LODのOLAP分析を可能にするETLフレームワークの提案

井上　寛之† 　天笠　俊之†,††† 　北川　博之††

† 筑波大学大学院システム情報工学研究科コンピュータサイエンス専攻　〒305–8577 茨城県つくば市天王台 1-1-1

†† 筑波大学システム情報系情報工学域　〒305–8573 茨城県つくば市天王台 1–1–1

††† 宇宙航空研究開発機構宇宙科学研究所宇宙科学情報解析研究系　〒252–5210 神奈川県相模原市中央区由野台 3-1-1

E-mail: †inohiro@kde.cs.tsukuba.ac.jp, ††{amagasa,kitagawa}@cs.tsukuba.ac.jp

あらまし　コンピュータ処理に適した形式によるデータの公開は年々増加しており，その一つの方法として Linked Open Data (LOD) が挙げられる．LOD には，数値およびテキストが含まれているデータも多く存在し，これらのデータに対する既存の OLAP システムを用いた分析処理が強く求められている．本稿では，LOD を OLAP 分析のための多次元モデルに変換する ETL フレームワークを提案する．我々が提案するフレームワークは他の関連する手法と異なり，変換の為の特定の RDF 語彙の利用を必要としない．この代わりに，分析対象とするデータセット内のリソース間の関係，内存する階層構造，および外部のリソースを用いて多次元モデルとその概念階層を導き出す．本手法は OLAP 分析のための特定の RDF 語彙を用いた LOD のみならず，より多くの一般的な LOD に対する分析処理を可能にするものである．実験として既存のデータセットに対して本手法を適用した例を示し，その有効性について述べる．

キーワード　Linked Open Data, ETL, OLAP

Hiroyuki INOUE†, Toshiyuki AMAGASA††,†††, and Hiroyuki KITAGAWA††

† Graduate School of Systems and Information Engineering, University of Tsukuba
1-1-1 Tennodai, Tsukuba, Ibaraki, 305–8577 Japan

†† Faculty of Engineering, Information and Systems, University of Tsukuba
1–1–1 Tennodai, Tsukuba, Ibaraki, 305–8573, Japan

††† Institute of Space and Astronautical Science, Japan Aerospace Exploration Agency.
3-1-1 Yoshinodai, Chuo-ku, Sagamihara, Kanagawa 252-5210, Japan

E-mail: †inohiro@kde.cs.tsukuba.ac.jp, ††{amagasa,kitagawa}@cs.tsukuba.ac.jp

1. はじめに

近年，セマンティックウェブから発展した Linked Data [3] と呼ばれる取り組みが注目されている．そもそもセマンティックウェブとは，Web 上のリソース（文書，画像）などに対してメタデータを付与することで，コンピュータがリソースの意味を理解し，データの取得や分析などに活用する取り組みである．セマンティックウェブにおいてメタデータを記述する枠組みとして，Resource Description Framework（RDF） [4] が規格化されている．RDF はリソース自身，及びリソースと他のリソースの関係を記述することができる．RDF において，リソースは URI（Uniform Resource Identifirer，統一資源識別子）で一意に識別される．このセマンティックウェブの対象を生データ（Raw Data）まで拡張した取り組みが，Linked Data である．

Linked Data は，構造化されたデータの公開方法の一つであり，他の Linked Data として公開されたデータとリンクすることで,「データの Web」を構築し，新たな価値を創りだすのが狙いである．この取り組みは米国や EU 諸国を先駆けとした Open Data(注1) と呼ばれる取り組みによって，更に加速されている．Open Data は，政府や研究機関などがデータを積極的に公開することで始まったが，それらのデータを RDF で記述し，相互にリンクすることで Linked Open Data (LOD) として広まりつつある．

LOD として公開されているデータセットの多くは，テキストおよび数値で構成される．[12] は，EU における統計情報を

（注1）：Open Definition. http://opendefinition.org/

公開している Eurostat (注2) のデータを LOD として公開したものである．また，Linked Sensor Data [14] は，米国における約一千箇所の気象観測所のメタデータ，およびそれらのセンサで観測された気温，湿度，気圧などの気象観測データをハリケーン毎にまとめ，LOD として公開されているものである．これらの統計情報や，センサから取得された大量の数値データなどが LOD として公開され始めている．従って，それらのデータの分析が必要とされていると考えられる．

