**Генератор отчетов GenRep**

[Аггрегатные функции 2](#_Toc185825641)

[Имя поля или выражение 2](#_Toc185825642)

[Формат отображения 3](#_Toc185825643)

[Зарезервированные слова 6](#_Toc185825644)

[Структура файла шаблона 6](#_Toc185825645)

[Не печатаемые управляющие блоки 6](#_Toc185825646)

[Важные замечания 7](#_Toc185825647)

[Печатаемые блоки-выражения 8](#_Toc185825648)

[Создание и изменение шаблонов отчетов 8](#_Toc185825649)

[Общие положения 9](#_Toc185825650)

# Аггрегатные функции

Большинство справок и отчетов обязаны содержать итоговые результаты. Для подсчета итогов в данном генераторе предусмотрены следующие агрегатные функции

|  |  |  |
| --- | --- | --- |
| **Функция** | **Пример** | **Описание** |
| **CNT** | **[CNT:Amount]** | Количество не повторяющихся значений в данном поле или выражении. В примере подсчитывается количество не повторяющихся значений в поле **Amount**.  Для пояснения того, что имеется в виду под словами "не повторяющиеся значения" предположим, что в поле **Amount** в порядке следования есть следующие значения: 1,1,1,2,2,3,1,1. В результате использования функции **CNT** мы получим значение 4. Т.е. значение счетчика увеличивается на 1 только в тот момент, когда последующее значение отлично от предыдущего. Эту функцию удобно использовать для подсчета количества группировок по какому-либо полю. |
| **CNTR** | **[CNTR:Amount]** | Количество записей. Т.е. общее количество значений в поле **Amount** вне зависимости от уникальности этих значений. |
| **SUM** | **[SUM:Platezh-Nalog]** | Сумма по данному полю или выражению. В примере производится суммирование значений выражения **(Platezh-Nalog)**, т.е. "чистый" платеж без налога. |

Следует иметь в виду, что вне зависимости, от типа данных, использованных в указанном поле или выражении по которым производится расчет, итоговое выражение будет иметь тип Double для того, чтобы с запасом "перекрыть" все возможные значения, которые могут появиться в результате суммирования. Это значит, что у итога появиться дробная часть, даже если Вы складывали целочисленные значения.

Для того, чтобы "подавить" не нужное отображение дробной части следует использовать опцию "Формата" в выражении, например:

**[%.0f:CNTR:Amount]**

Разумеется, можно использовать и формат для отображения целочисленных значений

**[%i:CNTR:Amount]**

Но в этом случае Вы должны отдавать себе отчет, что может произойти переполнение данного типа, т.е. в результате подсчета общее количество может оказаться больше чем поддерживает данный целочисленный формат.

Для подсчета количества записей можно использовать и функцию **SUM** в следующем виде **[SUM:1]**. Или при суммировании по условию, например, подсчитать количество записей, где значение поля **Amount**>10 будет выглядеть так **[SUM:IF(Amount>10;1;0)]**

# Имя поля или выражение

В качестве отображаемого значения, наряду с собственно именем поля, можно выводить некоторую функцию. Результатом этой функции обязательно должно быть число. Выражение функции может включать в себя имена полей, знаки операций, скобки и конструкцию **IF**

