ERP - система «КОМПАС»

Руководство системного администратора

Приложение 1. Краткое описание и правила использования языка бизнес-процедур ERP-системы «КОМПАС»

Содержание

1. ОБЩИЕ СВЕДЕНИЯ 3

2. ВЫРАЖЕНИЯ 3

3. ПЕРЕМЕННЫЕ 4

4. КЛАССЫ ОБЪЕКТОВ 5

4.1. Набор данных 6
4.1.1. Свойства 6
4.1.2. Методы 7
4.2. Табличная форма 9
4.2.1. Дополнительные свойства: 9
4.2.2. Дополнительные методы: 9
4.3. Запросная форма 11
4.3.1. Дополнительные свойства 11
4.3.2. Дополнительные методы 11
4.4. Запросная форма с возможностью редактирования 11
4.5. Реестр документов 12
4.5.1. Дополнительные свойства 12
4.5.2. Дополнительные методы 12
4.6. Серверная временная таблица 13
4.6.1. Дополнительные свойства: 13
4.7. Отчет 13
4.7.1. Свойства 13
4.7.2. Методы: 13
4.8. Список 14
4.8.1. Свойства 14
4.8.2. Методы 14
4.9. Транслятор формул 17
4.10. Период времени 17
4.10.1. Свойства 17
4.10.2. Методы: 17
4.11. Текстовый файл (TextFile) 18
4.11.1. Свойства 18
4.11.2. Методы 18
4.12. Транзакция (Transaction) 18
4.12.1. Свойства 18
4.12.2. Методы 18
4.13. Работа с почтой (PopMail) 18
4.13.1. Свойства 19
4.13.2. Методы 19
4.14. Блокировки (LockOpers) 19
4.14.1. Методы 20
4.15. XML 21
4.15.1. Методы 21
4.16. XML Node 21
4.16.1. Свойства 22
4.16.2. Методы 22
4.17. Служебные классы 22

5. ОПЕРАТОРЫ И ФУНКЦИИ 23

5.1. Функции ввода/вывода 23
5.2. Математические функции (возвращают число) 27
5.3. Строковые функции (возвращают текст) 28
5.4. Функции для работы с датами 31
5.5. Функции для работы с объектами 31

1. Общие сведения

Язык бизнес-процедур ERP-системы «КОМПАС» похож на широко распространенный язык Basic.

Каждый оператор пишется на отдельной строке, то есть в качестве разделителей операторов используется перевод строки.

В бизнес-процедурах используются:

- Выражения
- Переменные
- Конструкции
- Классы объектов
- Операторы и функции
    - Ввода/Вывода
    - Математические
    - Строковые
    - Для работы с датами
    - Для работы с объектами

В этом документе приведен классы и функции языка бизнес-процедур, используемых во всем модулях ERP-системы «КОМПАС». Кроме того, отдельные модули могут иметь собственные функции, используемые в бизнес-процедурах. Описание этих функций приведено в справочных системах модулей.

2. Выражения

В выражении допускается использовать:

- константы. В качестве констант могут применяться все константы, используемые в конструкторе формул, например в качестве логических констант определены true, false, да, нет, ...;
- функции и переменные;
- операции:

-	вычитание:	a-b
+	сложение:	a+b
*	умножение:	a*b
/	деление:	a/b
%	остаток от деления:	a%b
^	возведение в степень:	a^b
<	меньше:	a<b
<=	меньше либо равно:	a<=b
<>	не равно:	a<>b
>=	больше либо равно:	a>=b

>	больше:	a<b
=	равно: a=b	

MOD	остаток от деления:	a MOD b
NOT	логическое НЕ:	NOT a
OR	логическое ИЛИ:	a OR b
XOR	исключающее ИЛИ:	a XOR b
AND	логическое И:	a AND b

Все бинарные операции необходимо использовать с переменными одного типа.

Приоритеты операций:

1	NOT
2	^
3	*, /, %, MOD
4	+, -
5	<, <=, <>, >=, >, =
6	AND
7	OR, XOR

3. Переменные

Переменная - это единица данных определенного типа. Каждая переменная имеет ИМЯ и ТИП.Имя переменной необходимо, чтобы отличать переменные друг от друга.Имя может состоять из любых символов латинского и русского алфавитов, @ и _Имя переменной должно начинаться только с символа, _ или @.

Типы переменных:

- **число: целое, вещественное**
- **строка**
- **дата/время**
- **логическое**
- **массив**
- **объект некоторого класса**

Переменные могут быть существующими уже при запуске процедуры либо переменные могут создаваться операторами самой процедуры.

Обратите внимание! Указание объекта "m" позволяет отличить *переменную* от *поля* в процедурах-событиях на табличную форму или элемент МОМ. Если префикс "m." есть, то это переменная, если нет - то это поле.

Номенклатура переменных, существующих при вызове, зависит от того, из какого контекста вызвана бизнес-процедура. В большинстве случаев существующими являются переменные, совпадающие с именами полей той табличной формы, из которой вызвана бизнес-процедура. При этом правила использования свойств и методов этой табличной формы аналогичны случаю, когда текущий объект *Табличная форма* назначается специально с помощью функции *Use*.

При вызове процедуры для следующих событий в экранной форме:

· нажатие на кнопку,

· модификацию данных в таблице, входящей в экранную форму,

в виде переменных дополнительно становятся доступными элементы массива общих мест.

Например:

```
_FIELDS@TEST=10_FIELDS@TEST=_FIELDS@TEST+1
```

Особые правила действуют для элементов массива общих мест, имеющих тип "дата". Обычно таким элементам надо присваивать другое значение типа "дата". Исключение составляет случай, когда необходимо очистить дату. В этом случае следует присвоить элементу любое пустое значение: ноль или пустую строку, например:

```
_FIELDS@DATE_N=""
```

Следует учитывать, что для того, чтобы изменения в элементе массива общих мест отразились в экранной форме, следует использовать не прямое присвоение элементу массива, а специальный метод **SetValue**, например:

```
_FIELDS@TEST.SetValue(10)_FIELDS@TEST.SetValue(_FIELDS@TEST+1)
```

Следует также учитывать, что в экранной форме могут быть отображены вычисляемые колонки табличной формы, значения в которых могут изменяться по мере работы бизнес процедуры. Для того, чтобы эти изменения также отразились в экранной форме, необходимо предусмотреть в процедуре явное изменение значений соответствующих элементов массива общих мест с помощью метода SetValue.

Если процедура назначена в качестве реакции на события в табличной форме, которая может использоваться как в экранной форме, так и за пределами экранной формы, в тексте процедуры важно знать, являются ли доступными элементы массива общий мест. Для этого можно использовать функцию FindELMOM. В качестве параметра в функцию передается текст, являющийся именем элемента массива общих мест. Если элемент доступен функция возвращает значение истина. Иначе функция возвращает значение ложь. Элементы массива общих мест, являющиеся таблицами, считаются доступными только в том случае, если соответствующая таблица имеется в экранной форме и уже была инициализирована.

Следует также учитывать, что в большинстве случаев, когда табличная форма открывается в качестве подчиненной к другой табличной форме, в процедурах доступны элементы массива общих мест, представляющие родительскую табличную форму. То есть в этом случае можно использовать те же процедуры, которые написаны для случая включения табличной формы в экранную форму.

4. Классы объектов

Переменные языка могут иметь тип: объект некоторого класса. Такая переменная создается одной из функций языка и имеет свойства (аналогичны переменным) и методы (аналогичны функциям). Доступ к отдельным свойствам и методам осуществляется с помощью имени переменной, точки и имени свойства или метода, например:

```
a=TForm "SPR_DOK"a.goFirstShowModal a
```

В настоящее время в синтаксисе языка предусмотрены следующие классы:

Набор данных

Табличная форма
Запросная форма
Запросная форма с возможностью редактирования
Реестр документов
Серверная временная таблица
Отчет
Список
Транслятор формул
Период времени
Текстовые файлы (TTextFile),
Служебные классы

4.1. Набор данных

Объекты этого класса создаются функцией ***SQL***.

4.1.1. Свойства

поля - переменные, одноименные полям данных. Тип соответствует типу поля. Имеют метод ***Find* значение** (тип значения должен совпадать с типом поля);

Bof - переменная равна 1, когда курсор стоит на первой записи;

Eof - переменная равна 1, когда курсор стоит на последней записи;

Count - количество записей.

CountWithFilter - отличие от функции count заключается в том, что новая учитывает все наложенные на набор данных фильтры/

Filter - фильтр, доступен для чтения и для записи.

ServerFilter - серверный фильтр. Обратите внимание, что использование этого свойства возможно только в том случае, если набор данных связан с какой-либо табличной формой: созданной с помощью функции TForm или полученной в качестве UsedObject при вызове бизнес-процедуры в качестве реакции на события табличной формы, либо при нажатии на кнопки табличной формы. Значением свойства является текст, написанный по правилам языка SQL, который может быть использован в предложении WHERE запроса на выборку данных для табличной формы. Во избежание двусмысленности, рекомендуется все поля основной таблицы в тексте серверного фильтра снабжать префиксом "MAIN."

Любое изменение свойства ServerFilter приводит к перезагрузке данных в табличной форме. Если для свойства задано пустое значение, выборка данных при перезагрузке осуществляется без дополнительных условий. Иначе исползуется условие отбора, описанное в серверном фильтре.

TableName - имя таблицы, по которой построена форма. Это свойство может быть использовано только для табличных форм.

State - текстовая строка, характеризующая состояние набора данных:

"dsInsert" - состояние добавления записи;
"dsEdit" - редактирование старой записи;
"dsBrowse" - перемещение по записям;
"dsUnknown" - остальные состояния.

@Temp.Value - служебная переменная, связанная с данным набором данных и доступная из всех бизнес процедур, которые с этим набором данных работают. Переменная создается при создании первого объекта, ссылающегося на этот набор данных. Уничтожается при закрытии набора данных. Обычно эта переменная используется для обмена данными между процедурами, вызываемыми как реакция на события табличной формы.

Posted - устанавливается значение «true», если во временную таблицу в составе экранной формы изменения вносились запросами. Если использовался только метод SQLPost, данное свойство можно не использовать.

4.1.2. Методы

goFirst - перемещение в начало;

goLast - перемещение в конец;

goNext - перемещение на следующую запись;

goPrev - перемещение на предыдущую запись;

Insert - переход в режим вставки;

Delete - удаление записи;

Edit - перевод записи в состояние редактирования;

Post - сохранение изменений;

Cancel - отмена изменений;

Refresh - обновление данных;

Close - закрытие формы.
Данный метод имеет смысл использовать в том случае, если набор данных является табличной формой. Также следует учитывать, что метод не приводит к немедленному закрытию табличной формы, а лишь устанавливает признак того, что при уничтожении объекта форма должна быть закрыта. Объект может быть уничтожен путем присвоения переменной другого значения. Кроме того, при завершении бизнес-процедуры, все объекты уничтожаются автоматически.

