Dokumentation SFBusinessData

 TSFConnector

 TSFBusinessDataSet/TSFDataSet

 TSFBusinessDataWrap/TSFBusinessDataWrapSource

 TSFStmt

Entwicklung von SFBusinessData unterstützen

Inhaltsverzeichnis

Dokumentation SFBusinessData ... 1
Inhaltsverzeichnis ... 2
Einleitung ... 6
Themen ... 7
- Verbindung ... 7
  - Herstellen einer Datenverbindung ... 7
- DataSet ... 8
  - Unterscheidung TSFDataSet und TSFBusinessData ... 8
  - Erstellen einer eigenen, spezialisierte Klasse (TSFBusinessDataSet) ... 8
  - Verwendung der eigenen spezialisierten Klasse (TSFBusinessDataSet) zur Designzeit und zur Laufzeit ... 9
  - Verwendung des DataSet ohne spezialisierte Klasse ... 9
  - Verarbeitung von Datenbankänderungen (Update, Insert, Delete) ... 10
  - AutoInc-Spalten ... 10
  - Feldformate ... 10
  - Transaktionen ... 11
  - Relationen/Master-Detail-Beziehungen ... 11
  - Datenmenge intern sortieren ... 11
  - Datenmenge intern filtern ... 12
  - Berechnete und Lookup-Felder in eigenen, spezialisierten BusinessDataSet-Klassen ... 13
  - Aktualiserung bzw. Neuberechnung von berechneten Feldern ... 13
  - Verwendung ohne Datenbankverbindung (ausschließlich im Speicher) ... 13
  - Hilfsklassen TSFBusinessDataWrap und TSFBusinessDataWrapSource ... 14
- Abfragegenerator ... 14
  - Hinzufügen einer SQL-Anweisung zur Designzeit und zur Laufzeit ... 14
  - Programmatische Generierung einer Select-Abfrage ... 15
  - Programmatische Generierung einer Insert-Abfrage ... 16
  - Programmatische Generierung einer Update-Abfrage ... 16
  - Programmatische Generierung einer Delete-Abfrage ... 17
  - Like-Suche ... 17
  - Subselects ... 18
  - Joins ... 19
  - Berechnete Spalten in SQL-Abfragen ... 20
  - Aggregatfunktionen ... 20
  - Benutzerdefinierte Angaben in der Select-Klausel ... 20

Verwendung von Parametern in einer SQL-Abfrage ... 20
Klassen-/Funktionsreferenz ... 22
TSFConnector ... 22
Beschreibung ... 22
Index ... 22
Funktionen ... 23
Eigenschaften ... 29
Ereignisse ... 30
TSFCustomBusinessData/TSFBusinessData/TSFDataSet ... 31
Beschreibung ... 31
Index ... 31
Funktionen ... 33
Eigenschaften ... 51
Ereignisse ... 57
TSFBDSAutoValueGenerator ... 63
Beschreibung ... 63
Index ... 63
Funktionen ... 63
Eigenschaften ... 64
TSFBDSCompareRecord ... 65
Beschreibung ... 65
Index ... 66
Funktionen ... 66
TSFBusinessDataRelation ... 67
Beschreibung ... 67
Index ... 67
Eigenschaften ... 67
TSFBusinessDataRelationDesigner ... 70
Beschreibung ... 70
Index ... 70
Eigenschaften ... 70
TSFBusinessDataWrap ... 72
Beschreibung ... 72
Index ... 72
Funktionen ... 73
Eigenschaften ... 73

TSFBusinessDataWrapSource 74
Beschreibung 74
Index 74
Eigenschaften 74
TSFStmt 75
Beschreibung 75
Index 75
Funktionen 77
Eigenschaften 103
Ereignisse 106
TSFStmtTable 109
Beschreibung 109
Index 109
Funktionen 110
Eigenschaften 115
TSFStmtTableJoin 118
Beschreibung 118
Index 118
Funktionen 118
Eigenschaften 119
TSFStmtAttr 119
Beschreibung 119
Index 119
Funktionen 120
Eigenschaften 126
TSFStmtAttrItem 128
Beschreibung 128
Index 128
Funktionen 128
Eigenschaften 129
TSFStmtCondition 130
Beschreibung 130
Index 130
Funktionen 131
Eigenschaften 132
TSFStmtConditionExists 133

Beschreibung 133
Index 133
Funktionen 133
Eigenschaften 134
TSFStmtDBDialectConv 135
Beschreibung 135
Index 135
Funktionen 136
Eigenschaften 140
TSFBDSFormatOptions 140
Beschreibung 140
Index 140
Eigenschaften 140
Typen/Konstanten 143
Index 143
Konstanten 144
Typen 145
Funktionen/Events 149

Einleitung

SFBusinessData ist eine Menge von Komponenten für Delphi®, die:

- Businesslogik und somit ganze Anwendungen durch spezialiserte DataSets klar strukturiert, indem Businesslogik in Klassen abgebildet wird, die das Datenmodell referenzieren
- Code durch eine saubere Trennung von Business- und Anzeigelogik wiederverwendbar macht
- SQL-Abfragen datenbankunabhängig macht, indem diese nicht hardcordiert im Code programmiert, sondern über einen Abfragegenerator verwaltet und generiert werden

Kern bildet, neben dem Abfrage-Generator, das BusinessDataSet (TSFBusinessData = Basisklasse für eigene, spezialisierte BusinessDataSet-Klassen), welches auch ohne konkrete Datenbankverbindung verwendet werden kann. Hierdurch können bspsw. auch datenabhängige Controls und Gültigkeitsprüfungen bei der Eingabe - analog der Logik bei sämtlichen DataSets - für sämtliche Daten angewandt werden.

Zur Verbindung mit Datenbanken verwenden die Komponenten keine eigene Datenbankverbindungen, sondern greifen auf die von Delphi® hierfür zur Verfügung gestellten Mechanismen zurück.
Das Bindeglied zwischen BusinessDataSet und Datenbankverbindung ist ein Connector (TSFConnector).

Zur Verwendung von BusinessDataSets über den Designer (IDE) ist ein Wrapper (TSFBusinessDataWrap) erforderlich, der zur Laufzeit das spezialiserte BusinessDataSet kapselt.
Zur Bindung an Controls ist der Wrapper einer speziellen DataSource (TSFBusinessDataWrapSource) zuzuweisen, die einer gewöhnlichen DataSource (TDataSource) gleicht, nur eben eine Instanz von TSFBusinessDataWrap referenzieren kann.

Themen

Verbindung

Herstellen einer Datenverbindung

TSFBusinessData lieferert keine eigenen Datenbank- oder Verbindungskomponenten mit. Die Datenbankverbindung erfolgt über entsprechende Standardkomponenten (abhängig von der gewählten Zugriffstechnologie).
Die Komponente TSFConnector ist die Verbindung zwischen der Datenbankverbindung und den DataSets.

Zum Herstellen einer Datenbankverbindung gehen Sie wie folgt vor:

1. Erstellen Sie einen TSFConnector, indem Sie diesen z. B. im Designer auf eine Form oder ein Datenmodul ziehen
2. Wählen die gewünschte Zugriffstechnologie in der Eigenschaft *ConnectionType* aus
3. Erstellen Sie eine Verbindungskomponte, indem Sie diese z. B. über den Designer auf eine Form oder ein Datenmodul ziehen. Abhängig vom gewählten Typ, verwenden Sie folgende Komponenten:
   - ctADO = TADOConnection
   - ctDBExpress = TSQLConnection
   - ctFireDac = TFDConnection
   - ctInterbase = TIBDatabase

   Bei FireDac-Verbindungen (ctFireDac) ist zusätzlich eine passender *DriverLink* erforderlich, z. B. bei Verwendung von MsAccess eine Komponente *TFDPhysMsAccessDriverLink*.

4. Ordnen Sie die erstellte Verbindungskomponente dem TSFConnector unter *Connection* zu.
5. Der Datenbanktyp im TSFConnector wird, je nach Zugriffstechnologie, automtisch ermittelt und gesetzt. Wenn dieser nicht ermittelt werden konnte, wählen Sie den passenden Datenbanktyp in der Eigenschaft *ConnectionDBType* aus.
6. Wenn der TSFConnector für die gesamte Anwendung zuständig sein soll, setzen Sie die Eigenschaft *CommonConnector* auf *true*. Hierdurch wird der TSFConnector von DataSets automatisch erkannt und verwendet (sofern einem DataSet nicht explizit ein eigener TSFConnector zugewiesen wurde).
   Beachten Sie bei Verwendung von *CommonConnector* bitte, dass ein mit *CommonConnector* gekennzeichneter TSFConnector nur dann von DataSets gefunden werden kann, wenn dieser vorher auch erzeugt wurde. Wenn Sie z. B. einen TSFConnector in einem Datenmodul verwenden, muss das Datenmodul instanziert sein, bevor das erste DataSet auf den TSFConnector zugreift.

DataSet

Unterscheidung TSFDataSet und TSFBusinessData

TSFBusinessData ist die Basisklasse für spezialisierte (eigene) DataSet-Klassen, wohingegen TSFDataSet nicht spezialisiert ist und wie ein normales DataSet verwendet werden kann.

Die Basisfunktionalitäten sind in beiden Klassen gleich.

Erstellen einer eigenen, spezialisierte Klasse (TSFBusinessDataSet)

Um eine spezialisierte DataSet-Klasse zu erstellen, gehen Sie wie folgt vor (siehe auch UBusinessDataTmpl.pas):

1. Erstellen Sie eine neue Unit (Datei > Neu > Unit - Delphi)
2. Definieren Sie eine neue Klasse, die von TSFBusinessData abgeleitet ist und benennen die Klasse nach dem Tabellennamen in der referenzierenden Datenbank.
3. Überschreiben Sie in Ihrer Klasse den constructor
4. Setzen Sie im constructor die String-Eigenschaften *TableName*, *CatalogName*, *SchemaName*
5. Fügen Sie Ihrer Unit einen initialization-Abschnitt hinzu und registrieren dort Ihre Klasse.

Bsp. einer Klassendefinition:

```
unit ...;

uses SFBusinessData, SFBusinessDataCustom, Data.DB;

type

	Customer = class(TSFBusinessData)
	public
		constructor Create(Component: TComponent); override;
	end;

implementation

constructor Customer.Create(Component: TComponent);
begin

	inherited;

	TableName := 'customer';
	SchemaName := '';
	CatalogName := '';
end;
```

```
initialization
begin
        TSFBusinessClassFactory.RegisterClass(customer, 'customer);
end;
```

Verwendung der eigenen spezialisierten Klasse (TSFBusinessDataSet) zur Designzeit und zur Laufzeit

Um eine Instanz einer spezialisierten Klasse von TSFBusinessData zur ***Designzeit*** zu integrieren, ist die Hilfsklasse TSFBusinessDataWrap erforderlich.

Erstellen Sie auf Ihrer Form/Ihrem Datamodul eine Komponente vom Typ TSFBusinessDataWrap und geben bei dieser in der Eigenschaft *BusinessClassName* den Namen Ihrer Klasse an.
Hierauf ist unter der Eigenschaft *BusinessDataSet* eine Instanz von TSFBusinessDataSet verfügbar, über welche Sie Eigenschaften konfigurieren, den Query-/Statementgenerator (über die Eigenschaft *Stmt*) aufrufen und Events belegen können.

Während der Designzeit ist die Instanz von TSFBusinessDataSet unspezialisiert, erst zur Laufzeit ist das *BusinessDataSet* eine spezialisierte Instanz der angegebenen Klasse.

Wenn Sie im Code - um bspsw. Funktionen Ihrer Klasse aufzurufen - das BusinessDataSet ansprechen wollen, verwenden Sie *cast*, z. B.

```
        Customer(BusinessDataWrapper1.BusinessDataSet).MyFuntion1;
```

Zur Verknüpfung BusinessDataSet mit datengebunden Controls erstellen Sie eine Komponente vom TSFBusinessDataWrapSource, der Sie in der Eigenschaft BusinessDataWrapper die Komponenten vom Typ TSFBusinessDataWrap zuweisen.

Um eine Instanz Ihrer spezialisierten Klasse zur ***Laufzeit*** zu instanzieren, rufen Sie einfach den *constructor* Ihrer Klasse auf. Ihr spezialisiertes DataSet können Sie verwenden, wie ein gewöhnliches DataSet (einer DataSource zuweisen, usw.).
Bsp.:

```
var mySpecDataSet: Customer;
…
mySpecDataSet := Customer.Create;
DataSource1.DataSet := mySpecDataSet;
```

Verwendung des DataSet ohne spezialisierte Klasse

Zur Verwendung eines DataSets ohne spezialisierte Klasse verwenden Sie TSFDataSet und setzen die Eigenschaften *TableName*, *CatalogName* und *SchemaName*.

Den Query-/Statementgenerator rufen Sie über die Eigenschaft *Stmt* oder das Konextmenü der Komponente auf.

Bei Instanzierung zur ***Laufzeit*** können Sie auch die Eigenschaften TableName, CatalogName und SchemaName auch bereits im constructor übergeben.

Bsp.:

```
var myDataSet: TSFDataSet;
…
// Create(TableName, CatalogName, SchemaName: String; Owner: TComponent = nil)
myDataSet:= TSFDataSet.Create('customer', '', '');
DataSource1.DataSet := myDataSet;
```

Verarbeitung von Datenbankänderungen (Update, Insert, Delete)

Datenänderungen werden automatisch verarbeitet, d. h. hierfür sind weder separate Statements/Queries noch separate Komponenten erforderlich.

Über die Eigenschaft UpdateMode können Sie einstellen, wie Update-/Delete-Statements generiert werden.

Änderungen an Feldwerten sind nur für Felder der durch *TableName*, *CatalogName*, *SchemaName* identifizierten Basistabelle möglich. Diese Tabelle bildet zugleich auch die Basistabelle für den Query-/Statementgenerator.

AutoInc-Spalten

Je nach Zugriffstechnologie (dbExpress, FireDac, ADO, Interbase) werden AutoInc-Spalten automatisch erkannt.

Um programmatisch eine AutoInc-Spalte zu definieren verwenden Sie die Funktion AddAutoValueForField. Diese Funktion gibt eine Instanz vom Typ TSFBDSAutoValueGenerator zurück, welche Sie dann anpassen können (z. B. setzen einer Sequenz). Wollen Sie eine automatisch erkannte AutoInc-Spalte anpassen, verwenden Sie die Funktion GetAutoValueForField.

Weiter könne Sie auch eine eigene Klasse (abgeleitet von TSFBDSAutoValueGenerator) zur Generierung von Autowerten definieren, siehe hierzu TSFBDSAutoValueGenerator, GetAutoValueCls.

Beim Einfügen eines Datensatzes in ein DataSet mit Autowerten, wird den Autowerten eine temporäre (negative) Id zugewiesen.

Feldformate

Zur Definition von Formaten für Feldklassen verwenden Sie TSFConnector.FormatOptions oder TSFCustomBusinessData.FormatOptions.
Sind keine Feldformate definiert, wird das jeweilige Format der Quelldatenmenge übernommen, welches ggf. über die entsprechende Verbindungskomponte definiert weden kann.

Transaktionen

Abhängig von der gewählten Zugriffstechnologie, können/müssen (FireDac, Interbase) Sie mit Transaktionen arbeiten.

Hierfür können Sie Ihrem DataSet in den Eigenschaften Transaction und UpdateTransaction Transaktionskomponeten zuordnen. Der Typ der Transaktionskomponenten ist abhängig von der gewählten Zugriffstechnologie:

- ctFireDac = TFDTransaction
- ctInterbase = TIBTransaction

Auch können Sie die Transaktionskomponenten direkt Ihrer Verbindungskomponente zuweisen, siehe TFDConnection.Transaction, TFDConnection.UpdateTransaction, TIBDatabase.DefaultTransaction.

Relationen/Master-Detail-Beziehungen

Relationen sind DataSets, die in einer Master-Detail-Beziehung stehen. Detail-DataSets werden vom Master-DataSet automatisch synchronisiert.

Zur Defintion von Relationen verwenden Sie zur Designzeit die Eigenschaft ParentRelationDesigner.

Zur Laufzeit oder in spezialisierten BusinessData-Klassen siehe AddRelation.

Datenmenge intern sortieren

Sie können eine Datenmenge intern sortieren, d. h. dass der interne Datensatzpuffer sortiert und keine neue Anfrage an die Datenbank gesendet. Hierfür wird ein QuickSort-Algorithmus verwendet.

Zum Vergleich der Werte ruft die Sortierfunktion die Funktion SortBuffer die Vergleichsfunktion CompareRecords und das Ereignis OnCompareRecords auf.
Zur Verwendung der Sortierung müssen die Funktion (für spezialisierte BusinessData-Klassen) oder das Ereignis behandeln.

Der Funktion und dem Ereignis werden 2 Records/Datensätze vom Typ TSFBDSCompareRecord übergeben, um einzelne Werte zu vergleichen, verwenden Sie die Funktionen dieser Klasse.

Bsp.:

…

```
myDataSet.SortBuffer;
```

…

```
function TForm1.TSFBusinessDataCompareRecords(CompareRecordFrom,
            CompareRecordTo: TSFBDSCompareRecord): TSFBDSRecordCompareResult;
begin
   // compare given records (called by SortBuffer)
   if (CompareRecordFrom.GetFieldValByName('Field1') >
      CompareRecordTo.GetFieldValByName('Field1')) then
     Result := compareResultGreater
   else if (CompareRecordFrom.GetFieldValByName('Field1') <
            CompareRecordTo.GetFieldValByName('Field1')) then
     Result := compareResultLess
   else
     Result := compareResultEqual;
end;
```

Datenmenge intern filtern

Mit Filtern einer Datenmenge können bestimmte Datensätze aus dem Datensatzpuffer ausgeblendet werden ohne dass eine neue Anfrage an die Datenbank gesandt wird.

Um Datensätze zu filtern, müssen Sie die Funktion FilterRecord überschreiben (für spezialisierte BusinessData-Klassen) oder das Ereignis TDataSet.OnFilterRecord behandeln. Der var-Parameter Accept gibt hierbei an, ob der aktuelle Datensatz angezeigt wird oder nicht (true = Datensatz wird angezeigt; false = Datensatz wird ausgeblendet). Auf Feldwerte können Sie innerhalb der Funktion/des Ereignisses über die regulären Funktionen zugreifen.

Bsp.:

…

```
myDataSet.Filtered := true;
```

…

```
procedure TForm1.TSFBusinessDataFilterRecord(var Accept: Boolean);
begin
        Accept := (myDataSet.FieldByName('Field1').AsInteger > 10);
end;
```

Berechnete und Lookup-Felder in eigenen, spezialisierten BusinessDataSet-Klassen

Zur Designzeit können Sie, wie in anderen DataSets, über den Feldeditor persistente Felder einschl. Lookup-Feldern und berechneten Feldern definieren. Beachten Sie hierzu die Hilfe-Datei der Entwicklungsumgebung.

Für BusinessDataSets können Sie weiter auch dynamisch, d. h. ohne dass Sie alle Felder persistent im Feldeditor erstellen müssen, Lookup-Felder und berechnete Felder im Code definieren.
Verwenden Sie hierzu die Funktionen AddDynCalcField und AddDynLkpField. Diese Funktionen können in auch spezialisierten BusinessData-Klassen verwendet werden, z. B. dann, wenn ein berechnetes Feld für alle Instanzen verfügbar sein soll.

Im Feldcontainer (TDataSet.Fields) sind die über obige Funktionen hinzugefügten Felder nach dem Öffnen der Datenmenge (wie alle anderen nicht persistente Felder) verfügbar.

Um die Werte von berechneten Feldern zu setzen, verwenden Sie die virtuelle Funktion TDataSet.DoOnCalcFields oder das Ereignis TDataSet.OnCalcFields.

Aktualiserung bzw. Neuberechnung von berechneten Feldern

Um berechnete Felder im Code manuell zu aktualisieren, verwenden Sie die Funktion RecalcCalculatedFields. Hierdurch wird wiederum die Funktion TDataSet.DoOnCalcFields aufgerufen und das Ereignis TDataSet.OnCalcFields ausgelöst.

Verwendung ohne Datenbankverbindung (ausschließlich im Speicher)

Sie können auch Instanzen von TSFDataSet bilden, die keine Datenbanktabelle referenzieren. D. h., dass das DataSet rein im Speicher ist, aber verwendet werden kann, wie ein reguläres DataSet.

Um ein DataSet ohne Datenbankverbindung zu erzeugen rufen Sie den constructor ohne Angabe von *TableName*, *SchemaName* und *CatalogName* auf. Setzen die genannten Eigenschaften auch nicht nachträglich.

```
// constructor Create(pOwner: TComponent); overload; override;
// constructor Create; reintroduce; overload; virtual;
myBufferDataSet := TSFDataSet.Create;
```

Fügen Sie Ihrem DataSet im nächsten Schritt Felder hinzu. Verwenden Sie hierzu die Funktion AddField.

```
myBufferDataSet.AddField ('Field1', ftInteger, 0);
myBufferDataSet.AddField ('Field2', ftString, 20);
```

Um Felder für ein gepuffertes DataSet analog der Felder einer Datentabelle bzw. eines spezialisierten BusinessDataSet zu erzeugen, verwenden Sie die Funktion InitFieldsFromBusinessData.

```
myBufferDataSet.InitFieldsFromBusinessData('customer', '', '', True, True, True);
```

Anschließend kann das DataSet geöffnet werden.

```
myBufferDataSet.Open;
```

Einfüge- und Änderungsoperation führen Sie mit den regulären Funktionen von TDataSet durch.

```
myBufferDataSet.Insert;
myBufferDataSet.FieldByName(‘Field1').AsInteger := 1;
myBufferDataSet.FieldByName(‘Field2').AsString := 'Test';
myBufferDataSet.Post;

myBufferDataSet.Edit;
myBufferDataSet.FieldByName(‘Field2').AsString := 'For testing';
myBufferDataSet.Post;
```

Die Funktionen zur internen Sortierung und Filterung können ebenfalls angewandt werden.

Hilfsklassen TSFBusinessDataWrap und TSFBusinessDataWrapSource

Die Klasse TSFBusinessDataWrap ist erforderlich, um eine Instanz einer spezialisierten BusinessData-Klasse über den Designer zu integrieren.

TSFBusinessDataWrapSource ist eine DataSource (abgeleitet von TDataSource), die anstatt eines DataSets eine Komponenten/eine Instanz von TSFBusinessDataWrap entgegennimmt.

Siehe hierzu Verwendung der eigenen spezialisierten Klasse von TSFBusinessDataSet zur Designzeit und zur Laufzeit

Abfragegenerator

Hinzufügen einer SQL-Anweisung zur Designzeit und zur Laufzeit

Zur Designzeit steht ein Assistent zum Hinzufügen einer Abfragedefinition zur Verfügung. Dieser wird über die Eigenschaft Stmt einer BusinessData-Komponente oder über das Kontextmenü bei einer TSFStmt-Komponente aufgerufen.

Der Aufbau des Assistenten orientiert sich an der Klassenstruktur:

- General = Allgemeine Optionen, siehe Eigenschaften TSFStmt
- Tables = Tabellen und Joins
- Attributes = Attribute/Felder und Items
- Conditions = allgemeine Suchbedingungen und Exists-Bedingungen
- Order = Sortierung
- Group = Gruppierung
- Test = Generiert die SQL-Abfrage, siehe GetSelectStmt

Zur Laufzeit stehen Funktionen zur Definition von SQL-Abfragen zur Verfügung, siehe hierzu Programmatische Generierung einer Select-Abfrage.

