

ルールをトリガーする順序

GeneXus™

これまでトランザクションで記述可能なルールを見てきましたが、このコースでは、いつ実行されるべきか指定する必要がないルールから着手しました。
これまでのルールは、GeneXus がトリガーするタイミングを判断しているため、いつ実行されるべきか指定する必要がありません。

ルール



通常、定義したルールは期待したタイミングで実行されます。しかし、そのタイミングを調整する必要がある場合があります。

例: フライトの座席数

名前	タイプ	デスクリプション	式	Null
Flight	Flight	フライト		
FlightId	Id	フライト番号		No
FlightDepartureAirportId	Id	出発空港番号		No
FlightDepartureAirportName	Name	出発空港名		
FlightDepartureCountryId	Id	出発空港国番号		
FlightDepartureCountryName	Name	出発空港国名		
FlightDepartureCityId	Id	出発空港都市番号		
FlightDepartureCityName	Name	出発空港都市名		
FlightArrivalAirportId	Id	到着空港番号		No
FlightArrivalAirportName	Name	到着空港名		
FlightArrivalCountryId	Id	到着空港国番号		
FlightArrivalCountryName	Name	到着空港国名		
FlightArrivalCityId	Id	到着空港都市番号		
FlightArrivalCityName	Name	到着空港都市名		
FlightPrice	Price	フライト価格		No
FlightDiscountPercentage	Percentage	フライト割引率		No
AirlineId	Id	航空会社番号		Yes
AirlineName	Name	航空会社名		
AirlineDiscountPercentage	Percentage	航空会社割引率		
FlightFinalPrice	Price	フライト最終価格	FlightPrice*(1-AirlineDiscountPerc...	
FlightCapacity	Numeric(4,0)	フライト座席数	count(FlightSeatLocation)	
Seat	Seat	座席		
FlightSeatId	Id	座席番号		No
FlightSeatChar	SeatChar	座席文字		No
FlightSeatLocation	Location	座席位置		No

FlightCapacity >= 8

例を見てください。

ここでは、航空会社が座席数を管理している各フライトは「8 未満の座席数ではないこと」が、求められているとしましょう。新しいフライトが入力されたときに、指定されている条件に従っていない場合は、フライトを保存できないようにします。

これを実現するために、フライトを記録するトランザクションに式項目属性 FlightSeatCapacity を追加し、カウント式として定義するため、次の初期式を割り当てました: Count(FlightSeatLocation)。これにより、この式ではフライトに割り当てられたすべての座席をフィルタ条件なしでカウントすることができます。

例: フライトの座席数

Flight * X

Structure | Web Form | Win Form | Rules * | Events | Variables | Help | Documentation | Patterns

1 Error("座席は8つ以上必要です") if FlightCapacity <8;

Application Name

Recentsフライト

フライト

選択

フライト番号 0 座席は8つ以上必要です X

出発空港番号

出発空港名

出発空港国番号 0

座席を入力する前に
エラーがトリガーされる

次に、[Rules] セクションで、フライトの座席数が 8 未満の場合はフライトを保存できないように、Error ルールを宣言します。

次に、F5 キーを押します。

Flight トランザクションを開いて新しいフライトを作成します。

エラーがトリガーされたことが分かります。なぜでしょうか。新しい行の追加に伴い、FlightCapacity の式の値が変化し、それを利用するルールが、可能な範囲でできるだけ早くトリガーされるためです。この問題は、最初に行を入力するまでの時間が考慮されていなかったため、FlightCapacity 式の結果がゼロ (8 を下回っている) でトリガーされたことにより起こりました。

例: フライトの座席数



次に、エラー条件に `FlightCapacity > 0` を追加して、行を入力する時間を確保できるようにします。

例: フライトの座席数

航空会社割引率	30
フライト最終価格	2100.00
フライト座席数	0

座席

	座席番号	座席文字	座席位置	
×	1	A ▼	窓 ▼	座席は8つ以上必要です
	0	A ▼	窓 ▼	
	0	A ▼	窓 ▼	
	0	A ▼	窓 ▼	
	0	A ▼	窓 ▼	

