異種データベースの仮想化技術
-スキーマ変換方式-

渡辺 裕太† 菖蒲 佳右† 和田 雄次‡ 澤本 潤‡‡ 加藤 貴司‡‡
東京電機大学大学院 情報環境学研究科† 東京電機大学 情報環境学部‡
岩手県立大学 ソフトウェア情報学部‡‡

1 はじめに

今日では，ユビキタスセンサーネットワーク環境上から大量のデータが収集されており，これらのデータの中に隠された知識や傾向を，データマイニング技術を用いて発見・分析し，業務の意思決定などに役立てることが重要となっている．そして，それらのデータは分散配置された多種多様なデータベース（すなわち，マルチデータベース）に存在する．

しかし，このマルチデータベースに対し，データマイニングを行う分析者は本来的には分析やルール抽出作業に集中したいにも拘わらず，データマイニングの準備過程であるデータベース選択やデータ収集などの作業に多大な時間を費やしてしまうといった課題がある．

そこで，本研究ではこのデータ分析者の負担を軽減するために，ユビキタスコンピューティング環境上のマルチデータベースが，あたかも一つのデータベースであるかのように利用できるデータベース仮想化技術の開発を目的とする．

マルチデータベースにはデータモデルの違いやベンダーの違う様々な種類のものが存在する．データモデルの違うデータベースでは，データの表現方法が異なり，格納方式や制約がそれぞれ特有である．代表例としてRDB，XMLDB，OODB などが挙げられる．また，同じモデルのデータベースでも，様々なベンダーによる製品が開発されており，それらの機能や操作方法には多少の違いがある．例えば RDB においては，MySQL[*1] や PostgreSQL[*2] などが挙げられ，それらの SQL はそれぞれ特有である．

これらのモデルやベンダーの機能の違いは，いくつかの不都合が生じる．例えば，複数の異種 DB のデータを扱うのであれば，それぞれの API が必要となり，また運用保守の際にシステムの仕様変更が発生した場合，その影響が大きくなるほど多くの保守作業が必要となる．それらの異種データモデル DB やベンダーの違うデータベースを仮想化し，手続きを一体化させることによって負荷やコストなどが軽減され，柔軟にアプリケーション設計やデータベース管理を行うことが出来ると考えられる．

2 関連研究

仮想化データベースに関連する，いくつかの研究がある．参考文献[1]では，広域ネットワークに接続された異種データベース群を情報源としてモバイルコンピューティング環境上のユーザへ能動的に情報配信するシステムの実現方式を提案している．異種データベースとして，ローカルデータベース群から写像されたデータを統合し，メタデータベースを構築することによって，それを基に異種データベースからの検索を行っている．また，参考文献[2]では，同様に異種データの仮想化を行うものであり，Teiid 仮想化システムを経由し，RDB，Web サービス，ERP やCRMの業務アプリケーションなど，さまざまなデータソースにリアルタイムでアクセスし統合することができる．しかし，異ベンダーRDBのみ仮想化する機能がサポートされており，異種データモデルDBに対しての仮想化機能はサポートされていない．

本研究では，共通スキーマであるXML スキーマを元に，一つのクエリによって様々な異種データベースの操作が行えるといったデータベース仮想化技術を提案する．

3 仮想化技術

データベース仮想化技術のレイヤをFig1に示す．データベース仮想化技術として，構造の違いや位置の違いなどを利用者に意識させないため，同種分散データベース管理機能や異種分散データベース管理機能，位置透過性機能といった機能を取り入れ，あらゆるデータベースに対しても柔軟に対応できるようにすることを目的とする．