ところで，ビジネスにおける商品の売上など，数値，統計データの分析には以前から On-Line Analytical Processing（OLAP）システムが広く利用されてきた．OLAP はデータウェアハウス（DWH）などに大量に蓄積されたデータに対して，統計的な分析処理を行う手法の一つであり，分析対象の数値データ（事実）と，それに関連付けられた階層をもつ属性（次元）を利用して，目的の分析を行う．

LOD データセットに OLAP 分析を適用する研究はいくつか存在する [7,8,13]．一つの手法は，LOD データセットを関係表に格納し，既存の OLAP システムによる分析を実現した [13]．他の手法では，LOD データセットの RDF リソース集合に対して，直接 OLAP 的な分析を実現している [7]．両者を比較すると，前者のアプローチは既存の OLAP システムが存在している点や性能の面から優位であると言える．

LOD データセットに対して，データの抽出（Extract），変換（Transform），OLAP システムへのロード（Load）処理（ETL [10]）を適用する場合，いくつかの技術的な問題点が考えられる: 1）事実表および次元表を考慮して，分析のためのスター（もしくはスノーフレーク）スキーマを設計する必要がある．2）次元表と関連付けられた概念階層を抽出する必要がある．また，3）データセットから URL を用いて参照可能な外部リソースをどのように扱うか，である．

これらは容易に解決可能な問題では無い．例えば，Kampgen ら [13] は，OLAP で用いられるデータキューブを RDF で記述するための RDF 語彙 [9] である RDF Data Cube Vocabulary（QB）[6] を用いて記述された LOD データセットに対して，既存の OLAP システムによる OLAP 分析を可能にした．しかし QB を用いて記述されていない LOD データセットに対してこの手法を適用するのは困難である．

本稿では，我々は LOD データセットに対する一般的な ETL 処理を提案する．基本的な特徴として，1）QB 等の特定の RDF 語彙の利用に関わらず適用可能であり，2）データセットに内存する概念階層，およびリンクする GeoNames [17] や DBpedia [1] 等の外部リソースを利用して，分析に必要な次元を導出する．

本稿の構成は以下のとおりである．2. 節では本論文に必要な前提知識について解説する．3. 節では，提案フレームワークである LOD データセットの OLAP 分析を可能にする ETL フレームワークについて説明する．4. 節では，本フレームワークをいくつかの既存の LOD データセットに適用した例を挙げ，その結果に対する考察を述べる．5. 節で関連研究について説明し，最後に 6. 節でまとめと今後の課題について述べる．

2. 前提知識

2.1 RDF と LOD

RDF [4] は，Web 上のリソース自身と，他のリソースとの関係を記述する枠組み（データモデル）である．あるリソースの一つのメタデータを，主語（Subject），述語（Predicate），目的語（Object）を用いて記述する．主語はメタデータを記述する対象のリソース，述語は主語に関する情報のプロパティもしくは特徴を定義し，目的語は述語の対象となる値を格納する．これらは三つ組でトリプルと呼ばれる．主語および述語は Universal Resource Identifier（URI，統一資源識別子）で，目的語は URI もしくはリテラルで記述しその値のデータ型を付与することも可能である．

LOD [2] は，構造化されたデータを Web 上で公開，共有する一つの方法である．またセマンティック・ウェブの領域で,「データの Web」を構築する運動そのものを指す．LOD は標準化された技術を基に構築されたデータセットの集合である．データとメタデータの記述には RDF や URI を，他のデータとのリンクには HTTP を，データセットに対する問合せには SPARQL [16] などの RDF 問合せ言語を用いて行う．

2.2 OLAP

OLAP [5, 10] とは，DWH などに大量に蓄積されたデータに対して多次元分析を行う手法である．分析対象となる数値データが格納された事実表と，事実表に関連付けられた階層を持つ属性を次元表としてキューブと呼ばれるデータモデルを構成する．このキューブに対して，ロールアップ，ドリルダウン，スライシングなどの操作を適用することによって，主に集約分析を行う．製品の売上げデータなど，場所や時刻といった様々なの属性を持つデータの分析に適しているため，主にビジネスインテリジェンスの分野で利用されている．