|  |  |  |
| --- | --- | --- |
| **Конструкция** | **Пример** | **Описание** |
| **+** | **[Amount+10]** | Сложение |
| **-** | **[Amount-10]** | Вычитание |
| **/** | **[Amount/10]** | Деление |
| **\*** | **[Amount\*10]** | Умножение |
| **%** | **[Amount%3]** | Целочисленный остаток от деления. Если в данном примере **Amount**=8, то 8%3=2.  **Замечание** - данный символ работает только с целой частью числа, т.е. если написать выражение 5.2%2.6, то это будет интерпретировано как 5%2=1 |
| **$** | **[Platezh$1]** | Округление до указанного количество знаков после запятой. Если в данном примере **Platezh**=5.26, то 5.26$1=5.30 Если указанное количество знаков до которого следует округлить имеет отрицательное значение, то предполагается, что округлять следует до целой части. |
| **#** | **[Platezh#2]** | Деление с округлением результата до целого в большую сторону. Если в данном примере **Platezh**=4.01, то 4.01#2=3.00 |
| **|** | **[Platezh|2]** | Деление с округлением результата до целого в меньшую сторону. Если в данном примере **Platezh**=4.99, то 4.99|2=2.00 |
| **( )** | **[Platezh/(Platezh+Nalog)\*100]** | Скобки позволяют объединять группы выражений |
| **IF**(Усл**;**Выр**;**Выр) | **[IF(Platezh>0;Nalog/Platezh;0)]** | Условный оператор. Если указанное выражение принимает значение "Истина", то выводится первое выражение, если "Ложь" - второе. Условный оператор должен возвращать только числовые значения. К сожалению, его нельзя использовать для того, чтобы печатать разный текст в зависимости от некоторого условия. |

Следует обратить особое внимание на то, что использование внутри квадратных скобок пробелов недопустимо. Это приведет к неверному распознованию выражения и как следствие - отсутствию результата.

Условный оператор, как впрочем и любое выражение, можно использовать вместе с функциями аггрегирования, например, подсчитать только те записи, где значение поля **Amount**>10

**[SUM:IF(Amount>10;1;0)]**

Разумеется, можно использовать и формат для отображения только целой части

**[%.0f:SUM:IF(Amount>10;1;0)]**

Обратите внимание, что даже если используется функция, которая возвращает целую часть, то это никак не влияет на формат вывода результата. Например, 4.01#2=3.00. Т.е. для отображения результат в нужном формате необходимо использовать опции формата.

В качестве имени поля могут выступать также [зарезервированные слова](#topic_Tamplate.Rezerv)

# Формат отображения

Данные можно отображать не только "как есть", но и "как надо", для чего используются специальные форматы отображения. В данном генераторе отчетов можно использовать форматы отображения языка программирования "Си", формат встроенного языка программирования "СП Бухгалтерия", а также ряд дополнительных форматов, разработанных специально для данного генератора.

**Внимание** - при вводе формата следует **учитывать регистр букв**. Т.е. имеет принципиальное значение большие или маленькие буквы. Это единственное место в данном генераторе отчетов, где учитывается регистр букв. В остальных местах регистр букв значения не имеет.

В самом общем виде формат отображения также имеет свой формат:

**%**[flags] [width] [.precision] [{**h** | **l** | **I64** | **L**}]type

Т.е. если использовать формат отображения, то обязательным для использования является смвол "%" и единственная буква, отображающая тип данных (type), например %f Все другие части контролируют другие аспекты формата отображения, такие как:

flags - символ или набор символов, которые управляют выравниванием, отображением знака, фактом отображения ведущих нулей. Допустимо использовать комбинацию флагов

width - минимальное количество отображаемых символов.

.precision - максимальное количество отображаемых символов всего или части значения, или минимальное значение для целочисленных типов данных

{**h** | **l** | **I64** | **L**} - это опции определяющие максимально допустимые размерности для некоторых типов полей

type - собственно буква, обозначающая тип выходных данных. Единственный обязательный параметр из всех перечисленных

Форматы отображения языка программирования "Си" представлены на дополнительном файле **Format.rtf**. Здесь будет рассмотрена только наиболее общеупотребительная часть значений указанных флагов этого языка

Некоторые значения опции **type** и что именно у них ограничивает опция **.precision**

|  |  |  |
| --- | --- | --- |
| **Обозначение** | **Тип** | **Влияние .precision**(т.е. число после точки в выражении формата) |
| **d** | Целое | Минимальное количество знаков. Недостающее добавляется слева нулями. Не ограничивает число, если количество знаков больше указанного |
| **X** | Целое в 16-ричном формате | Минимальное количество знаков. Недостающее добавляется слева нулями. Не ограничивает число, если количество знаков больше указанного |
| **f** | Дробное | Максимальное количество знаков после запятой. Недостающее добавляется справа нулями. Если количество знаков после запятой больше указанного, то лишнее отбрасывается. |
| **s** | Символьная | Максимальное количество печатаемых знаков. Если количество знаков в символьной строке больше указанного, то лишнее отбрасывается. |