Programmatische Generierung einer Select-Abfrage

Erster Schritt zur Definition einer Abfrage ist das Setzen einer Basistabelle. Bei einer Statement-Komponenten innerhalb eines BusinessData-Objekts wird die Basistabelle automatisch gesetzt. Innerhalb eines BusinessData-Objekts verwenden Sie zur Definition der Abfrage die Eigenschaft *Stmt*.

```
// SetBaseTable(pTableName, pSchema, pCatalog, pTableAlias: String)
myStmt.SetBaseTable('customer', '', '', '');
```

Der Alias wird i. d. R. automatisch vergeben und verwaltet, d. h. jede hinzugefügte Tabelle/Join bekommt einen automatisch generierten Alias zugewiesen, sofern kein eigener Alias übergeben wurde.
Mit der überladenen Funktion von *SetBaseTable* können Sie auch eine andere Instanz von TSFStmt als Basistabelle übergeben, womit die Basistabelle ein Subselect wäre.

Nun können Sie Ihrer Basistabelle Joins hinzufügen.

```
// SetTableJoin(pTableAlias, pTableName, pSchema, pCatalog, pSourceTableAlias:
// String; const pRelValsDest, pRelValsSource: Array of Variant; const pRelTypesDest,
// pRelTypesSource: Array of TSFStmtJoinRelItemType; pType: TSFStmtJoinType):
// TSFStmtTable; overload;
myStmt.SetTableJoin('', 'customertype', '', '', 'customer', ['customertypeid'],
['customertypeid'], [stmtJoinRelItemAttr], [stmtJoinRelItemAttr], stmtJoinTypeInner);
```

Weiter können Sie Joins auch verschachteln, d. h. einem Join bzw. einer "gejointen" Tabelle einen Join hinzufügen. Zur Referenzierung der Basistabelle können Sie wahlweise den (automatisch oder selbst vergebenen) Alias oder - wie im Beispiel - den Tabellennamen angeben.

Nächster Schritt ist das Konfigurieren der Select-Klausel, also das Hinzufügen von Attributen, siehe SetStmtAttr, SetStmtAggr, AddStmtAttr.
Wenn Sie keine sichtbaren Attribute hinzufügen, wird eine *Select * generiert.

```
// SetStmtAttr(pAttrName, pAttrAlias, pTableAlias: String; pOnlyForSearch: Boolean);
myStmt.SetStmtAttr('*', '', 'customer', False);
myStmt.SetStmtAttr('customertypedesc', '', 'customertype', False);

// AddStmtAttr(pAttrName: String; pOnlyForSearch: Boolean)
with myStmt.AddStmtAttr('incid', False) do
begin
    AddItemDbFld('customer', 'customerid', '');
    AddItemOperator(stmtAttrItemTypeOpPlus);
    AddItemValue(1);
end;
```

Wenn Sie auf ein Feld eine Suchbedingung oder eine Sortierung setzen wollen, ohne dass es explizit in der Select-Klausel aufgeführt werden soll, müssen Sie dieses Feld trotzdem als Attribut mit *OnlyForSearch* hinzufügen.

```
myStmt.SetStmtAttr('customerid', '', 'customer', True);
```

Abschließend setzen Sie Suchbedingungen und Ihre Sortierung, siehe AddConditionVal, AddConditionAttr, AddConditionIsNull, AddConditionIsNotNull, AddConditionType, AddConditionExists, AddOrderAttr

```
// AddConditionIsNotNull(pTableAlias, pAttrName: String; pRestrict: Boolean = False);
myStmt. AddConditionIsNotNull('customer', 'customerid');

// AddOrderAttr(pTableAlias, pAttrName: String; pOrderType: TSFStmtSortType =
// stmtSortTypeAsc);
myStmt.AddOrderAttr('customer', 'customerid');
```

Wenn zwischen Suchbedingungen kein Operator (AND, OR) angegeben ist, wird bei der Generierung automatisch ein AND-Operator generiert.

Das Flag Restrict bei Suchbedingungen gibt an, ob es sich um geschützte Suchbedingung handelt, welche mit ClearConditions nicht gelöscht werden. Suchbedingungen, die mit Restrict gekennzeichnet sind separat zu behandeln, d. h. diese werden separat generiert und können somit nicht direkt mit anderen Suchbedingungen verknüpft werden.

Die nun definierte Abfrage würde nachfolgendes Ergebnis generieren, siehe auch GetSelectStmt:

```
SELECT t1.*, t2.customertypedesc, t1.customerid + 1 as incid
FROM customer t1 inner join customertype t2 on t1.customertypeid = t2.customertypeid
WHERE t1.customerid is not NULL
ORDER BY t1.customerid
```

Programmatische Generierung einer Insert-Abfrage

Setzen Sie die Basistabelle, siehe SetBaseTable.

```
// SetBaseTable(pTableName, pSchema, pCatalog, pTableAlias: String)
myStmt.SetBaseTable('customer', '', '', '');
```

Fügen Sie die einzufügenden Werte hinzu

```
// AddInsertCondition(pAttrName: String; pVal: Variant; pValType:
// TSFStmtAttrItemValueType);
myStmt.AddInsertCondition('customerid', 1, stmtAttrItemTypeValue);
myStmt.AddInsertCondition('customername', 'Test', stmtAttrItemTypeValue);
```

Das SQL für Ihre Insert-Abfrage wird über GetInsertStmt generiert.

Programmatische Generierung einer Update-Abfrage

Setzen Sie die Basistabelle, siehe SetBaseTable.

```
// SetBaseTable(pTableName, pSchema, pCatalog, pTableAlias: String)
myStmt.SetBaseTable('customer', '', '', '');
```

Definieren Sie die SET-Klausel

```
// AddSetCondition(pAttrName: String; pVal: Variant; pValType:
// TSFStmtAttrItemValueType);
myStmt.AddSetCondition('customername', 'Test', stmtAttrItemTypeValue);
```

Definieren Sie die WHERE-Klausel, siehe auch Programmatische Generierung einer Select-Abfrage.

```
myStmt.SetStmtAttr('customerid', '', 'customer', True);
myStmt. AddConditionVal('customer', 'customerid', SFSTMT_OP_EQUAL, 1);
```

Das SQL für Ihre Update-Abfrage wird über GetUpdateStmt generiert.

Programmatische Generierung einer Delete-Abfrage

Setzen Sie die Basistabelle, siehe SetBaseTable.

```
// SetBaseTable(pTableName, pSchema, pCatalog, pTableAlias: String)
myStmt.SetBaseTable('customer', '', '', '');
```

Definieren Sie die WHERE-Klausel, siehe auch Programmatische Generierung einer Select-Abfrage.

```
myStmt.SetStmtAttr('customerid', '', 'customer', True);
myStmt. AddConditionVal('customer', 'customerid', SFSTMT_OP_EQUAL, 1);
```

Das SQL für Ihre Update-Abfrage wird über GetDeleteStmt generiert.

Like-Suche

Bei der Like-Suche ist nach einer Like-Einzelsuche und einer Like-Mengensuche zu unterscheiden.

Eine Like-Einzelsuche soll bedeuten, dass die Anzahl der Zeichen im Suchstring fest definiert ist, einzelne Zeichen aber variabel sein können, z. B. wenn Sie nach Namen mit verschiedenen Schreibweisen suchen, wie Mayer, Meyer, Maier, Meier.
Bei einer Like-Einzelsuche geben Sie das Platzhalterzeichen in Ihrem Suchstring an. Das Platzhalterzeichen können Sie über TSFCustomBusinessData.GetLikeWildcardSingle oder TSFStmt.GetDBDialectLikeWildcardSingle ermitteln.

```
…
    var wildcard, search: String;
…
    wildcard := myStmt.GetDBDialectWildcardSingle;
    search := 'M' + wildcard + wildcard + 'er';
    myStmt. AddConditionVal('customer', 'customername', SFSTMT_OP_LIKE, search);
```

Bei einer Like-Mengensuche suchen Sie nach einem Teilstring. Wenn Sie AutoEscapeLike verwenden, werden das Wildcard-Zeichen automatisch hinzugefügt, Wildcard-Zeichen innerhalb des Suchstrings werden, sofern von Datenbanksystem unterstützt, "escaped".

"Escaped" bedeutet, dass Wildcard-Zeichen innerhalb des Suchstrings mittels eines zusätzlichen ESCAPE-Zeichnen gekennzeichnet werden. Hierdurch wird dem Datenbanksystem mitgeteilt, dass es sich in diesem Fall nicht um das Wildcard-Zeichen handelt.

Bei Verwendung von AutoEscapeLike fügen Sie die Suchbedingung ohne Wildcard-Zeichen hinzu.

```
    myStmt. AddConditionVal('customer', 'customernotice', SFSTMT_OP_LIKE, '100%');
```

Wenn AutoEscapeLike nicht verwendet wird, fügen Sie der Suchbedingung wiederum das Wildcard-Zeichen hinzu.

```
…
    var wildcard, search: String;
…
    myStmt.AutoEscapeLike := false;
    wildcard := myStmt.GetDBDialectWildcardMany;
    search := wildcard + '100' + wildcard;
    myStmt. AddConditionVal('customer', 'customernotice', SFSTMT_OP_LIKE, search);
```

Mit LikeEscapeChar können Sie Ihre Like-Suche manuell "escapen". Das Setzen von LikeEscapeChar hat nur dann Wirkung, wenn AutoEscapeLike nicht verwendet wird und das verwendete Datenbanksystem die ESCAPE-Anweisung unterstützt.

```
…
    var wildcard, escape, search: String;
…
    escape := '#';
    myStmt.AutoEscapeLike := false;
    myStmt.LikeEscapeChar := escape;
    wildcard := myStmt.GetDBDialectWildcardMany;
    search := wildcard + '100' + escape + '%' + wildcard;
    myStmt. AddConditionVal('customer', 'customernotice', SFSTMT_OP_LIKE, search);
```

Siehe auch Programmatische Generierung einer Select-Abfrage.

Subselects

Subselects können hinzugefügt werden, als

- Attributeitem für die Select-Klausel
- Tabelle
- (NOT) EXISTS-Suchbedingung

Um einen Subselect über den Designer hinzuzufügen, erstellen Sie für den Subselect eine Komponente vom Typ TSFStmt, konfigurieren diese (Attribute, Suchbedingungen, usw.) über den Assistenten (s. Hinzufügen einer SQL-Anweisung zur Designzeit und zur Laufzeit).
Nun können den erstellten Subselect über den Assistenten der Basisabfrage an den entsprechenden Stellen referenzieren.

Um ein Subselect über den Code hinzuzufügen, bilden Sie eine neues Objekt vom Typ TSFStmt.

```
mySubselect := TSFStmt.Create(nil);
```

Setzen Sie ggf. Attribute/Felder, Joins, Suchbedingungen für Ihren Subselect - siehe hierzu Programmatische Generierung einer Select-Abfrage.

Fügen Sie Ihren Subselect der Basisabfrage hinzu,

als Attributitem für die Select-Klausel

```
with myStmt.AddAttr('mysub', False) do
    AddItemStmt(mySubselect);
```

als Basistabelle

```
myStmt.SetBaseTable(mySubselect,'');
```

als Join

```
myStmt.SetTableJoin('', 'customer', mySubselect, ['subselcustomerid'],
['customerid'], [stmtJoinRelItemAttr], [stmtJoinRelItemAttr], stmtJoinTypeInner);
```

als (NOT) EXISTS-Suchbedingung

```
myStmt.AddConditionExists(mySubselect, 'customer', 'subtab1', SFSTMT_OP_EXISTS,
['subselcustomerid'], ['customerid'], [stmtJoinRelItemAttr], [stmtJoinRelItemAttr]);
```

Geben Sie Subselects nicht frei, diese werden automatisch von der Basisabfrage freigegeben.

Joins

Sie können als Join eine Tabelle oder einen Subselect hinzufügen.

Siehe hierzu:

Hinzufügen einer SQL-Anweisung zur Designzeit und zur Laufzeit
Programmatische Generierung einer Select-Abfrage
Subselects
TSFStmt.SetTableJoin

Berechnete Spalten in SQL-Abfragen

Um berechnete Spalten in der Select-Klausel einer Abfrage hinzuzufügen, fügen Sie die einzelnen Elemente als Items hinzu.

```
with myStmt.AddStmtAttr('incid', False) do
begin
    AddItemDbFld('customer', 'customerid', '');
    AddItemOperator(stmtAttrItemTypeOpPlus);
    AddItemValue(1);
end;
```

Der Name des Attributs entspricht dem Spaltennamen in der Ergebinsmenge (… AS incid).

Aggregatfunktionen

Um eine einfache Aggregatfunktion für ein Datenbankfeld hinzuzufügen, verwenden Sie

```
myStmt.SetStmtAggr(SFSTMTAGGR_SUM, 'customerid', 'sumid', 'customer');
```

Auch können Sie berechnete Spalten und Aggregate kombinieren

```
with myStmt.AddStmtAttr('sumidinc', False) do
begin
    AddItemAggrFunc(SFSTMTAGGR_SUM);
    AddItemBracket(stmtAttrItemTypeBracketOpen);
    AddItemDbFld('customer', 'customerid', '');
    AddItemBracket(stmtAttrItemTypeBracketClose);
    AddItemOperator(stmtAttrItemTypeOpPlus);
    AddItemValue(1);
end;
```

Benutzerdefinierte Angaben in der Select-Klausel

Für die Select-Klausel besteht die Möglichkeit, benutzerdefinierte Attribute anzulegen. Dies kann z. B. dazu verwendet werden, um datenbankspezifische Funktionen hinzuzufügen. Für den Abfragetext innerhalb dieses Attributs sind Sie selbst verantwortlich, deshalb sollten Sie hierbei bedenken, dass Ihr Code - insbesondere bei Verwendung datenbankspezifischer Funktionen - vielleicht nicht mehr datenbankunabhängig ist.

```
with myStmt.AddStmtAttr('userdefattr', False) do
    AddItemDynamic ('substr(t1.customername, 1, 5)');
```

Verwendung von Parametern in einer SQL-Abfrage

Zum Hinzufügen von Parametern verwenden Sie

```
with myBusinessDataObj.Stmt.AddStmtAttr('myprm1', True) do
    AddItemParam ('myprm1');
```

Nun weisen Sie den Parameter einer Suchbedingung zu

```
myBusinessDataObj.Stmt.AddConditionAttr('customer', 'customerid',
SFSTMT_OP_EQUAL, '', ' myprm1');
```

Bei Ausführung der Abfrage über ein BusinessData-Objekt (*myBusinessDataObj.Open*) müssen Sie dem Parameter nun einen Wert zuweisen.

Hierfür verwenden Sie entweder das Ereignis OnSetParams oder setzen die Werte vor Ausführung direkt über die Eigenschaft StmtParamValues.

Klassen-/Funktionsreferenz

TSFConnector

Beschreibung

Stellt das Bindeglied zwischen der eigentlichen Datenbankverbindung (z. B. TSQLConnection) und den Instanzen von TSFBusinessData/TSFDataSet dar.

Sie können einen einzigen Connector für Ihre gesamte Anwendung verwenden, indem Sie *CommonConnector* auf *true* setzen.
Die Instanzen von TSFBusinessData/TSFDataSet ermitteln sich den Connector somit selbst, ohne dass hier ein Connector referenziert wird.
Hierbei sollten Sie allerdings darauf achten, dass der Connector auch erzeugt wird, z. B. indem Sie diesen auf einem automatisch erzeugten Datenmodul platzieren.

Index

ActiveTransactionForDataSet
AddCommonConnectedProc
AddConnectorMsgNotification
CanDBInsertion
CheckTransaction
CommitTransactionForDataSet
CommonConnector
Connection
ConnectionDBType
ConnectionType
FormatOptions
GetCommonConnector
GetConnectionDBType
GetFieldNames
GetKeyFields
GetNewQuery
GetNewTable
GetQueryParamName
HasDataSetTransaction
OnDataSetCreated
QueryExecSQL
RemoveCommonConnectedProc
RemoveConnectorMsgNotification
SequenceNameForField
SetQueryParamValue
SetSQLToQuery
StartTransactionForDataSet

Funktionen

GetNewQuery

Notation:

function GetNewQuery(pTransaction: TComponent; pActionType: TSFQueryActionType; pCanUniDir: Boolean = True): TDataSet;

Sichtbarkeit:

Public

Beschreibung:

Erstellt intern eine neue Query-Instanz, abhängig vom jeweiligen *ConnectionType*. Nachdem die Query-Instanz erzeugt wurde, wird das Ereignis *OnDataSetCreated* aufgerufen, um ggf. spezielle Einstellungen vorzunehmen.
In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

GetNewTable

Notation:

function GetNewTable(pTransaction: TComponent; pActionType: TSFQueryActionType; pTableName, pCatalog, pSchema: String): TDataSet;

Sichtbarkeit:

Public

Beschreibung:

Erstellt intern eine neue Table-Instanz, abhängig vom jeweiligen *ConnectionType*. Nachdem die Query-Instanz erzeugt wurde, wird das Ereignis *OnDataSetCreated* aufgerufen, um ggf. spezielle Einstellungen vorzunehmen.
In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

GetKeyFields

Notation:

function GetKeyFields(pTableName, pCatalog, pSchema: String): String;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt intern die Schlüsselfelder für die angegebene Datenbanktabelle.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

GetFieldNames

Notation:

function GetFieldNames(pTableName, pCatalog, pSchema: String): TStringList;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt intern die Felder/Spalten für die angegebene Datenbanktabelle.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

SetSQLToQuery

Notation:

function SetSQLToQuery(pSQL: String; pDataSet: TDataSet): TCollection;

Sichtbarkeit:

Public

Beschreibung:

Setzt intern den übergebenen SQL-String auf eine interne Query-Instanz.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

QueryExecSQL

Notation:

function QueryExecSQL(pDataSet: TDataSet): LongInt;

Sichtbarkeit:

Public

Beschreibung:

Führt intern die zuvor übergebenen SQL-Anweisung für eine interne Query-Instanz aus.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

SetQueryParamValue

Notation:

procedure SetQueryParamValue(pParam: TCollectionItem; pValue: Variant; pDataType: TFieldType = ftUnknown);

Sichtbarkeit:

Public

Beschreibung:

Setzt intern Parameterwerte für eine Abfrage.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

GetQueryParamName

Notation:

function GetQueryParamName(pParam: TCollectionItem): String;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt intern den Namen eines Abfrageparameters.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

SequenceNameForField

Notation:

function SequenceNameForField(pField: TField): String;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt intern den Namen einer Sequenz für das angegebene Datenbankfeld. Ob die Sequenz ermittelt werden kann, ist abhängig vom angebenen *ConnectionType*.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

GetConnectionDBType

Notation:

function GetConnectionDBType: TSFConnectionDBType;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt intern das verwendete Datenbanksystem anhand der angegebenen *Connection*.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

CheckTransaction

Notation:

function CheckTransaction(pTransaction: TComponent; pSilent: Boolean = False): Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft intern, ob die übergebene Transaktion gültig ist, also ob diese zur verwendeten Verbindung passt. Bspsw. kann bei Verwendung von FireDac auch nur eine TFDTransaction angegebenen werden.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

HasDataSetTransaction

Notation:

function HasDataSetTransaction(pDataSet: TDataSet; pTransaction: TComponent): Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft intern, ob das angegebene DataSet eine Transaktion zugewiesen wurde.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

StartTransactionForDataSet

Notation:

procedure StartTransactionForDataSet(pDataSet: TDataSet);

Sichtbarkeit:

Public

Beschreibung:

Startet intern eine Transaktion für das angegebene DataSet.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

CommitTransactionForDataSet

Notation:

procedure CommitTransactionForDataSet(pDataSet: TDataSet; pRetain: Boolean);

Sichtbarkeit:

Public

Beschreibung:

Führt intern auf die Transaktion des übergebenen DataSets einen Commit aus.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

ActiveTransactionForDataSet

Notation:

function ActiveTransactionForDataSet(pDataSet: TDataSet): Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft intern, ob das übergebenen DataSet eine aktive Transaktion hat.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

CanDBInsertion

Notation:

function CanDBInsertion(pDataSet: TDataSet): Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft intern, ob über das übergebene DataSet Einfügeoperationen möglich sind.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

AddConnectorMsgNotification

Notation:

procedure AddConnectorMsgNotification(pProc: TSFBDSMessageProc);

Sichtbarkeit:

Public

Beschreibung:

Wird intern verwendet, um auf Änderungen am Connector zu reagieren.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

RemoveConnectorMsgNotification

Notation:

procedure RemoveConnectorMsgNotification(pProc: TSFBDSMessageProc);

Sichtbarkeit:

Public

Beschreibung:

Wird intern verwendet, um auf Änderungen am Connector zu reagieren.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

GetCommonConnector

Notation:

class function GetCommonConnector: TSFConnector;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt intern den als *CommonConnector* gekennzeichneten Connector.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

AddCommonConnectedProc

Notation:

class procedure AddCommonConnectedProc(pProc: TSFBDSMessageProc);

Sichtbarkeit:

Public

Beschreibung:

Wird intern verwendet, um auf Änderungen am Connector zu reagieren.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

RemoveCommonConnectedProc

Notation:

class procedure RemoveCommonConnectedProc(pProc: TSFBDSMessageProc);

Sichtbarkeit:

Public

Beschreibung:

Wird intern verwendet, um auf Änderungen am Connector zu reagieren.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion direkt aufzurufen.

Eigenschaften

ConnectionType

Notation:

property ConnectionType: TSFConnectionType read mConnectionType write setConnectionType;

Sichtbarkeit:

Published

Beschreibung:

Der Verbindungstyp, also z. B. FireDac, dbExpress, ADO. Siehe auch TSFConnectionType.

Connection

Notation:

property Connection: TCustomConnection read mConnection write setConnection;

Sichtbarkeit:

Published

Beschreibung:

Die referenzierte Connection, abhängig vom *ConnectionType* (z. B. TFDConnection, TSQLConnection, TADOConnection).

CommonConnector

Notation:

property CommonConnector: Boolean read mCommonConnector write setCommonConnector;

Sichtbarkeit:

Published

Beschreibung:

Gibt an, ob es sich um den globalen Connector handelt. Ein Connector, der mit *CommonConnector* gekennzeichnet wird Instanzen von TSFBusinessData/TSFDataSet automatisch ermittelt.

Zu beachten ist allerdings, dass dieser Connector auch verfügbar sein muss, also vor Zugriff durch eine Instanz von TSFBusinessData/TSFDataSet erzeugt wurde (z. B. dadurch, dass der Connector auf einem automatisch erzeugten Datenmodul platziert wurde).

ConnectionDBType

Notation:

property ConnectionDBType: TSFConnectionDBType read getDBType write mDBType;

Sichtbarkeit:

Published

Beschreibung:

Das verwendete Datenbanksystem, welches über die Connection ermittelt wurde.

In manchen Fällen (z. B. bei ODBC-Verbindungen) kann das Datenbanksystem nicht über die Connection ermittelt werden. Hier muss das verwendete Datenbanksystem dann selbst gesetzt werden. Siehe auch TSFConnectionDBType.

FormatOptions

Notation:

property FormatOptions: TSFBDSFormatOptions read mFormatOptions write setFormatOptions;

Sichtbarkeit:

Published

Beschreibung:

Einstellungen zur Formatierung von Datumswerten, Fließkommawerten, etc. Sind weder im Connector noch direkt in der Instanz von TSFBusinessData/TSFDataSet Formateinstellungen gesetzt, werden die Formateinstellungen der *Connection* ermittelt.