Find *"имя поля", значение*	- поиск.
RestrictField *"значение"*	позволяет получить имя поля, которое используется для ограничения по ключу. Параметр: порядковый номер поля в ограничении (начиная с 1). Если ограничение не установлено или число полей меньше указанного, функция возвращает пустую строку.
RestrictValue *"значение"*	позволяет получить значение, которое используется для ограничения по ключу. Параметр: порядковый номер поля в ограничении (начиная с 1). Если ограничение не установлено или число полей меньше указанного, функция возвращает пустую строку. Обратите внимание: значение возвращается в том виде, в котором оно хранится в TRestrictInfo, то есть в виде изображения.
SetMaxStr *"имя поля", значение*	**-** параметры: имя поля с номерами строк, максимальный номер, записанный в поле. Вызывается, если изменения во временную таблицу в составе экранной формы вносились запросами или с помощью **SQLPost,** а в таблице имеется поле с номерами строк.
LoadBlob "имя поля", "имя файла"	- чтение из файла. Параметры: имя BLOB-поля, имя файла. Этот метод следует использовать только когда Набор данных (Dataset) находится в режиме Edit или Insert.
SaveBlob "имя поля", "имя файла"	- запись в файл. Параметры: имя BLOB-поля, имя файла
PrintForm, *"псевдоним печатной формы", "число копий при печати", "список параметров печатной формы"*	- формирование печатной формы по шаблону с последущим диалогом. По завершении процесса формирования документа производится стандартный диалог с оператором. Процедура продолжает работу только после закрытия диалогового окна. Параметр «список параметров печатной формы» - необязательный. Параметры, упомянутые в этом списке, исключаются из диалога ввода параметров. Если все параметры упомянуты в списке, диалог ввода параметров не выводится.
MakeForm, *"псевдоним печатной формы", "полное имя файла"*	- формирование печатной формы и сохранение в заданном файле.По завершении процесса формирования документа окно мониторинга закрывается, файл с указанным полным именем сохраняется, выполнение процедуры продолжается.
GetValue, *"имя поля"*	позволяет получить значение поля даже в том случае, если его имя совпадает с именем свойства или функции
SetValue, *"имя поля","значение"*	позволяет изменить значение поля даже в том случае, если его имя совпадает с именем свойства или функции

GetCheckIns *"список"* заполняет параметр списком уникальных ключей для отмеченных строк набора данных. Метод заполняет список только в том случае, если набор данных является табличной формой на основе компонентов TTfWndForm или TTfCtl. В остальных случаях переданный список очищается.

4.2. Табличная форма

Объекты этого класса создаются функцией ***TForm***.
Наследует все свойства и методы класса ***Набор данных.***

4.2.1. Дополнительные свойства:

Caption - заголовок табличной формы;

CheckIns - проверка отметки строк: возвращает true, если строка в ТФ отмечена, и false, если строка не отмечена;

SelectedField - при использовании в выражении свойство равно имени поля для текущей колонки табличной формы. Если табличная форма еще не открыта, свойство пусто. При использовании свойства в левой части оператора присваивания устанавливается текущая колонка, которая соответствует полю с указанным именем. Если табличная форма еще не открыта или в ней нет такого поля, фиксируется ошибка. Если поле есть, но нет колонки для него, ошибка не фиксируется, но и текущая колонка не меняется;

IsOpen - возвращает true, если табличная форма открыта, и false - если закрыта;

CalendarView - вид "*календарь*". Позволяет разместить в левой части формы элемент *календарь* для фильтрации записей табличной формы по датам. Может принимать значения от 0 до 4:

0 = Один день;
1 = Рабочая неделя (5 дней);
2 = Неделя;
3 = Месяц;
4 = Два календаря;

Calendar1Date - дата в первом календаре;

Calendar2Date - дата во втором календаре.

4.2.2. Дополнительные методы:

Show - отобразить не модально;

ShowModal - отобразить модально;

ShowLookup - выбрать значение;

ShowSF - открыть экранную форму;

Order *строка* сортировка;

- накладывает ограничение по ключу. Наряду с именами полей можно использовать служебные имена:

Restrict *"поле1=значение1" [, "поле2=значение2"]*	HideKeyColumns - следует ли скрывать колонки с ключевыми полями; KeyChkPrefix - следует ли для последнего из ключевых полей (обязательно символьного) включать режим отбора по префиксу. Значениями для этих служебных имен могут "0" (ноль) - нет, или "1" - да. Любое другое значение воспринимается как "1".
Mail *получатель, приложение, тема, содержание*	- посылает сообщение по электронной почте, использующее текущую строку табличной формы. Все параметры - текстовые. Данная функция может использоваться не во всех модулях ERP-системы «КОМПАС».
ShowDetail *идентификатор кнопки, для которой настроена связь с подчиненной табличной формой*	при вызове этого метода открывается подчиненная табличная форма, т.е. действие совпадает с нажатием на кнопку с переданным идентификатором. Если передан идентификатор кнопки, которая не открывает подчиненную табличную форму, выдается сообщение "Ошибка в параметрах функции".
ShowMemo *псевдоним поля*	При вызове этого метода для указанного в качестве параметра поля открывается многострочный редактор
ShowFilterDlg *[префикс]*	- вызывает диалог по установке условия фильтра для табличной формы, аналогичный диалогу по клавише F6. Однако, результаты данного диалога не приводят к фильтрации табличной формы, а возвращаются в качестве результата работы функции в виде текстовой строки, которую можно использовать в SQL-выражении WHERE для выборки данных из таблицы. Необязательный параметр задает префикс таблицы, используемый в SQL-выражении. Если оператор нажал кнопку *Отмена*, функция возвращает пустую строку. Если оператор задал условие, которое не может быть преобразовано к SQL-выражению, функция возвращает строку, состоящую из одного слова "нет".
xRefresh *псевдоним*	- выполняет обновление данных в табличных формах, связанных с указанной таблицей. Текущая табличная форма при этом не обновляется, даже если она связана с той же таблицей. При необходимости обновить текущую табличную форму, следует использовать метод ***Refresh***.

Обратите внимание!

Использование класса ***Табличная форма*** имеет особенности для некоторых табличных форм, для которых в программе предусмотрена специальная обработка. К таким табличным формам относятся:

- табличная форма CH_70_S (расчетная ведомость);
- табличная форма FONDRAB (фонды рабочего времени);

- табличная форма GRIDN (перечень назначений работника);
- табличная форма GRIDOTP (перечень отпусков и неявок работника);
- любые табличные формы, основанные на таблице LIC_CH (лицевой счет работника), в том числе - расчетный листок (табличная форма LIC_CH_M);
- табличная форма NAZN (штатная расстановка);
- табличная форма OTPUSK (журнал отпусков и других неявок);
- табличная форма SHTRAS_M (штатное расписание по работникам);
- табличная форма SHTRAS_W (штатное расписание по должностям);
- табличная форма SPR_70 (свод по заработной плате);
- табличная форма TABRAB (табель учета рабочего времени).

Хотя объект для указанных табличных форм может быть создан с помощью функции TForm, большинство методов класса «Табличная форма» для такого объекта не действуют. Допускается использование только метода Show, который, однако, никаких действий не выполняет, так как показ формы в немодальном режиме осуществляется сразу после вызова функции TForm. У такого объекта отсутствуют свойства, соответствующие полям таблицы, а использование свойств Caption, Filter и некоторых других не оказывает никакого влияния на открытую табличную форму.

Если в бизнес процедуре необходимо осуществить выбор из табличной формы как из справочника, рекомендуется создавать объект класса «Табличная форма» не с помощью функции TForm, а с помощью функции TLookupForm. Объект, созданный с помощью функции TLookupForm, может использовать только для выбора из справочника с помощью метода ShowLookup, тогда как методы Show, ShowModal и ShowSF у него отсутствуют. С другой стороны, такой объект ведет себя одинаково для любых табличных форм, в том числе - для перечисленных выше в настоящей статье. Кроме того, таблица для такого объекта открывается в режиме только для чтения, что ускоряет работу программы.

4.3. Запросная форма

Объекты этого класса создаются функцией ***QForm***.

Наследует все свойства и методы класса ***Набор данных***.

4.3.1. Дополнительные свойства

Caption - заголовок табличной формы.

4.3.2. Дополнительные методы

Show	- отобразить не модально;
ShowModal	- отобразить модально;
ShowLookup *“строка”*	- выбрать значение;
Order *“строка”*	- сортировка.

4.4. Запросная форма с возможностью редактирования

Объекты этого класса создаются функцией ***QFormLive***.

Наследует все свойства и методы класса ***Запросная форма (QForm)***. От базового класса ***QForm*** отличается тем, что запрос открывается с параметром *RequestLive=true.*

4.5. Реестр документов

Объекты этого класса создаются функцией ***Reestr***.

Имеет аналоги методов класса ***Табличная форма***. В качестве свойств вместо имен полей используются элементы *массива общих мест*.

4.5.1. Дополнительные свойства

Table	- псевдоним таблицы;
TForm	- псевдоним табличной формы;
Type	- тип документов в реестре;
DocName	- название типа документа реестра;
DocsName	- название типа документов во множественном числе;
Filter	- фильтр, доступен для чтения и для записи.

4.5.2. Дополнительные методы

Append	- новый документ (прозрачный режим);
Delete	- удалить документ из реестра;
DelOplat	- удалить документ из журнала взаиморасчетов;
DelProv	- удалить проводку по документу из журнала операций;
DubleDoc	- создание нового документа по текущему;
GetNextNDok	- возвращает новый номер документа для реестра в соответствии с правилами, настроенными в справочнике типов документов. Перед использованием этой функции реестр должен быть открыт для нужного типа документа и курсор должен находиться на какой-то записи, потому что метод GetNextNDok отрабатывает в зависимости от даты в текущем документе.
GetReestrNextNDok [параметр1, параметр2]	- функция принимает в качестве параметров тип документа и дату (оба параметра обязательны) и возвращает следующий новый номер документа по правилам, настроенным в справочнике типов документов
Index	- отсортировать реестр;
Mail	**-** *получатель, приложение, тема, содержание* - посылает текущий документ по электронной почте. Все параметры - текстовые. Данная функция доступна не во всех модулях ERP-системы «КОМПАС».

NewDoc	- создание нового документа;
ProvREE	- провести документ в журнале операций;
SaveDoc	- сохранить документ в реестре;
SetInBook	- включить документ в книгу покупок или продаж;
SetOplat	- включить документ в журнал взаиморасчетов;
ShowSF	- открыть экранную форму. В связи с особенностями класса, перед вызовом метода ShowSF автоматически вызывается метод *SaveDoc*, то есть, перед вызовом экранной формы изменения записываются в таблицу и текущая запись сохраняется;
ViewOplat	- просмотреть журнал взаиморасчетов по документу;

4.6. Серверная временная таблица

Объекты этого класса создаются функцией ***ServerTempTable***.

Объект соответствует таблице, указанной при вызове функции. Таблица существует до конца выполнения бизнес-процедуры.

Объект наследует все свойства и методы класса ***Набор данных***.

4.6.1. Дополнительные свойства:

TableName - имя временной таблицы.
KeyField - имя поля для первичного ключа.

