



GeneXus™

One Day トレーニング

ハンズオンでの GeneXus 開発の概要説明

Copyright © GeneXus S.A. 1988-2024.

All rights reserved. 本書は、GeneXus S.A.の明示的許可なしにはいかなるメディアにも複写することはできません。本書の内容は個人的使用のみを目的として提供するものです。

登録商標:

GeneXus は GeneXus S.A.の商標または登録商標です。本書において取り上げているその他の商標はすべて、それぞれの所有財産です。

目次

はじめに	3
ナレッジベースの作成：GeneXus の開始	5
トランザクションの作成：入力項目の定義	7
アプリケーションの実行	12
ダイアグラム①：定義構造の確認	16
トランザクションの変更①：データのマスタ化	19
トランザクションの変更②：画像項目の追加	27
トランザクションの変更③：ビジネスロジックの実装	31
トランザクションの変更④：計算項目の追加	35
ダイアグラム②：定義構造の変化の確認	39
Web パネルの作成：一覧画面の実装	42
プロシージャの作成①：PDF 出力	46
プロシージャの作成②：データの一括処理	54
デザインシステム：画面デザインのカスタマイズ	60
Work With Plus for Web：より効率的なアプリケーションの作成	65
その他の製品・機能	71
次のステップへ	72

はじめに

GeneXus は、たった数行のプログラミングで簡単にアプリケーションを作ることができる「ローコード開発ツール」です。

GeneXus には次のような特長があります。習得が簡単で、生産性が高く、クロスプラットフォームで、将来にわたって使用可能であるため、デジタル資産を守ることができ、新しい技術も容易に取り入れることができます。

テンプレートに沿って作成するため、少しいれとクリック操作で、すばやく簡単にアプリケーションを作ることができます。

本来、何千・何万行ものプログラミングコードを書かなければいけないところ、GeneXus はたった数行でアプリケーションを作ることができるのです。

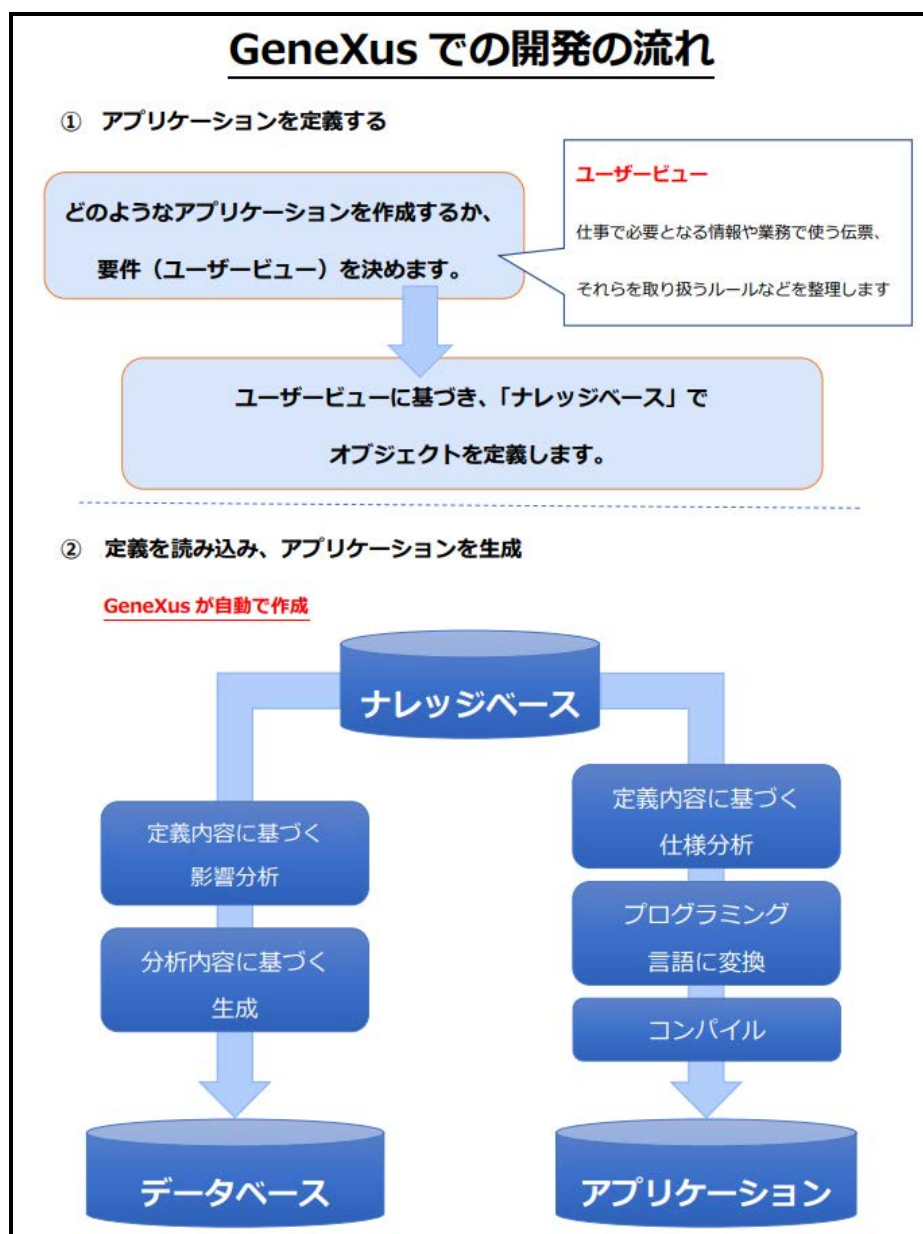
GeneXus は、簡単で自由度が高く、非常にバランスが取れている唯一無二の開発ツールになっています。

GeneXus は、定義された内容を.NET や Java などのプログラミング言語として生成します。

アプリケーションで利用するデータベースも自動で生成し、サポートしている製品として

Microsoft SQLServer や Oracle Database、PostgreSQL や MySQL など、幅広く対応しています。

また、Web アプリケーションだけではなく、iOS や Android などのスマートデバイス向けアプリケーションを作ることができます。



GeneXus によるアプリケーション開発がどのようなものか、学習・体験してみましょう。

今回は、「**薬局の商品を管理するアプリケーション**」を開発します。

ナレッジベースの作成 : GeneXus の開始

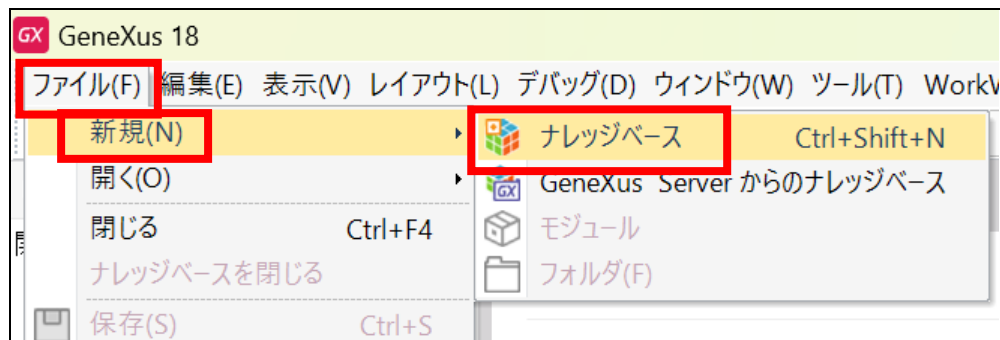
GeneXus を起動すると、以下のような画面が表示されます。



まず、ユーザービューに基づくアプリケーション定義を行うため、「ナレッジベース」を作成しましょう。

GeneXus における「ナレッジベース」は、一般開発における「プロジェクト」と同等に考えていただけます。

- ① ツールバーから、**【ファイル】** → **【新規】** → **【ナレッジベース】** と選択



② 表示される「新規ナレッジベース」ダイアログにおいて、以下の内容でナレッジベースを作成

- ・ 名前 : **PharmacySystem**
- ・ プロトタイプ環境 : **.NET**
- ・ データソース : **SQL Server**

新規ナレッジベース

名前(N): PharmacySystem

場所(L): C:¥KBs

基本 詳細

プロトタイプのターゲット(T)
ローカル

ユーザーインターフェースの言語(U)
Japanese

バックエンド

プロトタイプ環境(E)
.NET

データソース(D)
SQL Server

フロントエンド

☒ .NET Web (.NET)
☐ Android
☐ Apple
☐ Web (Angular)

作成 キャンセル

③ 作成ボタンをクリック

ナレッジベース作成において既定の設定やオブジェクトのインポートが実施され、処理が完了することで、ナレッジベースが作成されます。

トランザクションの作成：入力項目の定義

一般的な IT 用語において「トランザクション」とは、「複数の一連の処理」を意味するものですが、GeneXus においては、「オブジェクト」と呼ばれる機能テンプレートの 1 つを指し、アプリケーションにおいてデータの登録処理にかかわる機能を実装できるオブジェクトとなります。

それでは、「トランザクション」を作成し、データの入力項目を定義しましょう。

- ① ツールバーから、**【ファイル】** → **【新規】** → **【オブジェクト】** と選択
- ② 表示される「新規オブジェクト」ダイアログにおいて、以下の内容でトランザクションを作成
 - ・カテゴリを選択「**データ管理**」 → タイプを選択「**Transaction**」
 - ・名前：**Product**
 - ・デスクリプション：**商品**

新規オブジェクト

カテゴリを選択(S):

- データ管理
- ユーザーインターフェース
- BPM
- Chatbot
- リソース
- ドキュメンテーション
- 拡張性
- デブロイ
- テスト
- レポートイング
- すべて

タイプを選択(T):

- API
- Data Provider
- Data Selector
- Data View
- Domain
- Procedure
- Structured Data Type
- Subtype Group
- Transaction

データベース構造、ビジネスルール、およびデータ操作の UI を定義し、実際のオブジェクトやアクターを記述します。

名前(N): Product

デスクリプション(D): 商品

フォルダ(F): Root Module

作成 キャンセル

③ 作成ボタンをクリック

④ 以下の内容で、項目を定義

名前	タイプ	デスクリプション	式	Null...
Product	Product	商品		
ProductCode	Numeric(10.0)	商品コード		No
ProductName	Character(50)	商品名称		No
ProductPrice	Numeric(9.0)	商品価格		No

	名前	タイプ	デスクリプション
1 行目	ProductCode	Numeric(10.0)	商品コード
2 行目	ProductName	Character(50)	商品名称
3 行目	ProductPrice	Numeric(9.0)	商品価格

※入力のポイント

1. 「名前」の入力で、（ドット）キーを押すことでトランザクション名が自動で入力されます
2. Tab キーを押すことで右の入力列へ移動できます
(例：「名前」で Tab キーを押すと「タイプ」に入力フォーカスが移動)
3. 入力が確定された状態（入力フォーカス状態ではない）で Enter キーを押すと、
次の項目入力行が表示されます

トランザクション定義におけるタイプ列について

トランザクションで定義する項目には、どのような種類の値を入力できるかを指定するために「タイプ」列で「データタイプ」を指定しています。

Product トランザクションの定義では「**Numeric**」と「**Character**」を利用しています。

Numeric は、「数」という意味です。タイプを **Numeric** に設定すると、**数値だけが入力できる項目**になります。

Character は、「文字」という意味です。タイプを **Character** に設定すると、**文字列として値が入力できる項目**になります。

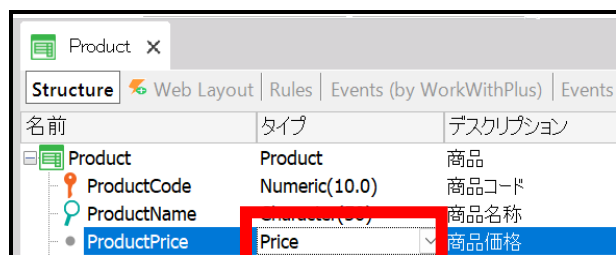
Character(**50**) Numeric(**10.0**)など、かっこ内の数字は、**入力可能な文字数（桁数）**を表しています。

⑤ 3行目の **ProductPrice** のタイプを「**Price=Numeric(9.0)**」に変更

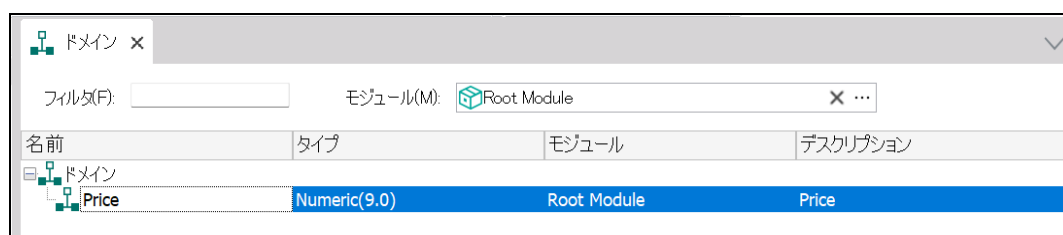
名前	タイプ	デスクリプシ...	式	Null...
Product	Product	商品		
ProductCode	Numeric(10.0)	商品コード		No
ProductName	Character(50)	商品名称		No
ProductPrice	Price=Numeric(9.0)	商品価格		No

「ドメイン」について

タイプを **Price=Numeric(9.0)**に変更すると、「Price」という表示になります。



これは、**Price** という「ドメイン」が追加され、このドメインを利用する定義になったことを表します。



※ドメインは、KB エクスプローラー内の「ドメイン」から確認することができます。

ドメインを使用するメリットは、**ドメインのデータタイプを変更するだけで、そのドメインを使用している全ての項目属性のデータタイプを一括で変更することができる点**です。

例えば、後になって価格の桁数を 10 桁に変更することになった場合、**ドメインにおいて Price の「タイプ」列でデータタイプを「Numeric(10.0)」と変更するだけで、ナレッジベース内の「タイプ」列が Price と設定されている全ての項目属性のデータタイプを Numeric(10.0)に一括で変更することができます。**

追加したドメインはナレッジベース内固有のデータタイプで、「PharmacySystem」内だけで使用することができます。

⑥ 以下の内容で、項目を追加で定義

Product * X				
Structure * Web Layout Rules Events (by WorkWithPlus) Events Variables Help Documentation Patterns				
名前	タイプ	デスクリプション	式	Null 許容
Product	Product	商品		
ProductCode	Numeric(10.0)	商品コード		No
ProductName	Character(50)	商品名称		No
ProductPrice	Price	商品価格		No
ProductStock	Numeric(4.0)	商品在庫		No
ProductType	Character(50)	商品種別		No

	名前	タイプ	デスクリプション
4 行目	ProductStock	Numeric(4.0)	商品在庫
5 行目	ProductType	Character(50)	商品種別

これで、Product トランザクションが完成しました。

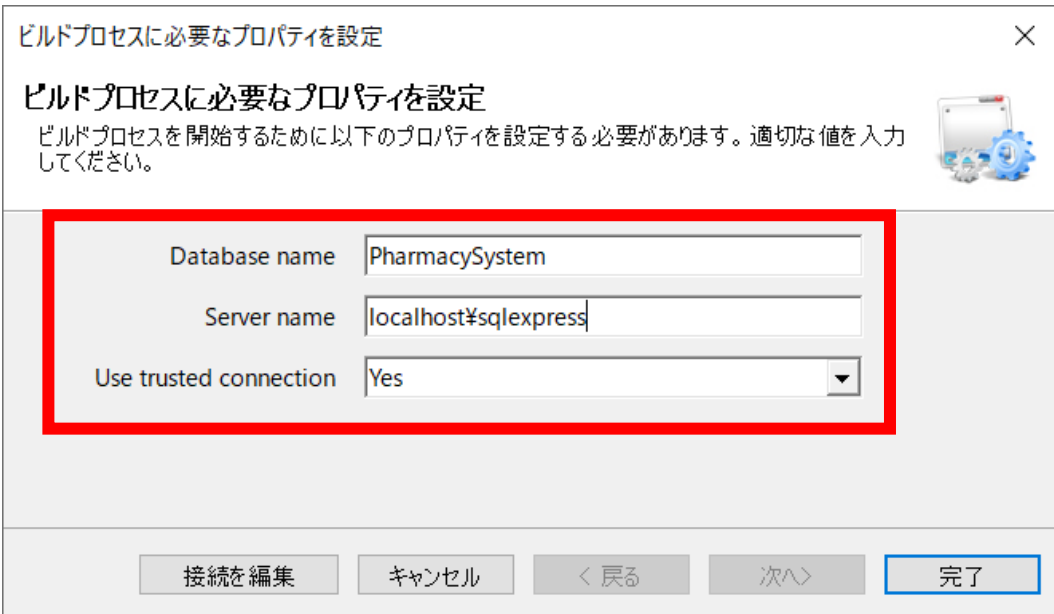
アプリケーションの実行

それでは、アプリケーションを実行してみましょう。

- ① メニューバーの [表示] → [その他のツールウィンドウ] → [設定] をクリック
- ② 表示された「設定」ウィンドウで「バックエンド」ノードをダブルクリックして展開
- ③ 展開された中に含まれる「Default (.NET)」ノードを右クリックし、「プロパティ」を選択
- ④ 表示された「プロパティ」ウィンドウを下へスクロールし、「Web Server」プロパティを「**Internet Information Server**」に変更
- ⑤ F5 キーを押して、実行

⑥ 「ビルドプロセスに必要なプロパティを設定」ダイアログ（下図）において、以下の内容を指定

- Database name : **PharmacySystem**
- Server name : **localhost¥sqlexpress**
- User trusted connection : **Yes**