座席数が 8 未満のため、想定よりも早くエラーがトリガーされています

[行追加]

実行

終了

削除

識別子の値は空のままにしますが、自動的に番号が付けられます。ブラジルのサンパウロにあるグアルーリョス空港から出発し、フランスのパリにあるシャルルドゴール空港に到着するフライトを入力します。フライト料金は 3000 で割引率は 10%、航空会社は TAM です。

ここで座席 1、文字 A、窓際と入力し、行を移動すると、エラーメッセージが表示されます。

すべての座席を入力し終わっていないため、ここでエラーメッセージがトリガーされるべきではありません。座席の入力が 1 つだけの場合、8 を下回っているため、Error ルールがトリガーされるのは当然です。この制御は、ユーザーがすべての座席を入力した後に実行される必要があります。

例：フライトの座席数

Flight * X

Structure * Web Form Win Form Rules Events Variables Help Documentation Patterns

名前	タイプ	デスクリプション	式	Null
Flight	Flight	フライト		
FlightId	Id	フライト番号		No
FlightDepartureAirportId	Id	出発空港番号		No

Flight * X

Structure Web Form Win Form Rules * Events Variables Help Documentation Patterns

```

1 Error("座席は8つ以上必要です")
2   if FlightCapacity >0 and FlightCapacity <8
3     on AfterLevel
4     Level FlightSeatChar;

```

FlightDiscountPercenta...	Percentage	フライト割引率		No
AirlineId	Id	航空会社番号		Yes
AirlineName	Name	航空会社名		
AirlineDiscountPercent...	Percentage	航空会社割引率		
FlightFinalPrice	Price	フライト最終価格	FlightPrice*(1-AirlineDiscountPerce...	
FlightCapacity	Numeric(4,0)	フライト座席数	count(FlightSeatLocation)	
Seat	Seat	座席		
FlightSeatId	Id	座席番号		No
FlightSeatChar	SeatChar	座席文字		No
FlightSeatLocation	Location	座席位置		No

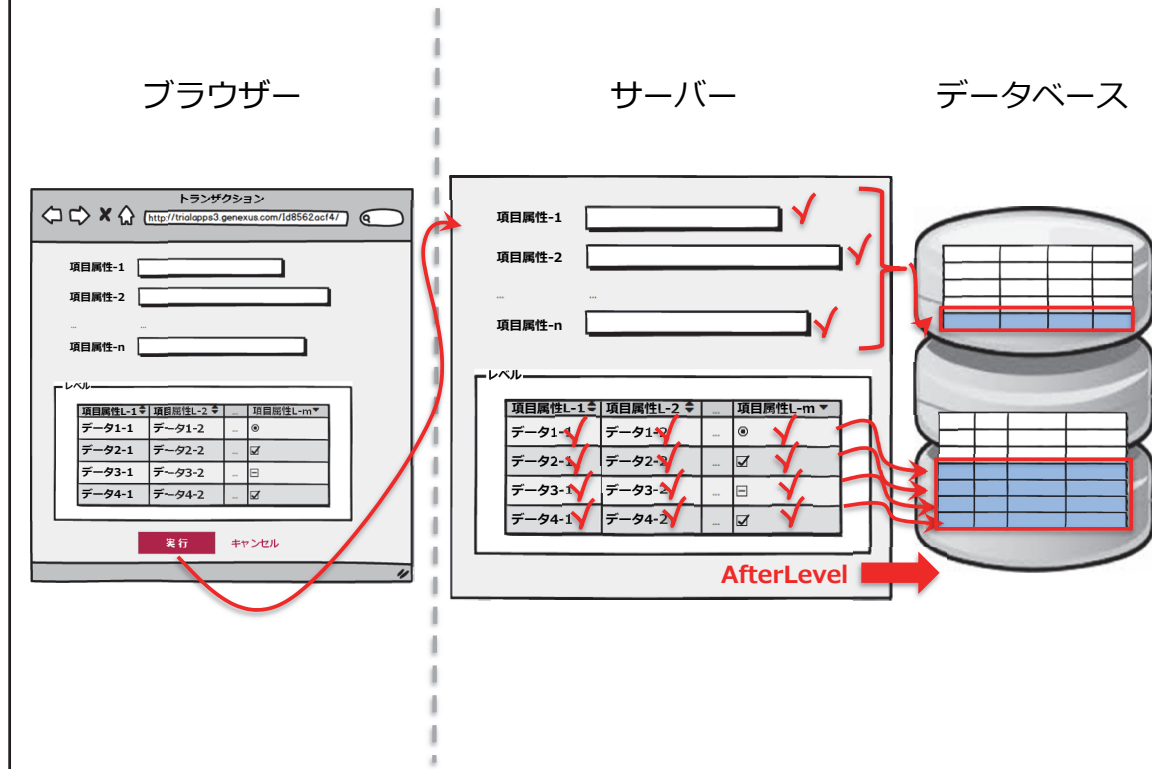
この動作を実現するために、グリッド内の行での作業が完了した後にルールがトリガーされるように条件付けます。そこで、次のように記述します。