Некоторые значения опции **Flags**

|  |  |
| --- | --- |
| **Значение** | **Описание** |
| **-** | Выравнивание значения по левому краю. По умолчанию (т.е. если знак "минус" не указан) выравнивание происходит по правому краю |
| **+** | Всегда отображать знак числового значения. По умолчанию (т.е. если знак "плюс" не указан) знак отображается только у отрицательных чисел |
| **0** | Если количество знаков в числе меньше, чем указано в опции width, то недостающее количество знаков отображается в виде ведущих нулей. По умолчанию (т.е. если знак "ноль" не указан), то недостающее количество символов ничем не дополняется. Данный символ игнорируется в комбинации со знаком "минус" (т.е. при выравнивании по левому краю) и при отображении целочисленных типов данных |
| Пробел | В данном генераторе отчетов наличие пробела указывает на то, что нулевые значения печатать не надо. |

Примеры использования форматов данных в данном генераторе отчета. Следует заметить, что в примерах использованы ряд специфических форматов разработанных специально для данного генератора отчетов. Такие форматы помечены символом "\*"

|  |  |  |
| --- | --- | --- |
| **Пример** |  | **Описание** |
| **Число с дробной частью.**  По умолчанию, такие числа отображаются с 2 знаками после запятой, при этом нулевые значения вообще не отображаются. | | |
| **[%.2f:Platezh]** |  | Отобразить только 2 знака после запятой не ограничивая количество знаков в целой части. При этом нулевые значения обязательно отображать. |
| **[% .2f:Platezh]** | \* | В отличии от предыдущего варианта, между символом "%" и точкой стоит пробел. Отобразить только 2 знака после запятой не ограничивая количество знаков в целой части. При этом нулевые значения не отображать. |
| **[%.0f:Platezh]** |  | Отобразить только целую часть числа (ноль знаков после запятой) не ограничивая количество знаков в целой части. При этом нулевые значения обязательно отображать. |
| **[%.2f руб:Platezh]** |  | Отобразить только 2 знака после запятой не ограничивая количество знаков в целой части. При этом нулевые значения обязательно отображать, а после числа добавить слово " руб". В приципе, того же самого результата можно добиться используя следующий формат вывода **[%.2f:Platezh] руб** |
| **[%5.2f:Platezh]** |  | Отобразить только 2 знака после запятой при этом общее количество отображаемых знаков (включая запятую) не должно быть меньше 5. Т.е. значения больше 99.99 благополучно отобразятся, а если количество знаков меньше 5, то недостающее будет дополнено ведущими пробелами (слева). Целую часть в значении формата можно использовать для выравнивания числовых данных при шрифте Courier. Например, формат **[%20.2f:Platezh]** означает, что под число отводится минимум 20 символов, причем недостающее будет дополнено пробелами слева, т.е. выравнивание по правому краю. |
| **[%05.2f:Platezh]** |  | Отобразить только 2 знака после запятой при этом общее количество отображаемых знаков (включая запятую) не должно быть меньше 5. Т.е. значения больше 99.99 благополучно отобразятся, а если количество знаков меньше 5, то недостающее будет дополнено не пробелами, как в предыдущем случае, а ведущими нулями. Напимер, число 1.2 будет отображено как 01.20 |
| **[%-5.2f:Platezh]** |  | Отобразить только 2 знака после запятой при этом общее количество отображаемых знаков (включая запятую) не должно быть меньше 5. Т.е. значения больше 99.99 благополучно отобразятся, а если количество знаков меньше 5, то недостающее будет дополнено пробелами справа. |
| **[%SP:Platezh]** | \* | Отобразить указанную сумму прописью (словами) на русском языке. Т.е. вместо числа 5.25 будет выведено "Пять рублей 25 копеек" |
| **[%SPU:Platezh]** | \* | Отобразить указанную сумму прописью (словами) на украинском языке. Т.е. вместо числа 5.25 будет выведено "П'ять грiвней 25 копiйок" |
| **Целое число.**  По умолчанию, такие числа отображаются "как есть". Т.е. как целое число. Однако при использовании в выражениях результат выражения всегда отображается как **Double** | | |
| **[%d:Amount]** |  | Отобразить "как есть". Т.е. как целое число |
| **[%+d:Amount]** |  | Отобразить знак перед числом, даже если число положительное. Например, при значении **Amount**=10 будет отображено **+10** |
| **[%.10d:Amount]** |  | Если количество знаков в числе меньше 10, то недостающее дополнить слева нулями |
| **[%010d:Amount]** |  | Если количество знаков в числе меньше 10, то недостающее дополнить слева нулями. Абсолютный аналог предыдущей записи, только формат записан немного по другому |
| **[%XH:Amount]** |  | Отобразить значение **Amount** в 16-ричной системе исчисления, дописав справа символ "Н". Например, при значении **Amount**=123 отобразится значение **7BH** |

