

# Глава 6. Запись, таблица, выборка и другие объекты базы данных

Запись.....	2
Запись.....	2
Таблица.....	3
ЗаписьТаблицы.....	5
Сохранить.....	6
Добавить.....	7
Скопировать.....	8
Удалить.....	9
ОбъединитьЗаписи.....	10
АдресЗаписи.....	11
АдресСвязаннойЗаписи.....	12
ЗаписатьАдресаЗаписей.....	13
ЕстьАдрес.....	14
ЕстьСвязаннаяЗапись.....	15
Загрузить.....	15
Загружена.....	17
Инициализировать.....	17
НазваниеТаблицы.....	18
НазваниеРаздела.....	19
ПрочитатьОбъект.....	20
ЗаписатьОбъект.....	21
ВыполнитьКоманду – для записи.....	22
ЗаблокироватьЗапись.....	23
РазблокироватьЗапись.....	23
ПопробоватьЗаблокироватьЗапись.....	24
Список записей.....	25
СписокЗаписей.....	25
ЗаписатьСписокЗаписей.....	25
ПрочитатьСписокЗаписей.....	26
Выборка.....	27
Выборка.....	27
УстановитьСвязаннуюЗапись.....	29
СвязаннаяЗапись.....	29
УстановитьКорень.....	30
УстановитьНачальныйКорень.....	31
КореньВыборки.....	32
УстановитьПараметрыВыборки.....	33
ПараметрыВыборки.....	34
ЗафиксироватьВыборку.....	34
СнятьФиксациюВыборки.....	35
Сортировать.....	36
Таблица – для объекта.....	37
Развернуть.....	38
РазвернутьСУзлами.....	39
Свернуть.....	40
ТекущийРаздел.....	41
НайтиРаздел.....	41

ПерейтиВРаздел .....	42
ПерейтиВКорень .....	44
ОбновитьВыборки .....	45
Поиск и перебор записей .....	45
УстановитьМаксКлюч .....	45
УстановитьМинКлюч .....	46
Следующий по индексу .....	47
Предыдущий по индексу .....	49
Следующий по связи .....	50
Предыдущий по связи .....	52
Найти - по индексу .....	53
НайтиСКонца - по индексу .....	54
Найти – по значению индекса .....	56
НайтиСКонца – по значению индекса .....	57
ДляВсех(Записей) .....	59
КоличествоЗаписей .....	60
Поля связи, иерархии и прочее .....	61
Корень .....	61
Лист .....	62
СкрытыйУзел .....	63
Узел .....	64
ЭтоКорень .....	64
ЭтоУзел .....	65
ЭтоСкрытыйУзел .....	66
ЭтоЛист .....	67
Связать .....	67
Активировать .....	69
Флаг .....	70
Регистрация изменений .....	72
ВЖурналИзменений .....	72
ВЖурналОшибок .....	73
ВЖурналСообщений .....	73
НайтиЖурналСообщений .....	74
ОткрытьЖурналСообщений .....	75
ОткрытЖурналСообщений .....	76
ЗапуститьЖурналСообщений .....	77
ОстановитьЖурналСообщений .....	77
ОчиститьЖурналСообщений .....	78

## Запись

### Запись



Возвращает запись от текущего объекта.

## Синтаксис

Запись ( )

## Параметры

Не указываются.

## Возвращает

Объект с текущей записью.

## Примеры

```
оДиалог = Диалог ("Счетафактуры" );  
оЗапись = Запись ( ) ; # возвратит пустую запись от  
текущего объекта ;  
Установить (оДиалог) ; # в качестве текущего объекта  
установим объект оДиалог ;  
оЗапись = Запись ( ) ; # возвратит запись от текущего  
объекта оДиалог ;
```

## См. также:

- [ЗаписьТаблицы](#)
- [АдресЗаписи](#)

## Таблица

Создаёт объект на основе таблицы базы данных или внешнего файла.

## Синтаксис

Таблица (ИмяТаблицы [ , ТолькоДляЧтения ] )

## Параметры

**ИмяТаблицы** (текст) – имя таблицы базы данных или имя внешнего файла. В первом случае указывается логическое имя таблицы, то есть не имя файла с таблицей, а то, как эта таблица именуется в программе, например, «Организации», «Документ» и так далее. Необходимо указывать точное название таблицы, учитывая регистры.

Во втором случае указывается физическое имя файла. Программа понимает файлы с расширениями – «.dbf» (файлы СУБД DBase, FoxPro). Именно по наличию в параметре одного из этих расширений программа определяет, что указано имя таблицы или имя файла. Если в имени файла не указан каталог, то файл ищется в том каталоге, откуда была запущена программа.

Файл «.dbf» обязательно должен существовать, иначе будет выдано сообщение об ошибке.

**ТолькоДляЧтения** (логическое значение) – значение «Да» разрешает только просмотр открываемой таблицы. Если в качестве параметра указать «Нет», то запись можно редактировать. В режиме «только для чтения» одна и та же таблица может быть открыта несколько раз, при этом навигация по каждой из открытых таблиц осуществляется независимо от всех остальных. Но при попытке открыть одну таблицу в разных режимах будет выдано сообщение об ошибке.

## Возвращает

**Объект**, содержащий запись указанной таблицы или внешнего файла. Возвращаемый функцией объект не содержит каких-либо реальных данных, все поля записи пусты. Для извлечения записи из базы данных можно использовать функции «Найти», «Следующий», «Предыдущий» и так далее.

## Примеры

```
oРасч = Таблица("Расчетные счета"); # определяем  
пустой объект oРасч на основе таблицы "Расчётные  
счета".
```

В случае dbf-файлов можно обращаться непосредственно к именам полей в исходном файле. Так в следующем примере идёт обращение к полю с именем «ИНН».

```
oЗапись = Таблица("man.dbf");  
Пока (Следующий(oЗапись))  
Сообщить(oЗапись.ИНН);
```

С помощью ниже приведенного фрагмента будет выведен список всех имен пользователей, причем после каждого имени будет выдаваться сообщение о том, какой пользователь идет следующим:

```
перем пТ1 = Таблица("Пользователи.tbl", ДА);  
перем пТ2 = Таблица("Пользователи.tbl", ДА);
```



```
Следующий ( пТ2 );  
Пока ( Следующий ( пТ1 ) )  
{  
    Следующий ( пТ2 );  
    Сообщить ( пТ1.Имя + " Следующий: " + пТ2.Имя );  
}
```

### См. также:

- [ЗаписьТаблицы](#)
- [НазваниеТаблицы](#)

## ЗаписьТаблицы

Возвращает запись указанной таблицы от записи выборки из базы данных.

### Синтаксис

`Запись.ЗаписьТаблицы (ИмяТаблицы)`

### Параметры

**ИмяТаблицы** (текст) – имя таблицы базы данных, связанную запись которой нужно получить.

### Возвращает

**Объект**, содержащий связанную запись указанной таблицы.

### Примеры

Чтобы получить запись таблицы «Лица» от записи выборки «Организации» оПост:

```
оПост = оПост.ЗаписьТаблицы ("Лица") ;
```

### См. также:

- [Запись](#)
- [Таблица](#)
- [АдресЗаписи](#)
- [Добавить](#)

## Сохранить

Сохраняет запись указанного объекта в базе данных.

### Синтаксис

**Запись . Сохранить ( )**

### Параметры

Не указываются.

### Комментарии

На момент вызова функции запись объекта обязательно должна быть извлечена из базы данных. В противном случае никаких изменений в базе данных сделано не будет.

### Возвращает

**Да** (логическое значение) – запись указанного объекта сохранена.

**Нет** (логическое значение) – запись не была сохранена.

### Примеры

Чтобы изменить поле «СуммаСебест» в наименовании расхода в расходной накладной, нужно после присваивания полю нового значения сохранить объект «Товар».

```
ДляВсех (Наименований)
{
    НАИМ.СуммаСебест = Окр(Цена * Курс, 0.01);
    НАИМ.Сохранить;
}
```

Как уже говорилось, запись объекта должна быть извлечена из базы данных, только что созданная запись функцией «Сохранить» игнорируется. Таким образом, следующий фрагмент работать *не будет*.

```
пЛицо = Таблица ("Лица");
пЛицо.Сохранить;
# здесь ничего не сохранится!!!
```

### См. также:

- [Добавить](#)

- [Скопировать](#)
- [Удалить](#)
- [ОбъединитьЗаписи](#)

## Добавить

Добавляет в базу данных запись указанного объекта. Запись может быть как вновь созданной, так и извлечённой из базы данных.

### Синтаксис

**Запись . Добавить ( )**

### Параметры

Не указываются.

### Возвращает

**Да** (логическое значение) – запись успешно добавлена.

**Нет** (логическое значение) – запись указанного объекта не была добавлена.

### Примеры

Так можно программно добавить в базу данных запись об организации:

```
пОрг = Выборка ("Организации") ;  
пОрг.Название = "Чудо" ;  
пОрг.Наименование = "ЗАО Чудо" ;  
пОрг.ИНН = "7714123412" ;  
пОрг.Корень ;  
пОрг.Добавить ; # добавляем запись об организации
```

Как уже говорилось, добавляемая запись обязательно должна быть вновь созданной. То есть, если в правиле операции написать:

```
Документ . Добавить ;
```

то новый документ не будет добавлен (будет сообщение: «Запись с указанной позицией отсутствует в таблице»).

Обратите внимание, что если нужно добавить подряд несколько записей, то каждый раз перед добавлением новой записи необходимо проинициа-

лизовать добавляемый объект функцией «Очистить». А уже после этого можно установить значения полей и вызвать «Добавить».

```
oOrg = Выборка ("Организации") ;  
oOrg.Название = "Чудо" ;  
oOrg.Корень ;  
oOrg.Добавить ;  
oOrg.Очистить () ;  
oOrg.Название = "Чудо1" ;  
oOrg.Корень ;  
oOrg.Добавить ;
```

### См. также:

- [Инициализировать](#)
- [Сохранить](#)
- [Удалить](#)
- [Скопировать](#)
- [ОбъединитьЗаписи](#)

## Скопировать

Создает копию записи указанного объекта.

### Синтаксис

```
Запись.Скопировать ()
```

### Параметры

Не указываются.

### Возвращает

Да (логическое значение) – если копия записи была создана успешно.

Нет (логическое значение) – если не удалось скопировать запись.

### Примеры

Нужно добавить подряд несколько записей об организациях:

```
oOrg = Таблица ("Организации") ;
```



```
oОрг.Название = "Чудо";  
oОрг.Наименование = "ЗАО Чудо";  
oОрг.ИНН = "7714123412";  
oОрг.Корень;  
oОрг.Добавить; # добавили запись организации Чудо  
oОрг.Скопировать; # скопировали запись  
oОрг.Название = "Чудо1"; # исправили название  
oОрг.ИНН = "7714123413";  
oОрг.Добавить; # добавили запись организации Чудо1
```

### См. также:

- [Добавить](#)
- [Сохранить](#)
- [Удалить](#)
- [ОбъединитьЗаписи](#)

## Удалить

Удаляет из базы данных запись указанного объекта.

### Синтаксис

Запись.Удалить ( )

### Параметры

Не указываются.

### Комментарии

Сам объект после вызова функции не удаляется, а вот его запись будет заново проинициализирована (все поля будут очищены).

### Возвращает

**Да** (логическое значение) – удаление прошло успешно.

**Нет** (логическое значение) – запись указанного объекта не была удалена. Это может быть в частности связано с тем, что на удаляемую запись ссылаются другие записи базы данных. Например, нельзя удалить запись организации, если на неё были выписаны какие-то документы.

### Примеры

Так можно программно удалить из базы данных запись об организации с ИНН 7714123412:

```
oOrg = Лицо ("7714123412", "Организации");  
oOrg.Удалить;
```

### См. также:

- [Добавить](#)
- [Сохранить](#)
- [Скопировать](#)
- [ОбъединитьЗаписи](#)

## ОбъединитьЗаписи

Позволяет объединить связанные записи двух объектов одинакового формата.