On AfterLevel Level FlightSeatChar

「on AfterLevel」のタイミングにより、あるレベルが完了した後にルールがトリガーされるようになります。

この場合は、座席グリッドの行のレベルが完了した後にトリガーする必要があるため、座席の行のレベル内にある項目属性「Level FlightSeatChar」を追加します。FlightSeatLocation など、このレベルのほかの項目属性を使用することもできます。

このようにして、FlightSeatChar 項目属性の場所にデータを入力した後にルールをトリガーする必要があることを、GeneXus に指示します。つまり、このフライトのすべての座席についてヘッダーデータを入力した後です。Error ルールがトリガーされる時には、ユーザーが予定していた座席の入力がすべて行われていて、少なくとも 8 つの座席が入力されていることが確認されるため、ルールを使用した評価には意味があります。



それでは実行してみましょう。4 つの座席を入力して [実行] をクリックし、フライトデータ (座席データを含む) の入力終了したこと、およびフライトをデータベースに保存できることを示します。

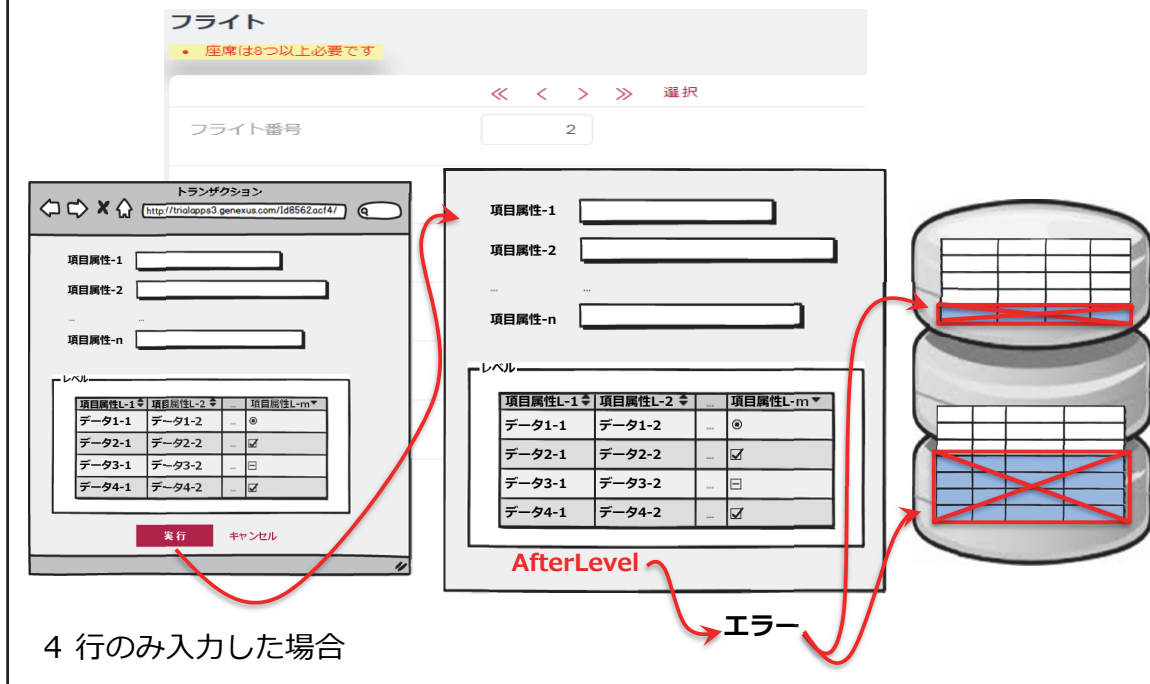
この時点でどのような処理が実行されるでしょうか。ヘッダーと明細行のデータがサーバーに送信され、1 つずつ処理されて、対応するルールがトリガーされます。