|  |  |  |
| --- | --- | --- |
| **Символьное.**  По умолчанию, у символьных данных отбрасываются ведущие и концевые пробелы. | | |
| **[%s:Text]** |  | Отобразить текстовую строку "как есть" с ведущими и концевыми пробелами (если есть) |
| **[%.14s:Text]** |  | Отобразить только первые 14 символов текстовой строки, остальные отбросить |
| **[%60.14s:Text]** |  | Отобразить только первые 14 символов текстовой строки, а остальные отбросить. После чего дополнить полученный результат до 60 символов пробелами слева (т.е. сделать "выравнивание" по правому краю) |
| **Дата и время**  По умолчанию, поля типа Date отображаются в формате ДД.ММ.ГГГГ. т.е. например 31.12.2001. К отображению даты и времени относятся также некоторые [зарезервированные слова.](#topic_Tamplate.Rezerv) | | |
| **[%dt:Date]** | \* | Отобразить дату в формате ДД.ММ.ГГГГ т.е. например 31.12.2001 |
| **[%dt2:Date]** | \* | Отобразить дату в формате ДД.ММ.ГГ т.е. например 31.12.01 |
| **[%dts:Date]** | \* | Отобразить дату в виде строки, т.е. например "31 декабря 2001 г" |
| **[%tm:Amount]** | \* | Исходя из предположение, что имеющееся целочисленное значение - это количество минут отобразить это число как время. Например, при значении **Amount**=345 отобразится **05:45** |
| **Логическое**  По умолчанию, если логическое поле имеет значени "Истина", то оно отображается в виде символа звездочки "\*", если "Ложь", то никак не отображается | | |
| **[%lg1:Flag]** | \* | "Истина" отображается как слово "Да", а "Ложь" как слово "Нет" |
| **[%lg2:Flag]** | \* | "Истина" отображается как символ "+", а "Ложь" как символ "-" |

# Зарезервированные слова

В данном генераторе отчетов в качестве значения отображаемого поля может быть указаны некторые зарезервированные слова. В связи с этим нежелательно, чтобы имена полей совпадали с этими зарезервированными словами. Зарезервированными считается имена начинающиеся и кончающиеся символом подчеркивания.

|  |  |
| --- | --- |
| **Слово** | **Назначение** |
| **[\_page\_]** | Вставка разрыва страницы |
| **[\_date\_]** | Вставка текущей даты в виде 31.12.01 |
| **[\_time\_]** | Вставка текущего времени в виде 01:45 |

# Структура файла шаблона

Файл-шаблон - это обычный файл формата **RTF**, но со встроенными управляющими блоками. Эти управляющие блоки заключены в квадратные скобки. Управляющие блоки делятся на 2 вида:

[Не печатаемые управляющие блоки](#topic_Tamplate.Stru_TamplateNoPrint)

[Печатаемые блоки-выражения](#topic_Tamplate.Stru_TamplatePrint)

## Не печатаемые управляющие блоки

Управляющие блоки предназначены для того, чтобы файл отчета "понял" откуда собственно следует брать данные и как их группировать.

