**Понятие системы 1С:Предприятие. Основные объекты дерева метаданных. Понятие метаданных. Архитектурное решение: файловая версия, клиент-серверная версия. Понятие тонкого, толстого, веб-клиента. Обзор SQL-серверов. Сервер 1С:Предприятия.**

1С:Предприятие 8 – это система программ, которая предназначена для решения широкого круга задач по автоматизации административно – хозяйственной деятельности предприятий и организаций разного профиля.

Основной особенностью системы "1С: Предприятие 8" является ее конфигурируемость. Иными словами, платформа "1С: Предприятие представляет собой совокупность механизмов, предназначенных для манипулирования различными типами объектов предметной области.

Конфигурация это конкретный набор объектов, структуры информационных массивов, алгоритмы обработки информации разработанных для решения определенного круга задач.

1С:Предприятие 8 включает в себя технологическую платформу и конфигурации (иногда их называют бизнес-приложения, прикладные решения и т.п.), разработанные на ее основе.

Технологическая платформа (framework) – это среда исполнения и фундамент для построения приложений (конфигураций), независимая от конкретного законодательства и методологии учета. Платформа представляет собой набор различных механизмов, которые используются для разработки, администрирования и поддержки конфигураций.

В настоящее время фирма 1С активно развивает платформу 1С:Предприятие 8.х

Сама платформа не является программным продуктом для использования конечными пользователями, которые обычно работают с одной из многих конфигураций, разработанных на данной платформе. Такой подход позволяет автоматизировать различные виды деятельности, используя единую технологическую платформу и, соответственно, сходные механизмы работы.

Конфигурация (прикладное решение) – это совокупность настроек программы (структура метаданных, наборы программных модулей, пользовательских интерфейсов и прав), которая может быть изменена специальными средствами системы 1С:Предприятие.

Типовые конфигурации фирмы "1С" предназначены для автоматизации типовых задач учета и управления предприятий и организаций.

Примеры типовых тиражных конфигураций:
"1С:Бухгалтерия предприятия",
"Управление торговлей",
"Зарплата и Управление Персоналом",
"Управление производственным предприятием".

Для отражения изменений законодательства фирма 1С выпускает обновления конфигураций. Обновления выходят в виде новых релизов и редакций.

В новых редакциях расширяется состав задач, решаемых программой, вводятся новые возможности (документы, справочники, отчеты), улучшаются алгоритмы и внешнее оформление. В некоторых случаях выпуск новой редакции обусловлен существенными изменениями законодательства, требующими перестройки механизмов учета.

Релиз (release – выпуск) – используется для обозначения обновления программы, незначительно отличающейся от предыдущего. В основном выпуски релизов связаны с исправлением обнаруженных ошибок или изменениями, связанными с вышедшими нормативными актами и новыми формами регламентированных отчетов.

Для ведения юридически грамотного учета всегда следует пользоваться последним релизом конфигурации.

Все данные учета хранятся на компьютере в виде информационных баз. В каждой типовой конфигурации устанавливаются две базы, одна из которых пуста и предназначена для реальной работы пользователя, а вторая (с пометкой «демо») содержит демонстрационный пример ведения учета условного предприятия.
В 1С есть несколько видов клиентов, что позволяет использовать программу на разной компьютерной технике, разных операционных системах, географически распределенно.
Один из клиентов 1С позволяет использовать 1С с помощью обычного интернет браузера на любой операционной системе (хоть на Маке). Другой клиент 1С — на КПК, например, производственный КПК на складе для инвентаризации, со сканером штрихкодов.
Давайте рассмотрим какие бывают клиенты 1С, в чем их различия, как они выглядят и как с ними работать?

Толстый клиент 1С

Самый простой и известный всем клиент 1С – это толстый клиент 1С («обычный»). До версии 1С 8.2 кроме него никаких других вариантов предоставлено и не было.
Конфигуратор 1С (в настоящее время) работает только в толстом клиенте 1С. С файловой базой данных рекомендуется работать также с помощью толстого клиента 1С.
Толстый клиент 1С работает под Windows. Толстым его называют по причине требовательности к ресурсам компьютера пользователя. Также толстый клиент 1С может запрашивать достаточно большие объемы данных по сети.
С точки зрения программиста основное отличие толстого клиента 1С в том, что большинство программ на встроенном языке 1С он выполняет на компьютере пользователя. Например, 1С хочет выполнить запрос из базы данных:
Клиент 1С запрашивает данные у сервера 1С
Сервер 1С запрашивает данные из базы данных
Данные пересылаются на клиент 1С
Клиент 1С обрабатывает данные.