ビルドプロセスに必要なプロパティを設定

ビルドプロセスに必要なプロパティを設定

ビルドプロセスを開始するために以下のプロパティを設定する必要があります。適切な値を入力してください。

Database name PharmacySystem

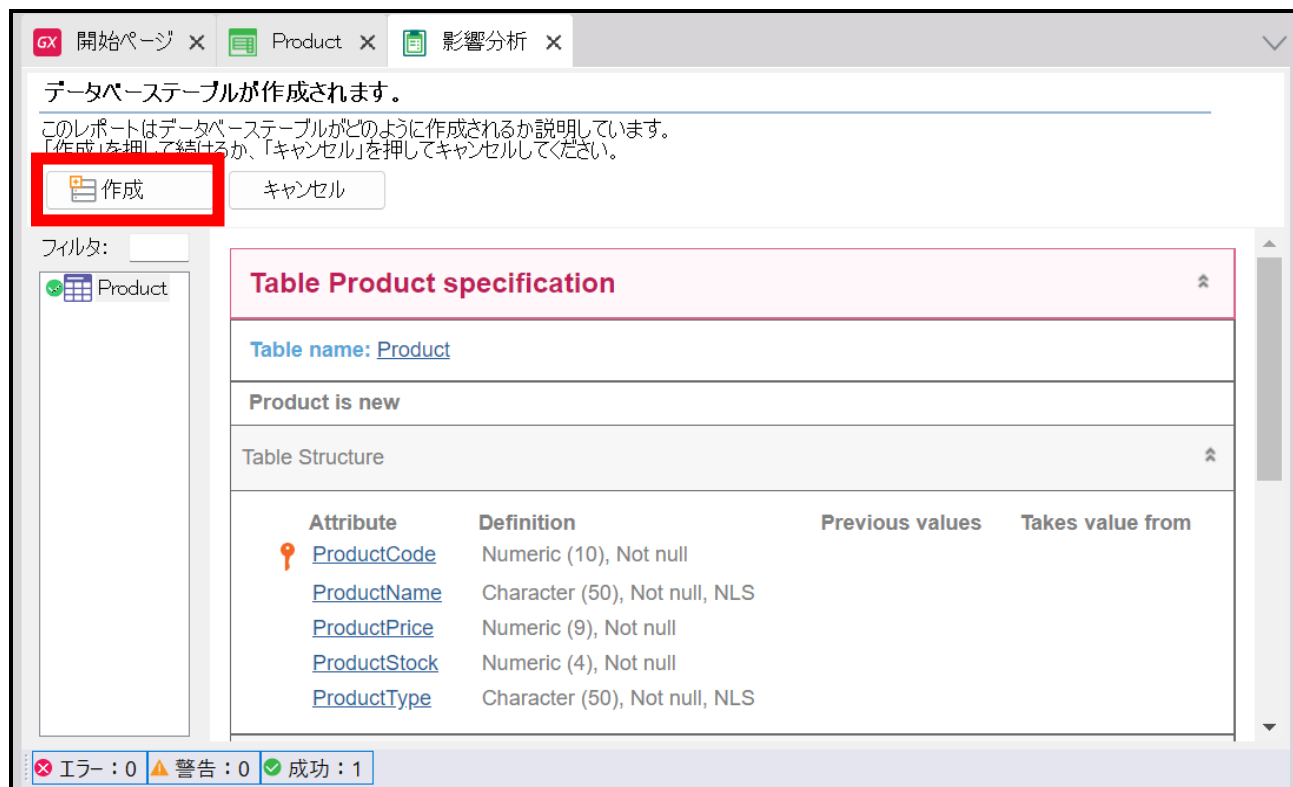
Server name localhost¥sqlexpress

Use trusted connection Yes

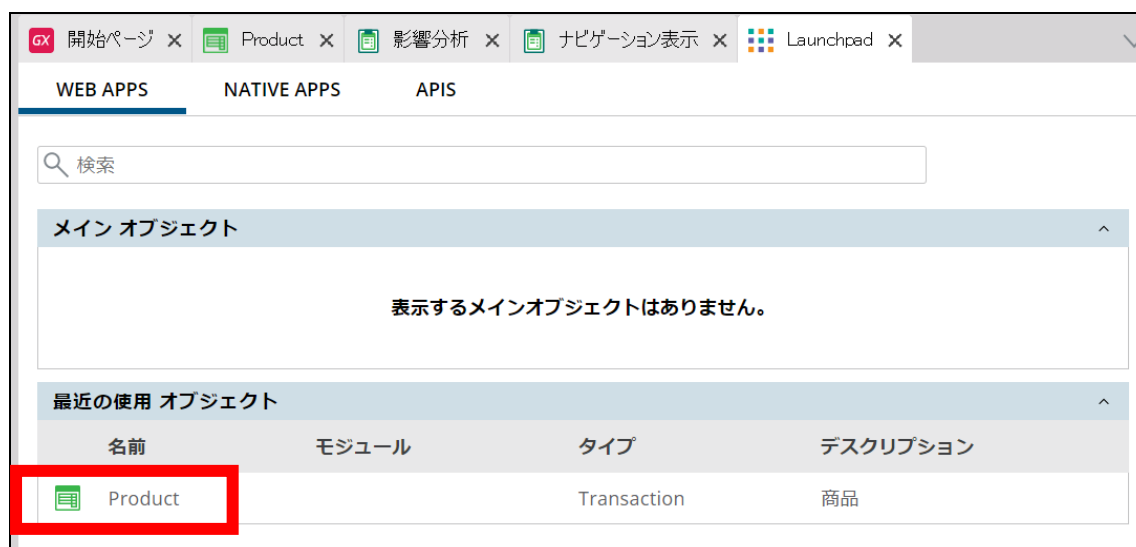
接続を編集 キャンセル < 戻る 次へ > 完了

⑦ 完了ボタンをクリック

- ⑧ 以下の画面が表示されるので、作成ボタンをクリック



- ⑨ Launchpad から、**Product** をクリック



アプリケーションが実行され、Product トランザクションが開きました。

The screenshot displays the GeneXus application interface. The top header bar is dark blue with the GeneXus logo on the left and 'Application Name' in the center. Below the header, a dark blue sidebar on the left contains the text 'UNANIMO'. The main content area has a light gray background and is titled '商品' (Product). Inside this area, there is a white form with several input fields: '商品コード' (Product Code) with a value of '0', '商品名称' (Product Name), '商品価格' (Product Price) with a value of '0', '商品在庫' (Product Inventory) with a value of '0', and '商品種別' (Product Type). Above the '商品コード' field, there are navigation arrows and the text '選択' (Select). At the bottom right of the form area, there are three buttons: '削除' (Delete), '終了' (End), and '実行' (Execute).

この画面から、データの登録・編集・削除をすることができます。

ダイアグラム①：定義構造の確認

「ダイアグラム」は、トランザクションで定義された構造を確認することができます。

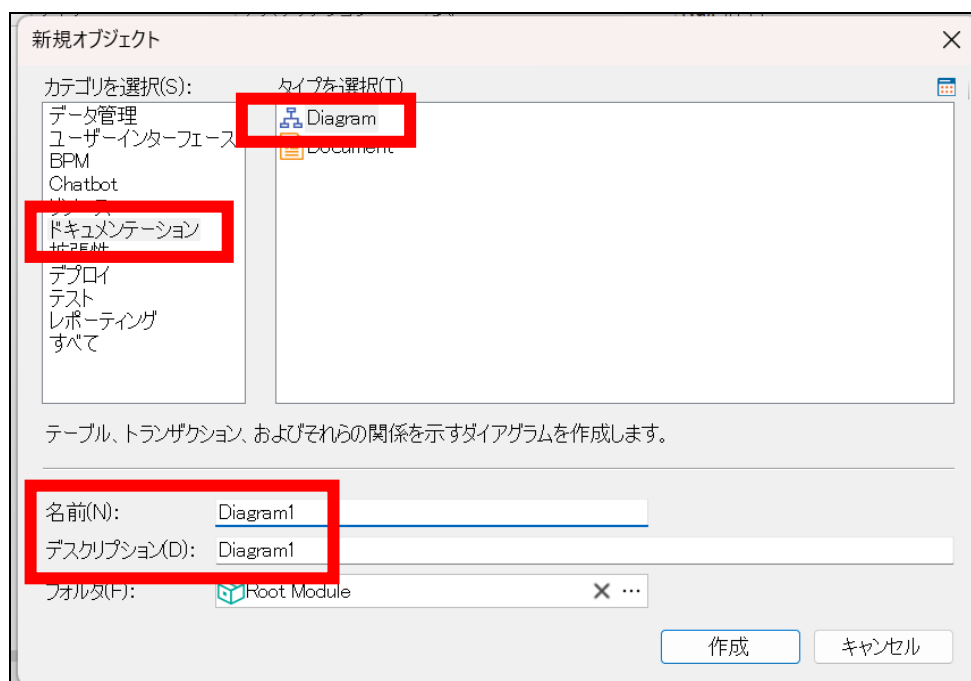
① ツールバーから、**【ファイル】** → **【新規】** → **【オブジェクト】** と選択

② 表示される「新規オブジェクト」ダイアログ（下図）において、以下の内容でダイアグラムを作成

・カテゴリを選択「**ドキュメンテーション**」 → タイプを選択「**Diagram**」

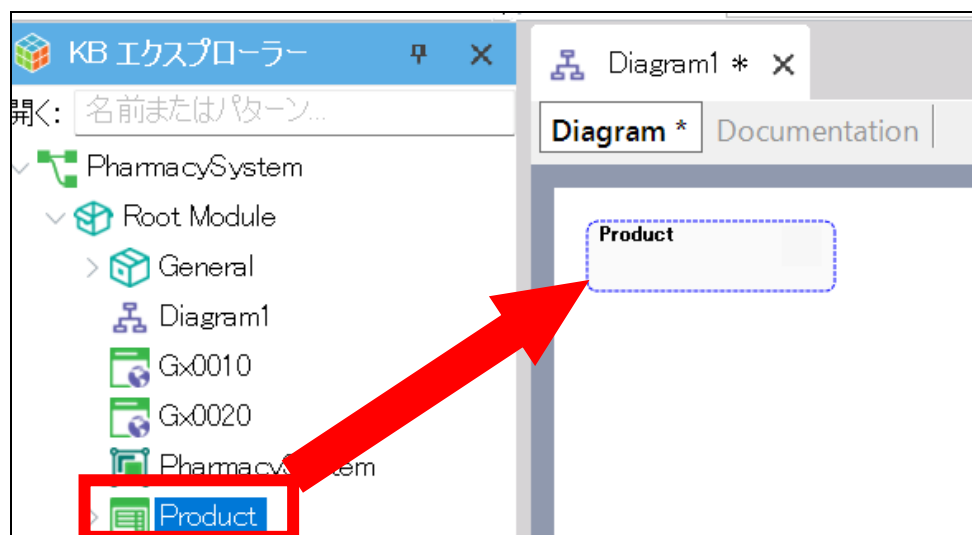
・名前：**Diagram1**

・デスクリプション：**Diagram1**



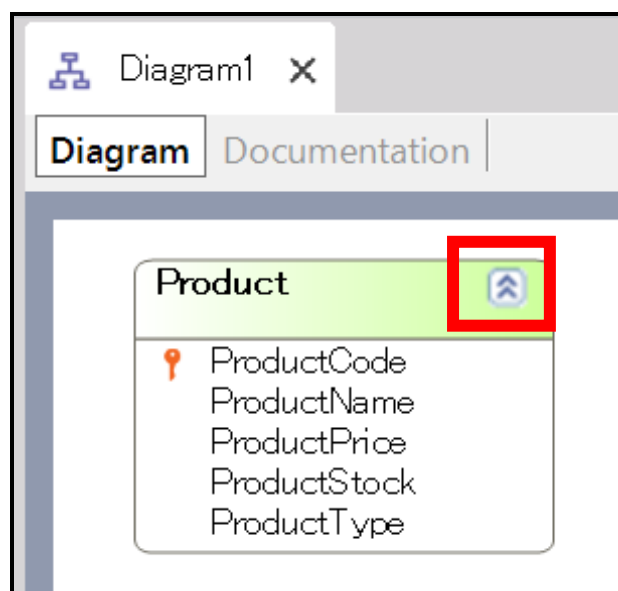
③ 作成ボタンをクリック

- ④ KB エクスプローラー内から「Product」をドラッグアンドドロップし、Diagram1 に配置



- ⑤ Ctrl+S を押し、保存

Product の右の矢印ボタンをクリックすると、トランザクションの項目を表示することができます。



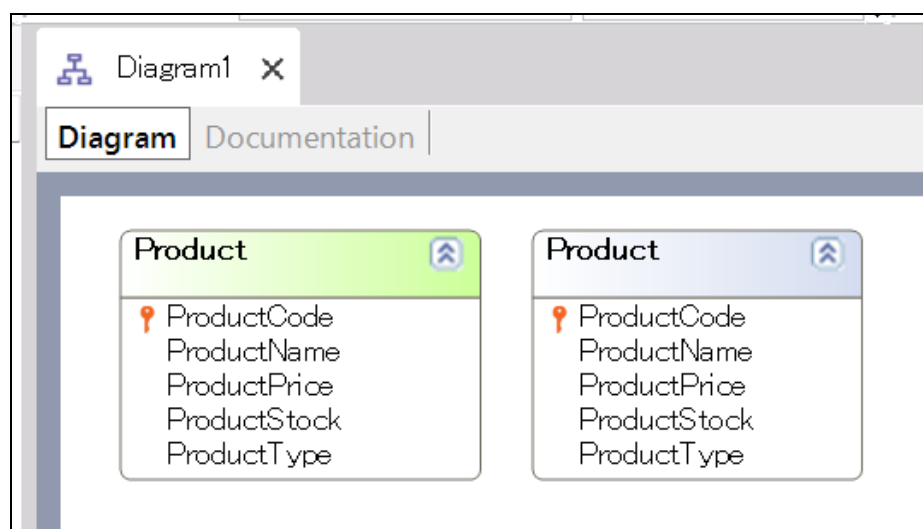
- ⑥ ツールバーから、[表示] → [テーブル] と選択

- ⑦ KB エクスプローラーに表示された「テーブル」ノード内の PRODUCT (※) テーブルをドラッグアンドドロップし、Diagram1 に配置

※トランザクションと同名になるため、資料内で区別するために大文字で標記しています。

- ⑧ Ctrl+S を押し、保存

PRODUCT の右の矢印ボタンをクリックすると、テーブルの項目を表示することが出来ます。



この時点では、トランザクションとテーブルで含まれている項目に差異はありません。

また、他に参照元、参照先もないため、特に関係性については確認できません。

この点については後ほど確認します。

トランザクションの変更①：データのマスタ化

ここまでで、Product トランザクションの作成ができ、商品データの登録・編集・削除ができるようになりました。

このデータには「商品種別」という項目属性がありますが、商品データの入力時に手入力することが出来るため、すべてのユーザーで統一された「商品種別」データとならない可能性があります。

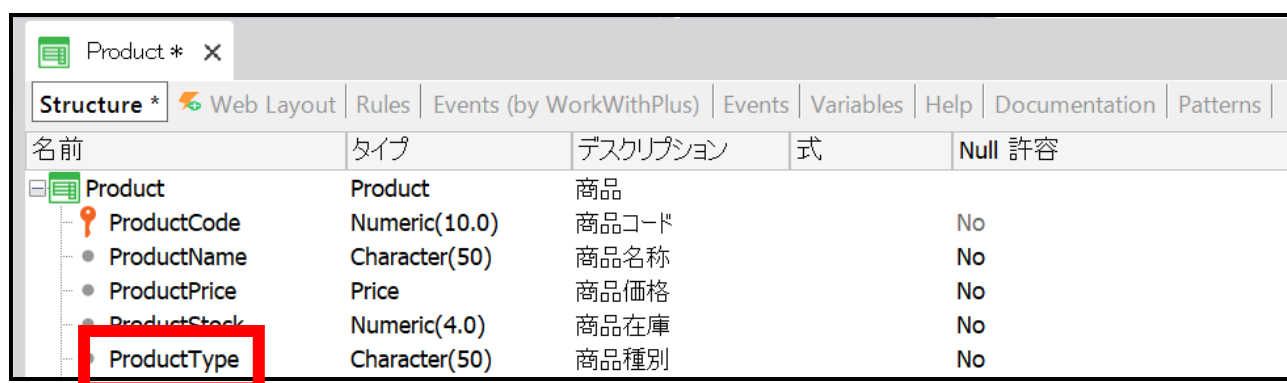
つまり、1つの商品に対して各ユーザーが意図した「商品種別」は同じでも、実際に登録された「商品種別」データの内容が異なることが想定されます。

この結果、もし「商品種別」項目属性をキーとした検索機能を追加した場合、意図した結果が表示できません。

そこで、予め統一された商品種別データの中から当てはまるものを選択するようにすれば、この問題を解決できるのではないのでしょうか。

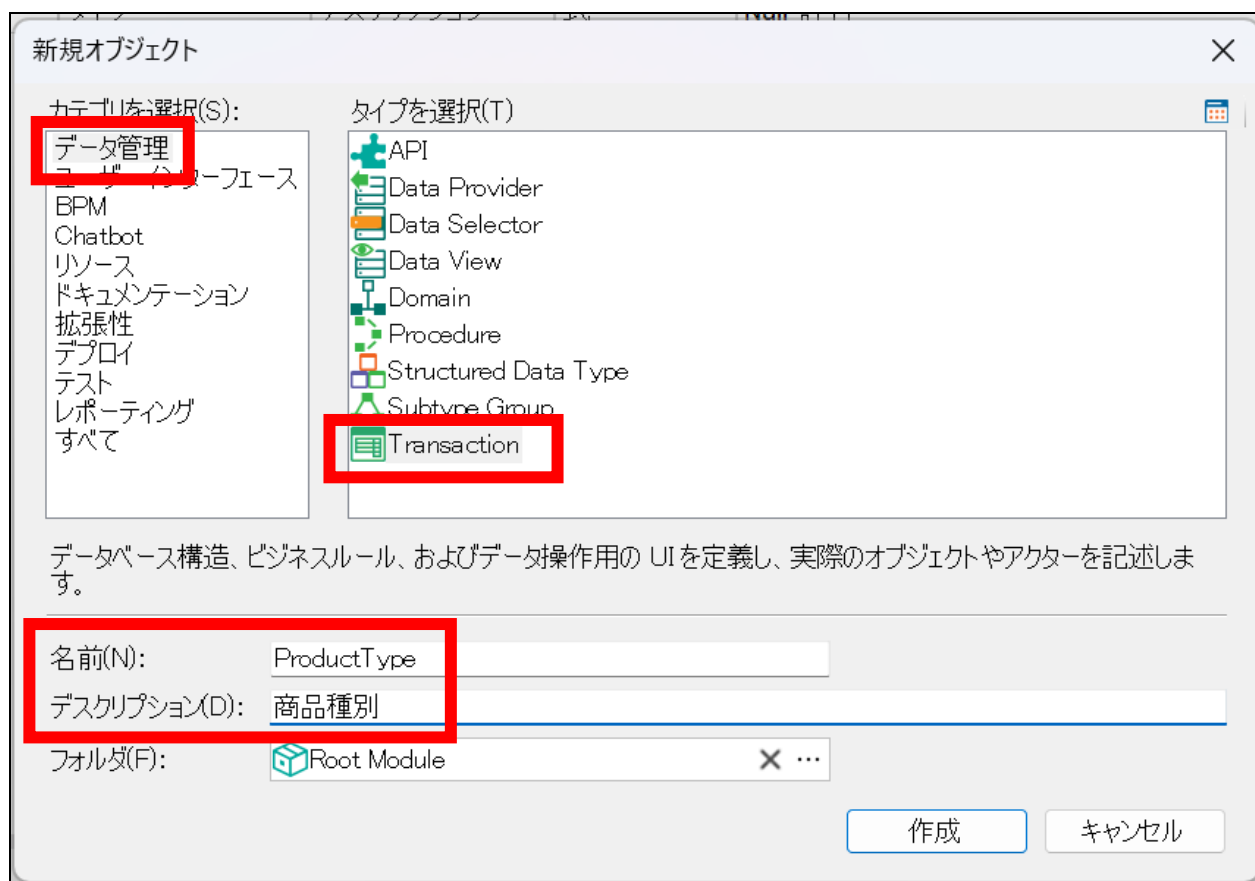
商品種別データを選択形式にするため、事前にマスターデータとして登録できるように ProductType トランザクションを新たに作成し、Product トランザクションから参照できるように変更しましょう。

