC_DefineChronoPosition(T.TRSID, T.CHID, 0),
    TC_DefineChronoYMDPosition(0, 0, 710, 0, 0, 0) )
  = 'Begins'     OR
TC_ComparePositions(
    TC_DefineChronoPosition(T.TRSID, T.CHID, 0),
    TC_DefineChronoYMDPosition(0, 0, 710, 0, 0, 0) )
  = 'Ends'       OR
TC_ComparePositions(
    TC_DefineChronoPosition(T.TRSID, T.CHID, 0),
    TC_DefineChronoYMDPosition(0, 0, 710, 0, 0, 0) )
  = 'BegunBy'    OR
TC_ComparePositions(
    TC_DefineChronoPosition(T.TRSID, T.CHID, 0),
    TC_DefineChronoYMDPosition(0, 0, 710, 0, 0, 0) )
  = 'EndedBy' );
```

(g) このテーブルに対して次の SELECT 文で検索することにより，編年要素（東晋）の期間内に含まれる王朝名の編年要素を抽出することができる.（実験データから 13 レコード抽出した.）

```
SELECT * FROM TC_Element as T
WHERE   T.TRSID = 1000
 AND TC_ComparePositions(
          TC_DefineChronoPosition(1000, 9, 0),
          TC_DefineChronoPosition(T.TRSID, T.CHID, 0) )
       = 'Contains' ;
```

4.3. 事象データテーブル

(h) このテーブルがアプリケーションデータを保持しており，多様な検索条件を設定する対象として取り扱うべきデータセットとなる.

次の SELECT 文で検索することにより，魏の時代に発生した時点の事象を抽出することができる.（実験データから 11 レコード抽出した.）

```
SELECT ID, TPosName1, Abstract, Description
FROM TC_TestEvent2 as T
WHERE EventType = '時点'
   AND TC_CompareEventByInstant(T.*,
            TC_DefineInstant(
                TC_DefineChronoPosition(1000, 3, 0)) )
        = 'During';
```

(i) 次の SELECT 文で検索することにより，魏の元号「景初」の期間に発生した時点の事象を抽出することができる.（実験データから 3 レコード抽出した.）

```
SELECT ID, TPosName1, Abstract, Description
FROM TC_TestEvent2 as T
WHERE EventType = '時点'
   AND TC_CompareEventByInstant(T.*,
            TC_DefineInstant(
                TC_DefineChronoPosition(1000, 3, 0)) )
        = 'During';
```

(j) 次の SELECT 文で検索することにより，魏の「景初二年」に発生した時点の事象を抽出することができる.（実験データから 2 レコード抽出した.）

```
SELECT ID, TPosName1, Abstract, Description
FROM TC_TestEvent2 as T
WHERE EventType = '時点'
   AND TC_CompareEventByInstant(T.*,
            TC_DefineInstant(
                TC_DefineChronoYMDPosition
                                (1001, 4, 2, 1, 1, 0)) )
        = 'After'
   AND TC_CompareEventByInstant(T.*,
            TC_DefineInstant(
                TC_DefineChronoYMDPosition
                                (1001, 4, 2, 12, 31, 0)) )
        = 'Before' ;
```

(k) 次の SELECT 文では複数の編年要素にまたがった期間を指定し，魏の「正始四年」から成漢が建国されるまでの期間に発生した時点の事象を検索することができる.（実験データから 7 レコード抽出した.）

```
SELECT ID, TPosName1, Abstract, Description
FROM TC_TestEvent2 as T
WHERE EventType = '時点'
 AND ( TC_CompareEventByInstant(T.*,
          TC_DefineInstant(
          TC_DefineChronoYMDPosition
                              (1001, 5, 4, 0, 0, 0)) )
    = 'Equals'
 OR TC_CompareEventByInstant(T.*,
          TC_DefineInstant(
               TC_DefineChronoYMDPosition
                              (1001, 5, 4, 0, 0, 0)) )
    = 'After' )
 AND TC_CompareEventByInstant(T.*,
           TC_DefineInstant(
                TC_DefineChronoPosition (1000, 6, 5001)) )
    = 'Before' ;
```

(l) 次の SELECT 文で検索することにより，魏の時代を包含する期間の事象を抽出することができる.（実験データから 3 レコード抽出した.）

