

## 利用可能なデータベース更新方法

ビジネスコンポーネントを使用する理由

GeneXus™

## トランザクション: 新規登録、更新、削除

観光名所

観光名所番号

観光名所名

国番号

国名 フランス

都市番号

都市名

カテゴリ番号

カテゴリ名 美術館

観光名所写真

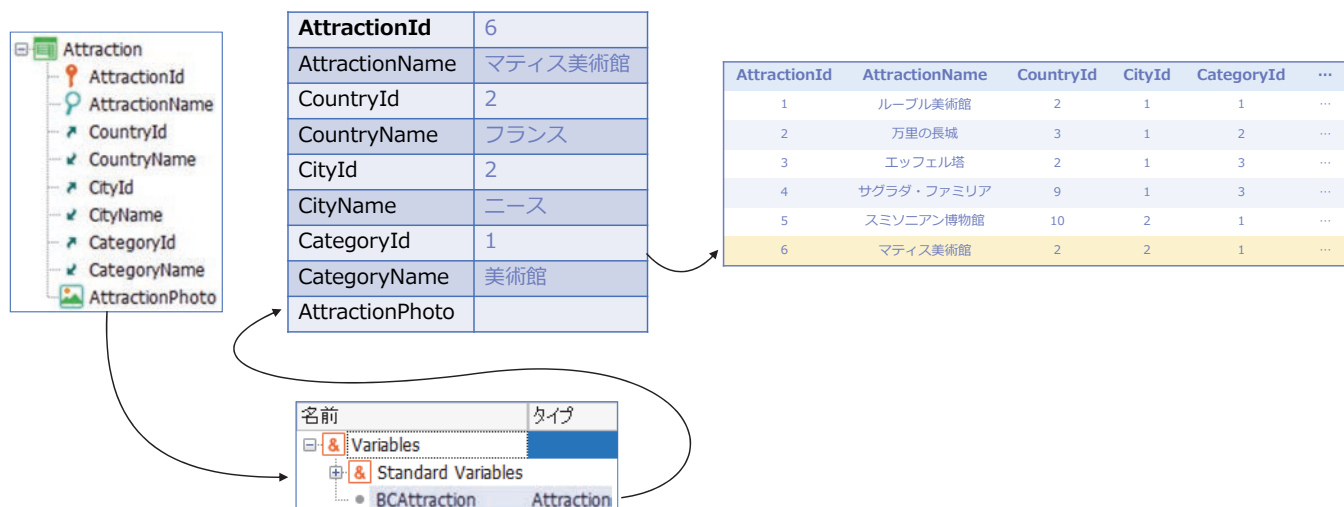
削除 終了 実行

AttractionId	AttractionName	CountryId	CityId	CategoryId	...
1	ルーブル美術館	2	1	1	...
2	万里の長城	3	1	2	...
3	エッフェル塔	2	1	3	...
4	サグラダ・ファミリア	9	1	3	...
5	スミソニアン博物館	10	2	1	...
6	マティス美術館	2	2	1	...

ここまでで説明してきたデータベースに登録されているデータの新規登録、更新、削除を実行する方法は、トランザクションオブジェクトの定義に基づき、自動で生成された Web アプリケーションの画面の利用でした。

つまり、GUI (グラフィカルユーザーインターフェース) を利用し、対話形式による操作でした。

## ビジネスコンポーネント: Insert(), Update(), Delete()



GeneXus で、データベースに登録されたデータを更新する方法として、コーディングによる実装も可能です。

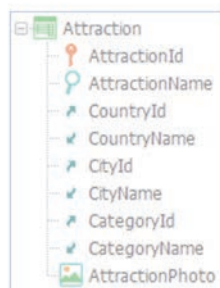
コーディングによる実装は、2 つの方法が用意されています。

1 つ目の方法は、「ビジネスコンポーネント」と呼ばれる機能を利用する方法となります。GeneXus として、コーディングによる実装で初めに検討いただきたい機能です。この資料で機能の概要及び推奨している理由について説明します。

