

# プロシージャ-とリスト

データベースにアクセスするためのコマンド

*GeneXus*<sup>TM</sup>

## Procedure オブジェクト

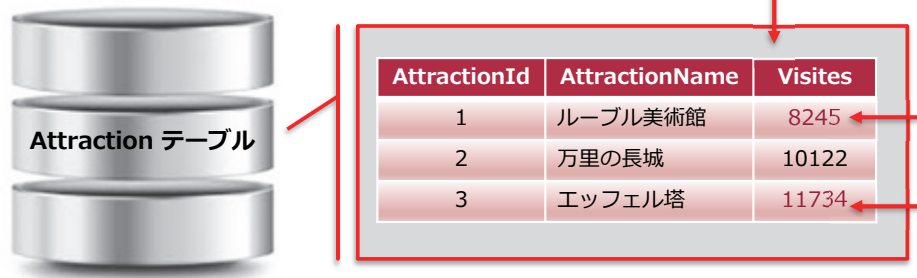
さまざまな目的でデータベース上のナビゲートされたテーブルにアクセスするプロセスを定義できる



いくつかの想定される目的についてこれから見ていきましょう。

## Procedure オブジェクト

- テーブルをナビゲートし、条件に一致する**レコードの一部**を、指定値で更新する



## Procedure オブジェクト

- テーブルをナビゲートし、その全レコードを任意の基準でソートし、**PDF** 形式で出力する



AttractionId	AttractionName	CountryId	...
1	ルーブル美術館	2	...
2	万里の長城	3	...
3	エッフェル塔	2	...

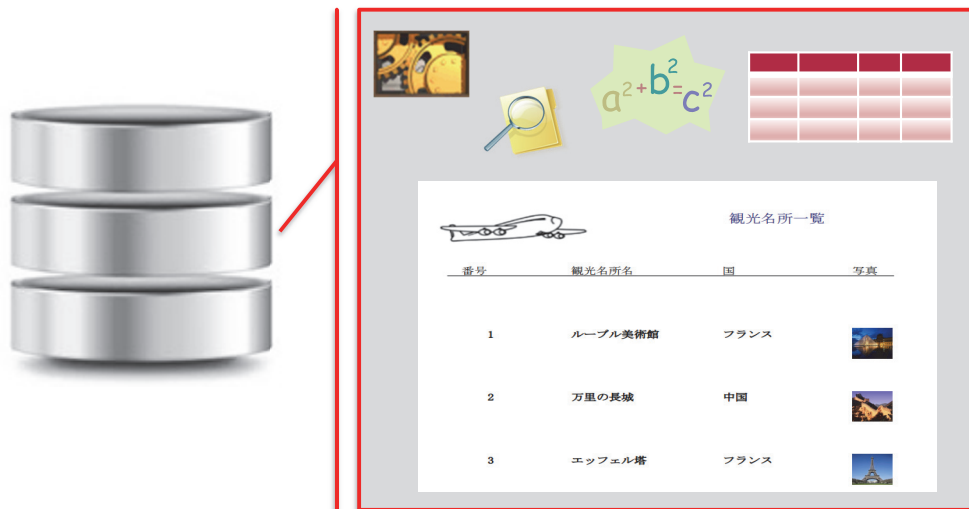


観光名所一覧

番号	観光名所名	国	写真
1	ルーブル美術館	フランス	
2	万里の長城	中国	
3	エッフェル塔	フランス	

## Procedure オブジェクト

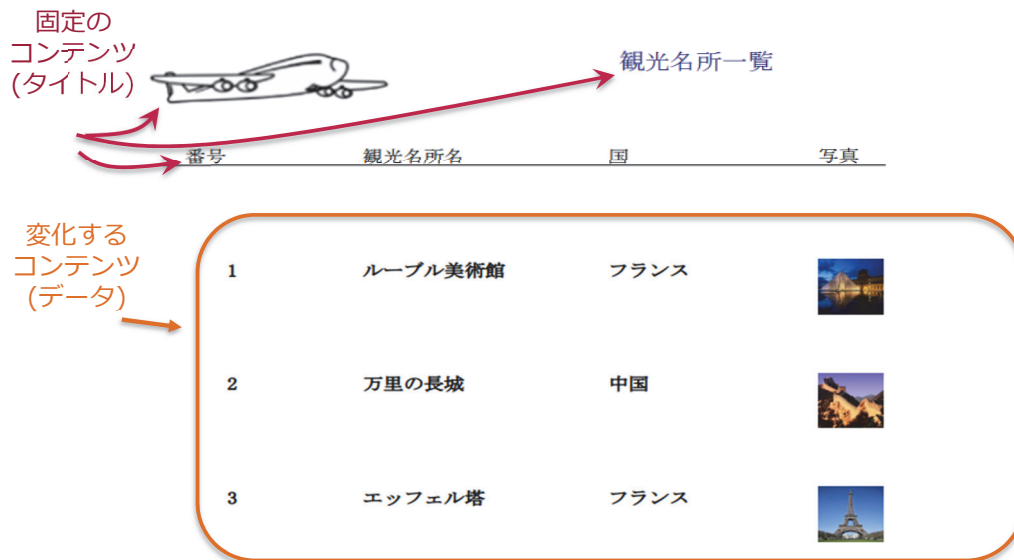
- 検索・計算・更新・出力等のプロセス定義 (いわゆるユーザー定義の関数)