① Product トランザクションの「ProductType」を右クリック → 削除



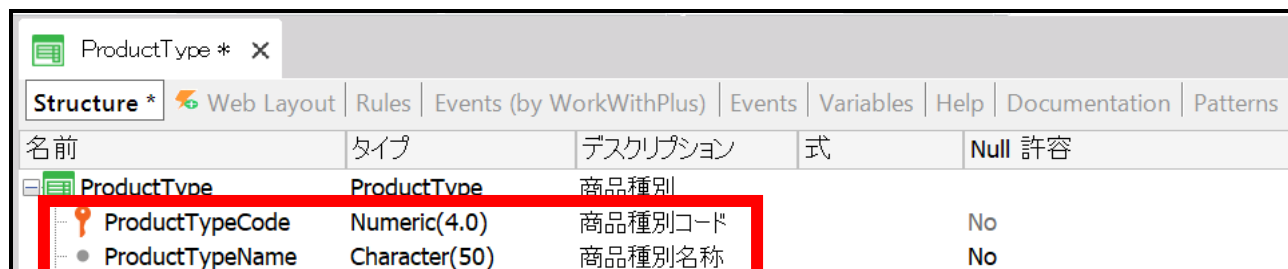
名前	タイプ	デスクリプション	式	Null 許容
Product	Product	商品		
ProductCode	Numeric(10.0)	商品コード		No
ProductName	Character(50)	商品名称		No
ProductPrice	Price	商品価格		No
ProductStock	Numeric(4.0)	商品在庫		No
ProductType	Character(50)	商品種別		No

- ② ツールバーから、**【ファイル】** → **【新規】** → **【オブジェクト】** と選択
- ③ 「新規オブジェクト」ダイアログ（下図）において、以下の内容でトランザクションを作成
- ・カテゴリを選択：**データ管理** → タイプを選択：**Transaction**
 - ・名前：**ProductType**
 - ・デスク립ション：**商品種別**



- ④ 作成ボタンをクリック

⑤ 以下の内容で、項目を定義



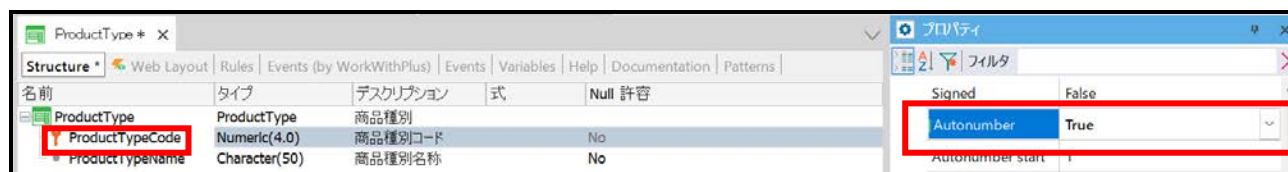
名前	タイプ	DESCRIPTION	式	Null 許容
ProductType	ProductType	商品種別		
ProductTypeCode	Numeric(4.0)	商品種別コード		No
ProductTypeName	Character(50)	商品種別名称		No

	名前	タイプ	DESCRIPTION
1 行目	ProductTypeCode	Numeric(4.0)	商品種別コード
2 行目	ProductTypeName	Character(50)	商品種別名称

⑥ ProductTypeCode をクリック

⑦ 画面右側の「プロパティ」から、「Autonumber」を「True」に変更

画面の右側にプロパティが表示されない場合、F4 キーを押します。



名前	タイプ	DESCRIPTION	式	Null 許容
ProductType	ProductType	商品種別		
ProductTypeCode	Numeric(4.0)	商品種別コード		No
ProductTypeName	Character(50)	商品種別名称		No

プロパティ	値
Signed	False
Autonumber	True
Autonumber start	1

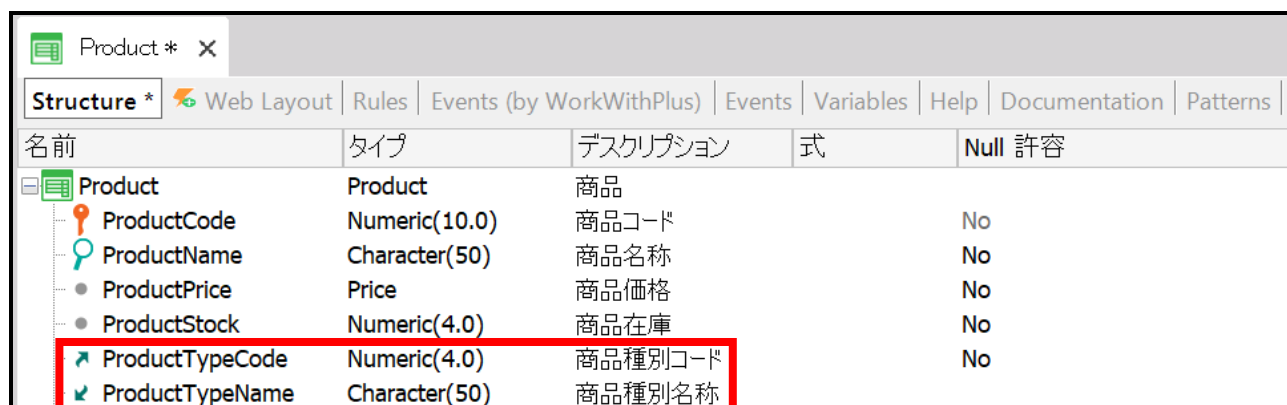
Autonumber とは？

Numeric タイプにおいて、特定のルールに沿って自動採番できる機能です。

⑧ Ctrl キー+S を押し、保存

⑨ 画面左側の「KB エクスプローラー」から、**Product トランザクション**をダブルクリックし、開く

⑩ 以下の内容で、項目を追加で定義



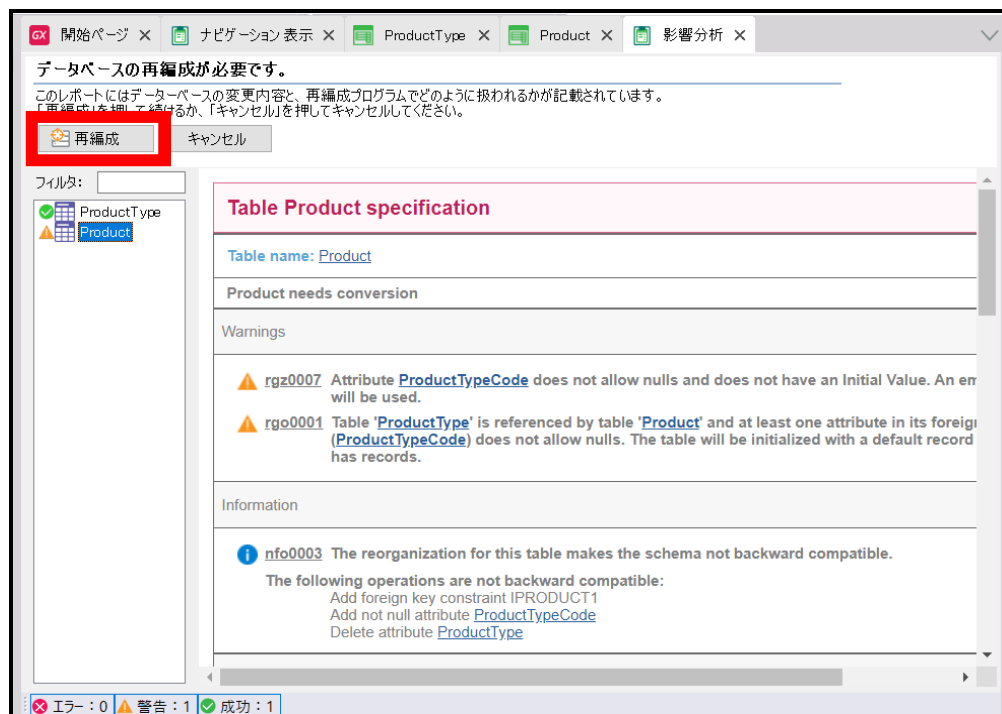
名前	タイプ	デスクリプション	式	Null 許容
Product	Product	商品		
ProductCode	Numeric(10.0)	商品コード		No
ProductName	Character(50)	商品名称		No
ProductPrice	Price	商品価格		No
ProductStock	Numeric(4.0)	商品在庫		No
ProductTypeCode	Numeric(4.0)	商品種別コード		No
ProductTypeName	Character(50)	商品種別名称		No

	名前	タイプ	デスクリプション
5 行目	ProductTypeCode	Numeric(4.0)	商品種別コード
6 行目	ProductTypeName	Character(50)	商品種別名称

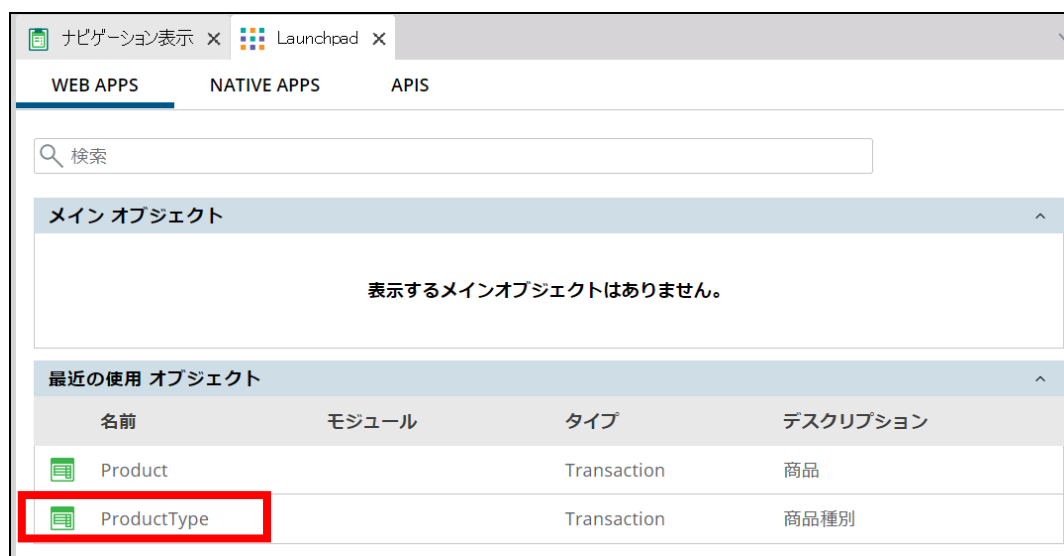
(名前を入力し、Enter キーを押すと、タイプとデスクリプションが自動入力されます)

⑪ F5 キーを押して、実行

⑫ 再編成ボタンをクリック



⑬ Launchpad から、ProductType をクリック



ここからは、商品種別データを2件登録しましょう。

- ⑭ 1つ目：商品種別名称に「化粧品」と入力し、実行ボタンをクリック

商品種別

商品種別コード 0

商品種別名称 化粧品

商品種別コードは何も入力しなくて大丈夫？

先程、**ProductTypeCode**(商品種別コード)の **[Autonumber]** を **[True]** に設定しました。

[Autonumber] を **[True]** に設定しているので、入力する必要はありません。

登録時の内部処理で自動採番された番号で記録されます。

- ⑮ 2つ目：商品種別名称に「医薬品」と入力し、実行ボタンをクリック

商品種別

商品種別コード 0

商品種別名称 医薬品

- ⑯ GeneXus に戻り、Launchpad から、**Product** をクリック

⑰ 以下の内容でデータを入力し、実行ボタンをクリック

・商品コード：101010

・商品名称：胃薬

・商品価格：2000

・商品在庫：120

・商品種別コード：2 （入力欄の右側にある四角形のアイコンをクリックすると、登録済みの商品種別が一覧表示されます。一覧から商品種別名称をクリックすると、選択することができます。）

商品

< < > >| 選択

商品コード	<input type="text" value="101010"/>
商品名称	<input type="text" value="胃薬"/>
商品価格	<input type="text" value="2000"/>
商品在庫	<input type="text" value="120"/>
商品種別コード	<div><input type="text" value="2"/> </div>
商品種別名称	医薬品

これで、表示された商品種別データの中から当てはまるものを選ぶようになりました。

選択リスト 商品種別

商品種別コード

商品種別名称

商品種別コード

商品種別名称

1 化粧品

2 医薬品

終了

GeneXus では、同じ名前の項目属性を複数作ることはできません。

例えば、ProductTypeCode という名前の項目属性はナレッジベース内には 1 つしかありません。

このナレッジベースにおける「**ProductTypeCode**」は、**ProductType トランザクションの主キー**です。

GeneXus では、あるトランザクションの主キーを別のトランザクションで定義した場合、外部参照キーとして役割を付与し、トランザクション間に関連性を生成します。

このトランザクション間の関係性を踏まえ、データベーステーブルは正規化された形で生成されます。GeneXus は「第三正規形」のレベルまでデータを正規化し、外部キーに紐づく参照項目属性は物理テーブル上では保有しません。

例えば、今回のケースの場合、PRODUCT テーブルにおいて「ProductTypeCode」は含まれますが、「ProductTypeName」は含まれません。

この点は後ほどダイアグラムオブジェクトを利用し、確認します。

トランザクションの変更②：画像項目の追加

商品ごとに画像を登録できるようにしましょう。

① GeneXus に戻り、**Product トランザクション**を開く

② 以下の内容で、項目を追加で定義

Product * X				
Structure * Web Layout Rules Events (by WorkWithPlus) Events Variables Help Documentation Patterns				
名前	タイプ	デスクリプション	式	Null 許容
Product	Product	商品		
ProductCode	Numeric(10.0)	商品コード		No
ProductName	Character(50)	商品名称		No
ProductPrice	Price	商品価格		No
ProductStock	Numeric(4.0)	商品在庫		No
ProductTypeCode	Numeric(4.0)	商品種別コード		No
ProductTypeName	Character(50)	商品種別名称		No
ProductPhoto	Image	商品イメージ		No

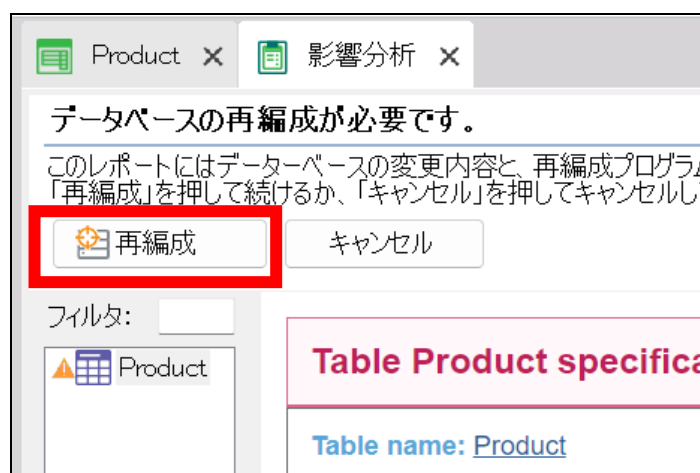
	名前	タイプ	デスクリプション
7 行目	ProductPhoto	Image	商品イメージ