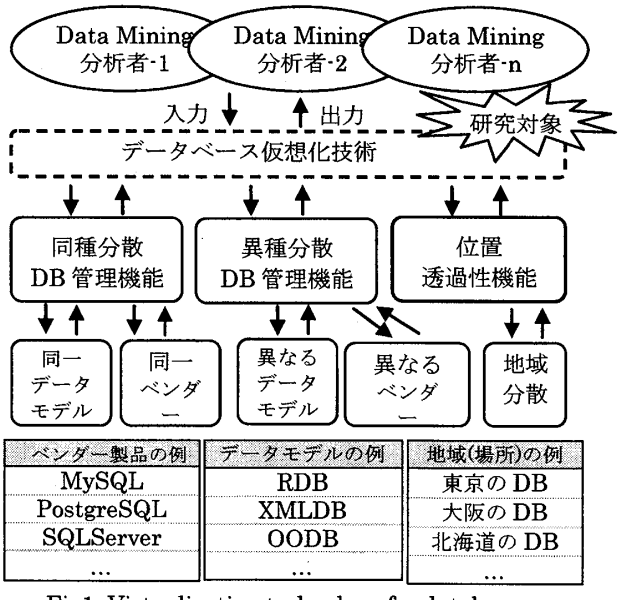

Fig1. Virtualization technology for databases

データベース仮想化技術の概要をFig2に示す．本研究では仮想化として，XMLDB やOODB といった異種データモデルDBとの仮想化を行うために，表現能力や移植性のあるXML スキーマを利用する．それぞれのDBのスキーマ情報を一つのXML スキーマで表現し，その共通スキーマを元にデータの検索や更新などを行えるよう

Virtualization technology for heterogeneous databases - Schema conversion method -
YutaWatanabe† Keisuke Shobu† YujiWada‡ Jun Sawamoto‡‡ Takashi Kato‡‡
†Graduate School of information Environment, Tokyo Denki University
‡School of Information Environment, Tokyo Denki University
‡‡Faculty of Software and Information Science, Iwate Prefectural University

[*1] "MySQL": Sun Microsystems
[*2] "PostgreSQL": PostgreSQL Global Development Group

にする．今回はその中の共通スキーマを生成するモジュールを開発し動作を確認した．

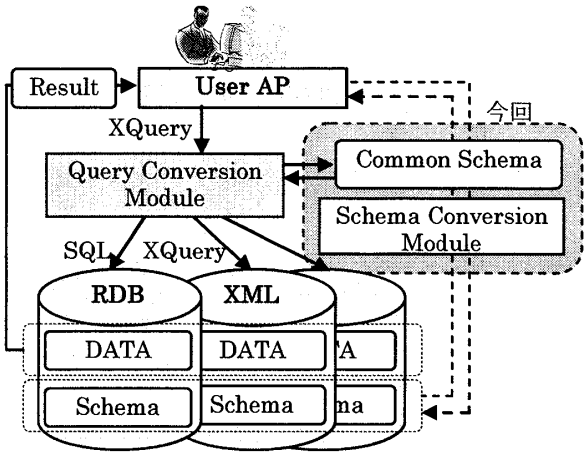

Fig2. Summary of Virtualization system for databases

3.1 共通スキーマ生成モジュール

共通スキーマはUser APに仮想化DBの構造情報を与え，Queryの構文や制約のチェックにも用いられる．今回はRDBから共通スキーマを生成するプログラムを作成した．RDBをXMLスキーマで表した構造はFig3のような階層になる．制約情報については，以前の研究方法であるXMLスキーマ機能だけでの表現では全ての情報を表現することができなかった．例えば，外部キー制約の細かな情報やCHECK制約などである．そこで，今回は <xsd:annotation>，<xsd:appInfo>を使用し，情報用のコメントとして制約情報を持たせるようにした．主な詳細は次のとおりである．主キー(ユニーク)制約: <r:index index-key="カラム名" primary(unique)="yes">，外部キー制約: <f:key fkey-column="カラム名" ref-column="参照カラム" ref-table="参照テーブル" rule-delete="削除時の操作" rule-update="更新時の操作" />，チェック制約: <r:check check-column="カラム名" rule="規則">として出力される．