### Синтаксис

```
ОбъединитьЗаписи (Запись1, Запись2)
```

### Параметры

**Запись1, Запись2** (объект) – записи одного формата, связи которых нужно объединить.

### Возвращает

**Да** (логическое значение) – связи объектов были успешно объединены и запись второго объекта была удалена.

**Нет** (логическое значение) – если не удалось объединить записи.

### Примеры

Необходимо объединить расчетные счета двух филиалов организации и оставить только одну из двух записей:

```
oOrg1 = Лицо ("7714123412", "Организации"); # запись  
первого филиала  
oOrg2 = Лицо ("7714123413", "Организации"); # запись  
второго филиала, которая будет удалена
```



ОбъединитьЗаписи (oOrg1, oOrg2); # в результате останется запись oOrg1, к расчетным счетам которой добавятся счета записи oOrg2

### См. также:

- [Добавить](#)
- [Сохранить](#)
- [Удалить](#)
- [Скопировать](#)
- [ОбъединитьЗаписи](#)

## АдресЗаписи

Возвращает адрес записи в таблице базы данных.

### Синтаксис

Запись . АдресЗаписи ( )

### Параметры

Не указываются.

### Возвращает

Число (целое) – адрес записи.

### Комментарий

Если запись не из таблицы базы данных, то функция возвратит «-1».

### Примеры

Узнаем адрес выбранной записи:

```
oНовОбъект = Окно ("Виды цен" );  
oНовОбъект . ВыполнитьВыбор ;  
oНовЗапись = oНовОбъект . Запись ;  
oНовЗапись . АдресЗаписи ( ) ;
```

### См. также:

- [АдресСвязаннойЗаписи](#)

- [Загрузить](#)
- [ЗаписатьАдресаЗаписей](#)
- [ЕстьАдрес](#)

## АдресСвязаннойЗаписи

Возвращает адрес записи, связанной с текущей по указанному имени связи.

### Синтаксис

**Запись . АдресСвязаннойЗаписи (ИмяСвязи)**

### Параметры

**ИмяСвязи** (текст) – имя связи, по которому необходимо найти запись, связанную с текущей. Нужное имя связи можно узнать в описании словарей или в описании кнопки связи в ресурсах по <Ctrl+F12>.

### Возвращает

**Число** (целое) – адрес связанной записи.

### Комментарий

Использование данной функции идентично действию функции «АдресЗаписи», но позволяет значительно быстрее обратиться к адресу связанной записи, предварительно не извлекая эту запись из базы данных.

### Примеры

Узнаем адрес документа-основания для текущей расходной накладной:

```
функция НаДокументЗакрыть ( )  
{  
    ДляВсех (Оснований ( ) )  
        Сообщить (Документ . АдресСвязаннойЗаписи ("Лицо1" ) ) ;  
}
```

### См. также:

- [АдресЗаписи](#)

- [ЕстьСвязаннаяЗапись](#)
- [Загрузить](#)
- [ЗаписатьАдресаЗаписей](#)
- [ЕстьАдрес](#)

## ЗаписатьАдресаЗаписей

Записывает адреса записей из таблицы базы данных в двоичное поле объекта.

### Синтаксис

**ЗаписатьАдресаЗаписей** (ДвоичноеПоле, Запись1 [, Запись2] , ...)

### Параметры

**ДвоичноеПоле**(переменная) – поле с двоичными данными в объекте.

**Запись1, Запись2,...** (объект) – записи из таблицы базы данных, адреса которых нужно записать в двоичное поле объекта. Эти записи обязательно должны быть одностипными, т.е. из одной таблицы.

### Возвращает

**Да** (логическое значение) – если в двоичное поле записаны адреса записей.

**Нет** (логическое значение) – если в двоичное поле не записаны адреса записей.

### Примеры

Необходимо в реестре отобрать документы по указанным лицам. Для этого можно воспользоваться примером:

```
оДиалог = Диалог("Параметры складских документов");  
ЗаписатьАдресаЗаписей(оДиалог.ПЛ1,  
Лицо("7605015260", "Орг"), Лицо("7605016030",  
"Орг"));  
оДиалог.Выполнить();
```

Необходимо отобрать документы, отражающие движение товаров и материалов по оптовому и розничному складам:

```
оДиалог = Диалог ("Параметры складских документов") ;  
ЗаписатьАдресаЗаписей (оДиалог . ПЛСП , НайтиСклад (1) ,  
НайтиСклад (4) ) ;  
оДиалог . Выполнить ( ) ;
```

### См. также:

- [АдресЗаписи](#)
- [ЕстьАдрес](#)
- [Загрузить](#)

## ЕстьАдрес

Проверяет существование записи с указанным адресом в базы данных.

### Синтаксис

```
Запись . ЕстьАдрес (Число)
```

### Параметры

**Число** (целое) – адрес записи.

### Возвращает

**Да** (логическое значение) – есть запись с указанным адресом.

**Нет** (логическое значение) – нет записи с указанным адресом.

### Примеры

Узнаем ФИО сотрудника у записи с адресом 8:

```
оНовОбъект = Окно ("Сотрудники") ;  
оНовЗапись = пНовОбъект . Запись ( ) ;  
Если (оНовЗапись . ЕстьАдрес (8) )  
{  
    оНовЗапись . Загрузить (8) ;  
    Сообщить (оНовЗапись . ФИО) ;  
}  
Иначе  
    Сообщить ("Нет записи с указанным номером") ;
```

### См. также:



- [Запись](#)
- [АдресЗаписи](#)
- [Загрузить](#)

## ЕстьСвязаннаяЗапись

Проверяет существование записи, связанной с текущей по указанному имени связи.

### Синтаксис

**Запись . ЕстьСвязаннаяЗапись (ИмяСвязи)**

### Параметры

**ИмяСвязи** (текст) – имя связи, по которому необходимо проверить существование записи, связанной с текущей. Нужное имя связи можно узнать в описании словарей или в описании кнопки-связи по <Ctrl+F12>.

### Возвращает

Да (логическое значение) – есть связанная запись.

Нет (логическое значение) – нет связанной записи.

### Примеры

Узнаем, есть ли записи, связанные с выбранным сотрудником по связи «Лицо2»:

```
oНовОбъект=Окно ("Сотрудники" ) ;  
oНовОбъект . ВыполнитьВыбор ;  
oНовЗапись=oНовОбъект . Запись ;  
Сообщить (oНовЗапись . ЕстьСвязаннаяЗапись ("Лицо2" ) ) ;
```

### См. также:

- [Запись](#)
- [АдресЗаписи](#)
- [Загрузить](#)

## Загрузить

Загружает запись с указанным адресом.

## Синтаксис

**Запись . Загрузить (Число)**

## Параметры

**Число** (целое) – адрес записи.

## Комментарии

В качестве параметра можно указать значение «**Нет**». В этом случае будет загружена пустая запись, т.е. функция «**Загружена**» возвратит «**Нет**». Хочется отметить, что результат выполнения кода **оЗапись . Загрузить (нет)** будет совпадать с результатом выполнения кода **оЗапись . Загрузить (-1)**. Соответственно, если **оЗапись . Загружена ()** возвращает «**Нет**», то **оЗапись . АдресЗаписи ()** вернёт «**-1**».

## Возвращает

**Да** (логическое значение) – запись с указанным адресом существует и загружена.

**Нет** (логическое значение) – запись с указанным адресом не загружена.

## Примеры

Изменим запись сотрудника под номером 6:

```
оНовОбъект = Окно ("Сотрудники" );  
оНовЗапись = оНовОбъект . Запись ( ) ;  
Если (оНовЗапись . ЕстьАдрес ( 6 )  
{  
    оНовЗапись . Загрузить ( 6 ) ;  
    оНовОбъект . ИзменитьЗапись (оНовЗапись ) ;  
}
```

## См. также:

- [Запись](#)
- [АдресЗаписи](#)

- [ЕстьАдрес](#)
- [Загружена](#)

## Загружена

Проверяет существование загружаемой записи. Запись должна быть создана в базе данных.

### Синтаксис

`Запись . Загружена ( )`

### Параметры

Не указываются.

### Возвращает

**Да** (логическое значение) – запись с указанным адресом существует и загружена.

**Нет** (логическое значение) – запись с указанным адресом не загружена.

### Примеры

Сообщить полное название раздела иерархии, в котором находится загружаемая запись объекта `пОбъект`:

```
Если (оОбъект . Запись ( ) . Загружена ( ) )  
    Сообщить (оОбъект . НазваниеРаздела ( ) ) ;
```

### См. также:

- [Запись](#)
- [Загрузить](#)

## Инициализировать

Выполняется процесс инициализации записи - возможность задания у записи начальных значений ( значений по умолчанию).

### Синтаксис

`Запись . Инициализировать ( )`

## Параметры

Не указываются.

## Возвращает

Да (логическое значение) – инициализация прошла успешно.

Нет (логическое значение) – инициализация не произошла.

## Пример

Для создания документа в реестре «Бухгалтерские расчеты» нужно выполнить следующий порядок действий:

```
Перем оДокумент = Выборка ("Бухгалтерские расчеты" );  
оДокумент . ПерейтиВРаздел (оДокумент . КореньВыборки ( ) );  
оДокумент . Инициализировать ( ) ;  
оДокумент . Дата = 01.01.08 ;  
оДокумент . Номер = 1 ;  
оДокумент . Добавить ( ) ;
```

Если не вызвать функцию «Инициализировать», то у документа не проставится «ТипДокумента» и другие необходимые поля, и его не будет видно в реестре документов.

## См. также:

- [ЗаписьТаблицы](#)
- [Запись](#)

## НазваниеТаблицы

Возвращает имя таблицы, в контексте которой выполняется функция.

## Синтаксис

```
Запись . НазваниеТаблицы ( )
```

## Параметры

Не указываются.

## Возвращает



**Текст** - строка с именем таблицы.

## Примеры

Если в правиле операций для расходной накладной написать:

**Сообщить (НазваниеТаблицы ( ) ) ;**

то при закрытии накладной будет выдано сообщение – «Расходные накладные». Если написать:

**ДляВсех (Наименований)**

**Сообщить (НазваниеТаблицы ( ) ) ;**

то будет сообщение – «Расход».

Для того, чтобы получить имя таблицы для конкретного объекта достаточно указать перед именем функции имя объекта и точку. Например, чтобы узнать имя таблицы объекта «пОбъект», нужно написать:

**Сообщить (пОбъект . НазваниеТаблицы ( ) ) ;**

## См. также:

- [НазваниеРаздела](#)

## НазваниеРаздела

Формирует полное название раздела иерархии, в котором находится текущая запись.

## Синтаксис

**Запись . НазваниеРаздела ( [Иерархия] )**

## Параметры

**Иерархия** (текст) – имя поля иерархии. Если параметр не указан, берётся первое поле типа «Иерархия» в записи.

## Возвращает

**Текст** - строку с полным названием папки иерархии, в котором находится текущая запись. В качестве разделителя между именами папок используется точка. В связи с этим, если в названии папки уже присутствует точка, то чтобы не путать с разделителем, функция заменить точку в названии папки на символ «\». Фактически возвращается то же самое, что содержится в верхней строке окна с иерархической таблицей.

## Примеры

Полное название подразделения сотрудника можно получить так:

```
oСотр = Лицо ("102", "Сотрудники")
oСотр.НазваниеРаздела ()
```

Например, если сотрудник находится в подразделении «Участок сборки», которое в свою очередь находится в отделе «Цех 10», то этот фрагмент вернёт строку «Цех 10.Участок сборки».

Если в поле документа «Лицо1», указана организация, то полное название раздела, в котором находится запись этой организации выводится так:

```
Лицо1.Лицо_ .НазваниеРаздела ();
```

Запись «Лицо1.Лицо\_» перед вызовом функции означает, что будет браться не текущая запись, запись объекта «Лицо1.Лицо\_». Почему не просто «Лицо1»? Дело в том, что поле иерархии находится не в самой таблице «Лицо», запись которой содержит объект «Лицо1», а в таблице, связанной с ней по связи «Лицо\_».

```
НазваниеРаздела ("Прайс-лист"); # возвращает полное
название раздела иерархии «ПрайсСлист», в котором
находится текущая запись.
```

## См. также:

- [НазваниеТаблицы](#)
- [ПерейтиВРаздел](#)

## ПрочитатьОбъект

Считывает данные из поля записи из базы данных в объект.