Пример:

```
Trsh = ServerTempTable("TMPRASHR")
RunSQL "INSERT INTO " + Trsh.TableName + " (PART_NO, STR_NO, NOM_NO, SKL_NO, MOL) SELECT
PART_NO,STR_NO, NOM_NO,MOL FROM RASHOD"
```

4.7. Отчет

Объекты этого класса создаются функцией ***Report.***

4.7.1. Свойства

колонки - переменные, одноименные колонкам отчета;
Title - заголовок отчета;
Footer - подвал отчета.

4.7.2. Методы:

Start	- начать формирование отчета. Если в качестве параметра метода Start передать строку "Excel", отчет будет формироваться в формате MS Excel.;
Next	- перейти к следующей записи;
Stop	- закончить формирование отчета;
Options	- установить опции отчета (см. ниже);

Print - напечатать отчет.

При помощи объектов этого класса можно создавать отчеты "руками", т.е. запись за записью. Это избавляет разработчиков от необходимости сначала копировать данные во временную таблицу, а затем строить из нее отчет.

Пример работы:

```
Spr_Dok=TForm "SPR_DOK"
Rep=Report "SPR_DOK"    ' Отчет называется SPR_DOK,
Rep.Start          '  Начало заполнения отчета
DO UNTIL Spr_Dok.Eof      '  Идем по SPR_DOK'у
Rep.TIP = Spr_Dok.TIP      '  Заполняем поименованные колонки значениями из таблицы
Rep.NAME_D=Spr_Dok.Name_D
Rep.Next           '  Переходим на следующую запись отчета
Spr_Dok.goNext  '  Переходим на следующую запись таблицы
LOOP
Rep.Stop           ' Закончили возиться с отчетом
Rep.Print          ' Вывели его на печать
```

У объекта класса **Отчет** есть свойство - ***Options***. Можно использовать следующие опции:

"Normal" - нормальный шрифт
"Bold" - жирный шрифт
"Italic" - наклонный шрифт
"LineTop" - верхняя линия
"BoldLineTop" - двойная верхняя линия
"LineBottom" - нижняя линия
"BoldLineBottom" - двойная нижняя линия

Пример:

```
a=Report "SPR_DOK"
a.Options "Bold","BoldLineTop","BoldLineBottom"
```

Примечание:
Опция, будучи включена один раз, действительна до тех пор, пока ее не заменит какая-нибудь другая. Например, если включить опцию «Bold» все следующие строчки отчета будут жирными до тех пор, пока не будет включена опция «Normal».

4.8. Список

Объекты этого класса создаются функцией ***ListBox*.**

4.8.1. Свойства

Count - количество элементов.

4.8.2. Методы

Clear - очистить список;

Add*(строка)* - добавляет элемент. Если строка имеет вид "name=value", левую часть можно получить из свойства Name, вторую - Value;

Load*(имя файла)* - загружает в список строки из указанного файла;

Save*(имя файла)*	- сохраняет строки из ListBox в указанный файл;
String*(число)*	- возвращает элемент, индекс которого задан числом. Обратите внимание: индексация элементов начинается с нуля;
Name(*число)*	- левая часть элемента, индекс которого задан числом;
Value(*число)*	- правая часть элемента, индекс которого задан числом;
IndexOf(*строка)*	- индекс строки в списке или минус 1, если строка в списке не найдена;
IndexOfName(*строка)*	- индекс элемента, левая часть которого равна данной строке, или минус 1, если такой элемент не найден;
SetValue*(число,строка)*	- изменяет правую часть элемента с указанным индексом: после выполнения оператора правая часть будет равна указанной строке;
Delete число	- удаляет элемент, индекс которого задан числом;
ProtocolView *(заголовок,параметр,ширина,дата)*	- выводит на экран окно с изображением текущего содержания списка. Обычно эта возможность используется для показа протокола с результатами работы процедуры. Все параметры метода являются необязательными. *Параметры:* 1. *Текст*: заголовок, помещаемый в верхней части окна. Если не задан, то подразумевается текст "Протокол выполнения". 2. *Число*: определяет специальные режимы вывода окна. По умолчанию равен нулю. Для включения специальных режимов может принимать одно из следующих значений или их сумму: **1** - показывать дату в окне протокола. По умолчанию дата в окне отсутствует; **2** - использовать только шрифты с фиксированным шагом. По умолчанию могут также использоваться пропорциональные шрифты. Шрифт может быть изменен в процессе просмотра протокола, однако, если задан параметр 1, можно выбрать только шрифт с фиксированным шагом; **4** - скрыть кнопку «Отмена» и оставить только кнопку «ОК»; **8 -** скрыть кнопку «ОК» и оставить только кнопку «Отмена». Следует обратить внимание, что если функция включена в состав выражения, она возвращает значение «истина», если оператор нажал кнопку «ОК», в противном случае функция возвращает значение «ложь». Если функция использована в качестве оператора, кнопка «Отмена» скрыта по умолчанию.

3. *Число*: первоначальная ширина окна в пикселях. Если ширина не задана или меньше 1, окно имеет стандартную ширину.
4. *Текст*: изображает дату, которая указывается в верхней части окна, если 2-й параметр включает значение 1. По умолчанию используется текущая календарная дата.

AskChoice(*заголовок, текст, начальная, группа, кнопка, маска)*

- выбор вариантов. При организации диалога каждый элемент списка является отдельным вариантом. Все параметры метода являются необязательными.

Параметры:

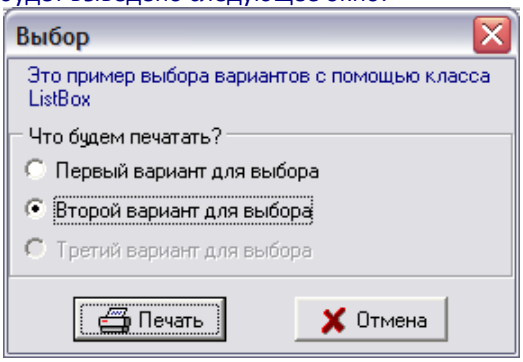
1. *Текст*: заголовок, помещаемый в верхней части окна выбора. Если не задан, то подразумевается текст "Выберите вариант".
2. *Текст*: поясняющий текст, который располагается в окне выбора над перечнем вариантов.
3. *Число*: номер варианта (начиная с нуля), который устанавливается по умолчанию. Если не задан, по умолчанию устанавливается первая альтернатива.
4. *Текст*: располагается в разрезе перечня альтернатив. Если не задан, то подразумевается текст "Что будем делать?".
5. *Текст*: располагается на исполнительной кнопке ОК. Если задан текст "Печать", изображение на кнопке автоматически изменяется.
6. *Число*: маска для отключения альтернатив. Для первой альтернативы используется 1, для второй - 2, для третьей - 4 и т.п. Для того, чтобы отключить одновременно несколько альтернатив, надо задать сумму соответствующих значений.

Возвращаемое значение: номер варианта, который выбрал оператор, начиная с нуля. Если оператор нажал кнопку *Отмена*, возвращается -1.

Пример:

```
list = ListBox
list.Add("Первый вариант для выбора")
list.Add("Второй вариант для выбора")
list.Add("Третий вариант для выбора")
text = "Это пример выбора вариантов с помощью класса ListBox"
i = list.AskChoice("Выбор",text,1," Что будем печатать? ","Печать",4)
```

В этом случае будет выведено следующее окно:



Примечание: в некоторых случаях могут не отображаться строчные кириллические символы (например, лидирующий строчный символ "я"). Для разрешения этой ситуации рекомендуется использовать капитализацию первой буквы (функция proper)

4.9. Транслятор формул

Объекты этого класса создаются функцией ***Evaluate***.

Методы:

Evaluate *"строка"* - вычисляет значение по формуле, указанной в качестве текстового параметра. Можно использовать для вычисления значения по формуле, хранящейся в поле таблицы. В случае использования в бизнес-процедуре, вызванной функцией ПРОЦ_ЧИСЛО (bprun) или ПРОЦ_ТЕКСТ (bprun_str), данному объекту доступны все переменные, доступные в трансляторе, запустившем эти функции.

4.10. Период времени

Создается функцией ***BP_Period***.

4.10.1. Свойства

BegDate - дата начала периода,
EndDate - дата окончания периода,
flag - дополнительный флаг,
config – конфигурационный параметр для хранения этого флага (если пусто, то конфигурация не используется).

4.10.2. Методы:

Input - диалог с оператором по вводу периода времени и дополнительного флага.

Параметры:

- заголовок диалога,
- дополнительный текст в окне диалога,
- текст в разрезе группы для ввода дат, текст на кнопке ОК (если равен "Печать", то меняется изображение на кнопке),
- текст флажка для установки свойства flag.

Метод возвращает true, если оператор нажал кнопку ОК. Параметры не обязательны. Если не задан текст для флажка, флажок на форме отсутствует, а свойство flag после диалога имеет значение false, но в конфигурации значение не сохраняется. Конфигурационный параметр считается общим, если имя начинается с символов "kfg_". Иначе конфигурационный параметр имеет свое значение для каждого оператора.

4.11. Текстовый файл (TextFile)

Создается функцией ***BPTextFile***.

4.11.1. Свойства

EndOfFile – конец файла.

4.11.2. Методы

Open *“файл”,"параметр"** - открыть файл,
Write *“файл”*- писать в конец файла,
Read – прочитать файл,
Save *"параметр"** **–** сохранить файл,
SaveAs *“файл”,"параметр"** – сохранить файл с заданными в качестве параметра именем;
NextStr – перейти на следующую строку;
Clear – очистить файл,
Close - закрыть файл.

**"параметр"* - необязательный параметр с двумя возможными значениями: "WIN" (умолчание) и "DOS". Значения этих параметров указывают, в какой кодировке будет интерпретироваться текст (win1251 или cp866). Текст внутри класса всегда интерпретируется как "WIN". Поэтому следующая конструкция:

```
Text = TTextFile
Text.Open "XXXXXXXX.TXT", "DOS"
Text.SaveAs "XXXX_DOS.TXT"
Text.Close
```

загрузит файл в кодировке DOS, но сохранит в WIN.

4.12. Транзакция (Transaction)

Класс **Transaction** предназначен для управления транзакциями в бизнес-процедурах. При завершении работы процедуры, если транзакция остается открытой, происходит автоматический откат транзакции. Если транзакция была открыта до вызова бизнес-процедуры (внешним кодом), то работа класса никак не влияет на транзакцию.

4.12.1. Свойства

Database (только чтение) - имя псевдонима BDE,
InTransaction (только чтение) - признак того, что БД находится в транзакции.

4.12.2. Методы

Конструктор может принимать один **параметр** с именем псевдонима БД. При вызове без параметров базой данных считается "KOMPAS"

Start - начало транзакции,

Commit - завершение транзакции,

Rollback - откат транзакции.

4.13. Работа с почтой (PopMail)

Класс PopMail предназначен для работы с электронной почтой.