すべてのヘッダーデータの検証が終了すると、レコードがテーブルに挿入されます。

各明細行について、同じ処理が繰り返されます。

すべての明細行の処理が完了すると、**AfterLevel** のタイミングになり、そのタイミングに条件付けられたルールがすべてトリガーされます。

ヘッダーデータと明細行データは、既にデータベースに保存されています。



実行時にトランザクションに戻ると、期待どおり、[実行] をクリックした後に GeneXus でエラーが報告されます。これは、入力したのが 4 座席のみであるのが原因です。

そして Error ルールによって保存操作が元に戻されるため、このフライトはデータベースに保存されません。

例: フライトの座席数

フライト座席数 8

座席			
	座席番号	座席文字	座席位置
×	1	A ▼	窓 ▼
×	1	B ▼	通路側 ▼
×	2	A ▼	窓 ▼
×	2	B ▼	通路側 ▼
×	3	A ▼	窓 ▼
×	3	B ▼	通路側 ▼
×	4	A ▼	窓 ▼
×	4	B ▼	通路側 ▼

[行追加]

✓

8つの座席が記録されている

実行 終了

必須の 8 座席を入力して [実行] をクリックし、フライト詳細 (座席データを含む) の入力が完了したこと、およびこのフライトをデータベースに保存できることを示します。AfterLevel の後に、ヘッダーと行が記録されていることが分かります。

GeneXus が初期設定で判断したルールのトリガーするタイミングを遅らせることで、この処理が可能になります。

例: フライトの座席数

フライト

• データが更新されました。

座席なしのフライト

《 < > 》 選択

開始ページ X Flight X

Structure Web Form Win Form Rules Events Variables Help Documentation Patterns

```

1 Error("座席は8つ以上必要です")
2 if FlightCapacity > 0 and FlightCapacity < 8
3 on AfterLevel
4 Level FlightSeatChar;

```

開始ページ X Flight * X

Structure Web Form Win Form Rules Events Variables Help Documentation Patterns

```

1 Error("座席は8つ以上必要です")
2 if FlightCapacity < 8
3 on AfterLevel
4 Level FlightSeatChar;