## Синтаксис

```
ПрочитатьОбъект (ИмяПоля)
```

## Параметры

**ИмяПоля** (текст) – имя поля записи (типа двоичные данные) из базы данных, из которого считываются данные в объект.

## Возвращает

**Объект** с данными, считанные из поля записи.

## Примеры

Необходимо узнать значения считываемых в объект оПрочит данных. Для этого можно воспользоваться примером:

```
перем оИнф [ ] ;
оИнф.1 = 112 ;
оИнф.3 = 545 ;
оИнф.5 = "Числа" ;
оЗ = Выборка ("Формы отчетности") ;
оЗ.Добавить ( ) ; # добавляем запись формы в базу
данных
ЗаписатьОбъект (оЗ.ПараметрыОтчета , оИнф) ; #
записываем данные объекта оИнф в поле записи оЗ ;
оПрочит = ПрочитатьОбъект (оЗ.ПараметрыОтчета) ; #
считываем данные из поля в объект оПрочит ;
ДляВсех (Переменных (оПрочит , пИмя) )
Сообщить (оПрочит [пИмя] ) ;
```

**См. также:**

- [ЗаписатьОбъект](#)

## ЗаписатьОбъект

Записывает данные объекта в указанное поле записи из базы данных.

### Синтаксис

**ЗаписатьОбъект (ИмяПеременной , Объект)**

### Параметры

**ИмяПеременной** (текст) – поле записи (типа двоичные данные) из базы данных, в которое записываются данные объекта.

**Объект** (объект) - имя объекта с данными.

### Возвращает

**Число** - размер записанной информации в байтах.

## Примеры

Необходимо узнать размер записанных данных объекта оИнф в поле «ПараметрыОтчета» записи выборки. Для этого можно воспользоваться следующим примером:

```
перем оИнф [ ] ;  
оИнф.1 = 112 ;  
оИнф.3 = 545 ;  
оИнф.5 = "Числа" ;  
оЗ = Выборка ("Формы отчетности") ;  
оЗ.Добавить ( ) ;  
Сообщить (ЗаписатьОбъект (оЗ.ПараметрыОтчета , оИнф) ) ;
```

в результате функция возвратит 36 байт записанных в указанное поле данных.

## См. также:

- [ПрочитатьОбъект](#)

## ВыполнитьКоманду – для записи

Выполняет указанную команду. Команды можно посмотреть, открыв любое меню в файле описания ресурсов.

## Синтаксис

**Запись . ВыполнитьКоманду (ИмяКоманды)**

## Параметры

**ИмяКоманды** (текст) – имя команды.

## Возвращает

Да (логическое значение) – если команда выполнена.

Нет (логическое значение) – если команда не выполнена.

## Примеры

```
оОрг = Выборка ("Складская картотека") ;  
оПарам = пОрг.ПараметрыВыборки ( ) ;  
оПарам.Допфлаги = "2" ;  
оПарам.Пометки = "1" ;
```

```
oПарам.Признаки = "0";  
oОрг.УстановитьПараметрыВыборки (пПарам) ;  
oПарам.ВыполнитьКоманду ("ClearSelParam"); # очистим  
значения параметров выборки поОрг.
```

### См. также:

- [Запись](#)

## ЗаблокироватьЗапись

Запрещает всем другим пользователям редактировать данную запись.

### Синтаксис

```
Запись .ЗаблокироватьЗапись ( )
```

### Параметры

Не указываются.

### Возвращает

**Да** (логическое значение) – если удалось заблокировать запись.

**Нет** (логическое значение) – если запись не заблокировалась.

### Примеры

Необходимо заблокировать запись таблицы «Организации» на время работы с ней:

```
O1 = Таблица ("Организации") ;  
O1.Загрузить (2) ;  
Если ( O1.ЗаблокироватьЗапись ( ) )  
Сообщить ("работаем") ;
```

### См. также:

- [РазблокироватьЗапись](#)
- [ПопробоватьЗаблокироватьЗапись](#)

## РазблокироватьЗапись

Разрешает всем пользователям редактировать данную запись, если она была ранее заблокирована.

## Синтаксис

`Запись . РазблокироватьЗапись ( )`

## Параметры

Не указываются.

## Возвращает

Ничего не возвращает.

## Примеры

Необходимо заблокировать запись таблицы «Организации» на время работы с ней, а затем сделать доступной для всех:

```
O1 = Таблица ("Организации" );
O1. Загрузить (2) ;
Если ( O1. ЗаблокироватьЗапись ( ) )
{
    Сообщить ("работаем" ) ;
}
Иначе
    Сообщить ("отдыхаем" ) ;
O1 . РазблокироватьЗапись ( ) ;
```

## См. также:

- [ЗаблокироватьЗапись](#)
- [ПопробоватьЗаблокироватьЗапись](#)

## ПопробоватьЗаблокироватьЗапись

Пытается запретить всем другим пользователям редактировать данную запись.

## Синтаксис

`Запись . ПопробоватьЗаблокироватьЗапись ( )`

## Параметры

Не указываются.

**Возвращает**

Да (логическое значение) – если удалось заблокировать запись.

Нет (логическое значение) – если запись не заблокировалась.

**См. также:**

- [ЗаблокироватьЗапись](#)
- [РазблокироватьЗапись](#)

## Список записей

**СписокЗаписей**

Возвращает объект, содержащий поля записи указанного формата.

**Синтаксис**

**СписокЗаписей (Имя)**

**Параметры**

**Имя** (текст) – имя формата записи. Они описываются в ресурсах, их имена можно посмотреть, открыв соответствующий файл.

**Возвращает**

Объект с полями указанного формата записи.

**Примеры**

Вывести объект с полями записи формата «БухОтч Баланс стр990»:

```
СписокЗаписей ("БухОтч Баланс стр990") ;
```

**См. также:**

- [ЗаписатьСписокЗаписей](#)
- [ПрочитатьСписокЗаписей](#)

**ЗаписатьСписокЗаписей**

Записывает в диалоге данные указанного формата записи.

## Синтаксис

`Запись . ЗаписатьСписокЗаписей (ИмяПеременной)`

## Параметры

**ИмяПеременной** (переменная) – имя переменной в диалоге, в которую записываются данные указанного формата записи.

## Возвращает

**Число** – размер записанной информации в байтах.

## Примеры

Запишем в диалог данные формата записи «БухОтч Баланс стр990»:

```
оЗаписи = СписокЗаписей ("БухОтч Баланс стр990");
оДиалог = Диалог ("БухОтч Баланс стр990");
оЗаписи.п000060099001 = "формат";
оЗаписи.п000060099003 = 993;
оЗаписи.п000060099004 = 904;
оЗаписи.Добавить ();
пЗ = оЗаписи.ЗаписатьСписокЗаписей ('оДиалог.БухОтч
Баланс стр990');
```

## См. также:

- [СписокЗаписей](#)
- [ПрочитатьСписокЗаписей](#)

## ПрочитатьСписокЗаписей

Считывает данные указанного формата из диалога.

## Синтаксис

`Запись . ПрочитатьСписокЗаписей (ИмяПеременной)`

## Параметры

**ИмяПеременной** (переменная) – имя переменной в диалоге, из которой считываются данные указанного формата записи.



## Возвращает

**Число** – размер считываемой информации в байтах.

## Примеры

Просмотреть в диалоге «БухОтч Баланс стр990» данные указанного формата записи:

```
оЗаписи = СписокЗаписей("БухОтч Баланс стр990");  
оДиалог = Диалог("БухОтч Баланс стр990");  
пПр = оЗаписи.ПрочитатьСписокЗаписей('оДиалог.БухОтч  
Баланс стр990');
```

## См. также:

- [СписокЗаписей](#)
- [ЗаписатьСписокЗаписей](#)

# Выборка

## Выборка

Создаёт объект на основе выборки из базы данных.

## Синтаксис

**Выборка (Имя)**

## Параметры

**Имя** (текст) – имя выборки из базы данных. Выборки описаны в ресурсах, их имена можно посмотреть, открыв соответствующий файл ресурсов.

## Возвращает

**Объект**, содержащий запись указанной выборки. Созданный объект не будет содержать каких-либо реальных данных. Для извлечения записей выборки используется функция «ДляВсех(Записей)».

## Примеры

Функция будет полезна при создании сложных составных объектов, таких, например, как объекты типа «Документ». Дело в том, что документы реально состоят, как минимум, из двух записей разных таблиц. А описание

того, какие записи входят в документ данного типа, лежит в соответствующем описании выборки из базы данных. Поэтому, чтобы создать объект-входящую платежку, пишем так:

```
оДокумент = Выборка ("Входящие платежи");
... # Здесь нужно задать поля и связи объекта
оДокумент
оДокумент.Добавить;
```

Пример поиска номера склада по его названию:

```
пНаимСкл = "Капитальные вложения";
оСкл = выборка ("Склады")
Пока (Следующий (оСкл) )
  Если ( пНаимСкл == оСкл.Наименование)
  {
    пНомСкл = оСкл.НомерСклада;
    сообщить (пНомСкл)
  }
```

Выведение списка элементов (с их количеством), перемещенных между складами:

```
пСкладП = "Инвентарь, спецодежда в эксплуатации
(забалансовый учет)"; # склад-получатель
пСкладО = "Склад материалов и инвентаря"; # склад-
отправитель
оПер = выборка ("Внутренние перемещения");
  Если (оПер.ПерейтиВРаздел ("Инвентарь, спецодежда
(на затраты при выдаче).Выдача в эксплуатацию"))
  {
    Пока ( Следующий (оПер) )
    {
      Установить (оПер);
      Если (СкладПолучатель.Наименование ==
пСкладП И СкладОтправитель.Наименование == пСкладО)
      ДляВсех (Наименований ( ))
      {
        сообщить (Название + ": " + Кол_во)
      }
    }
  }
```

**См. также:**



- [Таблица](#)
- [Запись](#)
- [ПараметрыВыборки](#)
- [ДляВсех\(Записей\)](#)
- [КоличествоЗаписей](#)

## УстановитьСвязаннуюЗапись

Устанавливает связанные записи в выборке по указанному объекту.

### Синтаксис

**Запись . УстановитьСвязаннуюЗапись (Объект)**

### Параметры

**Объект**, по которому в выборке устанавливается связанная запись.

### Возвращает

Ничего не возвращает.

### Примеры

По установленной связанной записи для организации оПлательщик узнаем расчетные счета этой организации:

```
oB = Выборка ("Расчетные счета") ;
oПлательщик = НайтиЛицо ("7605015260" ,
"Организации") ;
oB . УстановитьСвязаннуюЗапись (oПлательщик) ;
Пока (Следующий (oB) )
    Сообщить (oB . Pсчет) ;
```

### См. также:

- [Запись](#)
- [СвязнаяЗапись](#)
- [УстановитьКорень](#)

## СвязаннаяЗапись

Возвращает объект, по которому в выборке устанавливается связанная запись.

### Синтаксис

`Запись . СвязаннаяЗапись ( )`

### Параметры

Не указывается.

### Возвращает

Объект, по которому в выборке устанавливается связанная запись.

### Примеры

```
пВ = Выборка ("Расчетные счета" );  
оплательщик = НайтиЛицо ("7605015260" ,  
"Организации" ) ;  
пВ . УстановитьСвязаннуюЗапись (оплательщик ) ;  
Пока (Следующий (оВ )  
Сообщить (оВ . Рсчет) ;  
оСвязь = пВ . СвязаннаяЗапись ( ) ; # в оСвязь объект  
оплательщик ;
```

### См. также:

- [УстановитьСвязнуюЗапись](#)
- [УстановитьКорень](#)

## УстановитьКорень

Устанавливает раздел корнем выборки.