4.13.1. Свойства

Subject *"строка"* – тема сообщения,

Body *"строка"* – текст сообщения,

Sender *"строка"* – имя отправителя. Если это свойство указывается до первого вызова Next, то является условием фильтра, иначе в этом поле находится имя отправителя сообщения,

Address *"строка"* – электронный адрес отправителя,

Date *"дата"* – дата сообщения,

DateTime - дата и время получения письма,

Time - время получения письма,

DateBegin *"дата"* - начальная дата (фильтр),

DateEnd *"дата"* - конечная дата (фильтр),

Eof – возвращает true если нет сообщений для обработки. До первого вызова Next свойство всегда возвращает false.

4.13.2. Методы

PopMail *(Recipients, Attachments)* – инициализирует сессию с MAPI, а так же загружает mapi32.dll. *Recipients* и *Attachments* - объекты типа *ListBox*, в которые будет возвращаться список получателей и список прикреплённых файлов для сообщения. Если *Attachments* не передан, то при чтении сообщения не обрабатываются вложения, что приводит к значительному ускорению работы,

Next – получает следующее сообщение и возвращает *true*, если сообщение получено, и *false* в противном случае. После вызова *Next* сообщение доступно через свойства класса. Данный метод, при получении сообщения реагирует на значения фильтров, указываемых через поля *DateBegin*, *DateEnd* и *Sender*.

UnreadOnly – указывает, что необходимо получить только непрочитанные сообщения.

Пример использования класса:

```
Recipients = ListBox
Attachments = ListBox
letter = PopMail(Recipients, Attachments)
letter.DateBegin = "01.11.2009"
letter.DateEnd = "05.11.2009"
` Цикл пока есть сообщения
while not letter.Eof
` Получаем сообщение
if letter.Next() then
` Выводим тело сообщения
print letter.Body
end if
end
```

4.14. Блокировки (LockOpers)

Класс **LockOpers** предусмотрен для блокировки объектов на время монопольных операций, которые выполняются из бизнес-процедур. Какие именно объекты блокируются, зависит

от настройки монопольных операций (настраивается в соответствующем пункте меню «Настройка / Операции»).

Переменная класса **LockOpers** создается с помощью функции **LockOpers** без параметров.

4.14.1. Методы

Класс имеет следующие методы:

Add – добавить монопольную операцию. Одновременно могут быть заблокированы объекты для нескольких монопольных операций.

Параметры:

1) строковый – идентификатор монопольной операции;
2) строковый – название монопольной операции. Можно использовать название, отличающееся от названия в справочнике операций, если нужно конкретизировать особенности выполнения операции. Используется только в тексте сообщения о невозможности блокировать объекты;
3) список – должен содержать значения параметров операции в формате:
   <код параметра> = <значение параметра>

Обязательными являются первые два параметра метода Add. Если третий параметр не задан или содержит значения не для всех параметров операции, остальные параметры могут быть заданы при вызове метода Lock (см. ниже).

Возвращаемого значения нет.

Lock – заблокировать объекты.

Параметры:

1) набор данных – если задан, должен быть таблицей, которая используется как источник данных для заполнения недостающих параметров. Параметры заполняются по совпадению элементов массива общих мест (МОМ).

   Параметр не является обязательным. Если параметр не задан, недостающие параметры считаются неопределенными. Обращаем внимание, что блокирование объекта с неопределенным значением ключевого параметра приводит к блокированию объекта по всем возможным значениям этого параметра.

   Исключение составляет случай, когда метод Lock вызывается из бизнес-процедуры, запущенной с помощью кнопки в экранной форме. В этом случае в качестве источника данных для заполнения параметров используется МОМ, соответствующий экранной форме. При запуске процедуры из табличной форме, подчиненной к другой табличной форме (за пределами экранной формы), в качестве источника данных используется МОМ, построенный по родительской таблице.

   Возвращаемое значение: **true,** если объекты удалось заблокировать, иначе **false.** Если объекты не удалось заблокировать, выдается протокол стандартной формы.

Unlock – снять блокировку с объектов.

Параметры:

1) числовой – режим снятия блокировки (1-операция завершено, 2 – операция прервана). Если процедура будет завершена или прервана без вызова метода Unlock, все объекты будут разблокированы в режиме 2;
2) строковый – текст, используемый в сообщении о том, что блокировка была нарушена (для тех блокировок, которые допускают нарушение).

Параметры не являются обязательными. Если не задан первый параметр, подразумевается значение 1. Если не задан второй параметр, дополнительный текст в сообщении о нарушении блокировки отсутствует.

Возвращаемого значения нет.

4.15. XML

Создается фукцией XML, которая принимает три параметра:

1. путь к файлу,
2. режим работы (0 - выгрузка xml-файла, 1 - загрузка xml-файла),
3. кодировка xml-файла.

4.15.1. Методы

Класс XML имеет методы: ***AddRoot, GetRoot*** и ***Close.***

AddRoot - добавляет первый узел в xml-файл. Возвращает объект класса XMLNode.

GetRoot - читает корневой элемент из xml-файла. Возвращает объект класса XMLNode.

Close - если выполняется формирование xml-файла, метод ***Close*** сохраняет все изменения, и закрывает xml-файл. Если выполняется чтение xml-файла, метод ***Close*** только закрывает xml-файл.

4.16. XML Node

Класс XMLNode имеет методы ***Add***, ***AddClosed***, ***GetChild***, ***HasChild*** и свойства (только для чтения) *Tag*, *Attributes*, *Text*.
Метод ***Add*** добавляет узел в xml-файл. Метод принимает два параметра: *имя тега* и *список атрибутов*. Возвращает новый объект класса XMLNode. Для создания списка атрибутов следует воспользоваться классом *ListBox*.
Метод ***AddClosed*** добавляет узел, который не имеет починенных узлов. Метод имеет три параметра: *имя тега*, *текст* (который будет добавлен между открывающим и закрывающим тегом) и *список атрибутов.*
Метод ***GetChild*** читает первый не прочитанный подчиненный узел. Возвращает новый объект класса XMLNode.
Метод ***HasChild*** возвращает true, если имеются вложенные узлы, не прочитанные методом GetChild.
Методы ***Add*** и ***AddClosed*** доступны если **XMLNode** получен методами *AddRoot*, *Add*.
Свойства *Tag*, *Attributes*, *Text* доступны если **XMLNode** получен методами *GetRoot*, *GetChild*.

4.16.1. Свойства

Класс **XMLNode** свойства (только для чтения) Tag, Attributes, Text.
Tag - возвращает имя тега,
Attributes - список атрибутов в формате "имя=значение",
Text - текст между открывающим и закрывающим тегами.

4.16.2. Методы

Класс XMLNode имеет методы ***Add***, ***AddClosed***, ***GetChild***, ***HasChild***.

Add - добавляет узел в xml-файл. Метод принимает два параметра: *имя тега* и *список атрибутов*. Возвращает новый объект класса XMLNode. Для создания списка атрибутов следует воспользоваться классом *ListBox*.

AddClosed - добавляет узел, который не имеет починенных узлов. Метод имеет три параметра: *имя тега*, *текст* (который будет добавлен между открывающим и закрывающим тегом) и *список атрибутов.*

GetChild - читает первый не прочитанный подчиненный узел. Возвращает новый объект класса XMLNode.

HasChild - возвращает true, если имеются вложенные узлы, не прочитанные методом GetChild.

Методы ***Add*** и ***AddClosed*** доступны если **XMLNode** получен методами *AddRoot*, *Add*.

Свойства *Tag, Attributes*, *Tex* доступны, если **XMLNode** получен методами *GetRoot*, *GetChild*.

4.17. Служебные классы

Эти классы передаются в качестве параметра в процедуры, назначенные для базового элемента массива общих мест в качестве реакции на выбор из справочника или изменение значения. Вызов процедуры производится при работе с экранной формой. Параметры доступны в процедуре с помощью массива @Arg.

Если процедура назначена для выбора из справочника, ей в качестве параметра передается объект **класса F7Info**. Объекты этого класса имеют следующие свойства:

Value	- значение поля в экранной форме. При вызове процедуры содержит значение до начала корректировки. В результате работы процедура может изменить это значение другим значением, например, тем, которое выбрал оператор. Не следует с этой целью напрямую изменять значение элемента массива общих мест;
Apply	- признак того, что значение Value должно быть записано в поле. При вызове процедуры имеет значение false. Если по окончании работы процедуры это свойство сохранит значение false, считается, что выбор не был произведен и значение в поле не должно измениться;
F7Only	- признак того, что процедура вызвана в качестве реакции на нажатие клавиши F7.

F7Shift - признак того, что процедура вызвана в качестве реакции на нажатие сочетания клавиш Shift+F7.

F7Alt - признак того, что процедура вызвана в качестве реакции на нажатие сочетания клавиш Alt+F7.

F7Ctrl - признак того, что процедура вызвана в качестве реакции на нажатие сочетания клавиш Ctrl+F7.

*Пример процедуры, осуществляющей выбор с помощью объекта класса **Список**:*

```
Res = @Arg[1]list = ListBox
list.Add("Первый вариант для выбора")
list.Add("Второй вариант для выбора")
list.Add("Третий вариант для выбора")
text = "Это пример выбора вариантов для поля экранной формы"
i = list.AskChoice("Выбор",text,0," Укажите значение ")
if (i>=0) then
  Res.Apply = true
  Res.Value = list.String(i)
end if
```

Если процедура назначена для проверки значения, ей в качестве параметра передается объект **класса *ValidInfo*.** Объекты этого класса имеют следующие свойства:

Value - новое значение, подлежащее проверке. В результате работы процедура может изменить это значение, если это необходимо по результатам проверки. Не следует с этой целью напрямую менять значение элемента массива общих мест;

OldValue - старое значение в поле ввода до начала корректировки;

Apply - признак того, что значение Value должно быть записано в поле. При вызове процедуры имеет значение true. Если по окончании работы процедуры это свойство приобретет значение false, введенное значении отвергается: поле ввода остается в состоянии редактирования, так что оператор может либо ввести другое значение, либо нажать клавишу Esc для восстановления старого значения.

Пример процедуры, осуществляющей проверку значения:

```
Res = @Arg[1]if (Res.Value=Res.OldValue) then
  return
end if
if (Res.Value="Какое-то недопустимое значение") then
  print Res.Value+" нельзя"
  Res.Apply = false
end if
```

5. ОПЕРАТОРЫ И ФУНКЦИИ

5.1. Функции ввода/вывода

***Print** список* - выводит список значений, разделенных символами:
, - значение выводится на следующей строке;
; - значение выводится на сразу за предыдущим.

