

演習

GeneXus Advanced コース

GeneXus™ 18

2026 10 月 初版

GeneXus. All rights reserved. 本書は、GeneXus™の明示的許可なしには如何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。

GeneXus™
www.genexus.com

MONTEVIDEO - URUGUAY
CIUDAD DE MÉXICO - MÉXICO
MIAMI - USA
SÃO PAULO - BRASIL
TOKYO - JAPAN

Av. Italia 6201- Edif. Los Pinos, P1
Hegel N° 221, Piso 2, Polanco V Secc.
7300 N Kendall Drive, Suite 470
Rua Samuel Morse 120 Conj. 141
2-27-3, Nishi-Gotanda
Shinagawa-ku, Tokyo, 141-0031

(598) 2601 2082
(52) 55 5255 4733
(1) 201 603 2022
(55) 11 4858 0300
(81) 3 6303 9381
(81) 3 6303 9980

内容

課題	5
新しいプロジェクトのためのナレッジベース	5
データモデル:トランザクションと初期設定	6
実行画面からのデータ入力	8
ルールの定義(対象教材動画番号:2~3).....	9
クライアントサイドバリデーションの挙動確認(推奨).....	11
「Schedule」トランザクションの挙動確認.....	11
その他のルール.....	13
オプション(任意実施課題)	17
トランザクションにおけるルールのトリガーイベント(対象教材動画:4~6).....	18
トリガーイベントを利用したルール.....	18
第 2 レベル以降の行単位のルールとトリガーイベント	20
オプション	22
ルールとグローバル式をトリガーするための評価ツリー	23
新規要望の実装①.....	27
新規要望の実装②.....	28
オプション:更新モードでの行単位の制御	29
オプション:評価ツリーとトリガーイベントに関する演習	30
トリガーイベントの活用	31
複合外部キーの部分的な NULL 許容(対象教材動画番号:7).....	33
チケットにおける参照整合性チェックの回避	33
データが存在するテーブルに外部キー項目属性が追加される場合の影響(推奨)	33

並行トランザクション(推奨)	34
サブタイプ(対象教材動画番号:8~9)	35
サブタイプ:複数参照化	35
サブタイプ:特殊化	37
サブタイプのその他の使用例	37
式(対象教材動画番号:10~13)	38
「式」対「ルール」	38
Horizontal 式(オプション)	38
Aggregate 式	39
部屋による価格変動	39
プレミアム映画の価格割り増し	39
チケット販売数の管理	40
ダイナミックトランザクション(対象教材動画番号:14)	41
トランザクションにおけるイベント(対象教材動画番号:15)	42
FOR EACH コマンドの詳細(対象教材動画番号:17~18)	43
サブルーチン(対象教材動画番号:19)	45
UNIQUE 節(対象教材動画番号:20)	46
データセクター(対象教材動画番号:21)	48
ビジネスコンポーネントの利用(対象教材動画番号:23~26)	51
担当従業員の更新	51
映画の出演者に対する操作	51
プロシージャオブジェクトの利用(対象教材動画番号:27~19)	52

担当従業員の更新	52
映画の削除	52
ユーザー向け画面:WEB パネルオブジェクト(対象教材動画番号:33~40)	53
スケジュール一覧表示画面	53
複数グリッド	54
並行グリッドの利用	54
ネストされたグリッドの利用	55
ベーステーブルなしのグリッドの利用	55
顧客向けに重点を置いたアプリケーション(対象教材動画番号:42~44)	56
WEB サービス(対象教材動画番号:46~47)	59

課題

映画館を運営する企業から、業務で扱う情報を保存、管理および表示するシステムの開発を依頼されました。システムは以下のモジュールで構成されているものとします:

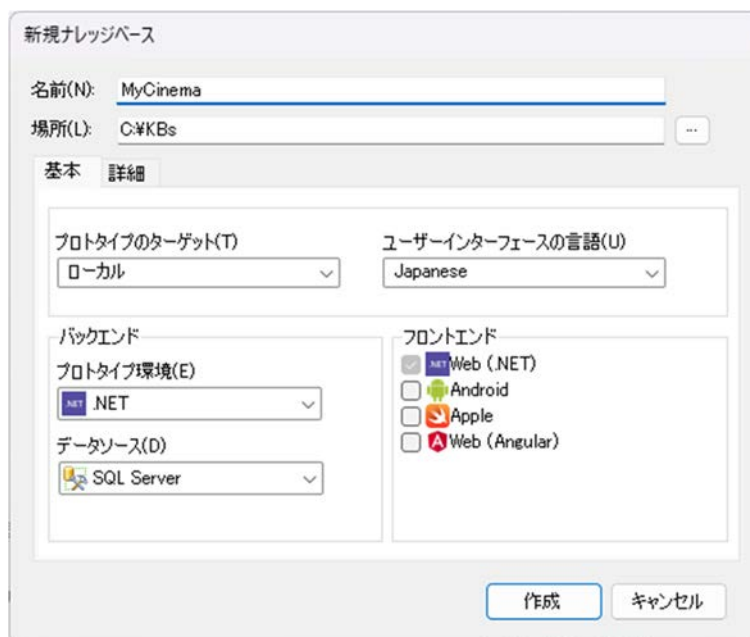
- **バックオフィス:** 企業の従業員がインターネット接続のある任意の場所から情報を扱えるように、Web サーバー上で実行されるアプリケーションの一部
- **フロントエンド:** 顧客に表示される情報が見栄えの良いものになるように、Web サーバー上で実行されるアプリケーションの一部

※利用データは、AI によって生成された架空のものです。

新しいプロジェクトのためのナレッジベース

GeneXus を起動し、「MyCinema」という名前のナレッジベースを作成して、アプリケーションの開発を開始します。

本課題では、プロトタイプ環境は、「.NET」、データソースは「SQLServer」を利用することを想定しています。



注:本資料には、以下 2 種類の演習が含まれています。

1. GeneXus IDE 上で操作を行う演習
2. アプリケーション上での操作は省略可能で、受講者が回答を記入する演習

データモデル: トランザクションと初期設定

この映画館システムは、市内の各劇場で上映される映画のスケジュールを管理します。利用者はスケジュールに基づき、特定の劇場で上映される映画のチケットを購入することができます。

企業は、映画に関する情報(監督、主要キャスト、ジャンル、製作国など)を管理する必要があります。

また、映画館の顧客に関する情報も管理し、顧客はサービスの利用頻度に応じて特典を得ることができます。

基本的なトランザクションを用いて KB を初期化し、この要件のモデリングを開始するには、同梱の MyCinema.xpz ファイルを KB にインポートする必要があります。

「Trn」フォルダ内には、以下のようなトランザクションが含まれています:

- Actor
- Category
- Country
- Customer
- FilmDirector
- Movie
- Room
- Schedule
- Ticket

各トランザクションの Structure エlement を分析し、モデル化されたエンティティ間の関係を推測してください。

そのために、ダイアグラムオブジェクトを利用し、トランザクションのダイアグラム図を作成することをお勧めします。

テーブルのダイアグラム図も作成することができ、メニューバーの「表示」→「テーブル」を選択することで、KB エクスプローラー内に「Tables」ノードが追加され、テーブルをダイアグラムオブジェクトへドラッグできます。

テーブルのダイアグラム図については、あらかじめ「Trn」フォルダ内に「Diagram1」という名前のダイアグラムオブジェクトを用意してあるため、確認にあたって新規作成は必須ではありません。

次のページからの記載内容にのっとり、各トランザクションの役割を確認します。

「Movie」トランザクション

映画に関する情報(名称、公開年、ポスター画像、説明、制作国、カテゴリー、監督、出演者リスト)を記録します。

出演者リストには、主演か助演かを指定する項目もあります。

出演者の出身国は、映画の製作国と異なる場合がある点に注意してください。

「Room」トランザクション

映画が上映される部屋に関する情報(名称、タイプ、座席数、価格)を記録します。

部屋ごとに異なる価格が設定できる要件を満たしています。

「Schedule」トランザクション

実際の上映に関する情報(上映される部屋、上映日、上映時刻、上映される映画、残席数)を記録します。

上映に関する情報は、上映される部屋、上映日、上映時刻による複合主キーであることに注目します。

上映時刻は、列挙型ドメインにより、あらかじめ定義された 4 つの時刻から選択します。

この実装では、同じ映画が同じ日の同じ時間に異なる部屋で上映されるようにデータの管理が可能です。

「Ticket」トランザクション

購入したチケットに関する情報(上映スケジュール、購入した座席数、合計金額)を記録します。

もし、購入者が登録済みの顧客である場合、顧客とチケットの紐づけが行われます。

「Customer」トランザクション

映画館の顧客に関する情報(名前、メールアドレス、VIP フラグ、ポイント)を記録します。

ポイントは、チケットとの紐づけにより加算され、ポイントを利用することで、賞品と交換することができます。

交換した商品も記録します。

なお、名前(または名称)を持つほとんどのテーブルで、候補キーを確立するために、一意のインデックス(Unique Index)が作成されています。

実行画面からのデータ入力

メニューバーの「ビルド」→「開発者メニューの実行」を選択(または、F5 キー押下)します。「ビルドプロセスに必要なプロパティを設定」ダイアログで、必要情報を自身の環境に合わせて入力し、「完了」ボタンをクリックし、表示される「影響分析」ウィンドウで「作成」ボタンをクリックします。

完了までの間、Id ドメインの AutoNumber プロパティが True であるため、注意すべき以下の点を確認してください:

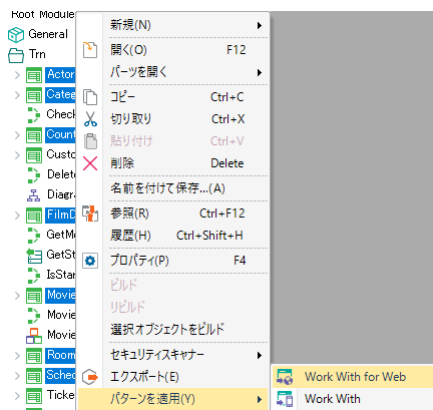
1. 「影響分析」ウィンドウに表示される「CustomerReward」テーブルと、「Customer」テーブルの警告が何を意図しているか確認します。
AutoNumber プロパティによって自動採番が可能なケースは、その項目属性自体がテーブルの主キーである場合のみでした。
複合主キーの一部である場合や、キー項目でない場合、自動採番が行われることはありません。そのため、このようにプロパティの設定が無視されることを警告として表示します。
2. 「Room」、「FilmDirector」、「CustomerReward」、「Customer」、「Actor」、「MovieActor」、「Movie」、「Country」、「Category」については、テーブルを新規作成しますが、データプロバイダーによって初期データの登録が行われる点に注目します。
KB エクスプローラー内で、各テーブルの基となるトランザクションオブジェクトのノードを展開すると、該当のデータプロバイダーオブジェクトがあることを確認します。
この機能については、本コースで改めて取り上げるため、存在の確認のみを行います。

実行したアプリケーションを操作し、初期登録されたデータを利用し、各トランザクションの役割を確認してください。

「Schedule」トランザクションと、「Ticket」トランザクションについては、初期データはありません。

その後、「Ticket」トランザクションと「Customer」トランザクションを除くすべてのトランザクションオブジェクトに Work With for Web パターンを適用してください。

パターンを適用する場合、KB エクスプローラー内で複数選択し、右クリックメニューから一括で適用することも可能です。



ルールの定義(対象教材動画番号:2~3)

「Room」トランザクションの Rules エlementを確認します。
パターンによって自動生成されているもの以外に注目します。

```
11 日 Msg("部屋名の入力が必要です。")
12  L   if RoomName.IsEmpty();
13
14 日 Error("部屋座席数は、0より大きい必要があります。")
15  L   if Insert and RoomSeats = 0;
16
17 日 Msg("部屋価格の入力が必要です。")
18  L   if RoomPrice.IsEmpty();
```

この実装は、以下の要件を適切に満たしているでしょうか？

- 「部屋の名前が未入力」、「部屋の価格が未入力」、「部屋のタイプが未指定」の 3 つのケースで、ユーザーに警告を表示する必要があります。
(部屋のタイプ(RoomType)項目属性のデータタイプに注意が必要)
- 部屋の座席数が、0 のデータは保存できないようにする必要があります。

部屋の一覧(WWRoom)を実行し、以下の 3 点を確認し、必要に応じて実装します。

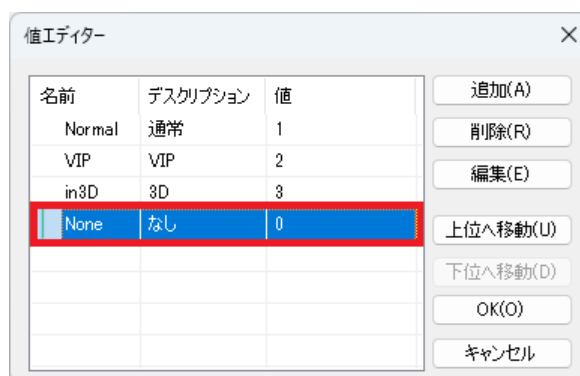
1. 任意のデータに表示される「更新」リンクをクリックし、「部屋名」に入力された値を空の状態に変更し、フォーカスを移動するとメッセージが表示されることを確認します。
このメッセージは、モードの指定が条件にないため、どのモードでも実行されます。
では、この状態(部屋名が空)でデータを保存(更新)することができるでしょうか？
2. 任意のデータに表示される「更新」リンクをクリックし、「部屋タイプ(RoomType)」のコンボボックスで、「(なし)」を選択します。
ルールに何も定義されていませんが、「フィールド 部屋タイプ は範囲外です」というエラーメッセージが表示され、保存(更新)することができません。
この挙動は要件を満たしているでしょうか？
今回、RoomType 項目属性では、「Control Info」プロパティ群に含まれる Empty Item プロパティを True に設定しているため、コンボボックスに「(なし)」という選択肢が表示されます。
しかし、RoomType ドメインの Enum values プロパティで、「0」(空の値)が定義されていないため、許可されていません。