Image とは？

Image は、「画像」という意味です。タイプを Image に設定すると、**画像を登録できる項目**になります。

③ F5 キーを押して、実行

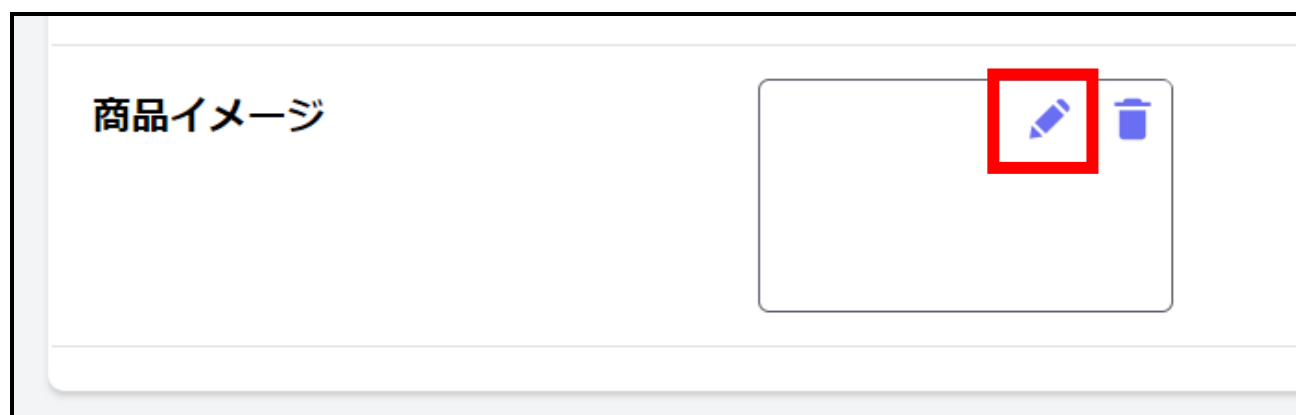
④ 再編成ボタンをクリック



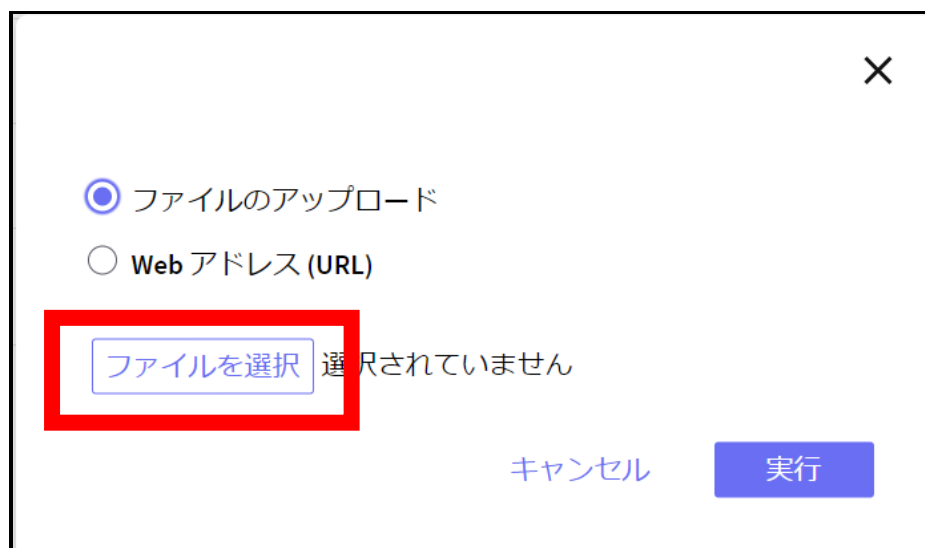
⑤ Launch Pad から、**Product** をクリック

⑥ 表示された画面の「選択」をクリック → 一覧から「胃薬」 をクリック

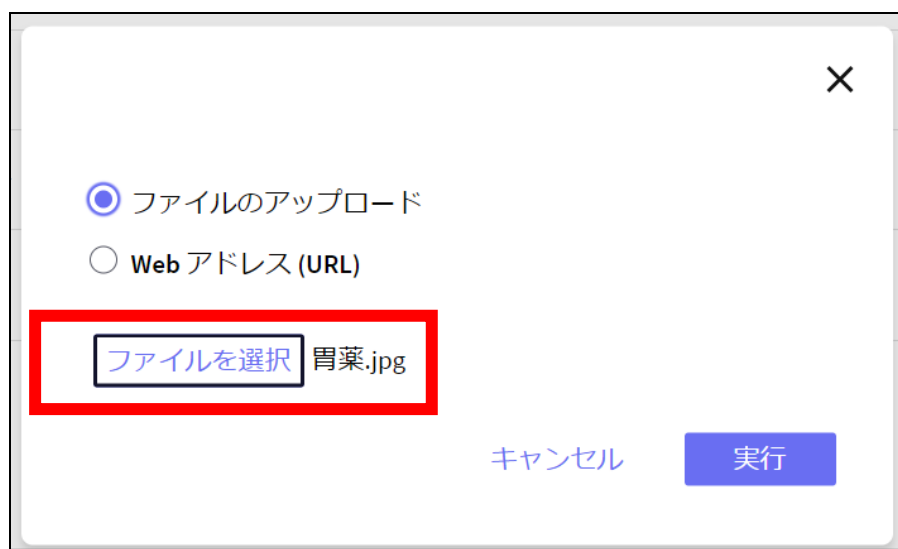
⑦ 商品イメージの鉛筆のアイコンをクリック



- ⑧ 表示される「ファイル選択」ダイアログでファイルを選択 をクリック



- ⑨ OneDay トレーニング素材の「胃薬.jpg」を選択



- ⑩ 「ファイル選択」ダイアログで実行ボタンをクリック

- ⑪ 商品登録画面内の実行ボタンをクリック

これで、既に登録されているデータに画像を追加し、データを更新することができました。

商品

商品コード

101010

商品名称

胃薬

商品価格

2000

商品在庫

120

商品種別コード

2

商品種別名称

医薬品

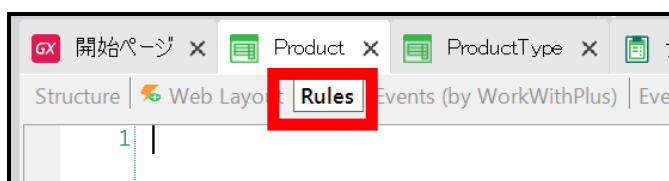
商品イメージ



トランザクションの変更③：ビジネスロジックの実装

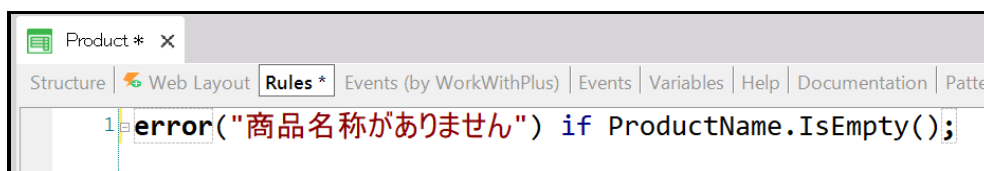
商品名称が入力されていない状態での登録ができないようにするため、**Error** ルールを追加しましょう。

- ① **Product** トランザクションの「**Rules**」を開く



- ② 次の 1 行を入力

error("商品名称がありません") if ProductName.IsEmpty();



注意！

Rules では、最後に必ず「 ; 」（半角のセミコロン）を入力しましょう。

； が入力されていないと、保存ができません。


error("商品名称がありません") if ProductName.IsEmpty(); とは？

「もし、ProductName 欄が空だったら、『商品名称がありません』というエラー文を表示してください」という意味になります。

③ F5 キーを押して、実行

④ Launchpad から、**Product** をクリック

⑤ 商品コード「111111」のみ入力し、**商品名称が未入力のまま**、実行をクリック



The screenshot shows a form with a label '商品名称' (Product Name) on the left. To its right is an empty text input field with a red border. Further right is a red error message icon (a circle with an exclamation mark) followed by the text '商品名称がありません' (Product name is missing).

これで、エラー文が表示され、商品名称が入力されていない状態での登録ができないようになりました。

また、Error ルールのほかに、**Msg** ルールがあります。

Error ルールと Msg ルールの違いは、**登録ができるかできないか**という点です。

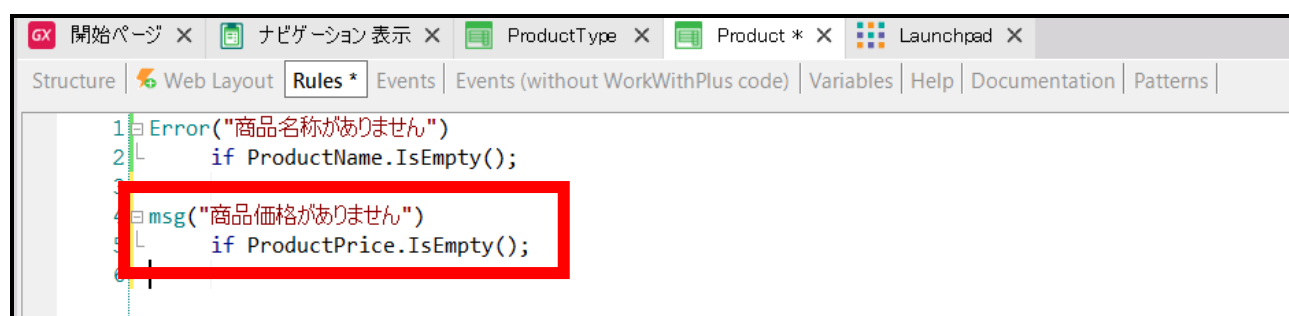
Error ルールは、条件を満たさないと登録ができません。

しかし、Msg ルールは条件を満たしていなくても登録をすることができます。

それでは、Msg ルールを追加し、動きを確認してみましょう。

① **Product** トランザクションの「**Rules**」に、次の 1 行を追記

msg("商品価格がありません") if ProductPrice.IsEmpty();



msg("商品価格がありません") if ProductPrice.IsEmpty(); とは？

「もし、ProductPrice 欄が空だったら、『商品価格がありません』という警告文を表示してください」という意味になります。

② F5 キーを押し、実行

③ Launchpad から、**Product** をクリック

④ 以下の内容でデータを入力し、**商品価格が未入力のまま**、実行をクリック

- ・ 商品コード : **111111**
- ・ 商品名称 : **風邪薬**
- ・ 商品価格 :
- ・ 商品在庫 : **100**
- ・ 商品種別コード : **2 (医薬品)**
- ・ 商品イメージ : **風邪薬.jpg**

商品	
	◀ ◯ ▶ ▷ 選択
商品コード	<input type="text" value="111111"/>
商品名称	<input type="text" value="風邪薬"/>
商品価格	<input type="text" value=""/> ⚠ 商品価格がありません
商品在庫	<input type="text" value="100"/>
商品種別コード	<input type="text" value="2"/> 
商品種別名称	医薬品
商品イメージ	

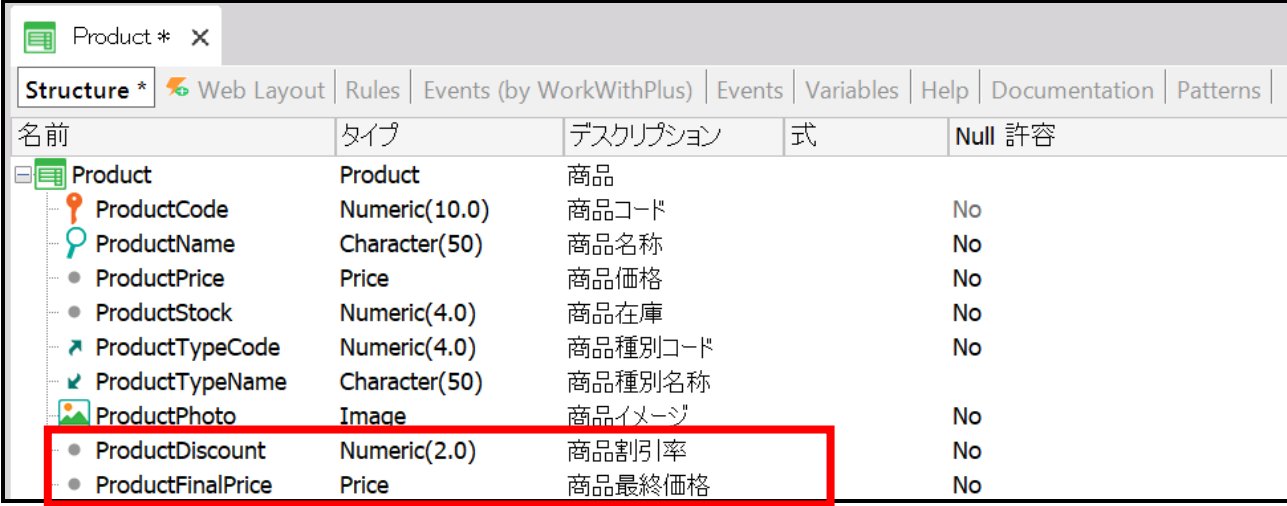
警告文は表示されましたが、商品価格が未入力のままでも、データを登録できることが確認できます。

トランザクションの変更④：計算項目の追加

商品の割引率と、割引後の値段が表示される項目を追加しましょう。

① **Product** トランザクションを開く

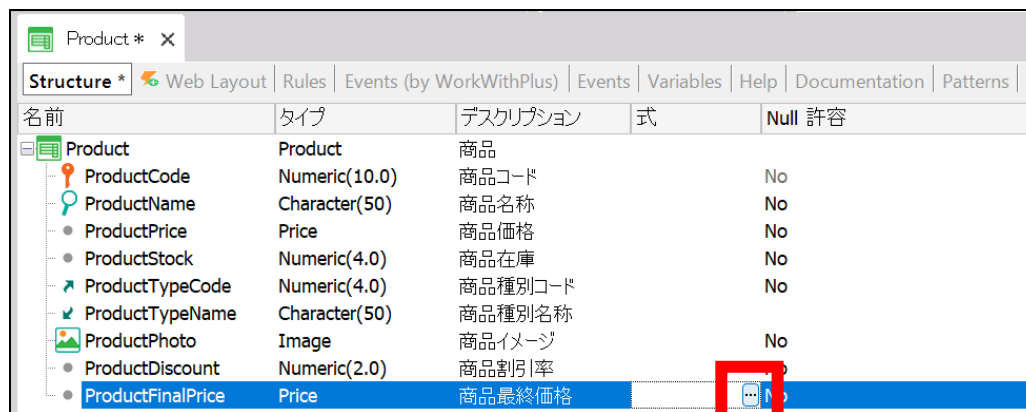
② 以下の内容で、項目を追加で定義



名前	タイプ	デスクリプション	式	Null 許容
Product	Product	商品		
ProductCode	Numeric(10.0)	商品コード		No
ProductName	Character(50)	商品名称		No
ProductPrice	Price	商品価格		No
ProductStock	Numeric(4.0)	商品在庫		No
ProductTypeCode	Numeric(4.0)	商品種別コード		No
ProductTypeName	Character(50)	商品種別名称		No
ProductPhoto	Image	商品イメージ		No
ProductDiscount	Numeric(2.0)	商品割引率		No
ProductFinalPrice	Price	商品最終価格		No

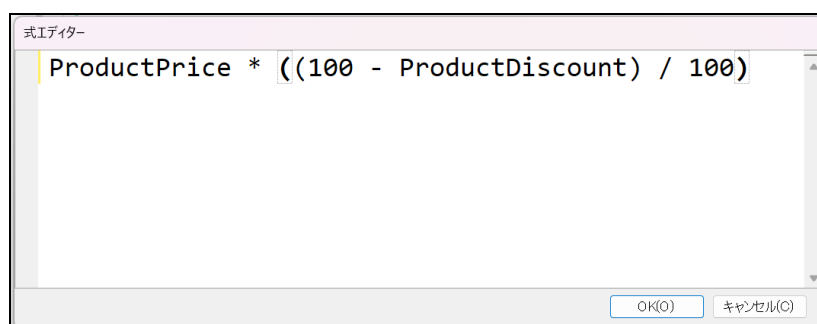
	名前	タイプ	デスクリプション
8 行目	ProductDiscount	Numeric(2.0)	商品割引率
9 行目	ProductFinalPrice	Price	商品最終価格

③ **ProductFinalPrice** の「式」の欄をクリックし、「…」ボタンをクリック



④ 割引後の値段の計算式として、次の 1 行を表示された「式エディター」で入力

ProductPrice * ((100 - ProductDiscount) / 100)



注意！ ここで定義する式では、 ; （セミコロン）は入力しません。

ProductPrice * ((100 - ProductDiscount) / 100) とは？

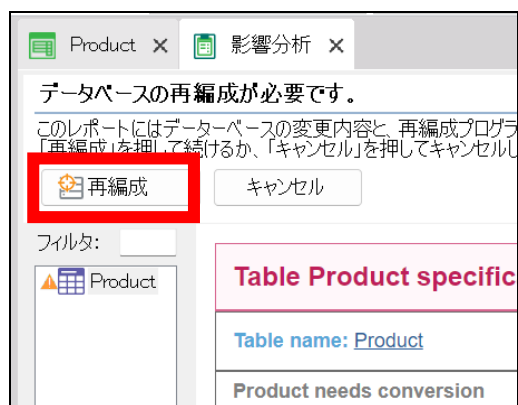
「ProductPrice 欄に入力された数 × {(100 - ProductDiscount 欄に入力された数) ÷ 100}」

という意味になります。

※ここでは、商品割引率の単位は「%」として入力を想定します。

⑤ F5 キーを押して、実行

⑥ 再編成ボタンをクリック



⑦ Launchpad から、**Product** をクリック

⑧ 表示された画面の「選択」をクリック → 一覧から「胃薬」をクリック

⑨ 商品割引率に数値を入力

The screenshot shows a web application window with a '商品イメージ' (Product Image) section at the top, which contains a product image of a box. Below this, there is a form with two rows. The first row is '商品割引率' (Product Discount Rate) with a text input field containing the value '10'. The second row is '商品最終価格' (Product Final Price) with a text input field containing the value '1800'. The first row is highlighted with a red rectangular box.

