

## Методика оперативного проведения и управляемые блокировки в 1С:Предприятие 8.3

### Введение

Несмотря на то, что у меня уже есть статья, посвященная механизму оперативного проведения, очень часто... нет не так, очень-очень-очень часто спрашивают, что это такое.

Я подумал и решил написать ну очень-очень-очень подробную статью про методику оперативного проведения и управляемые блокировки. А без управляемых блокировок, собственно, методику использовать бессмысленно.

При написании статьи, демопример я вводил на платформе 8.3.3.641. Всю задачу создавал на пустой базе.

### Постановка задачи

Необходимо автоматизировать 2 операции: покупка, продажа товаров. При продаже товаров необходимо контролировать наличие товаров в остатках предприятия и рассчитывать себестоимость списания товаров по методу *FIFO*.

### Начнем...

Структура метаданных:

Справочник: Товары

Документы: Приходная и Расходная с табличными частями "Список товаров" (товар, количество, сумма).

Регистры: ОстаткиТоваров, в регистре одно измерение: Товар, и один ресурс: Количество.

Назначение регистра – быстрое принятие решения, можно ли проводить документ.

СтоимостьТоваров, в регистре два измерения: Товар, Партия, и два ресурса: Количество, Стоимость. Назначение регистра: хранение информации о стоимости остатков товаров в разрезе партий.

Описывать проведение документа "Приходная" не буду, там все очевидно, все можно конструктором сделать.

В расходной опишем проведение по регистру ОстаткиТоваров...

## Методика оперативного проведения

Под термином "Методика оперативного проведения" я понимаю следующее: при проведении документа, вместо того, чтобы, сформировать запрос и понять хватает ли товаров или нет, мы сначала сформируем движения, а затем уже проверим, не ушли ли мы в минус? Подробнее с плюсами и минусами этой технологии можно познакомиться в моей [прошлой статье](#).

Опишем формирование движений в документе "Расходная":

Процедура ОбработкаПроведения(Отказ, Режим)

```
//1
Движения.ОстаткиТоваров.Записывать = Истина;

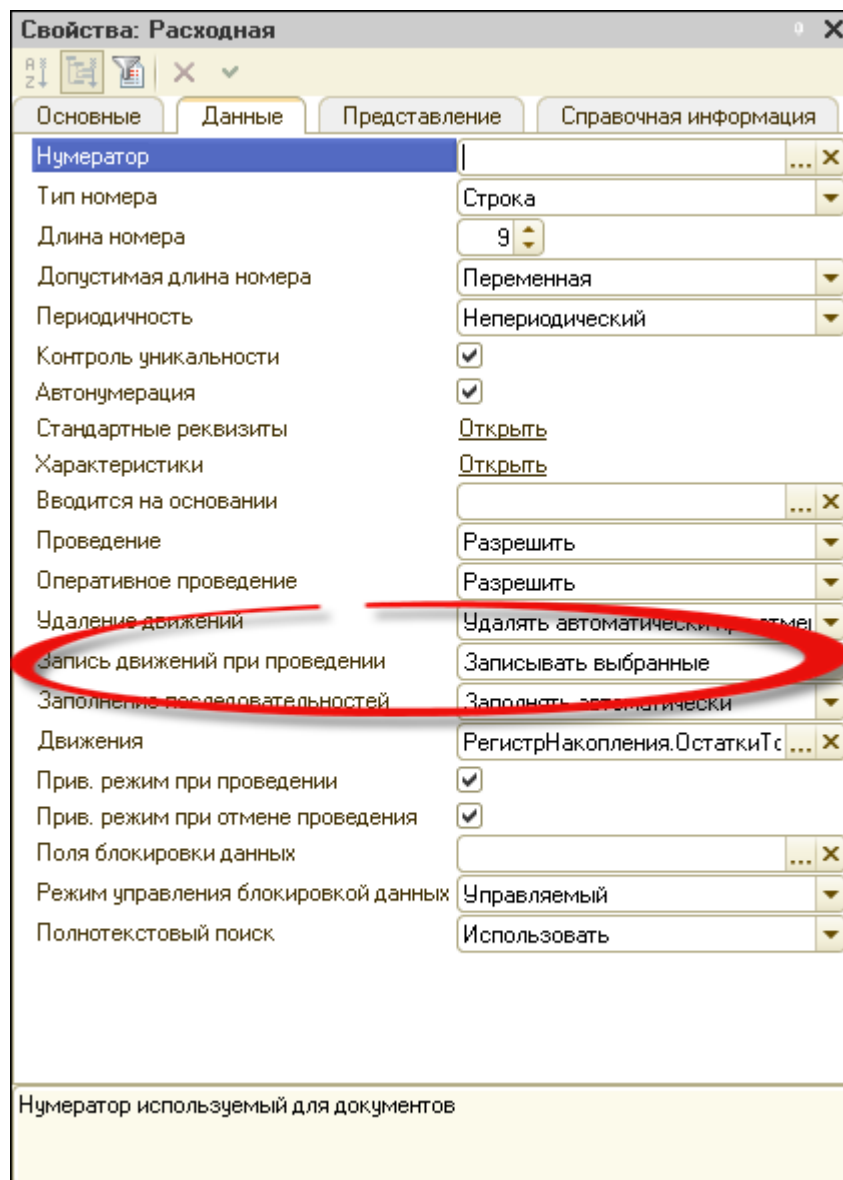
//2
Движения.ОстаткиТоваров.Очистить();

Для каждого Стр Из СписокТоваров Цикл
    Движение = Движения.ОстаткиТоваров.ДобавитьРасход();
    Движение.Период = Дата;
    Движение.Товар = Стр.Товар;
    Движение.Количество = Стр.Количество;
КонецЦикла;

//3
КонецПроцедуры
```