RoomType 項目属性	RoomType ドメイン																																												
Type Definition <table> <tr><td>Supertype</td><td></td></tr> <tr><td>Based on</td><td>RoomType:Domain</td></tr> <tr><td>Data Type</td><td>Numeric</td></tr> <tr><td>Length</td><td>1</td></tr> <tr><td>Decimals</td><td>0</td></tr> <tr><td>Signed</td><td>False</td></tr> <tr><td>Initial value</td><td></td></tr> </table> Validation Control Info <table> <tr><td>Control Type</td><td>Combo Box</td></tr> <tr><td>Values</td><td>通常:1, VIP:2, 3D:3</td></tr> <tr><td>Empty Item</td><td>True</td></tr> <tr><td>Empty Item Text</td><td>GX_EmptyItemText</td></tr> </table>	Supertype		Based on	RoomType:Domain	Data Type	Numeric	Length	1	Decimals	0	Signed	False	Initial value		Control Type	Combo Box	Values	通常:1, VIP:2, 3D:3	Empty Item	True	Empty Item Text	GX_EmptyItemText	Type Definition <table> <tr><td>Based on</td><td>(none)</td></tr> <tr><td>Data Type</td><td>Numeric</td></tr> <tr><td>Length</td><td>1</td></tr> <tr><td>Decimals</td><td>0</td></tr> <tr><td>Signed</td><td>False</td></tr> <tr><td>Enum Values</td><td>Normal, 通常, 1; VIP, VIP, 2; in3D, 3D, 3</td></tr> <tr><td>Collection</td><td>False</td></tr> <tr><td>Initial value</td><td></td></tr> </table> Validation Control Info <table> <tr><td>Control Type</td><td>Combo Box</td></tr> <tr><td>Values</td><td>通常:1, VIP:2, 3D:3</td></tr> <tr><td>Empty Item</td><td>False</td></tr> </table>	Based on	(none)	Data Type	Numeric	Length	1	Decimals	0	Signed	False	Enum Values	Normal, 通常, 1; VIP, VIP, 2; in3D, 3D, 3	Collection	False	Initial value		Control Type	Combo Box	Values	通常:1, VIP:2, 3D:3	Empty Item	False
Supertype																																													
Based on	RoomType:Domain																																												
Data Type	Numeric																																												
Length	1																																												
Decimals	0																																												
Signed	False																																												
Initial value																																													
Control Type	Combo Box																																												
Values	通常:1, VIP:2, 3D:3																																												
Empty Item	True																																												
Empty Item Text	GX_EmptyItemText																																												
Based on	(none)																																												
Data Type	Numeric																																												
Length	1																																												
Decimals	0																																												
Signed	False																																												
Enum Values	Normal, 通常, 1; VIP, VIP, 2; in3D, 3D, 3																																												
Collection	False																																												
Initial value																																													
Control Type	Combo Box																																												
Values	通常:1, VIP:2, 3D:3																																												
Empty Item	False																																												

項目属性で Empty Item プロパティを True に設定した理由ですが、RoomType 項目属性に基づく変数を Web パネル等で作成したとき、「(なし)」が自動で選択できるようにするためでした。

しかし、トランザクションにおける項目属性としては、「(なし)」「(空の値)」は有効な値ではなく、適切な設定ではありませんでした。

空の値を入力できるようにするため、下図のように RoomType ドメインの Enum Values プロパティに「空の値(0)」を追加します。



この結果、RoomType 項目属性の Empty Item プロパティは、False に戻すことができ、「なし(0)」を選択した場合、「IsEmpty」メソッドは、True を返します。

再度アプリケーションを実行すると、「なし」を選択してもエラーメッセージは表示されず、未指定のまま登録することができます。

ただし、これでは要件を満たせていないため、Msg ルールを追加し、未指定であることの警告が表示されるように変更します。

3. 「追加」ボタンをクリックし、「部屋残座席」を「0」のままで登録できるでしょうか？
結果の確認後、部屋の一覧画面に戻り、任意のデータに表示される「更新」リンクをクリックし、「部屋残座席」の値を「0」に変更し、保存（更新）できるでしょうか？
もし、部屋の残座席が「0」のデータを記録できる場合、要件を満たせていないため、ルールの条件を確認し、必要に応じて修正してください。

注: 新規登録または更新で、既存の部屋名と同じ名前の部屋(重複)は登録できません。
これは、RoomName 項目属性を対象に、一意インデックスが作成されているためです。

クライアントサイドバリデーションの挙動確認(推奨)

課題を引き続き進めるまえに、エラールールがトリガーされたとき、すべてのトランザクションで次のフィールドにフォーカスを進められないように変更を行ってください。

また、ルールによって表示されるメッセージは、すべてフィールドの下に表示されるように設定してください。

これらの変更を行った後、「Room」トランザクションを利用し、設定内容を確認してください。

「Schedule」トランザクションの挙動確認

続いて、「Schedule」トランザクションに注目します。

このトランザクションの主キーは、「上映する部屋」、「日付」、「上映時間」で構成されます。
上映時間は、MovieTime という列挙型ドメインに基づき、4つの時刻から選択します。

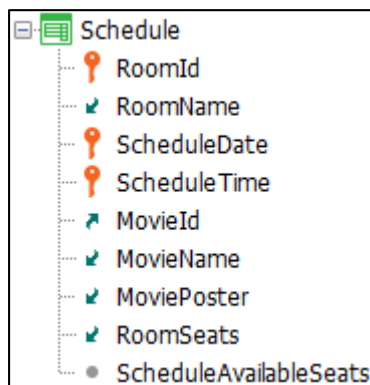
名前	デスクリプション	値
Early Morning	午前11時	11
Afternoon	午後2時	14
Evening	午後5時	17
Night	午後8時	20

追加(A)
削除(R)
編集(E)
上位へ移動(U)
下位へ移動(D)
OK(O)
キャンセル

MoovieTime ドメインの Enum Values プロパティでは、空の値は追加していません。
しかし、ScheduleTime 項目属性で、Empty Item プロパティを True にしているため、空の値が選択肢として表示される実装になっています。

このトランザクションは、前述の通り、主キーは複数の項目属性に基づく複合主キー(下図)です。そのため、自動採番は利用できません。

この設計に基づいた場合、同じ映画が同じ日時に異なる 2 つの部屋で上映することは記録できるでしょうか？



答えは、要件としても定義されている通り「はい」です。

また、残座席を記録する ScheduleAvailableSeats 項目属性の値は、部屋の定員が記録されている RoomSeats 項目属性の値に基づき既定値が設定されます。

それでは、上映一覧(WWSchedule)を実行し、「追加」ボタンをクリックした画面で、新しい上映情報を記録します。

この時、パターンによって生成されたルール以外に基づき、次の 5 点を確認してください。

1. 部屋番号を指定しない場合、どうなるでしょうか？
確認後、存在する部屋を指定します。
2. 上映日に今日より前の日付を指定した場合、どうなるでしょうか？
確認後、今日より後の日付を指定します。
3. 上映時刻を「(なし)」のままにし、次の項目へフォーカスを進めた場合、どうなるでしょうか？
確認後、「(なし)」以外の択肢のいずれかを選択します。
4. 映画番号を指定しないもしくは、存在しない値を指定した場合、どうなるでしょうか？
確認後、存在する映画を指定します。
5. 部屋座席数と上映残座席数はなぜ読み取り専用になっているのでしょうか？
また、いつ値が入力されたでしょうか？
値が入力されたタイミングを見逃している場合には、もう一度新規登録を開始し、部屋番号に値を入力し、フォーカスを次の項目へ進めるタイミングで注目します。

その他のルール

特定の上映を顧客へ販売したことを記録している「Ticket」トランザクションを確認します。
もし、顧客が「Customer」トランザクションで登録されている場合、紐づけが行えます。
登録されていない場合、顧客に関する情報は空のままにできます。
また、チケット購入日は、購入日の日付となります。

チケットには、上映スケジュールの紐づけとともに、販売した座席数が記録されます。
合計金額は、販売した座席数と、映画が上映される部屋で定められた価格から算出されます。

名前	タイプ	デスクリプション	式	Null 許容
 Ticket	Ticket	チケット		
• TicketId	Id	チケット番号		No
• TicketDate	Date	チケット購入日		No
• CustomerId	Id	顧客番号		Yes
• CustomerName	Name	顧客名		
• RoomId	Id	部屋番号		No
• RoomName	Name	部屋名称		
• RoomType	RoomType	部屋タイプ		
• ScheduleDate	Date	上映日		No
• ScheduleTime	MovieTime	上映時刻		No
• MovieId	Id	映画番号		
• MovieName	Name	映画名称		
• RoomSeats	Numeric(4.0)	部屋座席数		
• RoomPrice	Price	部屋価格		
• TicketSeatsQuantity	Numeric(4.0)	チケットの座席数		No
• TicketTotal	Price	チケット合計金額	RoomPrice*TicketSeatsQuantity	

以下の要件が満たしているか定義されたルールを確認してください。
満たしていないと判断できるものがある場合、修正を行ってください。

「Ticket」トランザクション

- チケット購入日は、既定で現在の日付が入力されるようにする。
また、新規登録時は変更できず、その他のモードでは、変更を許可する。
- 新規登録時のみ上映日に指定できる日付は、チケット購入日より後の日付だけにする。

チケット(Ticket トランザクション)を実行し、ルールが実行されていることをデータの登録および更新を通して確認します。

特に以下の 5 点に注目します。

1. チケット購入日(TicketDate)に関連するルールがトリガーされるタイミング
2. 登録したデータを読み込み、チケット購入日を変更できるか。
3. 2 ができる場合、上映日より後の日付に変更し、保存することができるか。
4. 3 ができる場合、なぜ要件を違反したデータが登録できるか確認します。
5. 登録したデータを読み込み、「顧客番号」や「上映日」が変更できるか。

追加の要件が提示されたため、これを満たすルールを追加してください。

1. チケット購入時に指定する座席数が選択した上映における残座席数を越えて指定できないようにしたい。
もし、指定した座席数を確保できない場合、販売を行わない(データを記録しない)ようにしたい。
2. 上映残座席数(ScheduleAvailableSeats)から購入された座席数分減算したい。
3. チケットに顧客が紐づけられた場合、ポイントを加算する。
ポイントは、購入したチケットの合計金額の 10%です。

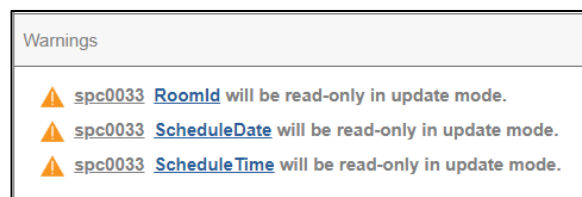
注:トランザクションオブジェクトの Rules エlementで使用する項目属性は、外部参照キーの従属項目であっても、Structure Element内に定義されている必要があります。

なお、「Customer」トランザクションには、ポイントを記録するための項目属性として、CustomerPoints が定義されています。

ルールの追加後、以下の点を確認してください。

(想定される動作は、本コースで学習した実装方法を最適に実装した場合です。)

1. 「Ticket」トランザクションを実行し、顧客を紐づけず、利用可能な座席数を指定したチケットを新規登録します。
登録後、「WWSchedule」を実行し、対象の上映スケジュールの残座席数が更新されていることを確認します。
2. 1 点目で登録したチケットを読み込み、部屋番号、上映日、上映時間に変更できないことを確認します。(変更できる場合、本課題想定通りの実装ではありません)
GeneXus に戻り、ナビゲーション表示の Ticket トランザクションに対して表示される警告を確認します。(下図の通り。CustomerId については一度省略)



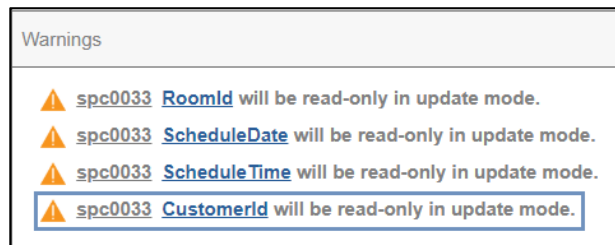
これらの警告は、Subtract ルールの追加後に表示されるようになります。

この事実を確認するため、Subtract ルールの行をコメントアウトし、Ticket トランザクションを右クリックし、「ナビゲーションを表示」オプションを利用し、ナビゲーション表示を再表示します。(オブジェクトは再生成(ビルド)しません)

その結果、警告は表示されていないことを確認できます。

コメントアウトを元に戻し、なぜこの警告が表示されるか考えてみてください。

3. 「Ticket」トランザクションを実行し、任意の顧客を紐づけ、利用可能な座席数を指定したチケットを新規登録します。
登録後、「WWSchedule」を実行し、対象の上映スケジュールの残席数が更新されていることと、「Customer」を実行し、紐づけた顧客のポイントが更新されていることを確認します。
4. 3 点目で登録したチケットを読み込み、顧客番号が変更できないことを確認します。
(変更できる場合、本課題想定通りの実装ではありません)
GeneXus に戻り、ナビゲーション表示の Ticket トランザクションに対して表示される警告を確認します。(下図の通り)
この警告は、Add ルール追加後に表示されるようになります。
なぜこの警告が表示されるか考えてみてください。



5. 3 点目で登録したチケットを読み込み、座席数を変更します。
変更後、「WWSchedule」を実行し、対象の上映スケジュールの残座席が更新されていることと、「Customer」を実行し、紐づけた顧客のポイントが更新されていることを確認します。
6. 登録済みのチケットを読み込むか、新規登録を開始し、上映残座席数を越える座席数を指定し、保存できないことを確認します。
7. 3 点目で登録したチケットを読み込み、削除します。
「Customer」を実行し、紐づけられていた顧客のポイントが元の値に戻ることを確認してください。

「Ticket」トランザクションの Structure は、以下の構造になっていますか？
 (結果が要件を満たしている場合、「TicketAddPoint」項目属性の名前、式の計算式は、一致する必要はありません)

