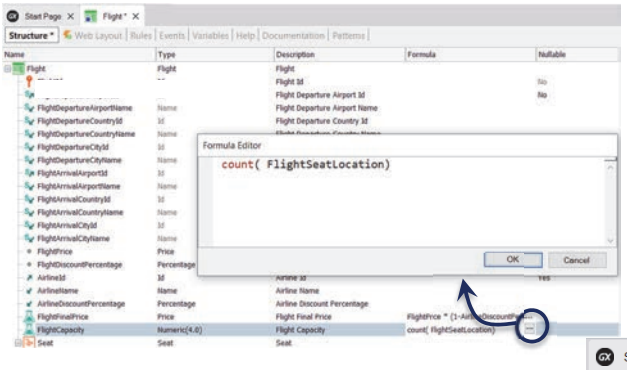
式と割り当てルール

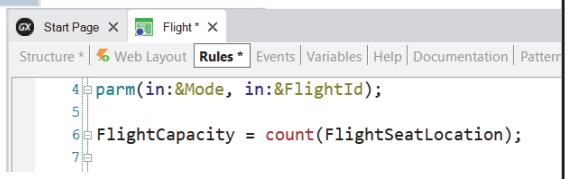
GeneXus™

この章では、式の概念について詳しく見ていきます。まず、計算によって値が決まる項目属性に、式を使用する場合と、割り当てルールを使用する場合の違いについて説明します。

