

For each、New、Delete
コマンドを使用してデータベースを
直接更新する方法

GeneXus[™]

トランザクションを実行し、対話形式のデータ入力画面によって更新を実行

カテゴリ

《 < > 》 選択

カテゴリ番号

カテゴリ名

実行 終了 削除

ここまで、データベースのデータを更新するには、次の 2 つの方法でトランザクションを使用してきました。

- 画面でのトランザクションの実行、対話形式のデータ入力

トランザクションに関連付けられたビジネスコンポーネントを通じて更新

The screenshot displays the GeneXus IDE interface. The top window, titled 'Category', shows the 'Structure' tab with a table-like view of the business component. The bottom window, titled 'InsertCategoryUpdateAttractions', shows the 'Variables' tab with a list of variables: 'Attraction' (Type: Attraction, Is Collection: ☐) and 'Category' (Type: Category, Is Collection: ☐). The 'Properties' window on the right shows the 'BusinessComponent: Category' properties, with 'Business Component' set to 'True' (highlighted by a red box). The 'Code' window on the right shows the following code:

```
&Category.CategoryName = "人気/おすすめ"  
&Category.Save()
```

別の方法として、関連付けられたビジネスコンポーネントを通じ、変数を利用することで、画面を使用せずに更新を実行できます。

挿入を実行するには New コマンドを使用

```
1 □ New
2     CategoryID =5
3     CategoryName = "人気/おすすめ"
4 - Endnew

6 □ New
7     CategoryName = "人気/おすすめ"
8 - Endnew
9
```

Procedure オブジェクトに対してのみ有効

ほかにも、データベースに対して挿入、変更、削除を実行する方法があります。
この方法は、Procedure オブジェクトに対してのみ有効です。

挿入を実行するには、New コマンドを使用します。

ここでは、Category テーブルに新しいカテゴリを挿入します。CategoryId が自動採番の場合は、CategoryId の値は入力しません。

データを編集するには For each コマンドを使用

```
6 □ For each Attraction
7   where CityName = "東京" and CategoryName ="有名なランドマーク"
8   CategoryID = find(CategoryID, CategoryName="人気/おすすめ")
9 Endfor
```

Procedure オブジェクトに対してのみ有効

更新操作の場合は For each コマンドを利用できます。たとえば、北京の「遺跡」カテゴリに属するすべての観光名所を「観光地」カテゴリに変更するには、For each コマンドを入力して、更新する項目属性に新しい値を割り当てます。

ここでは、ベーステーブル上の項目属性の更新を行っていますが、拡張テーブルの項目属性でも更新可能です。

データを削除するには Delete コマンドを使用

```
For each Attraction  
  Delete  
Endfor
```

Procedure オブジェクトに対してのみ有効

削除操作の場合は、削除するレコードを指定して、Delete コマンドを利用できます。

プロシージャ内でコマンドを使用するメリットとデメリット

- デメリット
 - B Cとは異なり自動的に参照整合性をチェックしないので一貫性やDBエラーを考慮する必要がある。
 - トランザクションルールがトリガーされないため、データの一貫性が失われる可能性がある
- メリット
 - 処理が速い (高パフォーマンス)

ビジネスコンポーネントを使用する方法と比較して、コマンドを使用した場合、「参照整合性を制御するメカニズムが存在しない」、「トランザクションルールがトリガーされない」というデメリットがあります。

一方で、「処理が速い」というメリットがあり、「高パフォーマンス」であると言えます。

たとえば、数百万件のレコードがあるテーブル全体からデータを削除する必要がある場合、Delete コマンドを使用すると処理時間を大幅に短縮できます。ただし、この削除処理は制御されておらず、データベースが一貫性のない状態に陥る可能性があります。