名前	タイプ	デスクリプション	式	Null 許容
Ticket	Ticket	チケット		
TicketId	Id	チケット番号		No
TicketDate	Date	チケット購入日		No
CustomerId	Id	顧客番号		Yes
CustomerName	Name	顧客名		
CustomerPoints	Numeric(4.0)	顧客保有ポイント		
RoomId	Id	部屋番号		No
RoomName	Name	部屋名称		
RoomType	RoomType	部屋タイプ		
ScheduleDate	Date	上映日		No
ScheduleTime	MovieTime	上映時刻		No
ScheduleAvailableSeats	Numeric(4.0)	上映残席数		
MovieId	Id	映画番号		
MovieName	Name	映画名称		
RoomSeats	Numeric(4.0)	部屋座席数		
RoomPrice	Price	部屋価格		
TicketSeatsQuantity	Numeric(4.0)	チケットの座席数		No
TicketTotal	Price	チケット合計金額	RoomPrice*TicketSeatsQuantity	
TicketAddPoint	Numeric(4.0)	チケット加算ポイント数	TicketTotal * 0.1	

また、部屋タイプは、外部参照キーの従属項目であるため、「Ticket」トランザクションからは、変更できません。

この画面からも変更できる方法があるため、ルールを追加し、挙動を確認します。

エラーメッセージの内容についても要望がありました。

上映情報を管理する「Schedule」トランザクションを参照する場合、「RoomId」、「ScheduleDate」、「ScheduleTime」の組み合わせが外部キーでした。

そのため、すべての項目に値が入力されなければ、整合性チェックが行われず、もし違反していた場合に表示されるメッセージも明確ではありません。(下図の通り)

部屋番号	1	
部屋名称	ルーム 1	
部屋タイプ	通常	
上映日	26/09/10	
上映時刻	午前1	
'Schedule' の該当レコードはありません。		
上映残席数		

下図のメッセージが表示される場合、より明確になります。

部屋番号	1 
部屋名称	ルーム 1
部屋タイプ	通常 
上映日	26/09/10 
上映時刻	午前1 
 部屋番号：1、上映日：26/09/10、上映時間：午前11時で予定されている映画はありません。	
上映座席数	

この要望を満たすように実装を行ってください。

なお、外部参照キーは前述の通り複合主キーであるため、カンマ区切りで複数記載する必要がある点に注意します。

表示するメッセージ例として、以下の文字列を変数に代入し、利用可能です。

”部屋番号：” + RoomId.ToString() + ”、上映日：” + ScheduleDate.ToFormattedString() + ”、上映時間：” + ScheduleTime EnumerationDescription() + ”で予定されている映画はありません。”

アプリケーションを実行し、追加したルールによる挙動を確認してください。

オプション(任意実施課題)

「Customer」トランザクションには、ポイントと引き換えに受け取った賞品も記録できました。賞品は、対象の顧客固有となるため、弱いエンティティとして実装されています。(下図の通り)

名前	タイプ	デスクリプション
Customer	Customer	Customer
CustomerId	Id	顧客番号
CustomerName	Name	顧客名
CustomerEmail	Email, GeneXus	顧客メールアドレス
CustomerVIP	Boolean	顧客VIPフラグ
CustomerPoints	Numeric(4.0)	顧客保有ポイント
CustomerLastRewardId	Id	顧客最終賞品番号
Reward	Reward	賞品
CustomerRewardId	Id	顧客賞品番号
CustomerRewardDescription	Description	顧客賞品詳細
CustomerRewardPoints	Numeric(4.0)	顧客賞品消費ポイント

各賞品を明細に入力する時点で、消費ポイントの値に基づき、保有ポイントを更新する必要があります。

もし、消費ポイントが保有ポイントを越える場合、保存することはできない必要があります。

この要件を満たす実装を行い、実行結果を確認してください。

トランザクションにおけるルールのトリガーイベント(対象教材動画:4~6)

以降の課題を進める前に、トランザクションにおけるルールの「トリガーイベント」について実行される順序を参照できるように書き出しておくことを推奨します。

明確な順序を認識したうえで、課題を進めることで学習効果を高めることができます。

課題を進めるにあたり、「Movie」トランザクションの Work With Web パターンインスタンスで、「Selection (映画 s)」ノード配下の「Attributes」ノード内の「MovieDescription」ノードの Visible プロパティを False にすることを推奨します。また、トリガーイベントの課題については、以下の映画情報に基づき、確認を進めてください。

映画名称	ステラ・エコー
映画監督名	ミア・ハルストロム
国名	アメリカ
カテゴリー名	ドラマ
出演者	クラーク・ケブル
	オリヴィア・デ・ナヴィランド
	ヴィヴィアン・レイ

トリガーイベントを利用したルール

映画の登録を行うために活用できる Web サービスがあると仮定します。

この Web サービスは、映画のタイトルを送信すると、公開年を取得できます。

ただし、この時点で GeneXus における Web サービスについて扱っていないため、初めにインポートされたオブジェクトのうち、「GetMovieYear」プロシーチャーの呼び出しで代用します。

「Movie」トランザクションに以下のルールを追加した場合、どのような挙動になるでしょうか？

```
MovieYear = GetMovieYear(MovieName) If Insert;
```

実行して確認する前に、このルールがどのタイミングで実行されるかを考えましょう。

新しい映画を登録する場合であることは明らかですが、その中でより具体的なタイミングは、次のうちどのタイミングでしょうか？

1. トランザクションが新規登録画面として表示され、映画名称項目に移動する前
2. 映画名称項目から映画公開年項目へフォーカスが移動されるとき
3. 映画公開年項目からフォーカスが進むとき
4. 実行ボタンをクリックするとき

挙動を明確に確認するため、「Movie」トランザクションの Structure エlementで、MoviePoster を MovieName と MovieYear の間に移動した後、「Movie」トランザクションを実行します。

アプリケーションで、映画名称に「ステラ・エコー」と入力し、映画ポスターには、本資料と同梱された Movies フォルダ内から画像を探し、選択します。

そのほか、事前に提示された内容に沿ってデータを入力し(映画公開年は空のまま)、実行ボタンをクリックし、登録を完了します。

登録されたデータを表示し、映画名称を変更しても映画公開年が変わらないことを確認します。

では、先ほどのルールの条件を以下のように変更し、いつトリガーされるか確認します。

```
MovieYear = GetMovieYear(MovieName) If MovieYear.IsEmpty() on BeforeInsert;
```

実行し、再度「ステラ・エコー」という名称の映画を提示済み内容に沿って登録します。

実行ボタンを押すまでにルールは実行されませんでした。

なぜ変更前と違い、実行ボタンを押した後にルールが実行されたのでしょうか？

先ほど登録されたデータは削除し、もう一度「ステラ・エコー」の登録を行います。

ただし、今回は、事前に提示された内容に加え、映画公開年に「2026」と入力し、登録します。

登録後、定義されているルールが実行されたか確認します。

では、以下に記載した現在定義しているルール(No.1)と No.2 のルール定義では、どのような違いがあるのでしょうか？

1. `MovieYear = GetMovieYear(MovieName) If MovieYear.IsEmpty() on BeforeInsert;`
2. `MovieYear = GetMovieYear(MovieName) If MovieYear.IsEmpty() and Insert on AfterValidate;`

答えは、結論としては実質的に同じ結果になります。

AfterValidate は、BeforeInsert/BeforeUpdate/BeforeDelete よりも前に実行されるトリガーイベントですが、この間に実行される他のトリガーイベントはありません。

また、条件として、新規登録時に True を返す「Insert」が条件に指定されているため、更新や削除時には実施されないためです。

次に、定義したルールのトリガーイベントを BeforeInsert の直後のタイミングとなる AfterInsert に変更し、実行したアプリケーションで、映画公開年を空のまま「ステラ・エコー」を再登録します。登録されたデータの映画公開年の値は、「1939」と指定されたのでしょうか？確認後、登録されたデータは削除し、ルールのトリガーイベントを BeforeInsert に戻します。

第 2 レベル以降の行単位のルールとトリガーイベント

続いて実際には Web サービスの利用を想定した実装ですが、ナレッジベース内のオブジェクトで代用し、実装を進めます。

映画名称と出演者名を引数に受け取り、その出演者が映画の主演だった場合、True を返す Web サービスによって、主演であるかどうかを自動で設定します。

本課題においてこの実装を行うため、MovieStarringActors SDT 型を出力する GetStarrings データプロバイダーがあり、このデータプロバイダー内に主演かどうかの情報が記録されています。

そのため、このデータプロバイダーに基づき、結果を返す IsStarring プロシージャを利用します。

「Movie」トランザクションに以下のルールを追加します。

```
MovieActorStarring = IsStarring(MovieName, ActorName) If Insert;
```

アプリケーションを実行し、改めて「ステラ・エコー」を提示された内容に沿って入力します。

明細で、出演者を入力するとこのルールが実行されていることが確認できます。

つまり、クライアントサイドでルールが実行されます。(登録時、サーバーサイドでも実行されます)

出演者を入力し、ルールが実行される正確なタイミングについて確認してください。

この映画の登録を確定するために、実行ボタンをクリックするまでにルールは何度実行されたのでしょうか？

また、実行ボタンをクリックした後、サーバーサイドで実行されるタイミングはいつでしょうか？

同じ Web サービス(本課題ではプロシージャオブジェクト)を利用し、実装する機能が別の要望だったケースを題材に課題を継続します。

そのため、先ほど追加した自動入力のルールは削除(またはコメントアウト)します。

ユーザーは、主演俳優であるかどうかを自動で入力することは望まず、手動で入力します。

ただし、Web サービスから受け取る値が False で、主演(True)に設定されている場合、警告を表示することを要望しています。

この要望を満たすルールの実装は以下の通りです。

```
Msg("主演として登録しますか?") If MovieActorStarring <> IsStarring(MovieName, ActorName);
```

そして、ユーザーが False にしている場合、Web サービスから取得した値で上書きを行う必要があるということで、以下のルールも必要です。

```
MovieActorStarring = IsStarring(MovieName, ActorName) If Insert and MovieActorStarring = False;
```

新たなルールが定義されたため、改めて提示された内容に沿って「ステラ・エコー」のデータ入力を進めます。

この時、定義されたルールが、映画出演者主演フラグから次のフィールドへフォーカスを移動した後に実行されていることを確認してください。

登録する出演者の「クラーク・ケブル」、「オリヴィア・デ・ナヴィランド」の映画出演者主演フラグを False のままにすることで挙動を確認できます。

「クラーク・ケブル」のフラグがチェック状態になるため、手動でチェックを外しますが、フォーカスを移動すると、再度チェックされた状態に変わることを確認できます。つまり、ルールが再実行されていることを確認できます。

ここまでの内容に基づき、ユーザーは各出演者の映画出演者主演フラグを必要に応じて手動で変更できる状態になりました。

映画出演者主演フラグが False の場合、実行ボタンがクリックされた後にのみ Web サービス(プロシージャ)による値の上書きが行われるようにするには、ルールをどのように変更すべきでしょうか？

次の 4 つの選択肢が想定されます。

1. `MovieActorStarring = IsStarring(MovieName, ActorName) If MovieActorStarring = False on BeforeInsert;`
2. `MovieActorStarring = IsStarring(MovieName, ActorName) If MovieActorStarring = False on AfterInsert;`
3. `MovieActorStarring = IsStarring(MovieName, ActorName) If MovieActorStarring = False on AfterLevel Level ActorId;`
4. `MovieActorStarring = IsStarring(MovieName, ActorName) If MovieActorStarring = False on Aftercomplete;`

これまでに出てきた次の 2 つのルールについて注目します。

1. `MovieYear = GetMovieYear(MovieName) if MovieYear.IsEmpty() on BeforeInsert;`
2. `MovieActorStarring = IsStarring(MovieName, ActorName) if MovieActorStarring = False on BeforeInsert;`

GeneXus は、1 つ目のルールは、ヘッダーとなる MOVIE テーブルへの新規登録処理の前に実行し、2 つ目のルールは、明細となる MOVIEACTOR テーブルに対し、各行に対する行データの登録処理の前に実行することを決定します。

この挙動は、どのようにして決定されているのでしょうか？

オプション

理解を深めるため、次のような確認も行えます。

「Movie」トランザクションを右クリックし、「名前を付けて保存」を選択し、「MovieCopy1」として保存し、Rules エlement、Events Element内のコードをすべて削除します。

また、「MovieCopy1」の Data Provider プロパティを False に変更します。

その後、Numeric 型の ActorLine 変数を定義し、以下の 2 つのルールを定義します。

```
&ActorLine = &ActorLine + 1
    on AfterInsert
        Level ActorId;

Msg(&ActorLine.ToString() + "人の出演者。映画監督：" + FilmDirectorName)
    If Insert
        on AfterLevel
            Level ActorId;
```

これらのルールによって、MOVIEACTOR テーブルにデータを新規登録するたびに、ActorLine 変数が 1 ずつ加算され、すべての行に対する操作が完了した時点でメッセージが出力されます。

実際に実行し、結果を確認してください。

次に、ActorLine 変数を加算しているルールの Level 節の後に記載された「ActorId」を「FilmDirectorId」に変更し、どのような結果となるか確認してください。

ルールとグローバル式をトリガーするための評価ツリー

ここまでに、「Movie」トランザクションにおける以下のルールの実行タイミングについて確認を進めていました。

```
MovieYear = GetMovieYear(MovieName) If Insert;
```

Web Layout エLEMENTで、コントロールが下図の通り配置されていることが確認できます。

The screenshot shows a web form with the following elements:

- 映画 (Movie)
- <ErrorViewer: ErrorViewer>
- <Toolbar>
- 映画番号 MovieId
- 映画名称 MovieName
- 映画ポスター (Movie Poster) with a placeholder image
- 映画公開年 MovieYear
- 映画説明 MovieDescription
- 国番号 CountryId

この配置を前提に、ルールは、「可能な限り早く」実行されると仮定すると、MovieYear 項目属性へ値を代入するために情報が必要となり、それはトランザクションのモードが Insert であるかと、MovieName 項目属性に入力された名前でした。

このルールの定義に対し、2 つのケースからさらに確認を行います。

- もし、パターンが適用された場合のように、パラメーターとしてモードを受け取る場合、トランザクションの画面が表示された時点でモードが確定しています。Insert モードの指定を受け取った場合、その時点で新規登録であることが確定します。ただし、今回のルールでは、MovieName 項目属性の値も必要であるため、この項目属性のフィールドからフォーカスを移動するまではルールは実行できません。では、以下のルール定義だった場合、どうなるか考えてください。

```
MovieYear = GetMovieYear() If Insert;
```