このスライドでは、簡単に機能のポイントを列記します。

- ・対象トランザクションの構造を SDT オブジェクトのように変数で利用
- ・対象トランザクションのルールを考慮
- ・データの新規登録、更新、削除が可能
- ・GUI は持たない

## プロシージャー: New、For each、Delete



```
New
    AttractionId = 6
...
Endnew
```

AttractionId	AttractionName	CountryId	CityId	CategoryId	...
1	ルーブル美術館	2	1	1	...
2	万里の長城	3	1	2	...
3	エッフェル塔	2	1	3	...
4	サグラダ・ファミリア	9	1	3	...
5	スミソニアン博物館	10	2	1	...
6	マティス美術館	2	2	1	...

もう 1 つの方法は、プロシージャーオブジェクトのみで利用可能なコマンドを利用する方法です。

この方法は、別の資料で説明します。

1 点特徴を上げますが、この機能を利用した場合、トランザクションオブジェクトからは独立し、直接テーブルを操作することができます。

ただし、デメリットもあり、この点も機能の説明時に合わせて説明しています。

## 並行トランザクション

Attraction		
名前	タイプ	DESCRIPTION
Attraction	Attraction	観光名所
AttractionId	Id	観光名所番号
AttractionName	Name	観光名所名
CountryId	Id	国番号
CountryName	Name	国名
CityId	Id	都市番号
CityName	Name	都市名
CategoryId	Id	カテゴリ番号
CategoryName	Name	カテゴリ名
AttractionPhoto	Image	観光名所写真

AttractionWithoutPattern		
名前	タイプ	DESCRIPTION
AttractionWithoutPattern	AttractionWithoutPattern	観光名所(パターンなし)
AttractionId	Id	観光名所番号
AttractionName	Name	観光名所名
CountryId	Id	国番号
CountryName	Name	国名
CityId	Id	都市番号
CityName	Name	都市名
CategoryId	Id	カテゴリ番号
CategoryName	Name	カテゴリ名
AttractionPhoto	Image	観光名所写真



テーブル: ATTRACTION

この資料では、既に定義済みのトランザクションを利用しますが、これまでに定義された内容を含めず利用したいため、トランザクションオブジェクトを「名前を付けて保存」を利用し、複製します。

本資料の本題から少し外れ、「並行トランザクション」という概念について補足します。この概念の詳細については、別のコースで取り扱っています。

トランザクションオブジェクトを複製した場合、[Structure] エLEMENTで、主キー項目属性に変更がない場合、テーブルは、2 つのトランザクションオブジェクトで共有されます。

つまり、どちらのトランザクションオブジェクトから生成された画面からも同じテーブルのデータを操作することができます。

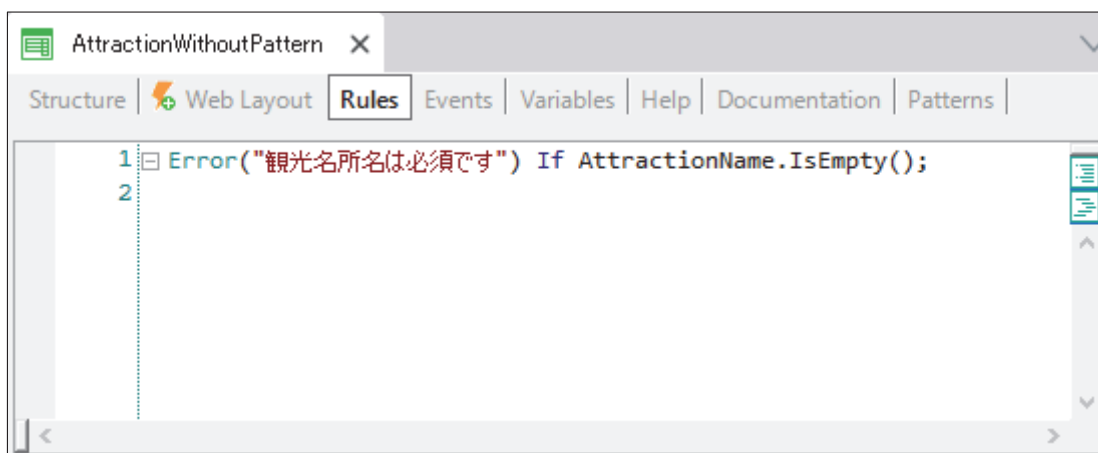
ただし、別のオブジェクトが実装されているため、Web アプリケーションとしてのプログラムは、それぞれのオブジェクトごとに生成されます。