Weitere Informationen zu den Formateinstellungen unter TSFBDSFormatOptions.

Ereignisse

OnDataSetCreated

Notation:

property OnDataSetCreated: TSFConnectorDSCreatedEvt read mOnDataSetCreated write mOnDataSetCreated;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, wenn eine interne Query- oder Table-Instanz erzeugt wird. Siehe auch GetNewQuery und GetNewTable.

TSFCustomBusinessData/TSFBusinessData/TSFDataSet

Beschreibung

TSFBusinessData bildet die Basisklasse für eigene, spezialisierte Klassen zur Kapselung von Businesslogik. Zur Erstellung und Verwendung einer eigenen, spezialisierten BusinessData-Klasse siehe die entsprechenden Themenbeschreibungen.

TSFDataSet ist die Komponente, mit der die gesamte Logik ohne eigene, spezialisierte Klassen verwendet werden kann. Entweder mit Datenverbindung oder als Dataset, welches sich ausschließlich im Speicher befindet.

Zur Verwendung von eigenen, spezialisierten BusinessData-Klassen über den Designer (IDE) ist zu beachten, dass hierfür die Hilfsklassen TSFBusinessDataWrap und TSFBusinessDataWrapSource erforderlich sind.

Index

AddAutoValueForField
AddDynCalcField
AddDynLkpField
AddField
AddRelation
AfterDBDeleteRow
AfterDBEditRow
AfterDBInsertRow
AfterRefreshFull
AfterRefreshRow
AllBaseFieldsToStmt
ApplyUpdates
BeforeDBDeleteRow
BeforeDBEditRow
BeforeDBInsertRow
BeforeRefreshFull
BeforeRefreshRow
CachedUpdates
CancelUpdates
CatalogName
CompareRecords
Connector
ConnectorUsed
DatabaseNameForFieldName
DBTableIdentifier
DeleteByStmtConditions
DeleteDepended
DisableSync
EnableSync

ExchangeRecordPositions
ExplicitSyncRel
FieldNameForDBField
FilterRecord
FormatOptions
FullRefresh
GetAutoValueCls
GetAutoValueForField
GetAutoValueOptionsForDBType
GetBaseTableFields
GetCanSelectWithoutTable
GetCanSubSelectInFrom
GetKeyFields
GetLikeWildcardMany
GetLikeWildcardSingle
GetNameInBaseFieldsList
GetNeedTableOnSubSelectInFrom
GetSupportsLikeEscape
HasDynCalcField
HasDynLkpField
HasPassKeysRel
InitFieldsFromBusinessData
LocateNext
MappedStmtDBDialect
NotifyCurrentRecModified
OnAfterDBDeleteRow
OnAfterDBEditRow
OnAfterDBInsertRow
OnAfterPassKeyToObj
OnAfterRefreshFull
OnAfterRefreshRow
OnAfterSyncRelObj
OnBeforeDBDeleteRow
OnBeforeDBEditRow
OnBeforeDBInsertRow
OnBeforePassKeyToObj
OnBeforeRefreshFull
OnBeforeRefreshRow
OnBeforeSyncRelObj
OnCompareRecords
OnFieldChange
OnGetAutoValCls
OnRecordChange
OnSetParams
ParentRelationDesigner
PassKeysOnCachedUpdates
Prepare
QueryQuoteType
RecalcCalculatedFields
Refilter
RefreshMode
RefreshRelations
RefreshStmtParamValues
RemoveRelation
SchemaName
SelectNameForIdentifier

SetDisableSyncRel
SetPassKeysRel
SetQueryParams
SortBuffer
Stmt
StmtParamValues
SyncDisabled
TableName
Transaction
UpdateMode
UpdatesPending
UpdateTransaction

Funktionen

GetKeyFields

Notation:

function GetKeyFields: String; virtual;

Sichtbarkeit:

Protected

Beschreibung:

Ermittelt die Schlüsselfelder (primary keys) der - dem DataSet zugrunde liegenden - Tabelle.

GetBaseTableFields

Notation:

function GetBaseTableFields: TStringList; overload; virtual;

function GetBaseTableFields(pTableName, pSchemaName, pCatalogName: String): TStringList; overload;

function GetBaseTableFields(pStmtTable: TSFStmtTable): TStringList; overload;

Sichtbarkeit:

Protected

Beschreibung:

Ermittelt die Tabellenfelder der - dem DataSet zugrunde liegenden - Tabelle.

GetNameInBaseFieldsList

Notation:

function GetNameInBaseFieldsList(pName: String; pList: TStringList): Boolean;

Sichtbarkeit:

Protected

Beschreibung:

Prüft, ob der übergebene (Feld-)Name in der übergebenen Liste vorhanden ist. Die Prüfroutine berücksichtigt hierbei auch, dass der Name ggf. in Anführungszeichen (reguläre oder abhängig vom verwendeten Datenbanksystem) gesetzt ist.

NotifyCurrentRecModified

Notation:

procedure NotifyCurrentRecModified;

Sichtbarkeit:

Protected

Beschreibung:

Wird intern dazu verwendet, dem DataSet mitzuteilen, dass Daten extern geändert wurde.

In Anwendungen sollte es nicht erforderlich sein, diese Funktion aufzurufen.

GetAutoValueCls

Notation:

function GetAutoValueCls(pFieldName: String; pAutoDetected: Boolean): TSFBDSAutoValueGeneratorCls; virtual;

Sichtbarkeit:

Protected

Beschreibung:

Über diese Funktion wird die Klasse ermittelt, die zur Generierung von AutoInc-Feldern zuständig ist. Der Parameter *pFieldName* gibt den Namen des Feldes an, der Parameter *pAutoDetected*, ob das Feld automatisch ermittelt wurde oder manuell hinzugefügt wurde.

Siehe auch TSFBDSAutoValueGenerator, AddAutoValueForField, GetAutoValueForField

GetAutoValueOptionsForDBType

Notation:

function GetAutoValueOptionsForDBType(pDBType: TSFConnectionDBType; pMode: TSFBDSAutoValueGetMode): TSFBDSAutoValueOptions; virtual;

Sichtbarkeit:

Protected

Beschreibung:

Ermittelt, abhängig vom verwendeten Datenbanksystem, die Optionen zur Generierung von AutoInc-Werten.

Siehe auch TSFBDSAutoValueGenerator, TSFBDSAutoValueGetMode, TSFBDSAutoValueOption, TSFBDSAutoValueOptions

BeforeDBEditRow

Notation:

procedure BeforeDBEditRow; virtual;

Sichtbarkeit:

Protected

Beschreibung:

Wird aufgerufen, bevor Änderungen in die Datenbank geschrieben werden, also das interne Update-Statement ausgeführt wird.

Siehe auch OnBeforeDBEditRow

AfterDBEditRow

Notation:

procedure AfterDBEditRow; virtual;

Sichtbarkeit:

Protected

Beschreibung:

Wird aufgerufen, nachdem Änderungen in die Datenbank geschrieben wurden, also das interne Update-Statement ausgeführt wurde.

Siehe auch OnAfterDBEditRow

BeforeDBInsertRow

Notation:

procedure BeforeDBInsertRow; virtual;

Sichtbarkeit:

Protected

Beschreibung:

Wird aufgerufen, bevor eine neuer Datensatz in die Datenbank geschrieben wird, also das interne Insert-Statement ausgeführt wird.

Siehe auch OnBeforeDBInsertRow

AfterDBInsertRow

Notation:

procedure AfterDBInsertRow; virtual;

Sichtbarkeit:

Protected

Beschreibung:

Wird aufgerufen, nachdem ein neuer Datensatz in die Datenbank geschrieben wurden, also das interne Insert-Statement ausgeführt wurde.

Siehe auch OnAfterDBInsertRow

BeforeDBDeleteRow

Notation:

procedure BeforeDBDeleteRow; virtual;

Sichtbarkeit:

Protected

Beschreibung:

Wird aufgerufen, bevor ein Datensatz aus der Datenbank gelöscht wird, also das interne Delete-Statement ausgeführt wird.

Siehe auch OnBeforeDBDeleteRow

AfterDBDeleteRow

Notation:

procedure AfterDBDeleteRow; virtual;

Sichtbarkeit:

Protected

Beschreibung:

Wird aufgerufen, nachdem ein Datensatz aus der Datenbank gelöscht wurde, also das interne Delete-Statement ausgeführt wurde.

Siehe auch OnAfterDBDeleteRow

BeforeRefreshRow

Notation:

procedure BeforeRefreshRow; virtual;

Sichtbarkeit:

Protected

Beschreibung:

Wird aufgerufen, bevor ein Datensatz aktualisiert wird, also das interne Refresh-Statement ausgeführt wird.

Siehe auch OnBeforeRefreshRow

AfterRefreshRow

Notation:

procedure AfterRefreshRow; virtual;

Sichtbarkeit:

Protected

Beschreibung:

Wird aufgerufen, nachdem ein Datensatz aktualisiert wurde, also das interne Refresh-Statement ausgeführt wurde.

Siehe auch OnAfterRefreshRow

BeforeRefreshFull

Notation:

procedure BeforeRefreshFull; virtual;

Sichtbarkeit:

Protected

Beschreibung:

Wird aufgerufen, bevor die Datenmenge komplett aktualisiert wird, also das interne Refresh-Statement ausgeführt wird.

Siehe auch OnBeforeRefreshFull

AfterRefreshFull

Notation:

procedure AfterRefreshFull; virtual;

Sichtbarkeit:

Protected

Beschreibung:

Wird aufgerufen, nachdem die Datenmenge komplett aktualisiert wurde, also das interne Refresh-Statement ausgeführt wurde.

Siehe auch OnAfterRefreshFull

FilterRecord

Notation:

procedure FilterRecord(var pAccept: Boolean); virtual;

Sichtbarkeit:

Protected

Beschreibung:

Funktion zur Filterung von Datensätzen. Der Parameter *pAccept* gibt an, ob der Datensatz ausgefiltert werden soll (true = Datensatz wird angezeigt, false = Datensatz wird nicht angezeigt).

Siehe auch TDataSet.Filtered, TDataSet.OnFilterRecord, Refilter

CompareRecords

Notation:

function CompareRecords(CompareRecordFrom, CompareRecordTo: TSFBDSCompareRecord): TSFBDSRecordCompareResult; virtual;

Sichtbarkeit:

Protected

Beschreibung:

Wird bei Sortierung der Datenmenge (Sortierung des Datensatzpuffers) aufgerufen, um einzelne Datensätze zu vergleichen. Wenn die Funktion SortBuffer verwendet wird, muss diese Funktion überschrieben oder das dazugehörige Event OnCompareRecords behandelt werden.

Siehe auch SortBuffer, OnCompareRecords, TSFBDSCompareRecord

SetQueryParams

Notation:

procedure SetQueryParams(pType: TSFBDSExecParamsType; pParams: TCollection); virtual;

Sichtbarkeit:

Protected

Beschreibung:

Über diese Funktion können Parameter einer SQL-Abfrage zur Laufzeit mit Werten belegt werden. Der Parameter *pType* gibt die Art der SQL-Abfrage an, der Parameter *pParams* ist die Liste mit Parametern. Der Typ von *pParams* ist abhängig von der verwendeten Datenbankverbindung, z. B. bei FireDac ist der Typ TFDParams.

Siehe auch OnSetParams, StmtParamValues, TSFBDSExecParamsType

MappedStmtDBDialect

Notation:

function MappedStmtDBDialect: TSFStmtDBDialect;

Sichtbarkeit:

Protected

Beschreibung:

Ermittelt den SQL-Typ des Abfragegenerators anhand des Datenbanktyps des Connectors.

LocateNext

Notation:

function LocateNext(const KeyFields: string; const KeyValues: Variant;

Options: TLocateOptions): Boolean;

Sichtbarkeit:

Public

Beschreibung:

Positioniert die Datenmenge auf dem nächsten Datensatz, der die angegebenen Suchkritieren (*KeyFields* = Suchfelder; *KeyValues* = Suchwerte) erfüllt. Im Gegensatz zu Locate beginnt die Suche dabei beim aktuellen Datensatz (und nicht beim ersten Datensatz).

Siehe auch TDataSet.Locate

Prepare

Notation:

procedure Prepare;

Sichtbarkeit:

Public

Beschreibung:

Bereitet die interne Abfrage zur Ausführung vor und erzeugt und die Felddefinitionen.

SortBuffer

Notation:

procedure SortBuffer;

Sichtbarkeit:

Public

Beschreibung:

Sortiert die Datenmenge bzw. den Datensatzpuffer, d. h. es wird keine neue Abfrage an die Datenbank gesandt. Zur Sortierung wird ein QuickSort-Algorithmus verwendet, der Vergleich einzelner Datensätze erfolgt über die Funktion CompareRecords oder das Ereignis OnCompareRecords.

ApplyUpdates

Notation:

procedure ApplyUpdates;

Sichtbarkeit:

Public

Beschreibung:

Schreibt gepufferte Änderungen in die Datenbank.

Siehe auch CachedUpdates, CancelUpdates, UpdatesPending.

CancelUpdates

Notation:

procedure CancelUpdates;

Sichtbarkeit:

Public

Beschreibung:

Verwirft gepufferte Änderungen.

Siehe auch ApplyUpdates, CachedUpdates, UpdatesPending.

FullRefresh

Notation:

procedure FullRefresh;

Sichtbarkeit:

Public

Beschreibung:

Aktualisiert die Datenmenge, indem Sie diese neu von der Datenbank abruft.

Refilter

Notation:

procedure Refilter;

Sichtbarkeit:

Public

Beschreibung:

Aktualisiert den Filter, indem sämtliche Datensätze neu überprüft werden.

Siehe auch FilterRecord, TDataSet.OnFilterRecord

AddAutoValueForField

Notation:

function AddAutoValueForField(pFieldName: String; pAutoValueClass: TSFBDSAutoValueGeneratorCls = nil): TSFBDSAutoValueGenerator;

Sichtbarkeit:

Public

Beschreibung:

Rufen Sie diese Funktion auf, um ein Feld als AutoWert zu definieren. Der Parameter *pFieldName* ist der Name des gewünschten Feldes, *pAutoValueClass* gibt die Klasse an, die für die Generierung der Werte zuständig ist. Wird der Parameter *pAutoValueClass* nicht übergeben, wird der Wert von der Standardklasse TSFBDSAutoValueGenerator generiert.

Siehe auch GetAutoValueCls

GetAutoValueForField

Notation:

function GetAutoValueForField(pFieldName: String): TSFBDSAutoValueGenerator;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt die Instanz, die zur Generierung des Autowerts für den übergebenen Feldnamen hinterlegt ist.

Siehe auch TSFBDSAutoValueGenerator, AddAutoValueForField, GetAutoValueCls

AddField

Notation:

function AddField(pFieldName: String; pDataType: TFieldType; pSize: Integer; pPrecision: Integer = 0; pRequired: Boolean = False; pReadOnly: Boolean = False): TField;

Sichtbarkeit:

Public

Beschreibung:

Fügt der Datenmenge programmatisch ein Feld hinzu. Dies ist insbesondere dann erforderlich, wenn die Datenmenge als rein gepufferte Datenmenge, also ohne Datenverbindung bzw. ohne referenzierte Tabelle und Abfrage verwendet wird.

Wenn Sie einer (rein gepufferten) Datenmenge Felder anhand einer Tabelle/einer spezialisierten BusinessData-Klasse hinzufügen wollen, verwenden Sie InitFieldsFromBusinessData.

InitFieldsFromBusinessData

Notation:

procedure InitFieldsFromBusinessData(pTabObjName: String; pCatalog: String = ''; pSchema: String = ''; pPreventAutoValues: Boolean = False; pPreventReadOnly: Boolean = False; pPreventRequired: Boolean = False); overload;

procedure InitFieldsFromBusinessData(pObj: TSFCustomBusinessData; pPreventAutoValues: Boolean = False; pPreventReadOnly: Boolean = False; pPreventRequired: Boolean = False); overload;

Sichtbarkeit:

Public

Beschreibung:

Fügt einer (rein gepufferten) Datenmenge programmatisch Felder anhand einer Tabelle/einer spezialisierten BusinessData-Klasse hinzu.

Siehe auch AddField

AllBaseFieldsToStmt

Notation:

procedure AllBaseFieldsToStmt(pOnlySearch: Boolean = False);

Sichtbarkeit:

Public

Beschreibung:

Fügt alle Felder der - einer BusinessData-Klasse zugrunde liegenden - Tabelle dem Abfragegenerator hinzu. Der Parameter pOnlySearch gibt, ob die Felder dem Abfragegenerator als Suchfelder hinzugefügt werden sollen, die nicht in der Select-Klausel aufgeführt werden.

AddDynCalcField

Notation:

procedure AddDynCalcField(pFieldName: String; pDataType: TFieldType; pSize: Integer; pPrecision: Integer = 0);

Sichtbarkeit:

Public

Beschreibung:

Fügt der Datenmenge programmatisch eine Definition für ein berechnetes Feld hinzu. Das Feld wird erst beim Öffnen der Datenmenge erstellt.

AddDynLkpField

Notation:

procedure AddDynLkpField(pFieldName: String; pDataType: TFieldType; pLkpDs: TDataSet; pKeyFlds, pLkpKeyFlds, pLkpRsltFld: String; pCached: Boolean; pSize: Integer; pPrecision: Integer = 0);

Sichtbarkeit:

Public

Beschreibung:

Fügt der Datenmenge programmatisch eine Definition für ein Lookup-Feld hinzu. Das Feld wird erst beim Öffnen der Datenmenge erstellt.

HasDynCalcField

Notation:

function HasDynCalcField(pFieldName: String): Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft, ob die Datenmenge programmatisch hinzugefügte Definition für berechnete Felder hat.

HasDynLkpField

Notation:

function HasDynLkpField(pFieldName: String): Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft, ob die Datenmenge programmatisch hinzugefügte Definition für Lookup-Felder hat.

RecalcCalculatedFields

Notation:

procedure RecalcCalculatedFields;

Sichtbarkeit:

Public

Beschreibung:

Erzwingt eine Neuberechnung von berechneten Feldern.

DatabaseNameForFieldName

Notation:

function DatabaseNameForFieldName(pFieldName: String; var pTableAlias, pTableName, pTableSchema, pTableCatalog, pAttrName: String): Boolean;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt im Abfragegenerator die Quelle (Tabelle, Feld, usw.) für den übergebenen Feldnamen (Name des Feldes im DataSet). Bei Erfolg gibt die Funktion true zurück, die Ergebnisdaten werden in die var-Parameter geschrieben.

Diese Funktion kann insbesondere dann sinnvoll sein, wenn in der Select-Klausel Aliase verwendet werden.

SelectNameForIdentifier

Notation:

function SelectNameForIdentifier(pIdentifier: String; var pTableAlias, pTableName, pTableSchema, pTableCatalog, pAttrName: String): String;

Sichtbarkeit:

Public

Beschreibung:

Diese Funktion ermittelt im Abfragegenerator den Anzeigenamen für den übergebenen Identifier. Der Identifier ist der Feldname einer Datenbanktabelle oder ein Alias.

Gibt im Erfolgsfall den Anzeigename des Identifiers zurück, die Quelle (Tabellenalias, Tabellenname, usw.) werden in die var-Parameter geschrieben.

FieldNameForDBField

Notation:

function FieldNameForDBField(pDBFieldName: String; pOnlyBaseFields: Boolean): String;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt den Feldnamen im DataSet für den übergebenen Feldnamen einer Datenbanktabelle. Der Parameter pOnlyBaseFields gibt an, ob nur die Felder der Basistabelle durchsucht werden.

GetLikeWildcardSingle

Notation:

function GetLikeWildcardSingle: String;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt anhand des Abfragegenerators das Wildcard-Zeichen für eine Like-Suche im verwendeten Datenbanksystem. I. d. R. ist dieses Zeichen "_".

GetLikeWildcardMany

Notation:

function GetLikeWildcardMany: String;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt anhand des Abfragegenerators das Wildcard-Zeichen für eine Like-Suche im verwendeten Datenbanksystem. I. d. R. ist dieses Zeichen "%".

GetSupportsLikeEscape

Notation:

function GetSupportsLikeEscape: Boolean;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt anhand des Abfragegenerators, ob das verwendete Datenbanksystem die ESCAPE-Anweisung bei einer Like-Suche unterstützt. Durch die ESCAPE-Anweisung kann auch nach Zeichen gesucht werden, die dem Wildcard-Zeichen entsprechen.

ExchangeRecordPositions

Notation:

procedure ExchangeRecordPositions(pFrom, pTo: Integer);

Sichtbarkeit:

Public

Beschreibung:

Tauscht die Position von 2 Datensätzen innerhalb der Datenmenge

RefreshStmtParamValues

Notation:

procedure RefreshStmtParamValues;

Sichtbarkeit:

Public

Beschreibung:

Aktualisiert die Defitionen der Abfrageparameter anhand des Abfragegenerators.

Siehe auch StmtParamValues.

GetCanSelectWithoutTable

Notation:

function GetCanSelectWithoutTable(var pDummyTable: String): Boolean;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt anhand des Abfragegenerators, ob das verwendete Datenbanksystem Select-Anweisungen ohne Angabe einer Tabelle unterstützt.

GetCanSubSelectInFrom

Notation:

function GetCanSubSelectInFrom: Boolean;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt anhand des Abfragegenerators, ob das verwendete Datenbanksystem Select-Anweisungen mit einem Subselect in der From-Klausel unterstützt.

GetNeedTableOnSubSelectInFrom

Notation:

function GetNeedTableOnSubSelectInFrom: Boolean;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt anhand des Abfragegenerators, ob das verwendete Datenbanksystem Select-Anweisungen mit einem Subselect in der From-Klausel unterstützt, der (der Subselect) keine Tabelle referenziert.

AddRelation

Notation:

procedure AddRelation(pDestObj: TSFBusinessData; pSourceAttrs, pDestAttrs: Variant; pPassKeys: Boolean = False); overload;

procedure AddRelation(pDestObj: TSFBusinessData; pSourceAttrs, pDestAttrs: String; pPassKeys: Boolean = False); overload;

Sichtbarkeit:

Public

Beschreibung:

Fügt eine Master-Detail-Beziehung hinzu. Der Parameter *pPassKeys* gibt an, ob Änderungen an einem der Schlüsselfelder im Master-Objekt an das Detail-Objekt weitergegeben werden.

Die Beziehungsfelder (*pSourceAttrs*, *pDestAttrs*) können als VarArray oder als String (einzelne Feldangaben mit Semikolon trennen) angegeben werden. Als Feldbezeichner können Namen von Datentabellenfeldern oder Feldnamen im DataSet übergeben werden.

Siehe auch RemoveRelation, RefreshRelations, DisableSync, EnableSync

RemoveRelation

Notation:

procedure RemoveRelation(pDestObj: TSFBusinessData);

Sichtbarkeit:

Public

Beschreibung:

Löscht die Master-Detail-Beziehung für das angegebene Objekt.

Siehe auch AddRelation, RefreshRelations, DisableSync, EnableSync

RefreshRelations

Notation:

procedure RefreshRelations;

Sichtbarkeit:

Public

Beschreibung:

Aktualisiert alle Master-Detail-Beziehungen, sofern erforderlich.

Siehe auch AddRelation, RemoveRelation, DisableSync, EnableSync, ExplicitSyncRel

DisableSync

Notation:

procedure DisableSync;

Sichtbarkeit:

Public

Beschreibung:

Deaktiviert die Synchronisation von Detail-Beziehungen.

