DBFluteの紹介

2007.5.27

株式会社ビルドシステム
久保　雅彦

Agenda

- **Goal**
- Introduction
- DBFlute Style
- ConditionBean
- Sql2Entity
- History
- Summary/Q&A

DBFluteって何？

を知っていただくこと

Agenda

- Goal
- **Introduction**
- DBFlute Style
- ConditionBean
- Sql2Entity
- History
- Summary/Q&A

DBFluteは、S2Daoの利用を
サポートするToolです。

※Apache TorqueのGenerator機能を
S2Daoに適応させたものです。

Introduction : Function of DBFlute

- S2Daoで利用するDao/EntityをDatabaseのSchema情報を利用して自動生成します。
- S2Daoの“DTOによるSQL自動生成”の機能を拡張したDTOを自動生成します。
- DatabaseのSchema情報を表示する'SchemaHTML'を自動生成します。
- S2Daoの“外だしSQL”(SQLファイル)から対応するEntityを自動生成します。

Introduction : DBFluteのPolicy

【O/R-MapperとしてPolicy】

DBFluteは、SQL文を自動生成する機能を有しますが、SQL文を隠蔽するつもりはありません。

【実装Policy 】

[簡単なSQL＋複雑だけれども定型的なSQL]

ConditionBeanでTypeSafe実装

[複雑なSQL]

Sql2Entityの支援と共に“外だしSQL”で実装

- Goal
- Introduction
- **DBFlute Style**
- ConditionBean
- Sql2Entity
- History
- Summary/Q&A

DBFlute Style

- 1. DBのSchemaとTableを作成する。
- 2. DBFluteの環境を準備する。
    - Project固有の設定を行うなど
- 3. DBFluteでJdbc-Taskを実行し、DB情報を取得する。
- 4. DBFluteでGenerate-Taskを実行し、DB情報から「Dao/Entity/dicon/DBFlute独自Class」を自動生成する。
- 5. 自動生成されたClassを利用してApplicationを実装する。

【DB変更発生】

A. DBのSchemaとTableを変更する。

B. 「3」/「4」を実行する。

C. Compile-Errorになった部分を修正する。

DB変更に強い！

Agenda

- Goal
- Introduction
- DBFlute Style
- **ConditionBean**
- Sql2Entity
- History
- Summary/Q&A

What is ConditionBean?