このルールでは、トランザクションが新規登録(Insert モード)で表示された時点で、ルールが実行されます。

ただし、引数に MovieName を指定せず、プロシージャオブジェクトを呼び出すため、MovieYear には何も代入されません。

- もし、Parm ルールの定義がなく、パラメーターとしてモードを受け取らない場合、モードの特定は、MovieId 項目属性のフィールドからフォーカス移動が発生するまでできません。

しかし、フォーカスが移動したとしてもルールの実行は行われません。

その理由は、ルールの実行は、MovieName 項目属性の値も必要であるため、この項目属性のフィールドからフォーカスを移動するまではルールは実行できません。

では、以下のルール定義だった場合、どうなるか考えてください。

```
MovieYear = GetMovieYear() If Insert;
```

このルールでは、MovieId 項目属性のフィールドからフォーカスが移動された時点で、ルールが実行されます。

ただし、引数に MovieName を指定せず、プロシージャオブジェクトを呼び出すため、MovieYear には何も代入されません。

では、「Movie」トランザクションに定義されているルールが以下の通りの順序で記述されていた場合、それぞれのルールが実行されるタイミングはどうなるでしょうか？

```
MovieYear = GetMovieYear(MovieName) If Insert;
```

```
Error("映画名称が未入力です。")If MovieName.IsEmpty();
```

この 2 つのルールは、どちらも MovieName 項目属性のフィールドからフォーカスを移動した場合に実行されます。

同じタイミングで実行されるルールの定義のため、記述された順でルールが実行されます。

ただし、1 つ目のルールは、新規登録時のみで、2 つ目はモードにかかわらず MovieName 項目属性のフィールドが空の値の場合のみ実行されます。

もし、上記 2 つのルールの前に Msg ルールを追加し、下記のように定義された場合、この Msg ルールはいつ実行されるでしょうか？

```
Msg("この映画はクラシック映画の時代に属します。")
```

```
    If MovieYear < 1950;
```

```
        MovieYear = GetMovieYear(MovieName)
```

```
        If Insert;
```

```
            Error("映画名称が未入力です。")
```

```
            If MovieName.IsEmpty();
```

Msg ルールは、MovieYear 項目属性に依存しているため、2 番目の MovieYear 項目属性に値を代入するルールより後で、入力された値によって実行が決定するため、MovieYear 項目属性からフォーカスが移動タイミングで実行されます。

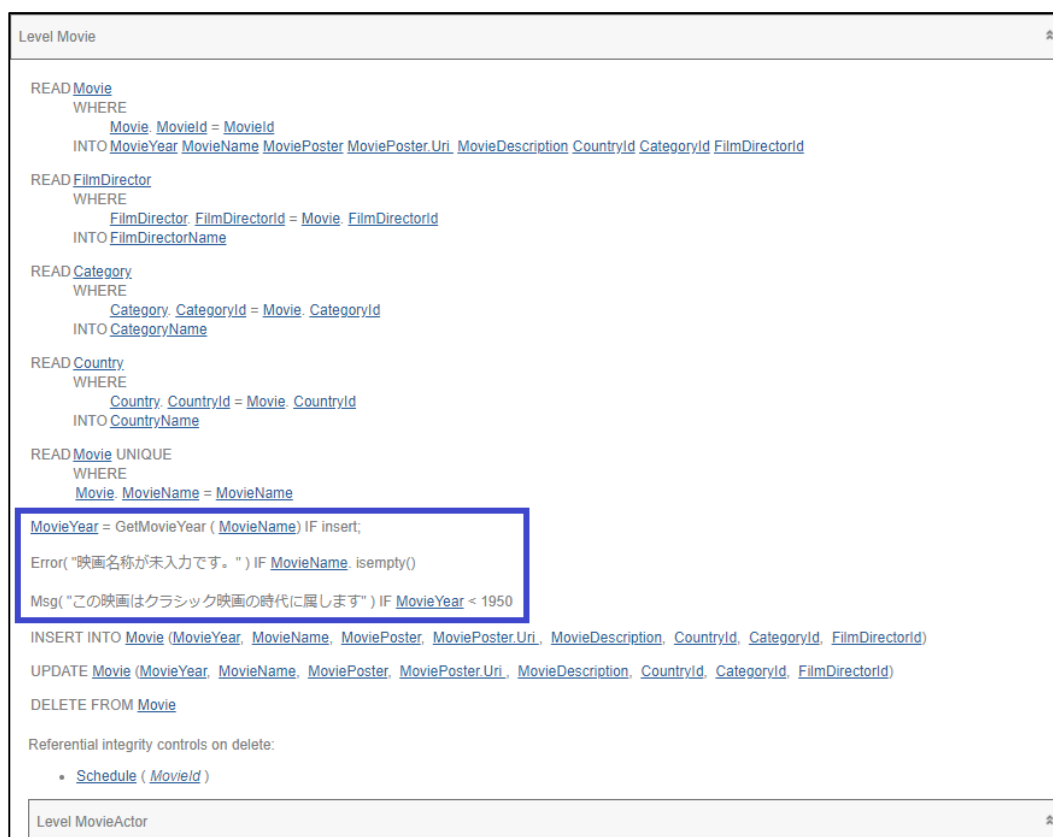
この実行順序を決定するために重要となるのが、依存関係に基づき内部的に決定される評価ツリーと、実行画面のコントロールの配置、ユーザーによる操作です。

「Movie」トランザクションにおいて、次のフローを考えてみます。

1. 映画の新規登録を開始
2. MovieName 項目属性のフィールドに既存の映画と同じ映画名称を入力
(MovieYear 項目属性のフィールドに自動的に値が代入されます)
3. MoviePoster 項目属性に任意の画像を指定
4. MovieYear 項目属性のフィールドにフォーカスを移動し、次のフィールドへフォーカスを移動
(変更の有無にかかわらず、Msg ルールの条件判定が行われ、条件を満たす場合、実行)
5. 手順 2 で入力した映画名称を変更するため、MovieName 項目属性へ移動し、入力後、フィールドからフォーカスを移動

手順 5 の後、どのような挙動が発生することが想定されますか？

ナビゲーション表示を詳細表示にし(メニューバーの「ツール」→「オプション」を選択し、「ビルド」ノードを選択して表示される「ナビゲーション詳細」を True)、「Movie」(または「[MovieCopy1](#)」)を右クリックし、「ナビゲーションを表示」をクリックし、表示を確認します。
(下図は、「MovieCopy1」で変更を行い、表示したもの)



もし、下記のように条件をトリガーイベントに置き換えた場合、どうなるでしょうか？

```
Msg("この映画はクラシック映画の時代に属します。")
```

```
  If MovieYear < 1950;
```

```
    MovieYear = GetMovieYear(MovieName)
```

```
    on BeforeInsert;
```

```
Error("映画名称が未入力です。")
```

```
  If MovieName.IsEmpty();
```

この場合、MovieYear 項目属性に値を代入するルールは、各フィールドの処理が行われる際の評価ツリーと独立して動作するようになります。

(各フィールドの処理は、クライアントサイド、サーバーサイド両方で実施されます。サーバーサイドの場合、BeforeValidate トリガーイベントに基づく実行タイミングの前です)

MovieYear 項目属性に値が代入されるルールは、サーバーサイドでの実行のみとなり、各フィールドの処理や関連するルールの実行からかなり遅れて実行されます。

具体的には、実行ボタンがクリックされ、ヘッダーのフィールドに関するすべての検証が終了し、テーブルへの Insert 文が発行される直前です。

このルールの定義がある状態で、ナビゲーション表示を再度表示し、以前の表示と比較します。

The screenshot shows the 'Level Movie' editor with the following content:

```

READ Movie
WHERE
  Movie.MovieId = MovieId
INTO MovieYear MovieName MoviePoster MoviePoster.Uri MovieDescription CountryId CategoryId FilmDirectorId

READ FilmDirector
WHERE
  FilmDirector.FilmDirectorId = Movie.FilmDirectorId
INTO FilmDirectorName

READ Category
WHERE
  Category.CategoryId = Movie.CategoryId
INTO CategoryName

READ Country
WHERE
  Country.CountryId = Movie.CountryId
INTO CountryName

READ Movie UNIQUE
WHERE
  Movie.MovieName = MovieName

Error("映画名称が未入力です。") IF MovieName.isempty()

Msg("この映画はクラシック映画の時代に属します") IF MovieYear < 1950

Before Insert Rules
  MovieYear = GetMovieYear ( MovieName);

INSERT INTO Movie (MovieYear, MovieName, MoviePoster, MoviePoster.Uri, MovieDescription, CountryId, CategoryId, FilmDirectorId)

UPDATE Movie (MovieYear, MovieName, MoviePoster, MoviePoster.Uri, MovieDescription, CountryId, CategoryId, FilmDirectorId)

DELETE FROM Movie

Referential integrity controls on delete:
  • Schedule ( MovieId )
  
```

A blue box highlights the following section:

```

Error("映画名称が未入力です。") IF MovieName.isempty()

Msg("この映画はクラシック映画の時代に属します") IF MovieYear < 1950

Before Insert Rules
  MovieYear = GetMovieYear ( MovieName);
  
```

新規要望の実装①

映画を保存する場合、最低 2 人の出演者が登録される必要があります。
もし、この条件を満たせない場合、映画の保存を許可しない制御が必要です。
例えば、下図のように登録を許可せず、メッセージを表示します。

この実装には、MovieActorsQty 項目属性という入力された出演者の数を格納する項目属性の定義が必要です。

この項目属性を利用し、Error ルールの定義が必要です。

以下の Error ルールの条件はどのような記述とする必要があるでしょうか？

Error("2人以上の出演者の登録が必要です。")

If <条件>

on <トリガーイベント>;

この要望は、トリガーイベントを「AfterLevel Level MovieActorStarring」と記述することで満たすことができました。

では、「BeforeComplete」を利用しても要件を満たせるでしょうか？

例えば、サウンドトラックを記録するため、出演者を記録する第 2 レベルと並行する第 2 レベルがある場合で考えてみてください。

新規要望の実装②

映画館では、制作会社からフィルムの納品についても管理したいという要望がありました。そのため、映画の情報(ヘッダーおよび明細)の入力を行い、データの登録(コミット)が行われる前に、フィルムが受領済みか確認できるように実装が必要です。

もし、届いていない場合、データは登録されないようにします。

KB 内にフィルムの受領有無を検証する「MovieInStock」プロシージャが用意されています。本課題では、このプロシージャオブジェクト内のコード内容はフォーカスしません。(テスト目的のため、ランダムに True か False を返す実装です)

「Movie」トランザクションで、IsInStock という Boolean 型変数を用意し、以下のルールの定義が想定されます。

```
&IsInStock = MovieInStock(MovieId) If Insert;
```

ただし、このルールだけでは要件を満たしていません。

今回の要件においては、そのプロシージャオブジェクトは、MOVIE テーブルにアクセスすることが想定されているため、少なくともヘッダーのレコードが挿入された後に実行する必要があります。

しかし、実際には可能な限り処理の実行を遅らせることが望ましい状況です。

この要件を満たすため、ルールにはどのようなトリガーイベントを追加すべきでしょうか？また、フィルムが届いていない場合、映画を登録しないという要件を満たすために、これ以外のルールが必要な場合、その記述も行います。

処理を実装し、確認する場合、プロシージャオブジェクトはランダムな値を返すため、明確に確認するためにプロシージャオブジェクトを変更しても問題ありません。

オプション: 更新モードでの行単位の制御

「Movie」トランザクションに少し変更を加えます。

MovieStatus という同名の列挙型ドメイン(初回インポート済)型の項目属性をヘッダーに追加します。

この列挙型ドメインは、「保留」と「完了」という 2 つの値から選択できます。

この項目属性により、完全な情報がなくても映画の登録を開始できます。

この変更に基づき、出演者の登録に関するルールの実行は、MovieStatus を「完了」に設定した後に実行するように変更する必要があります。

MovieStatus を追加した Structure エlement は下図の通りです。

名前	タイプ
Movie	Movie
MovieId	Id
MovieName	Name
MoviePoster	Image
MovieYear	Numeric(4.0)
MovieDescription	LongVarChar(1M)
CountryId	Id
CountryName	Name
CategoryId	Id
CategoryName	Name
FilmDirectorId	Id
FilmDirectorName	Name
MovieStatus	MovieStatus
Actor	Actor
ActorId	Id
ActorName	Name
MovieActorStarring	Boolean
MovieActorsQty	Numeric(4.0)

本実装を正しく実装できることを確認するため、「Movie」トランザクションで定義されているパターンによるもの以外のルールは一度コメントアウトしておきます。

そのうえで、「ok」という Boolean 型変数を定義し、以下のルールを定義します。

どのトリガーイベントを利用すると適切に動作するでしょうか？

```
Default(MovieStatus, MovieStatus.Pending);

&ok = CheckMovieActors(MovieId)
    If MovieStatus = MovieStatus.Completed
        on <トリガーイベント>;

Error("出演者に問題があります。")
    If not &ok and MovieStatus = MovieStatus.Completed
        on <トリガーイベント>;
```

「CheckMovieActors」プロシーチャーは、インポート済です。

実行結果を確認する前に、Web Layout エlement で、MovieStatus 項目属性が追加されていることを確認してください。

もし、追加されていない場合、メニューバーの「編集」→「既定を適用(Web Layout)」をクリックします。

アプリケーションを実行し(再編成が必要)、もし「ステラ・エコー」が登録されたままの場合、削除し、再度「[ステラ・エコー](#)」として提示された内容に基づき入力を行います。