計算によって値が決まる項目属性の考慮事項: ルールと式



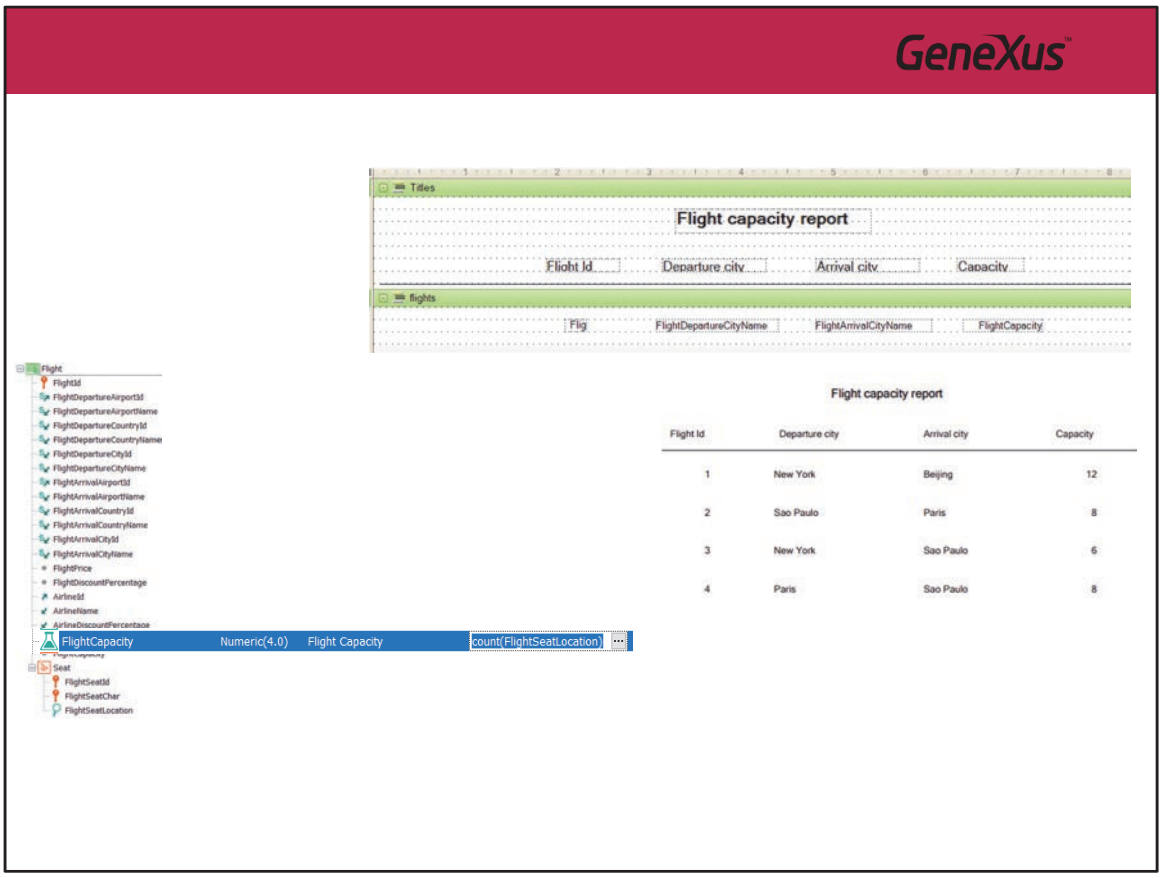
または



項目属性の値が計算によって得られるものである場合、通常はトランザクション構造の式として定義します。
これは、ルールを使用して項目属性に計算を割り当てることもできます。

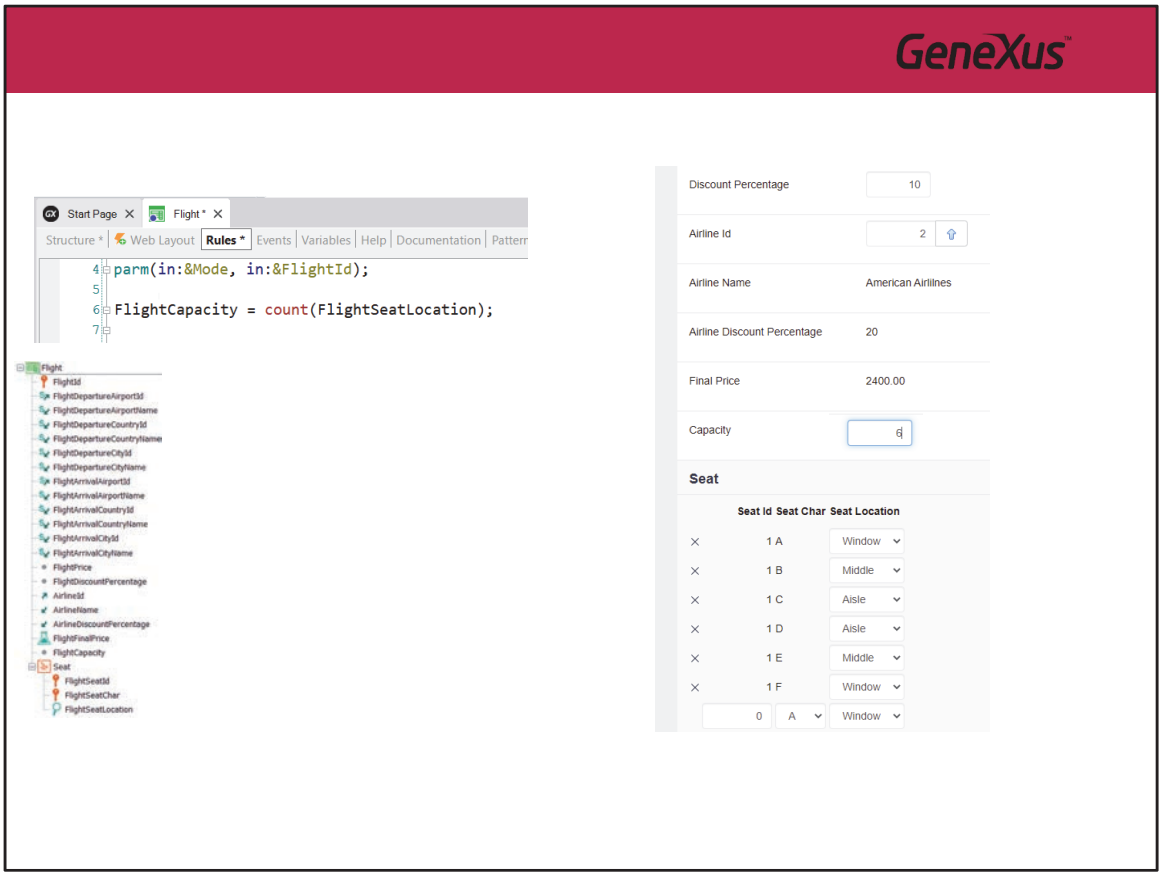
計算を割り当てるのに、ルールを使用するか、項目属性を式として定義するかを判断するには、何を考慮する必要があるでしょうか。項目属性が式の場合、GeneXusでは、関連付けられたテーブル (式項目属性が保存される場合に属するテーブル) に値を格納するフィールドが作成されないことは既に説明しました。これは、計算によって値を取得できることが明らかであるためです。このような項目属性は、トランザクション構造内には存在するものの、関連付けられているテーブルには存在しないため、「仮想」とみなされます。
テーブルの定義では、「論理」項目属性と表示されます。