Прокомментируем текст модуля:

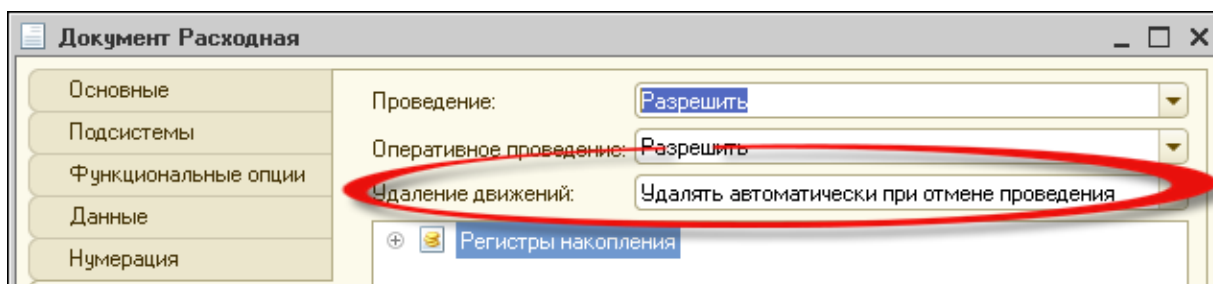
1. Устанавливаем маркер необходимости записи движений. Это необходимо делать в том случае, если у документа установлено свойство "Записывать выбранные".



Что дает нам это свойство?

Интересную фишку, раньше мы могли принудительно записать движения документа (наборы данных), но при окончании транзакции проведения наборы данных записывались еще раз. Теперь мы вольны в выборе, что и когда нужно записывать. Маркер записи автоматически снимается при записи набора, тем самым, при окончании транзакции проведения, набор записей еще раз записываться не будет. К чему это приведет? Правильно, к уменьшению блокировок таблиц и к сокращению времени транзакций при проведении.

2. Очищаем движения. Зачем? Все дело в еще одном новом свойстве документов "Удалять движения".



При установленном свойстве "Удалять автоматически" в самом начале транзакции проведения система автоматически записывает пустые наборы записей в регистры, тем самым очищая старые движения. Это физически происходит запись со всеми вытекающими транзакциями и блокировками.

Для облегчения нагрузки на таблицы баз данных теперь есть возможность не очищать автоматом старые движения...

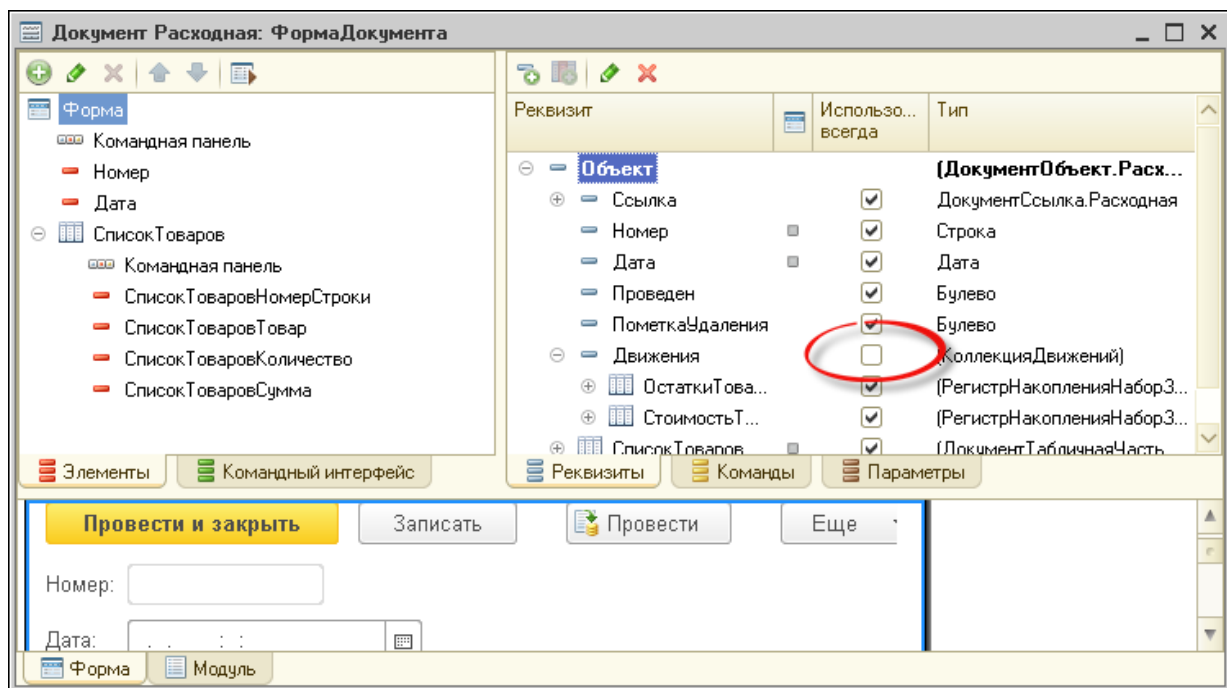
И тут обычно я слышу: "Так и всегда можно было "Не очищать автоматически"!"