MovieStatus は「保留」のままにし、出演者を登録する中で「オリヴィア・デ・ナヴィランド」の主演フラグにチェックを入れ、保存します。

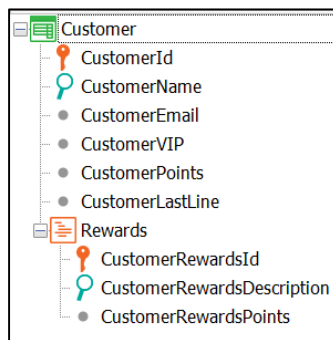
登録した「ステラ・エコー」を更新モードで表示し、MovieStatus を「完了」に変更し、実行ボタンをクリックします。

正常に保存できたでしょうか？

GetStarrings データプロバイダーで記録された俳優のみが主演となるように修正することで、データが保存できるようになることを確認します。

オプション: 評価ツリーとトリガーイベントに関する演習

下図の「Customer」トランザクションの定義があります。



Rules エlementには以下の定義が行われています。

実行される順序について以下の括弧内に記載してください。

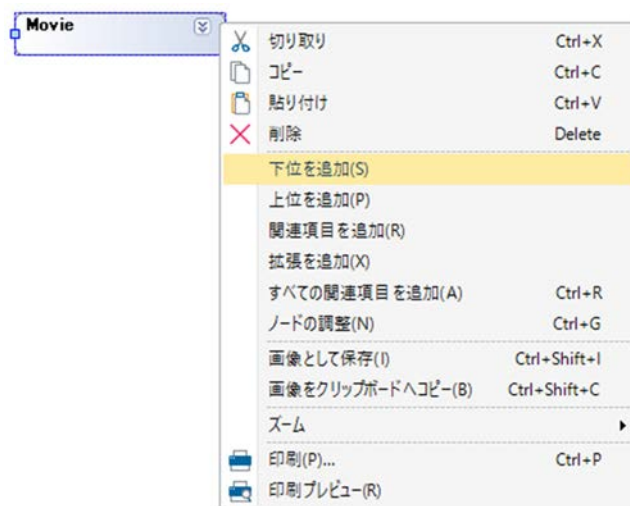
なお、一部のオブジェクトは定義されていないため、実際にルールを定義し、確認することは必須ではありません。

```

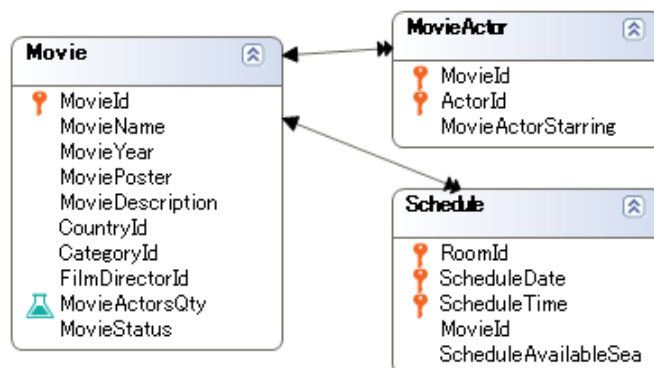
( )Error("顧客名が未入力です。") If CustomerName.IsEmpty();
( )Msg("顧客メールアドレスが未入力です。") If CustomerEmail.IsEmpty();
( )PrintCustomerList() on AfterComplete;
( )Subtract(CustomerRewardsPoints, CustomerPoints);
( )Error("Not enough points") If CustomerPoints < 0;
( )PrintRewards(CustomerId, CustomerRewardsId) on AfterInsert;
( )Serial(CustomerRewardsId, CustomerLastLine, 1);
( )NoAccept(CustomerRewardsId);
( )Checkpoints(CustomerId) If Update on AfterInsert;
  
```

トリガーイベントの活用

Diagram タイプのオブジェクトを新規作成し、MOVIE テーブルをドラッグして配置します。
 下図のように MOVIE テーブルを右クリックし、「下位を追加」をクリックします。
 この結果、MOVIE テーブルを外部参照するテーブルがすべて追加されます。



追加されるテーブルは 2 つだけです。



1 つは、「Movie」トランザクションの第 2 レベルに基づくテーブルです。
 もう 1 つは、「Schedule」トランザクションに基づくテーブルでした。

では、「Schedule」トランザクションから参照されている映画を「Movie」トランザクションから削除しようとする、どうなるでしょうか？

試してみると、参照整合性チェックが発生し、削除できません。

では、この制御はどのタイミングで実行されるでしょうか？

もし、参照整合性チェックの前に関連するスケジュールを削除し、エラーにならずに映画を削除できるように挙動を変更したい場合、どのように実装できるでしょうか？

あらかじめ MovieId をパラメーターとして受け取り、その映画を参照するスケジュールをすべて削除する「DeleteSchedules」プロシーチャーが取り込まれています。

以下のように、このプロシーチャーオブジェクトの呼び出しをモードの条件と、トリガーイベントの指定で要件を満たせますが、どのタイミングが適切でしょうか？

```
DeleteSchedules(MovieId)
  If Delete
    on <トリガーイベント>;
```

「MOVIE」テーブルからレコードを削除する前で、参照整合性のチェック(検証)を行う前を指定する必要があります。

以下のトリガーイベントが候補となり、適切なタイミングを選択してください。

1. BeforeValidate
2. AfterValidate
3. BeforeDelete
4. AfterDelete
5. BeforeComplete
6. AfterComplete

実行結果を確認する際、チケットから参照されていないスケジュールのみの映画を対象に削除を行います。

もし、「DeleteSchedule」プロシーチャーで削除するスケジュールが、チケットから参照されている場合、データベース例外が発生します。この理由については、後ほど取り上げます。「Schedule」トランザクションの Business Component プロパティを True に変更し、「DeleteSchedule」プロシーチャーの Source エlement に記載された処理を以下のようにビジネスコンポーネントによる削除に変更した場合、どうなるでしょうか？
(「schedule」という Schedule 型変数は別途定義)

```
For each Schedule
  &schedule.Load(RoomId, ScheduleDate, ScheduleTime)
  &schedule.Delete()
Endfor
```

チケットから参照されているスケジュールから参照された映画の場合、削除はできませんが、データベース例外の発生は発生しなくなります。

上記ケースに該当しない映画の削除か、チケットを削除し、挙動を確認します。

複合外部キーの部分的な Null 許容 (対象教材動画番号:7)**チケットにおける参照整合性チェックの回避**

このシステムでは、チケットを購入する場合、鑑賞する映画の「部屋番号」、「上映日」、「上映時刻」の指定が必須となっていますが、これらを指定しなくてもチケットを購入できるようにしたいという要望がありました。

この要望を満たすため、「Ticket」トランザクションにおいて RoomId、ScheduleDate、ScheduleTime を指定しない場合に発生する参照整合性チェックを回避する実装を行ってください。

(チケット購入日と上映日を比較するエラールールもコメントアウトする必要があります)

必要な変更を行い、「部屋番号」、「上映日」、「上映時刻」を入力せず、登録することができることを確認してください。(ルールによるエラーも回避する必要があります。)

登録できたデータは削除し、変更内容をもとに戻しておきます。

データが存在するテーブルに外部キー項目属性が追加される場合の影響 (推奨)

従業員情報を管理したいという要望があり、下図のトランザクションを追加します。

Employee	Employee	従業員
EmployeeId	Id	従業員番号
EmployeeName	Name	従業員名
EmployeePhoto	Image	従業員画像

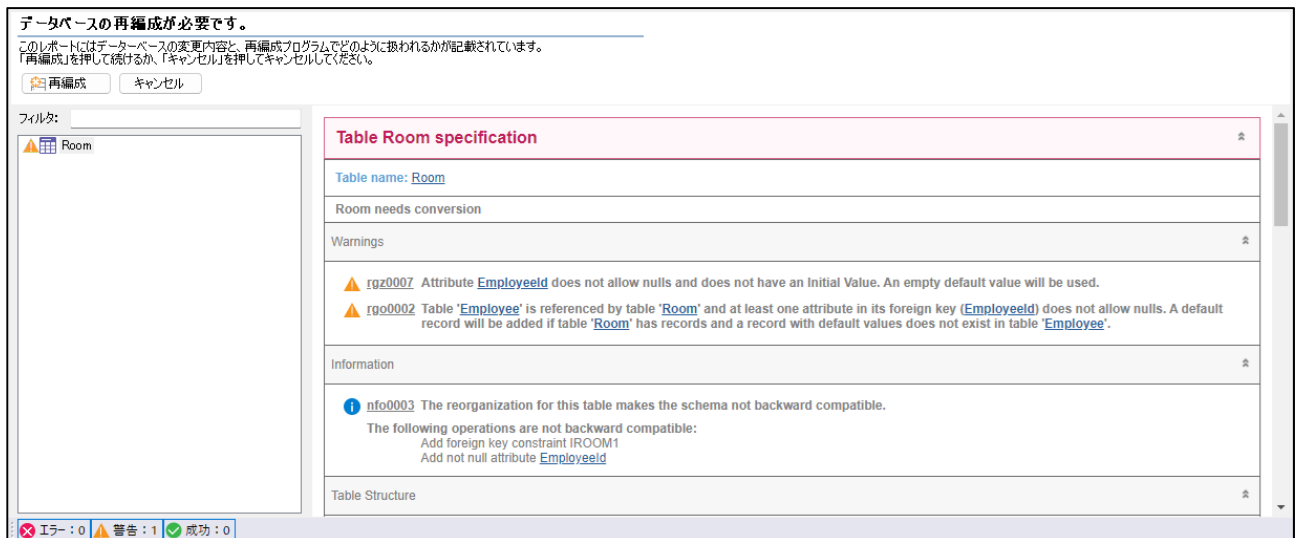
アプリケーションを実行し、従業員レコードを数件登録します。

続いて、各上映用の部屋には、担当の従業員を紐づける必要があり、一人の従業員は、1 つ以上の部屋を担当できるように変更が必要という要望がありました。

GeneXus で、「Employee」トランザクションと「Room」トランザクションを利用し、どのように実装できるでしょうか？

実際にこの要望に基づき、修正を行います。

修正後、メニューバーの「ビルド」→「データベーステーブルの影響分析」をクリックします。本課題想定通りの変更が行われた場合、影響分析の結果は次ページの通りです。



このまま再編成を実行すると、「EMPLOYEE」テーブルに対し、何が起きますか？
この警告に基づく処理を行わせたくない場合、どのような手段があるでしょうか？

並行トランザクション(推奨)

映画会社は、映画館に公開前の映画に関するすべての情報を事前に共有します。しかし、一部の情報は、上映スケジュールが公開されるまで一般公開されません。本システムでは、内部向けに映画のすべての情報が登録、表示可能な画面と、公にすることができる一部の項目(映画名称、映画公開年、映画ポスター、制作した国、カテゴリー、映画監督)だけを登録、表示可能な画面が必要です。

この要件を満たすためには、公にすることができる一部の項目属性のみを含む並行トランザクションを作成することで対応可能です。

実際に作成し、確認を行います。

並行トランザクションは、オプションの課題を実施している場合、ここまでに一度作成を行っています。

パターンを適用していない映画を管理する画面が必要だったため、「[MovieCopy1](#)」という名前で、並行トランザクションを作成する課題がありました。

サブタイプ(対象教材動画番号:8~9)**サブタイプ:複数参照化**

次の要望がありました。

映画には、製作国となる国情報を紐づけています。

そして、出演者も紐づけますが、出演者についても出身国の情報を管理できる必要がある。

「映画の国を保存」し、「出演者の国を参照」できるようにするため、トランザクションの構造は、どのように変国するべきでしょうか？

この要件を満たす 3 つの異なる設計案を検討してください。

それぞれについて、メリットとデメリット(ある場合)を記述し、GeneXus で最も適した設計案を選択します。

選択した設計に基づき、実際に実装を行ってください。

実装を行う際、ここまでの課題で並行トランザクションを作成している場合には、これらの並行トランザクションにも同様の変更が必要な点に留意してください。

設計 1:

メリット	デメリット

設計 2:

メリット	デメリット

設計 3:

メリット	デメリット

サブタイプ:特殊化

このシステムで管理する従業員は、次の分類のいずれかに該当します。

- **事務員(Clerical):**
チケットの販売や、顧客の対応を担当します。この分類となる従業員の場合、他の言語を話すことができるか記録する必要があります。
- **オペレーター(Operators):**
映画館において、映画の上映日時をスケジュールする業務を担当します。この分類となる従業員の場合、22 歳以上であることが条件となるため、年齢が確認できる必要があります。
- **技術者(Technicians):**
映画館内のメンテナンス(機器の修理、設備の設置等)業務を担当します。この分類となる従業員の場合、技術的な研修歴についても記録する必要があります。

GeneXus でこの分類をどのように実装すればよいでしょうか？

実装し、結果を確認してください。

サブタイプのその他の使用例

従業員とその上司の情報を記録するトランザクションが必要となりました。上司も一人の従業員として記録する必要があり、どのように実装できるでしょうか？

「Employee」トランザクションでは、役職を持たない従業員を上司として割り当てた場合、エラーを表示する必要があります。また、表示している従業員と、選択した上司が同一人物である場合、保存させない処理が必要です。

どのようにこの実装を行うためのオブジェクトを定義し、ルールを記述すればよいでしょうか？

以下のヒントをもとにルールの実装を行い、結果を確認してください。

Error("従業員と上司が同じです。")

If _____ = _____;

Error("選択した従業員は、役職者ではありません。")

if Not _____ and Not _____;

式(対象教材動画番号:10~13)

「式」対「ルール」

顧客が保有する賞品と交換できるポイント残数の管理に再びフォーカスします。

Add ルールと Subtract ルールによって実装された処理をグローバル式に置き換えて実装は可能でしょうか？

同じ操作が実装可能な場合、実装方法を検討してください。

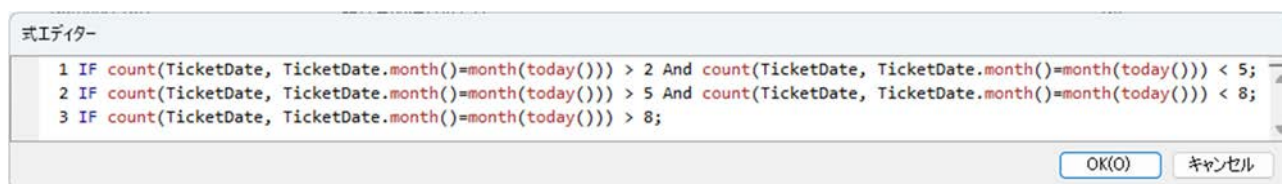
Horizontal 式(オプション)