このような処理をする場合は、ナレッジベースにプロシージャーを作成します (GeneXus **Procedure** オブジェクト)。

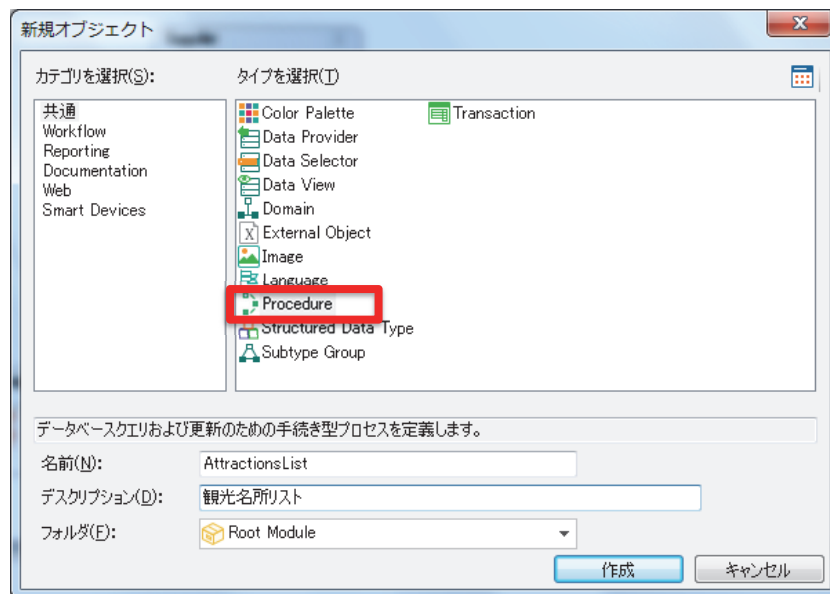
## 例: データを PDF 形式で出力

- 旅行代理店が扱うすべての観光名所のリストを、観光名所名順で PDF ファイルに出力する



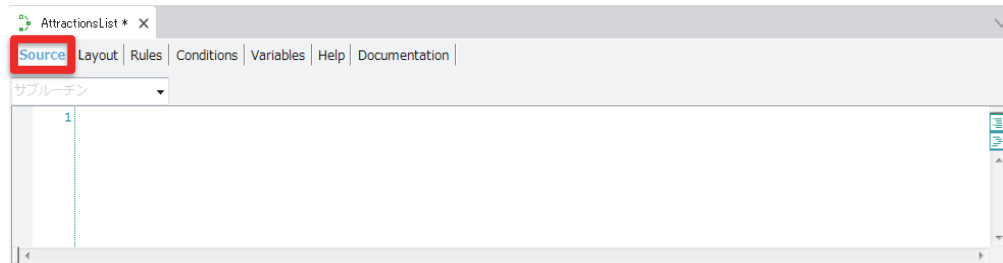
はじめに、旅行代理店が扱うすべての観光名所のリストを観光名所名順で生成するプロシージャーを定義してみましょう。

## Procedure オブジェクト

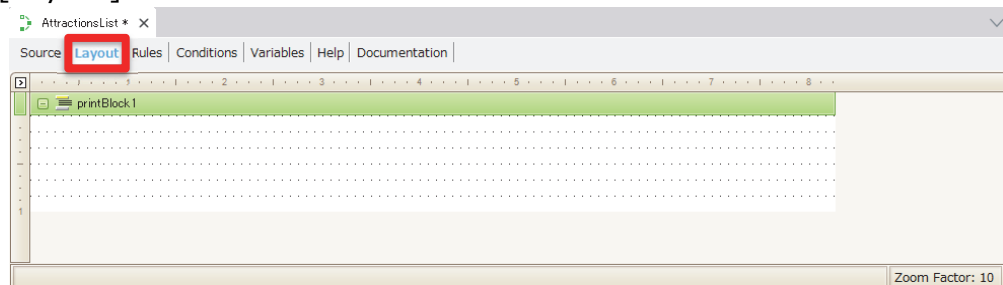


## プロシージャー: [Source] と [Layout]

- [Source]: 実行する命令



- [Layout]: 出力のデザイン



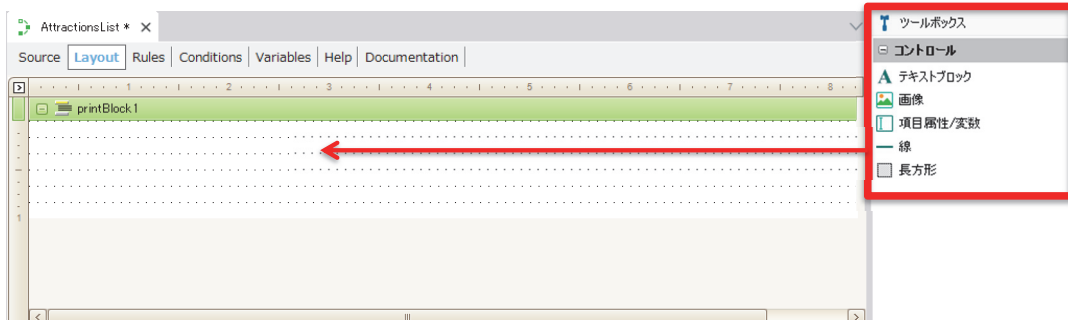
オブジェクトを作成すると、[Source] エlementが表示されます。[Source] Elementには、プロシージャーで目的の処理を実行するためのコマンドや命令を入力します。この例の場合は、観光名所のリストを印刷することが目的です。

次に、[Layout] Elementを見てみましょう。[Layout] Elementでは出力のデザインをします。つまり、データをどのように表示するかを設定します。



## [Layout] エlement

- プリントブロックで構成される



- 既定で 1 つのプリントブロックが表示されている (printBlock1)

[Layout] Elementはプリントブロックで構成され、プリントブロックの中に表示するものを指定します。



タイトル、線、長方形、画像のほか、項目属性や変数の値も表示できます。これらを表示するには、目的のものをツールバーからプリントブロックへドラッグします。

注意: 1 つのプリントブロックは [Layout] Elementに自動的に含まれています。

このプリントブロックを使用してタイトルや日付を表示したり、このセクションにプリントブロックを追加したりできます。詳細については、これから見ていきましょう。

## [Layout] エレメント

- リスト用の**プリントブロック**:

Title	 観光名所一覧			
Column Titles	番号	観光名所名	国	写真
Attractions	1	ルーブル美術館	フランス	
	2	万里の長城	中国	
	3	エッフェル塔	フランス	

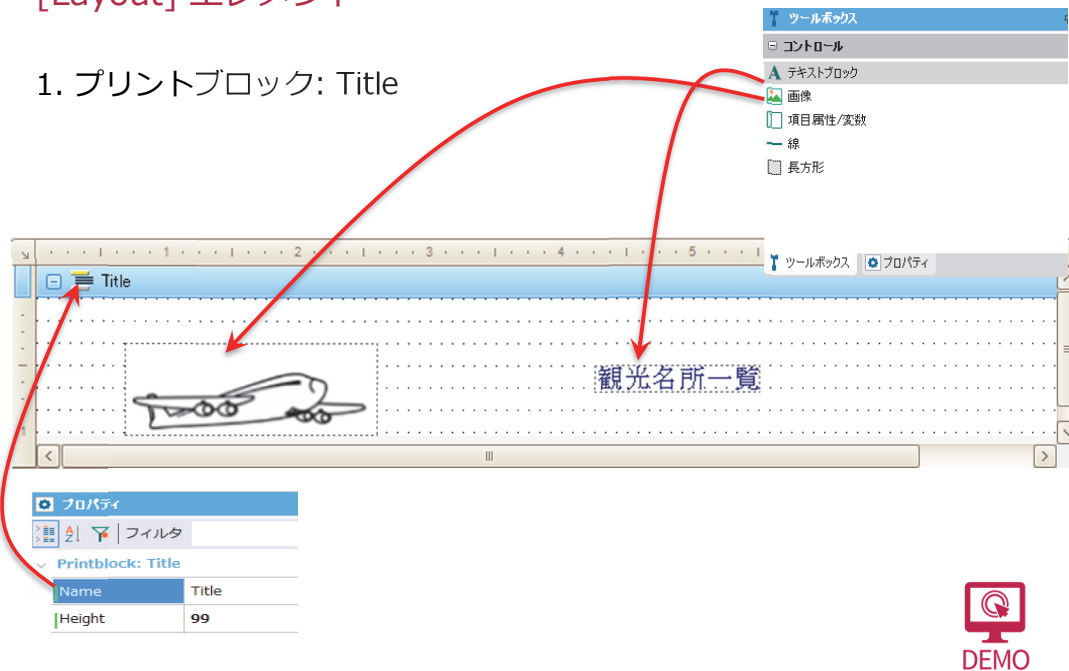
3 つのプリントブロックを定義します。

- 1 つ目はリストのタイトルと画像で、Title という名前にします。
- 2 つ目は列のタイトルに下線を付けたもので、ColumnTitles という名前にします。
- 3 つ目は観光名所の詳細を表示するもので、Attractions という名前にします。