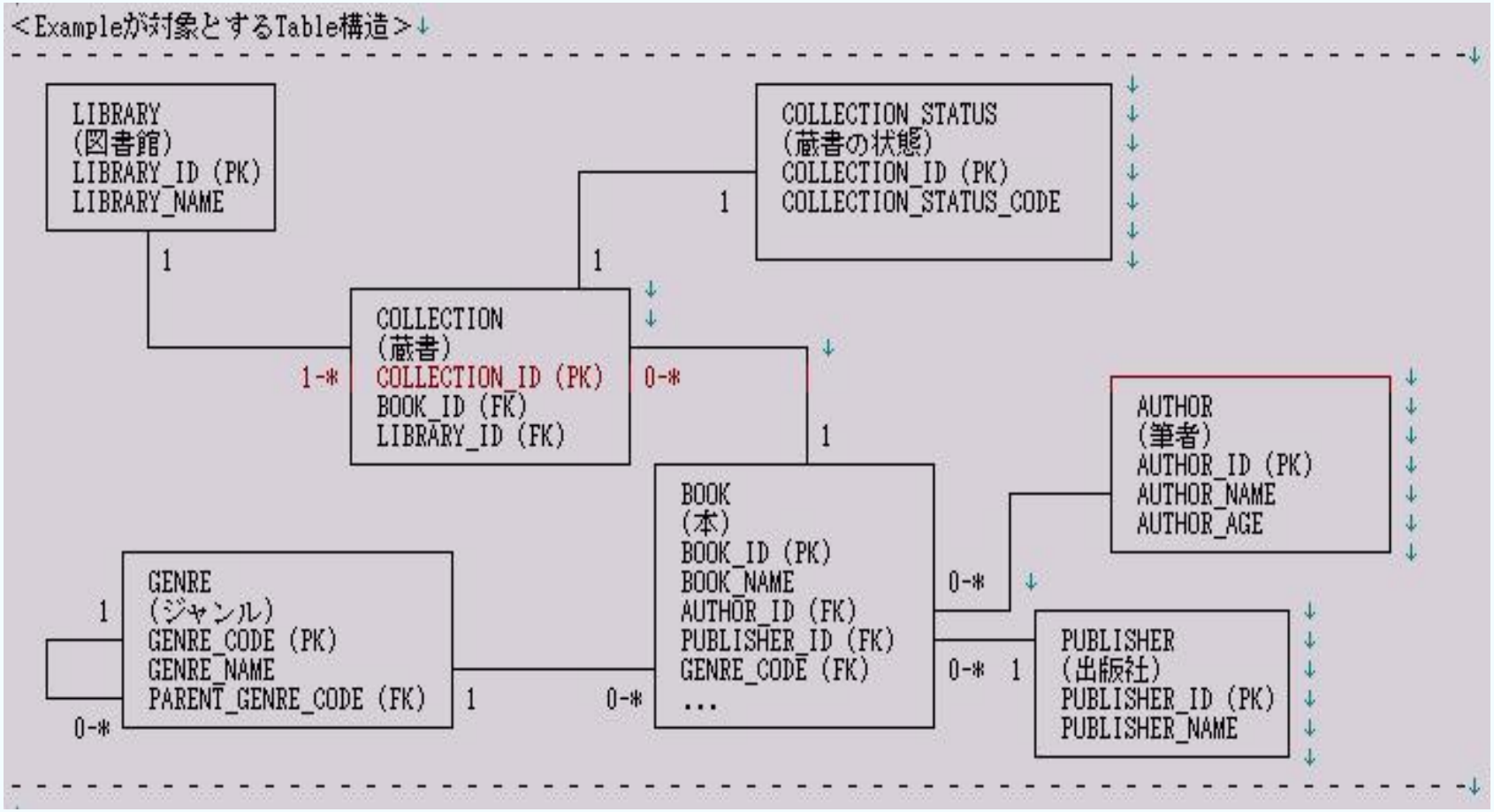
S2Daoの「DTO引数によるSQL自動生成機能」を拡張し、SQL自動生成の幅を広げたもの。

```
[Dto]
public class EmployeeSearchCondition {
        public static final String dname_COLUMN = "dname_0";
        private String job;
        private String dname;
        ...
}
[Dao]
List getEmployeesBySearchCondition(EmployeeSearchCondition dto);
```

What has been enhanced to it?

1. Where句に指定する条件で{=}以外の演算子もサポート
2. OrderBy句に指定するColumn & {Asc or Desc}の指定をサポート
3. Select句に含める結合先Tableの指定をサポート
4. Where句に指定する条件で無限階層のForeignTableのColumnの指定をサポート
5. fetchFirst(), fetchScope(), fetchPage()によるLimitSearch/PagingSearchをサポート
6. lockForUpdate()による更新Lockのサポート
7. …

Sample DB「LDB」



ConditionBean : Query

Where句に指定する条件で{=}以外の演算子もサポート

ex) BOOKに対してBOOK_NAMEの‘S2Dao’前方一致で、かつ、GENRE_CODEがnullでないものを検索

```
/- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
// select ...
//   from BOOK book
//  where book.BOOK_NAME like 'S2Dao%'
//    and book.GENRE_CODE is not null
//
final LdBookCB cb = new LdBookCB();
cb.query().setBookName_PrefixSearch("S2Dao");
cb.query().setGenreCode_IsNotNull();// ☆Point
final java.util.List<LdBook> ls = bhv.selectList(cb);
- - - - - - - - - - -/
```