### Синтаксис

`Запись . УстановитьКорень (Раздел)`

### Параметры

Раздел (текст или объект) – название раздела или запись раздела, который необходимо установить корнем выборки.



### Возвращает

Да (логическое значение) – раздел установлен корнем выборки.

Нет (логическое значение) – раздел не установлен корнем выборки.

### Комментарии

Функция имеет смысл, если выборка является иерархической, т.е. должно быть поле типа иерархия.

### Примеры

Пример описан ниже (см. функцию «КореньВыборки»),

### См. также:

- [КореньВыборки](#)
- [УстановитьСвязнуюЗапись](#)
- [СвязнаяЗапись](#)
- [УстановитьНачальныйКорень](#)

## УстановитьНачальныйКорень

Позволяет вернуться к корневому разделу выборки. Функция, обратная к функции **УстановитьКорень**.

### Синтаксис

**Запись . УстановитьНачальныйКорень ( )**

### Параметры

Не указываются.

### Возвращает

Да (логическое значение) – раздел установлен корнем выборки.

Нет (логическое значение) – раздел не установлен корнем выборки.

### Комментарии

Функция имеет смысл, если выборка является иерархической, т.е. должно быть поле типа иерархия. Если перед вызовом данной функции корнем

выборки был установлен конкретный раздел выборки, то данная функция сбрасывает предыдущие установки.

## Примеры

Действие функций **УстановитьКорень** и **УстановитьНачальныйКорень** можно увидеть на примере выполнения следующего фрагмента кода:

```
оОрг = Выборка ("Организации") ;
оОбъектДляПоиска = Таблица ("Организации") ;
оОбъектДляПоиска.Название = "НАЛОГОВЫЕ ОРГАНЫ" ;
Если (Найти (оОбъектДляПоиска, "Название") )
    оОрг.УстановитьКорень (оОбъектДляПоиска) ;
перем оКорВыб = оОрг.КореньВыборки () ;
Сообщить (оКорВыб.Название) ;
оОрг.УстановитьНачальныйКорень () ;
перем оКорВыб = оОрг.КореньВыборки () ;
Сообщить (оКорВыб.Название) ;
```

## См. также:

- [КореньВыборки](#)
- [УстановитьСвязнуюЗапись](#)
- [СвязнаяЗапись](#)
- [УстановитьКорень](#)

## КореньВыборки

Возвращает объект с корневой записью выборки.

## Синтаксис

**Запись** . **КореньВыборки** ( )

## Параметры

Не указываются.

## Возвращает

**Объект**, содержащий корневую запись выборки.

## Комментарии

Функция имеет смысл, если выборка является иерархической, т.е. должно быть поле типа иерархия.

## Примеры

Ищем в справочнике «Организации» раздел «Налоговые органы» и устанавливаем его корнем выборки:

```
oOrg = Выборка ("Организации") ;
oОбъектДляПоиска = Таблица ("Организации") ;
oОбъектДляПоиска.Название = "НАЛОГОВЫЕ ОРГАНЫ" ;
Если (Найти (oОбъектДляПоиска, "Название") )
    oOrg.УстановитьКорень (oОбъектДляПоиска) ;
перем oКорВыб = oOrg.КореньВыборки () ;
```

## См. также:

- [УстановитьКорень](#)
- [УстановитьСвязнуюЗапись](#)
- [СвязнаяЗапись](#)

## УстановитьПараметрыВыборки

Устанавливает параметры выборки.

## Синтаксис

```
Запись.УстановитьПараметрыВыборки (Объект)
```

## Параметры

**Объект** (переменная) – имя объекта с параметрами выборки.

## Возвращает

Да (логическое значение) – установлены параметры выборки.

Нет (логическое значение) – не установлены параметры выборки.

## Примеры

Установим период отбора накладных в реестре приходных накладных:

```
oOrg = Выборка ("Приходные накладные") ;
```

```
oПарам = oОрг.ПараметрыВыборки ( ) ;  
oПарам.ДатНач = 01.01.05 ;  
oПарам.ДатКнц = 31.12.05 ;  
пУст = oОрг.УстановитьПараметрыВыборки (oПарам) ;
```

**См. также:**

- [Выборка](#)
- [ПараметрыВыборки](#)

## ПараметрыВыборки

Возвращает объект с параметрами выборки.

**Синтаксис**

```
Запись . ПараметрыВыборки ( )
```

**Параметры**

Не указываются.

**Возвращает**

Объект с параметрами выборки.

**Примеры**

Вывести список параметров выборки в справочнике «Организации»:

```
пОрг = Выборка ("Организации") ;  
пПарам = пОрг.ПараметрыВыборки ( ) ;
```

**См. также:**

- [Выборка](#)
- [УстановитьПараметрыВыборки](#)

## ЗафиксироватьВыборку

Фиксирует последние действия, сделанные в выборке.

**Синтаксис**

Запись .ЗафиксироватьВыборку ( )

### Параметры

Не указываются.

### Возвращает

Ничего не возвращает.

### Примеры

В нижеописанном примере происходит следующее: при переходе в справочнике организаций в указанный раздел мы зафиксировали все находящиеся записи в этом разделе. Далее, после добавления новой записи в этот раздел, перебираем записи этого раздела. При этом перебираются только старые записи (новая запись перебираться не будет):

```
oOrg = Выборка ("Организации" );  
пНазваниеРаздела = "ОРГАНИЗАЦИИ" ;  
oOrg1 = Выборка ("Организации" );  
oOrg1 .Очистить ( ) ;  
oOrg1 .Название = "Апрель-плюс" ;  
oРаз = oOrg .НайтиРаздел (пНазваниеРаздела ) ;  
oOrg .ПерейтиВРаздел (пНазваниеРаздела ) ;  
oOrg .ЗафиксироватьВыборку ( ) ;  
Связать (oOrg1 .Раздел , oРаз ) ;  
oOrg1 .Добавить ( ) ;  
Пока (Следующий (oOrg )  
    Сообщить (oOrg .Название ) ;
```

### См. также:

- [СнятьФиксациюВыборки](#)

## СнятьФиксациюВыборки

Снимает последние зафиксированные действия, сделанные в выборке.

### Синтаксис

Запись .СнятьФиксациюВыборки ( )

### Параметры

Не указываются.

## Возвращает

Ничего не возвращает.

## Примеры

В нижеописанном примере при зафиксированном переходе в раздел справочника «Организации» перебираются записи только этого раздела, несмотря на попытку перейти в раздел «Служебные» и перебрать там записи:

```
oOrg = Выборка ("Организации") ;
пНазваниеРаздела = "ОРГАНИЗАЦИИ" ;
oOrg . ПерейтиВРаздел (пНазваниеРаздела) ;
oOrg . ЗафиксироватьВыборку ( ) ;
Пока (Следующий (oOrg) )
    Сообщить (oOrg . Название) ;
Сообщить (1) ;
oOrg . ПерейтиВРаздел ("СЛУЖЕБНЫЕ") ;
Пока (Следующий (oOrg) )
    Сообщить (oOrg . Название) ;
```

В следующем примере после снятия фиксации выборки мы перейдем в раздел «Служебные» и переберем записи уже в этом разделе:

```
oOrg = Выборка ("Организации") ;
...
oOrg . СнятьФиксациюВыборки ( ) ;
пР = oOrg . ПерейтиВРаздел ("СЛУЖЕБНЫЕ") ;
Пока (Следующий (oOrg) )
    Сообщить (oOrg . Название) ;
```

## См. также:

- [ЗафиксироватьВыборку](#)

## Сортировать

Сортирует записи выборки по указанным полям.

## Синтаксис

```
Запись . Сортировать ( [Поле1 [ , Поле2... ] )
```

## Параметры

**Поле1,Поле2,...** (текст) – имена полей, по которым нужно отсортировать выборку. По умолчанию сортировка идёт в порядке возрастания значений полей, если требуется наоборот, нужно поставить перед именем поля знак минус « - ». Текстовые поля сортируются только по первому 31-ому символу.

Если ни одного поля не указано, то с выборки будет снята вся временная сортировка. В большинстве случаев это означает, что выборка будет отсортирована по какому-то из индексов основной таблицы.

## Возвращает

Ничего не возвращает.

## Примеры

Функция «Сортировать» чаще всего используется для сортировки отчётов. В этом случае очень важно указать вызов в разделе описания переменных, а не в разделе «Начало таблицы».

Например, чтобы отсортировать на печать справочник сотрудников по дате рождения в обратном порядке, нужно написать в разделе описания переменных данного отчёта следующее.

```
oОрг = Выборка ("Сотрудники" );  
oОрг.Сортировать (" -ДатаРождения" );
```

Если же нужно отсортировать по полу (то есть сначала женщин, потом мужчин), а внутри по дате рождения в обратном порядке:

```
oОрг.Сортировать ("Пол" , " -ДатаРождения" );
```

После печати такого отчёта справочник сотрудников так и останется отсортирован по полу и году рождения, что не очень удобно. Можно сбросить сортировку, указав в разделе «Конец таблицы» такой вызов:

```
oОрг.Сортировать ( ) ;
```

## Таблица – для объекта

Берёт запись основной таблицы от записи выборки из базы данных.

## Синтаксис

```
Запись . Таблица ( )
```

## Параметры

Не указываются.

## Возвращает

Объект, содержащий запись основной таблицы от записи выборки.

## Примеры

Функция может понадобиться только в достаточно редких случаях, когда функциональность выборки мешает выполнению операций. Например, при добавлении документа может возникать сообщение «Запись не удовлетворяет текущим условиям выборки». Чтобы этого избежать, можно воспользоваться такой конструкцией:

```
перем оДокумент = Выборка ("Входящие платежи");  
... # здесь нужно проинициализировать поля  
оДокумент.а  
оДокумент.Таблица ();
```

## См. также:

- [Запись](#)
- [Таблица](#)
- [Выборка](#)

## Развернуть

Разворачивает реестр без узлов, аналогично комбинации <Ctrl+V>.

## Синтаксис

```
Запись . Развернуть ()
```

## Параметры

Не указываются.

## Возвращает

Да (логическое значение) – если реестр развернут.

Нет (логическое значение) – если реестр не развернут.

## Примеры

Покажем содержимое справочника «Организации» без указания разделов иерархии:

```
oOrg = Окно ("Организации") ;  
oЗапOrg = oOrg.Запись () ;  
oЗапOrg.Развернуть () ;  
oOrg.Открыть () ;
```

## См. также:

- [РазвернутьСУзлами](#)
- [Свернуть](#)
- [ПерейтиВРаздел](#)
- [ПерейтиВКорень](#)

## РазвернутьСУзлами

Разворачивает реестр с узлами, аналогично комбинации <Ctrl+B>.

## Синтаксис

```
Запись.РазвернутьСУзлами ()
```

## Параметры

Не указываются.

## Возвращает

**Да** (логическое значение) – если реестр развернут с узлами.

**Нет** (логическое значение) – если реестр не развернут.

## Примеры

Покажем содержимое справочника «Организации» с указанием разделов иерархии:

```
oOrg = Окно ("Организации") ;  
oЗапOrg = oOrg.Запись () ;  
oЗапOrg.РазвернутьСУзлами () ;  
oOrg.Открыть () ;
```

### См. также:

- [Развернуть](#)
- [Свернуть](#)
- [ТекущийРаздел](#)
- [НайтиРаздел](#)
- [ПерейтиВРаздел](#)
- [ПерейтиВКорень](#)

## Свернуть

Сворачивает развернутый реестр, аналогично нажатию клавиши <Esc>.

### Синтаксис

Запись .Свернуть ( )

### Параметры

Не указываются.

### Возвращает

Да (логическое значение) – если реестр свернут.

Нет (логическое значение) – если реестр не свернут.

### Примеры

Свернем развернутый справочник «Организации»:

```
oOrg = Окно ("Организации") ;  
oЗапOrg = oOrg.Запись ( ) ;  
oЗапOrg.Свернуть ( ) ;
```

### См. также:

- [Развернуть](#)
- [РазвернутьСУзлами](#)



## ТекущийРаздел

Возвращает текущий раздел выборки.