これで、商品割引率の入力欄と、割引後の値段が自動で計算され、表示される項目を追加できました。

また、式が定義された項目属性は計算することで最新の値を表示できるため、GeneXus では、テーブルへ物理項目を生成しません。

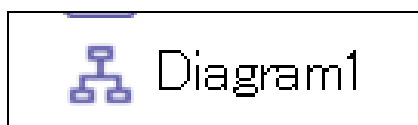
ダイアグラム②：定義構造の変化の確認

先程、トランザクション/テーブルの内容を確認しましたが、ダイアグラムはトランザクション/テーブルの関係性を表示することもできます。

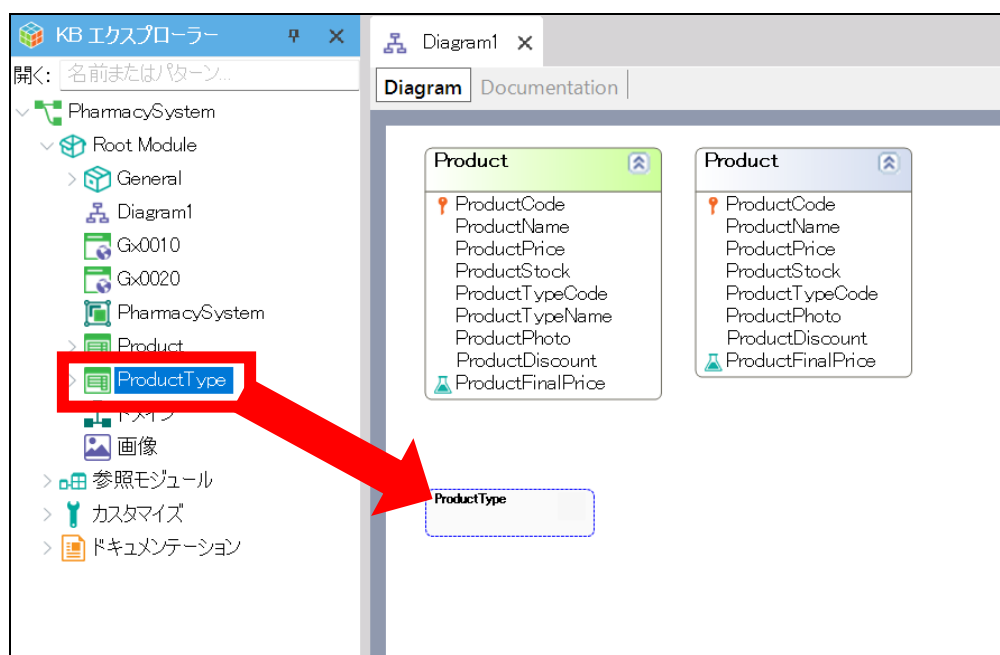
Product トランザクションと ProductType トランザクション/テーブルの関係性を確認してみましょう。

- ① 画面左側の KB エクスプローラーから、Diagram1 をダブルクリックし、開く

(ウィンドウを開いたままの場合、一度閉じて開きなおしてください)

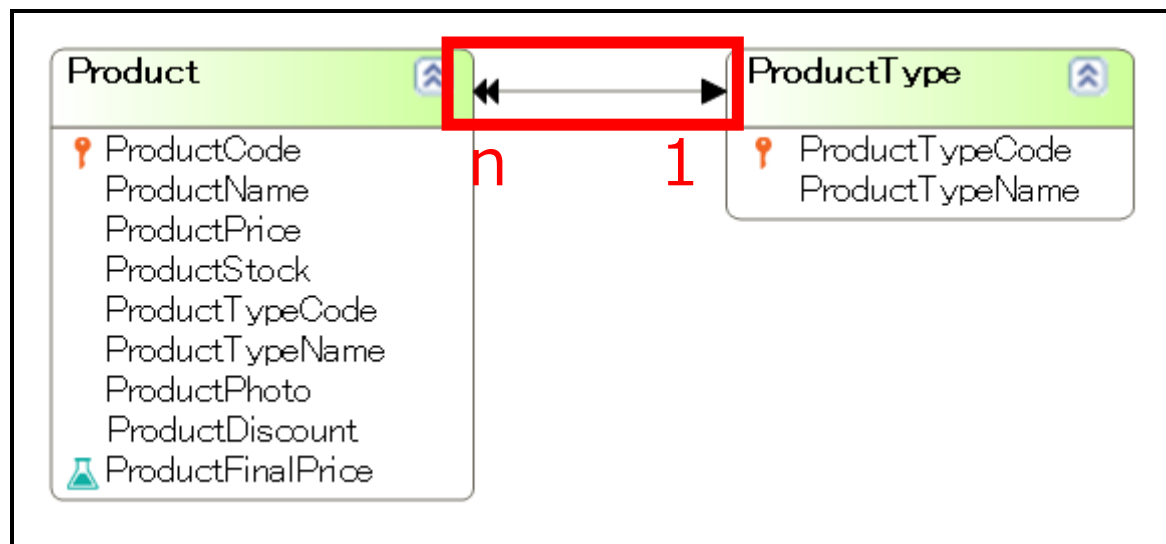


- ② KB エクスプローラーから「ProductType」をドラッグアンドドロップし、Diagram1 に配置



Product と ProductType の間に、矢印が表示されました。

これは、Product が n、ProductType が 1 の関係性であることがわかります。



例えば、「Product : ファンデーション」と言えば「ProductType : 化粧品」と、Product からは **1 個** の ProductType を連想できます。

しかし、「ProductType : 化粧品」と言っても、必ずしも「Product : ファンデーション」とは限りません。

なぜなら、口紅やアイシャドウなどの Product も化粧品だからです。ProductType からは **n 個** の Product が連想できます。

このような関係性を、**1 対 n の関係性**と呼びます。

ダイアグラムでトランザクションの内容や関係性を把握することで、開発に役立てることができます。

先ほど作成時に配置した PRODUCT テーブルの項目と、Product トランザクションの項目を比較してください。

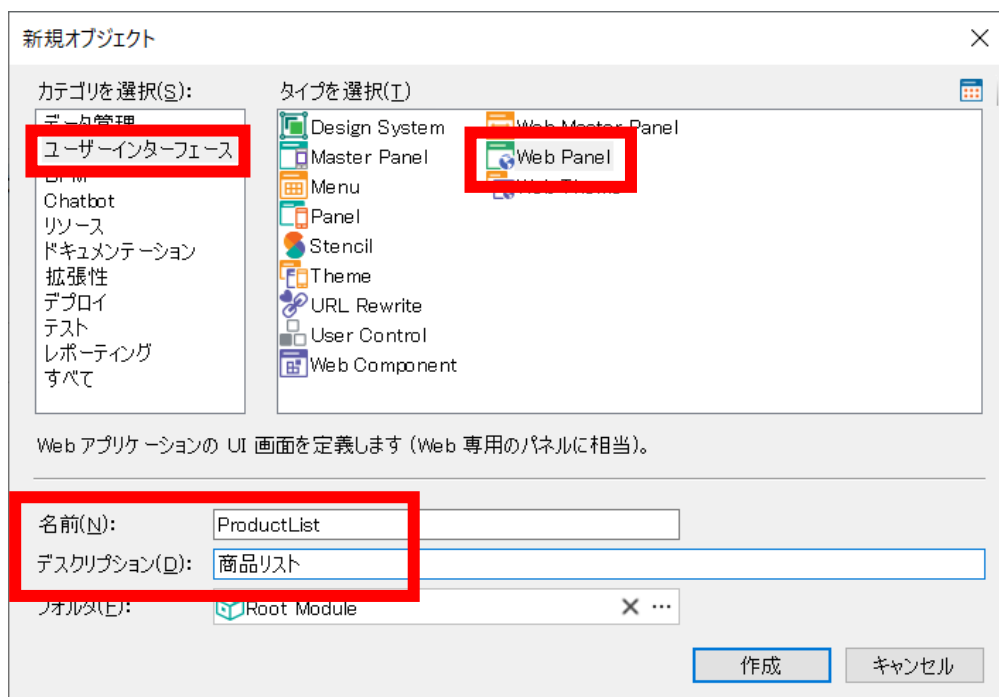
GeneXus が第三正規形で生成するため、テーブルとトランザクションで項目が異なることが確認できます。

Product	Product
ProductCode	ProductCode
ProductName	ProductName
ProductPrice	ProductPrice
ProductStock	ProductStock
ProductTypeCode	ProductTypeCode
ProductPhoto	ProductTypeName
ProductDiscount	ProductPhoto
ProductFinalPrice	ProductDiscount
	ProductFinalPrice

Web パネルの作成：一覧画面の実装

次は、「Web パネル」を作成し、商品一覧画面を実装しましょう。

- ① ツールバーから、**【ファイル】** → **【新規】** → **【オブジェクト】** と選択
- ② 「新規オブジェクト」ダイアログ（下図）において、以下の内容で Web パネルを作成
 - ・カテゴリを選択「**ユーザーインターフェース**」 → タイプを選択「**Web Panel**」
 - ・名前：**ProductList**
 - ・デスクリプション：**商品リスト**

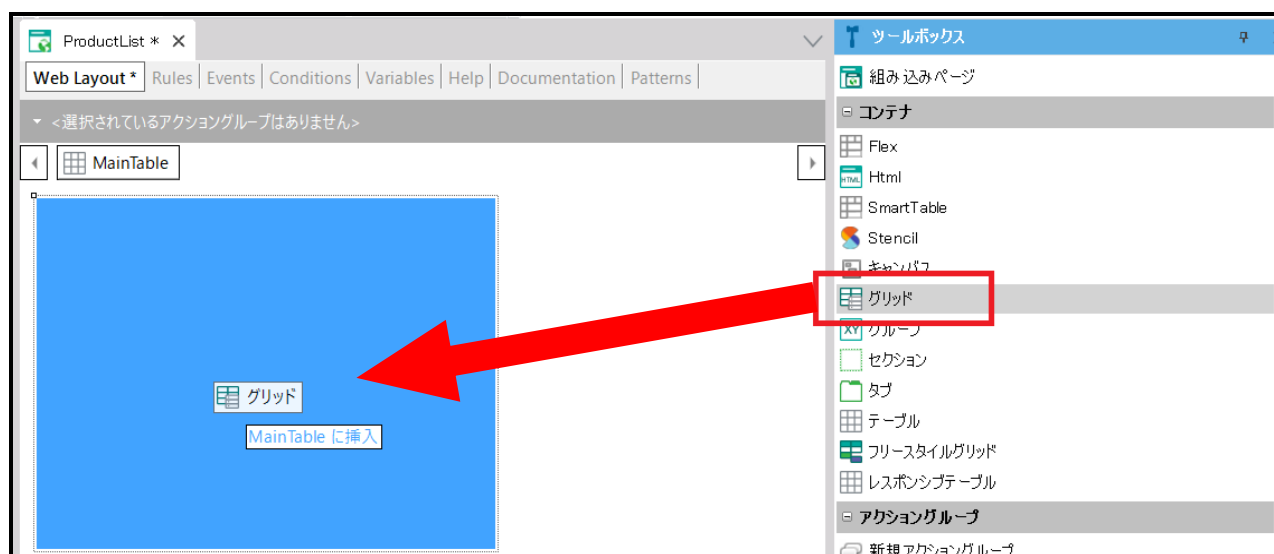


- ③ 作成ボタンをクリック

- ④ 画面右側のウィンドウ下部にあるタブのうち**ツールボックス**をクリック

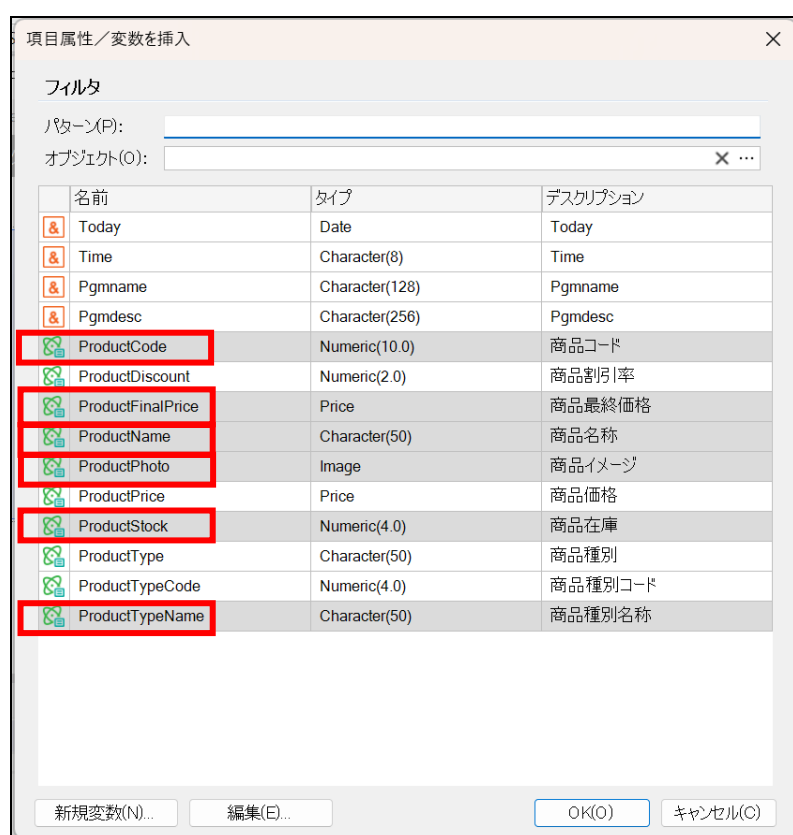


- ⑤ 画面右側のツールボックスから、**グリッド**をドラッグアンドドロップし、枠内に追加



⑥ Ctrl キーを押しながら、表示する項目 6 つを選択

- ・ ProductCode
- ・ ProductName
- ・ ProductFinalPrice
- ・ ProductStock
- ・ ProductTypeName
- ・ ProductPhoto



⑦ OK をクリック

- ⑧ 選択した項目属性がグリッド上に配置されます。



項目の並び順が違う場合は、ドラッグアンドドロップで並び替えができます。

- ⑨ F5 キーを押して、実行

- ⑩ Launchpad から、**ProductList** をクリック

これで、商品の一覧画面が作成できました。

商品コード	商品名称	商品最終価格	商品在庫	商品種別名称	商品イメージ
101010	胃薬	2000	120	医薬品	
111111	風邪薬	0	100	医薬品	

プロシージャの作成①：PDF 出力

次は、「プロシージャ」を作成します。

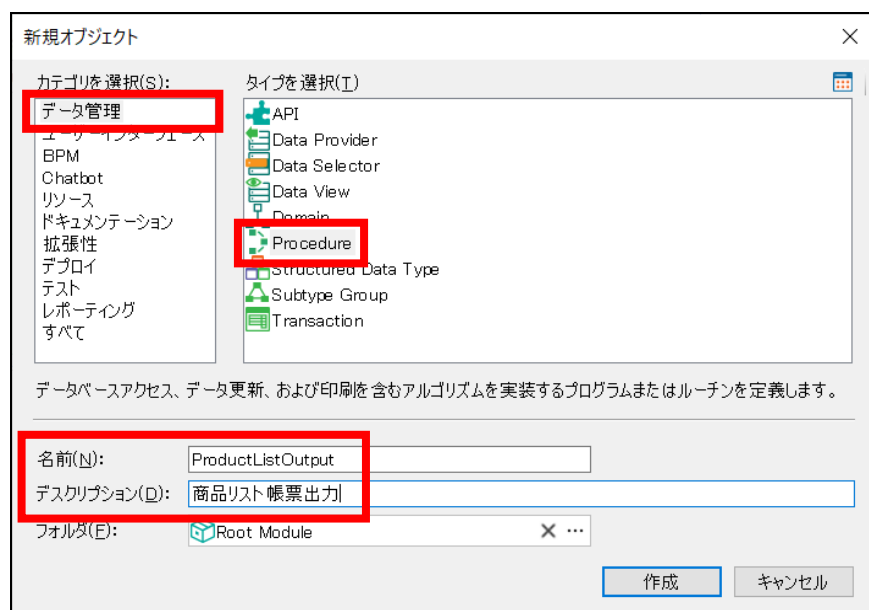
「プロシージャ」では、GeneXus における帳票出力機能として「PDF を出力する」用途と、内部処理を実装する 2 種類の用途があります。

まずは、商品一覧を PDF ファイルで表示させましょう。

① ツールバーから、**[ファイル]** → **[新規]** → **[オブジェクト]** と選択

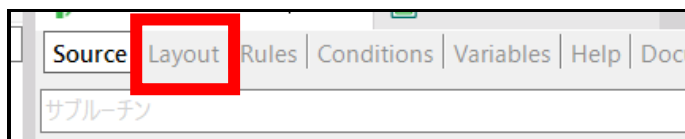
② 「新規オブジェクト」ダイアログ（下図）において、以下の内容でプロシージャを作成

- ・カテゴリを選択「**データ管理**」 → タイプを選択「**Procedure**」
- ・名前：**ProductListOutput**
- ・デスク립ション：**商品リスト帳票出力**



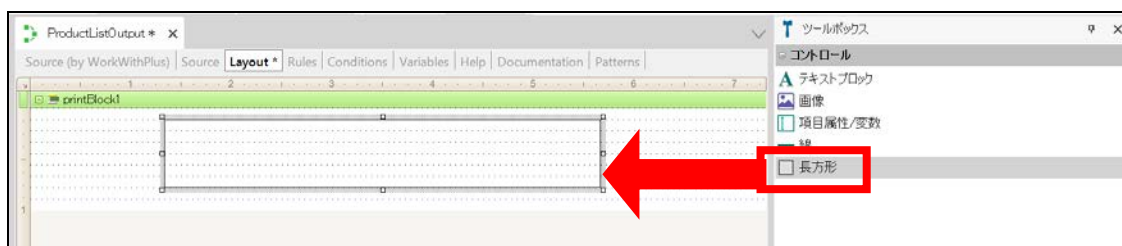
③ 作成ボタンをクリック

④ **Layout** をクリック

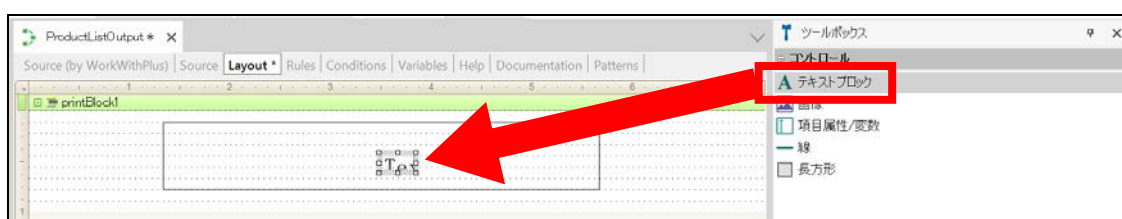


⑤ 画面右側のツールボックスから、**長方形**をドラッグアンドドロップし、**printBlock1** 内に追加

(長方形の角をドラッグすることで、サイズを調整できます)