それではこれらを定義していきましょう。

## [Layout] エlement

### 1. プリントブロック: Title



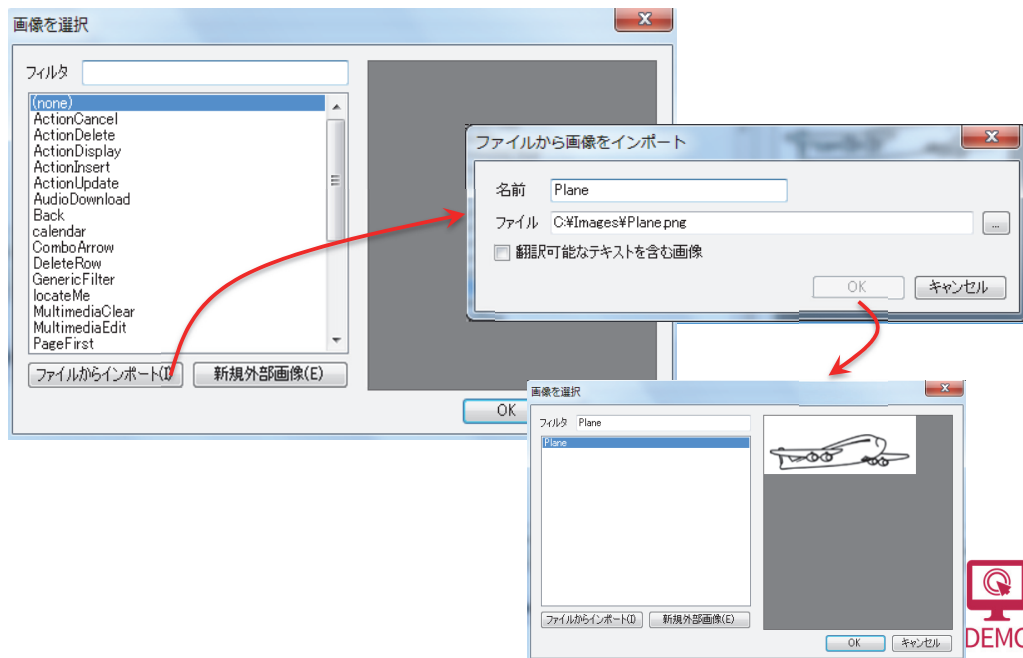
Procedure オブジェクトを作成したときに生成されたプリントブロックをタイトルと画像用に利用できます。

まずタイトルから始めましょう。ツールボックスからテキストブロックコントロールをドラッグし、[Text] プロパティに「観光名所一覧」と入力します。色も MidnightBlue に変更し、フォントのサイズを 14、フォントをボールド (Bold = True) に設定します。テキストを表示する位置を選択 ([Alignment] プロパティで設定) します。

このプリントブロックには、表示するものを表す分かりやすい名前を付けます。名前を付けるには、プリントブロックの [Name] プロパティを編集します。この例では「Title」という名前にします。

次に、飛行機の画像を左側に挿入するため、ツールボックスから目的の場所まで画像コントロールをドラッグアンドドロップします。この操作について、これから詳しく説明します。

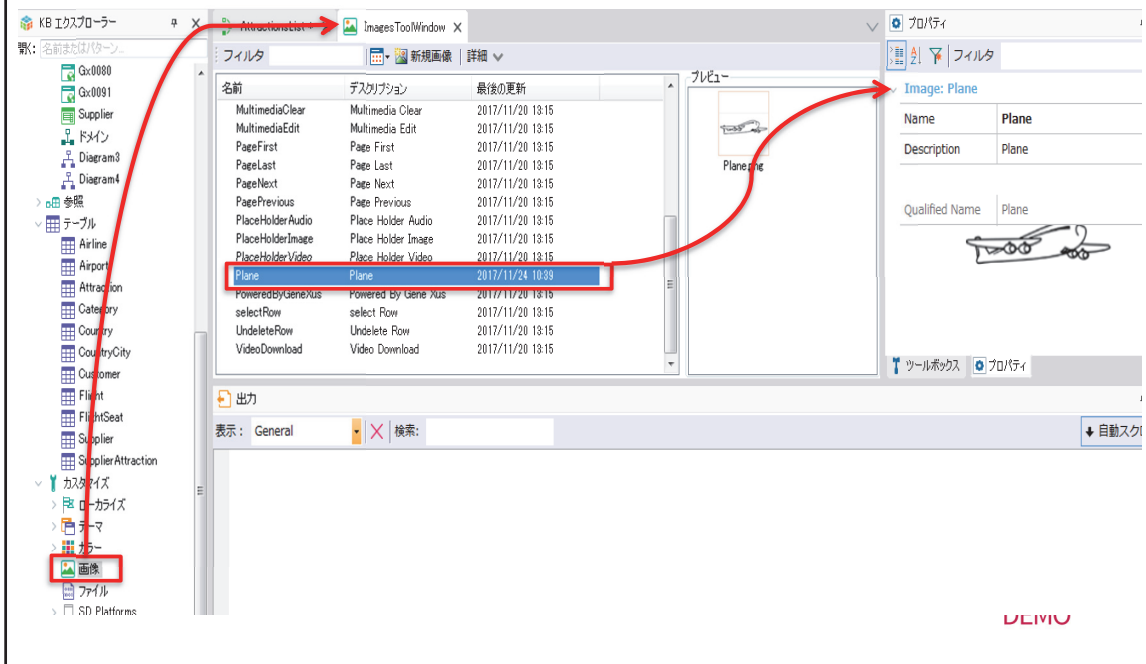
## ナレッジベース内の画像



ウィンドウが開いて、ナレッジベース内にある画像の 1 つを選択できます。または、ファイルからインポートするなどして新しい画像を追加することもできます。

[ファイルからインポート] ボタンをクリックすると、ファイルシステムを参照して画像を選択できます。選択したファイルは**画像**タイプの GeneXus オブジェクトとして作成され、イメージのファイル名が既定の名前として使用されます。この画像はナレッジベース内で自由に使用できるようになります。

## ナレッジベース内の画像



補足: ナレッジベース内のすべての画像をみるには [KB エクスプローラー] → [カスタマイズ] → [画像] からアクセスできます。上のスライドでは飛行機の画像を選択しています。

## [Layout] エlement

### 2. プリントブロック: ColumnTitles



別のプリントブロックを作成し、列のタイトルを下線付きで表示するようにします。**どれか特定のプリントブロック**を右クリックして [プリントブロックを挿入] を選択すると、新しいプリントブロックが**その下**に挿入されます。

[Layout] Element内のプリントブロックの順序で印刷されるわけではないので、ここでの順序は重要ではありません。**各プリントブロックを印刷するタイミング**はプロシージャーの [Source] Elementに入力するコードで決定します。これについて後で説明します。