Тонкий клиент 1С

Тонкий клиент 1С появился сравнительно недавно. Для тонкого клиента 1С уже вышла конфигурация Управление торговлей (редакция 11). Тонкий клиент 1С устанавливается по умолчанию в комплекте с другими вариантами клиентов 1С, однако его можно установить отдельно (только его).
Конфигуратор 1С не работает в тонком клиенте 1С. Он может работать с файловым вариантом базы данных, однако лучше использовать клиент серверный режим.
Тонкий клиент 1С также работает под Windows. Тонким его называют по причине правильной организации клиент серверной организации программы. В отличии от толстого клиента 1С, запрос из базы данных будет выглядеть следующим образом:
Клиент 1С передает на сервер 1С необходимость пользователя запросить данные у сервера 1С
Сервер 1С запрашивает данные из базы данных
Сервер 1С обрабатывает данные

Результат обработки данных пересылается на клиент 1С.
Как Вы понимаете – сразу же рождается плюс и минус. Плюс – никакой требовательности к ресурсам компьютера пользователя, предполагается меньший трафик. Минус – требовательность к ресурсам сервера Выше.
Последний минус для больших компаний снимается тем, что сервер 1С можно масштабировать, то есть установить систему из нескольких серверов 1С на разных компьютерах и они будут работать в связке.
Интерфейс тонкого клиента 1С выглядит следующим образом. По умолчанию открывается рабочий стол пользователя. Он разбит на блоки по видам учета. Пользователь открывает закладку и использует гиперссылки для открытия списков.
Дополнительное отличие тонкого клиента 1С от толстого состоит в том, что он может работать не только по протоколу TCP/IP, как толстый, но и через HTTP, как веб клиент 1С.

Веб клиент 1С

Веб клиент 1С позволяет использовать 1С через обычный интернет браузер. Для использования не нужно ничего устанавливать дополнительно. Использовать можно под любой операционной системой, в том числе, например, на iPad.
Для использования веб клиента 1С требуется поднимать веб сервер. Он используется исключительно как транспорт и передает запросы на сервер 1С. Логика выполнения запросов и обработки данных в веб клиенте 1С такая же как и в тонком клиенте 1С. Для работы используется автоматическая конвертация встроенного языка 1С в JavaScript.
В веб клиенте 1С нельзя использовать любые конфигурации 1С — только написанные специально для работы с тонким клиентом 1С. По идее, разработка конфигурации для тонкого клиента 1С и для веб клиента 1С одинаковы (интерфейс и поведения системы должно быть одинаково тоже).
Однако есть слухи, что по крайней мере на данный момент не все так гладко и некоторые функции вызывают в веб клиенте 1С ошибки, хотя работают в тонком клиенте 1С.
Интерфейс веб клиента 1С мало отличается от тонкого клиента 1С.
В основе системы 1С:Предприятия лежит единая технологическая платформа и конфигурация. Конфигурации могут быть типовыми, т. е. созданными разработчиками компании 1С или создаваться на базе платформы под конкретные потребности заказчика.
Основу концепции системы 1С:Предприятие составляет понятие метаданные. Однако прежде чем дать расшифровку этого понятия, введем понятие объекта метаданных.

Под объектом метаданных в системе 1С:Предприятие понимается формальное описание группы понятий предметной области со сходными характеристиками и одинаковым предназначением.

Приведем такой пример. Объект метаданных «Справочник» в системе 1С:Предприятие предназначен для ведения списков однородных элементов данных — справочников, картотек, нормативных сборников и тому подобное. Использование объектов метаданных этого типа позволяет организовать ведение любых справочников, необходимых для автоматизации деятельности предприятия.

Как правило, объекты метаданных типа «Справочник» являются компьютерными аналогами реально существующих на предприятии справочников, например, справочника сотрудников или номенклатуры товаров, хотя могут использоваться и для организации списков, не имеющих явных физических аналогов.