⑥ 画面右側のツールボックスから、**テキストブロック**をドラッグアンドドロップし、長方形内に追加

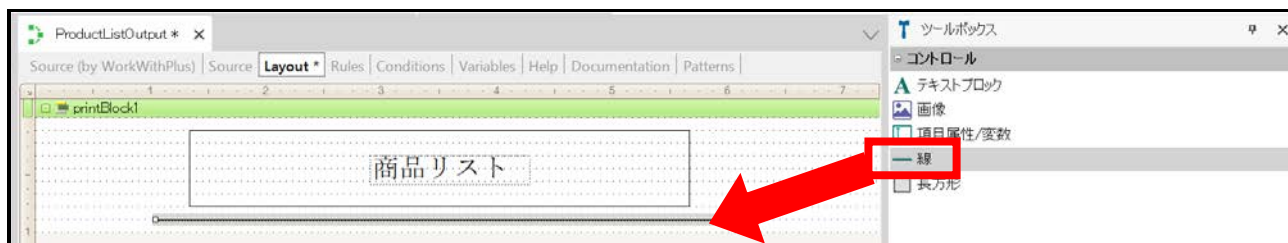


⑦ テキストブロックをダブルクリックし、「商品リスト」と書き換える

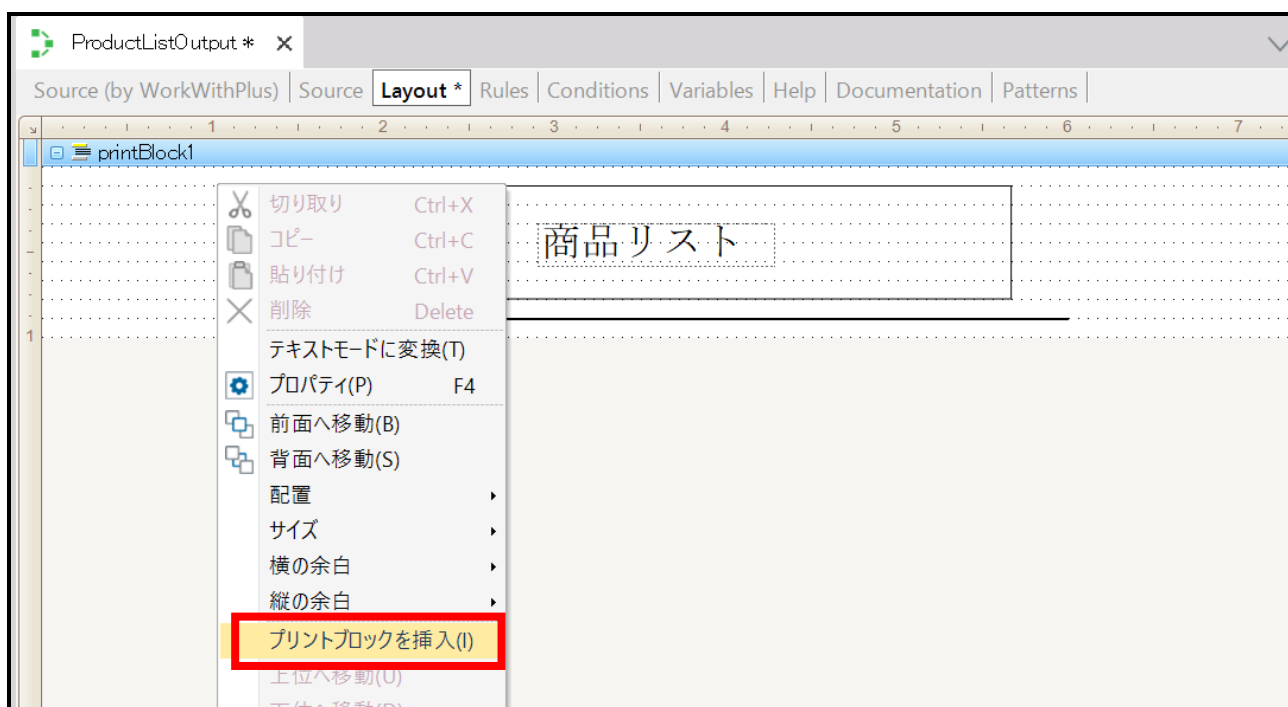


- ⑧ 画面右側のツールボックスから、線をドラッグアンドドロップし、長方形の下に追加

(線の端をドラッグすることで、サイズを調整できます)



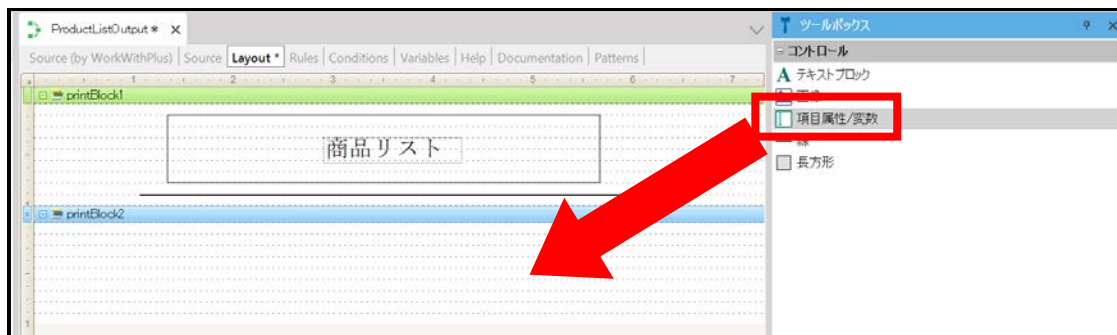
- ⑨ Layout の枠内で右クリック → 「プリントブロックを挿入」をクリック



printBlock2 が、printBlock1 よりも上に配置されてしまった場合

printBlock2 内を右クリック → 「下位へ移動」をクリックすることで、並び順を変更できます。

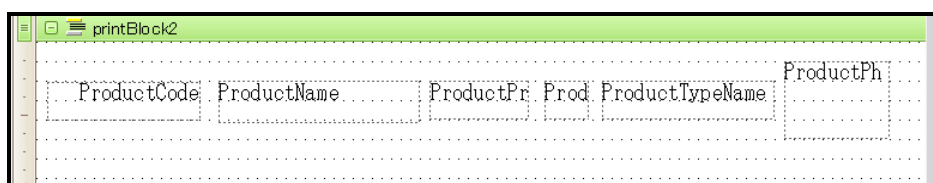
- ⑩ 画面右のツールボックスから、**項目属性/変数**をドラッグアンドドロップし、**printBlock2** に追加



- ⑪ 以下の 6 つの項目属性を **printBlock2** に追加

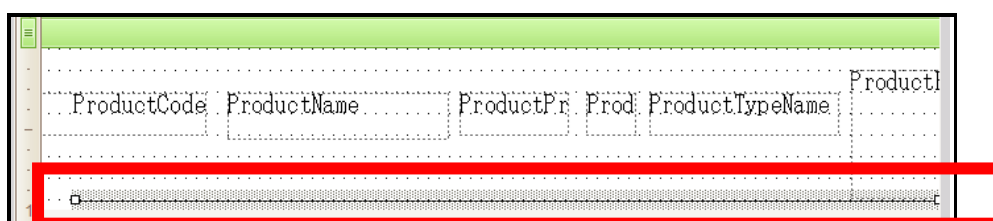
※Web Panel と違い、**プロシージャの Layout** では 1 つずつしか追加できません。

- **ProductCode**
- **ProductName**
- **ProductPrice**
- **ProductStock**
- **ProductTypeName**
- **ProductPhoto**



※配置した項目属性の大きさは必要に応じて調整します。

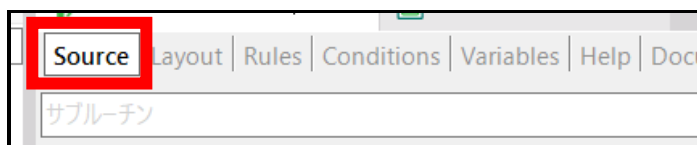
- ⑫ 画面右側のツールボックスから**線**をドラッグアンドドロップし、項目属性の下に追加



これで、Layout の定義は完了しました。

次は、PDF を出力するための処理について定義します。

⑬ **Source** をクリック



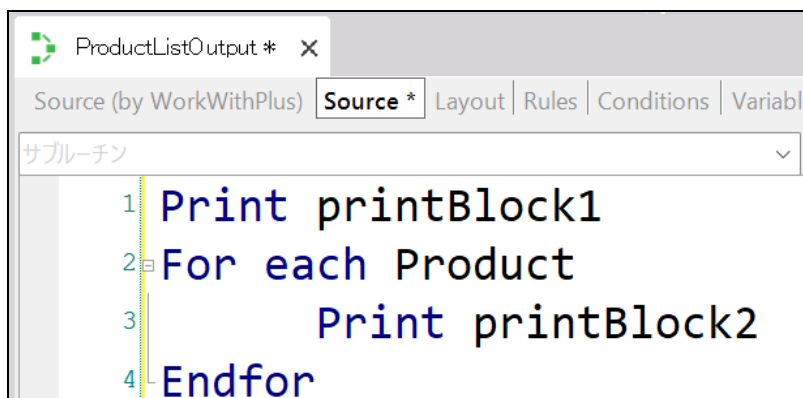
⑭ printBlock1 と printBlock2 の内容を表示するためのコードを入力

Print printBlock1

For each Product

Print printBlock2

Endfor



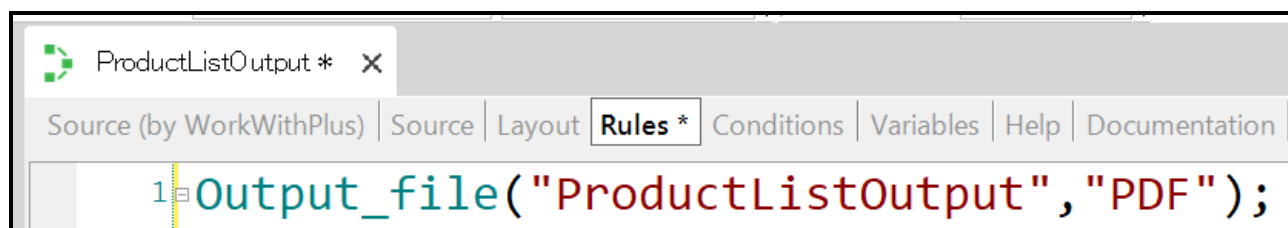
Print printBlock1 ~ Endfor とは？

「printBlock1 の内容を表示してください。Product に登録されているデータを 1 件ずつ繰り返し、printBlock2 の内容を表示してください。」という意味になります。

For each は、「それぞれに対して」という意味で、繰り返し処理を行うときに使います。

⑮ **Rules** をクリック

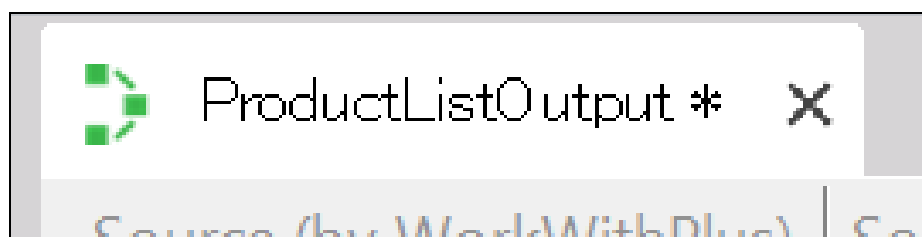
⑯ PDF 出力するためのルール **Output_file(" ProductListOutput ","PDF");** を入力



Output_file("ProductListOutput","PDF"); とは？

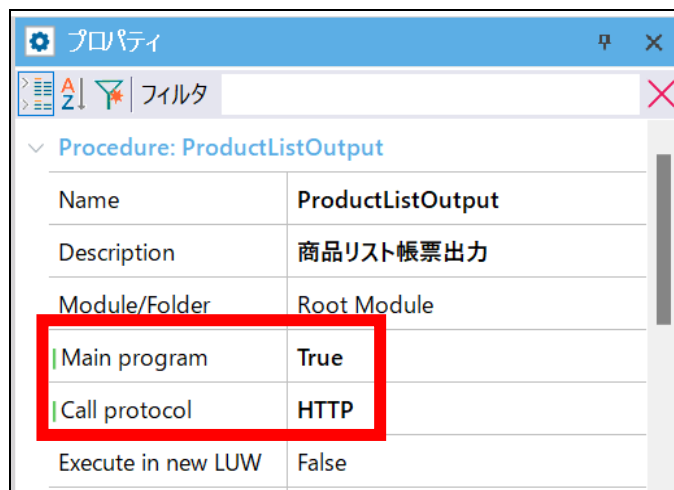
「**ProductListOutput** というタイトルで、**PDF** 形式のファイルを出力してください」という意味になります。

⑰ **ProductListOutput** と書かれているタブ部分をクリック



⑱ F4 キーを押して、画面右側のプロパティから、以下の内容に設定

- Main program : True
- Call protocol : HTTP

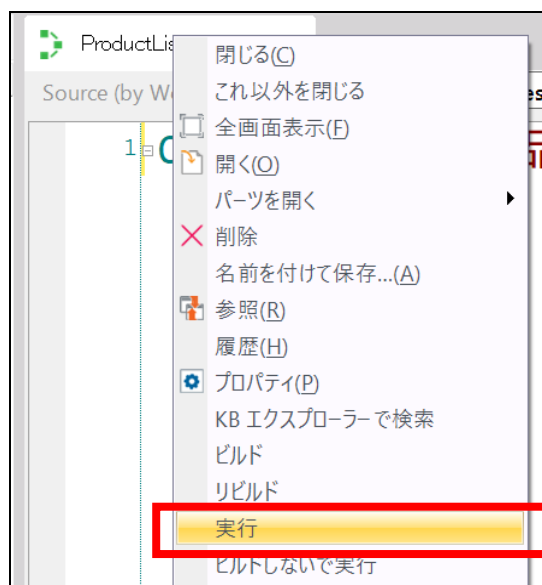


Main program、Call protocol とは？

Main program を **True** にすると、オブジェクトを単体で実行することができます。

Call protocol を **HTTP** にすると、**HTTP (ブラウザ)** で表示することができます。

⑲ **ProductListOutput** の**タブ部分**を右クリック → **実行**をクリック



これで、PDF 出力ができました。

商品リスト					
101010	胃薬	2000	120	医薬品	
111111	風邪薬	0	100	医薬品	

プロシージャの作成②：データの一括処理

プロシージャのもう一つの用途となる「内部処理」を実装します。

商品種別が「医薬品」の商品の値段を 1.2 倍にするように、データの一括処理をしてみましょう。

① ツールバーから、[ファイル] → [新規] → [オブジェクト] と選択

② 「新規オブジェクト」ダイアログ（下図）において、以下の内容で、プロシージャを作成

・カテゴリを選択「**データ管理**」 → タイプを選択「**Procedure**」

・名前：**ProductPriceUp**

・デスクリプション：**商品価格改定**

新規オブジェクト

カテゴリを選択(S):

- データ管理
- ユーザーインターフェイス
- BPM
- Chatbot
- リソース
- ドキュメンテーション
- 拡張性
- デプロイ
- テスト
- レポートング
- すべて

タイプを選択(T):

- API
- Data Provider
- Data Selector
- Data View
- Domain
- Procedure
- Structured Data Type
- Subtype Group
- Transaction

データベースアクセス、データ更新、および印刷を含むアルゴリズムを実装するプログラムまたはルーチンを定義します。

名前(N): ProductPriceUp

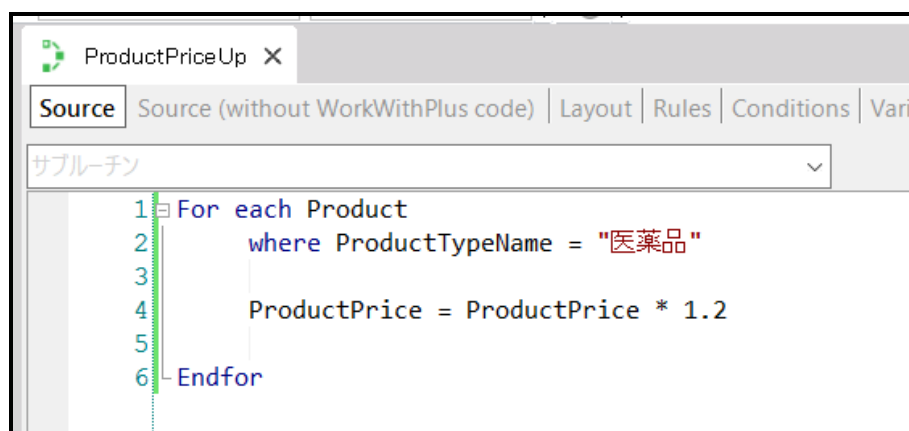
デスクリプション(D): 商品価格改定

フォルダ(F): Root Module

作成 キャンセル

- ③ 作成ボタンをクリック
- ④ **Source** をクリック
- ⑤ 以下の通りに、コードを入力

```
For each Product
    where ProductTypeName = "医薬品"
    ProductPrice = ProductPrice * 1.2
Endfor
```



For each Product ～ Endfor とは？

「ProductTypeName が**医薬品**の Product について、登録されているデータを 1 件ずつ繰り返し参照し、ProductPrice を 1.2 倍にしてください」という意味になります。

プロシージャはこれで完成です。

次は、プロシージャをアプリケーションから実行するため、ボタンを配置した **Web パネル**を作成しましょう。

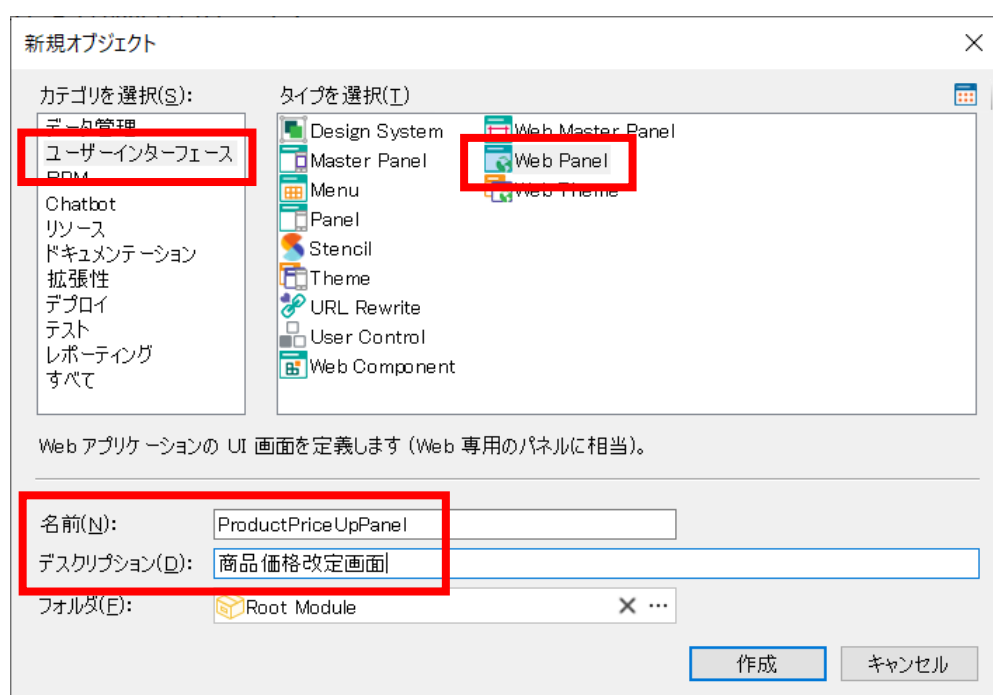
⑥ ツールバーから、**[ファイル]** → **[新規]** → **[オブジェクト]** と選択