Siehe auch AddRelation, RemoveRelation, EnableSync, RefreshRelations, ExplicitSyncRel

EnableSync

Notation:

procedure EnableSync;

Sichtbarkeit:

Public

Beschreibung:

Aktiviert die Synchronisation von Detail-Beziehungen.

Siehe auch AddRelation, RemoveRelation, DisableSync, RefreshRelations, ExplicitSyncRel

SyncDisabled

Notation:

function SyncDisabled: Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft, ob die Synchronisation von Detail-Beziehungen deaktiviert ist.

Siehe auch AddRelation, RemoveRelation, EnableSync, DisableSync, RefreshRelations, ExplicitSyncRel

SetDisableSyncRel

Notation:

procedure SetDisableSyncRel(pObj: TSFBusinessData; pDisabled: Boolean);

Sichtbarkeit:

Public

Beschreibung:

Aktiviert/Deaktiviert die Synchronistion der Detail-Beziehung für das angegebene Objekt.

Siehe auch AddRelation, RemoveRelation, EnableSync, DisableSync, RefreshRelations, ExplicitSyncRel

ExplicitSyncRel

Notation:

procedure ExplicitSyncRel(pObj: TSFBusinessData);

Sichtbarkeit:

Public

Beschreibung:

Aktualisiert die Detail-Beziehung für das angegebene Objekt, sofern erforderlich.

Siehe auch RefreshRelations

SetPassKeysRel

Notation:

procedure SetPassKeysRel(pObj: TSFBusinessData; pPassKeys: Boolean);

Sichtbarkeit:

Public

Beschreibung:

Setzt *PassKeys* der Detail-Beziehung für das angegebene Objekt. Wenn *PassKeys* gesetzt ist, werden Datenänderungen an den Schlüsselfeldern des Master-Objekts an an das Detail-Objekt weitergegeben.

Siehe auch AddRelation

HasPassKeysRel

Notation:

function HasPassKeysRel: Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft, ob PassKeys an einer oder mehrern Detail-Beziehungen gesetzt ist.

Siehe auch AddRelation, SetPassKeysRel

DeleteByStmtConditions

Notation:

procedure DeleteByStmtConditions(pParamValues: Variant; pWithRefresh: Boolean);

Sichtbarkeit:

Public

Beschreibung:

Führt ein DELETE-Statment anhand des Suchcontainers im Abfragegenerator (Stmt) aus. Der Parameter *pParamValues* gibt evtl. erforderliche Abfrageparameter an (als VarArray), ist der Parameter *pWithRefresh* gesetzt, wird die Datenmenge nach Ausführung aktualisiert.

DeleteDepended

Notation:

procedure DeleteDepended(pTableName, pCatalog, pSchema, pSrcAttr, pDestAttr: String); overload;

procedure DeleteDepended(pTableName, pCatalog, pSchema, pDestAttr: String; pSrcVal: Variant); overload;

Sichtbarkeit:

Public

Beschreibung:

Führt ein DELETE-Statement auf eine abhängige (in Relation stehende) Tabelle durch.

Eigenschaften

TableName

Notation:

property TableName: String read mTableName write setTableName;

Sichtbarkeit:

Protected/Published (TSFDataSet)

Beschreibung:

Gibt den Namen der zugrunde liegenden Datenbanktabelle an.

Siehe auch CatalogName, SchemaName

CatalogName

Notation:

property CatalogName: String read mCatalogName write setCatalogName;

Sichtbarkeit:

Protected/Published (TSFDataSet)

Beschreibung:

Gibt den Katalog der zugrunde liegenden Datenbanktabelle an.

Siehe auch TableName, SchemaName

SchemaName

Notation:

property SchemaName: String read mSchemaName write setSchemaName;

Sichtbarkeit:

Protected/Published (TSFDataSet)

Beschreibung:

Gibt das Schema der zugrunde liegenden Datenbanktabelle an.

Siehe auch TableName, CatalogName

QueryQuoteType

Notation:

property QueryQuoteType: TSFBDSQuoteType read getQueryQuoteType;

Sichtbarkeit:

Protected

Beschreibung:

Gibt an, ob Identifier bei Generierung von Datenbankabfragen in Anführungszeichen gesetzt werden.

Siehe auch TSFBDSFormatOptions.QuoteType

Connector

Notation:

property Connector: TSFConnector read mConnector write setConnector;

Sichtbarkeit:

Published

Beschreibung:

Der zugewiesene Connector. Wenn Sie einen *CommonConnector* verwenden, kann diese Angabe leer sein.

ConnectorUsed

Notation:

property ConnectorUsed: TSFConnector read getConnectorUsed;

Sichtbarkeit:

Public

Beschreibung:

Der verwendete Connector.

DBTableIdentifier

Notation:

property DBTableIdentifier: String read getDBTableIdentifier;

Sichtbarkeit:

Public

Beschreibung:

Der anhand *TableName*, *SchemaName* und *CatalogName* generierte Identifier.

UpdateMode

Notation:

property UpdateMode: TUpdateMode read mUpdateMode write mUpdateMode;

Sichtbarkeit:

Published

Beschreibung:

Gibt an, nach welchen Feldern in internen UPDATE- und DELETE-Anweisungen gesucht wird - nur nach Schlüsselfeldern oder nach allen Feldern.

Siehe auch TDataSet.Edit, TDataSet.Post, TDataSet.Delete

CachedUpdates

Notation:

property CachedUpdates: Boolean read mCachedUpdates write setCachedUpdates;

Sichtbarkeit:

Published

Beschreibung:

Gibt an, ob Änderungen am DataSet gepuffert werden.

Siehe auch ApplyUpdates, CancelUpdates, UpdatesPending

UpdatesPending

Notation:

property UpdatesPending: Boolean read mUpdatesPending;

Sichtbarkeit:

Public

Beschreibung:

Gibt an, ob gepufferte Änderungen vorhanden sind.

Siehe auch ApplyUpdates, CancelUpdates, CachedUpdates

RefreshMode

Notation:

property RefreshMode: TSFBDSRefreshMode read mRefreshMode write mRefreshMode;

Sichtbarkeit:

Published

Beschreibung:

Gibt an, ob beim Aufruf der Methode *Refresh* die gesamte Datenmenge oder nur der aktuelle Datensatz aktualisiert wird.

Siehe auch TSFBDSRefreshMode

Stmt

Notation:

property Stmt: TSFStmt read mStmt;

Sichtbarkeit:

Published

Beschreibung:

Referenz auf den internen Abfragegenerator

StmtParamValues

Notation:

property StmtParamValues: TCollection read getStmtParamValues write setStmtParamValues;

Sichtbarkeit:

Published

Beschreibung:

Aus dem internen Abfragegenerator erzeugte Defintionen für Abfrageparameter.

Siehe auch RefreshStmtParamValues

Transaction

Notation:

property Transaction: TComponent read mTransaction write setTransaction;

Sichtbarkeit:

Published

Beschreibung:

Verweis auf ein Transaktionsobjekt für SELECT-Anweisungen. Das Transaktionsobjekt muss zum verwendeten ConnectionType des verwendeten Connectors passen. D. h. wenn Sie z. B. mit FireDac arbeiten, muss die Transaktion vom Typ TFDTransaction sein.

Siehe auch TSFConnector.CheckTransaction, UpdateTransaction

FormatOptions

Notation:

property FormatOptions: TSFBDSFormatOptions read mFormatOptions write setFormatOptions;

Sichtbarkeit:

Published

Beschreibung:

Einstellungen zur Formatierung von Datumswerten, Fließkommawerten, etc. Sind weder im Connector noch direkt in der Instanz von TSFBusinessData/TSFDataSet Formateinstellungen gesetzt, werden die Formateinstellungen der *Connection* ermittelt.

Weitere Informationen zu den Formateinstellungen unter TSFBDSFormatOptions.

UpdateTransaction

Notation:

property UpdateTransaction: TComponent read mUpdateTransaction write setUpdateTransaction;

Sichtbarkeit:

Published

Beschreibung:

Verweis auf ein Transaktionsobjekt für UPDATE-, INSERT- und DELETE-Anweisungen. Das Transaktionsobjekt muss zum verwendeten ConnectionType des verwendeten Connectors passen. D. h. wenn Sie z. B. mit FireDac arbeiten, muss die Transaktion vom Typ TFDTransaction sein.

Siehe auch TSFConnector.CheckTransaction, Transaction

ParentRelationDesigner

Notation:

property ParentRelationDesigner: TSFBusinessDataRelationDesigner read mParentRelationDesigner;

Sichtbarkeit:

Published

Beschreibung:

Instanz zur Definition von Master-Detail-Beziehungen zur Designzeit.

Siehe auch AddRelation, RemoveRelation, RefreshRelations

PassKeysOnCachedUpdates

Notation:

property PassKeysOnCachedUpdates: Boolean read mPassKeysOnCachedUpdates write mPassKeysOnCachedUpdates;

Sichtbarkeit:

Published

Beschreibung:

Gibt an, ob Datenänderungen an Schlüsselfeldern auch dann an Detail-Beziehungen weitergegeben werden, wenn Datenänderungen gepuffert werden.

Siehe auch AddRelation, SetPassKeysRel

Ereignisse

OnBeforeDBEditRow

Notation:

property OnBeforeDBEditRow: TDataSetNotifyEvent read mOnBeforeDBEditRow write mOnBeforeDBEditRow;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, bevor Änderungen in die Datenbank geschrieben werden, also das interne Update-Statement ausgeführt wird.

Siehe auch BeforeDBEditRow

OnAfterDBEditRow

Notation:

property OnAfterDBEditRow: TDataSetNotifyEvent read mOnAfterDBEditRow write mOnAfterDBEditRow;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, nachdem Änderungen in die Datenbank geschrieben wurden, also das interne Update-Statement ausgeführt wurde.

Siehe auch AfterDBEditRow

OnBeforeDBInsertRow

Notation:

property OnBeforeDBInsertRow: TDataSetNotifyEvent read mOnBeforeDBInsertRow write mOnBeforeDBInsertRow;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, bevor eine neuer Datensatz in die Datenbank geschrieben wird, also das interne Insert-Statement ausgeführt wird.

Siehe auch BeforeDBInsertRow

OnAfterDBInsertRow

Notation:

property OnAfterDBInsertRow: TDataSetNotifyEvent read mOnAfterDBInsertRow write mOnAfterDBInsertRow;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, nachdem ein neuer Datensatz in die Datenbank geschrieben wurden, also das interne Insert-Statement ausgeführt wurde.

Siehe auch AfterDBInsertRow

OnBeforeDBDeleteRow

Notation:

property OnBeforeDBDeleteRow: TDataSetNotifyEvent read mOnBeforeDBDeleteRow write mOnBeforeDBDeleteRow;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, bevor ein Datensatz aus der Datenbank gelöscht wird, also das interne Delete-Statement ausgeführt wird.

Siehe auch BeforeDBDeleteRow

OnAfterDBDeleteRow

Notation:

property OnAfterDBDeleteRow: TDataSetNotifyEvent read mOnAfterDBDeleteRow write mOnAfterDBDeleteRow;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, nachdem ein Datensatz aus der Datenbank gelöscht wurde, also das interne Delete-Statement ausgeführt wurde.

Siehe auch AfterDBDeleteRow

OnBeforeRefreshRow

Notation:

property OnBeforeRefreshRow: TDataSetNotifyEvent read mOnBeforeRefreshRow write mOnBeforeRefreshRow;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, bevor ein Datensatz aktualisiert wird, also das interne Refresh-Statement ausgeführt wird.

Siehe auch BeforeRefreshRow

OnAfterRefreshRow

Notation:

property OnAfterRefreshRow: TDataSetNotifyEvent read mOnBeforeRefreshRow write mOnBeforeRefreshRow;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, nachdem ein Datensatz aktualisiert wurde, also das interne Refresh-Statement ausgeführt wurde.

Siehe auch AfterRefreshRow

OnBeforeRefreshFull

Notation:

property OnBeforeRefreshFull: TDataSetNotifyEvent read mOnBeforeRefreshFull write mOnBeforeRefreshFull;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, bevor die Datenmenge komplett aktualisiert wird, also das interne Refresh-Statement ausgeführt wird.

Siehe auch BeforeRefreshFull

OnAfterRefreshFull

Notation:

property OnAfterRefreshFull: TDataSetNotifyEvent read mOnAfterRefreshFull write mOnAfterRefreshFull;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, nachdem die Datenmenge komplett aktualisiert wurde, also das interne Refresh-Statement ausgeführt wurde.

Siehe auch AfterRefreshFull

OnSetParams

Notation:

property OnSetParams: TSFBDSSetParamsEvt read mOnSetParams write mOnSetParams;

Sichtbarkeit:

Published

Beschreibung:

Über dieses Ereignis können Parameter einer SQL-Abfrage zur Laufzeit mit Werten belegt werden. Der Parameter *pType* gibt die Art der SQL-Abfrage an, der Parameter *pParams* ist die Liste mit Parametern. Der Typ von *pParams* ist abhängig von der verwendeten Datenbankverbindung, z. B. bei FireDac ist der Typ TFDParams.

Siehe auch SetQueryParams, StmtParamValues, TSFBDSExecParamsType, TSFBDSSetParamsEvt

OnCompareRecords

Notation:

property OnCompareRecords: TSFBSDRecordCompareEvent read mOnCompareRecords write mOnCompareRecords;

Sichtbarkeit:

Published

Beschreibung:

Wird bei Sortierung der Datenmenge (Sortierung des Datensatzpuffers) ausgelöst, um einzelne Datensätze zu vergleichen. Wenn die Funktion SortBuffer verwendet wird, muss dieses Ereignis behandelt oder die Funktion CompareRecords überschrieben werden.

Siehe auch SortBuffer, CompareRecords, TSFBDSCompareRecord, TSFBSDRecordCompareEvent

OnGetAutoValCls

Notation:

property OnGetAutoValCls: TSFBDSGetAutoValueCls read mOnGetAutoValCls write mOnGetAutoValCls;

Sichtbarkeit:

Public

Beschreibung:

Über dieses Ereignis wird die Klasse ermittelt, die zur Generierung von AutoInc-Feldern zuständig ist. Der Parameter *pFieldName* gibt den Namen des Feldes an, der Parameter *pAutoDetected*, ob das Feld automatisch ermittelt wurde oder manuell hinzugefügt wurde.

Siehe auch TSFBDSAutoValueGenerator, AddAutoValueForField, GetAutoValueForField, GetAutoValueCls, TSFBDSGetAutoValueCls

OnFieldChange

Notation:

property OnFieldChange: TDataChangeEvent read mOnFieldChange write mOnFieldChange;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, wenn sich der Wert eines Feldes ändert

OnRecordChange

Notation:

property OnRecordChange: TDataSetNotifyEvent read mOnRecordChange write mOnRecordChange;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, wenn sich der aktuelle Datensatz ändert

OnBeforeSyncRelObj

Notation:

property OnBeforeSyncRelObj: TDataSetNotifyEvent read mOnBeforeSyncRelObj write mOnBeforeSyncRelObj;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, bevor das Objekt einer Detail-Beziehung synchronisiert wird.

Siehe auch AddRelation, RefreshRelations, ExplicitSyncRel

OnAfterSyncRelObj

Notation:

property OnAfterSyncRelObj: TDataSetNotifyEvent read mOnAfterSyncRelObj write mOnAfterSyncRelObj;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, nachdem das Objekt einer Detail-Beziehung synchronisiert wurde.

Siehe auch AddRelation, RefreshRelations, ExplicitSyncRel

OnBeforePassKeyToObj

Notation:

property OnBeforePassKeyToObj: TDataSetNotifyEvent read mOnBeforePassKeyToObj write mOnBeforePassKeyToObj;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, bevor Datenänderungen an Schlüsselattributen der Master-Datenmenge an die angegebene Detail-Datenmenge weitergereicht wird.

Siehe auch AddRelation, SetPassKeysRel, HasPassKeysRel

OnAfterPassKeyToObj

Notation:

property OnAfterPassKeyToObj: TDataSetNotifyEvent read mOnAfterPassKeyToObj write mOnAfterPassKeyToObj;

Sichtbarkeit:

Published

Beschreibung:

Wird ausgelöst, nachdem Datenänderungen an Schlüsselattributen der Master-Datenmenge an die angegebene Detail-Datenmenge weitergereicht wurden.

Siehe auch AddRelation, SetPassKeysRel, HasPassKeysRel

TSFBDSAutoValueGenerator

Beschreibung

TSFBDSAutoValueGenerator ist die Klasse, die für die Generierung von Autowerten zuständig ist.

Wenn eine eigene Logik zur Generierung von Autowerten erforderlich ist, muss eine Klasse von TSFBDSAutoValueGenerator abgeleitet werden und in TSFBusinessDataSet.GetAutoValueCls oder TSFBusinessDataSet.OnGetAutoValCls übergeben werden.

Index

AutoDetected
DataSet
ExplicitInsertByDBMS
FieldName
GetGeneratorValue
Options
SequenceName

Funktionen

GetGeneratorValue

Notation:

function GetGeneratorValue(pMode: TSFBDSAutoValueGetMode): Variant; virtual;

Sichtbarkeit:

Protected

Beschreibung:

Ermittelt den nächsten Autowert.

Siehe auch TSFBDSAutoValueGetMode

Eigenschaften

SequenceName

Notation:

property SequenceName: String read mSequenceName write mSequenceName;

Sichtbarkeit:

Public

Beschreibung:

Wenn eine Sequenz bzw. ein Generator verwendet wird, kann hier der Name dieser Sequenz bzw. des Generators angegeben werden.

Abhängig vom Verbindungstyp können Sequenzen auch intern ermittelt werden.

DataSet

Notation:

property DataSet: TSFCustomBusinessData read mDataSet;

Sichtbarkeit:

Public

Beschreibung:

Referenz auf das DataSet, für welchen der Autowert generiert wird.

FieldName

Notation:

property FieldName: String read mFieldName;

Sichtbarkeit:

Public

Beschreibung:

Feldname, für welchen der Autowert generiert wird.

AutoDetected

Notation:

property AutoDetected: Boolean read mAutoDetected;

Sichtbarkeit:

Public

Beschreibung:

Gibt an, ob die AutoWert-Spalte automatisch ermittelt wurde

ExplicitInsertByDBMS

Notation:

property ExplicitInsertByDBMS: Boolean read mExplicitInsertByDBMS write mExplicitInsertByDBMS;

Sichtbarkeit:

Public

Beschreibung:

Gibt an, ob der AutoWert durch das Datenbanksystem eingefügt wird.

Options

Notation:

property Options[pMode: TSFBDSAutoValueGetMode]: TSFBDSAutoValueOptions read getOptions write setOptions;

Sichtbarkeit:

Public

Beschreibung:

Optionen, wann und wie der AutoWert eingefügt/generiert werden soll.

Siehe auch TSFBDSAutoValueOptions, TSFBDSAutoValueGetMode

TSFBDSCompareRecord

Beschreibung

Hilfsklasse zum Vergleich von Werten einzelner Datensätzen bei der Sortierung, siehe TSFBusinessDataSet.SortBuffer

Index

GetBlobFieldValByIdx
GetBlobFieldValByName
GetFieldValByIdx
GetFieldValByName

Funktionen

GetFieldValByName

Notation:

function GetFieldValByName(pFieldName: String): Variant;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt einen Feldwert anhand des Feldnamens

GetFieldValByIdx

Notation:

function GetFieldValByIdx(pFieldIdx: Integer): Variant;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt einen Feldwert anhand des Feldindexes

GetBlobFieldValByName

Notation:

function GetBlobFieldValByName(pFieldName: String): TArray<Byte>;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt den Wert eines Blob-Feldes anhand des Feldnamens

GetBlobFieldValByIdx

Notation:

function GetBlobFieldValByIdx(pFieldIdx: Integer): TArray<Byte>;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt den Wert eines Blob-Feldes anhand des Feldindexes

TSFBusinessDataRelation

Beschreibung

Klasse zur Verwaltung von Relationen (Master-Detail-Beziehungen).

Index

DestAttrs
DestDBIdent
DestFieldNames
DestObj
PassKeys
SrcAttrs
SrcDBIdent
SrcFieldNames
SrcObj
SyncDisabled

Eigenschaften

SrcObj

Notation:

property SrcObj: TSFBusinessData read mSrcObj write mSrcObj;

Sichtbarkeit:

Public

Beschreibung:

Referenz auf das Master-Objekt.

Siehe auch AddRelation

DestObj

Notation:

property DestObj: TSFBusinessData read mDestObj write mDestObj;

Sichtbarkeit:

Public

Beschreibung:

Referenz auf das Detail-Objekt

SrcAttrs

Notation:

property SrcAttrs: Variant read mSrcAttrs write mSrcAttrs;

Sichtbarkeit:

Public

Beschreibung:

Die übergebenen Feldnamen im Master-Objekt. Ein Feldname kann der Feldname in der Datenbanktabelle oder der Feldname im DataSet (wenn in der SQL-Abfrage mit Alias gearbeitet wird) sein.

DestAttrs

Notation:

property DestAttrs: Variant read mDestAttrs write mDestAttrs;

Sichtbarkeit:

Public

Beschreibung:

Die übergebenen Feldnamen im Detail-Objekt. Ein Feldname kann der Feldname in der Datenbanktabelle oder der Feldname im DataSet (wenn in der SQL-Abfrage mit Alias gearbeitet wird) sein.

SrcFieldNames

Notation:

property SrcFieldNames: Variant read mSrcFieldNames write mSrcFieldNames;

Sichtbarkeit:

Public

Beschreibung:

Die zu *SrcAttrs* korrespondierenden Feldnamen im DataSet. Diese unterscheiden sich dann zu *SrcAttrs*, wenn in der SQL-Abfrage mit Alias gearbeitet wird.

DestFieldNames

Notation:

property DestFieldNames: Variant read mDestFieldNames write mDestFieldNames;

Sichtbarkeit:

Public

Beschreibung:

Die zu *DestAttrs* korrespondierenden Feldnamen im DataSet. Diese unterscheiden sich dann zu *DestAttrs*, wenn in der SQL-Abfrage mit Alias gearbeitet wird.

SrcDBIdent

Notation:

property SrcDBIdent: Variant read mSrcDBIdent write mSrcDBIdent;

Sichtbarkeit:

Public

Beschreibung:

Die zu *SrcAttrs* korrespondierenden Identifier in der Datenbanktabelle. Diese unterscheiden sich dann zu *SrcAttrs*, wenn in der SQL-Abfrage mit Alias gearbeitet wird.

DestDBIdent

Notation:

property DestDBIdent: Variant read mDestDBIdent write mDestDBIdent;

Sichtbarkeit:

Public

Beschreibung:

Die zu *DestAttrs* korrespondierenden Identifier in der Datenbanktabelle. Diese unterscheiden sich dann zu *DestAttrs*, wenn in der SQL-Abfrage mit Alias gearbeitet wird.

SyncDisabled

Notation:

property SyncDisabled: Boolean read mSyncDisabled write mSyncDisabled;

Sichtbarkeit:

Public

Beschreibung:

Gibt an, ob die Relation aktuell von der Synchronisation ausgenommen ist.

Siehe auch SetDisableSyncRel,

PassKeys

Notation:

property PassKeys: Boolean read mPassKeys write mPassKeys;

Sichtbarkeit:

Public

Beschreibung:

Gibt an, ob Änderungen an den Schlüsselfeldern im Master-Objekt an das Detail-Objekt weitergereicht werden.