Input *строка, переменная*	- ввод значения: переменной присваивается тип "строка".
InputVal *значение*	- ввод значения. Позволяет выполнить диалог по вводу значения в формате FInpAll. Параметры: заголовок, текст на метке, собственно значение (на входе - умолчание и определяет тип значения, на выходе - результат ввода), текст на разрезе группы, текст на кнопке "ОК", название параметра (метка). Обязательными являются первые три параметра. Функция возвращает true, если была нажата кнопка ОК, и false, если была нажата кнопка "Отмена". При использовании функции для поля (полей) ввода предусмотрен выбор и проверка значения, если в тексте меток используется фрагмент "ггггмм".
MessageBox *заголовок, сообщение*	- выдает сообщение. При использовании в качестве функции возвращает логическое значение, показывающее, нажал ли оператор кнопку "Да" ("Yes"). Предусмотрена возможность управления переводом строки с помощью символа '\n'
AskChoice *заголовок, текст, варианты*	- выбор вариантов (см. также Список). Параметры: 1. Текст: заголовок окна выбора. 2. Текст: поясняющий текст, который располагается в окне выбора над перечнем вариантов. 3. Текст: определяет перечень вариантов, разделенных переводами строки или возвратами каретки. Для вставки в перечень вариантов символа перевода каретки можно использовать функцию char Возвращаемое значение: номер варианта, который выбрал оператор, начиная с нуля. Если оператор нажал кнопку *Отмена*, возвращается -1. Например: Variants = "Вариант 1"+char(13)+"Вариант 2" a = AskChoice("Выбор","Выберите вариант",Variants)
WaitBox *[строка], строка*	- выводит окно ожидания, если строка не задана - скрывает. Если второй параметр задан, окно мониторинга имеет кнопку "Отмена", при нажатии на которую выдается сообщение, текст которого задан вторым параметром. Если оператор подтвердил нажатие кнопки, устанавливается признак того, что кнопка нажата, (можно проверить с помощью функции *WaitClicked*). Если второй параметр пуст, подтверждение не запрашивается, а сразу устанавливается признак того, что кнопка была нажата. Обратите внимание: для того, чтобы нажатие на кнопку обработалось, необходимо время от времени обновлять вторую строку окна мониторинга с помощью функции *WaitBox2* (*WaitBox2 "строка" - позволяет изменить вторую строчку окна ожидания*).
WaitBox2 *строка*	- позволяет изменить вторую строчку окна ожидания.

Mail *получатель, приложение, тема, содержание*	- посылает сообщение по электронной почте. Все параметры - текстовые. Данная функция может работать только в некоторых задачах пакета КОМПАС.
ExecCmd *программа, параметры, режим*	- вызывает программу операционной системы. Параметры: 1. Текст: имя программы. Если имя задано с полным путем, то папка, в которой расположена программа, будет текущей при вызове программы. 2. Текст: передается вызываемой программе в качестве параметра (параметров) командной строки. 3. Текст: определяет режим, в котором вызывается программа. Может принимать значения: HIDE - программа вызывается в скрытом режиме; SHOWMAX - программа вызывается в максимизированном виде; SHOWMIN - программа вызывается в минимизированном виде; Иначе - программа вызывается в нормальном виде. Все параметры являются обязательными. Вместо выполнимой программы может быть вызван любой файл, для которого в операционной системе предусмотрена обработка, например, документ или картинка.
ExecQuery *запрос*	- выполняет запрос

Spawn *программа, параметры, признак*	- вызывает внешнюю программу.
	Параметры:
	1. Текст: имя программы. Если имя задано с полным путем, то папка, в которой расположена программа, будет текущей при вызове программы.
	2. Текст: передается вызываемой программе в качестве параметра (параметров) командной строки.
	3. Текст: показывает, нужно ли ждать окончания работы внешней программы. Может принимать значения:
	WAIT - нужно ждать завершения;
	NOWAIT - продолжать выполнение процедуры, не дожидаясь завершения внешней программы.
	Возвращаемое значение: 0 в случае успешного завершения внешней программы. В противном случае ненулевое число, возвращаемое внешней программой. Например:
	`i=Spawn("C:\BHT\tu3.exe", "C:\BHT\referenc.dat", "WAIT")`
	`IF i<>0 THEN`
	`PRINT i`
	`END IF`

BatchMove *получатель, набор данных [, флаг]*

- копирует данные в таблицу.

Параметры:

1. Текст: псевдоним таблицы, являющейся получателем данных.
2. Набор данных, являющийся источником данных.
3. Число. Может принимать значения:

    1 - если таблица уже существует, она будет удалена и создана заново;

    Иначе - таблица создается только, если она не существует.

Третий параметр можно не указывать, значение по умолчанию 1.

Пример:

```
TR = SQL("SELECT * FROM MOVIES")
BatchMove("MY_TEMP_MOVIES", TR)
```

TableExist таблица - проверка существования таблицы с псевдонимом, заданным текстовым параметром.

Возвращаемое значение: логическое. Например:

```
A = TableExists("MY_SUPER_TABLE")
IF A THEN
PRINT "Таблица уже существует!"
ELSE
T = CreateTable("MY_SUPER_TABLE").
END IF
```

***FormatImpExp** набор данных, флаг1, конфигурация, флаг2, заголовок, признак, имя файла* - используется для экспорта и импорта произвольных данных в следующих форматах:
1) *.dbf - база данных FoxPro;
2) *.txt - текстовый файл без разделителей;
3) *.txt - текстовый файл с разделителями (табуляцией).
Параметры:
1 Набор данных, из которого осуществляется экспорт или в который осуществляется импорт. . В качестве набора данных в данном случае указывается табличная форма.
2. Логический: следует ли выполнять импорт (истина) или экспорт (ложь).
3. Текст: имя файла конфигурации, в котором описан формат данных. Если значение параметра не указано или пусто, будет открыт стандартный диалог выбора файла конфигурации.
4. Логический: надо ли показывать протокол после завершения операции. Допустимые значения: "истина" - показывать протокол, "ложь" - не показывать протокол.
5. Заголовок окна просмотра протокола.
6. Признак, показывающий действия в случае, если при экспорте файл уже существует. Может принимать следующие значения:
0 - переспросить у оператора (умолчание);
1 - перезаписать файл (таблицу) новыми данными;
2 - дописать в конец файла (таблицы).
7. Имя файла – если указано, то задает имя файла, из которого производится импорт или в который производится экспорт данных. В этом случае имя файла, указанное в первой строке файла конфигурации, не учитывается.
Возвращаемое значение: числовое, признак успешного (0) или неуспешного (минус 1) завершения операции. Например:

```
MyTable = ServerTempTable("MY_SUPER_TABLE")
IniFile = "C:\KompSQL\SKLAD\IMPORT\MyTable.ini"
i = FormatImpExp(MyTable,true,IniFile,false,"Импорт данных")
IF i = 0 THEN
MyTable.Refresh
ELSE
PRINT "Не удалось выполнить импорт!"
RETURN
END IF
```

5.2. Математические функции (возвращают число)

***abs**(число)* - модуль (абсолютное значение) числа.

***cos**(число)* - косинус.

***DecRound**(число,дес)* - округление до ближайшего числа с числом десятичных знаков, заданных вторым параметром (аналогично функции ОКРУГ в трансляторе).

***fix**(число)*	- округление вниз (отбрасывает дробную часть числа)
***int**(число)*	- округление к ближайшему целому
***Length**(массив)*	- размер массива.
***Length**(строка)*	- длина строки.
Pos(строка1,строка2)	- позиция строки2 в строке1 (ищет подстроку в строке).
***rnd**(число)*	- генерирует случайное целое число в диапазоне от нуля до параметра функции.
***round**(число)*	- округление вверх (к ближайшему целому, которое не меньше заданного).
***sgn**(число)*	- знак (возвращает -1, если число отрицательное, 0, если число равно 0, 1, если число положительное).
***sin**(число)*	- синус.
***StrToInt**(строка)*	- привести строку к числу.

5.3. Строковые функции (возвращают текст)

***Char**(число)*	- строка из одного символа по его коду.
CreateGuid	получение 36-символьного уникального идентификатора
***Date2SQL**(дата,флаг)*	- преобразует аргумент типа "дата" в строку, пригодную для использования в запросах. Флаг (логический) показывает, нужно ли заключать результат в кавычки. Например: D = Date2SQL(Now, true) CheckDateOuery = SQL("SELECT * FROM MY_TABLE WHERE DATA <= %D%")
***FormatFloat**("Формат", Число)*	преобразует число в строку в соответствии со спецификатором формата, содержащимся в параметре *Формат.* Описание спецификаторов формата приведено в *уточнении* ниже.
***Left**(строка,число)*	- заданное числом кол-во символов слева.
***Lower**(строка)*	- привести к нижнему регистру.
***Ltrim**(строка)*	- удалить пробелы слева.
***PARSEDELIM** (строка1, строка2, число)*	Позволяет получить один из фрагментов строки, в которой фрагменты разделены определенным разделителем. Параметры: 1. Строка: исходная строка, фрагмент из которой требуется. 2. Строка: разделитель. Должна состоять из одного символа. 3. Число: порядковый номер фрагмента. Если меньше 1 или больше общего числа фрагментов, результатом вызова функции будет пустая строка. Результат: строка, содержащая текст фрагмента.

***Right**(строка,число)*	- заданное числом кол-во символов справа.
***Rtrim**(строка)*	- удалить пробелы справа.
***Str**(число)*	- привести число к строке.
String2SQL	Приводит строку к виду, пригодному для использования в SQL-предложениях, в частности, одинарные кавычке в строке удваиваются. Параметры: 1. текст: исходная строка; 2. логическое: следует ли заключать строку во внешние кавычки.
***Substring**(строка,число1, число2*)	- часть строки с позиции число1, длиной число2.
***Trim**(строка)*	- удалить пробелы справа и слева.
***Upper**(строка)*	- привести к верхнему регистру.

Уточнение

Функция FormatFloat используется для преобразования чисел в строку предопределённого формата:

FormatFloat("Формат", Число)

Эта функция преобразует число в строку в соответствии со спецификатором формата, содержащимся в параметре *Формат*, который может принимать следующие значения:

0	Указывает на цифру. Если форматируемая величина имеет в этой позиции цифру, то вставляется она, в противном случае вставляется "0".
#	Указывает на цифру. Если форматируемая величина имеет в этой позиции цифру, то вставляется она, в противном случае ничего не вставляется.
.	Десятичный разделитель. Сюда вставляется символ, определенный константой DecimalSeparator (DecimalSeparator global variable; задается в Региональных настройках ОС). Второе и последующие указания этого спецификатора в одной части формата игнорируются.
,	Разделитель тысяч. Если параметр Формат содержит один или более знаков ",", то группы по три цифры, считая влево от десятичной точки, будут разделяться специальным символом, который задан константой ThousandSeparator (ThousandSeparator global variable, задается в Рег иональных настройках ОС). Местоположение поля может быть произвольным.
E+, E-, e+, e-	Признаки представления числа в научном формате. Появление любого из этих аргументов означает, что число будет отформатировано с использованием научной нотации. Вставка нулей (от 1 до 4) после такого аргумента позволяет определить минимальное количество разрядов экспоненты. Спецификаторы **E+** и **e+** генерируют знак "плюс" для положительных экспонент и "минус" для отрицательных. Спецификаторы **E-** и **e-** опускают знак "плюс" для положительных экспонент.

'XX' "XX"	Символы, заключенные в одинарные или двойные кавычки, напрямую включаются в выходную строку.
;	Разделяет спецификаторы формата для положительных, отрицательных и нулевых чисел.