Ага, можно было, но в каждом документе приходилось описывать очистку каждого набора записей при отмене проведения... Удобно...

Короче, 1С пошла на поводу у здравого смысла и сделала новое свойство "Удалять автоматически при отмене проведения". Это гарантирует нам, что все движения документа будут очищены в том случае, если у документа отработает событие "ОтменаПроведения".

Теперь вопрос – а зачем мы в модуле написали "Очистить()"?

Тут все дело в том, что теперь у нас движения автоматом не очищаются... НО! При работе с управляемыми формами копия объекта БД может не загрузить старые движения, к примеру, зависит это и от свойства данных формы "Использовать всегда".



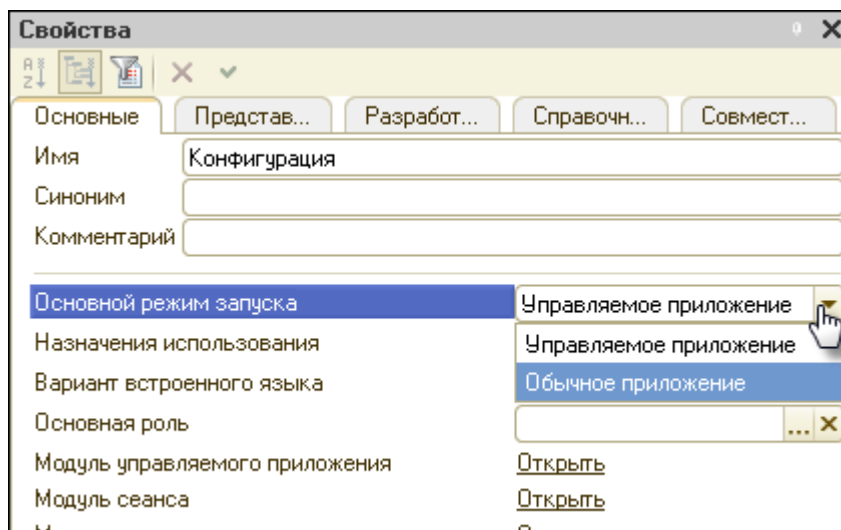
К примеру, в зависимости от этого свойства, движения документа будут прочитаны (если галка стоит) при открытии формы или нет. А если даже галка не установлена, и в форме отображаются движения, то движения будут прочитаны при открытии формы.

В обычных формах движения будут однозначно прочитаны при открытии формы.

А если движения прочитаны, то наборы записей в свойстве документа "Движения" не пустые, и добавление при проведении новых движений, как правило, приводит к дублям движений.

Так вот, чтобы не зависеть от обстоятельств, мы на всякий случай удаляем то, что возможно может находиться в коллекции движений по регистру "ОстаткиТоваров". Обращаю внимание, что запись данных в базу тут не производится, просто в памяти очищается набор записей.

Кстати, если свойство конфигурации "Основной режим запуска" будет установлено в значение "Обычное приложение", то такую строку "Движения.ИмяРегистра.Очистить()" будет писать и сам конструктор формирования движений.



3. Итак, движения сформировали, самое время их записать. Тут есть два варианта: либо "Движения.Записать()", либо "Движения.ОстаткиТоваров.Записать()". В чем разница и что выбрать?

Начнем с последнего: "Движения.ОстаткиТоваров.Записать()"; Этот способ безусловно запишет данные в регистр накопления. Но при этом флаг "Записывать" у набора записей снят не будет.

Но это ерунда, главное тут то, что при большом количестве наборов записей у документа нам придется самостоятельно контролировать, что в каком порядке в базу пишется, это может (да что там "может", точно скажется) негативно сказаться на проблеме взаимных блокировок (DeadLock), когда одна транзакция заблокирует таблицу "А" и будет ждать освобождения таблицы "Б", а другая транзакция будет вести себя строго наоборот.

Теперь посмотрим, как работает метод "Движения.Записать()"; Метод записывает только те движения документа, у которых установлен флаг "Записывать", при этом флаг в итоге снимается, что не приводит к повторной записи движений по окончании транзакции проведения.