Siehe auch SetPassKeysRel

TSFBusinessDataRelationDesigner

Beschreibung

Hilfsklasse zur Verwaltung von Relationen (Master-Detail-Beziehungen) über den Designer (IDE).

Siehe auch TSFBusinessDataRelation

Index

DestAttrs
DestObj
DestWrapper
PassKeys
SrcAttrs
SrcObj

Eigenschaften

SrcObj

Notation:

property SrcObj: TSFBusinessData read mSrcObj;

Sichtbarkeit:

Public

Beschreibung:

Referenz auf das Master-Objekt.

Siehe auch AddRelation

DestWrapper

Notation:

property DestWrapper: TSFBusinessDataWrap read mDestWrapper write setDestWrapper;

Sichtbarkeit:

Published

Beschreibung:

Referenz auf das Detail-Objekt, welches in einem Wrapper gekapselt ist. In einem Wrapper werden spezialisierte BusinessData-Objekte gekapselt.

Siehe auch TSFBusinessDataWrap

DestObj

Notation:

property DestObj: TSFBusinessData read getDestObj write setDestObj;

Sichtbarkeit:

Published

Beschreibung:

Referenz auf das Detail-Objekt. Das Detail-Objekt kann entweder direkt zugewiesen werden oder das in einem Wrapper gekapselte, spezialisierte BusinessData-Objekt darstellen.

Siehe auch TSFBusinessDataWrap

SrcAttrs

Notation:

property SrcAttrs: String read mSrcAttrs write setSrcAttrs;

Sichtbarkeit:

Published

Beschreibung:

Die Feldnamen im Master-Objekt. Ein Feldname kann der Feldname in der Datenbanktabelle oder der Feldname im DataSet (wenn in der SQL-Abfrage mit Alias gearbeitet wird) sein.

DestAttrs

Notation:

property DestAttrs: String read mDestAttrs write setDestAttrs;

Sichtbarkeit:

Published

Beschreibung:

Die Feldnamen im Detail-Objekt. Ein Feldname kann der Feldname in der Datenbanktabelle oder der Feldname im DataSet (wenn in der SQL-Abfrage mit Alias gearbeitet wird) sein.

PassKeys

Notation:

property PassKeys: Boolean read mPassKeys write setPassKeys;

Sichtbarkeit:

Published

Beschreibung:

Gibt an, ob Änderungen an den Schlüsselfeldern im Master-Objekt an das Detail-Objekt weitergereicht werden.

Siehe auch SetPassKeysRel

TSFBusinessDataWrap

Beschreibung

Hilfsklasse zur Verwaltung von spezialisierten BusinessData-Objekten über den Designer (IDE).

Index

AddDataSetNotification
BusinessClassName
BusinessDataSet
RemoveDataSetNotification

Funktionen

AddDataSetNotification

Notation:

procedure AddDataSetNotification(pProc: TSFBusinessDataChanged);

Sichtbarkeit:

Public

Beschreibung:

Interne Funktion zum Handling des DataSets.

RemoveDataSetNotification

Notation:

procedure RemoveDataSetNotification(pProc: TSFBusinessDataChanged);

Sichtbarkeit:

Public

Beschreibung:

Interne Funktion zum Handling des DataSets.

Eigenschaften

BusinessClassName

Notation:

property BusinessClassName: String read mBusinessClassName write setBusinessClassName;

Sichtbarkeit:

Published

Beschreibung:

Der Klassennamen des spezialisierten BusinessData-Objekt.

BusinessDataSet

Notation:

property BusinessDataSet: TSFBusinessData read mBusinessDataSet;

Sichtbarkeit:

Published

Beschreibung:

Das gekapselte, spezialisierte BusinessData-Objekt. Achtung, das BusinessData-Objekt ist nur während der Laufzeit eine Instanz der unter BusinessClassName angegebenen Klasse. Zur Designzeit ist das BusinessData-Objekt unspezialisiert.

TSFBusinessDataWrapSource

Beschreibung

Eine DataSource, die ihr DataSet aus einem Wrapper bezieht.

Siehe auch TDataSource, TSFBusinessDataWrap

Index

BusinessDataWrapper
DataSet

Eigenschaften

BusinessDataWrapper

Notation:

property BusinessDataWrapper: TSFBusinessDataWrap read mWrapper write setWrapper;

Sichtbarkeit:

Public

Beschreibung:

Verweis auf den Wrapper.

DataSet

Notation:

property DataSet: TDataSet read getDataSet;

Sichtbarkeit:

Public

Beschreibung:

Verweis auf das, im Wrapper gekapselte, DataSet.

TSFStmt

Beschreibung

Klasse zur Verwaltung und Generierung von SQL-Abfragen.

Index

AddBaseRestriction
AddConditionAttr
AddConditionExists
AddConditionIsNotNull
AddConditionIsNull
AddConditionType
AddConditionVal
AddGroupAttr
AddInsertCondition
AddOrderAttr
AddSetCondition
AddStmtAttr
AssignStmt
AssignStmtTo
AttrDatabaseNameForAttrName
AttrDisplayName
AttrExists
AutoEscapeLike
BaseTable
ClearBaseRestrictions
ClearClientRectrictions
ClearConditions
ClearGroup
ClearInsConditions
ClearOrder
ClearSetConditions
ConfigStmtTimeValue
ConvertArrayValueToStr
ConvertValueInType
DBDialect
GenerateLevel
GenerateSubId
GenerateUnionId
GetConvertedValue
GetDBDialectCanSelWithoutTab
GetDBDialectCanSubInFrom
GetDBDialectLikeSupportsEscape
GetDBDialectLikeWildcardMany
GetDBDialectLikeWildcardSingle
GetDBDialectNeedTableOnSubInFrom
GetDeleteStmt
GetInsertStmt
GetLastAutoValueStmt
GetNextAutoValueStmt
GetNextTableNo
GetQuotedIdentifier

GetReferencedStmtByNamePath
GetReferencedStmtForParent
GetReferencedStmtNamePath
GetRelItemsForJoin
GetSelectStmt
GetStmtDatePart
GetStmtTimePart
GetTableAliasForAttr
GetTableForAttr
GetTableNameForAttr
GetTypeForValue
GetUpdateStmt
HasConditions
HasStmtDatePart
HasStmtTimePart
HasUnion
LikeEscapeChar
ListAttributeParams
ListAttributes
ListConditions
ListGroup
ListOrder
ListRestrictions
ListTables
LoadFromXml
LoadFromXmlDoc
ModfiyTableJoinType
OnAfterGenDelete
OnAfterGenInsert
OnAfterGenSelect
OnAfterGenUpdate
OnBeforeGenDelete
OnBeforeGenInsert
OnBeforeGenSelect
OnBeforeGenUpdate
OnGetDBDialectCls
QuoteType
ReconfigBaseTable
Reset
SaveToXmlDoc
SaveToXmlStr
SetBaseTable
SetRelItemsForJoin
SetStmtAggr
SetStmtAttr
SetTableJoin
SetUnion
StmtGenInfos
TableJoinAliasesForAttr
Union
UseDistinct

Funktionen

SetBaseTable

Notation:

function SetBaseTable(pTableName, pSchema, pCatalog, pTableAlias: String): TSFStmtTable; overload;

function SetBaseTable(pStmt: TSFStmt; pTableAlias: String): TSFStmtTable; overload;

function SetBaseTable(pStmtName, pTableAlias: String): TSFStmtTable; overload;

Sichtbarkeit:

Public

Beschreibung:

Setzt die Basistabelle. Die Basistabelle kann auch eine Referenz auf ein anderes Statement (Subselect) sein.

Wenn das Statement ein internes Statement einer Instanz von TSFBusinessData/TSFDataSet ist, wird die Basistabelle automatisch gesetzt.

Aliase werden automatisch vergeben, dieser muss also nur dann angegeben werden, wenn ein bestimmter Alias gewünscht ist.

ReconfigBaseTable

Notation:

procedure ReconfigBaseTable(pTableName, pSchema, pCatalog, pTableAlias: String);

Sichtbarkeit:

Public

Beschreibung:

Aktualisiert die Basistabelle.

AddStmtAttr

Notation:

function AddStmtAttr(pAttrName: String; pOnlyForSearch: Boolean): TSFStmtAttr;

Sichtbarkeit:

Public

Beschreibung:

Fügt dem Stmt ein, nicht näher spezifiziertes, Attribut bzw. eine Feld für die Select-Klausel hinzu.

Wenn der Parameter pOnlySearch true ist, handelt sich bei dem Attribut um ein reines Suchattribut, d. h. dieses wird nicht in der Select-Klausel aufgelistet. Wenn Sie bspsw. eine Suchbedingung für ein Tabellenfeld (where fieldname = …) erstellen wollen, muss das Tabellenfeld zuvor als Attribut hinzugefügt werden (auch dann, wenn es nicht in der Select-Klausel aufgeführt werden soll).

Zur weiteren Beschreibung eines, nicht spezifierten, Attributs muss dem Attribut mind. 1 Item hinzugefügt werden - siehe hierzu TSFStmtAttr.

Siehe auch TSFStmtAttr, TSFStmtAttrItemType

SetStmtAttr

Notation:

procedure SetStmtAttr(pAttrName, pAttrAlias, pTableAlias: String; pOnlyForSearch: Boolean); overload;

procedure SetStmtAttr(pAttrName, pAttrAlias: String; pStmtTable: TSFStmtTable; pOnlyForSearch: Boolean); overload;

Sichtbarkeit:

Public

Beschreibung:

Fügt dem Statement ein Attribut als Tabellenfeld hinzu. Der Parameter pAttrName gibt den Tabellenfeldnamen an, der Parameter pAttrAlias ist optional und stellt den Aliasnamen für das Tabellenfeld in der Select-Klausel dar.
Der Parameter pTableAlias/pStmtTable ist eine Referenz auf eine bereits hinzugefügte Tabelle. Jede Tabelle erhält automatisch einen eindeutigen Alias, als pTableAlias kann entweder dieser Alias oder der Tabellenname (sofern eindeutig) übergeben werden.

Siehe auch AddStmtAttr

SetStmtAggr

Notation:

procedure SetStmtAggr(pAggr, pAttrName, pAttrAlias, pTableAlias: String); overload;

procedure SetStmtAggr(pAggr, pAttrName, pAttrAlias: String; pStmtTable: TSFStmtTable); overload;

Sichtbarkeit:

Public

Beschreibung:

Fügt dem Statement ein Attribut als Tabellenfeld mit Aggregatfunktion hinzu.

Siehe auch SetStmtAttr, Konstanten

SetTableJoin

Notation:

function SetTableJoin(pTableAlias, pTableName, pSchema, pCatalog, pSourceTableAlias: String; pRelItems: TSFStmtJoinRelItems; pType: TSFStmtJoinType): TSFStmtTable; overload;

function SetTableJoin(pTableAlias, pTableName, pSchema, pCatalog, pSourceTableAlias: String; const pRelValsDest, pRelValsSource: Array of Variant; const pRelTypesDest, pRelTypesSource: Array of TSFStmtJoinRelItemType; pType: TSFStmtJoinType): TSFStmtTable; overload;

function SetTableJoin(pTableAlias, pSourceTableAlias: String; pDestStmt: TSFStmt; pRelItems: TSFStmtJoinRelItems; pType: TSFStmtJoinType): TSFStmtTable; overload;

function SetTableJoin(pTableAlias, pSourceTableAlias: String; pDestStmt: TSFStmt; const pRelValsDest, pRelValsSource: Array of Variant; const pRelTypesDest, pRelTypesSource: Array of TSFStmtJoinRelItemType; pType: TSFStmtJoinType): TSFStmtTable; overload;

function SetTableJoin(pTableAlias, pTableName, pSchema, pCatalog: String; pSourceTable: TSFStmtTable; pRelItems: TSFStmtJoinRelItems; pType: TSFStmtJoinType): TSFStmtTable; overload;

function SetTableJoin(pTableAlias, pTableName, pSchema, pCatalog: String; pSourceTable: TSFStmtTable; const pRelValsDest, pRelValsSource: Array of Variant; const pRelTypesDest, pRelTypesSource: Array of TSFStmtJoinRelItemType; pType: TSFStmtJoinType): TSFStmtTable; overload;

function SetTableJoin(pTableAlias: String; pSourceTable: TSFStmtTable; pDestStmt: TSFStmt; pRelItems: TSFStmtJoinRelItems; pType: TSFStmtJoinType): TSFStmtTable; overload;

function SetTableJoin(pTableAlias: String; pSourceTable: TSFStmtTable; pDestStmt: TSFStmt; const pRelValsDest, pRelValsSource: Array of Variant; const pRelTypesDest, pRelTypesSource: Array of TSFStmtJoinRelItemType; pType: TSFStmtJoinType): TSFStmtTable; overload;

function SetTableJoin(pTableAlias: String; pSourceTable: TSFStmtTable; pDestStmtName: String; pRelItems: TSFStmtJoinRelItems; pType: TSFStmtJoinType): TSFStmtTable; overload;

function SetTableJoin(pTableAlias: String; pSourceTable: TSFStmtTable; pDestStmtName: String; const pRelValsDest, pRelValsSource: Array of Variant; const pRelTypesDest, pRelTypesSource: Array of TSFStmtJoinRelItemType; pType: TSFStmtJoinType): TSFStmtTable; overload;

Sichtbarkeit:

Public

Beschreibung:

Fügt dem Statement einen Join für die unter pSourceTable/pSourceTableAlias angegebene Tabelle hinzu. Die erste Tabelle (Basistabelle) für ein Statement fügen Sie mit der Funktion SetBaseTable hinzu. Wenn das Statement ein internes Statement einer Instanz von TSFBusinessData/TSFDataSet ist, wird die Basistabelle automatisch gesetzt.

Wenn die Join-Tabelle eine Subselect sein soll, übergeben Sie dieser Funktion die entsprechende Instanz von TSFStmt.
Bei Verwendung von Subselects ist grundsätzlich zu beachten, dass die hierfür referenzierten Statements automatisch freigegeben werden, wenn diese keinen Owner haben.

Die Relationsattribute können mit dem entsprechenden Typ TSFStmtJoinRelItems oder über entsprechende Arrays übergeben werden. Ein Relationsattribut muss nicht grundsätzlich ein Tabellenfeld darstellen, sondern kann auch ein Wert sein.

Aliase werden automatisch vergeben, dieser muss also nur dann angegeben werden, wenn ein bestimmter Alias gewünscht ist.

Siehe auch SetBaseTable, TSFStmtJoinRelItems, TSFStmtJoinRelItem, TSFStmtJoinRelItemType, TSFStmtJoinType

ModfiyTableJoinType

Notation:

procedure ModfiyTableJoinType(pDestTableAlias, pSourceTableAlias: String; pTypeFrom, pTypeTo: TSFStmtJoinType); overload;

procedure ModfiyTableJoinType(pDestTable, pSourceTable: TSFStmtTable; pTypeFrom, pTypeTo: TSFStmtJoinType); overload;

Sichtbarkeit:

Public

Beschreibung:

Mit dieser Funktion kann der Typ eines Joins geändert werden.

Siehe auch TSFStmtJoinType

TableJoinAliasesForAttr

Notation:

function TableJoinAliasesForAttr(pSourceTableAlias, pAttr: String): Variant; overload;

function TableJoinAliasesForAttr(pSourceTable: TSFStmtTable; pAttr: String): Variant; overload;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt die Joins (Tabellen) für welche das übergebenen Attribut/Feld als Relationsattribut angegeben ist.

GetRelItemsForJoin

Notation:

function GetRelItemsForJoin(pSourceTable, pDestTable: TSFStmtTable): TSFStmtJoinRelItems; overload;

function GetRelItemsForJoin(pSourceTableAlias, pDestTableAlias: String): TSFStmtJoinRelItems; overload;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt die Relationsattribute für einen Join.

Siehe auch TSFStmtJoinRelItems

SetRelItemsForJoin

Notation:

procedure SetRelItemsForJoin(pSourceTable, pDestTable: TSFStmtTable; pRelItems: TSFStmtJoinRelItems); overload;

procedure SetRelItemsForJoin(pSourceTableAlias, pDestTableAlias: String; pRelItems: TSFStmtJoinRelItems); overload;

Sichtbarkeit:

Public

Beschreibung:

Ändert die Relationsattribute für einen Join.

Siehe auch TSFStmtJoinRelItems

GetNextTableNo

Notation:

function GetNextTableNo: Integer;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt die nächste, eindeutige Tabellennummer. Tabellen/Joins werden innerhalb des Statements automatisch nummeriert und mit einem eindeutigen Alias gekennzeichnet.

AddConditionVal

Notation:

procedure AddConditionVal(pTableAlias, pAttrName, pOp: String; pVal: Variant; pRestrict: Boolean = False);

Sichtbarkeit:

Public

Beschreibung:

Fügt eine Bedingung für die Where-Klausel hinzu, bei welcher als Suchbedingung ein Wert übergeben wird. Die Parameter *pTableAlias* und *pAttrName* kennzeichnen das Feld/Attribut für die Bedingung, als Alias kann auch der Tabellenname übergeben werden.

Wenn der Parameter *pRestrict* = *true* ist, dann wird die Bedingung durch die Funktion ClearConditions nicht gelöscht. Die Bedingung wird hierdurch als spezielle Suchbedingung (ClientRestriction) gekennzeichnet, ClientRestrictions können durch die Funktion ClearClientRestrictions gelöscht werden.

Ist das Feld/Attribut, für welches die Bedingung gesetzt wird, kein Tabellenfeld, kann der Parameter *pTableAlias* auch leer sein, der Parameter *pAttrName* entspricht in diesem Fall dem Alias des Attributs.

Wenn eine Bedingung für ein Feld/Attribut gesetzt wird, muss das Feld/Attribut dem Statement zuvor hinzugefügt worden sein (wenn dieses nicht in der Select-Klausel aufgeführt werden soll, mit *OnlyForSearch* = true).

Siehe auch SetStmtAttr, AddStmtAttr, Konstanten

AddConditionAttr

Notation:

procedure AddConditionAttr(pSrcTabAlias, pSrcAttrName, pOp, pDestTabAlias, pDestAttrName: String; pRestrict: Boolean = False);

Sichtbarkeit:

Public

Beschreibung:

Fügt eine Bedingung für die Where-Klausel hinzu, bei welcher als Suchbedingung ein anderes Feld/Attribut übergeben wird. Die Parameter *pSrcTabAlias/pDestTabAlias* und *pSrcAttrName/pDestAttrName* kennzeichnen das jeweilige Feld/Attribut für die Bedingung, als Alias kann auch der Tabellenname übergeben werden.

Wenn der Parameter *pRestrict* = *true* ist, dann wird die Bedingung durch die Funktion ClearConditions nicht gelöscht. Die Bedingung wird hierdurch als spezielle Suchbedingung (ClientRestriction) gekennzeichnet, ClientRestrictions können durch die Funktion ClearClientRestrictions gelöscht werden.

Ist ein Feld/Attribut, für welches die Bedingung gesetzt wird, kein Tabellenfeld, kann der Parameter *pSrcTabAlias/pDestTabAlias* auch leer sein, der Parameter *pSrcAttrName/pDestAttrName* entspricht in diesem Fall dem Alias des Attributs.

Wenn eine Bedingung für ein Feld/Attribut gesetzt wird, muss das Feld/Attribut dem Statement zuvor hinzugefügt worden sein (wenn dieses nicht in der Select-Klausel aufgeführt werden soll, mit *OnlyForSearch* = true).

Siehe auch SetStmtAttr, AddStmtAttrKonstanten

AddConditionIsNull

Notation:

procedure AddConditionIsNull(pTableAlias, pAttrName: String; pRestrict: Boolean = False);

Sichtbarkeit:

Public

Beschreibung:

Fügt eine Bedingung für die Where-Klausel hinzu, bei welcher als Suchbedingung auf NULL geprüft wird. Die Parameter *pTableAlias* und *pAttrName* kennzeichnen das Feld/Attribut für die Bedingung, als Alias kann auch der Tabellenname übergeben werden.

Wenn der Parameter *pRestrict* = *true* ist, dann wird die Bedingung durch die Funktion ClearConditions nicht gelöscht. Die Bedingung wird hierdurch als spezielle Suchbedingung (ClientRestriction) gekennzeichnet, ClientRestrictions können durch die Funktion ClearClientRestrictions gelöscht werden.

Ist das Feld/Attribut, für welches die Bedingung gesetzt wird, kein Tabellenfeld, kann der Parameter *pTableAlias* auch leer sein, der Parameter *pAttrName* entspricht in diesem Fall dem Alias des Attributs.

Wenn eine Bedingung für ein Feld/Attribut gesetzt wird, muss das Feld/Attribut dem Statement zuvor hinzugefügt worden sein (wenn dieses nicht in der Select-Klausel aufgeführt werden soll, mit *OnlyForSearch* = true).

Siehe auch SetStmtAttr, AddStmtAttr

AddConditionIsNotNull

Notation:

procedure AddConditionIsNotNull(pTableAlias, pAttrName: String; pRestrict: Boolean = False);

Sichtbarkeit:

Public

Beschreibung:

Fügt eine Bedingung für die Where-Klausel hinzu, bei welcher als Suchbedingung auf NICHT NULL geprüft wird. Die Parameter *pTableAlias* und *pAttrName* kennzeichnen das Feld/Attribut für die Bedingung, als Alias kann auch der Tabellenname übergeben werden.

Ist das Feld/Attribut, für welches die Bedingung gesetzt wird, kein Tabellenfeld, kann der Parameter *pTableAlias* auch leer sein, der Parameter *pAttrName* entspricht in diesem Fall dem Alias des Attributs.

Wenn eine Bedingung für ein Feld/Attribut gesetzt wird, muss das Feld/Attribut dem Statement zuvor hinzugefügt worden sein (wenn dieses nicht in der Select-Klausel aufgeführt werden soll, mit *OnlyForSearch* = true).

Siehe auch SetStmtAttr, AddStmtAttr

AddConditionType

Notation:

procedure AddConditionType(pType: TSFStmtConditionType; pRestrict: Boolean = False);

Sichtbarkeit:

Public

Beschreibung:

Fügt der Where-Klausel einen Eintrag mit dem spezifizierten Typ (z. B. Klammer, AND, OR) hinzu.

Siehe auch TSFStmtConditionType

AddConditionExists

Notation:

procedure AddConditionExists(pDestStmt: TSFStmt; pTableAlias, pDestTableAlias, pOp: String; pRelItems: TSFStmtJoinRelItems; pRestrict: Boolean = False); overload;

procedure AddConditionExists(pDestStmt: TSFStmt; pTableAlias, pDestTableAlias, pOp: String; const pRelValsDest, pRelValsSource: Array of Variant; const pRelTypesDest, pRelTypesSource: Array of TSFStmtJoinRelItemType; pRestrict: Boolean = False); overload;

procedure AddConditionExists(pDestStmtName, pTableAlias, pDestTableAlias, pOp: String; pRelItems: TSFStmtJoinRelItems; pRestrict: Boolean = False); overload;

procedure AddConditionExists(pDestStmtName, pTableAlias, pDestTableAlias, pOp: String; const pRelValsDest, pRelValsSource: Array of Variant; const pRelTypesDest, pRelTypesSource: Array of TSFStmtJoinRelItemType; pRestrict: Boolean = False); overload;

Sichtbarkeit:

Public

Beschreibung:

Fügt der Where-Klausel eine EXISTS-Bedingung hinzu. Der Subselect für die EXISTS-Bedingung ist eine Referenz auf ein anderes Statement, welches der Funktion im Parameter *pDestStmt* übergeben wird.