Примечания:

1. Число всегда округляется до того количества знаков, которое задано для размещения цифр (символы '0' и '#') справа от десятичного разделителя ("."). Если десятичный разделитель не задан, округление производится до ближайшего целого числа.

2. Если у преобразуемого числа слева от десятичного разделителя получается больше знаков, чем задано символов для их размещения, то цифры все равно добавляются в строку. Если символов недостаточно справа от десятичного разделителя, происходит округление.

3. Допускается задавать в параметре *Формат* разные способы форматирования положительных, отрицательных чисел и нуля. В этом случае форматирующая строка может содержать от одной до трех секция, разделенных знаком ";" (точка с запятой):

- Одна секция: применяется для всех чисел;
- Две секции: первый формат применяется для чисел, больших или равных нулю, второй - для отрицательных;
- Три секции: первый формат применяется для положительных, второй - для отрицательных чисел, третий - для нуля.

Если форматы для отрицательных чисел или нуля пусты (между разграничивающими секции знаками ";" пусто), применяется формат для положительных чисел.
Если пуст формат для положительных чисел или спецификатор формата вообще не задан (пустая строка), то числа форматируются согласно обобщенному формату (General) с 15 значимыми разрядами (обобщенный формат определяет универсальное числовое форматирование, которое стремится сохранить результирующее значение как можно компактнее: удаляются конечные нули и десятичные точки, где это возможно; никакие разделители тысяч не показываются). Такое форматирование применяется также в случае, если количество значащих цифр слева от десятичной точки превышает 18, и не задан научный формат.

Применение спецификатора иллюстрируется в таблице на примере преобразования четырех чисел:

Спецификатор	1234	-1234	0.5	0
0	1234	-1234	1	0
0.00	1234.00	-1234.00	0.50	0.00
#.##	1234	-1234	.5	
#.##0.00	1,234.00	-1,234.00	0.50	0.00
#,##0.00;(#,##0.00)	1,234.00	(1,234.00)	0.50	0.00
#,##0.00;;Zero	1,234.00	-1,234.00	0.50	Zero
0.000E+00	1.234E+03	-1.234E+03	5.000E-01	0.000E+00
#.###E-0	1.234E3	-1.234E3	5E-1	0E0

5.4. Функции для работы с датами

Now	- текущая дата/время.
DateFormat *(строка,дата)*	- изображение даты, форматированное строкой.
DateSerial *(день, месяц, год)*	- сборная дата.
PeriodPart *(дата1, дата2, номер)*	- получение расстояния между двумя датами в годах, месяцах и днях. Функция возвращает числовое значение, смысл которого определяется третьим параметром: 1-лет; 2-месяцев; 3-дней. Если даты отстоят друг от друга на целое число лет (например, 15.06.2003 и 15.06.2005), в результате получится целое число лет (в данном примере - 2), 0 месяцев и 0 дней. Если даты отстоят на целое число месяцев (например, 15.06.2003 и 15.10.2005), в результате получится некоторое число лет (в данном примере - 2), целое число месяцев (в данном примере - 4) и 0 дней. Следует особо обратить внимание, что функция расшифровывает расстояние между датами, а не период времени, включающий обе даты. Если нужно получить расшифровку для периода времени, включающего обе даты, следует ко второй дате добавить единицу.
BP_period	**-** создает и возвращает объект класса *Период времени*.
DaysInPeriod *(дата1,дата2, номер)*	- позволяет получить число дней в указанном диапазоне времени. Тип значения: число. Параметры: 1. Дата начала периода. 2. Дата окончания периода. 3. Режим подсчета дней. Может принимать следующие значения: 5 - по 5-дневной рабочей неделе; 6 - по 6-дневной рабочей неделе; 7 - в календарных днях за вычетом праздников; 0 - просто в календарных днях.

5.5. Функции для работы с объектами

Call "имя процедуры", *параметры*	- вызов другой бизнес-процедуры. В вызываемой процедуре переданные параметры доступны через массив

	@Arg; доступна переменная @ArgCount, содержащая количество переданных параметров.
ClearIns	- снимает отметки со всех строк
Connect, *параметр*	- позволяет выполнить переключение на другую базу данных (псевдоним BDE задается параметром).
CreateTable псевдоним *[, признак1 [, признак2]]*	- создает таблицу. Параметры: 1. Текст: псевдоним таблицы. 2. Логический: нужно ли пересоздавать таблицу в случае, если она уже существует. Значение по умолчанию "истина". 3. Логический: нужно ли создавать индексы. Значение по умолчанию "истина". Возвращаемое значение: объект класса Набор данных. Например: T = CreateTable("MY_SUPER_TABLE")
EnumerateID *(кол-во параметров – 5)*	позволяет заполнить в заданном наборе данных (1-й параметр) заданное поле (2-й параметр) монотонно возрастающими значениями с заданным шагом (5-й параметр). При этом значение, с которого начнется заполнение определяется как max+1 для заданного поля (4-й параметр) заданной таблицы (3-й параметр).
Evaluate	- создает и возвращает объект класса *Транслятор формул*.
FieldByMesto	имя поля по элементу МОМ ***Параметры*** (все параметры обязательны): 1) строка - псевдоним таблицы; 2) строка - элемент МОМ. ***Возвращает***: имя поля, соответствующее указанному элементу МОМ, или пустую строку, если такого поля в таблице нет.
FileDelete *“файл”*	- удаляет указанный файл.
FileExists *“файл”*	**–** проверяет, существует ли указанный файл***.***
FillBPList *“параметр 1” “параметр 2”*	- заполняет список именами бизнес-процедур.
FindELMOM, *параметр*	- ищет элемент в списке доступных переменных. В качестве параметра в функцию передается текст, являющийся именем элемента массива общих мест. Если элемент доступен, функция возвращает значение истина. Иначе функция возвращает значение ложь. Элементы массива общих мест, являющиеся таблицами, считаются доступными только в том случае, если соответствующая таблица имеется в экранной форме, и уже была инициализирована.

GenUniqFileName *"путь к файлу"*	- по заданному в параметре пути возвращает имя файла с уникальным суффиксом.
GetAbsenceList *(5 параметров)*	Используется в модуле "**Управление персоналом**". Предназначена для получения списка видов неявок, исключаемых из стажа. Параметры: 1. *Строка*: табельный номер; 2. *Целое*: код трудовых отношений; 3. *Дата*: дата начала рабочего периода; 4. *Дата*: дата окончания рабочего периода; 5. *Целое*: код условий труда (необязательный). Возвращаемое значение - строка со списком видов неявок через запятую.
GetComputerName	**-** возвращает имя компьютера, на котором запущен модуль.
GetDir *"путь к исходной папке", "текст сообщения на форме выбора"*	- позволяет вывести диалог по выбору папки. Оба параметра необязательны. Функция возвращает полный путь к выбранной папке или пустую строку, если оператор нажал кнопку "Отмена".
GetELMOM	возвращение значения элемента МОМ по его имени
GetOtpOstAndFill *(6 параметров)*	Используется в модулях **кадрово-зарплатного блока** для расчета остатка отпуска и заполнения таблицы периодов. Параметры: 1. *Строка*: табельный номер; 2. *Целый*: вид отпуска; 3. *Дата*: дата расчета остатка; 4. *Целый*: код трудовых отношений; 5. *Целый*: признак расчета; 6. *Набор данных*: таблица PERIOD_PARTS (необязательный). Возвращаемое значение: вещественный - остаток отпуска.
InsCount	- позволяет узнать число отмеченных строк
IsFloat "строка"	Возвращает логическое значение *true,* если строка может быть преобразована к числу с плавающей точкой. При невозможности преобразования возвращает *false*.
InsertToExcel	выгрузка в Excel <u>Параметры</u>: 1) выгружаемый набор данных; 2) текст, помещаемый в первую строку страницы Excel. Если параметр 2 не задан или пуст, используется заголовок табличной формы. При использовании обеих функций набор данных должен быть табличной формой, в том числе допускается использовать элемент МОМ, соответствующий таблице в составе экранной формы.
IsInt "строка"	возвращает логическое значение *true,* если строка может быть преобразована к целому числу. При невозможности преобразования возвращает *false*.
ListBox	- создает и возвращает объект класса *Список.*

Normativ *(5 параметров)*

Используется в модуле "**Управление персоналом**". Позволяет получить значение норматива для указанного месяца, отдела, категории персонала.
Параметры:
1. *Текст*: обозначение норматива;
2. *Текст*: месяц в формате «ггггмм»;
3. *Текст*: номер отдела. По умолчанию – пусто;
4. *Число*: код категории персонала. По умолчанию – ноль;
5. *Число*: режим работы. По умолчанию – ноль. При ненулевом значении этого параметра учитываются только нормативы, заданные для указанного месяца. При нулевом значении параметра норматив может быть взят из ближайшего предыдущего месяца, в котором данный норматив указан.
Обязательными являются первые два параметра. Если номер отдела не задан или пуст, используются только значения норматива, общие для всех отделов. Если код категории персонала не задан или равен нулю, используются значения норматива, общие для всех категорий персонала.
Возвращаемое значение: числовое.

OpenDialog

– диалог выбора файла.

PostIf *“имя поля”*

- позволяет сохранить строку набора данных в случае, если поле, имя которого задано параметром, имеет непустое значение. Если значение пустое, никаких действий не выполняется.

PrintReport *"псевдоним","фильтр", "параметры"[,"формат" [,"макропараметры"]]*

четвертый параметр "формат" может принимать значения:
1 - стандартный формат;
2 - формат MS Excel;
полное имя файла. Имя файла должно быть нечисловым, так как числовое значение воспринимается как указание на формат В случае указания только имени файла (без полного пути) используется каталог, заданный по умолчанию для сохранения отчетов в формате Excel в **Конфигурации** на закладке **Печать**. Отчет формируется в формате Excel, сохранение производится в файл с заданным именем. Если имя файла не заканчивается символами “.xls”, эти символы автоматически добавляются к имени файла. Добавление расширения не производится, если имя файла уже имеет расширение, то есть заканчивается точкой, за которой следует не более 3-х символов. Диалог с пользователем по корректировке параметров на экран не выводится. При этом все настройки, кроме имени файла, сохраняют значения по умолчанию. Предусмотрена возможность указать имя страницы, в которую будет выгружаться отчет. Имя страницы отделяется от имени файла с помощью вертикальной черты, например: "ФайлОтчета.xls | Страница 2". Если вертикальной черты в тексте параметра нет, используется имя "Страница 1". Если файл уже существует, в него добавляется или замещается страница с

указанным именем. Если файл не существует, формируется новый файл с такой страницей.

В варианте с параметрами отчета в виде объекта класса ***Список*** можно передать список макропараметров для запросов, используемых в отчете.