一方、項目属性の値がルールによって割り当てられる場合は、テーブルに存在する値を割り当てるだけなので、仮想の項目属性ではなく格納された項目属性のままになります。



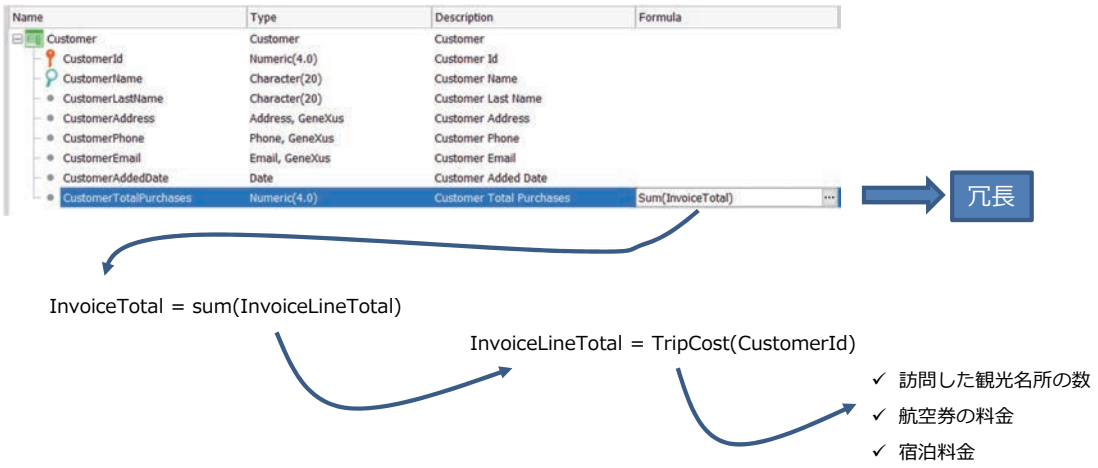
では、式よりも割り当てルールを使用するほうが良いのでしょうか。これは必ずしもそうとはいえません。項目属性をどのように使用するかによって異なります。

項目属性が他のオブジェクトでも使用されていて、その値が計算の実際の結果でなければならない場合は、式として定義します。この定義はナレッジベースに対してグローバルであるため、任意のオブジェクトが項目属性の値を問い合わせると、計算がトリガーされ、直ちに値が更新されます。



項目属性の値がトランザクションによってのみ更新される場合は、ルールを使用してローカルに値を割り当てることができます。この場合も項目属性が格納され、フォームを通じて値を変更することもできます。

計算によって項目属性の値が決まる場合の考慮事項



しかし、項目属性を毎回計算するということは、良いことだけではありません。

毎回、多数のレコードの計算を頻繁に実行しなければならない場合、リアルタイムのフィードバックが必要なケースでは、アプリケーションのパフォーマンスに影響が生じる可能性があります。

また、式項目属性を別の式項目属性から計算する場合も、多数の計算を実行して値を取得する必要があるため、パフォーマンスに影響が生じる可能性があります。

たとえば、旅行代理店の顧客の購入代金の合計金額を、顧客に発行される請求書を合計して計算する場合、各請求書の合計が顧客の旅行の合計金額として計算され、次に、各旅行の費用が訪問した観光名所の数、航空券の料金、宿泊料金などを考慮してプロシージャーによって計算されます。
旅行代理店の過去 5 年間の全顧客の購入代金合計を表示する場合、おそらく多数のレコードを確認して多数の計算を実行する必要があるため、結果を取得するために、システムの応答時間に影響を与える可能性があります。