新しいプリントブロックに「ColumnTitles」という名前を付け、列のタイトルに表示するテキスト用のテキストブロックを挿入します。

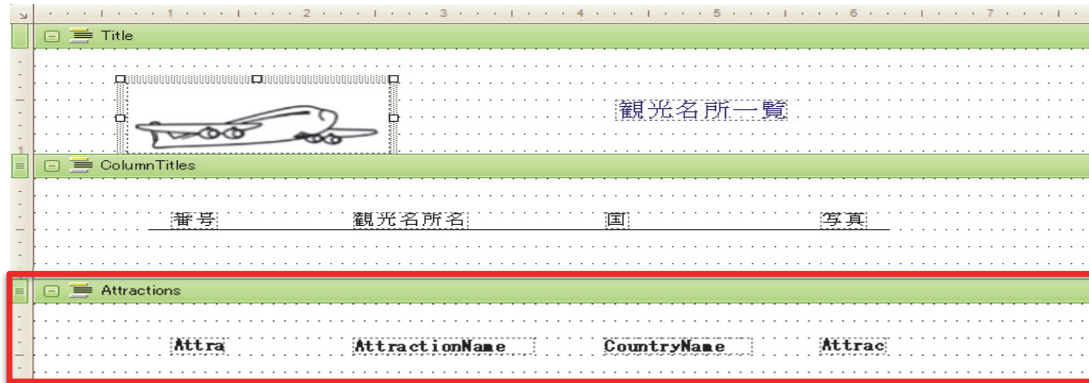
ツールボックスからテキストブロックをドラッグし、[Text] プロパティに「番号」と入力します。別のテキストブロックを追加し、[Text] プロパティに「観光名所名」と入力します。同様にしてテキストブロックを追加し、「国」と「写真」のタイトルを作成します。

これらのコントロールを目的の場所に配置します。一列に並べるには、すべてのコントロールを選択し、[メニュー] → [レイアウト] → [配置] → [下揃え] を選択します。

最後に、これらの列タイトルの下に線を挿入します。テキストボックスから「線」のコントロールをドラッグします。ここからドラッグし、必要な長さに設定します。

## [Layout] エlement

### 3. プリントブロック: Attractions



観光名所の詳細を表示するため、前述のように3 つ目のプリントブロックを追加する必要があります。新しいプリントブロックを挿入し、Attractions という名前を付けます。

データは項目属性に格納されているため、ツールボックスに戻って項目属性/変数タイプのコントロールを選択し、「番号」列見出しの下へドラッグします。

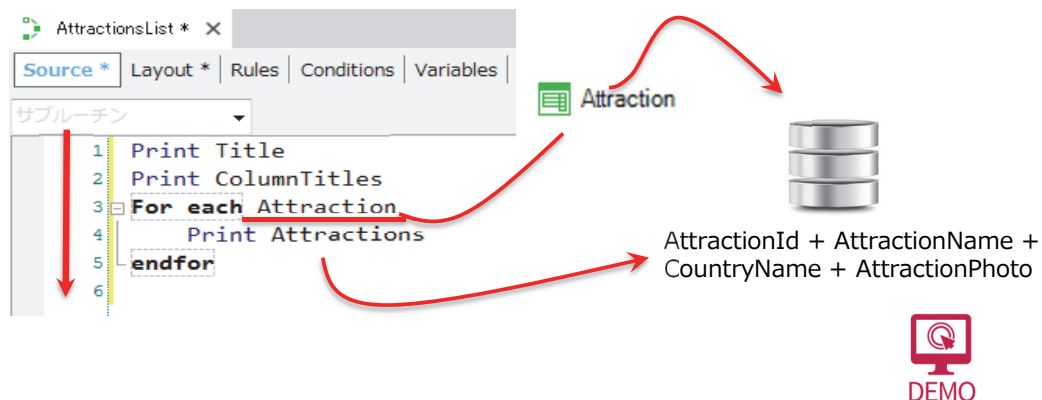
表示されたウィンドウで、このコントロールに表示する変数または項目属性を選択します。プロシージャーには、&today 以外にこれらのシステム変数があることが分かります。

**[追加] → [項目属性]** を使用してプリントブロックに項目属性を挿入することもできます。

## [Source] エlement

コマンド:

- **Print:** プリントブロックを出力
- **For each:** テーブルとその拡張テーブルを検索し、各データレコードに対して何らかの処理を実行



ここまでで、リストにデータを表示するデザインができました。

次に、データベースから適切な情報を取得し、プリントブロックの出力順序を指定するために必要なコードを入力する必要があります。  
[Source] を開きます。

最初にレポートのタイトルを出力する必要があるため、「print Title」と入力します。  
[Source] に入力した命令は上から順に実行されるため、**この命令が最初に実行されます**。つまり、「Title」というプリントブロックの内容 (リストのタイトル) を送信して印刷します。Print コマンドの後には、[Layout] で定義したプリントブロックの名前を必ず指定する必要があります。

次に列見出しを印刷するので、「ColumnTitles」プリントブロックを印刷する命令を入力します。

この 2 つの命令でレポートの固定の部分 (レポートのタイトルと飛行機の画像を含む部分と、列見出しを含む部分) の印刷順序を指定したことになります。これらはデータによって変化することはありません。

次に、データベースに保存されている観光名所のデータを出力します。観光名所のデータを出力するには、この情報が保存されている物理テーブル、つまり Attraction トランザクションに関連付けられているテーブルにアクセスする必要があります。

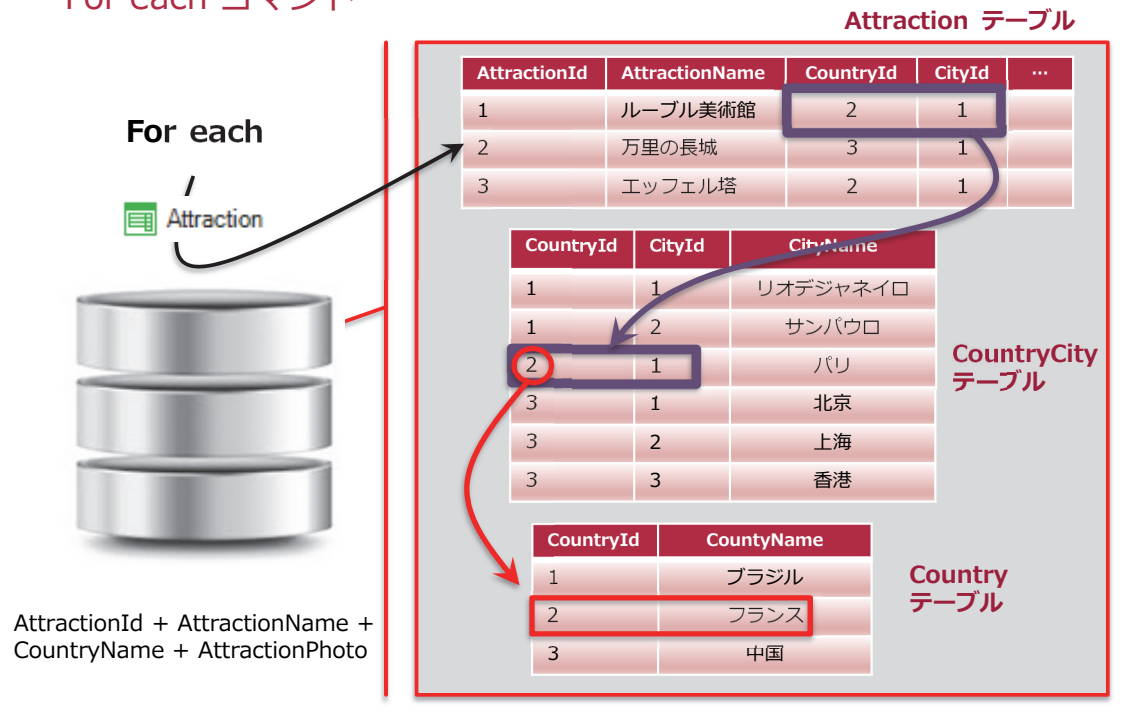
物理テーブルにアクセスするためのコマンドは For each です。アクセス先の物理テーブルは、For each コマンドのベーステーブルと呼ばれます。