この結果、1 つのテーブルのデータを操作できる画面が 2 つ作成された状態です。

それぞれのトランザクションで定義されたルールなどは、オブジェクトにローカルとなるため、一方で定義したエラールールは、もう一方で記述されていなければ、適用されません。

そのため、別の画面ではエラーとなるデータも、別の画面では登録できるという実装が可能となります。

## トランザクションによる新規登録



ビジネスコンポーネント機能の説明を進めます。  
この機能の挙動を理解するためには、トランザクションオブジェクトによる標準の GUI を利用したデータ操作をイメージすることが重要となります。  
そのため、改めて GUI を利用したデータ操作について説明します。

## GUI を利用した登録

観光名所 (パターンなし)

観光名所番号  選択

観光名所名

国番号  0 ☐

国名

都市番号  0 ☐

都市名

カテゴリ番号  0 ☐

カテゴリ名

観光名所写真

削除 終了 実行

AttractionId	AttractionName	CountryId	CityId	CategoryId	...
1	ルーブル美術館	2	1	1	...
2	万里の長城	3	1	2	...
3	エッフェル塔	2	1	3	...
4	サグラダ・ファミリア	9	1	3	...
5	スミソニアン博物館	10	2	1	...
6	マティス美術館	2	2	1	...

トランザクションオブジェクトによって生成された画面では、すべての項目が空の値で表示されます。

あらかじめフォーカスされている項目は、主キー項目の入力フィールドです。

このフィールドに入力された値で、アプリケーションは、データの「新規登録」または「更新」を決定します。

フィールドからフォーカスが離れたタイミングで、入力された値のレコード有無を検索し、ある場合、「更新」として登録済みのデータを他のフィールドに読み込みます。ない場合、「新規登録」としてすべてのフィールドが空のまま新しいデータの入力を開始できます。

もちろん、新規登録時に実行を想定したルールがある場合、実行され、一部のフィールドに既定値の代入などが行われます。

## エラールールの実行

観光名所 (パターンなし)

◀ ◻ ▶ ▶| 選択

観光名所番号 ◻ 0

観光名所名 ◻ ❗ 観光名所名は必須です

国番号 ◻ 0 

国名

都市番号 ◻ 0 

都市名

カテゴリ番号 ◻ 0 

カテゴリ名

観光名所写真  

◻

削除 終了 実行

もし、エラールールが定義されている場合、ルールの条件を満たした時点で、画面上にはエラーメッセージが表示されます。  
このような状態でも、このメッセージを無視し、他のフィールドの入力を続けることが可能でした。

エラーメッセージが表示されている場合、継続できない操作としては、実行ボタン押下によるデータの保存です。



## 参照整合性エラー：未入力

観光名所（パターンなし）

◀ ◯ ▶ >| 選択

観光名所番号

観光名所名

国番号   国'の該当レコードはありません。

国名

都市番号  

都市名

カテゴリ番号  

カテゴリ名

観光名所写真  

削除

終了

実行

外部キー項目属性の場合、GeneXus は、入力された値に対し、参照整合性のチェックを自動で行います。

既定の定義で実装している場合、値を未指定のままにすることができません。

そのため、空の値を指定した場合でも、この値をキーとしたレコードのチェックを行うため、該当レコードが存在しないことを示すエラーメッセージが表示されます。

このエラーメッセージの場合も、前述のルールによるメッセージ同様に、他のフィールドの入力を続けることができます。

## 参照整合性エラー：該当レコードなし

観光名所 (パターンなし)