И главное, "Движения.Записать()" всегда записывают движения в том порядке, в котором таблицы указаны в дереве метаданных, что на порядок уменьшает шансы взаимных блокировок, ведь все транзакции в одинаковом порядке блокируют таблицы.

Теперь, надеюсь, очевидно, выбираем метод "Движения.Записать()";

Запишем движения и проверим остатки в регистре.

//3

Движения.Записать ();

Запрос = Новый Запрос;

//4

Запрос.МенеджерВременныхТаблиц = Новый МенеджерВременныхТаблиц;

```
Запрос.Текст = "ВЫБРАТЬ
| Док.Товар КАК Товар,
| СУММА (Док.Количество) КАК Количество
| ПОМЕСТИТЬ ДокТЧ
| ИЗ
| Документ.Расходная.СписокТоваров КАК Док
| ГДЕ
| Док.Ссылка = &Ссылка
|
| СГРУППИРОВАТЬ ПО
| Док.Товар
|
| ИНДЕКСИРОВАТЬ ПО
| Товар
| ;
|
```

////////////////////////////////////  
////

```
| ВЫБРАТЬ
| Остатки.Товар.Представление КАК ТоварПредставление,
| Остатки.КоличествоОстаток
| ИЗ
| РегистрНакопления.ОстаткиТоваров.Остатки (
|     &ТочкаИтогов,
|     Товар В
|     (ВЫБРАТЬ
|         ДокТЧ.Товар
|     ИЗ
|         ДокТЧ КАК ДокТЧ) ) КАК Остатки
| ГДЕ
| Остатки.КоличествоОстаток < 0
| ;
|
```

////////////////////////////////////  
////

```
| ВЫБРАТЬ
| ДокТЧ.Товар
| ИЗ
| ДокТЧ КАК ДокТЧ";
```

Запрос.УстановитьПараметр ("Ссылка", Ссылка);

//5

Запрос.УстановитьПараметр ("ТочкаИтогов", Новый Граница (МоментВремени (), ВидГраницы.Включая) );

```
ПакетРезультатов = Запрос.ВыполнитьПакет();
РезультатЗапроса = ПакетРезультатов[1];

Если НЕ РезультатЗапроса.Пустой() Тогда
    //6
    Отказ = Истина;

    Выборка = РезультатЗапроса.Выбрать();
    Пока Выборка.Следующий() Цикл
        Сообщение = Новый СообщениеПользователю;
        Сообщение.Текст = "Мало товара " + Выборка.ТоварПредставление + " нужно
еще " + (-Выборка.КоличествоОстаток);
        Сообщение.Сообщить();
    КонецЦикла;
КонецЕсли;

Если Отказ Тогда
    Возврат;
КонецЕсли;
```

Комментарии к модулю:

4. Привязываем к запросу менеджер временных таблиц. Это позволит использовать временные таблицы позже для проведения по регистру с партиями товаров.

В запросе выбираем из табличной части документа товары и количество, группируем все и индексируем по полю Товар. Это, в общем случае, благоприятно скажется на быстродействии запроса.

5. "ТочкаИтогов", здесь мы используем объект "Граница". Так как передав в запрос просто "МоментВремени()", мы получили бы остатки на начало проведения документа, то есть не включили бы в расчет итогов только что сформированные движения документа. А нам нужно понять, что произошло с итогами после проведения документа.

В целом все.

Мы используем новую методику проведения документа. Сначала записали данные в регистр, а потом через кэш транзакции смотрим не ушли ли мы в минус. Если ушли – транзакцию отказываем.

Но это далеко не все. Назревает вопрос о "грязном чтении".



## Грязное чтение

Что это такое?

Я не буду вдаваться в теорию (в отличии от вас, вам я советую в теорию копнуть, лишним не будет), поясню на примере:

Проводим две расходных. И в одной и в другой продаем 5 ложек. А в остатках всего 6. Так получилось, что продавать ложки мы умудрились в один и тот-же момент времени.

Что произойдет?

Первая накладная сформирует движения и начнет читать запрос по остаткам через кэш собственной транзакции, что, собственно, будет делать и вторая накладная. Но ложек-то всего 6, а в сумме две накладные продадут 10... Потому что не знают о том, что данные в таблицах уже не актуальны...

Что делать?

Надо заблокировать данные от параллельного чтения.

В нашем примере это сделать очень просто. У набора записей есть свойство "БлокироватьДляИзменения". Это, как и свойство "Записывать", лишь маркер указывающий, что необходимо установить блокировку на те записи, которые были сформированы в наборе записей.

Когда устанавливать свойство "БлокироватьДляИзменения"? Не важно, главное до самой записи данных.

Когда произойдет блокировка записей? В момент записи данных.

Что означает эта блокировка?

Никто параллельно с нами читать данные из регистра по тем товарам, движения по которым были сформированы, не сможет.

Когда блокировка будет снята?

При окончании транзакции, в которой она началась, в нашем примере – при завершении проведения документа.