```

フライト

• 座席は8つ以上必要です

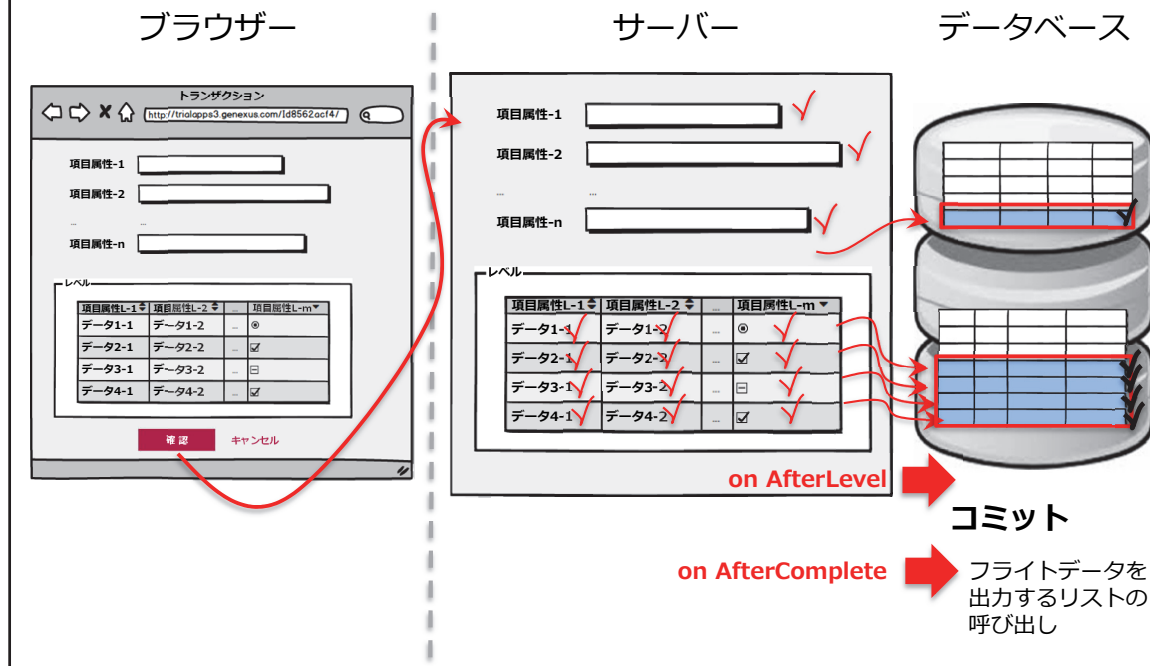
《 < > 》 選択

最後に、座席なしの新しいフライトを入力します。このフライトの保存は許可されます。

なぜでしょうか。これは、AfterLevel を認識せずに入力した、FlightCapacity > 0 という条件があるためです。ここでこの条件を削除し、ルールが正しく記述されるようにします。

F5 キーを押し、座席なしのフライトを編集してから [実行] を再度クリックすると、制御が適用されて入力できなくなります。

AfterLevel と AfterComplete



この例では、ルールをトリガーするタイミングが適切ではない場合、GeneXus にタイミングを指示する必要があることを示しました。

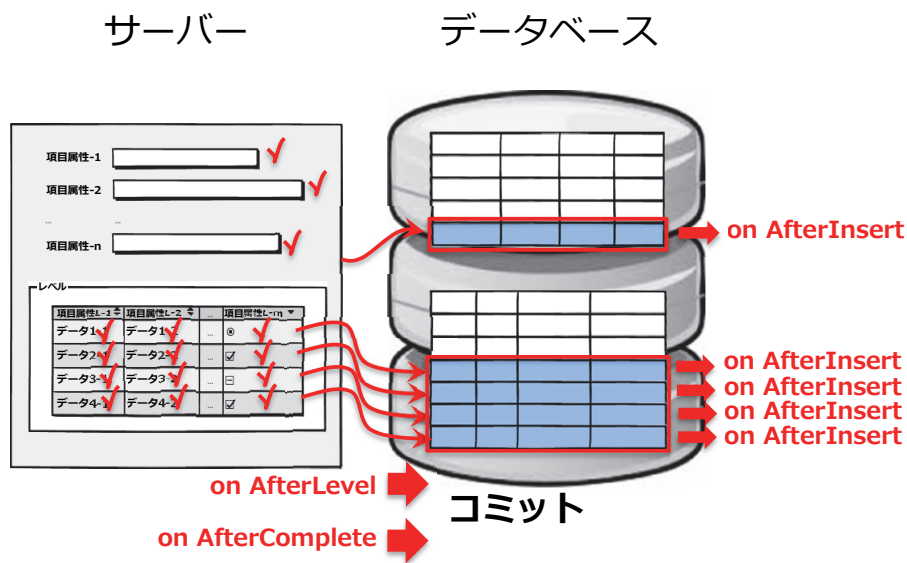
今回の場合、第 1 レベルのあとに、ルールが実行されるように指示するため、"**on AfterLevel**" というタイミングについて説明してきました。