Реализованный в системе 1С:Предприятие при помощи объекта метаданных компьютерный аналог конкретного понятия предметной области будем называть объектом данных.

Свойства объекта метаданных

Каждый объект метаданных обладает уникальным набором свойств. Этот набор описан на уровне системы и не может быть изменен в процессе настройки конфигурации задачи. Набор свойств объекта метаданных определяется, в основном, его назначением в системе 1С:Предприятие.

**Основные объекты конфигурации:**

Константы
Постоянные (условно-постоянные) величины. Константы хранят информацию, которая не изменяется или изменяется достаточно редко: название организации, се почтовый адрес и так далее.

Справочники
Списки однородных элементов данных. Используются для храпения нормативно-справочной информации.

Документы
Служат для ввода информации о совершенных хозяйственных операциях.

Журналы документов
Списки объектов данных типа «Документ». Служат для работы с документами.

Перечисления
Списки значений, задаваемые на этапе конфигурирования.

Отчеты
Средство получения выходной информации. Источником данных для построения отчетов служат документы, справочники и регистры, также используется информация, хранящаяся в константах.

Обработка
Объекты метаданных этого вида используются для выполнения различных действий над информационной базой.

Планы Счетов
Списки объектов данных типа «бухгалтерский счет» — учетных регистров, по которым будет выполняться группировка средств при работе с системой 1С:Предприятие. Понятие «план счетов» в системе 1С:Предприятие вполне соответствует общепринятому пониманию аналогичного термина в бухгалтерском учете.

Регистры
Средство накопления оперативной информации о наличии и движении средств.

Итак, теперь можно дать расшифровку самого понятия «метаданные».

Метаданными («данными о данных») в системе 1С:Предприятие называется совокупность объектов метаданных, настроенных на хранение и обработку информации о хозяйственной деятельности конкретного предприятия.

Наряду с понятием метаданные, в настоящем Руководстве будет широко использоваться

термин «структура метаданных». Данный термин более точно отражает суть метаданных, как сложной структуры взаимодействующих объектов метаданных. Фактически, структура метаданных является моделью предметной области.

Конфигурацией в системе 1С:Предприятие называется совокупность трех взаимосвязанных составных частей:

- структуры метаданных;
- набора пользовательских интерфейсов;
- набора прав.

Создание конфигурации выполняется при помощи Конфигуратора. Созданная конфигурация используется системой 1С:Предприятие для реализации программного окружения, пригодного для выполнения необходимых учетных задач.

Пользовательским интерфейсом в системе 1С:Предприятие называется совокупность команд главного меню и панелей инструментов, настроенных на работу с конкретными объектами данных — документами, справочниками, журналами и т. д. Как правило, пользовательский интерфейс создастся для конкретной категории пользователей. Цель создания интерфейса — обеспечить быстрый доступ пользователей к той информации, которая необходима им в соответствии с их обязанностями.
Права в системе 1С:Предприятие определяют полномочия пользователей на работу с информацией, которая обрабатывается в системе. Совокупность предоставляемых пользователю прав определяется, как правило, кругом его обязанностей.
Операция назначения прав пользователю решает две основные задачи.
С одной стороны, ограничивается круг пользователей конфиденциальной информации, которая, безусловно, всегда присутствует в любой системе учета.
С другой стороны, запрет выполнения определенных операций (в первую очередь, операций удаления и корректировки данных) позволяет в какой-то степени предотвратить возможные потери информации.
Все три составные части конфигурации тесно связаны между собой и требуют, как правило, согласованного внесения изменений (особенно это касается пользовательских прав).
1С является клиент серверным ПО и это значит, что 1С состоит из двух программ — клиентской и серверной. Серверная программа 1С запущена на сервере. Пользователь на своем компьютере работает в клиентской программе 1С, которую кратко называют клиент 1С.