### Синтаксис

Запись . ТекущийРаздел ( )

### Параметры

Не указываются.

### Возвращает

Объект – запись текущего раздела.

### Примеры

Узнаем название текущего раздела справочника:

```
oOrg = Выборка ("Организации" );  
oРаздел = oOrg.ПерейтиВРаздел ("НАЛОГОВЫЕ ОРГАНЫ" );  
Если (oРаздел)  
{  
    oТР = oOrg.ТекущийРаздел ( ) ;  
    Сообщить (oТР.Название) ;  
}
```

### См. также:

- [НайтиРаздел](#)
- [ПерейтиВРаздел](#)
- [ПерейтиВКорень](#)

## НайтиРаздел

Возвращает указанный раздел выборки, если он существует.

### Синтаксис

Запись . НайтиРаздел (НазвРаздела)

### Параметры

НазвРаздела (текст) – название раздела, который необходимо найти.

## Возвращает

**Объект** – запись раздела, который необходимо найти.

## Комментарии

При указании названия раздела следует указывать точное название раздела, соблюдая регистры.

## Примеры

Узнаем количество организаций в разделе «НАЛОГОВЫЕ ОРГАНЫ»:

```
oOrg = Выборка ("Организации" );
перем пСч = 0;
Если (oOrg.НайтиРаздел ("НАЛОГОВЫЕ ОРГАНЫ" )
{
    oOrg.ПерейтиВРаздел ("НАЛОГОВЫЕ ОРГАНЫ" );
    Пока (Следующий (oOrg) )
        пСч ++
}
Сообщить ("В разделе " + oРаздел.Название + " " + пСч
+ " организаций" );
```

## См. также:

- [ТекущийРаздел](#)
- [ПерейтиВРаздел](#)
- [ПерейтиВКорень](#)

## ПерейтиВРаздел

Переходит в указанную папку выборки.

## Синтаксис

**Запись** . **ПерейтиВРаздел** (Папка)

## Параметры

**Папка** (текст или объект) – название папки или запись папки, в которую необходимо перейти. Если в названии папки присутствует точка, то для



того, чтобы попасть именно в неё, необходимо в функции вместо точки указать «\\...».

## Возвращает

**Да** (логическое значение) – если перешли в указанную папку.

**Нет** (логическое значение) – если не удалось перейти в указанную папку.

## Комментарии

При указании названия раздела следует указывать точное название раздела, соблюдая регистры.

## Примеры

Узнаем названия организаций, указанных в разделе «ОРГАНИЗАЦИИ.ФОНДЫ и госструктуры»:

```
oOrg = Выборка ("Организации" );
пНазваниеРаздела = "ОРГАНИЗАЦИИ.ФОНДЫ и
госструктуры" ;
Если ( !oOrg.ПерейтиВРаздел (пНазваниеРаздела) )
{
    Ошибка ("Не найден раздел " + пНазваниеРаздела) ;
}
Иначе
{
    Пока (Следующий (oOrg) )
        Сообщить (oOrg.Название) ;
}
```

А теперь тоже самое сделаем для папки «ОРГАНИЗАЦИИ.», в названии которой присутствует точка:

```
oOrg = Выборка ("Организации" );
пНазваниеРаздела = "ОРГАНИЗАЦИИ\\...ФОНДЫ и
госструктуры" ;
...
...
...
```

## См. также:

- [ТекущийРаздел](#)
- [НайтиРаздел](#)

- [НазваниеРаздела](#)
- [ПерейтиВКорень](#)

## ПерейтиВКорень

Переходит в корневой раздел выборки.

### Синтаксис

Запись `.ПерейтиВКорень ( )`

### Параметры

Не указываются.

### Возвращает

Да (логическое значение) – если перешли в корень.

Нет (логическое значение) – если не перешли в корень.

### Примеры

Узнаем названия организаций указанного раздела, если такой есть, или в корне выборке, если такой раздел не существует:

```
oOrg = Выборка ("Организации") ;
пНазваниеРаздела = "ОРГАНИЗАЦИИ" ;
Если (oOrg . ПерейтиВРаздел (пНазваниеРаздела) )
{
    Пока (Следующий (oOrg) )
        Сообщить (oOrg . Название) ;
}
Иначе
{
    oOrg . ПерейтиВКорень ( ) ;
    Пока (Следующий (oOrg) )
        Сообщить (oOrg . Название) ;
}
```

### См. также:

- [ТекущийРаздел](#)



- [НайтиРаздел](#)
- [ПерейтиВРаздел](#)

## ОбновитьВыборки

Обновляет выборки.

### Синтаксис

**ОбновитьВыборки** ( [Имя] )

### Параметры

**Имя** (текст) – имя таблицы, на которой основаны используемые программой выборки.

### Комментарии

Если имя таблицы не указано, обновит все действующие выборки, то есть заново перечитает данные из базы данных.

### Возвращает

Ничего не возвращает.

### Примеры

Обновить выборки таблицы базы данных «Документы»:

**ОбновитьВыборки** ("Документы") ;

### См. также:

- [Выборка](#)

## Поиск и перебор записей

### УстановитьМаксКлюч

Устанавливает максимальные ключевые поля указанного индекса в объекте.

### Синтаксис

**Запись . УстановитьМаксКлюч (ИмяИндекса)**

### Параметры

**ИмяИндекса** (текст) – имя индекса таблицы.

### Возвращает

Ничего не возвращает.

### Примеры

Установим в документе максимальные ключевые поля индекса «Группа-Нумерации»:

```
оДок = Таблица ("Документы") ;  
оДок . УстановитьМаксКлюч ("ГруппаНумерации") ;
```

### См. также:

- [УстановитьМинКлюч](#)

## УстановитьМинКлюч

Устанавливает минимальные ключевые поля указанного индекса в объекте.

### Синтаксис

**Запись . УстановитьМинКлюч (ИмяИндекса)**

### Параметры

**ИмяИндекса** (текст) – имя индекса таблицы.

### Комментарий

Если ключ индекса убывающий, то функция установит у этого ключа максимальное значение.

### Возвращает

Ничего не возвращает.

### Примеры

Установим в документе минимальные ключевые поля индекса «Группа-Нумерации»:

```
оДокумент = Таблица ("Документы") ;
оДокумент . УстановитьМинКлюч ("ГруппаНумерации") ;
```

Пример перебора документов:

```
Перем оДок = Таблица ("Документы") ;
оДок . Очистить ( ) ;
оДок . УстановитьМинКлюч ( ) ;
оДок . ТипДокумента = "РасхЧек" ;
оДок . Дата = ДатНач ;
Связать (оДок , оРаздел) ;
Если ( Найти (оДок , "Раздел" ) )
{
    Пока (оДок . Дата <= ДатКнц)
    {
        Сообщить (оДок . Номер) ;
        Если ( !Следующий (оДок , "Раздел" ) ) Прервать ;
    }
}
```

**См. также:**

- [УстановитьМаксКлюч](#)

## Следующий по индексу

Берёт следующую запись таблицы по указанному индексу.

### Синтаксис

```
Следующий (Таблица [ , Индекс ] )
```

### Параметры

**Таблица** (объект) – запись указанной таблицы. Каждый раз при вызове функции сюда будет извлекаться очередная запись.

Если объект содержит «пустую» запись (то есть не извлеченную из базы данных), то будет взята первая запись в индексе. Если же индекс не указан, то извлекается соответственно первая запись таблицы.

Если же объект содержит запись, извлечённую из базы данных, то берётся следующая за ней.

**Индекс** (текст) – имя индекса, по которому определяется следующая запись. Список индексов таблицы можно посмотреть, открыв соответ-

вующий словарь данных (файл с расширением «.dic») и выбрав нужную таблицу.

Если в описании индекса стоит флаг «Нулевые ключи в индекс не вносятся», то записи, с нулевыми ключевыми полями в такой индекс не попадают. Соответственно такие записи не будут извлекаться функцией по этому индексу.

Если указан индекс иерархии, то записи будут перебираться в том же порядке, в котором вы видите записи при нажатии <Ctrl+B> в иерархической таблице. Записи, находящиеся вне иерархии, извлекаться в этом случае не будут. Чтобы перебрать по индексу иерархии записи, относящиеся к определённому уровню, можно использовать функцию «Найти».

Если имя индекса не указано, то берётся первый индекс иерархии. Если же в таблице нет иерархии или в качестве имени индекса указана пустая строка – "", записи извлекаются в том порядке, в котором они лежат в таблице ( то есть фактически без всякого порядка). В этом случае гарантированно перебираются все записи таблицы.

Перебор всех записей таблицы может занимать значительное время, в большинстве случаев лучше сначала найти подходящую запись функцией «Найти» и перебирать записи, пока они отвечают некоторому условию.

## Возвращает

Да (логическое значение) – была извлечена следующая запись.

Нет (логическое значение) – больше нет записей.

## Примеры

Для перебора основных средств (в том числе и разделов) в порядке инвентарных номеров достаточно написать:

```
oСредство = Таблица ("Основные средства") ;  
Пока (Следующий (oСредство , "ИнвНомер" ) )  
Сообщить (oСредство .ИнвНомер) ;
```

В таблице «Организации» перебрать записи по указанному индексу «ИНН» и сообщить название этих организаций:

```
oОрг = Таблица ("Организации" ) ;  
Пока (Следующий (oОрг , "ИНН" ) )  
Сообщить (oОрг .Название) ;
```

**См. также:**

- [Предыдущий по индексу](#)
- [УстановитьМаксКлюч](#)
- [УстановитьМинКлюч](#)
- [Найти – по индексу](#)
- [Найти – по значению индекса](#)

**Предыдущий по индексу**

Берёт предыдущую запись таблицы по указанному индексу.

**Синтаксис**

**Предыдущий** (Таблица [ , Индекс ] )

**Параметры**

**Таблица** (объект) – запись указанной таблицы. Каждый раз при вызове функции сюда будет извлекаться очередная запись.

Если объект содержит «пустую» запись (то есть не извлеченную из базы данных), то будет взята последняя запись в индексе. Если же индекс не указан, то извлекается соответственно последняя запись таблицы.

Если же объект содержит запись, извлечённую из базы данных, то берётся предыдущая.

**Индекс** (текст) – имя индекса, по которому определяется предыдущая запись. Список индексов таблицы можно посмотреть, открыв соответствующий словарь данных (файл с расширением «.dic») и выбрав нужную таблицу.

Если в описании индекса стоит флаг «Нулевые ключи в индекс не вносятся», то записи, с нулевыми ключевыми полями в такой индекс не попадают. Соответственно такие записи не будут извлекаться функцией по этому индексу.

Если указан индекс иерархии, то записи будут перебираться в том же порядке, в котором вы видите записи при нажатии <Ctrl+B> в иерархической таблице. Записи, находящиеся вне иерархии, извлекаться в этом случае не будут. Чтобы перебрать по индексу иерархии записи, относящиеся к определённому уровню, можно использовать функцию «Найти».

Если имя индекса не указано, то берётся первый индекс иерархии. Если же в таблице нет иерархии или в качестве имени индекса указана пустая строка – "", записи извлекаются в том порядке, в котором они лежат в таблице ( то есть фактически без всякого порядка). В этом случае гарантированно перебираются все записи таблицы.

Перебор всех записей таблицы может занимать значительное время, в большинстве случаев лучше сначала найти подходящую запись функцией «Найти» и перебирать записи, пока они отвечают некоторому условию.

### Возвращает

**Да** (логическое значение) – была извлечена предыдущая запись.

**Нет** (логическое значение) – больше нет записей.

### Примеры

Функцию удобно применять, например, при поиске максимального табельного номера сотрудника:

```
oСотр = Таблица ("Сотрудники") ;  
Пока (Предыдущий (oСотр, "" )  
Сообщить (oСотр . ТабНомер)
```

### См. также:

- [Следующий по индексу](#)
- [УстановитьМаксКлюч](#)
- [УстановитьМинКлюч](#)
- [Найти – по индексу](#)
- [Найти – по значению индекса](#)

### Следующий по связи

Извлекает следующую запись таблицы, связанную с указанной записью.