For each コマンドを入力し、その隣に「Attraction」と追加します。For each コマンドの後の「Attraction」は、ナビゲートしたい Attraction **物理テーブルに関連付けられている** トランザクションの名前です。

次に、すべての観光名所について、AttractionId、AttractionName、CountryName、AttractionPhoto の各項目属性の内容を印刷するため、これらを含む「Attractions」プリントブロックを出力する命令を入力します。Print Attractions と入力します。

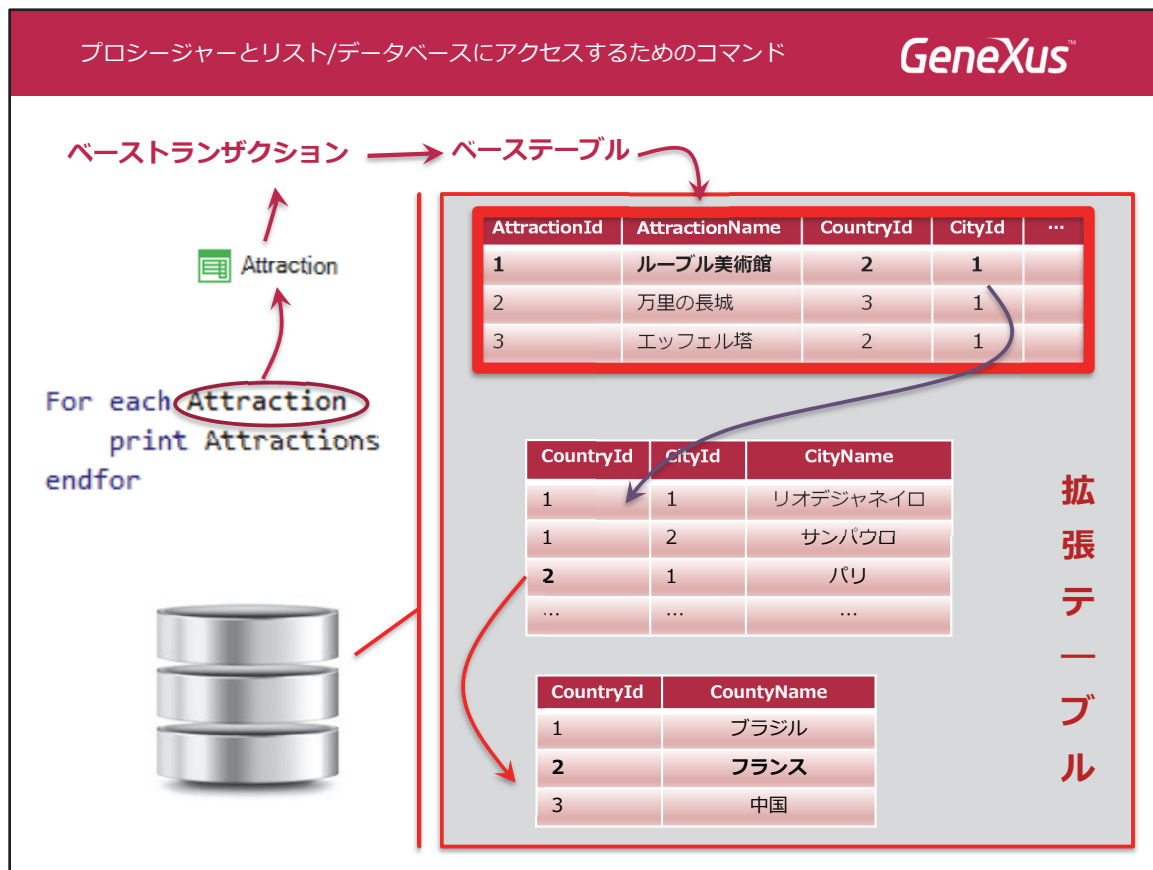


## For each コマンド



この方法で、開発者は GeneXus に、Attraction トランザクションに対応する Attraction 物理テーブルをナビゲートすることを指示できます。

For each コマンドでは Attraction テーブルと Country テーブルの項目属性を含むプリントブロックを呼び出しています。このとき各観光名所の国名を取得するために拡張テーブルの概念を適用し、Attraction テーブルの各データから CountryCity、さらに CountryCity から Country テーブルを呼び出しています。



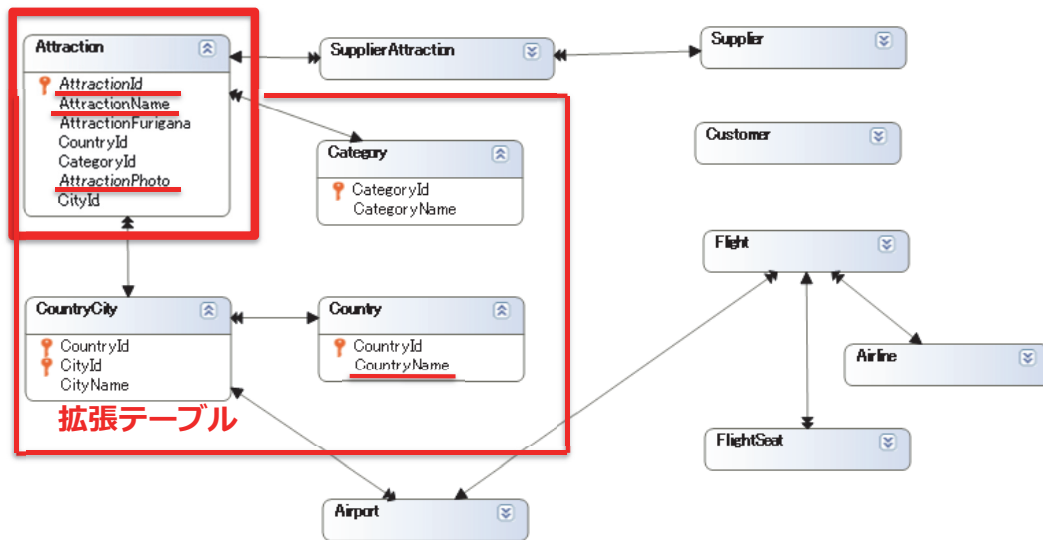
ベーストランザクションに基づき、For each コマンドでナビゲートされるベーステーブルを取得し、そのレコード群を処理 (今回の場合は、プリントブロックを出力) します。

プリントブロックには Attraction テーブルだけでなく Country テーブルの項目属性も含むため、For each コマンドで実行する Attraction テーブル内の各レコードから CountryCity のレコードにアクセスし、そこから Country テーブルの該当するレコードにアクセスして観光名所の国名を取得します。