QForm псевдоним *[,параметры]*	- создает и возвращает объект класса ***Запросная форм***а по ее псевдониму. Значения параметрам присваиваются в порядке их описания в запросе (не по именам). Если указано меньше параметров, чем имеется в запросе, то остальные сохраняют значения, указанные в описании запросной формы. Если в запросе имеются макропараметры, в качестве первого параметра может быть указан объект типа Список. Элементы списка используются для заполнения макропараметров. Если элементов списка меньше, чем число макропараметров, оставшиеся макропараметры сохраняют значения, указанные в описании запросной формы.
QFormLive псевдоним *[,параметры]*	- создает и возвращает объект класса ***Запросная форма*** (с возможностью редактирования) по ее псевдониму. Значения параметрам присваиваются в порядке их описания в запросе (не по именам). Если указано меньше параметров, чем имеется в запросе, то остальные сохраняют значения, указанные в описании запросной формы. Если в запросе имеются макропараметры, в качестве первого параметра может быть указан объект типа Список. Элементы списка используются для заполнения макропараметров. Если элементов списка меньше, чем число макропараметров, оставшиеся макропараметры сохраняют значения, указанные в описании запросной формы.
QFWizard	- вызывает Мастер запросов.
Reestr *"тип документа", "фильтр", "режим"*	- создает и возвращает объект класса Реестр документов.Тип документа задается числовым параметром функции. Строковый параметр "режим" может принимать следующие значения или их произвольное сочетание (например, “123”)[1]:

[1] не все значения параметра «режим» действуют при использовании метода ShowLookup:
1 - действует в полной мере для версии 11.85.01 после 25.06.2008. В более ранних версиях при использовании этого значения выбор производится без учета фильтра, что может привести к ошибке позиционирования на выбранную запись, если она не удовлетворяет условию фильтрации;
2 - не действует, так как работа с фильтром по F6 в форме выбора не предусмотрена;
3 - не действует для формы выбора, но является обязательным при создании табличной формы реестра с целью последующего использования метода ShowLookup. Если режим 3 не задан, для некоторых реестров в момент использования метода ShowLookup в таблице не будет ни одной записи, что вызовет ошибку позиционирования на выбранную запись. В версии 11.85.01 после 25.06.2008 при использовании метода ShowLookup в таких случаях предусмотрено сообщение об ошибке с указанием, что необходимо использовать режим 3. В более ранних версиях выводится обычное сообщение «Выбранная запись не

1 - задает режим установки фильтра (третий параметр) как серверного. Это надо учитывать при составлении текста для условия фильтрации. Во избежание двусмысленности рекомендуется поля основной таблицы снабжать префиксом MAIN. С другой стороны, в этом фильтре можно использовать сложные условия, которые применимы только в SQL, например, вложенные SELECT и т.п.;
2 - перед открытием табличной формы выводится диалог установки серверного фильтра. Если для табличной формы задан фильтр по умолчанию, он сразу же используется для фильтрации записей с помощью серверного фильтра, то есть табличная форма открывается без диалога. Следует также учитывать, что во всех случаях фильтр, заданный третьим параметром, действует одновременно с фильтром, заданным по F6;
3 - используется только для реестров и позволяет отключить режим вывода реестра, заданный в описании типа документа (помесячно, поквартально, по годам, реестр целиком). Следует учитывать, что при использовании таких режимов после выполнения функции TForm в табличной форме не будет ни одной записи, пока не будет выполнен метод Show, ShowModal или ShowLookup. Параметр “3” позволяет преодолеть эту проблему, если в бизнес-процедуре предполагается выполнять какие-либо действия с табличной формой без ее показа или еще до показа. Параметр “3” рекомендуется использовать в сочетании с тем или иным серверным фильтром, чтобы избежать открытия большой таблицы реестра целиком;
4 - учитывать фильтр по умолчанию. Если этот режим не задан, то при открытии табличной формы из бизнес-процедуры фильтр по умолчанию не учитывается, а в диалоге настройки фильтра отсутствует флажок «Использовать по умолчанию». Если режим задан, работа с табличной формой из бизнес-процедуры в этом отношении идентична обычной работе с той же табличной формой, то есть фильтр по умолчанию учитывается и может быть установлен пользователем в процессе работы с табличной формой;
0 - используется только для табличных форм LIC_CH (Лицевые счета) и ESN_BASE (Налоговая база ЕСН и суммы налогов) и позволяет отменить предварительный диалог по установке фильтра при открытии этих табличных форм без отбора по табельному номеру или коду получателя. Параметр “0” рекомендуется использовать в сочетании с тем или иным серверным фильтром, чтобы избежать открытия большой

RefreshCache *"псевдоним таблицы"* - используется в модуле «Управление персоналом». Обновляет кэшированные данные, используемые для открытой расчетной ведомости. Функция имеет один

найдена в табличной форме» либо никакого сообщения не выводится, но далее возможны самые разные ошибки при работе с этим реестром, в том числе – за пределами бизнес-процедуры;
4 - не действует, так как работа с фильтром по F6 в форме выбора не предусмотрена;
0 - не действует, так как работа с фильтром по F6 в форме выбора не предусмотрена.

	параметр: псевдоним таблицы, которую необходимо обновить в кэше. При пустом параметре (подразумевается по умолчанию), обновляется кэш для всех четырех таблиц (справочник видов оплаты KODTAB, кадровая картотека KADRY, штатная расстановка NAZN, табель учета рабочего времени TABRAB). Если табличная форма расчетной ведомости не открыта, функция ничего не делает.
Report	- создает и возвращает объект класса ***Отчет.***
ResrictValue “параметр”	- значения полей в ограничении по ключу
RestrictField “параметр”	- имена полей в ограничении по ключу
RunQuery псевдоним *[,параметры]*	- аналогично предыдущему, но имеет следующие особенности: - псевдоним запроса или группы может содержать пробелы; - если параметров меньше, чем предусмотрено в запросе, перед выполнением запроса выводится стандартный диалог ввода параметров запроса. Отказ от ввода параметров приводит к отказу от выполнения запроса; - может выполняться не только одиночный запрос, но и скрипт, состоящий из нескольких последовательно выполняемых запросов.
RunSQL *"Текст Sql-запроса"*	- просто выполнение запроса (не возвращает результата).
RunSQL псевдоним *[,параметры]*	- аналогично предыдущему, но вместо текста запроса указывается псевдоним запроса, сформированного с помощью Мастера запросов. Для того, чтобы эта конструкция отличалась от предыдущей, псевдоним запроса (включая группу) не должен содержать пробелов. Значения параметрам присваиваются в порядке их описания в запросе (не по именам). Если указано меньше параметров, чем имеется в запросе, остальные параметры сохраняют значения, указанные в описании запроса. Если в запросе имеются макропараметры, в качестве первого параметра может быть указан объект типа Список. Элементы списка используются для заполнения макропараметров. Если элементов списка меньше, чем число макропараметров, оставшиеся макропараметры сохраняют значения, указанные в описании запроса.
RV_OTDEL	Используется в модуле "**Управление персоналом**". Функция не имеет параметров и возвращает номер отдела, выделенного в дереве расчетной ведомости. Если расчетная ведомость не открыта или в ней выделена верхняя ветка дерева, функция возвращает пустую строку.

SaveDialog - диалог выбора и сохранения файла. Если ввести имя существующего файла, то выдается запрос о перезаписи файла.

SelectField - выбор поля таблицы или табличной формы
Параметры (обязателен только первый параметр):
1) строка - псевдоним таблицы;
2) строка - ранее выбранное поле (для позиционирования формы выбора). Если не задано, форма выбора позиционируется на первом поле.

Возвращает: имя поля, если выбор сделан, или пустую строку, если выбор не сделан.

ServerTempTable псевдоним - создает и возвращает объект класса *Серверная временная таблица*. Псевдоним таблицы задается текстовым параметром функции.

SetConfigParam *(2 параметра)* функция предназначена для изменения значения конфигурационной переменной.
Параметры:
1) имя конфигурационной переменной;
2) значение.

SetELMOM изменение элемента МОМ по его имени

SetNULL Предназначена для задания полю пустого значения.
Пример использования:

```
a=TFORM"<имя ТФ>"
a.SetNULL("DATA_D")
(DATA_D - поле таблицы)
```

Show *форма* - показывает табличную или запросную форму.

ShowLookup *форма* - отображает табличную или запросную форму в виде справочника и позволяет выбрать из него. Если заголовок формы не передан в качестве параметра, то используется заголовок, заданный свойством Caption. Возвращаемое значение - логическое: показывает, произвел ли оператор выбор. Для запросной формы в функцию может быть передан параметр, который используется в качестве заголовка формы. Пример:

```
QF = QForm "Справочники.DOPODR_TAB_N",POL@TAB_N
IF QF.ShowLookup("Договора подряда работника: "+POL@PYR3)
THEN
DOCUM@DOC_TYPE.SetValue(QF.TYP_D)
DOCUM@DOC_ID.SetValue(QF.ID)
DOCUM@DOC_NOM.SetValue(QF.N_DOK)
DOCUM@DOC_DATE.SetValue(QF.DATA_D)
SUM@SUM_ZPL.SetValue(QF.SUM_D)
END IF
```

ShowModal *форма* - показывает табличную или запросную форму модально.
Например:

```
ShowModal TForm "SPR_DOK"
```

Sleep *"число"*	Позволяет выполнить задержку на заданное количество миллисекунд.
SQL *"Текст Sql-запроса"*	- возвращает объект класса ***Набор данных***: результат выполнения запроса. В тексте запроса допустимо в качестве параметров использовать переменные Бейсика. Для этого нужно заключить имя переменной в проценты. Например: select * from %TableName% where Field1='%Value%'
SQL псевдоним *[,параметры]*	- аналогично предыдущему, но вместо текста запроса указывается псевдоним запроса, сформированного с помощью Мастера запросов. Для того, чтобы эта конструкция отличалась от предыдущей, псевдоним запроса (включая группу) не должен содержать пробелов. Значения параметрам присваиваются в порядке их описания в запросе (не по именам). Если указано меньше параметров, чем имеется в запросе, остальные параметры сохраняют значения, указанные в описании запроса. Если в запросе имеются макропараметры, в качестве первого параметра может быть указан объект типа Список. Элементы списка используются для заполнения макропараметров. Если элементов списка меньше, чем число макропараметров, оставшиеся макропараметры сохраняют значения, указанные в описании запроса.
SQLPost	- сохраняет запись, находящуюся в режиме редактирования или добавления, с помощью динамически генерируемого запроса. Может применяться вместо метода Post.
SysLog "название события", "текст сообщения с описанием события"	- добавляет запись о событии в системный протокол
SysLog *"название события", "текст сообщения с описанием события"*	- добавляет запись о событии в системный протокол
TableExist *“таблица”*	**-** проверка существования таблицы с псевдонимом, заданным текстовым параметром.
TableVersion *“таблица”*	- версия ядра КИС КОМПАС
TblAliasByTF	псевдоним таблицы по табличной форме ***Параметры*** (все параметры обязательны): 1) строка - псевдоним табличной формы. ***Возвращает***: псевдоним таблицы или пустую строку, если указанная табличная форма не найдена.