### Синтаксис

```
Следующий (Основной, Связанный [ , Связь ] )
```

### Параметры



**Основной** (объект) – объект, содержащий запись основной таблицы. При вызове функции эта запись не изменяется.

**Связанный** (объект) – объект, содержащий запись связанной таблицы. Каждый раз при вызове функции сюда будет извлекаться очередная запись.

Если объект содержит «пустую» запись (то есть не извлеченную из базы данных), то будет взята первая (для функции «Следующий») или последняя (для функции «Предыдущий») связанная запись. Если же объект уже содержит некоторую связанную запись, то берётся следующая за ней (или предыдущая для функции «Предыдущий»).

**Связь** (текст) – имя связи, по которой будет извлекаться связанная запись. В большинстве случаев этот параметр можно не указывать, программа сама определит имя связи, по которой связаны записи, указанные в первых двух параметрах. Возможна, однако, ситуация, когда между двумя указанными таблицами существует больше одной связи. В этом случае как раз и нужно использовать параметр «Связь»!

## Возвращает

**Да** (логическое значение) – была извлечена следующая запись.

**Нет** (логическое значение) – больше нет записей.

## Примеры

Узнаем расчётные счета организации оЛицо1 по указанной связи:

```
oРасСч = Таблица ("Расчетные счета");  
oЛицо1 = НайтиЛицо ("7605016030", "Орг");  
Пока (Следующий (oЛицо1, oРасСч, "ЛицоРСчета"))  
    Сообщить (oРасСч.РСчет);
```

Так как существует только одна связь с таблицей «Расчетные счета», то можно в примере эту связь не указывать (программа ее установит по умолчанию):

```
oРасСч = Таблица ("Расчетные счета");  
oЛицо1 = НайтиЛицо ("7605016030", "Орг");  
Пока (Следующий (oЛицо1, oРасСч))  
    Сообщить (oРасСч.РСчет);
```

## См. также:

- [Предыдущий по связи](#)

- [Следующий по индексу](#)
- [Предыдущий по индексу](#)

## Предыдущий по связи

Извлекает предыдущую запись таблицы, связанную с указанной записью.

### Синтаксис

**Предыдущий** (**Основной**, **Связанный** [ , **Связь** ] )

### Параметры

**Основной** (объект) – объект, содержащий запись основной таблицы. При вызове функции эта запись не изменяется.

**Связанный** (объект) – объект, содержащий запись связанной таблицы. Каждый раз при вызове функции сюда будет извлекаться очередная запись.

Если объект содержит «пустую» запись (то есть не извлеченную из базы данных), то будет взята первая (для функции «Следующий») или последняя (для функции «Предыдущий») связанная запись. Если же объект уже содержит некоторую связанную запись, то берётся следующая за ней (или предыдущая для функции «Предыдущий»).

**Связь** (текст) – имя связи, по которой будет извлекаться связанная запись. В большинстве случаев этот параметр можно не указывать, программа сама определит имя связи, по которой связаны записи, указанные в первых двух параметрах. Возможна, однако, ситуация, когда между двумя указанными таблицами существует больше одной связи. В этом случае как раз и нужно использовать параметр «Связь»!

### Возвращает

**Да** (логическое значение) – была извлечена предыдущая запись.

**Нет** (логическое значение) – больше нет записей.

### Пример

Примеры по аналогии с функцией «Следующий по связи».

**См. также:**

- [Следующий по связи](#)
- [Следующий по индексу](#)
- [Предыдущий по индексу](#)

**Найти - по индексу**

Ищет в таблице запись с начала по указанному индексу.

**Синтаксис**

**Найти (Объект , Индекс)**

**Параметры**

**Объект** (объект) – объект, содержащий запись. В случае успешного поиска сюда будет извлечена найденная запись.

Если объект на момент вызова функции содержит запись, ещё не извлеченную из базы данных, то в этой записи должны быть установлены ключевые поля. По значениям, указанным в ключевых полях, и будет вестись поиск.

Если же объект содержит запись, уже извлечённую из базы данных, то берётся по указанному индексу следующая запись.

**Индекс** (текст) – имя индекса. Именно индексом определяется, по каким полям и в какой последовательности будет искаться запись. Список индексов каждой таблицы можно посмотреть, открыв соответствующий словарь данных (файл с расширением «.dic») и выбрав нужную таблицу. Там же можно посмотреть и список полей, входящих в каждый из индексов.

При проведении поиска учтите, что, во-первых, некоторые поля в индексе могут быть отсортированы в обратном порядке, а, во-вторых, индекс может строиться не по целому полю, а только по его части (так индексируется большинство текстовых полей).

**Возвращает**

**1** – найдена запись с указанными значениями ключевых полей. Найденная запись устанавливается в объекте, указанном при вызове функции.

**0** – нужная запись не найдена. В этом случае запись в объекте инициализируется нулевыми значениями (становится «пустой»).

-1 - нужная запись не найдена. В этом случае в объекте устанавливается ближайшая запись с индексом, наиболее подходящим по заданному в поиске параметру.

## Примеры

Чтобы найти организацию по ИНН, нужно написать следующее:

```
oOrg = Таблица ("Организации");  
oOrg.ИНН = "7605015333"; # ИНН организации  
Если (Найти (oOrg, "ИНН"))  
    Сообщить (oOrg.Название);  
иначе  
    Сообщить ("Нет такой организации!");
```

Если точное название организации неизвестно, можно выбрать все организации, название которых начинается со слова «ИМНС»:

```
oOrg = Таблица ("Организации");  
oOrg.Название = "ИМНС";  
Пока (Найти (oOrg, "Название"))  
{  
    Если (ПодСтрока (oOrg.Название, 1, 4) != "ИМНС")  
        Прервать; # если не "ИМНС*", выходим из цикла  
    Сообщить (oOrg.ИНН, oOrg.Название);  
}
```

## См. также:

- [Следующий по индексу](#)
- [Предыдущий по индексу](#)
- [УстановитьМаксКлюч](#)
- [УстановитьМинКлюч](#)
- [НайтиСКонца – по индексу](#)
- [Найти – по значению индекса](#)

## НайтиСКонца - по индексу

Ищет в таблице запись с конца по указанному индексу.

## Синтаксис

## НайтиСКонца (Объект, Индекс)

### Параметры

**Объект** (объект) – объект, содержащий запись. В случае успешного поиска сюда будет извлечена найденная запись.

Если объект на момент вызова функции содержит запись, ещё не извлечённую из базы данных, то в этой записи должны быть установлены ключевые поля. По значениям, указанным в ключевых полях, и будет вестись поиск.

Если же объект содержит запись, уже извлечённую из базы данных, то берётся по указанному индексу предыдущая запись.

**Индекс** (текст) – имя индекса. Именно индексом определяется, по каким полям и в какой последовательности будет искаться запись. Список индексов каждой таблицы можно посмотреть, открыв соответствующий словарь данных (файл с расширением «.dic») и выбрав нужную таблицу. Там же можно посмотреть и список полей, входящих в каждый из индексов.

При проведении поиска учтите, что, во-первых, некоторые поля в индексе могут быть отсортированы в обратном порядке, а, во-вторых, индекс может строиться не по целому полю, а только по его части (так индексируется большинство текстовых полей).

### Возвращает

**1** – найдена запись с указанными значениями ключевых полей. Найденная запись устанавливается в объекте, указанном при вызове функции.

**0** – нужная запись не найдена. В этом случае запись в объекте инициализируется нулевыми значениями (становится «пустой»).

**-1** – нужная запись не найдена. В этом случае в объекте устанавливается ближайшая запись с индексом, наиболее подходящим по заданному в поиске параметру.

### Примеры

При помощи функции «НайтиСКонца» можно искать максимальное значение. Например, найти последний номер в реестре документов:

```
пд = Таблица ("Документы") ;  
'пд.ТипДокумента' = 'Документ.ТипДокумента' ;  
'пд.ГруппаНумерации' = 'Документ.ГруппаНумерации' ;  
пд.Номер = 999999999 ; # просто очень большое число
```

Если (НайтиСКонца (пД, "ГруппаНумерации" )  
Сообщить (пД.Номер) ;

### См. также:

- [Найти – по индексу](#)
- [Найти – по значению индекса](#)
- [НайтиСКонца – по значению индекса](#)

## Найти – по значению индекса

Ищет в таблице запись с начала по указанному значению ключевого поля индекса.

### Синтаксис

Найти (Объект , ИмяИндекса , ЗначениеКлючевогоПоля)  
Найти (Объект , ИмяИндекса , ИмяКлючевогоПоля1 ,  
Значение1 [ , ИмяКлючевогоПоля2 , Значение2..... ] )

### Параметры

**Объект** (объект) – объект, содержащий запись. В случае успешного поиска сюда будет извлечена найденная запись.

Если объект на момент вызова функции содержит запись, ещё не извлечённую из базы данных, то в этой записи должны быть установлены ключевые поля. По значениям, указанным в ключевых полях, и будет вестись поиск.

Если же объект содержит запись, уже извлечённую из базы данных, то берётся по указанному индексу следующая запись.

**ИмяИндекса** (текст) – имя индекса. Именно индексом определяется, по каким полям и в какой последовательности будет искаться запись. Список индексов каждой таблицы можно посмотреть, открыв соответствующий словарь данных (файл с расширением «.dic») и выбрав нужную таблицу. Там же можно посмотреть и список полей, входящих в каждый из индексов.

**ЗначениеКлючевогоПоля** (текст) – значение ключевого поля индекса.



**ИмяКлючевогоПоля1, ИмяКлючевогоПоля1,...** (текст) - имена ключевых полей индекса.

**Значение1, Значение2,...** (текст) – значения ключевых полей индекса.

При проведении поиска учтите, что, во-первых, некоторые поля в индексе могут быть отсортированы в обратном порядке, а, во-вторых, индекс может строиться не по целому полю, а только по его части (так индексируется большинство текстовых полей).

## Возвращает

**Да** (логическое значение) – найдена запись с указанными значениями ключевых полей. Найденная запись устанавливается в объекте, указанном при вызове функции.

**Нет** (логическое значение) – нужная запись не найдена. В этом случае запись в объекте инициализируется нулевыми значениями (становится «пустой»).

## Примеры

В таблице «Правила операций» найдем операцию по индексу «Имя» с указанным значением ключевого поля «СверНалБух»:

```
oОбъект = Таблица ("Правила операций") ;  
Если (Найти (oОбъект , "Имя" , "СверНалБух" )  
Сообщить (oОбъект .Операция) ;  
Иначе  
Сообщить ("Не нашли!") ;
```

## См. также:

- [НайтиСКонца – по значению индекса](#)
- [Найти – по индексу](#)
- [НайтиСКонца – по индексу](#)

## НайтиСКонца – по значению индекса

Ищет в таблице запись с конца по указанному значению ключевого поля индекса.

## Синтаксис

**НайтиСКонца** (Объект , ИмяИндекса ,

**ЗначениеКлючевогоПоля)**

**НайтиСКонца (Объект, ИмяИндекса, ИмяКлючевогоПоля1, Значение1 [, ИмяКлючевогоПоля2, Значение2.....] )**

## Параметры

**Объект** (объект) – объект, содержащий запись. В случае успешного поиска сюда будет извлечена найденная запись.

Если объект на момент вызова функции содержит запись, ещё не извлечённую из базы данных, то в этой записи должны быть установлены ключевые поля. По значениям, указанным в ключевых полях, и будет вестись поиск.

Если же объект содержит запись, уже извлечённую из базы данных, то берётся по указанному индексу предыдущая запись.

**ИмяИндекса** (текст) – имя индекса. Именно индексом определяется, по каким полям и в какой последовательности будет искаться запись. Список индексов каждой таблицы можно посмотреть, открыв соответствующий словарь данных (файл с расширением «.dic») и выбрав нужную таблицу. Там же можно посмотреть и список полей, входящих в каждый из индексов.