この場合、顧客の購入代金合計をテーブルに格納しておけば、結果に影響するような変更があっても、再計算された新しい結果が格納されます。何も変更がない場合は、格納されている値を使用して計算を実行できます。
そのためには、式項目属性を冗長として定義します。つまり、計算によって値を取得しますが、計算結果はデータベースに格納され、以降はデータベースから値が取得されます。

同様の理由で、推論された項目属性も冗長として定義できます。

冗長項目属性の定義については、別の章で改めて説明します。

式またはルールによる項目属性の更新

項目属性は、式として定義することも、ルールを使用して値を割り当てることも可能であるということを見てきましたが、それぞれの更新のメカニズムも考慮する必要があります。

Add (または Subtract) ルールの使用法を思い出してください。このルールを使用することで、拡張テーブルの項目属性の値を最新に保ち、レコードを挿入、削除、または変更する際に適切な演算を実行できました。

CustomerTotalMiles の例で取り上げた、Add ルールによって更新される項目属性は、格納されている項目属性であるため、値をすぐに取得することができます。ただし、Add ルールは Customer トランザクションに対してローカルであるため、既に見たとおり、Customer トランザクション画面またはトランザクションのビジネスコンポーネントが実行された場合に、CustomerTotalMiles の項目属性のみが更新されます。

顧客の合計マイル数の項目属性の値を常に最新に保ちたい場合は、式として定義する必要があります。この場合は Sum 式を使用して、各旅行のマイル数を顧客の合計マイル数に加算します。

項目属性を式として定義すると、継続的に更新されますが、格納はできなくなります。そのため、値を取得する際にパフォーマンスの問題が発生する可能性があります。これを解決するには、CustomerTotalMiles の式項目属性を冗長として定義して、格納される項目属性にします。では、テーブルの項目属性の値はいつ更新されるのでしょうか。

式またはルールによる項目属性の更新 (続き)

式項目属性を冗長として定義すると、GeneXus で自動的にプロシージャーが作成され、値が更新されてデータベースに格納されます。

先ほどの例では、Customer トランザクション (またはそのビジネスコンポーネント) により顧客の旅行が挿入または削除されるか、CustomerTripMiles の値が影響を受けると、Sum 式が再計算されて Customer テーブルに新しい値が格納されます。

トランザクションのフォームまたはビジネスコンポーネントから、冗長項目属性の計算に含まれる一部の項目属性の値が変更されると、値を更新するプロシージャーがトリガーされます。

したがって、更新および項目属性の格納という点では、Add ルールによって項目属性を更新することと冗長式として定義することは、**同等のソリューション**となります。

GeneXus

Add ルールと冗長式の使用の比較

Add ルールのメリット	Add ルールのデメリット
ルールによって割り当てられた項目属性は常に格納される	Procedure オブジェクトを介してテーブルレコードを挿入、変更、または削除する場合、Add ルールはトリガーされない
項目属性はトランザクションのフォームで編集できる	ルールは必要なときに強制的にトリガーできないため、項目属性を強制的に更新することはできない
ルールが定義されたトランザクションから、フォームまたはビジネスコンポーネントを介してレコードが挿入、変更または削除されたときのみ項目属性が更新される	
Add ルールは、トランザクションの状態 (挿入、更新または削除) に応じて実行される演算を認識する	
拡張テーブルにアクセスするため、非常に短時間で更新できる	

更新と格納の点では、Add ルールの使用と式項目属性を冗長として定義することは同等のソリューションですが、使用の際にはいくつかの違いを考慮する必要があります。

比較表を見て、それぞれのケースのメリットとデメリットを考えてみましょう。

まず、Add ルールのメリットとデメリットを検討します。

メリット:

- Add ルールによって割り当てられる項目属性は常に格納されるため、値をすぐに取得できる。
- 項目属性は、トランザクションのフォームで編集できる。
- 項目属性は、ルールがトリガーされるたびに更新される。つまり、ルールが定義されたトランザクションから、フォームまたはビジネスコンポーネントを介してレコードが挿入、変更、または削除されたときに項目属性が更新される。
- Add ルールは、トランザクションの状態に応じて実行すべき演算を認識できる。つまり、更新の実行時に、レコードが挿入された場合は加算、削除または変更の場合は減算を実行することを認識する。
- 拡張テーブルに属するテーブルのみにアクセスするため、非常に短時間で更新できる。