Установим блокировку:

```
//2
```

```
Движения.ОстаткиТоваров.Очистить();
```

```
Для каждого Стр Из СписокТоваров Цикл
```

```
Движение = Движения.ОстаткиТоваров.ДобавитьРасход();
```

```
Движение.Период = Дата;
```

```
Движение.Товар = Стр.Товар;
```

```
Движение.Количество = Стр.Количество;
```

```
КонецЦикла;
```

```
//7
```

```
Движения.ОстаткиТоваров.БлокироватьДляИзменения = Истина;
```

```
//3
```

```
Движения.Записать();
```

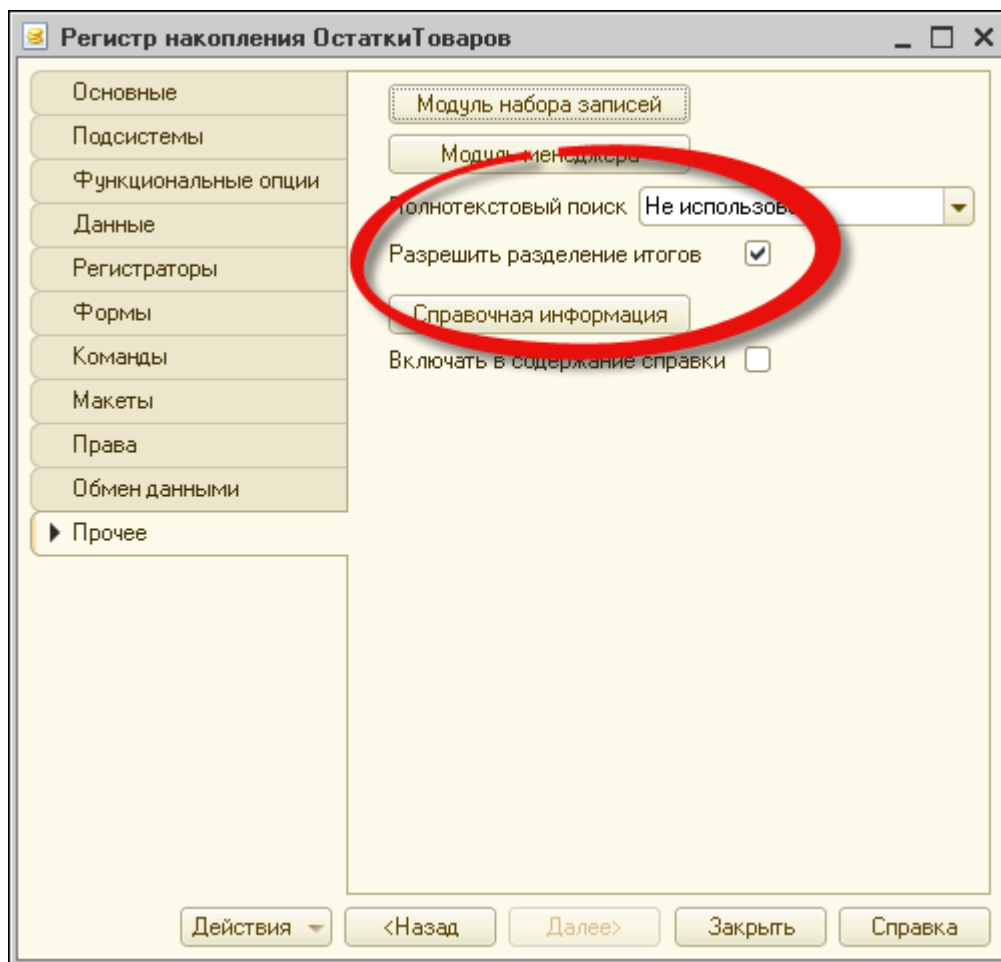
Казалось бы, все. А давайте проверим какие свойства нужно установить у объектов, для того, чтобы такая блокировка не просто работала, а еще и блокировала только те записи, по которым были сформированы движения...

**А.** Свойство конфигурации "Режим управления блокировкой данных в транзакции" – если, "Управляемый", то все хорошо.

Если "Автоматический", то системе глубоко фиолетово на то, что мы с вами написали. В нашем частном случае, блокировка в режиме "Автоматический" не будет установлена.

Если "и то, и другое", то важно проверить настройки объектов. Помните, что вид транзакции наследуется всеми вложенными транзакциями. То есть если документ записывается в автоматической транзакции, а движения делает по регистрам с управляемой... То система немного расстроится и вывалится с ошибкой.

**Б.** Так как блокируем мы не весь регистр, за что нам отдельное спасибо, а только те данные, по которым мы сформировали движения, за что спасибо ребятам из 1С, то важно установить свойство регистра "Разрешить разделение итогов".



Если разделение итогов не будет включено и в режиме пользователя, то в большей части случаев блокировка будет наложена на весь регистр с его таблицами.

Вот теперь точно с регистром "ОстаткиТоваров" все. Займемся себестоимостью.

## Партионное списание

При списании партий из регистра "СтоимостьТоваров" использовать методику оперативного проведения мы не можем.