Bei Verwendung von Subselects ist grundsätzlich zu beachten, dass die hierfür referenzierten Statements automatisch freigegeben werden, wenn diese keinen Owner haben.

Die Relationsattribute können mit dem entsprechenden Typ TSFStmtJoinRelItems oder über entsprechende Arrays übergeben werden. Ein Relationsattribut muss nicht grundsätzlich ein Tabellenfeld darstellen, sondern kann auch ein Wert sein.

Wenn der Parameter *pRestrict* = *true* ist, dann wird die Bedingung durch die Funktion ClearConditions nicht gelöscht. Die Bedingung wird hierdurch als spezielle Suchbedingung (ClientRestriction) gekennzeichnet, ClientRestrictions können durch die Funktion ClearClientRestrictions gelöscht werden.

Siehe auch TSFStmtJoinRelItems, TSFStmtJoinRelItem, TSFStmtJoinRelItemType, Konstanten

AddOrderAttr

Notation:

procedure AddOrderAttr(pTableAlias, pAttrName: String; pOrderType: TSFStmtSortType = stmtSortTypeAsc);

Sichtbarkeit:

Public

Beschreibung:

Kennzeichnet ein zuvor hinzugefügtes Feld/Attribut als Sortierattribut (ORDER BY).

Die Parameter *pTableAlias* und *pAttrName* kennzeichnen das Feld/Attribut, als Alias kann auch der Tabellenname übergeben werden.

Ist das Feld/Attribut kein Tabellenfeld, kann der Parameter *pTableAlias* auch leer sein, der Parameter *pAttrName* entspricht in diesem Fall dem Alias des Attributs.

Siehe auch SetStmtAttr, AddStmtAttr, TSFStmtSortType

AddGroupAttr

Notation:

procedure AddGroupAttr(pTableAlias, pAttrName: String);

Sichtbarkeit:

Public

Beschreibung:

Kennzeichnet ein zuvor hinzugefügtes Feld/Attribut als Gruppierungsattribut (GROUP BY).

Die Parameter *pTableAlias* und *pAttrName* kennzeichnen das Feld/Attribut, als Alias kann auch der Tabellenname übergeben werden.

Ist das Feld/Attribut kein Tabellenfeld, kann der Parameter *pTableAlias* auch leer sein, der Parameter *pAttrName* entspricht in diesem Fall dem Alias des Attributs.

Siehe auch SetStmtAttr, AddStmtAttr

AddSetCondition

Notation:

procedure AddSetCondition(pAttrName: String; pVal: Variant; pValType: TSFStmtAttrItemValueType);

Sichtbarkeit:

Public

Beschreibung:

Setzt einen Attribut-/Feldwert für SET-Klausel eines UPDATE-Statements.

Siehe auch TSFStmtAttrItemValueType

AddInsertCondition

Notation:

procedure AddInsertCondition(pAttrName: String; pVal: Variant; pValType: TSFStmtAttrItemValueType);

Sichtbarkeit:

Public

Beschreibung:

Setzt einen Attribut-/Feldwert für ein INSERT-Statement.

Siehe auch TSFStmtAttrItemValueType

Reset

Notation:

procedure Reset;

Sichtbarkeit:

Public

Beschreibung:

Setzt das Statement komplett zurück, d. h. es werden sämtliche Tabellen/Joins, Attribte, Bedingungen, usw. gelöscht.

GetSelectStmt

Notation:

function GetSelectStmt(pLevel: Integer = 0; pSubId: Integer = 0; pUnionId: Integer = 0): String;

Sichtbarkeit:

Public

Beschreibung:

Generiert das SELECT-Statment. Die Parameter *pLevel*, *pSubId* und *pUnionId* sind für die interne Generierung von Subselects.

GetDeleteStmt

Notation:

function GetDeleteStmt: String;

Sichtbarkeit:

Public

Beschreibung:

Generiert das DELETE-Statement. Bedingungen für das DELETE-Statement werden über die selben Funktionen hinzugefügt, wie bei einem SELECT-Statement.

GetUpdateStmt

Notation:

function GetUpdateStmt: String;

Sichtbarkeit:

Public

Beschreibung:

Generiert das UPDATE-Statement. Bedingungen für das UPDATE-Statement werden über die selben Funktionen hinzugefügt, wie bei einem SELECT-Statement. Zum Setzen von Feldwerten siehe AddSetCondition.

GetInsertStmt

Notation:

function GetInsertStmt: String;

Sichtbarkeit:

Public

Beschreibung:

Generiert das INSERT-Statement.

Siehe auch AddInsertCondition

GetNextAutoValueStmt

Notation:

function GetNextAutoValueStmt(pRefName: String = ''): String;

Sichtbarkeit:

Public

Beschreibung:

Generiert, abhängig vom DBDialect, das Statement zur Ermittlung des nächsten Autowerts.

Siehe auch TSFStmtDBDialectConv

GetLastAutoValueStmt

Notation:

function GetLastAutoValueStmt(pRefName: String = ''): String;

Sichtbarkeit:

Public

Beschreibung:

Generiert, abhängig vom DBDialect, das Statement zur Ermittlung des letzten eingefügten Autowerts.

Siehe auch TSFStmtDBDialectConv

GetDBDialectCanSelWithoutTab

Notation:

function GetDBDialectCanSelWithoutTab(var pTableName: String): Boolean;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt anhand des DBDialect, ob Select-Anweisungen ohne Angabe einer Tabelle unterstützt werden.

GetDBDialectCanSubInFrom

Notation:

function GetDBDialectCanSubInFrom: Boolean;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt anhand des DBDialect, ob Select-Anweisungen mit einem Subselect in der From-Klausel unterstützt werden.

GetDBDialectNeedTableOnSubInFrom

Notation:

function GetDBDialectNeedTableOnSubInFrom: Boolean;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt anhand des DBDialect, ob Select-Anweisungen mit einem Subselect in der From-Klausel unterstützt werden, der (der Subselect) keine Tabelle referenziert.

GetDBDialectLikeWildcardSingle

Notation:

function GetDBDialectLikeWildcardSingle: String;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt das Wildcard-Zeichen für eine Like-Einzelsuche anhand des DBDialect. I. d. R. ist dieses Zeichen "_".

GetDBDialectLikeWildcardMany

Notation:

function GetDBDialectLikeWildcardMany: String;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt das Wildcard-Zeichen für eine Like-Mengensuche anhand des DBDialect. I. d. R. ist dieses Zeichen "%".

GetDBDialectLikeSupportsEscape

Notation:

function GetDBDialectLikeSupportsEscape: Boolean;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt anhand des DBDialect, ob die ESCAPE-Anweisung bei einer Like-Suche unterstützt wird. Durch die ESCAPE-Anweisung kann auch nach Zeichen gesucht werden, die dem Wildcard-Zeichen entsprechen.

AddBaseRestriction

Notation:

procedure AddBaseRestriction(pTableAlias, pAttrName: String; pVal: Variant; pVisible, pPreventNull: Boolean);

Sichtbarkeit:

Public

Beschreibung:

Wird für interne Suchbedingungen innherhalb einer Abfrage verwendet. Verwenden Sie diese Funktion nicht in Anwendungen, verwenden stattdessen AddConditionVal.

ClearBaseRestrictions

Notation:

procedure ClearBaseRestrictions;

Sichtbarkeit:

Public

Beschreibung:

Löscht interne Suchbedingungen, AddBaseRestriction die mit hinzugefügt wurden.

ClearConditions

Notation:

procedure ClearConditions;

Sichtbarkeit:

Public

Beschreibung:

Löscht alle Suchbedingungen, die nicht als *Restricted* gekennzeichnet wurden.

Siehe auch AddConditionVal, AddConditionAttr, AddConditionIsNull, AddConditionIsNotNull, AddConditionType, AddConditionExists

ClearClientRectrictions

Notation:

procedure ClearClientRectrictions;

Sichtbarkeit:

Public

Beschreibung:

Löscht alle Suchbedingungen, die als *Restricted* gekennzeichnet wurden.

Siehe auch AddConditionVal, AddConditionAttr, AddConditionIsNull, AddConditionIsNotNull, AddConditionType, AddConditionExists

ClearOrder

Notation:

procedure ClearOrder;

Sichtbarkeit:

Public

Beschreibung:

Löscht die Sortierung (ORDER BY) der Abfrage.

Siehe auch AddOrderAttr

ClearGroup

Notation:

procedure ClearGroup;

Sichtbarkeit:

Public

Beschreibung:

Löscht die Gruppierung (GROUP BY) der Abfrage.

Siehe auch AddGroupAttr

ClearSetConditions

Notation:

procedure ClearSetConditions;

Sichtbarkeit:

Public

Beschreibung:

Löscht die SET-Klausel bei einer UPDATE-Anweisung.

Siehe auch AddSetCondition

ClearInsConditions

Notation:

procedure ClearInsConditions;

Sichtbarkeit:

Public

Beschreibung:

Löscht die Felder/Attribute und Werte für eine INSERT-Anweisung.

Siehe auch AddInsertCondition

AttrExists

Notation:

function AttrExists(pAttrName, pTableAlias, pAggr: String): Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft, ob das durch die Parameter angegebene Feld/Attribut bereits existiert.

Die Suche kann nach dem Namen des Datenbankfeldes oder nach dem Attributalias erfolgen. Der Parameter *pTableAlias* kann leer sein.

Siehe auch AddStmtAttr, SetStmtAttr, SetStmtAggr

AttrDisplayName

Notation:

function AttrDisplayName(pAttrName, pTableAlias: String): String;

Sichtbarkeit:

Public

Beschreibung:

Der Anzeigename des durch die Parameter definierten Attributs/Felds in der (späterenen) Ergebnismenge. Wenn ein Alias für das Attribut verwendet ist der Anzeigename der Alias, andernfalls der Name des Tabellenfelds.

Die Suche kann nach dem Namen des Datenbankfeldes oder nach dem Attributalias erfolgen. Der Parameter *pTableAlias* kann leer sein.

Siehe auch AddStmtAttr, SetStmtAttr, SetStmtAggr

GetTableNameForAttr

Notation:

function GetTableNameForAttr(var pAttrName: String; pIncludeInvisible: Boolean): String;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt den Tabellennamen für das in *pAttrName* definierte Attribut/Feld. Der Parameter *pAttrName* kann den Alias des Attributs oder den Tabellenfeldnamen enthalten. Wird das Attribut/Feld gefunden, wird im var-Parameter *pAttrName* der Tabellenfeldname zurückgegeben.

Der Parameter *pIncludeInvisible* gibt an, ob auch Attribute/Felder durchsucht werden, die mit OnlyForSearch gekennzeichnet sind.

Siehe auch AddStmtAttr, SetStmtAttr, SetStmtAggr

GetTableAliasForAttr

Notation:

function GetTableAliasForAttr(var pAttrName: String; pIncludeInvisible: Boolean): String;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt den Tabellenalias für das in *pAttrName* definierte Attribut/Feld. Der Parameter *pAttrName* kann den Alias des Attributs oder den Tabellenfeldnamen enthalten. Wird das Attribut/Feld gefunden, wird im var-Parameter *pAttrName* der Tabellenfeldname zurückgegeben.

Der Parameter *pIncludeInvisible* gibt an, ob auch Attribute/Felder durchsucht werden, die mit OnlyForSearch gekennzeichnet sind.

Siehe auch AddStmtAttr, SetStmtAttr, SetStmtAggr

GetTableForAttr

Notation:

function GetTableForAttr(var pAttrName: String; pIncludeInvisible: Boolean): TSFStmtTable;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt das Tabellenobjekt für das in *pAttrName* definierte Attribut/Feld. Der Parameter *pAttrName* kann den Alias des Attributs oder den Tabellenfeldnamen enthalten. Wird das Attribut/Feld gefunden, wird im var-Parameter *pAttrName* der Tabellenfeldname zurückgegeben.

Der Parameter *pIncludeInvisible* gibt an, ob auch Attribute/Felder durchsucht werden, die mit OnlyForSearch gekennzeichnet sind.

Siehe auch AddStmtAttr, SetStmtAttr, SetStmtAggr

HasConditions

Notation:

function HasConditions: Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft, ob die Abfrage Suchbedingungen enthält.

Siehe auch AddConditionVal, AddConditionAttr, AddConditionIsNull, AddConditionIsNotNull, AddConditionType, AddConditionExists

GetConvertedValue

Notation:

function GetConvertedValue(pValue: Variant; pExplicitCast: Boolean = False; pEscapeLike: Boolean = False): String;

Sichtbarkeit:

Public

Beschreibung:

Wird intern verwendet, hinzugefügte Werte (z. B. von AddConditionVal) für die Abfrage zu konvertieren.

Siehe auch TSFStmtDBDialectConv.ConvertValue

GetTypeForValue

Notation:

function GetTypeForValue(pValue: Variant): TSFStmtValueType;

Sichtbarkeit:

Public

Beschreibung:

Wird intern dazu verwendet, den Typ hinzugefügter Werte (z. B. von AddConditionVal) zu bestimmen.

Siehe auch TSFStmtDBDialectConv.ValueTypeForValue

ConvertValueInType

Notation:

function ConvertValueInType(var pValue: Variant; pType: TSFStmtValueType; pHandleArray: Boolean = False): Boolean;

Sichtbarkeit:

Public

Beschreibung:

Wird intern dazu verwendet, Werte für den Import zu konvertieren.

ConvertArrayValueToStr

Notation:

function ConvertArrayValueToStr(pValue: Variant): String;

Sichtbarkeit:

Public

Beschreibung:

Wird intern dazu verwendet, Array-Werte für den Export zu konvertieren

GetReferencedStmtByNamePath

Notation:

function GetReferencedStmtByNamePath(pNamePath: String): TSFStmt;

Sichtbarkeit:

Public

Beschreibung:

Sucht eine Instanz von TSFStmt anhand dessen Komponentennamen.

Diese Funktion wird intern beim Import bzw. der Generierung einer importierten Abfrage verwendet.

GetReferencedStmtForParent

Notation:

function GetReferencedStmtForParent(pNamePath: String; pParent: TComponent): TSFStmt;

Sichtbarkeit:

Public

Beschreibung:

Sucht eine Instanz von TSFStmt anhand dessen Komponentennamen, welches der Komponente in *pParent* untergeordnet ist.

Diese Funktion wird intern beim Import bzw. der Generierung einer importierten Abfrage verwendet.

GetReferencedStmtNamePath

Notation:

function GetReferencedStmtNamePath(pComp: TComponent = nil): String;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt den Namen und Pfad (Übergeordnete Komponenten) für eine Instanz von TSFStmt. Diese Funktion wird intern beim Export verwendet.

GetQuotedIdentifier

Notation:

function GetQuotedIdentifier(pIdentifier: String): String;

Sichtbarkeit:

Public

Beschreibung:

Setzt einen Identifier bei der Generierung einer Abfrage in Anführungszeichen, falls erforderlich.

Siehe auch QuoteType

SetUnion

Notation:

procedure SetUnion(pStmt: TSFStmt);

Sichtbarkeit:

Public

Beschreibung:

Fügt der Abfrage einen Union hinzu.

Bei Verwendung von Subselects ist grundsätzlich zu beachten, dass die hierfür referenzierten Statements automatisch freigegeben werden, wenn diese keinen Owner haben.

HasUnion

Notation:

function HasUnion: Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft, ob die Abfrage einen Union hat.

AssignStmt

Notation:

function AssignStmt: TSFStmt;

Sichtbarkeit:

Public

Beschreibung:

Kopiert die Abfrage.

AssignStmtTo

Notation:

procedure AssignStmtTo(pDest: TSFStmt);

Sichtbarkeit:

Public

Beschreibung:

Kopiert die Abfrage in die im Parameter pDest angegebene Instanz von TSFStmt.

AttrDatabaseNameForAttrName

Notation:

function AttrDatabaseNameForAttrName(pTableAlias, pAttrName: String): String; overload;

function AttrDatabaseNameForAttrName(pAttrName: String; var pTable: TSFStmtTable): String; overload;

Sichtbarkeit:

Public

Beschreibung:

Sucht den Tabellenfeldnamen für das durch die Parameter definierte Feld/Attribut.

Die Suche kann nach dem Namen des Datenbankfeldes oder nach dem Attributalias erfolgen. Der Parameter *pTableAlias* kann leer sein.

ListTables

Notation:

function ListTables: TObjectList<TSFStmtTable>;

Sichtbarkeit:

Public

Beschreibung:

Listet alle Tabellenreferenzen auf.

Siehe auch SetBaseTable, SetTableJoin

ListAttributes

Notation:

function ListAttributes: TObjectList<TSFStmtAttr>;

Sichtbarkeit:

Public

Beschreibung:

Listet alle Attributreferenzen auf.

Siehe auch AddStmtAttr, SetStmtAttr, SetStmtAggr

ListAttributeParams

Notation:

function ListAttributeParams: TStrings;

Sichtbarkeit:

Public

Beschreibung:

Listet alle Parameternamen auf.

Siehe auch AddStmtAttr, TSFStmtAttr.AddItemParam

ListConditions

Notation:

function ListConditions: TObjectList<TSFStmtCondition>;

Sichtbarkeit:

Public

Beschreibung:

Listet alle Bedingungsreferenzen auf, die nicht *Restricted* sind.

Siehe auch AddConditionVal, AddConditionAttr, AddConditionIsNull, AddConditionIsNotNull, AddConditionType, AddConditionExists

ListRestrictions

Notation:

function ListRestrictions: TObjectList<TSFStmtCondition>;

Sichtbarkeit:

Public

Beschreibung:

Listet alle Bedingungsreferenzen auf, die nicht *Restricted* sind.

Siehe auch AddConditionVal, AddConditionAttr, AddConditionIsNull, AddConditionIsNotNull, AddConditionType, AddConditionExists

ListOrder

Notation:

function ListOrder: TObjectList<TSFStmtAttr>;

Sichtbarkeit:

Public

Beschreibung:

Listet alle Sortierattribute auf.

Siehe auch AddOrderAttr

ListGroup

Notation:

function ListGroup: TObjectList<TSFStmtAttr>;

Sichtbarkeit:

Public

Beschreibung:

Listet alle Gruppierungsattribute auf.

Siehe auch AddGroupAttr

ConfigStmtTimeValue

Notation:

function ConfigStmtTimeValue(pTime: TTime): TDateTime;

Sichtbarkeit:

Public

Beschreibung:

Konvertiert übergebenen Zeitwert in einen Wert, der der Abfrage als Zeitwert übergeben werden kann - damit die Abfrage den Wert als reinen Zeitwert erkennt.

Siehe auch TSFStmtAttr.AddItemValueTime, TSFStmtAttr.AddItemValueDate, TSFStmtAttr.AddItemValueDateTime

HasStmtDatePart

Notation:

function HasStmtDatePart(pDate: TDateTime): Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft, ob der angegebenen Datum- und Zeitwert einen Datumsteil hat.

Siehe auch ConfigStmtTimeValue

HasStmtTimePart

Notation:

function HasStmtTimePart(pDate: TDateTime): Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft, ob der angegebenen Datum- und Zeitwert einen Zeitteil hat.

Siehe auch ConfigStmtTimeValue

GetStmtDatePart

Notation:

function GetStmtDatePart(pDate: TDateTime): TDate;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt den Datumsteil des übergebenen Datum- und Zeitwerts.

Siehe auch ConfigStmtTimeValue

GetStmtTimePart

Notation:

function GetStmtTimePart(pDate: TDateTime): TTime;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt den Zeitteil des übergebenen Datum- und Zeitwerts.

Siehe auch ConfigStmtTimeValue

SaveToXmlDoc

Notation:

function SaveToXmlDoc: IXmlDocument;

Sichtbarkeit:

Public

Beschreibung:

Speichert die gesamte Abfrage in einem Xml-Document.

Subselects werden beim Export nur referenziert, siehe GetReferencedStmtNamePath.

SaveToXmlStr

Notation:

procedure SaveToXmlStr(var pXmlStr: String);

Sichtbarkeit:

Public

Beschreibung:

Speichert die gesamte Abfrage in einem Xml-String.

Subselects werden beim Export nur referenziert, siehe GetReferencedStmtNamePath.

LoadFromXml

Notation:

procedure LoadFromXml(pXmlStr: String; pSuspendRefs: Boolean = True);

Sichtbarkeit:

Public

Beschreibung:

Lädt die Abfrage aus einem Xml-String.

Subselects werden beim Import nur als Referenz gelesen, siehe GetReferencedStmtByNamePath. Diese müssen spätestens bei Generierung der Abfrage erzeugt und verfügbar sein.

LoadFromXmlDoc

Notation:

procedure LoadFromXmlDoc(pXmlDoc: IXmlDocument; pSuspendRefs: Boolean = True);

Sichtbarkeit:

Public

Beschreibung:

Lädt die Abfrage aus einem Xml-Dokument.

Subselects werden beim Import nur als Referenz gelesen, siehe GetReferencedStmtByNamePath. Diese müssen spätestens bei Generierung der Abfrage erzeugt und verfügbar sein.

Eigenschaften

BaseTable

Notation:

property BaseTable: TSFStmtTable read mBaseTable;

Sichtbarkeit:

Public

Beschreibung:

Referenz auf die Basistabelle, siehe SetBaseTable

GenerateLevel

Notation:

property GenerateLevel: Integer read mGenerateLevel;

Sichtbarkeit:

Public

Beschreibung:

Wird intern zur Generierung von Subselects verwendet.

GenerateSubId

Notation:

property GenerateSubId: Integer read mGenerateSubId;

Sichtbarkeit:

Public

Beschreibung:

Wird intern zur Generierung von Subselects verwendet.

GenerateUnionId

Notation:

property GenerateUnionId: Integer read mGenerateUnionId;

Sichtbarkeit:

Public

Beschreibung:

Wird intern zur Generierung von Subselects verwendet.

QuoteType

Notation:

property QuoteType: TSFStmtQuoteType read mQuoteType write mQuoteType;

Sichtbarkeit:

Public

Beschreibung:

Gibt an, ob und wie Identifier in Anführungszeichen gesetzt werden sollen.

Siehe auch TSFStmtQuoteType

StmtGenInfos

Notation:

property StmtGenInfos: TSFStmtGenInfos read mStmtGenInfos;

Sichtbarkeit:

Public

Beschreibung:

Informationen zur Generierung einer Abfrage. Diese Informationen sind nach der Generierung der Abfrage verfügbar.

Siehe auch TSFStmtGenInfos

UseDistinct

Notation:

property UseDistinct: Boolean read mUseDistinct write mUseDistinct;

Sichtbarkeit:

Public

Beschreibung:

Gibt an, ob der Abfrage eine DISTINCT-Anweisung vorangestellt werden soll.

LikeEscapeChar

Notation:

property LikeEscapeChar: String read mLikeEscapeChar write mLikeEscapeChar;

Sichtbarkeit:

Public

Beschreibung:

Wenn *AutoEscapeLike* ausgeschalten ist, kann durch Angabe von *LikeEscapeChar* das Escape-Zeichen angegeben werden, mit dem dann - sofern der DBDialect Escape-Anweisungen unterstützt - alle LIKE-Bedingungen escapt werden.

AutoEscapeLike

Notation:

property AutoEscapeLike: Boolean read mAutoEscapeLike write mAutoEscapeLike;

Sichtbarkeit:

Public

Beschreibung:

Ist *AutoEscapeLike* gesetzt, werden alle LIKE-Bedingungen geprüft und ggf. automatisch escapt (sofern der DBDialect Escape-Anweisungen unterstützt).