⑦ 「新規オブジェクト」ダイアログ（下図）において、以下の内容で Web パネルを作成

・カテゴリを選択「**ユーザーインターフェース**」 → タイプを選択「**Web Panel**」

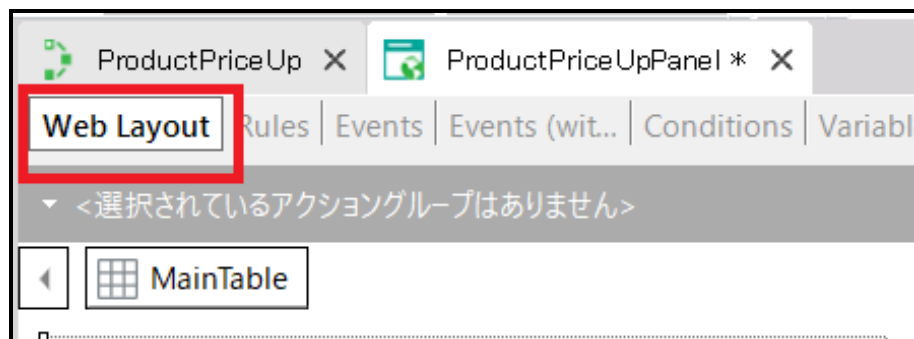
・名前：**ProductPriceUpPanel**

・デスクリプション：**商品価格改定画面**

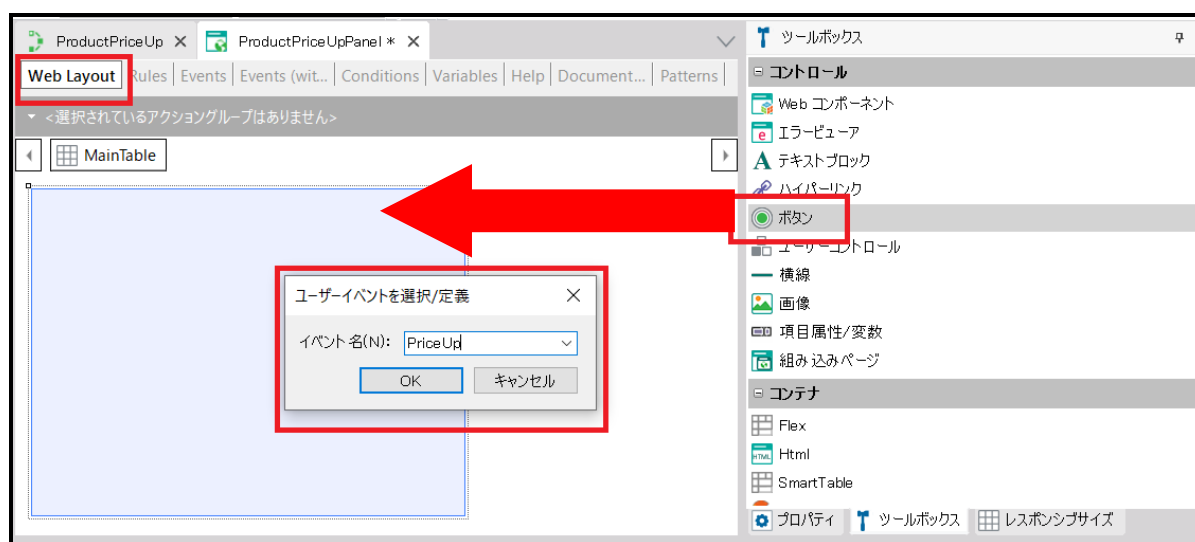


⑧ 作成ボタンをクリック

⑨ Web Layout をクリック

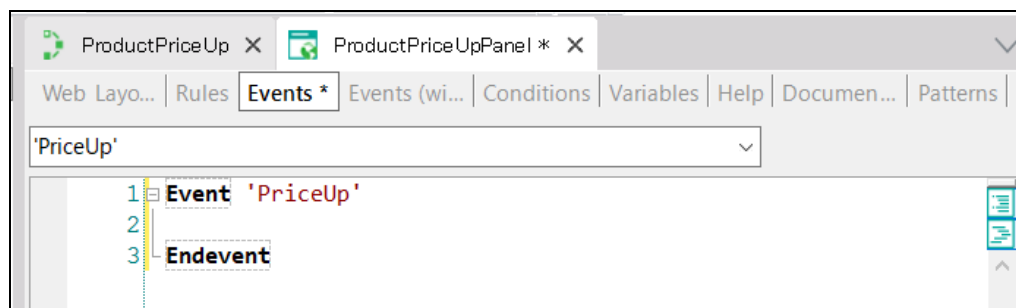


⑩ 画面右側のウィンドウ下部にあるタブのうちツールボックスをクリックし、ボタンをドラッグアンドドロップし、枠内に配置



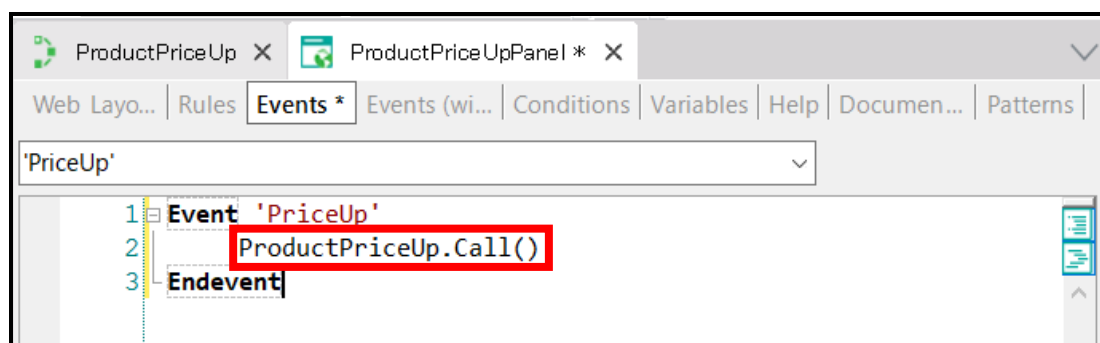
⑪ イベント名に「PriceUp」と入力

- ⑫ PriceUp ボタンをダブルクリックし、Events に移動



- ⑬ 2 行目に以下のコードを追記

ProductPriceUp.Call()



Event 'PriceUp' ~ Endevent とは？

「PriceUp が実行されたら（PriceUp ボタンを押したら）、ProductPriceUp という名前のオブジェクトを実行してください」という意味になります。

- ⑭ F5 キーを押し、実行

- ⑮ Launchpad から、**ProductPriceUpPanel** をクリック

- ⑩ Price Up ボタンを **1 回だけ** クリック（画面表示は変わりません）



- ⑪ GeneXus に戻り、Launchpad から **Product** をクリック

- ⑫ 表示された画面の「選択」をクリック → 一覧から「胃薬」をクリック

胃薬の値段が、**2000 円**から「**2400 円**」に変更されたことが確認できます。

これで、商品種別が「医薬品」の商品の値段が、1.2 倍になりました。

商品	
	◀ < > ▶ 選択
商品コード	101010
商品名称	胃薬
商品価格	2400
商品在庫	120

デザインシステム：画面デザインのカスタマイズ

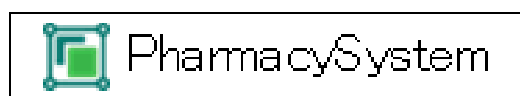
「デザインシステム」は、文字色や背景色、フォント、文字の大きさなど、画面デザインをカスタマイズすることができます。

商品一覧画面（Web パネル）に表示される、登録データの商品名称の文字を赤くしてみましよう。

- ① **ProductList Web パネル**を開き、ProductList のタブ部分をクリック
- ② 画面右のプロパティから、[Style]が **PharmacySystem** になっていることを確認

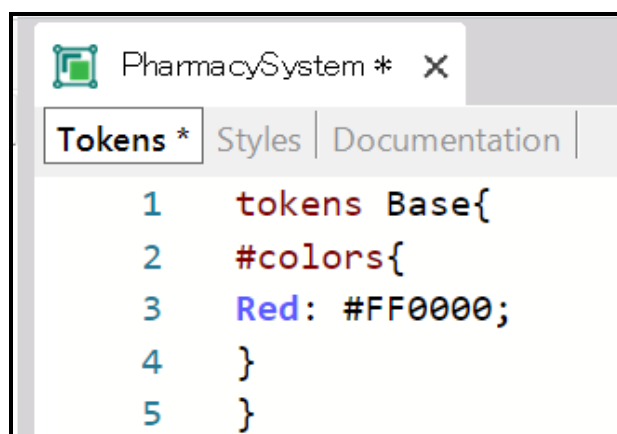


- ③ KB エクスプローラー内の **PharmacySystem** デザインシステムを開く（ナレッジベース作成時に自動的に生成されるオブジェクトです。）



- ④ **PharmacySystem** デザインシステムの **Tokens** に、Red の色を決めるためのコードを入力

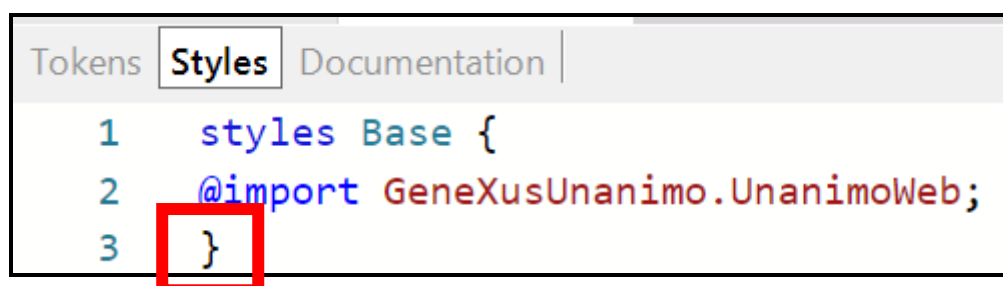
```
tokens Base{  
  
#colors{  
  
Red: #FF0000;  
  
}  
  
}
```



tokens Base{ ~ }とは？

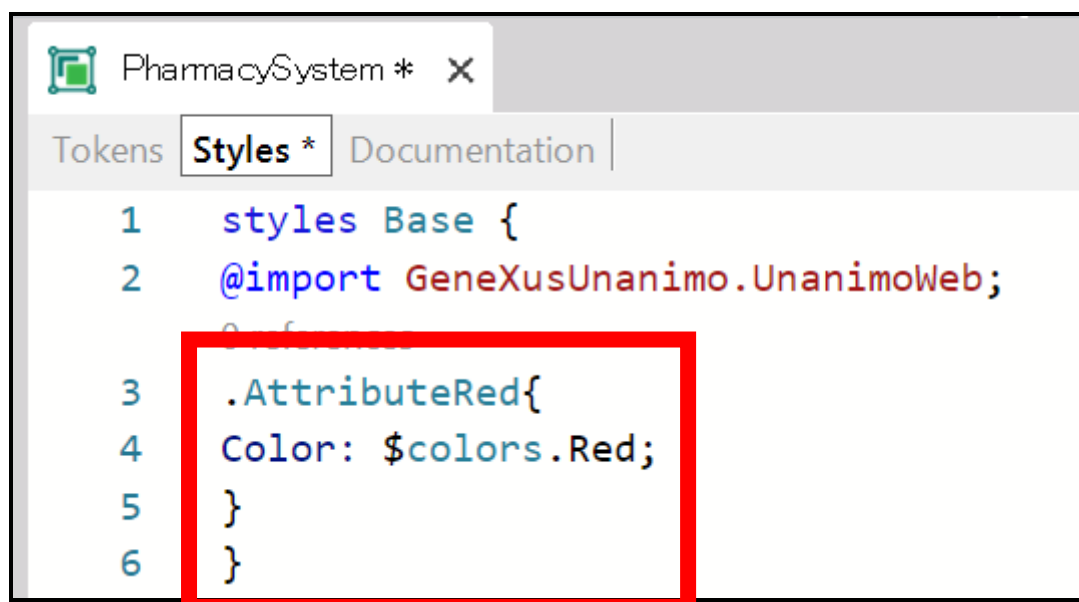
「Red」が、#FF0000 というカラーコードの色であることを定めています。

- ⑤ **Styles** を開き、**3 行目の }** を削除



- ⑥ 3 行目から、Web パネルの文字色を赤くするためのコードを入力

```
.AttributeRed{  
  
Color: $colors.Red;  
  
}  
  
}
```

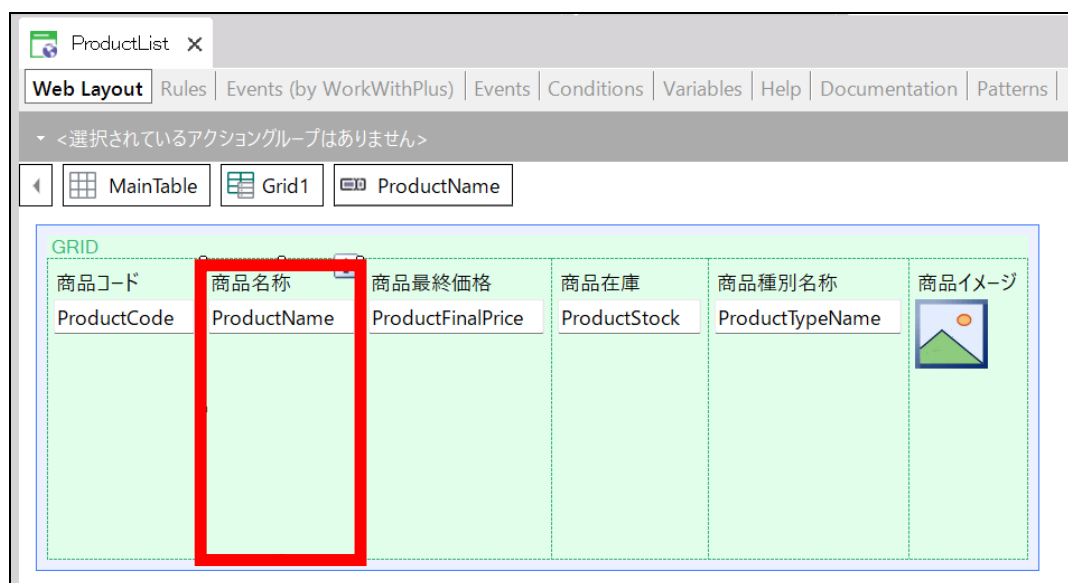


.AttributeRed{ ~ }とは？

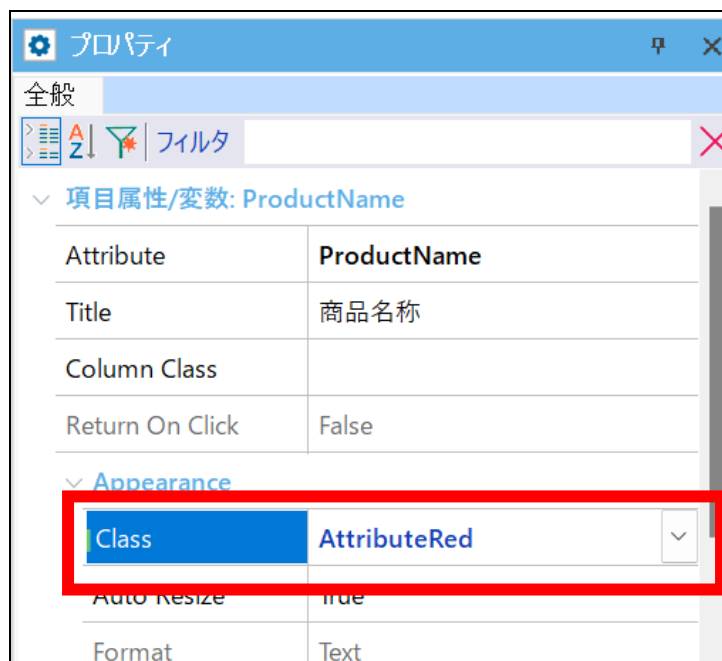
「クラス名が『AttributeRed』となっているコントロールの色を、Red（先程 Tokens タブで定めた #FF0000 という色）にする」という意味です。

- ⑦ Ctrl+S を押し、保存

- ⑧ KB エクスプローラー内の **ProductList** Web パネルを開き、Web Layout の**商品名称**の箇所をクリック



- ⑨ 画面右のプロパティから、[Class]を **AttributeRed** に変更し、Enter キーを押下



⑩ Ctrl+S を押し、保存

⑪ F5 キーを押し、実行

⑫ Launchpad から、**ProductList** をクリック

これで、商品名称の文字を赤くすることができました。

商品コード	商品名称	商品最終価格	商品在庫	商品種別名称	商品イメージ
101010	胃薬	2160	120	医薬品	
111111	風邪薬	0	100	医薬品	

Work With Plus for Web : より効率的なアプリケーションの作成

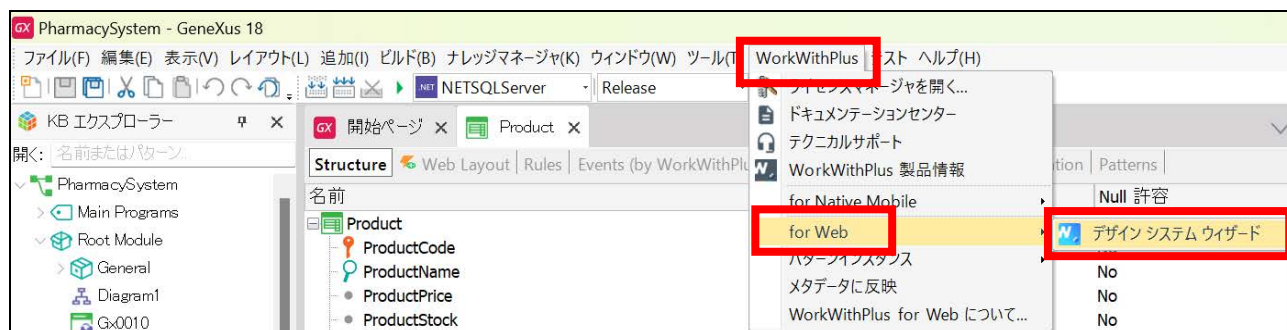
GeneXus には、より効率的にアプリケーションを作成することができる「Work With Plus for Web」というオプション製品があります。

例えば、グラフ、ログイン情報管理ページ、更新履歴の保存機能、PDF・Excel 出力ボタンなどの便利な機能を簡単に追加することができ、よりクオリティの高いアプリケーションを効率良く作成することができます。