2.2.1 ETL

OLAP 分析のためのデータを DWH に格納するために，一つまたは複数の情報源からデータを抽出（Extract）する必要がある．またスターもしくはスノーフレークスキーマを構成するためにデータの変換，加工（Transform）が必要である．最終的に変換したデータを DWH に読み込む（Load）．これらの一連の過程は一般的に *ETL*（*Extract, Transform, and Load*）[10] と呼ばれている．ETL では OLAP 分析のために幾つかの操作が行われる [15]．例えばデータの欠損値の補完やノイズ等の除去などが挙げられる．

本研究で提案するフレームワークは，対象のデータが関係データではない点で一般的な ETL 処理と異なる．我々の目的は対象の LOD データセットから OLAP 分析に用いるスタースキーマを導出する系統的な方法を提案することである．分析対象のスキーマを固定することで，データ展開処理を自動化する．さらに，多次元分析に必要な概念階層をデータセットから導出する．

(注2)：Eurostat. http://ec.europa.eu/eurostat/

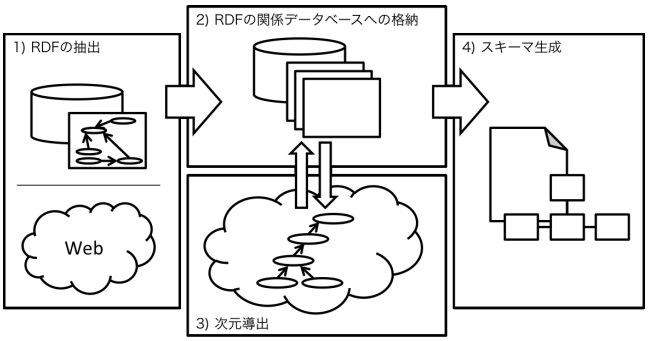


図 1: 提案フレームワーク

3. 提案フレームワーク

本節では提案フレームワークについて説明する．

3.1 処理の流れ

図 1 は提案フレームワークにおける処理の流れを示している．まずユーザーは分析対象の LOD を指定し，１）フレームワークは分析に必要なデータを抽出し（3.2 節），２）関係データベースに格納する（3.3 節）．次に，フレームワークはユーザーに対して分析対象の数値が含まれているレコードの指定を求め，３）指定されたレコードを含む表を事実表としたときの次元表を周辺の表から導出する（3.4 節）．事実表および次元表が揃ったところで，４）OLAP 分析に利用するスタースキーマを生成する（3.5 節）．また，3.6 節では，本フレームワークのプロトタイプ実装について述べる．

3.2 RDF データの抽出

フレームワークは，ユーザーにより示された分析対象の数値が含まれる LOD データセットを URI をもとに取得する．このとき，データセットの取得方法としては，おおまかに二つの方法が考えられる: １）ローカルもしくはリモートに存在するダンプされたデータセットを用いる，もしくは２）RDF ストアの SPARQL エンドポイントを用いてデータを取得する．１については容易に実行することができると考えられるが，２については取得する RDF トリプルの終端条件を設定する必要がある．本研究では，主語が分析対象のデータセットと同一ホスト名の URI で記述される RDF トリプルに限って取得する．

3.3 RDF データの格納

抽出された RDF データは，一つの主語に結びつく述語が関係表の属性に対応する，述語表（Property Table） [18, 19] アプローチで関係データベース（RDB）に格納される．述語表を利用する理由として: １）述語表と関係スキーマが直接対応する，２）表同士の関係が簡単に把握することができる，等が挙げられる．

述語表は，あるリソースについて，リソースのタイプ（`rdf:type`(注3) ）ごとに，リソースの述語を属性とした関係表を作成し格納する RDF 格納手法で，ある同一の主語を持つ RDF トリプルを一行で表現する方法である．ただし，データを辿る段階では，全ての属性を列挙することが出来ないため，関係表のスキーマを決定することが出来ない．そこで，`rdf:type` ごとに主語，述語，目的語の組みと目的語のデータ型をタプルとした Type-partitioned Triple Store（TPTS）アプローチで，一度すべての RDF トリプルを RDB へ格納する．すべてのリソースを参照し RDB に格納し終えると，各表の属性を把握することができるため，`rdf:type` 毎の関係表を述語表として作りなおす．

(注3)：`http://www.w3.org/1999/02/22-rdf-syntax-ns#type`

表 1: Triple Store 形式.