つまり、For each で使用する各項目属性は、ベーストランザクションで指定したベーステーブルの**拡張テーブル**に含まれている必要があります。

For each コマンドのベーストランザクション、ベーステーブル、拡張テーブル

### ベーステーブル



この図は、ナレッジベースに含まれるテーブル間の関係を示しています。For each コマンドの中では、AttractionId、AttractionName、AttractionPhoto、CountryName の各項目属性を使用しています。最初の 3 つは For each のベーステーブルに含まれていますが、最後の CountryName は拡張テーブルに含まれています。

## PDF 形式でリストを生成する方法

### Procedure オブジェクトのプロパティ

**プロパティ**

フィルタ

▼ Procedure: AttractionsList

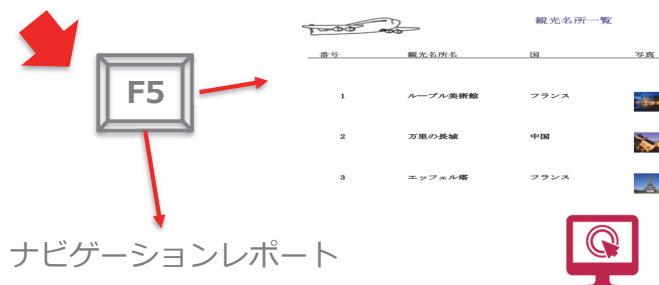
Name	AttractionsList
Description	観光名所リスト
Module/Folder	Root Module
Main program	True
Call protocol	HTTP
Execute in new	False
Qualified Name	AttractionsList
Object Visibility	Public

### ルール

AttractionsList X

Source | Layout | **Rules** | Conditions | Variables | Help | Documentation

```
1 Output_file("観光名所リスト", 'PDF');
```



実行して結果を確認しましょう。

まず、リストを PDF 形式で出力するために必要なプロパティをいくつか設定します。レポートのプロパティを開いて [Main program] プロパティを「True」に設定し、[Call protocol] プロパティで「HTTP」を選択します。

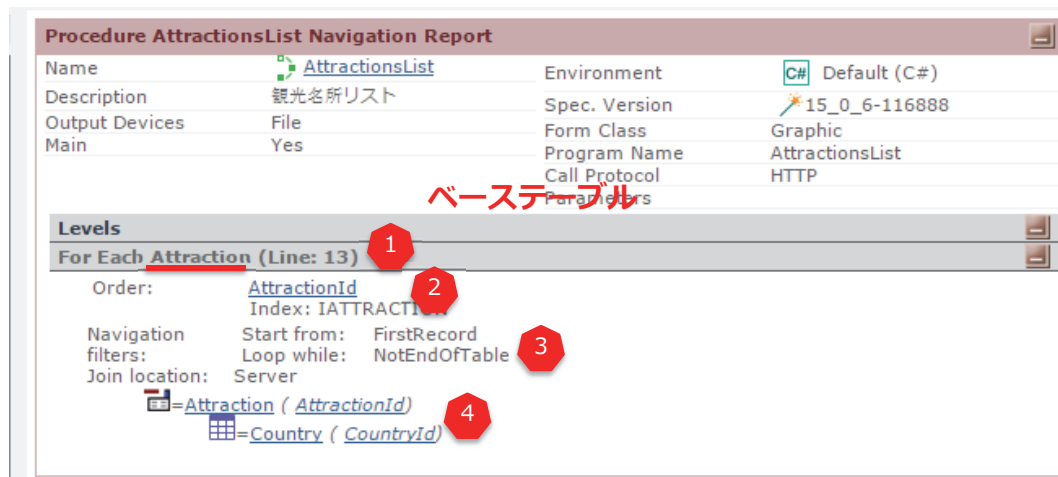
最後に、[ルール] セクションに Output\_File ルールを挿入する必要があります。トランザクションほど多くはありませんが、このタイプのオブジェクトではルールをいくつか定義できます。**[追加] → [ルール]** を選択し、ファイル名「AttractionsList.pdf」とファイル形式「pdf」を入力します。

保存し、プロシージャを実行してみましょう。

リストが作成されました。選択した形式で出力され、入力したすべての観光名所がリストに記載されています。各観光名所にはその国名と写真が表示されています。

さらに、[ナビゲーション表示] ウィンドウが開き、レポートが表示されます。

## ナビゲーションレポート



**For each コマンドで操作する物理テーブル**と、GeneXus で行ったその他の判断事項が、**プロシージャーのナビゲーションリスト**に表示されます。

ナビゲーションリストは、プロシージャーを実行すると自動的に生成されます。この場合は、F5 キーを押すと作成されます。

ナビゲーションリストには、GeneXus がデータベース内の情報にアクセスする方法が記載されています。

1. 「For each」の横に Attraction とあるので、これが **For each コマンドのベーステーブル**であることが分かります。

**For each コマンドは物理テーブル内を検索します。**ナビゲーションリストに表示される「Attraction」という名前は、プロシージャーで書いたベーストランザクションの名前ではなく、Attraction 物理テーブルを指しています。GeneXus はベーストランザクションに関連付けられているテーブルを推論します。

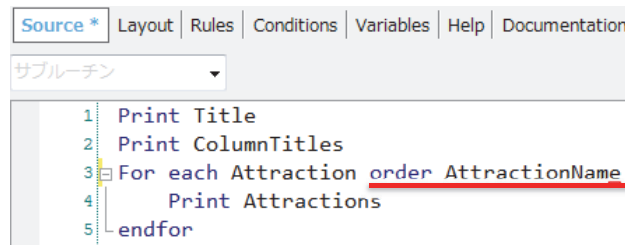
2. 観光名所のリストをソートするために、AttractionId 項目属性 (Attraction テーブルの主キー) を使用したことが示されています。

3. テーブル内のすべてのレコードを検索したことも示されています。テーブル内の最初のレコードから最後のレコードまで反復処理し、すべての観光名所を表示したことが分かります。

4. 最後に、情報を取得するために、**ナビゲート**されたテーブルは Attraction であり、そして Country に**アクセス**する必要があることを示しています。リスト内に国名を表示するためです。

## データの順序を変更する方法

- 要件: 旅行代理店が扱うすべての観光名所を、**観光名所名順で PDF ファイル**に出力する



```

Source * | Layout | Rules | Conditions | Variables | Help | Documentation
サブレーチン
1 Print Title
2 Print ColumnTitles
3 For each Attraction order AttractionName
4   Print Attractions
5 endfor
    
```

- For each でナビゲートするテーブルの**拡張テーブル**に含まれる項目属性で並べ替えできる

```

For each Attraction order CountryName
  print Attractions
endfor
    
```

残っている作業は、観光名所の名前の順 (アルファベット順) にリスト表示することです。

順序を指定するには、「For each Attraction」の横に「order AttractionName」節を記述します。

## ナビゲーションレポート

The screenshot shows the GeneXus IDE interface with the 'Navigation Report' for the 'AttractionsList' procedure. The report is divided into several sections:

- Procedure AttractionsList Navigation Report**: A table showing metadata for the procedure.
 