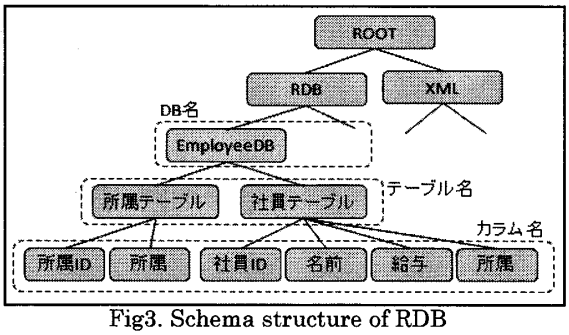

Fig3. Schema structure of RDB

3.2 動作例

今回は PostgreSQL を用いて，動作例に使用するサンプルスキーマとして以下の情報を持つデータベースを構築し，それらから XML スキーマを生成する．また，基本的な制約についても同様に確認を行うため，カラムには，社員 ID・所属 ID に主キー制約，所属にユニーク制約，所属 ID に外部キー制約，給与にチェック制約，名前に NotNull 制約をそれぞれ付与した．

- CREATEDB EmployeeDB;
- CREATE TABLE 社員テーブル(
  社員 ID int PRIMARY KEY,
  名前 varchar(50) NOT NULL,
  給与 int CHECK(0 < 給与),
  所属 ID int REFERENCES 所属テーブル (所属 ID)
  ON UPDATE CASCADE ON DELETE CASCADE);
- CREATE TABLE 所属テーブル(
  所属 ID int Primary Key,
  所属 varchar(5) UNIQUE);

以上の RDB スキーマから出力された XML スキーマを Fig4 に示す． Fig4 からカラム名や制約情報が出力されていることがわかる．

```
<!—省略-->
    <xs:element name="所属テーブル" type="所属テーブル Type"/>
    <xs:element name="社員テーブル" type="社員テーブル Type"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="所属テーブル Type">
  <xs:annotation>
    <xs:appinfo>
      <r:index index-key="所属 id" primary="yes"/>
      <r:index index-key="所属" unique="yes"/>
    </xs:appinfo>
<!—省略-->
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="1" name="所属 id" r:nullable="false"
        <xs:element minOccurs="0" name="所属" r:sqltype="varchar"
<!—省略-->
<xs:complexType name="社員テーブル Type">
  <xs:annotation>
    <xs:appinfo>
      <r:index index-key="社員 id" primary="yes"/>
      <r:check check-column="給与" rule="0 &lt; &quot;給与&quot;"
      <r:fkey fkey-column="所属 id" ref-column="所属 id" ref-table="
```

Fig4. Common Schema

4 おわりに

RDBから情報を取得し，構造化された共通スキーマを生成することができた．制約についても情報用コメントとして持たせることによって，制約の細かな情報も持たせることが可能である．今後は， XMLDBのXMLスキーマを共通スキーマへ統合する機能を取り入れる予定である．また，スキーマレス型のXMLDBからは，スキーマを取得する機能も取り入れる必要がある．

本研究において，仮想化データベースシステムの一部である共通スキーマ変換モジュールは開発段階である．このモジュールを完成させXQuery変換モジュールと統合し，位置透過性の機能を取り入れることによって，あらゆるデータベースにも対応できるようにする．そして最後にそれらについての評価を行う予定である．

謝辞

本研究は，科学研究費補助金（基盤研究(C) 課題番号 20500095「ユビキタスデータベース仮想化技術によるデータ効率化に関する研究」）の支援による．

参考文献

[1] 森薫, 倉林修一, 石橋直樹, 清木康: ”モバイルコンピューティング環境におけるユーザ情報の動的軽量による能動型情報配信方式”, DEW2004論文集. (2004)
[2] Red Hat: ”teiid”, http://www.jboss.org/teiid
[3] 渡辺裕太, 菖蒲佳右, 和田 雄次, 澤本 潤, 加藤 貴司: ”異種データベース仮想化技術 ”,第8回FIT講演論文集