Subject	Predicate	Object
SR_1	rdf:type	SalesResult
SR_1	store	Store_1
SR_1	category	Cat_1
SR_1	price	`"150000"^^xsd:integer`
SR_1	time	Time_1
Cat_1	rdf:type	Category
Cat_1	name	“Mac”
Cat_1	category	Cat_2
Cat_2	rdf:type	Category
Cat_2	name	“Computer”
Store_1	rdf:type	Store
Store_1	located_at	gn:7452809
Store_1	branch	“Tsukuba”
Store_1	tel	“029-800-000”
SR_2	rdf:type	SalesResult
SR_2	store	Store_2
SR_2	price	`"4000"^^xsd:integer`
Store_2	rdf:type	Store
Store_2	located_at	gn:2111901
Store_2	branch	“Mito”
Time_1	rdf:type	Instant
Time_1	inXSDDateTime	`"2012-12-01T13:15:00"^^xsd:dateTime`

手法の説明に利用する RDF リソース集合を表 1 に示す．この例では，製品の売上について考え，一つの売上実績（SalesResult）は売上店舗（Store），売上時刻（Instant），製品カテゴリ（Category）で記述される．表 1 の RDF リソースを `rdf:type` 毎に分割して格納した結果が表 2 である．TPTS で格納し終えると，それぞれの `rdf:type` がどのような述語（属性）を持つか把握することができる．具体的には，`Predicate` 列に対して，`DISTINCT` クエリを実行することで得ることができる．この情報を基にそれぞれのリソースを一行で格納する述語表に格納し直す（表 3）．

それぞれの表におけるリソースの関係を調べることで，他の表と関係が存在するか判定することができる．このようにして用意された表のうち，数値を含む表を事実表の候補とする．ユーザーによる分析対象の数値（メジャー）の選択を受け，数値を含む選択された表を事実表とする．

3.3.1 外部キーの濃度推定

スタースキーマを生成するために，外部キーの濃度（1:1, 1:N，N:1，もしくは M:N）を保持する必要がある．しかし濃度は明示的に与えられるものではないため，我々は関係表におけるレコード（RDF インスタンス）から濃度の推定を行う．それぞれの外部キーについて，他の表からの参照を調べることで，濃度を推定することができる．この情報もフレームワークの中で保持される（表 3(d)).

3.4 次元の導出

OLAP 分析を行うためには，分析の軸に対応する次元表が必

表 2: Type-Ppartitioned Triple Store 形式.

(a) rdf:type: SalesResult

Subject	Predicate	Object	DataType
SR_1	store	Store_1	Resource
SR_1	category	Cat_1	Resource
SR_1	price	“150000”	Integer
SR_1	time	Time_1	Resource
SR_2	store	Store_2	Resource
SR_2	price	“4000”	Integer

(b) rdf:type: Store

Subject	Predicate	Object	DataType
Store_1	located_at	gn:7452809	*GeoNames*
Store_1	branch	“Tsukuba”	String
Store_1	tel	“029-800-000”	String
Store_2	located_at	gn:2111901	*GeoNames*
Store_2	branch	“Mito”	String

(c) rdf:type: Instant

Subject	Predicate	Object	DataType
Time_1	inXSDDateTime	“2012-12-01T13:15:00”	DateTime

(d) rdf:type: Category

Subject	Predicate	Object	DataType
Cat_1	name	“Mac”	String
Cat_1	category	Cat_2	Resource
Cat_2	name	“Computer”	String

表 3: Property Table 形式.