K < > >I 選択

観光名所番号

観光名所名

国番号

国名 中国

都市番号

都市名 北京

カテゴリ番号  ● カテゴリの該当レコードはありません。

カテゴリ名  → このカテゴリは存在しない

観光名所写真

前のスライドで説明した外部キーの値を未指定にできないケースですが、トランザクションオブジェクトの定義において、[Null 許容] 列を Yes に変更することで、対応可能でした。

ただし、該当するレコードがない場合に、登録ができないようにエラーメッセージが表示する挙動については、変更できません。

値の未指定が許可されたフィールドでももちろん、入力された外部キーの値を主キーの値とするレコードがない場合、エラーメッセージが表示されます。

外部キーに入力された値に基づき、外部キーの従属項目の値を読み込む処理も自動で実装されています。

この処理は、内部的にデータベースと通信を行い、該当するレコードの情報を取得しています。

これらの挙動に基づき、実装された画面は、UX（ユーザーエクスペリエンス）を向上させます。

内部的にサーバーとの通信は発生しますが、クライアントとなるブラウザで処理が行われます。

そのため、これらは、「クライアント側の検証」と呼ばれ、実行ボタンをクリックした後に行われる「サーバー側の検証」においても再度実行されます。

## 登録時のエラー：サーバー側の検証

観光名所 (パターンなし)

K < > | 選択

観光名所番号

観光名所名  ❗ 観光名所名は必須です

国番号

国名 中国

都市番号

都市名 北京

カテゴリ番号

カテゴリ名 遺跡

観光名所写真

AttractionId	AttractionName	CountryId	CityId	CategoryId	...
1	ルーブル美術館	2	1	1	...
2	万里の長城	3	1	2	...
3	エッフェル塔	2	1	3	...
4	サグラダ・ファミリア	9	1	3	...
5	スミソニアン博物館	10	2	1	...
6	マティス美術館	2	2	1	...



AttractionId	0
AttractionName	
CountryId	3
CountryName	
CityId	1
CityName	
CategoryId	2
CategoryName	
AttractionPhoto	<画像データ>



→  
サーバー

ルールや、参照整合性に基づき、エラーメッセージが発生したままでも、実行ボタンを押すことが出来ました。  
この場合、画面上で入力された値で、サーバー側で処理を実行するようにプログラムが実行されます。

ただし、「サーバー側の検証」が実行され、画面上に表示されているエラーがこの検証でも発生するため、テーブルへのデータの登録処理は行われません。

## 新規登録時の処理

観光名所 (パターンなし)

K < > >| 選択

観光名所番号

観光名所名

国番号

国名 中国

都市番号

都市名 北京

カテゴリ番号

カテゴリ名 遺跡

観光名所写真

削除 終了 実行

AttractionId	AttractionName	CountryId	CityId	CategoryId	...
1	ルーブル美術館	2	1	1	...
2	万里の長城	3	1	2	...
3	エッフェル塔	2	1	3	...
4	サグラダ・ファミリア	9	1	3	...
5	スミソニアン博物館	10	2	1	...
6	マティス美術館	2	2	1	...

問題がなければ  
データベースに登録する

AttractionId	0
AttractionName	紫禁城
CountryId	3
CountryName	
CityId	1
CityName	
CategoryId	2
CategoryName	
AttractionPhoto	<画像データ>



サーバー

エラーが発生しない状態で、実行ボタンをクリックした場合も、すべてのフィールドに入力された値がサーバーへ送信されます。  
サーバー側で改めて検証が行われ、内容に問題がないことが確認されます。  
具体的な流れについて記載します。

サーバーに送信された値については、SDT タイプの 1 つの入れ物に、すべての値が格納されているような実装になります。  
ただし、実際に送信される値は、対象のテーブルに物理的に存在する項目のみです。  
つまり、外部キーに基づく従属項目やグローバル式は該当しません。