**ЗначениеКлючевогоПоля** (текст) – значение ключевого поля индекса.

**ИмяКлючевогоПоля1, ИмяКлючевогоПоля1,...** (текст) - имена ключевых полей индекса.

**Значение1, Значение2,...** (текст) – значения ключевых полей индекса.

При проведении поиска учтите, что, во-первых, некоторые поля в индексе могут быть отсортированы в обратном порядке, а, во-вторых, индекс может строиться не по целому полю, а только по его части (так индексируется большинство текстовых полей).

## Возвращает

**Да** (логическое значение) – найдена запись с указанными значениями ключевых полей. Найденная запись устанавливается в объекте, указанном при вызове функции.

**Нет** (логическое значение) – нужная запись не найдена. В этом случае запись в объекте инициализируется нулевыми значениями (становится «пустой»).



## Примеры

В таблице «Организации» найдем организацию с указанным «ИНН» и «Кодом филиала». Поиск ведется с конца индекса:

```
oOrg = Таблица ("Организации") ;
Если (НайтиСКонца (oOrg, "ИНН", "ИНН", "7605015333",
"КодФилиала", "4"))
Сообщить (oOrg.Название) ;
иначе
Сообщить ("Нет такой организации!") ;
```

## См. также:

- [Найти – по значению индекса](#)
- [Найти – по индексу](#)
- [НайтиСКонца – по индексу](#)

## ДляВсех(Записей)

Последовательно перебирает все записи указанной выборки, в том числе и записи внутри разделов.

## Синтаксис

```
ДляВсех (Записей (ИмяВыборки [ , Объект ] ) )
```

## Параметры

**ИмяВыборки** (текст) – имя выборки из базы данных, записи которой будут перебираться. Выборки из базы данных описываются в ресурсах, их имена можно посмотреть, открыв соответствующий файл (файл с расширением «.rs»).

**Объект** (объект) – объект, с записью которого будет связана перебираемая выборка. Дело в том, что есть ряд выборок, которые содержат список записей, связанных с некоторой другой записью. Например, выборка «Проводки по документу» содержит записи журнала операций, связанные с некоторым документом. Вот для таких выборок этот параметр и нужно указывать.

## Возвращает

Ничего не возвращает.

## Комментарии

Внутри цикла «ДляВсех( Записей )» создается контекстный объект с именем «Запись», содержащий запись перебираемой выборки. Поскольку объект контекстный, то к полям этой записи можно обращаться без уточнения имени объекта.

Чтобы внести какие-либо изменения в перебираемые записи, нужно использовать функцию «Сохранить» (смотрите пример ниже).

Чтобы узнать имя некоторой выборки, которая отображается в окне, можно просто нажать в окне <Ctrl+F12>, перейти к интересующей вас табличке, нажать <F3> и прочитать значение в поле «Имя выборки».

## Примеры

Для просмотра проводок по документам-основаниям необходимо написать:

```
ДляВсех(Оснований) # перебираем документы-основания
  ДляВсех(Записей("Проводки по документу", Связь))
    Сообщить('Дебет>Номер счета' + "-" +
      'Кредит>Номер счета' + "=" + Сумма);
```

Чтобы вывести список представителей организации, указанной в поле «Лицо1», можно использовать такой фрагмент:

```
ДляВсех(Записей("Представители", Лицо1))
  Сообщить(ФИО + " телефон " + Телефон);
```

А вот, чтобы поменять поле «Телефон» у представителя организации, нужно использовать функцию «Сохранить».

```
ДляВсех(Записей("Представители", Лицо1))
  Если(Телефон == "")
  {
    Телефон = Лицо1.Телефон;
    Запись.Сохранить();
  }
```

## См. также:

- [Выборка](#)

## КоличествоЗаписей



Возвращает количество записей в выборке.

### **Синтаксис**

**Выборка (Имя)**

### **Параметры**

Не указываются.

### **Возвращает**

**Целое** – количество записей.

### **Примеры**

Необходимо узнать общее количество записей в выборке «Расходные накладные»:

```
ОДок = Выборка ("Расходные накладные") ;  
Сообщить (ОДок . КоличествоЗаписей) ;
```

### **См. также:**

- [Выборка](#)

## **Поля связи, иерархии и прочее**

### **Корень**

Помещает запись указанного объекта в корневой раздел выборки.

### **Синтаксис**

**Запись . Корень ( [Иерархия] )**

### **Параметры**

**Иерархия** (текст) – имя поля иерархии. Если параметр не указан, берётся первое поле типа «Иерархия» в записи.

### **Возвращает**

**Да** (логическое значение) – запись помещена в корневой раздел выборки.

**Нет** (логическое значение) – запись не помещена в корневой раздел выборки.

## Примеры

Добавим новую запись в корень справочника «Организации»:

oОрг = Выборка ("Организации") ;

oОрг.Название = "Чудо" ;

oОрг.Корень ;

oОрг.Добавить ;

## См. также:

- [Выборка](#)
- [Лист](#)
- [Узел](#)
- [ЭтоКорень](#)

## Лист

Делает запись указанного раздела листом (обычной записью, а не разделом иерархии).

## Синтаксис

Запись . Лист ( [Иерархия] )

## Параметры

**Иерархия** (текст) – имя поля иерархии. Если параметр не указан, берётся первое поле типа «Иерархия» в записи.

## Возвращает

**Да** (логическое значение) – запись создана.

**Нет** (логическое значение) – запись не создана.

## Примеры

В предыдущем примере можно было бы написать так, чтобы создаваемая запись была листом:



...

`пОрг.Лист; # чтобы оказалась листом`

...

На самом-то деле эта запись не обязательна, так как изначально при создании запись и так является листом.

Будьте очень осторожны при использовании функции «Лист»! Действие этой функции очень деструктивно, и может нарушить логическую целостность базы данных. Например, если запись является разделом иерархии, и в этом разделе есть другие записи, то такую запись нельзя делать листом.

### См. также:

- [Корень](#)
- [Узел](#)
- [ЭтоЛист](#)

## СкрытыйУзел

Создает лист, на который могут быть ссылки.

### Синтаксис

`Запись . СкрытыйУзел ( [Иерархия] )`

### Параметры

**Иерархия** (текст) – имя поля иерархии. Если параметр не указан, берётся первое поле типа «Иерархия» в записи.

### Возвращает

**Да** (логическое значение) – запись создана.

**Нет** (логическое значение) – запись не создана.

### См. также:

- [Корень](#)
- [Лист](#)
- [Узел](#)
- [ЭтоСкрытыйУзел](#)

## Узел

Делает запись указанного объекта разделом иерархии.

### Синтаксис

**Запись . Узел ( [Иерархия] )**

### Параметры

**Иерархия** (текст) – имя поля иерархии. Если параметр не указан, берётся первое поле типа «Иерархия» в записи.

### Возвращает

**Да** (логическое значение) – удалось сделать запись разделом иерархии.

**Нет** (логическое значение) – не удалось сделать запись разделом иерархии.

### Примеры

Например, добавим в корень справочника организаций запись раздела с именем «Поставщики».

```
пОрг = Выборка ("Организации") ;  
пОрг.Название = "ПОСТАВЩИКИ" ;  
пОрг.Лицо_Название = "ПОСТАВЩИКИ" ;  
пОрг.Лицо_Наименование = "ПОСТАВЩИКИ" ;  
пОрг.Корень ; # чтобы оказалась в корне справочника  
пОрг.Узел # чтобы оказалась разделом справочника  
пОрг.Добавить
```

### См. также:

- [Корень](#)
- [Лист](#)
- [СкрытыйУзел](#)
- [ЭтоУзел](#)

## ЭтоКорень

Проверяет, находится ли запись указанного объекта в корне иерархии.



## Синтаксис

Запись .ЭтоКорень ( [Иерархия] )

## Параметры

**Иерархия** (текст) – имя поля иерархии. Если параметр не указан, берётся первое поле типа «Иерархия» в записи.

## Возвращает

**Да** (логическое значение) – запись находится в корне иерархии.

**Нет** (логическое значение) – запись находится в каком-либо подразделе или вообще вне данной иерархии.

## Примеры

Чтобы проконтролировать, находится ли запись, указанная в поле «Лицо1», в корне справочника, нужно написать так:

```
Если (Лицо1 . Лицо_ . ЭтоКорень)  
  Ошибка ("Запись не должна быть в корне  
  справочника!") ;
```

## См. также:

- [Корень](#)

## ЭтоУзел

Проверяет, является ли запись указанного объекта разделом иерархии.

## Синтаксис

Запись .ЭтоУзел ( [Иерархия] )

## Параметры

**Иерархия** (текст) – имя поля иерархии. Если параметр не указан, берётся первое поле типа «Иерархия» в записи.

## Возвращает

**Да** (логическое значение) – запись указанного объекта является узлом (разделом иерархии).

**Нет** (логическое значение) – запись является листом (не разделом иерархии).

## Примеры

Чтобы запретить закрывать документ, если в поле «Лицо1» указана группа лиц, можно написать так:

```
Если (Лицо1 . Лицо_ . ЭтоУзел )  
    Ошибка ("В Лицо1 не должно быть раздела  
    справочника! ") ;
```

Проверить, является ли запись справочника Организации узлом:

```
пСотр = Лицо ("X05", "Организации") ;  
пСотр . Лицо_ . ЭтоУзел ( ) ;
```

## См. также:

- [Выборка](#)
- [Узел](#)
- [СкрытыйУзел](#)

## ЭтоСкрытыйУзел

Проверяет, является ли запись скрытым узлом.

## Синтаксис

```
Запись . ЭтоСкрытыйУзел ( [Иерархия] )
```

## Параметры

**Иерархия** (текст) – имя поля иерархии. Если параметр не указан, берётся первое поле типа «Иерархия» в записи.

## Возвращает

**Да** (логическое значение) – запись указанного объекта является скрытым узлом.

**Нет** (логическое значение) - запись указанного объекта не является скрытым узлом.



### См. также:

- [Выборка](#)
- [Узел](#)
- [СкрытыйУзел](#)

## ЭтоЛист

Проверяет, является ли запись указанного раздела листом (обычной записью, а не разделом иерархии).

### Синтаксис

**Запись .ЭтоЛист ( [Иерархия] )**

### Параметры

**Иерархия** (текст) – имя поля иерархии. Если параметр не указан, берётся первое поле типа «Иерархия» в записи.

### Возвращает

**Да** (логическое значение) – запись указанного объекта является листом (обычной записью, а не разделом иерархии).

**Нет** (логическое значение) – запись не является листом(а разделом иерархии).

### См. также:

- [Лист](#)

## Связать

Устанавливает поле связи (связывает записи).

### Синтаксис

**Связать (ПолеСвязи, Объект)**

### Параметры

**ПолеСвязи** (идентификатор) – имя устанавливаемого поля связи.

**Объект** (объект) – объект, адрес записи которого будет помещён в указанное поле связи. Для очистки поля связи (разрыва связи), можно использовать специальный объект с именем «Нет» (смотрите пример). Учтите, что объект должен содержать запись уже вставленную в базу данных. То есть, если вы добавляете новый объект, то сначала нужно будет вызвать функцию «Добавить», а потом уже использовать объект в качестве второго параметра функции «Связать»

## Возвращает

**Да** (логическое значение) – записи связаны.

**Нет** (логическое значение) – записи не связаны.

## Примеры

Установить в поле «Лицо2» документа лицо из поля «Лицо1»:

```
Связать (Лицо2 , Лицо1 ) ;
```

Установить в поле «Лицо1» документа организацию с ИНН 7605012345:

```
Связать (Лицо1 , Лицо ("7605012345" , "Орг")) ;
```

Очистить поле «Лицо2»:

```
Связать (Лицо2 , Нет) ;
```

Установить в документе правило операции с именем «НачЗП».

```
оТаблица = Таблица ("Правила операций") ;
```

```
оТаблица.Имя = "НачЗП" ;
```

```
Если (Найти (оТаблица , "Имя"))
```

```
Связать (ПравилоДокументы , оТаблица) ;
```

Поместить запись организации, указанной в поле «Лицо2» в раздел справочника организаций с ИНН 100:

```
оЛицо = Лицо ("100" , "Орг") ;
```

```
оОрг = Лицо2.Лицо_ ;
```

```
Связать (пОрг.Раздел , оЛицо.Лицо_ ) ;
```

```
оОрг.Сохранить ( ) ;
```

## См. также:

- [Таблица](#)
- [Выборка](#)
- [Запись](#)



- [Запись Таблицы](#)
- [Активировать](#)

## Активировать

Активирует поле связи.