(a) rdf:type: SalesResult

Subject	store	category	price	time
SR_1	Store_1	Cat_1	“15000”	Time_1
SR_2	Store_2	*null*	“4000”	*null*

(b) rdf:type: Store

Subject	located_at	branch	tel
Store_1	gn:7452809	“Tsukuba”	“029-800-000”
Store_2	gn:2111901	“Mito”	*null*

(c) rdf:type: Category

Subject	name	category
Cat_1	“Mac”	Cat_2
Cat_2	“Computer”	*null*

(d) Relationships in each Property Tables.

table	column	f_table	f_column	is_one2one
SalesResult	store	Store	Subject	false
SalesResult	time	Instant	Subject	false
Store	located_at	GeoNames	Subject	false
Category	category	Category	Subject	false

要である．また，それぞれの次元は概念階層をもつ．本研究では，RDF および LOD の特徴を活かした次元表を導出する手法を三つ提案する．

3.4.1 リテラル値の利用

このケースでは，事実表における一つの属性は，他の一つ以上の表と 1:1 もしくは N:1 の関係を持ち，それらが直接概念階層を構成している場合である．例としては時刻や位置情報などが挙げられる．時刻は，年，月，日，時，分，秒などの階層構造を，位置情報は，国，州，街（町）などの階層構造をそれぞれ構成することができる．

ある表の一つの属性として格納されている時刻値を例とすると，決められたルールに従って階層構造として属性を組み立て直す．表 4(a) は`"2012-12-01T13:15:00"^^xsd:dateTime` と記述された時刻情報を，次元表として利用するために階層構造に分解した例である．

3.4.2 内在する階層構造の利用

RDF はそもそも概念体系を記述ことができるため，データセット自身が概念階層を記述している場合がある．このような階層

Algorithm 1 内在する階層構造の利用

```
procedure ExtractConceptHierarchy(PT, ref) ▷ PT: property table, ref: foreign key attribute in PT
    R ← ∅
    r ← the root resource where the ref attribute is null.
    P ← GetRootToLeafPaths(PT, ref, ∅, r)
    n ← max path length in P.
    Create table T table_name(Subject, layer_1, ..., layer_n)   ▷ “table_name” is retrieved from the rdf:type
    Insert each path in P in T.
end procedure
procedure GetRootToLeafPaths(PT, ref, P, r) ▷ PT: property table, ref: foreign key attribute, P: current paths, r: currentresource URI
    if P is empty then
        P ← {p/r}                                   ▷ / is the separator.
    else
        for all p ∈ P do
            P ← P − {p} + {p/r}                     ▷ / is the separator.
        end for
    end if
    R ← ∅                                           ▷ Set of root to leaf paths.
    C ← resources where ref is r.                   ▷ Child nodes.
    for all c ∈ C do
        R ← R+ GetRootToLeafPaths(PT, ref, P, c)
    end for
    Return R
end procedure
```

関係は多くの場合，`rdfs:subClassOf`，`rdfs:subPropertyOf` などのリソースの間の参照関係によって記述される為，関係表同士の関係がリソース間の関係を表している場合がある．特に，述語表における同一タイプのインスタンスへの参照は自己参照とみなし，そのタイプにおける階層構造を組み立てることが可能である．アルゴリズム 1 は自己参照から概念階層を組み立てる例である．また，表 4(b) は自己参照を持つ商品カテゴリの例を階層構造に分解した例である．

3.4.3 外部情報の利用

LOD はその特徴として他の LOD へのリンクを含んでおり，いくつかのよく知られた LOD データセットは多くの LOD から参照される傾向にある．例えば，地理情報の LOD として GeoNames [17]，百科事典の LOD としては DBpedia [1] などが挙げられる．さらに，“Time Ontology” [11]，“Timeline Ontology” (注4)，“WG84 Geo Position Ontology” (注5) など，時刻や地理情報など様々な分野でよく用いられるオントロジが存在する．これらの LOD やオントロジはよく知られており，しばしば参照される．アルゴリズム 1 を使うことでこれらの概念階層を抽出し，次元表として保持することで多次元分析に利用することができる．表 4(c) は GeoNames におけるつくば市の URI (注6) から階層構造を抽出した例である．

3.5 スキーマ生成

分析に用いる OLAP システムに用いるスタースキーマを生成する．フレームワークは導出された次元表の候補をユーザーに示し，ユーザーは OLAP 分析に利用する次元表を選択する．ユーザーの選択を基に OLAP システムで利用するスキーマを出力する．このとき，スキーマをシンプルに保つため，表同士の関係が 1:1 であるものはテーブル同士を結合する．

3.6 プロトタイプ実装

我々は Ruby 言語を用いて提案フレームワークのプロトタイ

(注4)：The Timeline Ontology `http://motools.sourceforge.net/timeline/`
(注5)：WG84 Geo Position Ontology. `http://www.w3.org/2003/01/geo/`
(注6)：`http://sws.geonames.org/7452809/about.rdf`

表 4: それぞれの次元導出手法を適用した例.