この送信された値を利用し、サーバー側の検証として、クライアント上で実行された処理と同じ検証が行われます。  
検証の結果、エラーが発生しない場合、テーブルに値が登録されます。

## 登録完了後の内部処理

観光名所 (パターンなし)

K < > >| 選択

観光名所番号

観光名所名

国番号

国名 中国

都市番号

都市名 北京

カテゴリ番号

カテゴリ名 遺跡

観光名所写真

AttractionId	AttractionName	CountryId	CityId	CategoryId	...
1	ルーブル美術館	2	1	1	...
2	万里の長城	3	1	2	...
3	エッフェル塔	2	1	3	...
4	サグラダ・ファミリア	9	1	3	...
5	スミソニアン博物館	10	2	1	...
6	マティス美術館	2	2	1	...
7	紫禁城	3	1	2	...

↓

AttractionId	7
AttractionName	紫禁城
CountryId	3
CountryName	中国
CityId	1
CityName	北京
CategoryId	2
CategoryName	遺跡
AttractionPhoto	<画像データ>

テーブルへの登録が完了した場合、サーバーに送信された入力データの構造のうち、外部キーの従属項目やグローバル式の項目属性の値も格納されます。同様に、主キー項目属性が自動採番を有効化していた場合、登録された番号が格納されます。

これは、登録したデータを後続の処理で利用する可能性を考慮した挙動です。

もし、対象のトランザクションに第 2 レベルの定義があった場合、この後、第 2 レベルの入力 1 行ごとに同様の処理が行われます。

この場合について、本コースでは、詳細は扱っていません。

## 登録の完了状態

観光名所（パターンなし）

● データが追加されました。

◀ ◻ ▶ ◻▶ 選択

観光名所番号


観光名所名

国番号  



国名

都市番号  

都市名

カテゴリ番号  

カテゴリ名

観光名所写真  

AttractionId	AttractionName	CountryId	CityId	CategoryId	...
1	ルーブル美術館	2	1	1	...
2	万里の長城	3	1	2	...
3	エッフェル塔	2	1	3	...
4	サグラダ・ファミリア	9	1	3	...
5	スミソニアン博物館	10	2	1	...
6	マティス美術館	2	2	1	...
7	紫禁城	3	1	2	...

AttractionId	
AttractionName	
CountryId	
CountryName	
CityId	
CityName	
CategoryId	
CategoryName	
AttractionPhoto	

内部処理が完了すると、画面上には、データの登録が成功したことを示すメッセージが表示されます。

また、画面上のすべてのフィールドは、値がクリアされ、画面をはじめて表示した際と同じ状態になります。

この場合、サーバーへの登録処理でメモリ領域に格納されていた登録データは、新規登録を開始すると、破棄されます。

## データの更新

観光名所 (パターンなし)

K < > | 選択

観光名所番号

観光名所名

国番号

国名 中国

都市番号

都市名 北京

カテゴリ番号

カテゴリ名

観光名所写真

AttractionId	AttractionName	CountryId	CityId	CategoryId	...
1	ルーブル美術館	2	1	1	...
2	万里の長城	3	1	2	...
3	エッフェル塔	2	1	3	...
4	サグラダ・ファミリア	9	1	3	...
5	スミソニアン博物館	10	2	1	...
6	マティス美術館	2	2	1	...
7	紫禁城	3	1		...

AttractionId	7
AttractionName	紫禁城
CountryId	3
CountryName	中国
CityId	1
CityName	北京
CategoryId	
CategoryName	
AttractionPhoto	<画像データ>



新規登録の操作で説明を始める前に紹介していましたが、主キー項目のフィールドに既存レコードの値を入力すると、データが読み込まれ、データを更新することができる状態になります。

データを読み込む際には、新規登録の処理が行われる際に、サーバーのメモリ領域が利用されましたが、更新時には、テーブルのデータをクライアントに読み込むために、このメモリ領域へ格納が行われます。