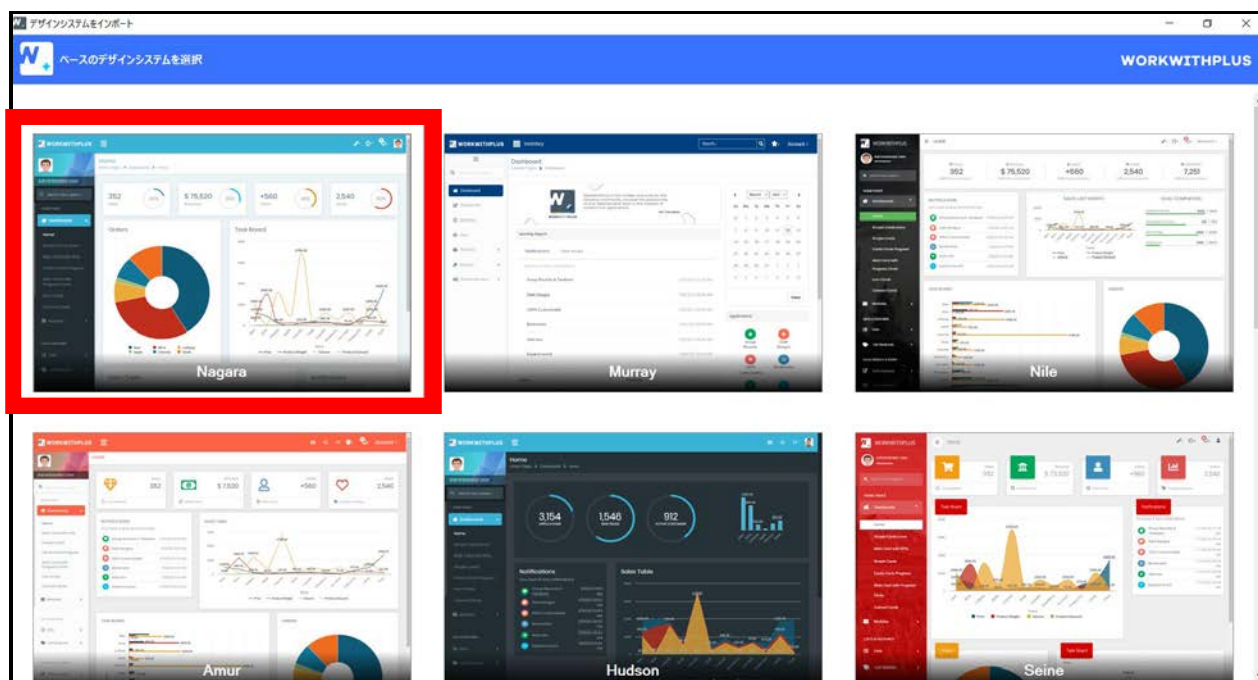
それでは、Work With Plus for Web を体験してみましょう。(※**Work With Plus for Web** をインストール済みの方のみ)

Work With Plus for Web をインストール済みの場合、ツールバーに WorkWithPlus のメニューが表示されます。

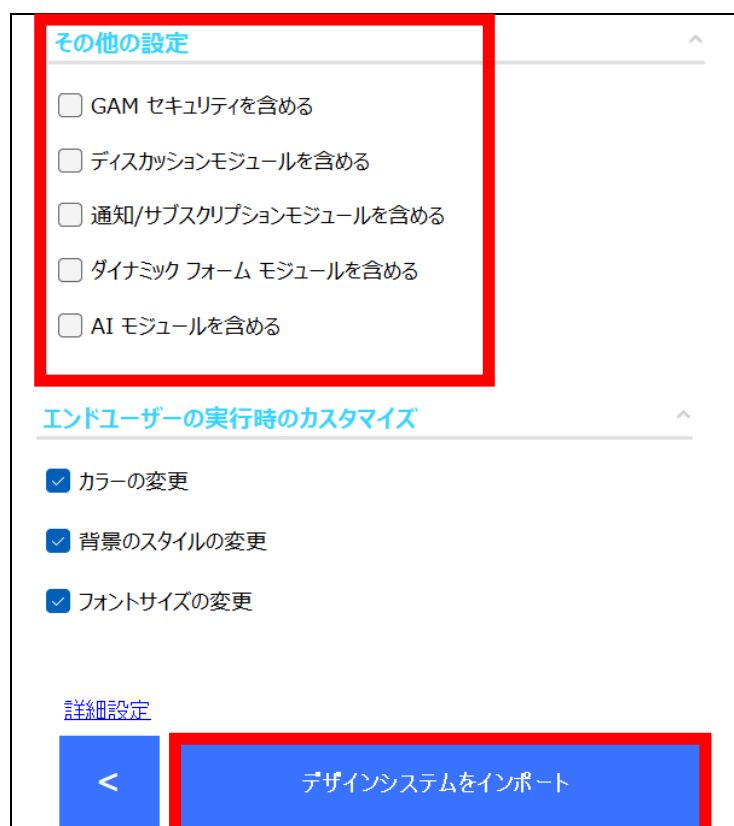
- ① ツールバーより「WorkWithPlus」→「for Web」→「デザインシステムウィザード」を選択



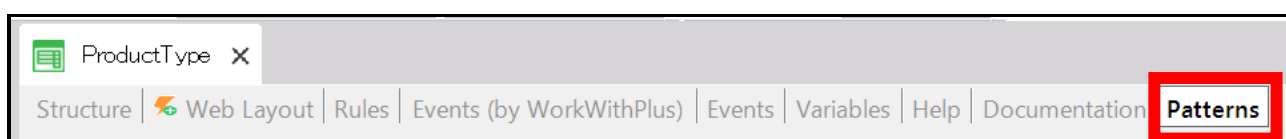
② デザインシステム「Nagara」を選択



③ 「その他の設定」のチェックを全て外し、デザインシステムをインポート をクリック



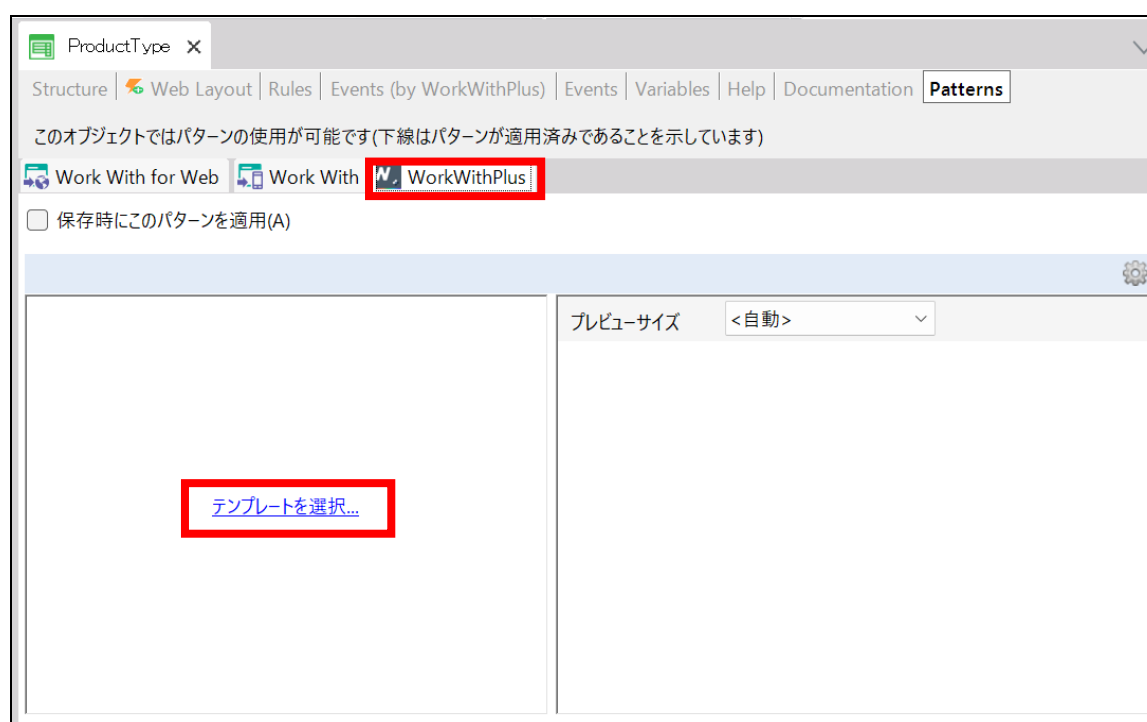
- ④ 表示されるダイアログで OK をクリック
- ⑤ KB エクスプローラーから ProductType トランザクションを開く
- ⑥ [Patterns]をクリック



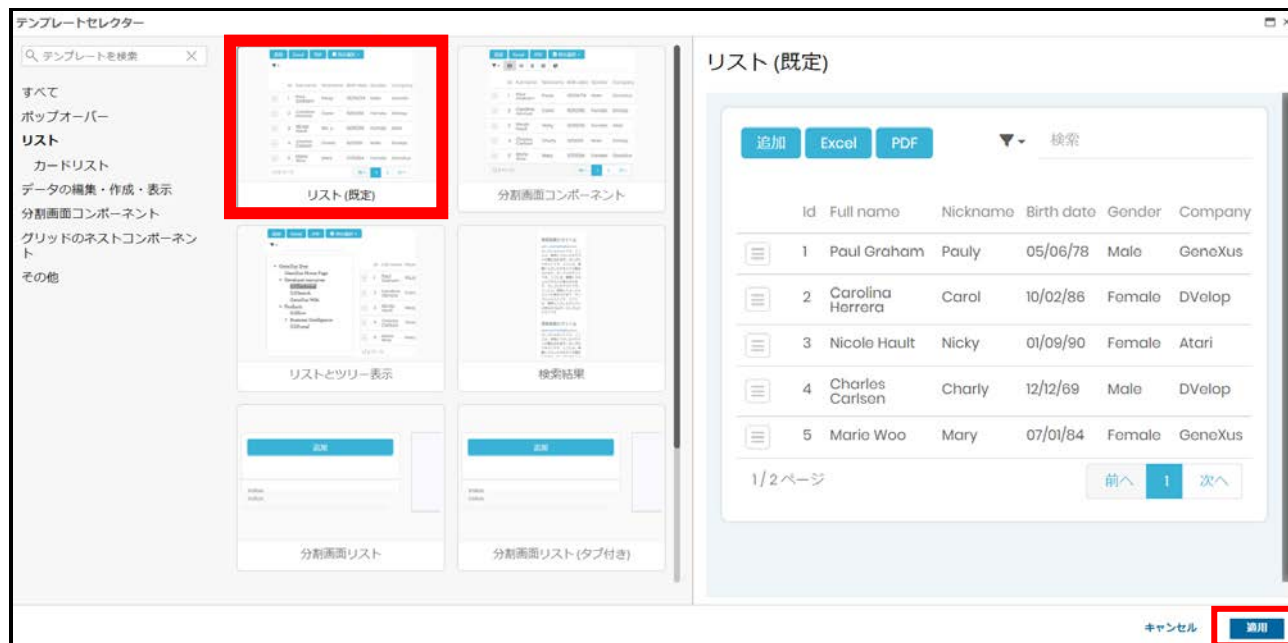
- ⑦ テンプレートセクターが自動で表示されます。

テンプレートセクターが自動で表示されない場合

WorkWithPlus タブをクリックし、**テンプレートの選択**をクリックしてください。

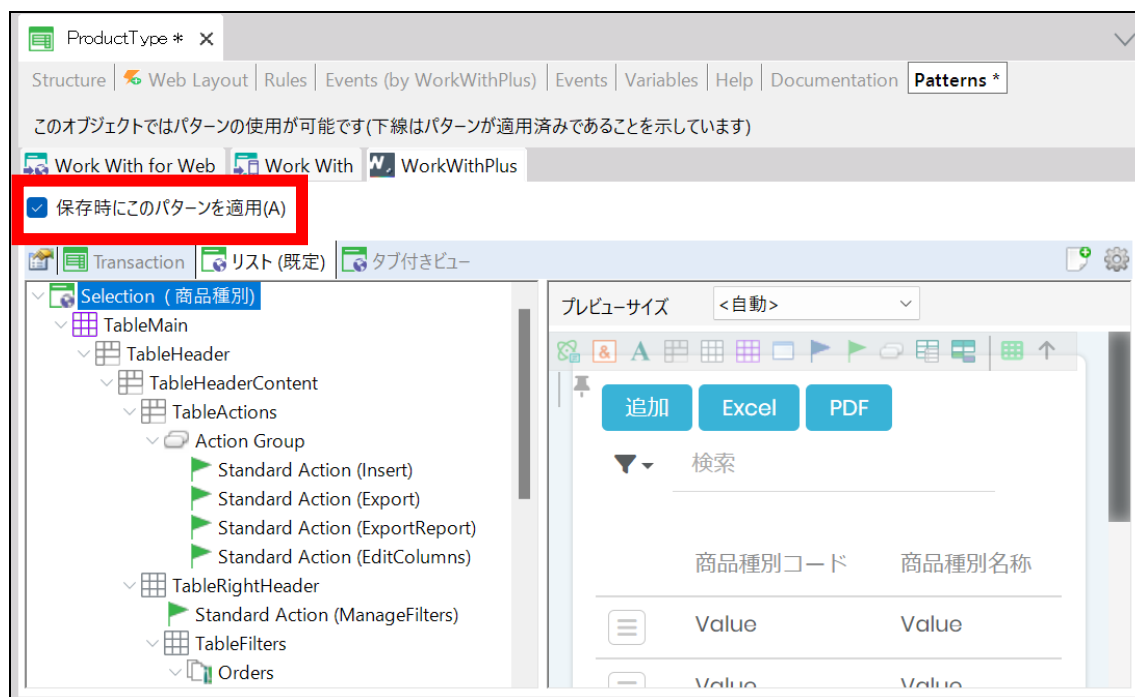


- ⑧ 「リスト（既定）」を選択し、適用ボタンをクリック



- ⑨ 次に表示されるダイアログで適用ボタンをクリック

- ⑩ [保存時にこのパターンを適用] チェックボックスにチェックを入れ、Ctrl+S で保存



⑪ F5 キーを押して、実行

⑫ 左のサイドバーから、開発者メニュー → 商品種別 をクリック

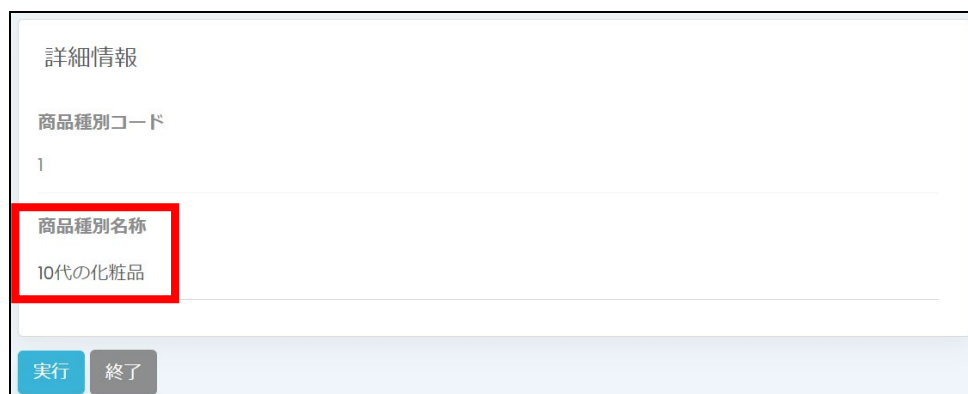


先程とは違う、洗練されたデザインの画面で、商品種別がすべて表示されています。

⑬ 商品種別コード 1 : 化粧品の、三本線のアイコンをクリック → [更新] をクリック



- ⑭ ProductType トランザクションが開くので、商品種別名称「**10代の化粧品**」と書き換え、実行ボタンをクリック



詳細情報

商品種別コード

1

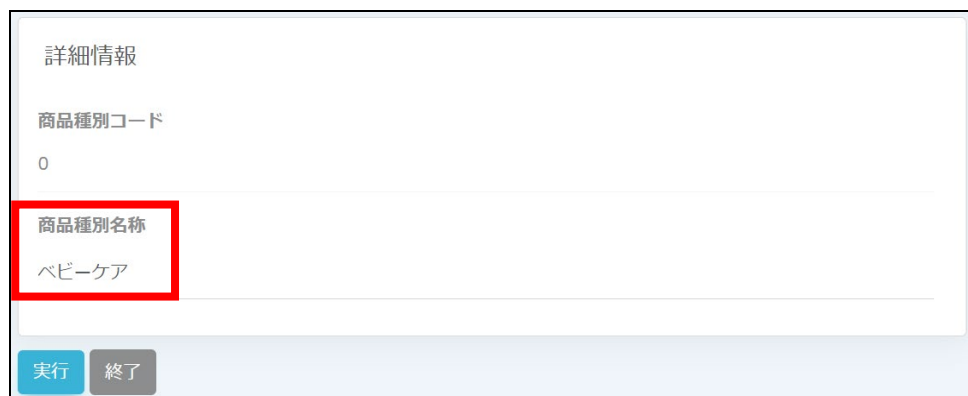
商品種別名称

10代の化粧品

実行 終了

- ⑮ 画面左上の追加ボタンをクリック

- ⑯ 新しい商品種別「ベビーケア」を入力し、実行ボタンをクリック



詳細情報

商品種別コード

0

商品種別名称

ベビーケア

実行 終了

- ⑰ 商品種別名称列「医薬品」をクリックすると、「医薬品」の詳細が確認できます。

このように、さまざまな機能を簡単に追加することができ、効率良く開発をすることができます。

その他の製品・機能

GeneXus では、クオリティの高いアプリケーションを、簡単に、効率良く開発できるよう、さまざまなオプション製品や機能を取り扱っています。

・ GeneXus Server : チーム開発をサポート

GeneXus Server は、ナレッジベースをサーバーにアップロードし、いつでも、どこでも、複数人で、効率良く開発をすることができます。チーム開発には必須の製品です。

・ GXtest : アプリケーションのテスト

GXtest は、テストを自動化できる製品です。手動での動作確認作業が不要になり、効率良く作業を進めることができます。

・ Work With Plus for Native Mobile : スマートフォンアプリケーション用の画面の生成

GeneXus は、Web 用の画面だけでなく、Android や iOS(iPhone)などのスマートフォンアプリ用の画面を簡単に生成することが可能です。Work With Plus for Native Mobile を使用することで、よりクオリティの高い画面テンプレートを使用することができます。

・ GeneXus Access Manager : セキュリティの管理

GeneXus Access Manager は、アプリケーションのセキュリティを強化することができる製品です。例えば、ログイン機能の追加や、特定の人だけがアクセスできるページを設定することができます。

・ Database Reverse Engineering Tool : アプリケーションの統合

Database Reverse Engineering Tool を使用することで、別のツールで作成したデータベースを使用して GeneXus での開発を進めることができます。また、データベースの内容からトランザクションを自動生成することも可能です。

次のステップへ

GeneXus を更に学習したい場合、次のサイトから深く掘り下げて学ぶことができます。

- GeneXus トレーニングコース : <https://www.genexus.jp/training>
- GeneXus Wiki : <http://wiki.genexus.jp/hwikibypageid.aspx?46071>
- 開発に役立つサイト :

<https://www.genexus.jp/community-and-support-jp/support-overview#link05>