(a) リテラル値の利用.

Subject	year	month	date	hour	min.	sec.
Time_1	2012	12	1	13	15	0

(b) 内在する階層構造の利用.

Subject	layer_1	layer_2
Cat_1	Cat_2 (“Computer”)	Cat_1 (“Mac”)

(c) 外部リソースの利用.

Subject	layer_1	layer_2	layer_3	layer_4	layer_5
gn:7452809	gn:6295630 (“地球”)	gn:6255147 (“アジア”)	gn:1861060 (“日本”)	gn:2112669 (“茨城”)	gn:7452809 (“つくば”)

プを実装した．この実装では，RDBMS と組み合わせて利用する OLAP システムとして Mondrian OLAP (注7) を利用することを想定している．RDBMS に MySQL を利用し，Mondrian 用のスキーマ定義ファイル（XML 形式）を出力する．この実装をライブラリ化したものを “LDETL” と名付け，オープンソースプロジェクトとして開発している (注8)．また，“LDETL” は，“Linked Open Data Challenge Japan 2012” のアプリケーション部門に応募されている (注9)．

4. 適 用 例

提案フレームワークの有効性を確認するために，LOD として公開されている数値・統計データに対してプロトタイプ実装を用いて，OLAP 分析に利用可能なスキーマの出力が可能か検証を行った．

4.1 日本における空間放射線量観測データへの適用

日本における都道府県別の環境放射線水準調査 (注10) を試験的に RDF 化しているサービス (注11) から，RDF を取得した．このデータセットには，2011 年 3 月 16 日から 2012 年 3 月 15 日までの，47 都道府県別の 1 時間ごとの空間放射線量観測結果が含まれている．

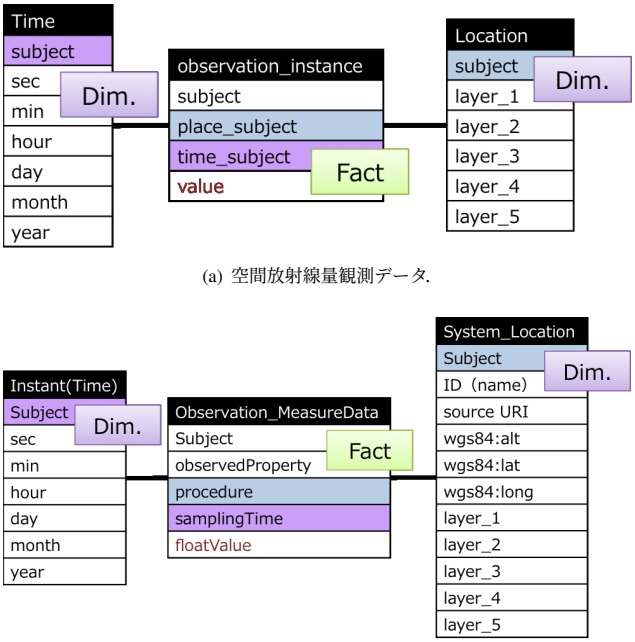
データセットは，複数の観測インスタンスとそれを記述するリテラル値で記述された観測値と観測時刻，リソースとして観測地がまとめられている．それぞれの観測インスタンスは “Event Ontology” (注12)，観測時刻は “The Timeline Ontology” をそれぞれ用いて記述され，観測地は GeoNames 上のリソースへリンクしている．

4.1.1 結 果

図 2(a) は，数値型の属性 `value` を含む観測インスタンステーブルを事実表とした時にプロトタイプが出力したスキーマである．選択された観測時刻と観測地は次元表として関係をもつ．観測時刻はリテラル値から概念階層として，観測地は外部リソースである GeoNames 上のリソースを取得して概念階層に展開された．

4.2 Linked Sensor/Observation Data への適用

Linked Sensor/Observation Data [14] と呼ばれる，米国における約一千箇所の気象観測所のメタデータ，およびそれらのセンサで観測された気象観測データをハリケーン毎にまとめ，LOD として公開されているものである．Linked Sensor Data は観測所メタデータとして，観測所 ID，座標，および GeoNames 上のリソースのうち最も観測所から近いリソースを含んでいる．Linked Observation Data は観測インスタンス，観測値，観測時刻，観測の種類（気温，湿度，降雨，など），および観測地として Linked Sensor Data に含まれる観測所インスタンスへのリンクを含んでいる．



(a) 空間放射線量観測データ.