Управляющие блоки обязательны должны быть парными. Т.е. должен быть открывающий управляющий блок и закрывающий управляющий блок. Открывающий управляющий блок начинается с символа "меньше" (<), а закрывающий управляющий блок должен завершаться символом "больше" (>). Для удобства распознования, рекомендуется распологать управляющие блоки в отдельных строках отчета. Т.е. в общем случае это выглядит так:

Тело отчета

**[<**НаименованиеБлока**]** - открытие управляющего блока

Тело отчета

**[**НаименованиеБлока**>]** - закрытие управляющего блока

Тело отчета

Всего предусмотрено 3 типа управляющих блоков:

* Имя файла **DBF** из которого собственно следует брать информацию
* Имя поля таблицы **DBF**, по содержимому которого следует группировать данные в отчете
* Специальное ключевое слово **Detal**, являющееся признаком того, что далее следует организовать перебор записей

Таким образом, в самом общем виде отчет выглядит следующим образом

Текст до того, как информация будет взята из какой-либо таблицы (заголовок отчета)

**[&ltTable1.dbf]**

Заголовочная часть при открытии таблицы Table1.dbf

**[&ltFieldGroup]**

Заголовочная часть при открытии группы по полю FieldGroup таблицы Table1.dbf

**[&ltFldSubGrp]**

Заголовочная часть при открытии группы по полю FldSubGrp таблицы Table1.dbf

**[&ltDetal]**

собственно отображение содержимого таблицы Table1.dbf

**[Detal>]**

Заключительная часть при закрытии группы по полю FldSubGrp таблицы Table1.dbf. Отображение итогов по данной группе

**[FldSubGrp>]**

Заключительная часть при закрытии группы по полю FieldGroup таблицы Table1.dbf. Отображение итогов по данной группе

**[FieldGroup>]**

Заключительная часть при закрытии таблицы Table1.dbf. Отображение итогов по таблице Table1.dbf

**[Table1.dbf>]**

Текст после того, как информация будет взята из таблицы Table1.dbf, но до того, как информация будет взята из таблицы Table2.dbf

**[&ltTable2.dbf]**

Заголовочная часть при открытии таблицы Table2.dbf

...

Заключительная часть при закрытии таблицы Table2.dbf. Отображение итогов по таблице Table2.dbf

**[Table2.dbf>]**

Текст после того, как информация будет взята из какой-либо таблицы (подвал отчета)

## Важные замечания

* **Недопустимо делать "пересечение" блоков** - т.е. и "открытие" и "закрытие" блока должны происходить только внутри таких же "скобок" другого блока. Блоки обязательно должны быть вложены друг в друга как матрешки.
* **Недопустимо открыть внутри уже открытого табличного блока другой табличный блок** - т.е. внутри информации из одной таблицы нельзя вывести информацию из другой. Только по очереди.
* В блоках описания отчета может отсутствовать блок детализации ("**Detal**"). В этом случае отчет все равно пройдет по всем записям указанной таблицы и распечатает исключительно заключительную часть при закрытии таблицы. Т.е. в нем можно указать итоговую часть отчета.
* Если блок отчета содержит блоки группировок, то блок детализации ("**Detal**") обязательно должен присутствовать.

Управляющие блоки рекомендуется распологать на отдельных строках исключительно для удобства. Однако в случае необходимости их можно расположить и в одну строку, например:

**[&ltTable1.dbf]** Итого по таблице Table1.dbf **[Table1.dbf>]** дополнительный текст **[&ltTable2.dbf]** Итого по таблице Table2.dbf **[Table2.dbf>]**

## Печатаемые блоки-выражения

Печатаемые блоки-выражения могут использоваться только внутри блока открытой таблицы. Вне блока таблицы они не имеют смысла. Печатаемые блоки также заключены в квадратные скобки как и не печатаемые управляющие блоки, но они не парные. Печатаемые блоки могут состоять из 3 частей:

[Формат отображения](#topic_Tamplate.Format)

[Аггрегатная функция](#topic_Tamplate.Calculate)

[Собственно отображаемое поле или выражение](#topic_Tamplate.Expration)