テストを進める前に、新しい要件がありました：

1ヶ月のうちに、顧客が購入したチケットの枚数に応じて、フリーパスを取得します。
条件と獲得枚数は、以下の通りです。

- 1ヶ月のうちに 2 枚より多く 5 枚未満の購入がある場合、1 枚取得
- 1ヶ月のうちに 5 枚より多く 8 枚未満の購入がある場合、2 枚取得
- 1ヶ月のうちに 8 枚より多い購入がある場合、3 枚取得

「Customer」トランザクションの第 1 レベルに追加するフリーパス(CustomerFreePasses)の項目属性で、下図のグローバル式を定義することでこの要望を満たすことができます。



別の選択肢として、対応する値を返す「FreePasses」プロシーチャーを定義し、呼び出す方法が考えられるため、検討し、実装してください。

このプロシーチャーオブジェクトでは、Parm ルールはどのように定義するのでしょうか？

また、CustomerFreePasses 項目属性がプロシーチャーオブジェクトから返された値を受け取るため、どのように呼び出し、取得する実装ができるのでしょうか？

Aggregate 式

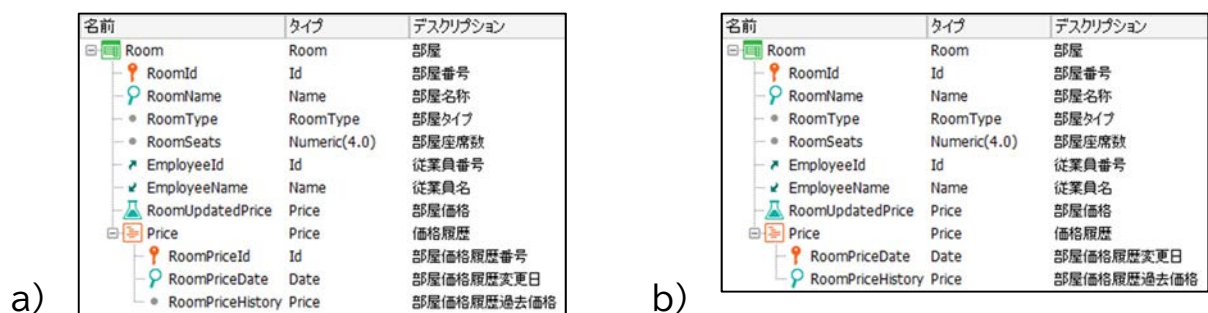
部屋による価格変動

これまで、チケット価格は購入時に決定されていましたが、選択された上映部屋に基づき価格が決定されるように変更します。

そのため、上映部屋ごとに価格を変更した日と、その価格が履歴として記録される必要があります。(※同一の変更日を記録することはできません。)

チケットの購入時に利用される価格は、最新の日付のものを利用する必要があります。

上記の新たな要件に基づき、モデル化した最適な選択肢はどちらでしょうか？



モデルを確定後、RoomUpdatedPrice に、最新の日付に基づく価格の値を取得するためのグローバル式を検討し、実装してください。

「Room」トランザクションの更新後、「Ticket」トランザクションに修正は必要でしょうか？
チケットの合計金額は有効な計算ができているでしょうか？

必要な変更を行い、実行結果を確認してください。

プレミアム映画の価格割り増し

「Movie」トランザクションにプレミアム映画であるかを記録する項目(MovieIsPremium)を追加します。

映画がプレミアムの場合、チケット購入時に価格を「7%」値上げし、そうでない場合には、通常価格を適用します。

必要な変更を行い、実行結果を確認してください。

チケット販売数の管理

スケジュールにおける販売済みのチケット数を確認できるように項目(ScheduleSelledSeats)を追加します。

また、チケットが売り切れているスケジュールに対する新規購入が開始された場合、エラーを表示し、登録できないように制御を追加する必要があります。

必要な変更を行い、実行結果を確認してください。

ダイナミックトランザクション(対象教材動画番号:14)

映画館システムの追加要件として、売店で扱う軽食を管理できるようにする必要があります。そのため、「CandyStoreProduct」トランザクションを作成し、商品の管理番号、名前、画像、価格を記録できるようにします。

商品の画像は、本資料同梱のものを利用してください。

また、映画と、軽食の情報を統合し、1つの画面を利用し、登録されたデータを確認する必要があります。

さらに、この状態でレポートとして出力できる必要があり、表示内容は、映画または軽食の「管理番号」、「名前」、「画像」、「種別」(映画か軽食)を表示します。

新たにダイナミックトランザクションとして「AllProduct」トランザクションを作成し、映画(Movie)と売店の商品(CandyStoreProduct)が記録されたトランザクション両方から情報を取得するように実装し、このトランザクションオブジェクトオブジェクトに基づき、PDF 出力の実装を行ってください。

出力例:

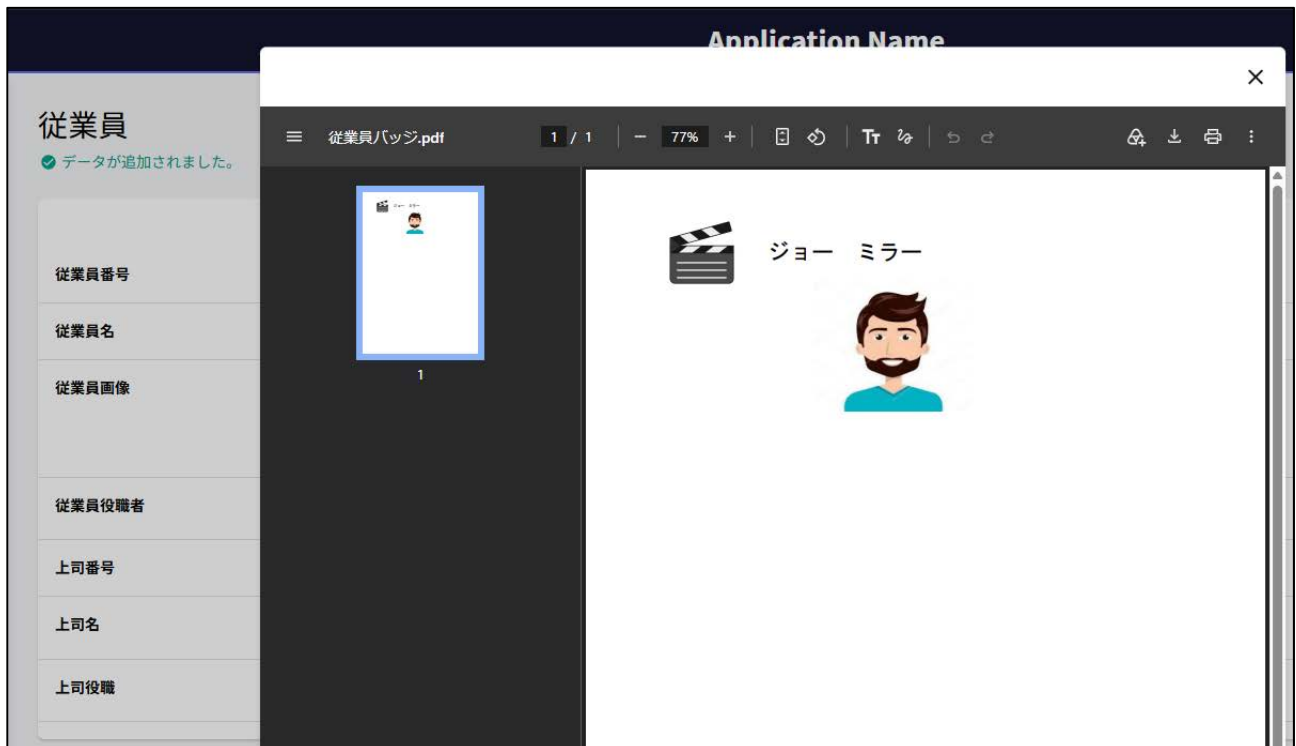
 商品一覧			
番号	種別	名前	画像
1	売店の商品	ポップコーン	
1	映画	顔撞の名声	
2	売店の商品	炭酸飲料	
2	映画	過去の木材	
3	売店の商品	グミ	
3	映画	失われたパレットの記憶	
4	売店の商品	ホットドッグ	

また、「AllProduct」トランザクションにおいて、「映画」や「売店の商品」の新規登録・更新・削除を行えるようにしたいという追加要望がありました。

この要望をかなえるため、必要な変更を追加してください。

トランザクションにおけるイベント(対象教材動画番号:15)

システムには、新規従業員を登録した場合、登録された従業員の名前と画像を含んだバッジを作成するための PDF がポップアップウィンドウで表示される必要がありました。
下図を参考に PDF を出力するためのオブジェクトと、その呼び出し処理を実装してください。



FOR EACH コマンドの詳細(対象教材動画番号:17~18)

下図の見栄えで、カテゴリーごとに映画を表示し、各映画に対し、出演者を一覧で表示します。カテゴリーに関連付けられた映画がない場合には、該当映画がないことを示すメッセージを表示します。

この要望を満たす実装を行ってください。



映画館システム

カテゴリ: コメディ

映画名称: 誤植の名声



映画公開年: 2019

制作国: アメリカ

映画監督名: クララ・フォン・マイヤー

出演者名: レオナルド・デカプリオ



出演者名: ブラッド・パット



映画名称: 図書館の夜明け前



映画公開年: 2012

制作国: アメリカ

映画監督名: マテオ・パルガス

出演者名: ティム・ハンクス



出演者名: レオナルド・デカプリオ



映画名称: ステラ・エコー

映画公開年: 1939

制作国: ウルグアイ

映画監督名: アーサー・ペンデルトン

カテゴリ: ホラー

該当映画なし

もし、同じ見栄えで、カテゴリーに関連付けられた映画がない場合、そのカテゴリーは表示したくないという要望があった場合、どのように実現できるでしょうか？

以下の枠内へ必要な変更を加えた Source に記載するコードを記述してください。



カテゴリーごとに映画を表示し、関連付けられた映画がない場合、メッセージを表示する実装を行った最初のプロシージャオブジェクトを「名前を付けて保存」します。

Parm ルールを定義し、呼び出し元から特定のカテゴリー (CategoryId 等) を受け取れるようにし、指定されたカテゴリーに関する映画と、その出演者のみを表示するようにします。

もし、カテゴリーが指定されずに呼び出された場合には、すべてのカテゴリーの情報を表示する必要があります。

この修正を行い、カテゴリーを指定し、プロシージャオブジェクトを呼び出すための Web パネルオブジェクトを作成し、実行結果を確認してください。

サブルーチン(対象教材動画番号:19)

「Movie」トランザクションで、映画の年齢制限区分(G、PG、PG-13、R)を記録するための項目属性(MovieRating)を追加してほしいという要望がありました。
この項目属性には、下図の Rating という列挙型ドメインを定義し、データタイプとして指定してください。

名前	DESCRIPTION	値
G	G	G
PG13	PG-13	PG-13
PG	PG	PG
R	R	R

Buttons: 追加(A), 削除(R), 編集(E), 上位へ移動(U), 下位へ移動(D), OK(O), キャンセル

列挙型ドメインの定義について

PG-13 を定義する場合、「名前」にハイフンやその他の特殊文字を利用できないため、ハイフンなしで入力します。

加えて、映画を選択し、クリックしたボタンによって年齢区分を設定する下図のようなレイアウトの Web パネルの作成も要望されています。

映画の更新を行う同一のコードが各ボタンのイベントに必要となるため、サブルーチンを活用し、繰り返し同じコードを記述せずに実装します。

各ボタンに対するイベントのうち、一例を下図に示しています。

```

1 Event 'G'
2     &Rating = Rating.G
3     Do "UpdateRating"
4 Endevent

```

Unique 節(対象教材動画番号:20)

スケジュールに登録がある映画についてレポート出力してほしいという要望がありました。このリストには、映画は重複せずに表示し、各映画がいくつのスケジュールに登録されているか表示する必要があります。



映画館システム

映画

スケジュール数

誤植の名声

3



過去の木材

2



失われたパレットの記憶

1



ザ・コンフリクト：ロスト・コード

1



図書館の夜明け前

1



スケジュールに登録がある映画については、アプリケーション上でも表示し、その画面からポップアップでレポート出力が行えるようにしてほしいという要望がありました。
この要望を満たすための Web パネルオブジェクトを作成し、実行結果を確認してください。

スケジュールされた映画

映画名称	スケジュール数
誤植の名声	3
過去の木材	2
失われたパレットの記憶	1
ザ・コンフリクト：ロスト・コード	1
図書館の夜明け前	1

レポート出力

1 / 1

77%

+

🔍

🔄

Tt

🔗

📄

📥

🖨

⋮

🎬

映画館システム

映画	スケジュール数
誤植の名声	3
過去の木材	2
失われたパレットの記憶	1
ザ・コンフリクト：ロスト・コード	1
図書館の夜明け前	1

データセレクトター(対象教材動画番号:21)

日付の範囲を入力し、その範囲内でスケジュール登録のある映画のみをレポートとしてポップアップ出力したいという要望があるため、日付の範囲を入力する Web パネルオブジェクトと、レポート出力を行うプロシーチャーオブジェクトを作成し、実行結果を確認してください。この時、日付の範囲でスケジュールを検索する実装は、本システム内で今後も頻繁に利用する条件であるという情報もあったため、今後の作業効率化のため、データセレクトターも作成します。



このプロシーチャーオブジェクトのナビゲーションの説明を以下に記載してください。

同じ画面から別のレポートを先ほどと同様にポップアップで出力したいという要望がありました。新しいレポートは、指定された日付の範囲内に含まれる映画についてより詳細な情報(上映時間、上映部屋、スケジュール日付)も含む一覧です。必要な実装を行い、実行結果を確認してください。

The screenshot shows a web application interface for a cinema system. On the left, there's a sidebar with filters for '範囲 (開始)' (26/09/01) and '範囲 (終了)' (26/09/30). The main area displays a report titled '指定範囲内映画詳細.pdf'. The report header includes a clapperboard icon and the title '映画館システム' with the date range '26/09/01 ~ 26/09/30'. Below the header, a table lists movies and their showtimes:

映画	上映日	上映時間	上映部屋
誤植の名声	26/09/16	11	ルーム 1
失われたバレットの記憶	26/09/23	11	ルーム 1
過去の木材	26/09/10	14	ルーム 2
誤植の名声	26/09/16	14	ルーム 2

ここで新たに作成したプロシージャオブジェクトと、直前に作成した一つ目のプロシージャオブジェクトを比較し、ナビゲーションの違いについて以下に説明を記載してください。

同じ画面から呼び出すレポートとして、最後に日付の範囲を入力し、その範囲内でスケジュール登録のある重複のない映画一覧をポップアップで出力したいという要望がありました。各映画に対し、指定された期間内にスケジュールされた回数のみを集計する必要があります。必要な実装を行い、実行結果を確認してください。

The screenshot shows a web application interface for a cinema system. On the left, there is a sidebar with navigation links: '映画リスト' (Movie List), '映画 (詳細) リスト' (Movie (Details) List), and '映画リスト (一意)' (Movie List (Unique)). The main area displays a report titled '映画館システム' (Cinema System) for the period '26/09/01 ~ 26/09/30'. The report is a table with two columns: '映画' (Movie) and 'スケジュール数' (Schedule Count). The table lists three movies with their respective schedule counts.