(b) ハリケーン Bill.

図 2: 結果として得たスキーマ.

Linked Observation Data はハリケーン毎にまとめられているため，今回は “Bill” と呼ばれるハリケーンに関する観測データをまとめたデータセットを検証に用いた．このデータセットには 231,023,108 個の RDF トリプルで記述された 21,272,790 回の観測インスタンスが含まれている．

4.2.1 結 果

図 2(b) は，float 型の属性 `floatValue` を含む観測インスタンステーブルを事実表とし，関係を持つ観測時刻，観測所を次元表とした時のスキーマである．時刻情報はリテラル値から概念階層として，観測地は座標情報と GeoNames 上の外部リソースが概念階層として展開された．

4.3 他のデータセットへの適用

生物情報学の分野で，マウスの遺伝子情報 (注13) や，シロイ

(注7)：Pentaho Mondrian Project. http://mondrian.pentaho.com/
(注8)：LDETL. http://inohiro.github.com/ldetl/
(注9)：LDETL. http://lod.sfc.keio.ac.jp/challenge2012/show_status.php?id=a044
(注10)：文部科学省 放射線モニタリング情報. http://radioactivity.mext.go.jp/ja/monitoring_by_prefecture/
(注11)：National Radioactivity Stat as Linked Data. http://www.kanzaki.com/works/2011/stat/ra/
(注12)：The Event Ontology. http://motools.sourceforge.net/event/
(注13)：Mouse MGI Gene. http://biolod.org/database/rib185i/Mouse_MGI_Gene

ヌナズナの遺伝子情報 (注14) に対して本手法の適用を実験的に行った．

5. 関 連 研 究

RDF や LOD を OLAP により分析しようとする試みはこれまでにいくつか行なわれている．手法は大きく分けて二つの方向性で分類できる．一つは，RDF データセットを関係データベースに格納し，ROLAP（Relational-OLAP）と呼ばれるタイプの OLAP システムを使うことで実現する手法，もう一方は RDF データセットに対して RDF 問合せ言語である SPARQL を使って，OLAP 的な操作を適用する手法である．それぞれ代表的な研究を挙げる．

まず，Linked Data として公開されたデータセットを関係データベースに格納する手法として，Kampgen ら [13] の手法がある．彼らは，RDF で統計データを記述するための RDF 語彙である，RDF Data Cube Vocabulary（QB） [6] を利用している Linked Data に対して，関係モデルとのマッピングルールを定義し，関係データベースへ格納することで一般的な OLAP システムによる分析の実現可能性を示した．

関係データベースに格納を行わない手法としては，Etcheverry ら [7] の手法が挙げられる．彼らはまず，RDF データモデルに対する OLAP 処理を定義し，その処理が適用可能な RDF データセットを作るための RDF 語彙である Open Cube Vocabulary を定義した．この語彙に準拠したデータセットに対して，SPARQL による OLAP 的な多次元分析処理の一部を実現した．また，分析処理を行う SPARQL クエリを生成するアルゴリズムを示した．さらに，彼らは QB4OLAP と呼ばれる RDF 語彙を提案した [8]．QB4OLAP は QB に OLAP モデルと操作を拡張した RDF 語彙で，既に QB に基づいて公開されている RDF データを QB4OLAP に準拠させることで，SPARQL を用いた OLAP 的な分析処理を可能にした．

これらに対して本研究では，前者の RDF データセットを関係データベースへ格納するアプローチを取り，OLAP 分析向けの特別な RDF 語彙の利用に関係なく，LOD として公開された任意の数値，統計データを対象に，既存の OLAP システムによる分析処理を可能にする一般的な変換手法について議論している点が異なる．