この AfterLevel は、これまで見てきたとおり、データがデータベースに記録されたあとに呼び出されるので、フライトのリスト印刷を呼びだすときなど、便利になります。しかし、AfterLevel は、エラールールにより記録が取り消されることもあります。

リストの場合は、後でデータが削除されないことが確実になったときに呼び出すのが最善の方法です。この処理は、データ挿入が妥当であるときに発行されるコミット後に可能です。

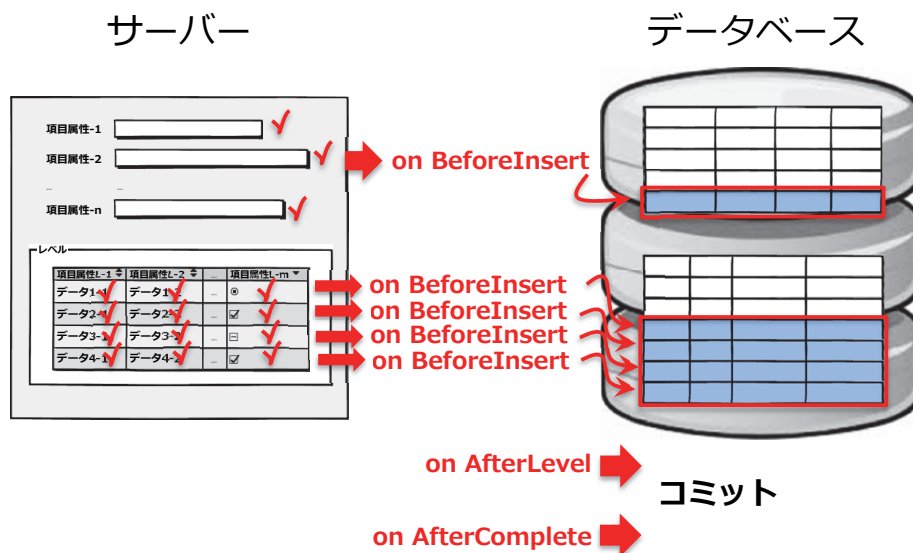
コミットの後のタイミングが **AfterComplete** で、ここでリストを呼び出すことができます。

AfterInsert



各ヘッダーまたは明細行を保存した直後にルールをトリガーするように指示する **on AfterInsert** などのタイミングもあります。

BeforeInsert



on BeforeInsert は、ヘッダーデータまたは明細行データをデータベースに保存する直前に何かを実行または評価します。

まとめ

- 場合によっては、GeneXus によって選択された、ルールをトリガーするタイミングが正しくないことがある。その場合は、ルールをいつトリガーするかを具体的に指示する必要がある。
 - On BeforeInsert
 - On AfterInsert
 - On AfterLevel
 - On AfterComplete
- ルールは、ルールステートメントの最後に接頭語「on」を使って記述することで、イベントに条件付けられる。

使用できるトリガータイミングはほかにもあります。

これらのトリガータイミングはすべて接頭語 **on** で始まり、常にルールステートメントの最後に記述することに注意してください。