映画	スケジュール数
誤植の名声	2
過去の木材	1
失われたパレットの記憶	1

ここで新たに作成したプロシージャオブジェクトと、ここまで作成した 2 つのプロシージャオブジェクトを比較し、ナビゲーションの違いについて以下に説明を記載してください。

ビジネスコンポーネントの利用(対象教材動画番号:23~26)

ビジネスコンポーネントを利用し、データベースに対する操作を実施します。

担当従業員の更新

新たな要望として、上映部屋を担当する従業員を一括で変更する機能が必要となりました。この変更には、部屋のタイプと、従業員をそれぞれ選択するコンボボックスを用意し、ボタンをクリックしたことで、選択された「部屋タイプ」と一致する上映部屋すべての担当従業員を選択した「従業員」に変更する処理を呼び出すボタンを配置します。(下図イメージの通り)必要な実装を行い、実行結果を確認してください。

映画の出演者に対する操作

新たな要望として、以下の 2 つの機能を保有する画面が必要となりました。

- 選択した出演者が主演か、そうでないかを切り替える
- 選択した出演者を映画から削除する

この画面には、映画を選択できるコンボボックスと、その映画の出演者のみが表示されるコンボボックス、上記 2 つの機能を実行するためのボタンを配置します。(下図イメージの通り)また、処理の結果を表示してほしいという要望もあるため、Messages タイプの変数利用も必要です。

必要な実装を行い、実行結果を確認してください。

ヒント:

選択した映画の出演者のみを表示するためには、ダイナミックコンボボックスの Conditions プロパティで、選択した映画を条件に指定します。

プロシージャオブジェクトの利用(対象教材動画番号:27~19)

プロシージャオブジェクトを利用し、データベースに対する操作を実施します。

担当従業員の更新

ビジネスコンポーネントを利用し、上映部屋を担当する従業員を一括で変更する実装を行った画面にボタンを追加し、同様の処理をプロシージャオブジェクトで実行するように追加を行います。

必要な実装を行い、実行結果を確認してください。

映画の削除

新たな要望として、映画の出演者に対する操作を実装した画面に選択した映画を削除するボタンの追加が必要となりました。

このボタンで実行する処理は、プロシージャオブジェクトを利用して実装してください。

必要な実装を行い、削除対象とする映画を新規登録したのち、実行結果を確認してください。

もしも、他のテーブル(例:スケジュール)から参照されている映画を削除対象とした場合、どのような結果になるでしょうか？

以下に説明を記載してください。

ユーザー向け画面: Web パネルオブジェクト(対象教材動画番号:33~40)

続いて、映画館システムにおいて対話型画面の要望を実装していきます。

スケジュール一覧表示画面

新たな要望として、**上映日順**に並べられたスケジュール一覧画面が必要となりました。
この画面では、以下の項目でフィルタリングできる必要があります。

- 上映日(既定値は「今日の日付」とし、指定日付以降の日付で表示)
- 上映時刻
- 部屋タイプ
- 映画名称(前方一致)

フィルタ項目がすべて未入力の場合、すべてのスケジュールを表示します。

さらに、一覧の各行には、スケジュールされた映画の詳細表示画面へ遷移するための画像を追加する必要があり、詳細画面では、一覧画面へ戻るためのボタンの実装も必要です。
必要な実装を行い、実行結果を確認してください。

上映日 26/09/01  上映時刻 午後2時  部屋タイプ 通常  映画名称 <table border="1"> <thead> <tr> <th>部屋タイプ</th> <th>上映日</th> <th>上映時刻</th> <th>映画名称</th> <th>映画ポスター</th> <th>詳細へ</th> </tr> </thead> <tbody> <tr> <td>通常</td> <td>26/09/10</td> <td>午後2時</td> <td>過去の木材</td> <td></td> <td></td> </tr> <tr> <td>通常</td> <td>26/09/16</td> <td>午後2時</td> <td>誤植の名声</td> <td></td> <td></td> </tr> </tbody> </table>	部屋タイプ	上映日	上映時刻	映画名称	映画ポスター	詳細へ	通常	26/09/10	午後2時	過去の木材			通常	26/09/16	午後2時	誤植の名声			<table border="1"> <tr> <td>映画名称</td> <td>誤植の名声</td> <td>戻る</td> </tr> <tr> <td>映画公開年</td> <td>2019</td> <td></td> </tr> <tr> <td>映画説明</td> <td colspan="2">世界中が熱狂するトップスターのエージェントが、ある新作映画の出演契約書で二人の名前を誤記してしまったことから、すべてが狂い出す。契約により、彼らは自分たちのイメージとは正反対の、中西部の配管工とフロリダのサーファーとしての生活を強いられる。下水道で悪戦苦闘するレオ、厳しい波にのまれるブラッド。二人は名声と「正しいスベル」を取り戻すため、スベルミスに犯した映画製作陣への復讐を決意する。名匠クララ・フォン・マイヤーが、ハリウッドの虚飾を爆笑と風刺で暴き出す、洗練されたバディ・コメディー！</td> </tr> <tr> <td>カテゴリー名</td> <td>コメディ</td> <td></td> </tr> <tr> <td>国名</td> <td>アメリカ</td> <td></td> </tr> <tr> <td>映画監督名</td> <td>クララ・フォン・マイヤー</td> <td></td> </tr> </table>	映画名称	誤植の名声	戻る	映画公開年	2019		映画説明	世界中が熱狂するトップスターのエージェントが、ある新作映画の出演契約書で二人の名前を誤記してしまったことから、すべてが狂い出す。契約により、彼らは自分たちのイメージとは正反対の、中西部の配管工とフロリダのサーファーとしての生活を強いられる。下水道で悪戦苦闘するレオ、厳しい波にのまれるブラッド。二人は名声と「正しいスベル」を取り戻すため、スベルミスに犯した映画製作陣への復讐を決意する。名匠クララ・フォン・マイヤーが、ハリウッドの虚飾を爆笑と風刺で暴き出す、洗練されたバディ・コメディー！		カテゴリー名	コメディ		国名	アメリカ		映画監督名	クララ・フォン・マイヤー	
部屋タイプ	上映日	上映時刻	映画名称	映画ポスター	詳細へ																																
通常	26/09/10	午後2時	過去の木材																																		
通常	26/09/16	午後2時	誤植の名声																																		
映画名称	誤植の名声	戻る																																			
映画公開年	2019																																				
映画説明	世界中が熱狂するトップスターのエージェントが、ある新作映画の出演契約書で二人の名前を誤記してしまったことから、すべてが狂い出す。契約により、彼らは自分たちのイメージとは正反対の、中西部の配管工とフロリダのサーファーとしての生活を強いられる。下水道で悪戦苦闘するレオ、厳しい波にのまれるブラッド。二人は名声と「正しいスベル」を取り戻すため、スベルミスに犯した映画製作陣への復讐を決意する。名匠クララ・フォン・マイヤーが、ハリウッドの虚飾を爆笑と風刺で暴き出す、洗練されたバディ・コメディー！																																				
カテゴリー名	コメディ																																				
国名	アメリカ																																				
映画監督名	クララ・フォン・マイヤー																																				

複数グリッド

カテゴリと、各カテゴリに関連付けられた映画を表示できる画面が要望されています。2つの異なる表示方法が求められているため、それぞれ画面を実装します。

並行グリッドの利用

次の2つの一覧を表示する画面が必要という要望がありました。

1. カテゴリ名と、そのカテゴリで登録された映画数の一覧
2. 上記一覧「1」で行選択し、そのカテゴリで登録された映画の一覧

カテゴリの一覧には、登録されているすべてのカテゴリを表示します。必要な実装を行い、実行結果を確認してください。

カテゴリ名	映画数	映画名称	映画ポスター	映画公開年
コメディ	3	過去の木材		2019
ホラー	0			
ドラマ	2	失われたパレットの記憶		2019
アニメーション	1			
アクション	1			
ファンタジー	1			

実行結果の確認後、関連付けられた映画のないカテゴリは表示したくないという要望がありました。

また、その場合、カテゴリ名が重複して表示されることもないようにしたいです。

この場合、どのように実装を行う必要があるでしょうか？

必要な実装を行い、実行結果を確認してください。

ネストされたグリッドの利用

対象映画のあるカテゴリーのみを表示し、各カテゴリーで登録された映画を一度に表示する画面が求められています。

また、表示された映画には、映画の詳細表示画面へ遷移するための画像を追加する必要があります。詳細画面では、知覧画面へ戻るためのボタンの実装も必要です。

この映画の詳細画面は、これまでに作成済みの画面を再利用可能です。

必要な実装を行い、実行結果を確認してください。

コメディ					ドラマ				
映画番号	映画名称	映画公開年	映画ポスター	Image Button	映画番号	映画名称	映画公開年	映画ポスター	Image Button
1	誤植の名声	2019		>	2	過去の木材	2019		>
5	図書館の夜明け前	2012		>	3	失われたバレットの記憶	2019		>
9	ステラ・エコー	1939		>					
アニメーション					アクション				
映画番号	映画名称	映画公開年	映画ポスター	Image Button	映画番号	映画名称	映画公開年	映画ポスター	Image Button
6	アンキオ：記憶の都市	2012		>	4	ザ・コンフリクト：ロスト・コード	2019		>
ファンタジー									
映画番号	映画名称	映画公開年	映画ポスター	Image Button					
7	記憶のカンヴァスと時の番人	2014		>					

ヒント: フリースタイルグリッドの Columns プロパティで、1 行あたり何件表示するか指定可能です

ベーステーブルなしのグリッドの利用

映画がスケジュールされた日付と、その日に予定されている映画の名前をすべて列記した一覧(「:」で映画名区切り)が必要という要望がありました。

画面は下図のような実装を求められています。

必要な実装を行い、実行結果を確認してください。

上映日	上映映画
26/09/10	:過去の木材
26/09/16	:誤植の名声:誤植の名声
26/09/23	:失われたバレットの記憶
26/10/02	:図書館の夜明け前:誤植の名声
26/10/03	:ザ・コンフリクト：ロスト・コード
26/10/06	:過去の木材

顧客向けに重点を置いたアプリケーション(対象教材動画番号:42~44)

本単元の課題を実行する場合、[Angular ジェネレーターの要件](#)を満たしている必要があります。

映画館システムにおいて、利用者向けに画面の実装が必要という要望がありました。この要望を満たすためには、Panel オブジェクトを利用した画面を Angular ジェネレーターで生成する必要があります。

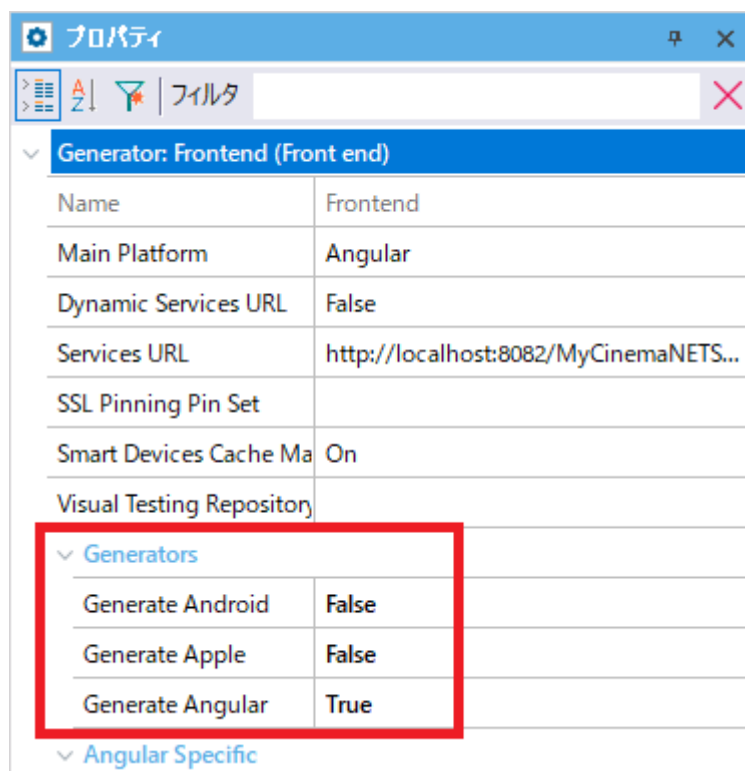
この画面では、グリッド形式で、上映日、上映時刻、上映部屋、映画名称、映画ポスターを一覧表示します。

一覧を自動で拡張できるように、Auto Grow プロパティを True に設定しておきます。必要な実装を行い、実行結果を確認してください。

操作補足:

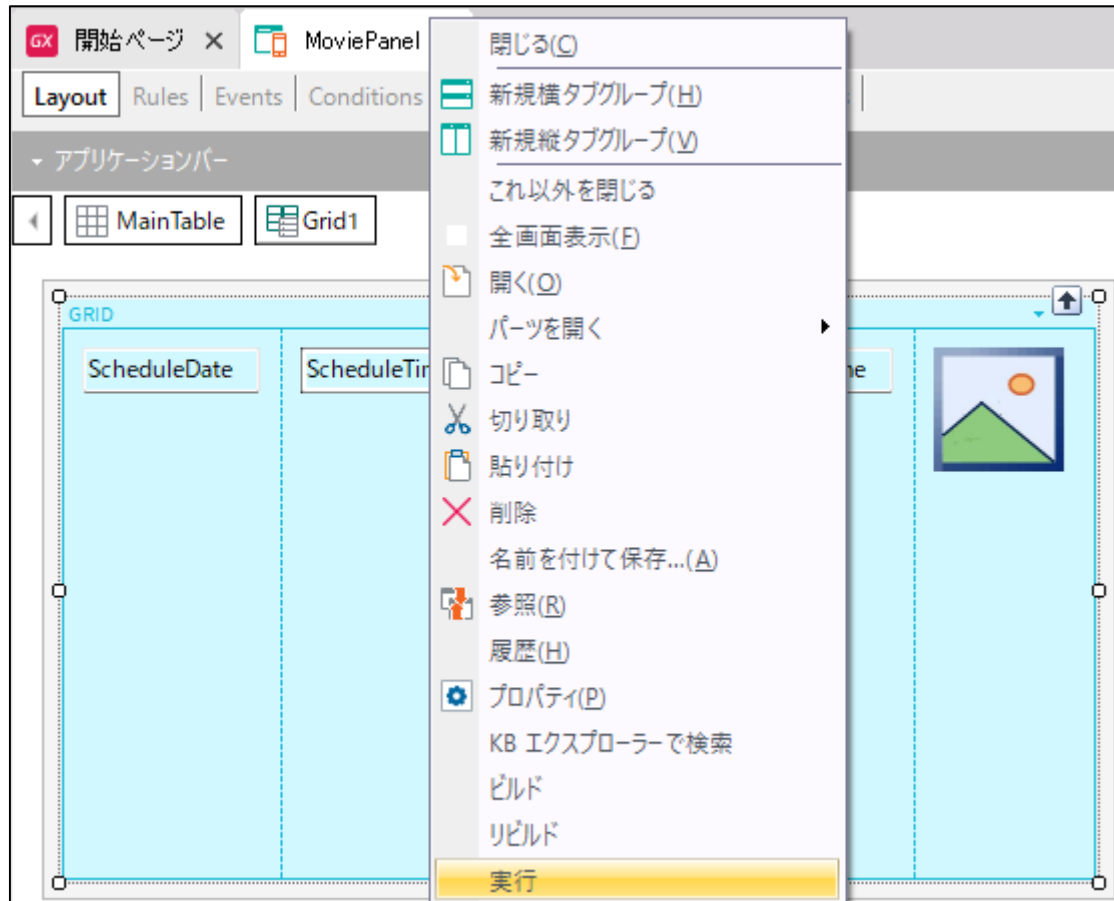
Angular ジェネレーターで Panel オブジェクトを生成するための操作について補足します。Panel オブジェクトを保存後、メニューバーの「表示」→「その他のツールウィンドウ」→「設定」をクリックし、表示された設定ウィンドウ内の「フロントエンド」ノードを右クリックし、「プロパティ」をクリックします。

表示されたプロパティウィンドウにおいて、Generate Android プロパティおよび Generate Apple プロパティを False に変更し、Generate Angular プロパティを True に変更します。(下図参照)



次ページへ続く

Panel オブジェクトの実行結果を確認するため、作成した Panel オブジェクトの Main program プロパティを True に変更し、保存します。
オブジェクトを右クリックし、「実行」オプションをクリックします。



下図のような結果が確認できることを確認します。

2026/9/17	午前11時	ルーム 1	誤植の名声	
2026/9/24	午前11時	ルーム 1	失われたパレットの記憶	
2026/10/4	午後8時	ルーム 1	ザ・コンフリクト：ロスト・コード	
2026/10/7	午前11時	ルーム 1	過去の木材	
2026/9/11	午後2時	ルーム 2	過去の木材	
2026/9/17	午後2時	ルーム 2	誤植の名声	
2026/10/3	午後2時	ルーム 3	図書館の夜明け前	
2026/10/3	午後5時	ルーム 3	誤植の名声	

ヒント: 日付が表示されない場合、クラスを attribute-date への変更を試してください。

実行結果を確認した結果、追加の要望がありました。
一覧に各スケジュールに対し、販売されたチケット数を表示する列が必要となりました。
必要な実装を行い、実行結果を確認してください。

※該当情報を格納する項目属性が定義済みですが、本課題では、下図のように変数を配置し、Events エlementで必要な実装を定義します。

GRID					
ScheduleDate	ScheduleTime ▾	RoomName	MovieName		&TicketQty

Web サービス(対象教材動画番号:46~47)

GeneXus において、Web サービス(REST)の公開と、Web サービスの利用について実際に実装を行い、確認します。

例題として、映画名称を受け取ると、その映画の情報を出力する Web サービスを実装します。

「Movie」トランザクションと同じ構造を持つコレクション型 SDT オブジェクトを作成します。作成する際、SDT オブジェクトの Structure エlement で、手動で構造を定義することもできますが、「Movie」トランザクションをドラッグアンドドロップすることで容易に定義を進められます。

この時、出演者のリストは不要なため、Actor レベルに関する構造は定義しない(または削除)します。

続いて、実際に Web サービスとして公開する Data Provider オブジェクトを作成し、直前に作成した SDT オブジェクトを Source Element へドラッグアンドドロップします。

割り当てる値は、SDT オブジェクト内で定義されたメンバー名と一致する名前の項目属性となる場合、イコールを含む右辺は記載不要です。

例題における要件となる受け取った映画名称によるフィルタリングが行えるように条件を記載します。

もし、映画名称を受け取らない場合、全件が表示されるように条件は定義します。

また、Expose as Web Service プロパティを True に設定し、新たに表示される Web Service Protocol プロパティを REST Protocol に変更します。

想定される実装は、下図の通りです。

The screenshot displays the GeneXus IDE interface. On the left, the 'Source' tab shows the structure of the SDT object 'SDTMoviesItem'. The structure is defined as follows:

```
1 SDTMovies from Movie
2 Where MovieName Like &MovieName when not &MovieName.IsEmpty()
3 {
4   SDTMoviesItem
5   {
6     MovieId
7     MovieName
8     MoviePoster
9     MovieYear
10    MovieDescription
11    CountryId
12    CountryName
13    CategoryId
14    CategoryName
15    FilmDirectorId
16    FilmDirectorName
17    MovieStatus
18    MovieRating
19    MovieActorsQty
20  }
21 }
22
```

On the right, the 'プロパティ' (Properties) tab shows the configuration for the Data Provider 'DPMovie'. The properties are as follows:

Data Provider: DPMovie	
Name	DPMovie
Description	DPMovie
Expose as Web Service	True
Web Service Protocol	REST Protocol
Generate OpenAPI interf	Use Environment property value
Exposed namespace	MyCinema
Main program	False
Call protocol	Internal
Module/Folder	Root Module
Qualified Name	DPMovie
Object Visibility	KnowledgeBase
Output	
Infer Structure	No
Output	SDTMovies

呼び出し元から値を受け取るため、Rules エlementで以下のルールを定義します。

```
Parm(IN:&MovieName);
```

Data Provider オブジェクトを保存し、「すべてリビルド」を実行します。

F5 キーを押下し、アプリケーションを起動し、ブラウザから下記 URL へアクセスします。

(不等号で囲まれた範囲は、ご自身の環境にそらえて書き換える必要があります。)

http://<server>/<BaseURL>/rest/<DataProviderName>

置き換え結果例: <http://localhost:8082/MyCinemaNETSQLServer/rest/DPMovie>

その結果、下記のような結果が表示されます:



Web サービスの公開ができたため、この Web サービスを利用する画面を実装します。

HttpClient データタイプ変数を利用し、この REST サービスを利用します。

実際に利用する場合、外部の Web サービスや、他の KB で実装された Web サービスを利用しますが、この課題では、同一 KB 内で定義された Web サービスの利用で結果を確認します。

新規 Web パネルオブジェクトを作成し、下記の通り各Elementの定義を行います。

Variables Element

名前	タイプ
HttpClient	HttpClient
MovieName	VarChar(50)
SDTMovies	Web サービス作成時に作成した SDT
StringParameter	VarChar(200)

Web Layout エlement

<ErrorViewer: ErrorViewer1>

映画名称

GRID

映画番号	映画名称	映画ポスター	映画公開年	映画説明
&SDTMovies.item(0).MovieId	&SDTMovies.item(0).MovieName		&SDTMovies.item(0).MovieYear	&SDTMov

- ・「映画情報取得」ボタンには、「GetMovieInfo」イベントを紐づけ
- ・SDTMovies 変数は、標準グリッドとして全メンバーを配置
(上図では、右方向へのスクロール範囲を省略)
- ・グリッドに配置された列はすべて Read Only プロパティを True に変更

Events Element

Event 'GetMovieInfo'

```

    &HttpClient.Host = <サーバーホスト名> //例: "localhost"
    &HttpClient.Port = <サーバーポート番号> //例 (.NETの既定値) : 8082
    &HttpClient.BaseUrl = "/" + <ベースURL> + "/rest" //例: "/MyCinemaNETSQLServer/rest/"
    &HttpClient.Secure = 0
    &StringParameter = <DPオブジェクト名> + "?" + "MovieName=" + &MovieName //例: "DPMovie?"
    &HttpClient.Execute("GET", &StringParameter)

    If &HttpClient.StatusCode = 200
        Msg("取得成功")
        &SDTMovies.FromJson(&HttpClient.ToString())
    Else
        Msg("エラー: " + &HttpClient.StatusCode.ToString())
    Endif

```

Endevent

- ・各不等号で囲まれた範囲については、Web サービス実行時に利用した URL より必要な文字列を取得し、指定してください。

アプリケーションを実行し、Web サービスを利用できることを確認します。
映画名称に入力がある場合と、ない場合どちらでも結果を取得できることも確認します。

▲ 取得成功

映画名称

映画情報取得

映画番号	映画名称	映画ポスター	映画公開年	映画説明	国番号	国名	カテゴリー番号	カテゴリー名	映画監督番号	映画監督名	映画ステータス	映画年齢区分	映画出演者数
1	訓練の名声		2019	世界中が熱狂するトップスターのエージェントが、ある新作映画の出演契約書で二人の名前を誤記してしまったことから、すべてが狂い出す。契約により、彼らは自分たちのイメージとは正反対の、中西部の配管工とフロリダのサーファーとしての生活を強いられる。下水道で悪戦苦闘するレオ、美しい波にのまれるブラッド。二人は名声と「正しいスベール」を取り戻すため、スベールミスを行った映画製作陣への復讐を決定する。名匠クララ・フォン・マイヤーが、ハリウッドの虚飾を爆笑と風刺で暴き出す、洗練されたパディ・コメディ！	2	アメリカ	1	コメディ	6	クララ・フォン・マイヤー	保留	G	2

Application Name

▲ 取得成功

映画名称

映画情報取得

映画番号	映画名称	映画ポスター	映画公開年	映画説明	国番号	国名	カテゴリー番号	カテゴリー名	映画監督番号	映画監督名	映画ステータス	映画年齢区分	映画出演者数
1	訓練の名声		2019	世界中が熱狂するトップスターのエージェントが、ある新作映画の出演契約書で二人の名前を誤記してしまったことから、すべてが狂い出す。契約により、彼らは自分たちのイメージとは正反対の、中西部の配管工とフロリダのサーファーとしての生活を強いられる。下水道で悪戦苦闘するレオ、美しい波にのまれるブラッド。二人は名声と「正しいスベール」を取り戻すため、スベールミスを行った映画製作陣への復讐を決定する。名匠クララ・フォン・マイヤーが、ハリウッドの虚飾を爆笑と風刺で暴き出す、洗練されたパディ・コメディ！	2	アメリカ	1	コメディ	6	クララ・フォン・マイヤー	保留	G	2
2	過去の木材		2019	都会で成功を収めた建築家のアダム（ブラッド・パット）は、疎遠になっていた兄のウィリアム（トミー・ビー・ジョーンズ）の訃報を受け、15年ぶりに故郷であるモンタナ州の田舎町に戻る。都会的な服装で、都会の孤憤を抱えるアダムは、自然の中に馴染めない。ウィリアムは地元の小さな木材会社を営んでおり、アダムに会社を譲っていた。しかし、会社は環境保護問題で地元コミュニティと激しく対立していた。アダムは、ウィリアムが隠し続けていた過去の秘密と向き合いながら、会社を売却するか、兄の遺志を継ぐかという決断に迫られる。田舎の静謐な自然と、都会の影が交差する中、アダムは本当の豊かさや家族の絆を再発見する。和解と贖罪のドラマ。	2	アメリカ	3	ドラマ	7	ドミニク・ゾーン			2
3	失われたパレットの記憶		2019	ブエノスアイレスで成功を収めた孤憤な建築家ホアキンは、疎遠だった父親の遺品整理のため、アンデス山脈の小さな村へ、そこで、父親の失われた夢と愛を遺物に隠し持つ博物館人マルティナと出会う。都会の野心的な美術商ルチアーノの介入で、ホアキンは父親の過去と向き合い、都会と自然、過去と現在を融合させた新しい建築プロジェクトを立ち上げる決断をする。家族の絆を再発見する、和解と贖罪の人間ドラマ。	7	アルゼンチン	3	ドラマ	8	イザベラ・ロッシ			3
4	ザ・コンフリクト			都会で成功を収めた洗練された元エージェント、アレックス（ロバート・タウニー・Jr）は、かつての同僚であり、疎遠になっていたタフなサバイバリスト、カイ（クリス・ヘインズワース）の隠された過去を暴く。二人は世界を揺るがす秘密データ「ロスト・コード」を巡り、都会の野心的な組織と激しく対峙することになる。						アーサー・...			