***TForm** "псевдоним ТФ", "тип документа", "фильтр", "режим"*

Все параметры, кроме первого, являются необязательными. Параметр "тип документа" учитывается только для реестров и позволяет обойти диалог по выбору типа документа для тех табличных форм, которые связаны с несколькими типами документов. При нулевом значении второго параметра табличная форма реестра открывается по общим правилам. Строковый параметр "режим" может принимать следующие значения или их произвольное сочетание (например, “123”)[2]:
1 - задает режим установки фильтра (третий параметр) как серверного. Это надо учитывать при составлении текста для условия фильтрации. Во избежание двусмысленности рекомендуется поля основной таблицы снабжать префиксом MAIN. С другой стороны, в этом фильтре можно использовать сложные условия, которые применимы только в SQL, например, вложенные SELECT и т.п.;
2 - перед открытием табличной формы выводится диалог установки серверного фильтра. Если для табличной формы задан фильтр по умолчанию, он сразу же используется для фильтрации

записей с помощью серверного фильтра, то есть табличная форма открывается без диалога. Следует также учитывать, что во всех случаях фильтр, заданный третьим параметром, действует одновременно с фильтром, заданным по F6;
3 - используется только для реестров и позволяет отключить режим вывода реестра, заданный в описании типа документа (помесячно, поквартально, по годам, реестр целиком). Следует учитывать, что при использовании таких режимов после выполнения функции TForm в табличной форме не будет ни одной записи, пока не будет выполнен метод Show, ShowModal или ShowLookup. Параметр “3” позволяет преодолеть эту проблему, если в бизнес-процедуре предполагается выполнять какие-либо действия с табличной формой без ее показа или еще до показа. Параметр “3” рекомендуется использовать в сочетании с тем или иным серверным фильтром, чтобы избежать открытия большой таблицы реестра целиком;
4 - учитывать фильтр по умолчанию. Если этот режим не задан,

---

[2] не все значения параметра «режим» действуют при использовании метода *ShowLookup*:
1 - действует в полной мере (для версии 11.85.01 после 25.06.2008). В более ранних версиях при использовании этого значения выбор производится без учета фильтра, что может привести к ошибке позиционирования на выбранную запись, если она не удовлетворяет условию фильтрации;
2 - не действует, так как работа с фильтром по F6 в форме выбора не предусмотрена;
3 - не действует для формы выбора, но является обязательным при создании табличной формы реестра с целью последующего использования метода ShowLookup. Если режим 3 не задан, для некоторых реестров в момент использования метода ShowLookup в таблице не будет ни одной записи, что вызовет ошибку позиционирования на выбранную запись. В версии 11.85.01 после 25.06.2008 при использовании метода ShowLookup в таких случаях предусмотрено сообщение об ошибке с указанием, что необходимо использовать режим 3. В более ранних версиях выводится обычное сообщение «Выбранная запись не найдена в табличной форме» либо никакого сообщения не выводится, но далее возможны самые разные ошибки при работе с этим реестром, в том числе – за пределами бизнес-процедуры;
4 - не действует, так как работа с фильтром по F6 в форме выбора не предусмотрена;
0 - не действует, так как работа с фильтром по F6 в форме выбора не предусмотрена.

то при открытии табличной формы из бизнес-процедуры фильтр по умолчанию не учитывается, а в диалоге настройки фильтра отсутствует флажок «Использовать по умолчанию». Если режим задан, работа с табличной формой из бизнес-процедуры в этом отношении идентична обычной работе с той же табличной формой, то есть фильтр по умолчанию учитывается и может быть установлен пользователем в процессе работы с табличной формой;
0 - используется только для табличных форм LIC_CH (Лицевые счета) и ESN_BASE (Налоговая база ЕСН и суммы налогов) и позволяет отменить предварительный диалог по установке фильтра при открытии этих табличных форм без отбора по табельному номеру или коду получателя. Параметр “0” рекомендуется использовать в сочетании с тем или иным серверным фильтром, чтобы избежать открытия большой таблицы целиком.
Возможности функции Tform по указанию параметров фильтрации ("фильтр", "режим") неприменимы для табличных форм SHTRAS (штатное расписание); NAZN (штатная расстановка); OTPUSK (журнал неявок); TABRAB (табель учета рабочего времени); CH_70_S (расчетная ведомость); SPR_70 (свод по заработной плате); LIC_CH (лицевой счет с отбором по одному работнику). Напомниная, что для таких табличных форм функцию TForm вообще нельзя использовать, так как для них эта функция только открывает табличную форму в немодальном режиме, но получить доступ к табличной форме нельзя. Если необходимо использовать такую табличную форму для выбора из справочника, следует вместо функции TForm использовать функцию TLookupForm.
символ **R** (регистр существенен) - табличная форма выводится в режиме "Только чтение".

TFWizard - вызывает Мастер табличных форм.

ThrowException – - генерирует исключение, которое может быть обработано за пределами бизнес-процедуры.

TLookupForm *"псевдоним ТФ", "тип документа", "фильтр", "режим"* – создает объект TForm для выбора из справочника. Если в бизнес процедуре необходимо осуществить выбор из табличной формы как из справочника, рекомендуется создавать объект класса “Табличная форма” не с помощью функции TForm, а с помощью функции TLookupForm. Объект, созданный с помощью функции TLookupForm, может использовать только для выбора из справочника с помощью метода ShowLookup, тогда как методы Show, ShowModal и ShowSF у него отсутствуют. С другой стороны, такой объект ведет себя одинаково для любых табличных форм, в том числе - для перечисленных выше в настоящей статье. Кроме того, таблица для такого объекта открывается в режиме только для чтения, что ускоряет работу программы.
Все параметры, кроме первого, являются необязательными. Параметр "тип документа" учитывается только для реестров и

позволяет обойти диалог по выбору типа документа для тех табличных форм, которые связаны с несколькими типами документов. При нулевом значении второго параметра табличная форма реестра открывается по общим правилам. Строковый параметр "режим" может принимать следующие значения или их произвольное сочетание (например, “123”)[3]:

1 - задает режим установки фильтра (третий параметр) как серверного. Это надо учитывать при составлении текста для условия фильтрации. Во избежание двусмысленности рекомендуется поля основной таблицы снабжать префиксом MAIN. С другой стороны, в этом фильтре можно использовать сложные условия, которые применимы только в SQL, например, вложенные SELECT и т.п.;

2 - перед открытием табличной формы выводится диалог установки серверного фильтра. Если для табличной формы задан фильтр по умолчанию, он сразу же используется для фильтрации записей с помощью серверного фильтра, то есть табличная форма открывается без диалога. Следует также учитывать, что во всех случаях фильтр, заданный третьим параметром, действует одновременно с фильтром, заданным по F6;

3 - используется только для реестров и позволяет отключить режим вывода реестра, заданный в описании типа документа (помесячно, поквартально, по годам, реестр целиком). Следует учитывать, что при использовании таких режимов после выполнения функции TForm в табличной форме не будет ни одной записи, пока не будет выполнен метод Show, ShowModal или ShowLookup. Параметр “3” позволяет преодолеть эту проблему, если в бизнес-процедуре предполагается выполнять какие-либо действия с табличной формой без ее показа или еще до показа. Параметр “3” рекомендуется использовать в сочетании с тем или иным серверным фильтром, чтобы избежать открытия большой таблицы реестра целиком;

4 - учитывать фильтр по умолчанию. Если этот режим не задан, то при открытии табличной формы из бизнес-процедуры фильтр по умолчанию не учитывается, а в диалоге настройки фильтра отсутствует флажок «Использовать по умолчанию». Если режим

---

[3] не все значения параметра «режим» действуют при использовании метода *ShowLookup*:

1 - действует в полной мере (для версии 11.85.01 после 25.06.2008). В более ранних версиях при использовании этого значения выбор производится без учета фильтра, что может привести к ошибке позиционирования на выбранную запись, если она не удовлетворяет условию фильтрации;

2 - не действует, так как работа с фильтром по F6 в форме выбора не предусмотрена;

3 - не действует для формы выбора, но является обязательным при создании табличной формы реестра с целью последующего использования метода ShowLookup. Если режим 3 не задан, для некоторых реестров в момент использования метода ShowLookup в таблице не будет ни одной записи, что вызовет ошибку позиционирования на выбранную запись. В версии 11.85.01 после 25.06.2008 при использовании метода ShowLookup в таких случаях предусмотрено сообщение об ошибке с указанием, что необходимо использовать режим 3. В более ранних версиях выводится обычное сообщение «Выбранная запись не найдена в табличной форме» либо никакого сообщения не выводится, но далее возможны самые разные ошибки при работе с этим реестром, в том числе – за пределами бизнес-процедуры;

4 - не действует, так как работа с фильтром по F6 в форме выбора не предусмотрена;

0 - не действует, так как работа с фильтром по F6 в форме выбора не предусмотрена

задан, работа с табличной формой из бизнес-процедуры в этом отношении идентична обычной работе с той же табличной формой, то есть фильтр по умолчанию учитывается и может быть установлен пользователем в процессе работы с табличной формой;
0 - используется только для табличных форм LIC_CH (Лицевые счета) и ESN_BASE (Налоговая база ЕСН и суммы налогов) и позволяет отменить предварительный диалог по установке фильтра при открытии этих табличных форм без отбора по табельному номеру или коду получателя. Параметр “0” рекомендуется использовать в сочетании с тем или иным серверным фильтром, чтобы избежать открытия большой таблицы целиком.

***Use** объект*

- указывает, что методы и свойства объекта будут использоваться по умолчанию. Если объектом является табличная или запросная форма, то как только выполнится этот оператор, появятся переменные через которые можно работать с полями задействованной рабочей формы. Их имена совпадают с именами полей. Например:

```
use TForm "SPR_DOK"
print TIP
```

В результате выведется значение поля TIP для первой строки таблицы.
Или аналогично:

```
a = TForm "SPR_DOK"
print a.TIP
```

ViewDoks

- просмотр документа.

Параметры:
1) набор данных, содержащий реквизиты документа;
2) часть сообщения о том, что нет реквизитов документа.
Если параметр 2 не задан или пуст, используется текст "таблицы"

***XFile** “таблица”, “ключевое поле”, “значение ключевого поля”*

- позволяет выполнять операции с файлами, хранимыми в поле таблицы (3 параметра)

***XFileEx** (6 параметров)*

Функция для работы с файлами.
Параметры:
1. *[Псевдоним таблицы]* - псевдоним таблицы с файлом;
2. *[Имя ключевого поля]* - имя ключевого поля таблицы с файлом (для идентификации строки);
3. *[Значение ключевого поля]* - значение ключевого поля таблицы с файлом (для идентификации строки);
4. *[Имя поля файла]* - имя поля, в котором хранится файл, если пустая строка, то определяется автоматически;
5. *[Действие]* - код действия (0 - Открыть файл, 1 - Сохранить в файл на диск, 2 - Сохранить файл в БД, 3 - Удалить файл из БД);

	6. *[Имя файла]* - если выполняется действие 1, то папка для сохранения файла (например 'c:\my dir\') или пустая строка (тогда будет диалог с пользователем). Если выполняется действие 2, то путь к файлу (например 'c:\autoexec.bat'). Для других действий игнорируется.
xRefresh псевдоним	- выполняет обновление данных во всех табличных формах, связанных с указанной таблицей.