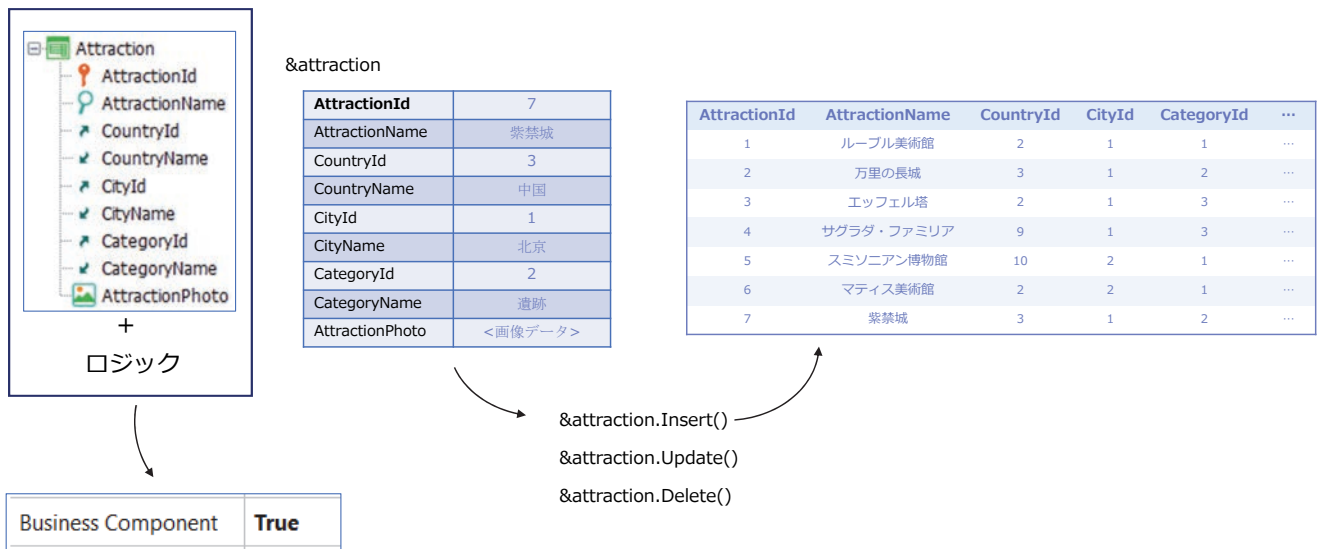
画面に読み込まれたデータは、変更することができ、実行ボタンをクリックすることで、新規登録時と同様の内部処理が行われ、データが更新できます。

データの更新時の場合、画面の表示は維持され、継続してデータの更新が可能な状態となります。

また、データを更新するために読み込んだ場合、削除ボタンが活性化され、データの削除が可能となります。

ここまでに紹介したトランザクションオブジェクトによるデータ操作のうち、GUI 部分を除外し、ルールや、内部処理のロジックを再利用できる変数が定義できれば、冒頭に紹介した「ビジネスコンポーネント」そのものになります。

## ビジネスコンポーネント



GeneXus におけるビジネスコンポーネント機能は、まさにこの説明の通りです。対象とするトランザクションオブジェクトの [Business Component] プロパティを True に変更することで、ナレッジベース内で、トランザクションオブジェクトと同名のデータタイプが利用可能となります。

このデータタイプを設定した変数に基づき、ビジネスコンポーネントという機能を利用できます。

この変数では、SDT タイプを指定した変数と同様に、トランザクションオブジェクトの構造がメンバーとして利用できる構造化データが扱える変数となります。そして、SDT タイプと異なる点として、トランザクションのロジックを再利用したメソッドが用意され、データの新規登録、更新、削除が変数を通して実行可能です。

トランザクションのロジックを再利用することができるため、コーディングによるデータの操作として利用を推奨しています。

次の資料で、この機能の利用について詳細を説明します。



*GeneXus*<sup>TM</sup>

[training.genexus.com](http://training.genexus.com)  
[wiki.genexus.com](http://wiki.genexus.com)