DBDialect

Notation:

property DBDialect: TSFStmtDBDialect read mDBDialect write mDBDialect;

Sichtbarkeit:

Public

Beschreibung:

Der Dialect bzw. das Datenbanksystem, für welches die Abfrage generiert werden soll.

Siehe auch TSFStmtDBDialect

Union

Notation:

property Union: TSFStmt read getUnion write SetUnion;

Sichtbarkeit:

Public

Beschreibung:

Referenz auf eine UNION-Abfrage.

Bei Verwendung von Subselects ist grundsätzlich zu beachten, dass die hierfür referenzierten Statements automatisch freigegeben werden, wenn diese keinen Owner haben.

Ereignisse

OnBeforeGenSelect

Notation:

property OnBeforeGenSelect: TSFStmtGenSelectEvent read mOnBeforeGenSelect write mOnBeforeGenSelect;

Sichtbarkeit:

Public

Beschreibung:

Wird ausgelöst, bevor die SELECT-Abfrage generiert wird. Bei Verwendung dieses Ereignisses ist zu beachten, dass dies von TSFBusinessData/TSFDataSet intern ebenfalls verwendet wird.

OnAfterGenSelect

Notation:

property OnAfterGenSelect: TSFStmtGenSelectEvent read mOnAfterGenSelect write mOnAfterGenSelect;

Sichtbarkeit:

Public

Beschreibung:

Wird ausgelöst, nachdem die SELECT-Abfrage generiert wurde. Bei Verwendung dieses Ereignisses ist zu beachten, dass dies von TSFBusinessData/TSFDataSet intern ebenfalls verwendet wird.

OnBeforeGenDelete

Notation:

property OnBeforeGenDelete: TSFStmtGenSelectEvent read mOnBeforeGenDelete write mOnBeforeGenDelete;

Sichtbarkeit:

Public

Beschreibung:

Wird ausgelöst, bevor die DELETE-Anweisung generiert wird.

OnAfterGenDelete

Notation:

property OnAfterGenDelete: TSFStmtGenSelectEvent read mOnAfterGenDelete write mOnAfterGenDelete;

Sichtbarkeit:

Public

Beschreibung:

Wird ausgelöst, nachdem die DELETE-Anweisung generiert wurde.

OnBeforeGenUpdate

Notation:

property OnBeforeGenUpdate: TSFStmtGenSelectEvent read mOnBeforeGenUpdate write mOnBeforeGenUpdate;

Sichtbarkeit:

Public

Beschreibung:

Wird ausgelöst, bevor die UPDATE-Anweisung generiert wird.

OnAfterGenUpdate

Notation:

property OnAfterGenUpdate: TSFStmtGenSelectEvent read mOnAfterGenUpdate write mOnAfterGenUpdate;

Sichtbarkeit:

Public

Beschreibung:

Wird ausgelöst, nachdem die UPDATE-Anweisung generiert wurde.

OnBeforeGenInsert

Notation:

property OnBeforeGenInsert: TSFStmtGenSelectEvent read mOnBeforeGenInsert write mOnBeforeGenInsert;

Sichtbarkeit:

Public

Beschreibung:

Wird ausgelöst, bevor die INSERT-Anweisung generiert wird.

OnAfterGenInsert

Notation:

property OnAfterGenInsert: TSFStmtGenSelectEvent read mOnAfterGenInsert write mOnAfterGenInsert;

Sichtbarkeit:

Public

Beschreibung:

Wird ausgelöst, nachdem die INSERT-Anweisung generiert wurde.

OnGetDBDialectCls

Notation:

property OnGetDBDialectCls: TSFStmtGetDialectConvEvent read mOnGetDBDialectCls write mOnGetDBDialectCls;

Sichtbarkeit:

Public

Beschreibung:

Mit diesem Ereignis können Sie eine eigene Konverter-Klasse integrieren.

Siehe auch TSFStmtGetDialectConvEvent, TSFStmtDBDialectConv

TSFStmtTable

Beschreibung

Klasse zur Verwaltung von Tabellen innerhalb des Statementgenerators

Index

AssignStmtTable
AssignStmtTableJoins
Catalog
GetJoinTableAliasesForAttr
GetJoinTableByAlias
GetJoinType
GetMaxTableNo
GetRelItemsForJoin
GetTableDef
HasJoins
ListJoinTables
LoadFromXmlTable
ModifyJoinType
ParentStmt
QuotedTableCatalog
QuotedTableIdentifier
QuotedTableName
QuotedTableSchema
ResetJoins
SaveToXmlTable
Schema
SetRelItemsForJoin
SetTableJoin
TableAlias
TableAliasNested
TableIdentifier
TableName
TableNo
TableStmt

Funktionen

SetTableJoin

Notation:

function SetTableJoin(pTableAlias, pTableName, pSchema, pCatalog: String; pTableNo: Integer; pRelItems: TSFStmtJoinRelItems; pType: TSFStmtJoinType): TSFStmtTable; overload;

function SetTableJoin(pTableAlias: String; pStmt: TSFStmt; pTableNo: Integer; pRelItems: TSFStmtJoinRelItems; pType: TSFStmtJoinType): TSFStmtTable; overload;

function SetTableJoin(pTableAlias, pStmtName: String; pTableNo: Integer; pRelItems: TSFStmtJoinRelItems; pType: TSFStmtJoinType): TSFStmtTable; overload;

Sichtbarkeit:

Public

Beschreibung:

Fügt der Tabelle einen Join hinzu - siehe TSFStmt.SetTableJoin.

GetJoinTableByAlias

Notation:

function GetJoinTableByAlias(pAlias: String; pSearchType: TSFStmtTableSearchType = stmtTableSearchAll): TSFStmtTable;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt einen Join (Tabelleninstanz) anhand dessen Alias- oder Tabellennamen.

GetJoinTableAliasesForAttr

Notation:

function GetJoinTableAliasesForAttr(pAttr: String): Variant;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt die Joins (Tabellen) für welche das übergebenen Attribut/Feld als Relationsattribut angegeben ist.

Siehe auch TSFStmt.TableJoinAliasesForAttr

ResetJoins

Notation:

procedure ResetJoins;

Sichtbarkeit:

Public

Beschreibung:

Setzt alle Joins zurück.

Siehe auch TSFStmt.Reset

HasJoins

Notation:

function HasJoins: Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft, ob die Tabelle Joins besitzt.

GetMaxTableNo

Notation:

function GetMaxTableNo: Integer;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt die letzte vergebene Tabellennummer.

Siehe auch TSFStmt.GetNextTableNo

GetTableDef

Notation:

function GetTableDef(pWithAlias: Boolean = True): String;

Sichtbarkeit:

Public

Beschreibung:

Generiert die SQL-Anweisung für die Tabelle einschl. deren Joins.

Siehe auch TSFStmt.GetSelectStmt, TSFStmt.GetDeleteStmt, TSFStmt.GetUpdateStmt, TSFStmt.GetInsertStmt

AssignStmtTable

Notation:

function AssignStmtTable(pDestStmt: TSFStmt): TSFStmtTable;

Sichtbarkeit:

Public

Beschreibung:

Kopiert die Tabelle einschl. Joins.

Siehe auch AssignStmtTableJoins, TSFStmt.AssignStmt, TSFStmt.AssignStmtTo

AssignStmtTableJoins

Notation:

procedure AssignStmtTableJoins(pDest: TSFStmtTable);

Sichtbarkeit:

Public

Beschreibung:

Kopiert die Joins einer Tabelle.

Siehe auch AssignStmtTable, TSFStmt.AssignStmt, TSFStmt.AssignStmtTo

GetJoinType

Notation:

function GetJoinType(pDest: TSFStmtTable): TSFStmtJoinType;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt den Typ des Joins für die übergebene Tabelle.

Siehe auch TSFStmtJoinType

ModifyJoinType

Notation:

procedure ModifyJoinType(pDest: TSFStmtTable; pTypeFrom, pTypeTo: TSFStmtJoinType);

Sichtbarkeit:

Public

Beschreibung:

Ändert den Typ des Joins für die übergebene Tabelle.

Siehe auch TSFStmt.ModfiyTableJoinType

GetRelItemsForJoin

Notation:

function GetRelItemsForJoin(pDest: TSFStmtTable): TSFStmtJoinRelItems;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt die Realationsattribute des Joins für die übergebene Tabelle.

Siehe auch TSFStmt.GetRelItemsForJoin

SetRelItemsForJoin

Notation:

procedure SetRelItemsForJoin(pDest: TSFStmtTable; pRelItems: TSFStmtJoinRelItems);

Sichtbarkeit:

Public

Beschreibung:

Ändert die Realationsattribute des Joins für die übergebene Tabelle.

Siehe auch TSFStmt.SetRelItemsForJoin

QuotedTableIdentifier

Notation:

function QuotedTableIdentifier: String;

Sichtbarkeit:

Public

Beschreibung:

Gibt den Tabellenidentifier in datenbankabhängigen Anführungszeichen zurück, falls erforderlich.

Siehe auch QuotedTableName, QuotedTableSchema, QuotedTableCatalog

QuotedTableName

Notation:

function QuotedTableName: String;

Sichtbarkeit:

Public

Beschreibung:

Gibt den Tabellennamen in datenbankabhängigen Anführungszeichen zurück, falls erforderlich.

Siehe auch QuotedTableIdentifier, QuotedTableSchema, QuotedTableCatalog

QuotedTableSchema

Notation:

function QuotedTableSchema: String;

Sichtbarkeit:

Public

Beschreibung:

Gibt den Tabellenschema in datenbankabhängigen Anführungszeichen zurück, falls erforderlich.

Siehe auch QuotedTableIdentifier, QuotedTableName, QuotedTableCatalog

QuotedTableCatalog

Notation:

function QuotedTableCatalog: String;

Sichtbarkeit:

Public

Beschreibung:

Gibt den Tabellenschema in datenbankabhängigen Anführungszeichen zurück, falls erforderlich.

Siehe auch QuotedTableIdentifier, QuotedTableName, QuotedTableSchema

ListJoinTables

Notation:

procedure ListJoinTables(pLst: TObjectList<TSFStmtTable>; pRecursive: Boolean = True);

Sichtbarkeit:

Public

Beschreibung:

Listet die Joins einer Tabelle auf.

Siehe auch TSFStmt.ListTables

SaveToXmlTable

Notation:

procedure SaveToXmlTable(pXmlTable: TSFStmtTableXML);

Sichtbarkeit:

Public

Beschreibung:

Speichert die Tabelle in ihre Xml-Entsprechung.

Siehe auch TSFStmt.SaveToXmlDoc, TSFStmt.SaveToXmlStr

LoadFromXmlTable

Notation:

procedure LoadFromXmlTable(pXmlTable: TSFStmtTableXML; pSuspendRefs: Boolean);

Sichtbarkeit:

Public

Beschreibung:

Liest die Tabelle aus ihrer Xml-Entsprechung.

Siehe auch TSFStmt.LoadFromXml, TSFStmt.LoadFromXmlDoc

Eigenschaften

TableName

Notation:

property TableName: String read mTableName;

Sichtbarkeit:

Public

Beschreibung:

Gibt den Tabellennamen an, sofern es sich um eine Datenbanktabelle handelt.

TableStmt

Notation:

property TableStmt: TSFStmt read getTableStmt;

Sichtbarkeit:

Public

Beschreibung:

Referenz auf den Subselect, sofern es sich um eine Abfrage handelt.

TableAlias

Notation:

property TableAlias: String read getTableAlias;

Sichtbarkeit:

Public

Beschreibung:

Gibt den Alias für die Tabelle an.

TableAliasNested

Notation:

property TableAliasNested[pLevel, pSubId, pUnionId: Integer]: String read getTableAliasNested;

Sichtbarkeit:

Public

Beschreibung:

Gibt den Alias für die Tabelle bei der Generierung als Subselect an.

TableNo

Notation:

property TableNo: Integer read mTableNo;

Sichtbarkeit:

Public

Beschreibung:

Gibt die, innerhalb einer Abfrage bzw. innerhalb eines Statements, eindeutige Tabellennummer an.

ParentStmt

Notation:

property ParentStmt: TSFStmt read mParentStmt;

Sichtbarkeit:

Public

Beschreibung:

Referenz auf die Abfrage bzw. das Statement.

Schema

Notation:

property Schema: String read mSchema;

Sichtbarkeit:

Public

Beschreibung:

Gibt das Schema an, sofern es sich um eine Datenbanktabelle handelt.

Catalog

Notation:

property Catalog: String read mCatalog;

Sichtbarkeit:

Public

Beschreibung:

Gibt den Katalog an, sofern es sich um eine Datenbanktabelle handelt.

TableIdentifier

Notation:

property TableIdentifier: String read getTableIdentifier;

Sichtbarkeit:

Public

Beschreibung:

Gibt den Tabellenidentifier an, sofern es sich um eine Datenbanktabelle handelt.

TSFStmtTableJoin

Beschreibung

Klasse zur Verwaltung von Joins innerhalb des Statementgenerators

Index

DestTable
GetJoinDef
SaveToXmlRelation

Funktionen

GetJoinDef

Notation:

function GetJoinDef: String;

Sichtbarkeit:

Public

Beschreibung:

Generiert die SQL-Anweisung für den Join.

Siehe auch TSFStmt.GetSelectStmt, TSFStmt.GetDeleteStmt, TSFStmt.GetUpdateStmt, TSFStmt.GetInsertStmt, TSFStmtTable.GetTableDef

SaveToXmlRelation

Notation:

procedure SaveToXmlRelation(pXmlRelation: TSFStmtTableRelationXML);

Sichtbarkeit:

Public

Beschreibung:

Speichert den Join in seine Xml-Entsprechung.

Siehe auch TSFStmt.SaveToXmlDoc, TSFStmt.SaveToXmlStr, TSFStmtTable.SaveToXmlTable

Eigenschaften

DestTable

Notation:

property DestTable: TSFStmtTable read mDestTable;

Sichtbarkeit:

Public

Beschreibung:

Referenz auf die Zieltabelle des Joins.

TSFStmtAttr

Beschreibung

Klasse zur Verwaltung von Attributen eines Statementsgenerators. Attribute sind die Teile des Statements/der Abfrage, die, sofern sie nicht als *OnlyForSearch* definiert sind, in der Select-Klausel aufgelistet werden. Ein Attribut kann 1 bis n Items unterschiedlicher Typen haben.

Index

AddItem
AddItemAggrFunc
AddItemBracket
AddItemDbFld
AddItemDynamic
AddItemOperator
AddItemParam
AddItemStmt
AddItemValue
AddItemValueDate
AddItemValueDateTime
AddItemValueTime
AssignStmtAttr
AttrName
DBAttrAggr
DBAttrName
DBAttrTable
GetAttrDef
GetSelectDef
HasItems
IsSingleDBFieldItem

IsSingleDBFieldUndefined
IsSingleItem
Items
LoadFromXmlAttr
OnlyForSearch
ParentStmt
SaveToXmlAttr
SetItemParamNamesToList
SortType

Funktionen

GetSelectDef

Notation:

function GetSelectDef: String;

Sichtbarkeit:

Public

Beschreibung:

Generiert die Attributdefinition für eine SELECT-Abfrage.

Siehe auch TSFStmt.GetSelectStmt

GetAttrDef

Notation:

function GetAttrDef(pWithSortType: Boolean = False; pWithAliases: Boolean = True; pExplicitCast: Boolean = False; pEscapeLike: Boolean = False): String;

Sichtbarkeit:

Public

Beschreibung:

Generiert die Definition des Attributs unabhängig von der Position innerhalb einer Abfrage (z. B. für die WHERE-Klausel als Suchattribut).

Siehe auch TSFStmt.GetSelectStmt, TSFStmt.GetDeleteStmt, TSFStmt.GetUpdateStmt, TSFStmt.GetInsertStmt

HasItems

Notation:

function HasItems: Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft, ob das Attribut Items besitzt.

IsSingleItem

Notation:

function IsSingleItem: Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft, ob das Attribut lediglich 1 Item besitzt.

IsSingleDBFieldItem

Notation:

function IsSingleDBFieldItem: Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft, ob das Attribut lediglich 1 Item besitzt, welches ein Datenbankfeld darstellt.

IsSingleDBFieldUndefined

Notation:

function IsSingleDBFieldUndefined: Boolean;

Sichtbarkeit:

Public

Beschreibung:

Prüft, ob das Attribut lediglich 1 Item besitzt, welches ein undefiniertes Datenbankfeld (entspricht = *) darstellt.

AddItem

Notation:

procedure AddItem(pType: TSFStmtAttrItemType; pTable: TSFStmtTable; pItemValue: Variant; pAggr: String);

Sichtbarkeit:

Public

Beschreibung:

Fügt ein Item mit dem definierten Typ hinzu.

AddItemDbFld

Notation:

procedure AddItemDbFld(pTable: TSFStmtTable; pAttrName, pAggr: String);

Sichtbarkeit:

Public

Beschreibung:

Fügt ein Item vom Typ *Datenbankfeld* hinzu

AddItemValue

Notation:

procedure AddItemValue(pValue: Variant);

Sichtbarkeit:

Public

Beschreibung:

Fügt ein Item vom Typ *Wert* hinzu.

AddItemValueDateTime

Notation:

procedure AddItemValueDateTime(pValue: TDateTime);

Sichtbarkeit:

Public

Beschreibung:

Fügt ein Item vom Typ *Datum und Zeit* hinzu.

AddItemValueDate

Notation:

procedure AddItemValueDate(pValue: TDate);

Sichtbarkeit:

Public

Beschreibung:

Fügt ein Item vom Typ *Datum* hinzu.

AddItemValueTime

Notation:

procedure AddItemValueTime(pValue: TTime);

Sichtbarkeit:

Public

Beschreibung:

Fügt ein Item vom Typ *Zeit* hinzu.

AddItemStmt

Notation:

procedure AddItemStmt(pStmt: TSFStmt); overload;

procedure AddItemStmt(pStmtName: String); overload;

Sichtbarkeit:

Public

Beschreibung:

Fügt ein Item vom Typ *Stmt (Subselect)* hinzu.

AddItemAggrFunc

Notation:

procedure AddItemAggrFunc(pAggrFunc: string);

Sichtbarkeit:

Public

Beschreibung:

Fügt ein Item vom Typ *Aggregatfunktion* (z. B. COUNT, SUM, MIN, MAX, usw.) hinzu.

AddItemParam

Notation:

procedure AddItemParam(pParamName: String);

Sichtbarkeit:

Public

Beschreibung:

Fügt ein Item vom Typ *Parameter* hinzu.

AddItemOperator

Notation:

procedure AddItemOperator(pType: TSFStmtAttrItemOperatorType);

Sichtbarkeit:

Public

Beschreibung:

Fügt ein Item vom Typ *Operator* hinzu.

Siehe auch TSFStmtAttrItemOperatorType

AddItemBracket

Notation:

procedure AddItemBracket(pType: TSFStmtAttrItemBracketType);

Sichtbarkeit:

Public

Beschreibung:

Fügt ein Item vom Typ *Klammer* hinzu.

Siehe auch TSFStmtAttrItemBracketType

AddItemDynamic

Notation:

procedure AddItemDynamic(pValue: String);

Sichtbarkeit:

Public

Beschreibung:

Fügt ein dynamisches Item hinzu. Der Paramter pValue gibt den gewünschten Freitext an.

AssignStmtAttr

Notation:

function AssignStmtAttr(pDestStmt: TSFStmt): TSFStmtAttr;

Sichtbarkeit:

Public

Beschreibung:

Kopiert das Attribut einschl. dessen Items.

Siehe auch TSFStmt.AssignStmt, TSFStmt.AssignStmtTo

SetItemParamNamesToList

Notation:

procedure SetItemParamNamesToList(pLst: TStrings);

Sichtbarkeit:

Public

Beschreibung:

Schreibt die Bezeichnungen aller Items vom Typ Parameter in die übergebene Liste.

Siehe auch TSFStmt.ListAttributeParams

SaveToXmlAttr

Notation:

procedure SaveToXmlAttr(pXmlAttr: TSFStmtAttrXML);

Sichtbarkeit:

Public

Beschreibung:

Speichert das Attribut in dessen Xml-Entsprechung.

Siehe auch TSFStmt.SaveToXmlDoc, TSFStmt.SaveToXmlStr

LoadFromXmlAttr

Notation:

procedure LoadFromXmlAttr(pXmlAttr: TSFStmtAttrXML; pSuspendRefs: Boolean);

Sichtbarkeit:

Public

Beschreibung:

Lädt das Attribut anhand dessen Xml-Entsprechung.

Siehe auch TSFStmt.LoadFromXml, TSFStmt.LoadFromXmlDoc

Eigenschaften

ParentStmt

Notation:

property ParentStmt: TSFStmt read mParentStmt;

Sichtbarkeit:

Public

Beschreibung:

Referenz auf die Abfrage bzw. das Statement.

AttrName

Notation:

property AttrName: String read mAttrName;

Sichtbarkeit:

Public

Beschreibung:

Der Name des Attributs, welcher bei der Generierung der Abfrage als Alias verwendet wird.

DBAttrName

Notation:

property DBAttrName: String read getDBAttrName;

Sichtbarkeit:

Public

Beschreibung:

Sofern das Attribut lediglich 1 Item besitzt, welches ein Datenbankfeld darstellt, ist hierüber der Name des Datenbankfelds verfügbar.

TSFStmt.SetStmtAttr

DBAttrTable

Notation:

property DBAttrTable: TSFStmtTable read getDBAttrTable;

Sichtbarkeit:

Public

Beschreibung:

Sofern das Attribut lediglich 1 Item besitzt, welches ein Datenbankfeld darstellt, ist hierüber der Name eine Referenz auf die Tabelle verfügbar.

TSFStmt.SetStmtAttr

DBAttrAggr

Notation:

property DBAttrAggr: String read getDBAttrAggr;

Sichtbarkeit:

Public

Beschreibung:

Sofern das Attribut lediglich 1 Item besitzt, welches ein Datenbankfeld darstellt, ist hierüber ggf. das Aggregat verfügbar.

Siehe auch TSFStmt.SetStmtAggr

SortType

Notation:

property SortType: TSFStmtSortType read mSortType write mSortType;

Sichtbarkeit:

Public

Beschreibung:

Der Sortiertyp des Attributs.

Siehe auch TSFStmt.AddOrderAttr, TSFStmtSortType

OnlyForSearch

Notation:

property OnlyForSearch: Boolean read mOnlyForSearch;

Sichtbarkeit:

Public

Beschreibung:

Gibt an, ob das Attribut nur für die Suche hinzugefügt wurde. Für Attribute, die nur zur Suche hinzugefügt wurden, können Suchbedingungen definiert werden, allerdings werden diese Attribute nicht in der SELECT-Klausel aufgeführt.

Siehe auch TSFStmt.AddConditionVal, TSFStmt.AddConditionAttr, TSFStmt.AddConditionIsNull, TSFStmt.AddConditionIsNotNull, TSFStmt.AddStmtAttr, TSFStmt.SetStmtAttr, TSFStmt.SetStmtAggr

Items

Notation:

property Items: TObjectList<TSFStmtAttrItem> read mItems;

Sichtbarkeit:

Public

Beschreibung:

Referenz auf die Items des Attributs.