Почему?

А для того, чтобы списать партии нужно узнать – какие конкретно, то есть нам нужно сначала прочитать данные из регистра, а потом уже формировать движения.

```
Если Отказ Тогда  
    Возврат;  
КонецЕсли;
```

```
//7
```

```
Запрос.Текст = "ВЫБРАТЬ  
    | ДокТЧ.Товар КАК Товар,  
    | ДокТЧ.Количество КАК Количество,  
    | СтоимостьТоваров.Партия,  
    | СтоимостьТоваров.КоличествоОстаток,  
    | СтоимостьТоваров.СтоимостьОстаток  
    | ИЗ  
    | ДокТЧ КАК ДокТЧ  
    | ЛЕВОЕ СОЕДИНЕНИЕ  
РегистрНакопления.СтоимостьТоваров.Остатки (  
    |     &ТочкаИтоговДляСебестоимости,  
    |     Товар В  
    |     (ВЫБРАТЬ  
    |     ДокТЧ.Товар  
    |     ИЗ  
    |     ДокТЧ КАК ДокТЧ) ) КАК СтоимостьТоваров  
    | ПО ДокТЧ.Товар = СтоимостьТоваров.Товар  
    |  
    | УПОРЯДОЧИТЬ ПО  
    |     СтоимостьТоваров.Партия.МоментВремени  
    | ИТОГИ  
    |     МИНИМУМ (Количество)  
    | ПО  
    |     Товар";
```

```
//8
```

```
Запрос.УстановитьПараметр ("ТочкаИтоговДляСебестоимости", МоментВремени ());
```

```
//9
```

```
Движения.СтоимостьТоваров.Очистить ();
```

```
//10
```

```
РезультатЗапроса = Запрос.Выполнить ();
```

```
ВыборкаТовар =
```

```
РезультатЗапроса.Выбрать (ОбходРезультатаЗапроса.ПоГруппировкам);
```

```
Пока ВыборкаТовар.Следующий () Цикл
```

```
    ОсталосьСписать = ВыборкаТовар.Количество;
```

```
    ВыборкаПартия = ВыборкаТовар.Выбрать ();
```

```
    Пока ВыборкаПартия.Следующий () И ОсталосьСписать <> 0 Цикл
```

```
        Списать = МИН(ОсталосьСписать, ВыборкаПартия.КоличествоОстаток);
```

```
        Движение = Движения.СтоимостьТоваров.ДобавитьРасход ();
```

```
        Движение.Период = Дата;
```

```
        Движение.Товар = ВыборкаПартия.Товар;
```

```
        Движение.Партия = ВыборкаПартия.Партия;
```

```
        Движение.Количество = Списать;
```

```
Движение.Стоимость = Списать / ВыборкаПартия.КоличествоОстаток *  
ВыборкаПартия.СтоимостьОстаток;
```

```
ОсталосьСписать = ОсталосьСписать - Списать;
```

```
КонецЦикла;
```

```
КонецЦикла;
```

```
//11
```

```
Движения.СтоимостьТоваров.Записывать = Истина;
```

Комментарии к модулю:

7. После проверки будем ли мы проводить документ, описываем новый текст запроса. Новый объект запрос создавать нет смысла, будем использовать уже существующий. Кроме всего, к нему уже подключен менеджер временных таблиц, в котором содержится проиндексированная таблица "ДокТЧ".

8. Устанавливаем "МоментВремени()", как точку расчета итогов таблицы остатков. Как было описано выше, по умолчанию эта точка (то бишь старые движения, которые мог бы сформировать этот документ) в расчет итогов не попадут. Но тут есть одна хитрость, к которой мы вернемся немного позже.

9. На всякий случай очистим движения документа, вдруг они были прочитаны.

10. Опишем списание по партиям.

11. Установим маркер необходимости записи данных по регистру "СтоимостьТоваров".

Когда транзакция проведения будет завершена, система самостоятельно запишет движения в базу.

Почти все, остались блокировки.

## Управляемые блокировки

По регистру "СтоимостьТоваров" воспользоваться флагом "БлокироватьДляИзменения" нам не удастся.

Почему?

Да потому, что блокировка с этим флагом происходит в момент записи данных набора записей, а нам блокировку нужно установить еще перед запросом, дабы не допустить параллельного чтения данных.

Поэтому, для установки блокировки, будем использовать объект "БлокировкаДанных".

Объект "БлокировкаДанных" представляет таблицу, каждая строка которой описывает, что и где нужно заблокировать.

```
Если Отказ Тогда
    Возврат;
КонецЕсли;

//12
Блокировка = Новый БлокировкаДанных;
//13
ЭлементБлокировки =
Блокировка.Добавить ("РегистрНакопления.СтоимостьТоваров");
ЭлементБлокировки.Режим = РежимБлокировкиДанных.Исключительный;
//14
ЭлементБлокировки.ИсточникДанных = ПакетРезультатов[2];
//15
ЭлементБлокировки.ИспользоватьИзИсточникаДанных ("Товар", "Товар");
Блокировка.Заблокировать ();

//7
Запрос.Текст = "ВЫБРАТЬ
```

Перед нашим вторым текстом запроса опишем блокировку.