デメリット:

- Procedure オブジェクトを介してテーブルレコードを挿入、変更、または削除する場合は、Add ルールがトリガーされないため、項目属性の値は更新されない。
- ルールは必要に応じて強制的にトリガーできないため、項目属性を強制的に更新することはできない。

GeneXus

Add ルールと冗長式の使用の比較

冗長式のメリット	冗長式のデメリット
冗長グローバル式の項目属性が常に格納される	フォームの項目属性は読み取り専用である
計算の項目属性が変更されると、項目属性が更新される。または、項目属性が定義されたトランザクションから、フォームまたはビジネスコンポーネントを介してレコードが挿入、変更、または削除されたときに項目属性が更新される	複数のプロシージャーの実行と複数のテーブルへのアクセスが必要なため、更新に時間がかかる
項目属性を最新に保つために実行する必要がある演算を冗長プロシージャーが認識している	冗長性を維持するために作成された Procedure オブジェクトが加えられるため、ナレッジベースのサイズが大きくなる
特殊なプロシージャーを使用して必要なタイミングで冗長性の更新を強制的にトリガーできる	式の定義に変更があった場合、GeneXus で冗長プロシージャーを更新する必要がある

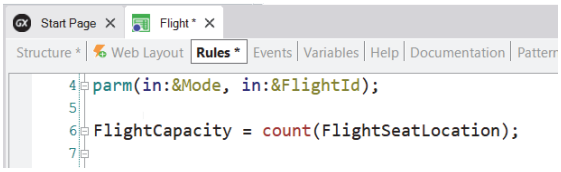
次に、式項目属性を冗長として定義する場合のメリットとデメリットを考えてみましょう。

メリット:

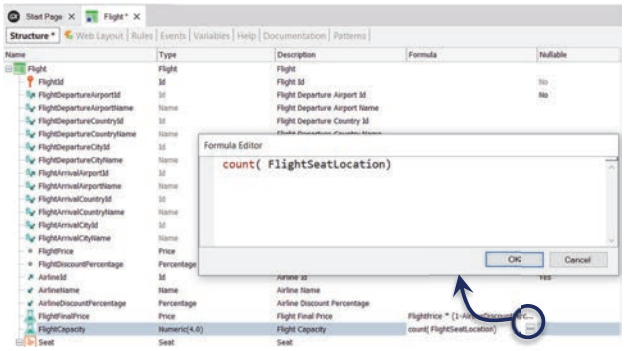
- 冗長グローバル式として定義された項目属性は常に格納される。
- 計算に含まれている項目属性が変更されると更新される。または、式項目属性が定義されたトランザクションのレコードが、フォームまたはビジネスコンポーネントを介して挿入、変更、または削除されたときに更新される。
- 冗長プロシージャーは、項目属性を最新に保つために実行すべき演算 (加算、減算、または値をどのように変更するか) を認識できる。
- 項目属性を冗長として定義したときに GeneXus で作成される特殊なプロシージャーを使用して、必要に応じて冗長性の更新を強制的にトリガーできる。

デメリット:

- 項目属性は、定義元のトランザクションのフォームでは読み取り専用になる。冗長の場合、項目属性は格納されるが、編集はできない。
- 冗長を更新する際には、複数のプロシージャーを実行し、通常は複数のテーブルにアクセスするため、時間がかかる。
- 式項目属性を冗長として定義すると、冗長性を維持するために作成されるプロシージャーが加わるため、ナレッジベースのサイズが大きくなる。
- 式の定義に変更があった場合、GeneXus で冗長プロシージャーを最新に保つ必要がある。



または



+ 冗長

したがって、項目属性の値をルールによって更新するのと、冗長式として定義するのはどちらが適切なのかは、ここで説明した考慮事項と項目属性の使用法に応じて検討する必要があります。

この後の章で、定義可能なさまざまなタイプの式を見ていきましょう。