ConditionBean : Order By

OrderBy句に指定するColumn&{Asc or Desc}の指定をサポート

ex) BOOKに対してAUTHOR_IDの降順かつBOOK_IDの昇順で全件検索

```
/- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
// select ...
//   from BOOK book
//  order by book.AUTHOR_ID desc, book.BOOK_ID asc
//
final LdBookCB cb = new LdBookCB();
cb.query().addOrderBy_AuthorId_Desc()
                  .addOrderBy_BookId_Asc();
final java.util.List<LdBook> ls = bhv.selectList(cb);
- - - - - - - - -/
```

ConditionBean : Select Tables

Select句に含める結合先Tableの指定をサポート

ex) BOOKに対してAUTHORを結合してSelect句に含める検索

```
/- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
// select book.BOOK_ID, book...
//          , author.AUTHOR_ID as authorId_0, author...
//   from BOOK book
//     left outer join AUTHOR author
//       on book.AUTHOR_ID = author.AUTHOR_ID
//
final LdBookCB cb = new LdBookCB();
cb.setupSelect_Author();
final java.util.List<LdBook> ls = bhv.selectList(cb);
- - - - - - - - -/
```

ConditionBean : Query ForeignTable

Where句に指定する条件で無限階層のForeignTableのColumnの指定をサポート

ex) BOOKに対してForeignTableのAUTHORを結合してAUTHOR_AGEで30歳以上を検索。(AUTHOR_AGE降順)

```
/- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
// select ...
//   from BOOK book
//     left outer join AUTHOR author
//       on book.AUTHOR_ID = author.AUTHOR_ID
//  where author.AUTHOR_AGE >= 30
//  order by author.AUTHOR_AGE desc
//
final LdBookCB cb = new LdBookCB();
cb.query().queryAuthor().setAuthorAge_GreaterEqual(30);
cb.query().queryAuthor().addOrderBy_AuthorAge_Desc();
final java.util.List<LdBook> ls = bhv.selectList(cb);
- - - - - - - - - -/
```

ConditionBean : LimitSearch/PagingSearch-1

fetchFirst(), fetchScope(), fetchPage()によるLimitSearch/PagingSearchをサポート

ex) BOOKに対して登録日時の降順で、最初の50件のみ検索

```
/- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
final LdBookCB cb = new LdBookCB();
cb.query().addOrderBy_RTime_Desc();
cb.fetchFirst(50);
final java.util.List<LdBook> ls = bhv.selectList(cb);
- - - - - - - - - -/
```

ConditionBean : LimitSearch/PagingSearch-2

fetchFirst(), fetchScope(), fetchPage()によるLimitSearch/PagingSearchをサポート

ex) BOOKに対して登録日時の降順で、81件目から100件目のみ検索

```
/- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
final LdBookCB cb = new LdBookCB();
cb.query().addOrderBy_RTime_Desc();
// 80件飛ばして20件分を取得(81件目から100件目)
cb.fetchScope(80, 20);
final java.util.List<LdBook> ls = bhv.selectList(cb);
- - - - - - - - - -/
```

ConditionBean : LimitSearch/PagingSearch-3

fetchFirst(), fetchScope(), fetchPage()によるLimitSearch/PagingSearchをサポート

ex) BOOKに対して登録日時の降順で、PageSize20件でPaging検索

```
/- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
final LdBookCB cb = new LdBookCB();
cb.query().addOrderBy_RTime_Desc();
cb.fetchFirst(20);
cb.fetchPage(1);// ☆Point - 1-20
final java.util.List<LdBook> lsFirstPage = bhv.selectList(cb);

cb.fetchPage(2);// ☆Point - 21-40
final java.util.List<LdBook> lsSecondPage = bhv.selectList(cb);

cb.fetchPage(3);// ☆Point - 41-60
final java.util.List<LdBook> lsThirdPage = bhv.selectList(cb);
- - - - - - - - -/
```