6. まとめと今後の課題

本稿では，LOD として公開されている数値，統計データに対して既存の OLAP システムを用いた多次元分析を可能にする一連の変換手法を ETL フレームワークとして提案した．本手法はこれまでの先行研究と異なり，RDF や LOD の特徴を利用することで，OLAP データモデル用の特定の RDF 語彙を必要とせず，より幅広い分野の LOD に対して適用可能であると言える．また，提案フレームワークのプロトタイプ実装に LOD として公開されているデータセットを適用し，OLAP 分析に必要なスキーマの生成に成功することを示した．

今後の課題として，1）rdf:type が記述されていない，もしくは複数記述されているリソース，2）LOD データセットが更新された場合に DWH 側へ更新を反映する仕組みなどについて検討する必要がある．

謝　　辞

本研究の一部は，科学研究費補助金・若手研究 B(23700102) による．

文　　献

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. DBpedia: A Nucleus for a Web of Open Data. In *ISWC/ASWC*, volume 4825 of *Lecture Notes in Computer Science*, pages 722–735. Springer, 2007.

[2] Tim Berners-Lee. Linked Data - Design Issues. http://www.w3.org/DesignIssues/LinkedData.html.

[3] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.

[4] Jeremy J. Carroll and Graham Klyne. Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C recommendation, W3C, February 2004. http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/.

[5] E.F. Codd, S.B. Codd, C.T. Salley, and Inc Codd & Date. *Providing OLAP (On-line Analytical Processing) to User-analysts: An IT Mandate*. Codd amp; Associates, 1993.

[6] Richard Cyganiak and Dave Reynolds. The RDF Data Cube Vocabulary. W3C working draft, W3C, April 2012. http://www.w3.org/TR/vocab-data-cube/.

[7] Lorena Etcheverry and Alejandro A. Vaisman. Enhancing OLAP Analysis with Web Cubes. In *ESWC*, volume 7295 of *Lecture Notes in Computer Science*, pages 469–483. Springer, 2012.

[8] Lorena Etcheverry and Alejandro A. Vaisman. QB4OLAP: A Vocabulary for OLAP Cubes on the Semantic Web. In *COLD*, volume 905 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.

[9] Ramanathan V. Guha and Dan Brickley. RDF Vocabulary Description Language 1.0: RDF Schema. W3C recommendation, W3C, February 2004. http://www.w3.org/TR/2004/REC-rdf-schema-20040210/.

[10] Jiawei Han and Micheline Kamber. Data Warehouse and OLAP Technology: An Overview. In *Data Mining: Concepts and Techniques*, pages 105–156. Morgan Kaufmann, second edition, 2006.

[11] Jerry R. Hobbs and Feng Pan. Time Ontology in OWL. W3C working draft, W3C, September 2006. http://www.w3.org/TR/owl-time/.

[12] Aftab Iqbal, Sarven Capadisli, Richard Cyganiak, and Michael Hausenblas. Eurostat - Linked Data. http://eurostat.linked-statistics.org/.

[13] Benedikt Kämpgen and Andreas Harth. Transforming statistical linked data for use in OLAP systems. In *I-SEMANTICS*, ACM International Conference Proceeding Series, pages 33–40. ACM, 2011.

[14] Harshal Patni, Cory A. Henson, and Amit P. Sheth. Linked Sensor Data. In *CTS*, pages 362–370, 2010.

[15] Erhard Rahm and Hong Hai Do. Data Cleaning: Problems and Current Approaches. *IEEE Data Eng. Bull.*, 23(4):3–13, 2000.

[16] Andy Seaborne and Steve Harris. SPARQL 1.1 Query Language. W3C working draft, W3C, October 2009. http://www.w3.org/TR/2009/WD-sparql11-query-20091022/.

[17] Bernard Vatant and Marc Wick. GeoNames Ontology. http://www.geonames.org/ontology/.

[18] Kevin Wilkinson, Craig Sayers, Harumi A. Kuno, and Dave Reynolds. Efficient RDF Storage and Retrieval in Jena2. In *SWDB*, pages 131–150, 2003.

[19] Kevin Wilkinson and Kevin Wilkinson. Jena Property Table Implementation. In *In SSWS*, 2006.

(注14)：TAIR10 Locus. http://biolod.org/class/crib102u0i/TAIR10_Locus