```
SELECT ID,TPosName1,TPosName2,Abstract,Description
FROM TC_TestEvent2 as T
WHERE EventType = '期間'
 AND ( TC_CompareEventByInstant(T.*,
          TC_DefineInstant(
              TC_DefineChronoPosition(1000, 3, 0)) )
       = 'Contains'
  OR   TC_CompareEventByInstant(T.*,
          TC_DefineInstant(
              TC_DefineChronoPosition(1000, 3, 0)) )
       = 'Begins'
  OR   TC_CompareEventByInstant(T.*,
          TC_DefineInstant(
              TC_DefineChronoPosition(1000, 3, 0)) )
       = 'BegunBy'
  OR   TC_CompareEventByInstant(T.*,
          TC_DefineInstant(
              TC_DefineChronoPosition(1000, 3, 0)) )
       = 'Ends'
  OR   TC_CompareEventByInstant(T.*,
          TC_DefineInstant(
              TC_DefineChronoPosition(1000, 3, 0)) )
       = 'EndedBy' );
```

(m) 次の SELECT 文で検索することにより，魏の建国から呉の滅亡までの期間に発生した時点の事象を抽出することができる．（実験データから 13 レコード抽出した．）

```
SELECT ID,TPosName1,TPosName2,Abstract,Description
FROM TC_TestEvent2 as T
WHERE EventType = '時点'
 AND TC_CompareEventByPeriod(T.*,
        TC_DefinePeriod(
            TC_DefineInstant(
                TC_DefineChronoPosition
                                    (1000, 1, 5001)),
            TC_DefineInstant(
                TC_DefineChronoPosition
                                    (1000, 3, 5002)) ) )
      = 'During';
```

(n) 次の SELECT 文で検索することにより，魏の建国から呉の滅亡まで（三国時代）の期間に発生した時点の事象を抽出することができる．(実験データから 13 レコード抽出した．)

```
SELECT ID,TPosName1,TPosName2,Abstract,Description
FROM TC_TestEvent2 as T
WHERE EventType = '時点'
 AND TC_CompareEventByPeriod(T.*,
        TC_DefinePeriod(
            TC_DefineInstant(
                TC_DefineChronoPosition
                                    (1000, 1, 5001)),
            TC_DefineInstant(
                TC_DefineChronoPosition
                                    (1000, 3, 5002)) ) )
      = 'During';
```

(o) 次の SELECT 文で検索することにより，西晋の建国から東晋の滅亡まで（五胡十六国時代）の期間に発生したすべての事象を抽出することができる．（実験データから 60 レコード抽出した．）

```
SELECT ID,TPosName1,TPosName2,Abstract,Description
FROM TC_TestEvent2 as T
WHERE   TC_CompareEventByPeriod(T.*,
            TC_DefinePeriod(
                TC_DefineInstant(
                    TC_DefineChronoPosition
                                    (1000, 5, 5001)),
                TC_DefineInstant(
                    TC_DefineChronoPosition
                                    (1000, 9, 5002)) ) )
          = 'During';
```

(p) 次の SELECT 文のようにサブクエリを使って検索することにより，事象データの ID=9 のレコード（**3.5** (2) にて例示したデータ）が示す期間（北魏の華北統一から隋の統一までの間の南北朝時代）に発生したすべての事象を抽出することができる．(実験データから 8 レコード抽出した．)