ConditionBean : LimitSearch/PagingSearch-4

fetchFirst(), fetchScope(), fetchPage()によるLimitSearch/PagingSearchをサポート

- MySQL → limit/offset構文利用
- Postgre → limit/offset構文利用
- Firebird → first構文を利用
- Oracle → rownum利用
- DB2 → fetch first x rows only構文利用
  - ※OffsetはResultSetにてすっ飛ばし
- SQLServer → TOP構文利用
  - ※OffsetはResultSetにてすっ飛ばし
- Sybase → ※TOPを利用しようと思ったら Sybase-12.x ではサポートされていない！？ (15.xからしい)

Agenda

- Goal
- Introduction
- DBFlute Style
- ConditionBean
- **Sql2Entity**
- History
- Q&A

What is Sql2Entity?

S2Daoの“外だしSQL”(SQLファイル)から、それに対応するEntityを自動生成

＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊

このような開発が可能になります。

1. {.sql}を作る。
2. {.sql}から対応するEntityを自動生成する。
3. DaoにMethodを定義して呼び出す。

＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊

Sql2Entity : Making SQL

1. {.sql}を作る。

exdao-Packageに{.sql}を作成。

ex) LdBookDao_selectBookCollectionStatistic.sql

```
/- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
--#BookCollectionStatistic#
select book.BOOK_ID
       , book.BOOK_NAME
       , (select count(*) from COLLECTION
          where BOOK_ID = book.BOOK_ID) as COLLECTION_COUNT
  from BOOK book
 /*BEGIN*/where
   /*IF bookName != null*/book.BOOK_NAME like /*bookName*/'S2Dao' || '%'/*END*/
 /*END*/
 order by COLLECTION_COUNT desc
;
- - - - - - - - - -/
```

※自動生成したいEntityのClass名を「--#EntityName#」という形式でSQLに記述。

Sql2Entity : Making SQL

2. {.sql}から対応するEntityを自動生成する。

sql2entityを実行。

{DBFlute内部処理}

1. ‘exdao’以下(ExtendedDao)のDirectoryに存在する‘.sql’ファイルを読み込む。
2. ‘.sql’の中に‘--#entityName#’と記載されているものを対象にSQLを実行する。
3. 結果のResultSetのMetaDataから、ColumnNameとColumnTypeを取得する。
4. CustomizeEntityを自動生成する。

※必ず2WAY-SQLにしておくこと

3. DaoにMethodを定義して呼び出す。

手動で exdaoにMethodを定義

```
public java.util.List<LdBookCollectionStatistic>
    selectBookCollectionStatistic(String bookName);
```

- Goal
- Introduction
- DBFlute Style
- ConditionBean
- Sql2Entity
- **History**
- Q&A

History

- 2005年1-4月頃：作者がApache Torqueを利用したProjectを経験。
  - » 同時にSeasar/S2Daoの存在も知る。
  - » Torqueの便利さを感じつつ不足部分も痛感する。
  - » S2DaoのSQLコメント機能に感動し、Torqueの不足を埋められると感じる。
- 2005年5-7月頃：Torqueを基にS2DaoのDaoやEntityを生成するS2DaoGenを実装。
  - » SeasarProjectへの参加まで、**株式会社アークシステム様**のADEサーバ上のCVSにて管理。(本当にありがとうございました)
- 2005年12月　：S2DaoGenを改良し、現在のConditionBeanの構造を確立。
  - » それまでは、S2Daoの“DTOによるSQL自動生成”の延長でしかなかったが、その方法ではPerformanceに問題があることが判明した。新しい構造により、それらは解消された。
- 2006年6-7月頃：C#版へ移植し、作者自らProjectにて利用する。
  - » このとき現場で必要な機能の多くを実装。
  - » 同時に別ProjectにてJava版を利用してもらい、多くの要望とバグを修正。
- 2006年10月　：DBFlute誕生。SeasarProjectのSandBoxとしてOpenSourceとなる。
- 2007年～　　：実績と実力を付けてSandBox卒業を目指す。
  - » DBFlute用EclipsePluginの開発

Q&A