12. Создаем объект.

13. Добавляем в объект новую строку с описанием блокируемой таблицы.

14. Для того чтобы не блокировать весь регистр, установим фильтр по товарам, данные о том по каким товарам необходимо заблокировать таблицы хранятся у нас в результате первого запроса, из пакета результатов достаем последний. Можно было и саму табличную часть документа привязать к источнику данных, но тогда система еще раз строила бы запрос к табличной части документа, а так мы все данные уже получили, сгруппировали.

15. Так как в результате запроса поле содержащее список с товарами может называться не так, как поле в регистре – описываем соответствие.

Установили блокировку. Теперь параллельно с нами никто не прочитает остатки по указанным товарам из регистра "СтоимостьТоваров".

Обращаю внимание, так как мы не знаем заранее какие партии будут выбраны запросом, то фильтр по ним установить мы не можем и блокируем все партии. Чем меньше фильтров описано, тем больше данных блокируется.

И вот тут казалось бы все, а нет.

## Проблема оперативного проведения

Если сейчас попробовать провести документ оперативно 2 раза, указывая последние товары в регистре, то система вылетит с ошибкой.

Почему?

При оперативном проведении, несмотря на то, что мы указали в параметре второго запроса "МоментВремени()" и он, вроде бы, не должен включить старые движения документа – система рассчитывает остатки на оперативную отметку времени, то есть старые движения будут включены в расчет итогов. Да, да... Вот такая "фишка".

Что делать?

Отследим оперативное проведение и очистим старые движения.

Перед описание второго запроса вставим такой код:

```
//16
Если Режим = РежимПроведенияДокумента.Оперативный Тогда
    Движения.СтоимостьТоваров.Очистить ();
//17
Движения.СтоимостьТоваров.БлокироватьДляИзменения = Истина;
Движения.СтоимостьТоваров.Записать ();
КонецЕсли;

//7
Запрос.Текст = "ВЫБРАТЬ
```

16. Проверяем оперативно ли проводится документ.

17. Устанавливаем свойство "БлокироватьДляИзменения". Это позволит заблокировать от чтения те данные, которые сейчас будут удалены из регистра. Вдруг транзакция будет отменена, не хотим, чтобы кто-либо вместе с нами работал с этими данными.

Подобное действие не мешает и при смене даты или времени документа, а не только при оперативном проведении.

Вот теперь все. Полный текст модуля документа "Расходная":

Процедура ОбработкаПроведения(Отказ, Режим)

```
//1
Движения.ОстаткиТоваров.Записывать = Истина;

//2
Движения.ОстаткиТоваров.Очистить();

Для каждого Стр Из СписокТоваров Цикл
    Движение = Движения.ОстаткиТоваров.ДобавитьРасход();
    Движение.Период = Дата;
    Движение.Товар = Стр.Товар;
    Движение.Количество = Стр.Количество;
КонецЦикла;

//7
Движения.ОстаткиТоваров.БлокироватьДляИзменения = Истина;

//3
Движения.Записать();

Запрос = Новый Запрос;

//4
Запрос.МенеджерВременныхТаблиц = Новый МенеджерВременныхТаблиц;
Запрос.Текст = "ВЫБРАТЬ
    | Док.Товар КАК Товар,
    | СУММА(Док.Количество) КАК Количество
    | ПОМЕСТИТЬ ДокТЧ
    | ИЗ
    | Документ.Расходная.СписокТоваров КАК Док
    | ГДЕ
    | Док.Ссылка = &Ссылка
    |
    | СГРУППИРОВАТЬ ПО
    | Док.Товар
    |
    | ИНДЕКСИРОВАТЬ ПО
    | Товар
    | ;
    |

| //////////////////////////////////////
| ///
|
| ВЫБРАТЬ
| Остатки.Товар.Представление КАК ТоварПредставление,
| Остатки.КоличествоОстаток
| ИЗ
```



```

| РегистрНакопления.ОстаткиТоваров.Остатки (
|   &ТочкаИтогов,
|   Товар В
|   (ВЫБРАТЬ
|     ДокТЧ.Товар
|     ИЗ
|     ДокТЧ КАК ДокТЧ) ) КАК Остатки
| ГДЕ
| Остатки.КоличествоОстаток < 0
| ;
|

|////////////////////////////////////
|////
|
| ВЫБРАТЬ
| ДокТЧ.Товар
| ИЗ
| ДокТЧ КАК ДокТЧ";
Запрос.УстановитьПараметр ("Ссылка", Ссылка);

//5
Запрос.УстановитьПараметр ("ТочкаИтогов", Новый Граница (МоментВремени(),
ВидГраницы.Включая) );

ПакетРезультатов = Запрос.ВыполнитьПакет ();
РезультатЗапроса = ПакетРезультатов [1];

Если НЕ РезультатЗапроса.Пустой() Тогда
//6
Отказ = Истина;

Выборка = РезультатЗапроса.Выбрать ();
Пока Выборка.Следующий() Цикл
Сообщение = Новый СообщениеПользователю;
Сообщение.Текст = "Мало товара " + Выборка.ТоварПредставление + " нужно
еще " + (-Выборка.КоличествоОстаток);
Сообщение.Сообщить ();
КонецЦикла;
КонецЕсли;

Если Отказ Тогда
Возврат;
КонецЕсли;

//12
Блокировка = Новый БлокировкаДанных;
//13
ЭлементБлокировки =
Блокировка.Добавить ("РегистрНакопления.СтоимостьТоваров");
ЭлементБлокировки.Режим = РежимБлокировкиДанных.Исключительный;
//14
ЭлементБлокировки.ИсточникДанных = ПакетРезультатов [2] ;
//15
ЭлементБлокировки.ИспользоватьИзИсточникаДанных ("Товар", "Товар");
Блокировка.Заблокировать ();