TSFStmtAttrItem

Beschreibung

Klasse zur Verwaltung von Items für ein Attribut. Durch Items können bspsw. Rechenoperationen durchgeführt werden (Feld1 + Feld2 + 5)

Index

Aggr
Attr
GetAttrItemDef
ItemRef
ItemType
SaveToXmlAttrItem
Table

Funktionen

GetAttrItemDef

Notation:

function GetAttrItemDef(pWithAlias: Boolean; pExplicitCast, pEscapeLike: Boolean): String;

Sichtbarkeit:

Public

Beschreibung:

Generiert die Definition des Attributitems.

Siehe auch TSFStmt.GetSelectStmt, TSFStmt.GetDeleteStmt, TSFStmt.GetUpdateStmt, TSFStmt.GetInsertStmt, TSFStmtAttr.GetAttrDef

SaveToXmlAttrItem

Notation:

procedure SaveToXmlAttrItem(pXmlAttrItem: TSFStmtAttrItemXML);

Sichtbarkeit:

Public

Beschreibung:

Speichert das Attributitem in dessen Xml-Entsprechung.

Siehe auch TSFStmt.SaveToXmlDoc, TSFStmt.SaveToXmlStr, TSFStmtAttr.SaveToXmlAttr

Eigenschaften

Attr

Notation:

property Attr: TSFStmtAttr read mAttr;

Sichtbarkeit:

Public

Beschreibung:

Referenz auf das Attribut

ItemType

Notation:

property ItemType: TSFStmtAttrItemType read mType;

Sichtbarkeit:

Public

Beschreibung:

Der Typ des Items. Siehe TSFStmtAttrItemType

Table

Notation:

property Table: TSFStmtTable read mTable;

Sichtbarkeit:

Public

Beschreibung:

Sofern es sich bei dem Attribut um eine Datenbankfeld handelt, ist hierüber eine Referenz auf die Tabelle verfügbar.

ItemRef

Notation:

property ItemRef: Variant read getItemRef;

Sichtbarkeit:

Public

Beschreibung:

Abhängig vom Typ ist hier der Identifier verfügbar. Bei einem Item vom Typ Datenbankfeld enthält ItemRef z. B. den Feldnamen.

Aggr

Notation:

property Aggr: String read mAggr;

Sichtbarkeit:

Public

Beschreibung:

Sofern es sich bei dem Attribut um eine Datenbankfeld handelt, ist hierüber ggf. ein definiertes Aggregat für das Datenbankfeld verfügbar. Für komplexere Defintionen können Aggregate auch als separates mit dem entsprechenden Typ hinzugefügt werden.

TSFStmtCondition

Beschreibung

Klasse zur Verwaltung von Suchbedingungen (WHERE-Klausel) eines Statementgenerators.

Index

AssignStmtCondition
CondOperator
CondType
CondValue
GetConditionDef

SaveToXmlCondition
StmtAttr

Funktionen

GetConditionDef

Notation:

function GetConditionDef(pWithAliases: Boolean): String; virtual;

Sichtbarkeit:

Public

Beschreibung:

Generiert die Definition der Suchbedingung.

Siehe auch TSFStmt.GetSelectStmt, TSFStmt.GetDeleteStmt, TSFStmt.GetUpdateStmt

AssignStmtCondition

Notation:

function AssignStmtCondition(pDestStmt: TSFStmt): TSFStmtCondition; virtual;

Sichtbarkeit:

Public

Beschreibung:

Kopiert die Suchbedingung.

Siehe auch TSFStmt.AssignStmt, TSFStmt.AssignStmtTo

SaveToXmlCondition

Notation:

procedure SaveToXmlCondition(pXmlCond: TSFStmtCondXML); virtual;

Sichtbarkeit:

Public

Beschreibung:

Speichert die Suchbedingung in ihre Xml-Entsprechung.

Siehe auch TSFStmt.LoadFromXml, TSFStmt.LoadFromXmlDoc

Eigenschaften

CondType

Notation:

property CondType: TSFStmtConditionType read mType;

Sichtbarkeit:

Public

Beschreibung:

Der Typ der Suchbedingung.

Siehe TSFStmtConditionType

StmtAttr

Notation:

property StmtAttr: TSFStmtAttr read mStmtAttr;

Sichtbarkeit:

Public

Beschreibung:

Abhängig vom Typ (stmtCondTypeAttribute, stmtCondTypeValue, stmtCondTypeIsNull, stmtCondTypeIsNotNull) ist hierüber eine Referenz auf das Attribut verfügbar.

CondValue

Notation:

property CondValue: Variant read mValue;

Sichtbarkeit:

Public

Beschreibung:

Der Suchwert für die Bedingung. Abhängig vom Typ kann hier z. B. auch eine Referenz auf ein Attribut gespeichert sein.

CondOperator

Notation:

property CondOperator: String read mOperator;

Sichtbarkeit:

Public

Beschreibung:

Der Operator für die Suchbedingung.

Siehe auch Konstanten

TSFStmtConditionExists

Beschreibung

Klasse zur Verwaltung von EXISTS-Bedingungen (WHERE-Klausel) eines Statementgenerators.

Index

AssignStmtCondition
DestStmt
DestTable
GetConditionDef
RelItems
SaveToXmlCondition
SrcTable

Funktionen

GetConditionDef

Notation:

function GetConditionDef(pWithAliases: Boolean): String; override;

Sichtbarkeit:

Public

Beschreibung:

Generiert die Definition der Exists-Bedingung.

Siehe auch TSFStmt.GetSelectStmt, TSFStmt.GetDeleteStmt, TSFStmt.GetUpdateStmt

AssignStmtCondition

Notation:

function AssignStmtCondition(pDestStmt: TSFStmt): TSFStmtCondition; override;

Sichtbarkeit:

Public

Beschreibung:

Kopiert die Exists-Bedingung.

Siehe auch TSFStmt.AssignStmt, TSFStmt.AssignStmtTo

SaveToXmlCondition

Notation:

procedure SaveToXmlCondition(pXmlCond: TSFStmtCondXML); override;

Sichtbarkeit:

Public

Beschreibung:

Speichert die Exists-Bedingung in ihre Xml-Entsprechung.

Siehe auch TSFStmt.LoadFromXml, TSFStmt.LoadFromXmlDoc

Eigenschaften

DestStmt

Notation:

property DestStmt: TSFStmt read getDestStmt;

Sichtbarkeit:

Public

Beschreibung:

Zielstatement (Subselect) für die Bedingung

SrcTable

Notation:

property SrcTable: TSFStmtTable read mSrcTable;

Sichtbarkeit:

Public

Beschreibung:

Die Quelltabelle der eigenen Abfrage/des eigenen Statements, mit welcher das Zielstatement/der Subselect verknüpft wird.

DestTable

Notation:

property DestTable: TSFStmtTable read getDestTable;

Sichtbarkeit:

Public

Beschreibung:

Die Tabelle innerhalb des Zielstatements/Subselects, mit welcher die Quelltabelle verknüpft wird.

RelItems

Notation:

property RelItems: TSFStmtJoinRelItems read mRelItems;

Sichtbarkeit:

Public

Beschreibung:

Relationsattribute zur Verknüpfung von Quell- und Zieltabelle.

Siehe auch TSFStmtJoinRelItem, TSFStmtJoinRelItemType

TSFStmtDBDialectConv

Beschreibung

Klasse für datenbankspezifische Konvertierungen. Die Instanzierung von Objekten dieser Klasse erfolgt abhängig vom DBDialect, bei Oracle z. B. wird eine Instanz der Klasse TSFStmtDBDialectConvOra gebildet.

Weiter können auch eigene Konverter (abgeleitet von TSFStmtDBDialectConv) erstellt und über das Ereignis TSFStmt.OnGetDBDialectCls integriert werden.

Index

ConvertValue
EscapeLike
GetCanSelectInFrom
GetCanSelectWithoutTable
GetEndQuote
GetLastAutoValue
GetLikeWildcardMany
GetLikeWildcardSingle
GetNeedTableOnSubInFrom

GetNextAutoValue
GetStartQuote
Stmt
SupportsLikeEscape
ValueTypeForValue

Funktionen

ConvertValue

Notation:

function ConvertValue(pValue: Variant; pUsedDecSeparator: String; pExplicitCast, pEscapeLike: Boolean): String;

Sichtbarkeit:

Public

Beschreibung:

Konvertiert einen Wert für die Generierung im SQL-Statement.

Siehe auch TSFStmt.GetConvertedValue

ValueTypeForValue

Notation:

function ValueTypeForValue(pValue: Variant): TSFStmtValueType;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt den Typ eines Werts.

Siehe auch TSFStmt.GetTypeForValue, TSFStmtValueType

EscapeLike

Notation:

function EscapeLike(var pValue: String): String;

Sichtbarkeit:

Public

Beschreibung:

Prüft LIKE-Bedingungen und escapt (sofern der DBDialect Escape-Anweisungen unterstützt) diese.

Siehe auch TSFStmt.AutoEscapeLike

GetNextAutoValue

Notation:

function GetNextAutoValue(pSeqName: String = ''): String; virtual;

Sichtbarkeit:

Public

Beschreibung:

Generiert die Abfrage zur Ermittlung des nächsten Autowerts.

Siehe auch TSFStmt.GetNextAutoValueStmt

GetLastAutoValue

Notation:

function GetLastAutoValue(pSeqName: String = ''): String; virtual;

Sichtbarkeit:

Public

Beschreibung:

Generiert die Abfrage zur Ermittlung des letzten Autowerts.

Siehe auch TSFStmt.GetLastAutoValueStmt

GetCanSelectWithoutTable

Notation:

class function GetCanSelectWithoutTable(var pTableName: String): Boolean; virtual;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt, ob Select-Anweisungen ohne Angabe einer Tabelle unterstützt werden.

Siehe auch TSFStmt.GetDBDialectCanSelWithoutTab

GetCanSelectInFrom

Notation:

class function GetCanSelectInFrom(pDBDialect: TSFStmtDBDialect): Boolean; virtual;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt, ob Select-Anweisungen mit einem Subselect in der From-Klausel unterstützt werden.

Siehe auch TSFStmt.GetDBDialectCanSubInFrom

GetNeedTableOnSubInFrom

Notation:

class function GetNeedTableOnSubInFrom: Boolean; virtual;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt, ob Select-Anweisungen mit einem Subselect in der From-Klausel unterstützt werden, der (der Subselect) keine Tabelle referenziert.

Siehe auch TSFStmt.GetDBDialectNeedTableOnSubInFrom

GetStartQuote

Notation:

class function GetStartQuote: String; virtual;

Sichtbarkeit:

Public

Beschreibung:

Gibt das datenbankspezifische Anführungszeichen an, das am Anfang verwendet wird.

GetEndQuote

Notation:

class function GetEndQuote: String; virtual;

Sichtbarkeit:

Public

Beschreibung:

Gibt das datenbankspezifische Anführungszeichen an, das am Ende verwendet wird.

GetLikeWildcardSingle

Notation:

class function GetLikeWildcardSingle: String; virtual;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt das Wildcard-Zeichen für eine Like-Einzelsuche. In der Basisklasse ist dieses Zeichen "_".

Siehe auch TSFStmt.GetDBDialectLikeWildcardSingle

GetLikeWildcardMany

Notation:

class function GetLikeWildcardMany: String; virtual;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt das Wildcard-Zeichen für eine Like-Mengensuche. In der Basisklasse ist dieses Zeichen "%".

Siehe auch TSFStmt.GetDBDialectLikeWildcardMany

SupportsLikeEscape

Notation:

class function SupportsLikeEscape: Boolean; virtual;

Sichtbarkeit:

Public

Beschreibung:

Ermittelt, ob die ESCAPE-Anweisung bei einer Like-Suche unterstützt wird. Durch die ESCAPE-Anweisung kann auch nach Zeichen gesucht werden, die dem Wildcard-Zeichen entsprechen.

Siehe auch TSFStmt.GetDBDialectLikeSupportsEscape

Eigenschaften

Stmt

Notation:

property Stmt: TSFStmt read mStmt;

Sichtbarkeit:

Protected

Beschreibung:

Referenz auf den Statementgenerator

TSFBDSFormatOptions

Beschreibung

Klasse für Formatierungsoptionen.

Index

DisplayFmtCurrency
DisplayFmtDate
DisplayFmtDateTime
DisplayFmtFloat
DisplayFmtTime
EditFmtCurrency
EditFmtFloat
EditMaskDate
EditMaskDateTime
EditMaskTime
QuoteType

Eigenschaften

DisplayFmtDateTime

Notation:

property DisplayFmtDateTime: String read mDisplayFmtDateTime write mDisplayFmtDateTime;

Sichtbarkeit:

Published

Beschreibung:

Anzeigeformat für Datum- und Zeitwerte.

DisplayFmtDate

Notation:

property DisplayFmtDate: String read mDisplayFmtDate write mDisplayFmtDate;

Sichtbarkeit:

Published

Beschreibung:

Anzeigeformat für Datumswerte (ohne Zeit).

DisplayFmtTime

Notation:

property DisplayFmtTime: String read mDisplayFmtTime write mDisplayFmtTime;

Sichtbarkeit:

Published

Beschreibung:

Anzeigeformat für Zeitwerte (ohne Datum).

DisplayFmtFloat

Notation:

property DisplayFmtFloat: String read mDisplayFmtFloat write mDisplayFmtFloat;

Sichtbarkeit:

Published

Beschreibung:

Anzeigeformat für Fließkommawerte.

DisplayFmtCurrency

Notation:

property DisplayFmtCurrency: String read mDisplayFmtCurrency write mDisplayFmtCurrency;

Sichtbarkeit:

Published

Beschreibung:

Anzeigeformat für Währungswerte.

EditMaskDateTime

Notation:

property EditMaskDateTime: TEditMask read mEditMaskDateTime write mEditMaskDateTime;

Sichtbarkeit:

Published

Beschreibung:

Format/Maske zur Änderung von Datum- und Zeitwerten.

EditMaskDate

Notation:

property EditMaskDate: TEditMask read mEditMaskDate write mEditMaskDate;

Sichtbarkeit:

Published

Beschreibung:

Format/Maske zur Änderung von Datumswerten (ohne Zeit).

EditMaskTime

Notation:

property EditMaskTime: TEditMask read mEditMaskTime write mEditMaskTime;

Sichtbarkeit:

Published

Beschreibung:

Format/Maske zur Änderung von Zeitwerten (ohne Datum).

EditFmtFloat

Notation:

property EditFmtFloat: String read mEditFmtFloat write mEditFmtFloat;

Sichtbarkeit:

Published

Beschreibung:

Änderungsformat für Fließkommawerte.

EditFmtCurrency

Notation:

property EditFmtCurrency: String read mEditFmtCurrency write mEditFmtCurrency;

Sichtbarkeit:

Published

Beschreibung:

Änderungsformat für Währungswerte.

QuoteType

Notation:

property QuoteType: TSFBDSQuoteType read mQuoteType write mQuoteType;

Sichtbarkeit:

Published

Beschreibung:

Angabe, ob Identifier bei Datenbankabfragen in Anführungszeichen gesetzt werden.

Typen/Konstanten

Index

SFSTMT_OP_EQUAL
SFSTMT_OP_EXISTS
SFSTMT_OP_GREATER
SFSTMT_OP_GREATEREQUAL
SFSTMT_OP_IN
SFSTMT_OP_LESS
SFSTMT_OP_LESSEQUAL
SFSTMT_OP_LIKE
SFSTMT_OP_NOT_EXISTS
SFSTMT_OP_NOT_IN
SFSTMT_OP_NOT_LIKE
SFSTMT_OP_NOTEQUAL
SFSTMTAGGR_AVG
SFSTMTAGGR_COUNT
SFSTMTAGGR_MAX
SFSTMTAGGR_MIN
SFSTMTAGGR_SUM
TSFBDSAutoValueGetMode
TSFBDSAutoValueOption
TSFBDSAutoValueOptions
TSFBDSExecParamsType
TSFBDSGetAutoValueCls

TSFBDSRecordCompareResult
TSFBDSRecordUpdateState
TSFBDSRefreshMode
TSFBDSSetParamsEvt
TSFBSDRecordCompareEvent
TSFBusinessDataChanged
TSFConnectionDBType
TSFConnectionType
TSFConnectorDSCreatedEvt
TSFQueryActionType
TSFStmtAttrItemBracketType
TSFStmtAttrItemOperatorType
TSFStmtAttrItemType
TSFStmtAttrItemValueType
TSFStmtConditionType
TSFStmtDBDialect
TSFStmtGenInfo
TSFStmtGenInfos
TSFStmtGenSelectEvent
TSFStmtGetDialectConvEvent
TSFStmtJoinRelItem
TSFStmtJoinRelItems
TSFStmtJoinRelItemType
TSFStmtJoinType
TSFStmtQuoteType
TSFStmtSortType
TSFStmtTableSearchType
TSFStmtTableSingleSearchTypes
TSFStmtValueType

Konstanten

SFSTMT_OP_EQUAL = '=';
SFSTMT_OP_NOTEQUAL = '<>';
SFSTMT_OP_LESSEQUAL = '<=';
SFSTMT_OP_GREATEREQUAL = '>=';
SFSTMT_OP_LESS = '<';
SFSTMT_OP_GREATER = '>';
SFSTMT_OP_LIKE = 'LIKE';
SFSTMT_OP_NOT_LIKE = 'NOT LIKE';
SFSTMT_OP_IN = 'IN';
SFSTMT_OP_NOT_IN = 'NOT IN';
SFSTMT_OP_EXISTS = 'EXISTS';
SFSTMT_OP_NOT_EXISTS = 'NOT EXISTS';

SFSTMTAGGR_COUNT = 'count';
SFSTMTAGGR_MIN = 'min';
SFSTMTAGGR_MAX = 'max';
SFSTMTAGGR_AVG = 'avg';
SFSTMTAGGR_SUM = 'sum';

Typen

```
TSFConnectionType =
   (ctFireDac,
     ctDBExpress,
     ctInterbase,
     ctADO
   );


TSFConnectionDBType =
   (dbtDB2,
     dbtFB,
     dbtIB,
     dbtMSSQL,
     dbtMySQL,
     dbtOra,
     dbtSQLLite,
     dbtPG,
     dbtMSAcc,
     dbtAdvantage,
     dbtInformix,
     dbtAnywhere,
     dbtSybase,
     dbtUnknown
   );


TSFQueryActionType =
   (
     atSelect,
     atModify
   );


TSFStmtJoinType =
   (stmtJoinTypeInner,
     stmtJoinTypeOuter,
     stmtJoinTypeROuter,
     stmtJoinTypeNone);


TSFStmtJoinRelItemType =
   (stmtJoinRelItemAttr,
     stmtJoinRelItemValue);
```

```
TSFStmtJoinRelItem = record
   riSrcType: TSFStmtJoinRelItemType;
   riSrcValue: Variant;
   riDestType: TSFStmtJoinRelItemType;
   riDestValue: Variant;
  end;


TSFStmtJoinRelItems = Array of TSFStmtJoinRelItem;


TSFStmtAttrItemType =
   (stmtAttrItemTypeDbField,
     stmtAttrItemTypeValue,
     stmtAttrItemTypeParameter,
     stmtAttrItemTypeStmt,
     stmtAttrItemTypeAggrFunc,
     stmtAttrItemTypeOpPlus,
     stmtAttrItemTypeOpMinus,
     stmtAttrItemTypeOpMultiply,
     stmtAttrItemTypeOpDivide,
     stmtAttrItemTypeBracketOpen,
     stmtAttrItemTypeBracketClose,
     stmtAttrItemTypeDynamic);


TSFStmtAttrItemOperatorType = stmtAttrItemTypeOpPlus..stmtAttrItemTypeOpDivide;


TSFStmtAttrItemBracketType =
stmtAttrItemTypeBracketOpen..stmtAttrItemTypeBracketClose;


TSFStmtAttrItemValueType = stmtAttrItemTypeValue..stmtAttrItemTypeParameter;


TSFStmtConditionType =
   (stmtCondTypeValue,
     stmtCondTypeAttribute,
     stmtCondTypeOpen,
     stmtCondTypeClose,
     stmtCondTypeAnd,
     stmtCondTypeOr,
     stmtCondTypeIsNull,
     stmtCondTypeIsNotNull,
     stmtCondTypeUndefined);
```

```
TSFStmtValueType = (
   stmtValTypeNumeric,
   stmtValTypeDate,
   stmtValTypeTime,
   stmtValTypeDateTime,
   stmtValTypeBool,
   stmtValTypeString,
   stmtValTypeOther
 );


TSFStmtSortType =
   (stmtSortTypeAsc,
     stmtSortTypeDesc);


TSFStmtGenInfo =
   (stmtGenSelect,
     stmtGenFrom,
     stmtGenWhere,
     stmtGenGroup,
     stmtGenOrder);


TSFStmtGenInfos = set of TSFStmtGenInfo;


TSFStmtDBDialect =
   (stmtDBDDflt,
     stmtDBDOra,
     stmtDBDDB2,
     stmtDBDIfx,
     stmtDBDAcc,
     stmtDBIB,
     stmtDBFB,
     stmtDBDSQLite,
     stmtDBDPG,
     stmtDBDMySQL,
     stmtDBDMSSQL,
     stmtDBDAdvantage,
     stmtDBDAnywhere,
     stmtDBDSybase);
```

```
TSFStmtTableSearchType =
   (stmtTableSearchAll,
     stmtTableSearchOnlyAlias,
     stmtTableSearchOnlyIdentifier,
     stmtTableSearchOnlyName);


TSFStmtTableSingleSearchTypes = stmtTableSearchOnlyAlias..stmtTableSearchOnlyName;


TSFStmtQuoteType = (
   stmtQuoteTypeAuto,
   stmtQuoteTypeAll,
   stmtQuoteTypeNone
 );


TSFBDSRecordUpdateState = (
     usUnmodified,
     usInserted,
     usModified,
     usDeleted
   );


TSFBDSRecordCompareResult = (
     compareResultLess,
     compareResultEqual,
     compareResultGreater,
     compareResultUndefined
   );


TSFBDSRefreshMode = (
     refreshModeRow,
     refreshModeFull
   );
```

```
TSFBDSAutoValueOption = (
    avoExecute,
    avoNeedSequence,
    avoNeedTable,
    avoExecWhenAuto,
    avoPreventWhenAuto,
    avoExecWhenExplicitByDBMS,
    avoPreventWhenExplicitByDBMS
  );


TSFBDSAutoValueOptions = set of TSFBDSAutoValueOption;


TSFBDSAutoValueGetMode = (
    avGMAfterInsert,
    avGMBeforePost,
    avGMAfterPost
  );


TSFBDSExecParamsType =
  (exPrmsTypeSelect,
    exPrmsTypeDelete);
```

Funktionen/Events

```
TSFConnectorDSCreatedEvt = procedure(pDataSet: TDataSet; pActionType:
TSFQueryActionType) of object;


TSFStmtGenSelectEvent = procedure(pStmt: TSFStmt; pLevel, pSubId, pUnionId: Integer) of
object;


TSFStmtGetDialectConvEvent = function(pDBDialect: TSFStmtDBDialect):
TSFStmtDBDialectConvCls of object;


TSFBDSSetParamsEvt = procedure(pType: TSFBDSExecParamsType; pParams:
TCollection) of object;


TSFBSDRecordCompareEvent = function(CompareRecordFrom, CompareRecordTo:
TSFBDSCompareRecord): TSFBDSRecordCompareResult of object;
```

TSFBDSGetAutoValueCls = function(pFieldName: String; pAutoDetected: Boolean): TSFBDSAutoValueGeneratorCls;

TSFBusinessDataChanged = procedure(pOldDS, pNewDS: TSFBusinessData) of object;