Name	AttractionsList	Environment	C# Default (C#)
Description	観光名所リスト	Spec. Version	15_0_6-116888
Output Devices	File	Form Class	Graphic
Main	Yes	Program Name	AttractionsList
		Call Protocol	HTTP
		Parameters	
- Warnings**: A warning message indicating a missing index for 'AttractionName'.
 

spc0038 There is no index for order AttractionName; poor performance may be noticed in group starting at line 3.
- Levels**: A section showing the 'For Each Attraction (Line: 13)' block. A red box highlights the 'Order: AttractionName ! No index' entry.
 

Order: AttractionName ! No index

Navigation Start from: FirstRecord  
 filters: Loop while: NotEndOfTable  
 Join location: Server

Below the 'For Each' block, there are two data source definitions:

```

=Attraction ( AttractionId) INTO CountryId AttractionPhoto.Uri
AttractionPhoto AttractionId AttractionName
=Country ( CountryId) INTO CountryName
            
```

The status bar at the bottom shows: エラー: 0 | 警告: 0 | 成功: 1 | すべて ▾

ここでは、リストに表示される警告を無視します。



しかし、ナビゲーションリストには、Order 節を指定した結果、出力時に項目属性がソートされたことが表示されているという点に注目してください。

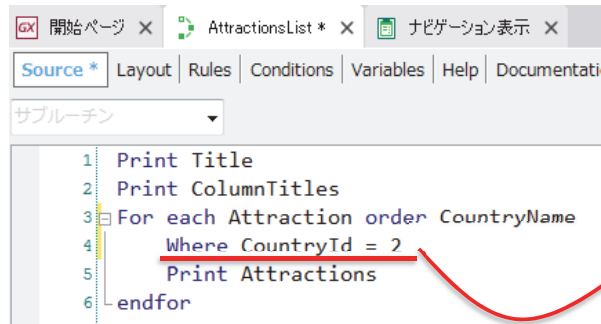
For each に「**order**」節を追加したように、For each の構文では、ほかにもいくつかの節や定義を追加できます。これから見ていきましょう。

## フィルタを定義する方法

- 要件: **フランス国内**のすべての観光名所を一覧表示する

観光名所一覧

番号	観光名所名	国	写真
3	エッフェル塔	フランス	
1	ルーブル美術館	フランス	
2	万里の長城	中国	



Where CountryName = "フランス"

ナビゲーションレポート

たとえば、旅行代理店から、フランス国内の観光名所のみを一覧表示したいと要望された場合はどうしたらよいでしょうか。

For each コマンドに **where** 節を追加して、目的の条件を満たすデータのみをフィルタして表示するようにします。

そこで、For each コマンドの次の行をクリックし、where…CountryId=2 (フランスの ID は 2) と記述します。

CountryId でフィルタする代わりに、**where CountryName="France"** と記述することもできます。



## フィルタを定義する方法: ナビゲーションレポート

フィルタ: AttractionsList

**Procedure AttractionsList Navigation Report**

Name	AttractionsList	Environment	C# Default (C#)
Description	観光名所リスト	Spec. Version	15_0_6-116888
Output Devices	File	Form Class	Graphic
Main	Yes	Program Name	AttractionsList
		Call Protocol	HTTP
		Parameters	

**Warnings**

spc0038 There is no index for order CountryName; poor performance may be noticed in group starting at line 3.

**Levels**

**For Each Attraction (Line: 13)**

Order: CountryName ! No index

Navigation Start from: CountryName = "フランス"

filters: Loop while: CountryName = "フランス"

Join location: Server

=Attraction ( AttractionId ) INTO CountryId AttractionPhoto.Uri  
AttractionPhoto AttractionName AttractionId  
 =Country ( CountryId ) INTO CountryName

エラー: 0 警告: 0 成功: 1 | すべて

Attraction テーブル全体を検索しなかったことが分かります。CountryName でソートしているため、「France」という名前の国を検索するには、テーブル全体ではなくテーブルの一部のみを検索すれば済みます。これは、「France」という言葉を辞書で探すときと同じです。辞書全体を探す必要はありません。文字「F」の部分から探せば済みます。

## 概念

For each コマンドを使用してテーブル内のすべてのレコードを検索し、関係のあるデータに対してアクションを実行

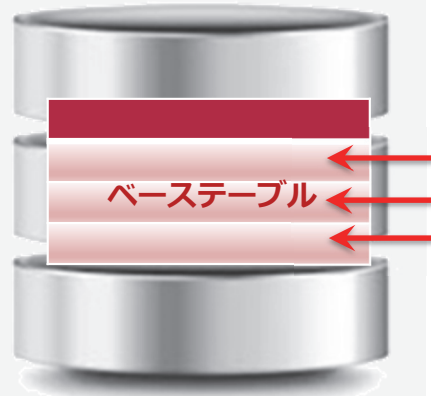
### ベーストランザクション

For each

**TransactionName.LevelName**

...

Endfor



For each コマンドを使用してテーブル内のすべてのレコードを検索し、関係のあるデータに対してアクションを実行します。

For each コマンドにはトランザクションの名前を指定します。正確には、検索対象のテーブルが関連付けられたトランザクションのレベル名を指定します。

このレベルを指定することを For each コマンドの**ベーストランザクション**と呼び、GeneXus はここから検索対象のテーブル (For each コマンドの**ベーステーブル**) を推論します。

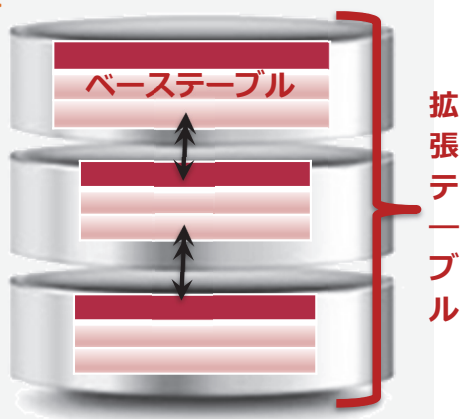
## 概念

For each <トランザクション名>.<レベル名>

...

ここに書く項目属性は、検索対象となるベーステーブルの拡張テーブルに含まれている必要がある

Endfor



For each と Endfor の間に書く項目属性は、検索対象となるベーステーブルの拡張テーブルに含まれている必要があります。

## 概念

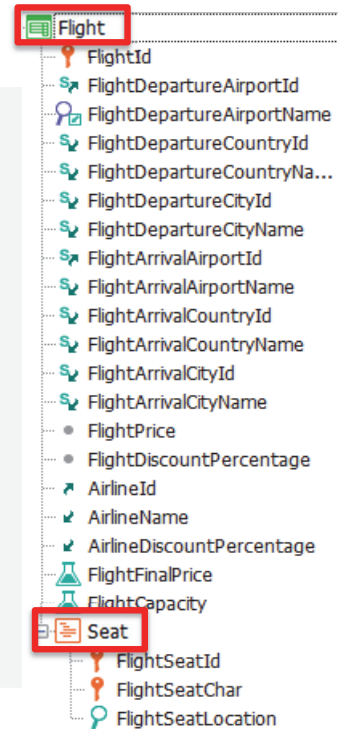
- **ベーストランザクション:**

- 最上位のトランザクションのレベル = **<トランザクション名>**

```
For each Flight
  Where FlightDepartureAirportId = 1
  ...
Endfor
```