```

```
//16
Если Режим = РежимПроведенияДокумента.Оперативный Тогда
    Движения.СтоимостьТоваров.Очистить ();
//17
    Движения.СтоимостьТоваров.БлокироватьДляИзменения = Истина;
    движения.СтоимостьТоваров.Записать ();
КонецЕсли;

//7
Запрос.Текст = "ВЫБРАТЬ
                | ДокТЧ.Товар КАК Товар,
                | ДокТЧ.Количество КАК Количество,
                | СтоимостьТоваров.Партия,
                | СтоимостьТоваров.КоличествоОстаток,
                | СтоимостьТоваров.СтоимостьОстаток
                | ИЗ
                | ДокТЧ КАК ДокТЧ
                | ЛЕВОЕ СОЕДИНЕНИЕ
РегистрНакопления.СтоимостьТоваров.Остатки (
                |      &ТочкаИтоговДляСебестоимости,
                |      Товар В
                |      (ВЫБРАТЬ
                |      ДокТЧ.Товар
                |      ИЗ
                |      ДокТЧ КАК ДокТЧ) ) КАК СтоимостьТоваров
                | ПО ДокТЧ.Товар = СтоимостьТоваров.Товар
                |
                | УПОРЯДОЧИТЬ ПО
                | СтоимостьТоваров.Партия.МоментВремени
                | ИТОГИ
                | МИНИМУМ(Количество)
                | ПО
                | Товар";

//8
Запрос.УстановитьПараметр ("ТочкаИтоговДляСебестоимости", МоментВремени());

//9
Движения.СтоимостьТоваров.Очистить ();

//10
РезультатЗапроса = Запрос.Выполнить ();
ВыборкаТовар =
РезультатЗапроса.Выбрать (ОбходРезультатаЗапроса.ПоГруппировкам);

Пока ВыборкаТовар.Следующий () Цикл

    ОсталосьСписать = ВыборкаТовар.Количество;

    ВыборкаПартия = ВыборкаТовар.Выбрать ();
    Пока ВыборкаПартия.Следующий () И ОсталосьСписать <> 0 Цикл

        Списать = МИН(ОсталосьСписать, ВыборкаПартия.КоличествоОстаток);

        Движение = Движения.СтоимостьТоваров.ДобавитьРасход ();
        Движение.Период = Дата;
```

```
Движение.Товар = ВыборкаПартия.Товар;  
Движение.Партия = ВыборкаПартия.Партия;  
Движение.Количество = Списать;  
Движение.Стоимость = Списать / ВыборкаПартия.КоличествоОстаток *  
ВыборкаПартия.СтоимостьОстаток;  
  
КонецЦикла;  
  
КонецЦикла;  
  
//11  
Движения.СтоимостьТоваров.Записывать = Истина;  
КонецПроцедуры
```

Надеюсь после этой статьи большая часть вопросов о "новой" методике проведения и о том, почему нельзя повсеместно писать "БлокироватьДляИзменения = Истина", будет закрыта.

## Павел Чистов

автор и тренер [курса по подготовке к Аттестации по Платформе 1С 8.2 / 8.3](#)

## Дополнительные материалы

Все статьи проекта **Курсы-по-1С.рф**: <http://курсы-по-1с.рф/blog/articles/>

## Курсы по программированию в 1С v.8

**Базовый и Продвинутой курсы по Программированию на Платформе 1С 8**

<http://www.Spec8.ru/>



### Базовый курс по программированию в 1С v.8

Курс про **готовые приемы и решения**  
**90% задач** по программированию в 1С



### Продвинутый курс по программированию в 1С v.8

Больше, чем Вы можете себе представить  
Детальнее требований на **1С:Специалист**

**«Курс по подготовке к Аттестации по Платформе 1С 8.2 / 8.3»**

<http://курсы-по-1с.рф/dev-attestation/>



### Подготовка к Аттестации по Платформе 1С v.8

Аттестация по Платформе – **с первого раза**  
**Экономия 100 - 150 часов** подготовки