При этом [Формат отображения обязательно предваряется символом процента (%), а завершается двоеточием, а](#topic_Tamplate.Format) [Аггрегатная функция также обязательно завершается двоеточием. Таким образом, в самом общем виде печатаемый блок имеет следующий формат:](#topic_Tamplate.Calculate)

**[%**[Формат отображения](#topic_Tamplate.Format)**:**[Аггрегатная функция](#topic_Tamplate.Calculate)**:**[Имя поля или выражение](#topic_Tamplate.Expration)**]**

Здесь обязательным является только [Имя поля или выражение, а](#topic_Tamplate.Expration) [Формат отображения и](#topic_Tamplate.Format) [Аггрегатная функция можно и не указывать. Они используются только по мере необходимости.](#topic_Tamplate.Calculate)

Если не задан [Формат отображения, то используется некий формат по-умолчанию](#topic_Tamplate.Format)

[Аггрегатная функция используются для подсчета итогов. Т.е. это функции типа "подсчитать сумму по полю", "подсчитать количество записей". Следует иметь в виду, что в результате использования аггрегатных функций итоговое значение принимает тип](#topic_Tamplate.Calculate) Double, т.е. как правило необходимо использовать [Формат отображения в паре с](#topic_Tamplate.Format) [Аггрегатными функциями](#topic_Tamplate.Calculate)

[Выражение, которое может использоваться вместо имени поля обязательно должно выдавать числовой результат. С другими типами выражений данный генератор не работает. Это не относится собственно к полям таблицы. Поля могут иметь любой допустимый тип данных. Кроме того, в качестве имен полей или выражений могут выступать](#topic_Tamplate.Expration) [зарезервированные слова](#topic_Tamplate.Rezerv)

# Создание и изменение шаблонов отчетов

Создание справок и отчетов - это одна из важнейших задач любого программного продукта (в особенности на территории бывшего СССР). Но еще более важным является возможность самому пользователю внести изменения как в итоговый отчет, так и в шаблон, на основе которого данный отчет создается.

На сегодняшний день, наиболее распространенным на территории бывшего СССР является текстовый редактор компании Microsoft с именем **Word**. Однако используя механизмы генерации отчетов, которые предоставляет этот продукт время формирования отчета катастрофически затягивается и подготовка простого отчета занимает минуты, а более сложного и вовсе уж неприлично много времени.

Для снятия противоречия между удобством просмотра и изменения отчета в **Word** и скоростью его создания многие программисты пошли по пути прямой генерации файлов отчета в формате **RTF**. Этот формат файлов удобен тем, что он прекрасно распознается, читается и редактируется текстовым редактором **Word**, сохраняя многие изменения производимые в **Word**, однако имеет более простую и открытую структуру по сравнению с документами самого **Word** формата **DOC**.

По этому же пути, пошел и автор данного генератора отчетов Александр Куденцов.

Если у Вас появится не работающий шаблон – просьба прислать его по электронной почте в виде SAB и соответствующие ему выходные файлы SP или DBF на Е-mail: [akudenzov@radon.ru Куденцову Александру.](mailto:akudenzov@radon.ru)

Более подробно ознакомиться со всеми возможностями данного генератора и нововведениях Вы можете на сайте <http://alkud.boom.ru/>

В данном описании рассматривается частный случай использования данного генератора отчетов при создании отчетов на основе содержимого DBF-таблиц в свете собственно форматирования файлов шаблона.

# Общие положения

Для просмотра готовых файлов отчета достаточно использовать специальную программу "просмотрщик" WordView.exe если он установлен на компьютере. В отличии от WinWord.exe эта программ не требует предварительной регистрации.

Однако хотя собственно создание итогового отчета и не требует непосредственно наличия программы WinWord.exe на компьютере пользователя, но тем не менее создание и внесение изменений в файл шаблона удобнее производить именно в ней.

Генерация отчета производится на основе

Предварительно созданного файла формата **RTF**, но с расширением **SAB** вместо **RTF**

Предварительно созданного файла **DBF** (или нескольких файлов **DBF**)

Следует отметить, что на момент генерации итогового отчета используемые файлы **DBF** не должны использоваться другими приложениями. Впрочем, как и сам файл шаблона **SAB**