- ネストされているトランザクションのレベル = **<トランザクション名>.<レベル名>**

```
For each Flight.Seat
  Where FlightId = 15
  ...
Endfor
```



最初の例では、AirportId=1 の空港から出発するフライトのテーブルを操作します。  
最上位のレベルでは、トランザクション名とレベル名が同じです。

2 つ目の例では、FlightId=15 のフライトの座席を操作します。

## For each 構文

```

For each <ベーストランザクション名>
  order <項目属性1>, <項目属性2>, ... , <項目属性n>
  where <条件1>
  where <条件2>
  ...
  where <条件n>
    <メインのコード>
Endfor
    
```

これは、For each コマンドについてこれまで説明したことをまとめたものです。  
ほかのトピックの説明でこの構文をさらに発展させていきます。

## For each 構文: order 節

For each <ベーストランザクション名>

**order** <項目属性<sub>1</sub>>, <項目属性<sub>2</sub>>, ... , <項目属性<sub>n</sub>>

where <条件<sub>1</sub>>

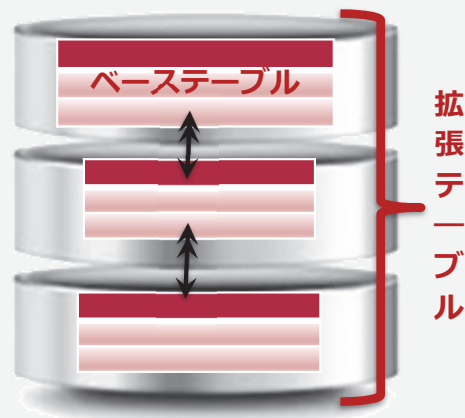
where <条件<sub>2</sub>>

...

where <条件<sub>n</sub>>

<メインのコード>

Endfor



例: order に複数の項目属性を指定した場合

**order** 節を使用して、For each コマンドで返される情報の並び順を指定できます。order 節には、For each コマンドのベーステーブルまたはその拡張テーブルに含まれる項目属性を指定できます。

ここに示したように、単一または複数の項目属性で順序を指定できます。

order に複数の項目属性を指定した例



たとえば、観光名所のカテゴリもリストに表示し、データを観光名所がある国名順でソートし、同じ国の中でもさらにカテゴリ順でソートする場合は、`order` 節に `CountryName`、`CategoryName` の順に項目属性を入力します。

`CountryName` と `CategoryName` は、Attraction ベーステーブルではなく、拡張テーブルに含まれている項目属性です。

## For each 構文: where 節



For each コマンドで返される情報をフィルタするには、**where** 節を使用します。where 節には選択するレコードの条件を指定します。

条件を複数指定したり、次のように複数の条件を and や or で結合したりできます。

- 条件<sub>a</sub> **and** 条件<sub>b</sub>: 両方の条件を同時に満たす必要があります。
- 条件<sub>a</sub> **or** 条件<sub>b</sub>: 一方の条件が満たされれば、該当するレコードとして選択されます。



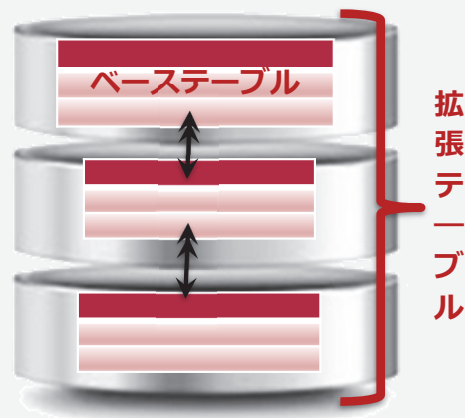
For each 構文: where 節

For each **<ベーストランザクション名>**

**order** <項目属性<sub>1</sub>>, <項目属性<sub>2</sub>>, ..., <項目属性<sub>n</sub>>

where  $\langle \text{条件}_1 \rangle$  and  
 where  $\langle \text{条件}_2 \rangle$  and  
 ... and  
 where  $\langle \text{条件}_n \rangle$  and

## <メインのコード>

**Endfor**

複数の **where** 節を追加することもできます。**where** 節を 1 つだけ追加し、複数の条件を **and** で結合した場合と同じです。

## For each 構文: メインのコード

For each <ベーストランザクション名>

order <項目属性<sub>1</sub>>, <項目属性<sub>2</sub>>, ..., <項目属性<sub>n</sub>>

where <条件<sub>1</sub>>

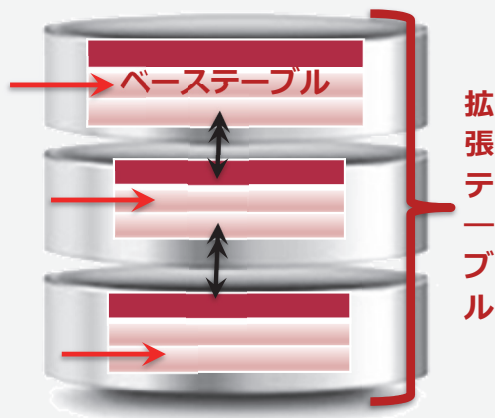
where <条件<sub>2</sub>>

...

where <条件<sub>n</sub>>

<メインのコード>

Endfor



例

For each コマンド内の**メインのコード**には、実行する一連のコマンドを順番に入力します。ベーステーブルおよび拡張テーブル内の該当レコードについて、1 行ずつ処理するのに必要な操作を記述します。

## For each 構文: メインのコード

```
print Title
print ColumnTitles
For each Attraction order CountryName, CategoryName
    print Attractions
endfor
```

Attractions			
AttractionPhoto			
Attra	AttractionName	CountryName	CategoryName



観光名所一覧

番号	観光名所名	国	写真	カテゴリ
3	エッフェル塔	フランス		モニュメント
1	ルーブル美術館	フランス		美術館
2	万里の長城	中国		

この例では、プリントブロックを出力しています。

## For each 構文

**For each** <ベーストランザクション名>

**order** <項目属性<sub>1</sub>>, <項目属性<sub>2</sub>>, ..., <項目属性<sub>n</sub>>

**where** <条件<sub>1</sub>>

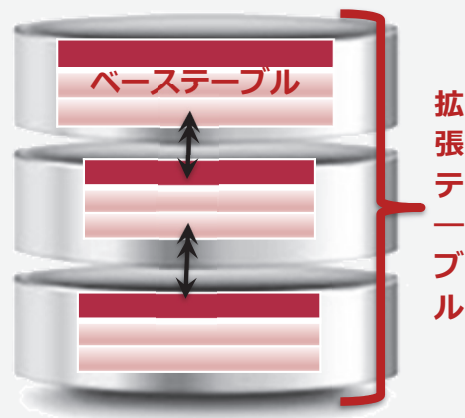
**where** <条件<sub>2</sub>>

...

**where** <条件<sub>n</sub>>

<メインのコード>

**Endfor**



これは、これまで説明してきた For each コマンドの構文をまとめたものです。

このコマンドにはさらに節やオプションを追加できます。追加可能な節やオプションについては、別の章やコースで説明していきます。