```
SELECT ID,TPosName1,TPosName2,Abstract,Description
FROM TC_TestEvent2 as T
WHERE   TC_CompareEventByPeriod(T.*,
            TC_DefinePeriod(
                TC_DefineInstant(
                    (SELECT TPosition1
                        from TC_TestEvent2
                        WHERE ID=9) ),
                TC_DefineInstant(
                    (SELECT TPosition1
                        from TC_TestEvent2
                        WHERE ID=9) ), )
          = 'During';
```

５．ＲＤＢ実装に対する考察

考察 1：検索条件の網羅性

前章に示したように，編年を用いて表現された時間属性に対して多様な条件による検索を実施することができた．実装した検索条件は表 4 のように整理することができ，今回の RDB 実装によって必要な検索条件パターンを網羅して検証できたことが確認できる．

表 4 検索条件の分類と実装したケース

検索条件の分類	ケース
編年時間参照系についての検索	
定義されている編年の確認	
編年の一覧を表示する	a b c
編年の設定内容を確認する	b など
既知の編年の次の編年を知る	e
編年要素どうしの相対関係を知る	d
ある時点との相対関係で抽出する	f
ある期間との相対関係で抽出する	g
事実データについての検索	
テーブル内の事実データの確認	
事実データの一覧を表示する	h
事実データの内容を確認する	h など
事実データどうしの関係で抽出する	p
ある時点との相対関係で抽出する	j
ある期間との相対関係で抽出する	i k l m n o

考察 2：検索条件設定の自由度確保

前章に記述した SELECT 文は一例であって，目的に応じてさらに多様な検索が可能である．この検索条件の自由度を生み出しているのは，本稿で作成したユーザー定義関数が積み木細工のように組み合わせ可能であったこと，PL/pgSQL が輻輳してコーディングされたユーザー定義関数を正しく処理してくれたことによる．

今回はデータ量が少ないためパフォーマンスを考慮する必要がなかった点で PL/pqSQL で実装できたが，本格的に実装する際には C 言語などで作成する必要があるものと考えられる．

考察 3：ユーザーインターフェースの重要性

実装ケース(f)をはじめとして，時間属性の相対関係を検索条件として設定することが少なくないが，その際に厳密に条件分類された Allen の分類を用いたことによって条件設定が複雑化したことは否めない．また，特定の編年を指定しようとした際，ユーザー定義関数のパラメータを暗号のように割り当てる必要があった．それらの点については，ユーザーインターフェースを充実させ，複雑な条件設定はプログラム内に埋め込むことが必須である．

考察 4：CI 値の割り当て方法

CI 値は，実装の前面には出現しなかったものの，システム内部において，編年どうしの比較を行う上でこの値が非常に重要な意味を持っている．本稿での RDB 実装では，3.3 節に記述した方法で CI 値を割り当て，これをもとに多様な検索が可能となった．

しかし，今回の割り当て方法が妥当か否かについては本稿の段階ではまだ評価できる状況にはない．この CI 値をどのように割り当てるべきかについては，事案データの特性が大きくかかわってくると想定できることから，今後，多様なデータで事例を積み重ねることが必要と考えられる．

考察 5：編年による時間表現の実用性

現在の GIS が管理する地物の大半は現時点で存在しているものを表現したものである．しかし今後は時間軸をもって情報が蓄積されていくことになり，地物の時間属性も重要性を増すこととなる．

情報蓄積の初期の段階は西暦年月日を属性値として持てば履歴データの管理としては可能だが，情報量が多くなり，情報を蓄積する期間がだんだん長くなると，時間軸での情報分類が必要となってくる．