### Синтаксис

**Активировать (ПолеСвязи [ , Таблица ] )**

### Параметры

**ПолеСвязи** (идентификатор) – имя активируемого поля связи.

**Таблица** (текст) – имя таблицы, с которой нужно активировать связь. Данный параметр необходим только для условной связи при активации связи со стороны таблицы, ссылающейся на несколько разных таблиц. Например, запись таблицы «Лицо» может ссылаться по связи «Лицо\_» на таблицу «Аналитики», или «Организации», или «Сотрудники». Чтобы определить, с записью какой таблицы активируется связь и нужен данный параметр. В остальных случаях можно просто указать пустую строку.

### Комментарии

Активация связи может потребоваться при создании записей, всегда идущих парами, например, записей таблицы «Лицо» (смотрите пример ниже).

### Возвращает

Ничего не возвращает.

### Примеры

Чтобы создать запись организации, нужно фактически создать две связанные между собой записи – в таблице «Лицо» и в таблице «Организации». Это и делается в нижеследующем фрагменте при активации связи «Лицо\_».

```
oТаблица = Таблица ("Лица") ;  
Активировать (oТаблица .Лицо_ , "Организации") ;  
oТаблица .Название = "Пуск ЗАО" ;  
oТаблица .Лицо_ .Название = "Пуск ЗАО" ;  
oТаблица .Лицо_ .Наименование = "ЗАО Пуск  
Интернэйшенел" ;
```

```
oТаблица.Лицо_.ИНН = "7623124312";  
oТаблица.Лицо_.Корень;  
# чтобы оказалась в корне справочника.  
oТаблица.Добавить; # добавляем запись в базу данных.
```

### См. также:

- [Таблица](#)
- [Выборка](#)
- [Запись](#)
- [ЗаписьТаблицы](#)
- [Связать](#)

## Флаг

Проверяет, установлено ли указанное значение у поля типа «Флаги».

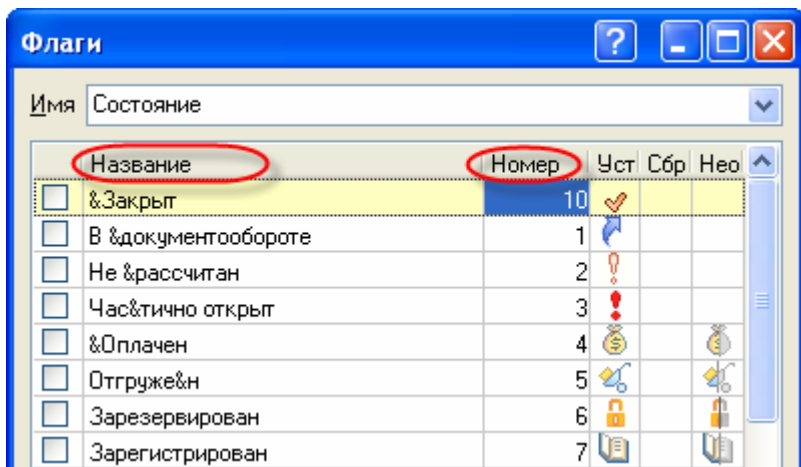
### Синтаксис

**флаг** (Поле, Название | Номер | Номер п/п)

### Параметры

**Поле** (идентификатор | целое) – имя проверяемого поля типа «Флаги».

**Название|Номер** (текст | целое) – проверяемое название или номер значения флага. Нумерация значений флага начинается с «0». Все возможные значения проверяемого поля можно посмотреть, открыв в словаре данных описание нужной таблицы. Затем открыть в ней описание интересующего поля и прочитать список значений (колонки «Название» и «Номер»).



В качестве значения может выступать номер значения, указанного по п/п. Нумерация начинается с «0» с верхней строки таблицы. В этом случае при установке значения флага с номером п/п надо писать "#+Номер п/п". Например, чтобы установить значение флага «Закрыт» можно написать либо "+Закрыт", либо "+10" (как показано на рисунке), либо "#+0".

### Возвращает

1 – указанное значение установлено в поле флагов. Т.е. в поле установлена галочка - .

0 – указанное значение не установлено в поле флагов. Поле принимает вид: .

-1 - значение неопределенно. Поле принимает вид: ?

-2 – флаг установлен, но значение флага неопределенно. Поле принимает вид: ?

### Комментарий

Последние два значения, возвращаемые функцией «Флаг», могут эффективно использоваться пользователем, например, чтобы отличать полностью оплаченные документы по сделке от частично-оплаченных. В программе такие документы имеет разные иконки: и соответственно, но одинаковое состояние поля «Оплачен»: неопределенно (?).

## Примеры

При выборе товара в расходной накладной можно проверить значение поля «Разрешены скидки»:

```
ДляВсех (Наименований)  
Сообщить (Флаг (Наим .СкладРасходы .Признаки ,  
"Разрешены скидки" ) ) ;
```

Тот же результат мы получим, если вместо названия значения поля укажем номер этого значения:

```
ДляВсех (Наименований)  
Сообщить (Флаг (Наим .СкладРасходы .Признаки , 2) ) ;
```

А таким образом можно установить значение флага в 1:

```
Наим .СкладРасходы .Признаки = "+Разрешены скидки" ;
```

...или в 0:

```
Наим .СкладРасходы .Признаки = "-Разрешены скидки" .
```

Узнаем номера частично оплаченных документов в реестре расходных накладных, воспользовавшись окном поиска по выражению <Ctrl+F>:

```
Если (Флаг (Состояние , "Оплачен" ) == -1)  
Сообщить (Документ .Номер)
```

## Регистрация изменений

### ВЖурналИзменений

Добавляет в журнал регистрации новую запись об изменениях в базе данных программы.

### Синтаксис

```
ВЖурналИзменений (Изменение)
```

### Параметры

Изменение (текст) – запись об изменении в базе данных.

### Возвращает



Ничего не возвращает.

### **Комментарий**

Записи этого журнала можно просмотреть в задаче «Администратор» в меню «База Данных».

### **См. также:**

- [ВЖурналОшибок](#)
- [ВЖурналСообщений](#)

## **ВЖурналОшибок**

Добавляет записи об ошибках в программе в протокол ошибок.

### **Синтаксис**

**ВЖурналОшибок (Сообщение)**

### **Параметры**

**Сообщение** (текст) – запись об ошибках в базе данных.

### **Возвращает**

Ничего не возвращает.

### **Комментарий**

Записи этого журнала можно просмотреть в задаче «Администратор» в меню «База Данных».

### **См. также:**

- [ВЖурналИзменений](#)
- [ВЖурналСообщений](#)

## **ВЖурналСообщений**

Добавляет запись сообщения в «Журнал сообщений» с возможностью перехода к передаваемому объекту по его адресу.

## Синтаксис

**ВЖурналСообщений** (Сообщение [ , Объект ] )

## Параметры

**Сообщение** (текст) – текст сообщения с ссылкой на передаваемый объект.

**Объект** (объект) – объект, содержащий запись, извлеченную из таблицы базы данных. Если данный параметр не указан, то в журнал добавляется текст обычного сообщения.

## Возвращает

Ничего не возвращает.

## Комментарий

Записи этого журнала можно просмотреть в любой задаче комплекса в меню «Сервис/ Вспомогательные средства/ Журнал сообщений».

Если в Журнале сообщений включен флаг «Перехватывать сообщения», то в него будут попадать и записи вызова функции «Сообщить».

## См. также:

- [ВЖурналИзменений](#)
- [ВЖурналОшибок](#)
- [ОткрытьЖурналСообщений](#)
- [ЗапуститьЖурналСообщений](#)
- [ОстановитьЖурналСообщений](#)
- [ОчиститьЖурналСообщений](#)

## НайтиЖурналСообщений

Находит запись журнала сообщений, если он открыт.

## Синтаксис

**НайтиЖурналСообщений** ( )



## Параметры

Не указываются.

## Возвращает

**Объект** – запись выборки «Протокол сообщений». Если окно с журналом сообщений не открыто, то функция возвратит пустую запись.

## См. также:

- [ОткрытьЖурналСообщений](#)
- [ВЖурналСообщений](#)
- [ЗапуститьЖурналСообщений](#)
- [ОстановитьЖурналСообщений](#)
- [ОчиститьЖурналСообщений](#)

## ОткрытьЖурналСообщений

Открывает окно с журналом сообщений. После этого все ошибки и сообщения будут попадать в него, вместо того, чтобы быть показанными на экране как окна с ошибками (сообщениями).

## Синтаксис

**ОткрытьЖурналСообщений** ( )

## Параметры

Не указываются.

## Возвращает

Ничего не возвращает.

## Комментарий

Сообщения будут попадать в журнал до тех пор, пока вы не закроете это окно вручную, либо не вызовете функцию **ОстановитьЖурналСообщений**.

## См. также:

- [ОткрытьЖурналСообщений](#)

- [ВЖурналСообщений](#)
- [ЗапуститьЖурналСообщений](#)
- [ОстановитьЖурналСообщений](#)
- [ОчиститьЖурналСообщений](#)
- [НайтиЖурналСообщений](#)

## ОткрытЖурналСообщений

Проверяет, открыто ли в программе окно журнала с сообщениями.

### Синтаксис

**ОткрытЖурналСообщений** ( )

### Параметры

Не указываются.

### Возвращает

**Да** (логическое значение) – окно с журналом уже открыто в программе.

**Нет** (логическое значение) – окно с журналом не открыто в программе.

### См. также:

- [ОткрытьЖурналСообщений](#)
- [ВЖурналСообщений](#)
- [ЗапуститьЖурналСообщений](#)
- [ОстановитьЖурналСообщений](#)
- [ОчиститьЖурналСообщений](#)
- [НайтиЖурналСообщений](#)



## ЗапуститьЖурналСообщений

Разрешает вывод сообщений в журнал сообщений. Действие функции равносильно установке флага в поле «Перехватывать сообщения» в окне журнала сообщений в программе.

### Синтаксис

**ЗапуститьЖурналСообщений ( )**

### Параметры

Не указываются.

### Возвращает

Ничего не возвращает.

### Комментарий

Сообщения будут попадать в журнал до тех пор, пока вы не снимите флажок в поле «Перехватывать сообщения», либо не вызовете функцию **ОстановитьЖурналСообщений**.

### См. также:

- [ВЖурналСообщений](#)
- [ОткрытьЖурналСообщений](#)
- [ОстановитьЖурналСообщений](#)
- [ОчиститьЖурналСообщений](#)
- [НайтиЖурналСообщений](#)

## ОстановитьЖурналСообщений

Запрещает вывод сообщений в журнал сообщений. Действие функции равносильно снятию флага в поле «Перехватывать сообщения» в окне журнала сообщений в программе.

### Синтаксис

**ОстановитьЖурналСообщений ( )**

### Параметры

Не указываются.

### **Возвращает**

Ничего не возвращает.

### **См. также:**

- [ВЖурналСообщений](#)
- [ОткрытьЖурналСообщений](#)
- [ЗапуститьЖурналСообщений](#)
- [ОчиститьЖурналСообщений](#)
- [НайтиЖурналСообщений](#)

## **ОчиститьЖурналСообщений**

Удаляет записи из журнала сообщений.

### **Синтаксис**

**ОчиститьЖурналСообщений** ( )

### **Параметры**

Не указываются.

### **Возвращает**

Да (логическое значение) – записи удалены в журнале сообщений.

Нет (логическое значение) – записи не удалены из журнала сообщений.

### **См. также:**

- [ВЖурналСообщений](#)
- [ОткрытьЖурналСообщений](#)
- [ЗапуститьЖурналСообщений](#)
- [ОстановитьЖурналСообщений](#)
- [НайтиЖурналСообщений](#)