日本経済を見た際にも，戦後混乱期，高度成長期，バブル期，低成長期などと分類することになる．この時間軸での分類は，すなわち編年に他ならない．

空間属性で，緯度と経度で位置を表すだけではなく，都道府県，市町村，あるいは別に定義された地域など，空間属性に対する分類を行っているのと同様に，時間属性にとっても分類が必要となる．そして，分類したカテゴリー単位で大括りに整理して，その変化を論じることとなる．その分類単位が即ち編年となることから，編年により時間属性を表現することは将来的には間違いなく実用的なものとなる．

考察 6：編年による時間属性と空間属性との組み合わせの実用性

本稿では，古代中国王朝を議論の対象として，編

年で表現された時間属性だけに着目して議論を進めてきた．しかし実際には，各王朝は，その勢力範囲が時期に応じて変化している．

他にも時間と共に空間属性が変化する地物も多数存在する．市町村合併により一方の市に吸収された場合，市名は変化しないで地域範囲は変化する．都道府県についても，過去に遡れば，都道府県の範囲も名称も時間と共に変化している．国内の旧国名も同様である．また，企業の支社・営業所の管轄区域も，選挙区の区割りも，時間と共に変化する．

上記の例は，空間属性が面的な地物だったが，線的なものでは，例えば国道は同じ番号でルートが変化する場合がある．東京－名古屋間の鉄道ルートは，東海道線によるルート，新幹線によるルート，また将来はリニア新幹線によるルートが存在するようになる．

点的な空間属性では，小学校の廃校に伴う別用途での利用ケースや，ネーミングライツの導入などで同じ建物が別の名前になったり，企業の本社所在地が変化したりする．

このように，時間属性に伴って空間属性の変化する地物は多数あるが，その空間属性の変化のタイミングは，時間属性において年月日のタイムスタンプが押されると共に，地物の意味づけが変化することが多い．地物の意味づけの変化を情報分類の尺度として取り扱う必要が生じた際に，編年を利用することが可能となる．そしてその際には，ISO に準拠した本モデルを使って編年を表現し取扱うことの意義が生まれることとなる．

６．まとめ

前章の考察 1 で述べたように，本稿による RDB 実装方法により，編年を用いた時間属性表現に対して SQL 文による柔軟な検索が可能となることが確認でき，本モデルの有効性を実装面でも検証することができた．

また，考察 5，考察 6 に示したように，今後，さまざまな情報が時間経過とともに蓄積されたとき，時間軸の変化に対して時代区分が当たり前のようになされていき，その時代区分に沿って地物を含む各種情報が分類されていくことになるだろうと想像できる．その際の時代区分は編年そのものであり，編年を用いた時間属性管理の必要性が具体化するものと考えられる．

筆者らが編年についての研究を始めたきっかけは．考古学における時間属性の尺度が土器型式などによる編年にあるので，それをいかに国際規格のもとで表現するか，という議論であった．しかし，編年を深く検討していくうちに，その対象は現代のデータに対する表現に昇華しつつある．まさに，温故知新と言えるのではないだろうか．

謝辞

本研究は JSPS 科研費 JP16K21715 の助成を受けたものである．

参考文献

Allen, J. F. (1983)” Maintaining Knowledge about Temporal Intervals”, Communications of the ACM, Nov. 1983, Vol. 26, No. 11, pp.832-843.

ISO 19108（2002）“Geographic Information－Temporal Schema”,［JIS X7108（2004）「地理情報－時間スキーマ」が翻訳版として対応］.

奈良文化財研究所（2011）:『遺構情報モデルに基づく地理空間データ作成のための製品仕様書』,「埋蔵文化財ニュース」, 144, 奈良文化財研究所.

村尾吉章，碓井照子，森本晋，清水啓治，藤本悠，清野陽一，玉置三紀夫（2014）: 遺構情報モデルに基づいた不確かな時間属性の適用,「地理情報システム学会講演論文集 2014」.

村尾吉章，森本晋，藤本悠，清野陽一，玉置三紀夫（2017）: 編年時間参照系モデルによる曖昧な時間属性に対する問合せ方式の実装,「地理情報システム学会講演論文集 2017」.

村尾吉章，清野陽一，藤本悠，玉置三紀夫（2020）: 地物の時間位相とその利用について,「地理情報システム学会講演論文集 2020」.