Поскольку SQL не является привычным процедурным языком программирования (то есть не предоставляет средств для построения циклов, ветвлений и т. д.), вводимые разными производителями расширения касались в первую очередь процедурных расширений. Это хранимые процедуры (stored procedures) и процедурные языки-«надстройки». Практически в каждой СУБД применяется свой процедурный язык. Стандарт для процедурных расширений представлен спецификацией SQL/PSM. Перечень процедурных расширений для самых популярных СУБД приведён в следующей таблице:СУБД Краткое название Расшифровка
InterBase/Firebird PSQL Procedural SQL
IBM DB2 SQL PL (англ.) SQL Procedural Language (расширяет SQL/PSM); также в DB2 хранимые процедуры могут писаться на обычных языках программирования: Си, Java и т. д.
MS SQL Server/
Sybase ASE Transact-SQL Transact-SQL
MySQL SQL/PSM SQL/Persistent Stored Module

Oracle PL/SQL Procedural Language/SQL (основан на языке Ada)
PostgreSQL PL/pgSQL Procedural Language/PostgreSQL Structured Query Language (очень похож на Oracle PL/SQL)

Лекция 2.

**Универсальная коллекция значений. Встроенный язык. Менеджеры справочников. Архитектура модуля, контекст среды исполнения, основной реквизит формы. Дуализм реляционной базы данных и предметно-ориентированной среды разработки. Ссылочная природа. Отношение сущностей и реализация отношений средствами 1С. Понятие справочника и регистра.**

«Другие» типы, не относящиеся к примитивным и «добавляемым», но поддержка которых во встроенном языке есть изначально очень часто являются коллекциями (можно «обойти» как по индексу, так и с помощью специального вида цикла «Для Каждого Из»). Часть этих типов входят в так называемые **Универсальные коллекции значений.**
Коллекция представляет собой не хранящуюся в базе данных совокупность значений или объектов — элементов коллекции. В 1С:Предприятии работа с коллекциями была унифицирована: добавление, удаление и обход элементов всех коллекций производится однотипным образом.
Некоторые коллекции называются *универсальными,* потому что разработчик может их создавать средствами встроенного языка для решения своих задач.
К универсальным коллекциям значений относятся следующие:
• массив,
• список значений,
• таблица значений,
• дерево значений,
• структура,
• соответствие.
Общее у этих объектов то, что они содержат несколько значений (элементов коллекции), позволяют добавлять и удалять значения. Они не хранятся в базе данных, а используются для работы с временными наборами данных для вспомогательной обработки, группировки и анализа информации.
Все эти объекты создаются средствами встроенного языка с помощью ключевого слова Новый.
Массив, структура и соответствие имеют конструкторы с параметрами, которые определяют начальное состояние объекта, например, количество элементов в массиве.
Для универсальных коллекций значений доступен обход элементов с помощью конструкции «Для Каждого». Также возможно обращение к элементу с помощью оператора [...] (квадратные скобки), куда в качестве аргумента обычно передается индекс элемента, а для структуры или соответствия — ключ элемента коллекции.
**Внимание!** Индексация элементов коллекций всегда начинается с 0.

## Массив

Массивы представляют собой пронумерованную совокупность значений любого типа. К элементу массива можно обращаться по индексу через оператор [...], причем индекс первого элемента равен 0.
В качестве значений элементов массива могут быть другие массивы, что, в частности, позволяет организовывать многомерные массивы.
Продемонстрируем на примерах возможности работы с массивами:
Создание нового массива производится следующим образом:

```
Мае = Новый Массив; //одномерный массив без элементов
Мас2 = Новый Массив(10); //одномерный массив из 10 элементов
Мас3 = Новый Массив(4,5); //массив с двумя измерениями
```

Структура и соответствие

Структура и соответствие являются динамическими наборами данных — коллекциями значений. Каждый элемент такой коллекции представляет собой пару «ключ» и «значение». Ключи структуры и соответствия уникальны, поэтому они однозначно идентифицируют элемент коллекции.
**Внимание! В** структуре ключи могут быть только строковые и должны подчиняться правилам именования переменных. В соответствиях ключи могут быть почти любого типа.
Структуры часто применяются в методах объектов для указания отборов, например:

```
Отбор = Новый Структура("Цена", 100);
```

```
Выборка = Справочники.Номенклатура.Выбрать(,Отбор);
```

## Список значений

Список значений — это аналог одномерного массива, который предназначен в основном для решения интерфейсных задач, например, при использовании элементов управления СписокВыбора и ПолеВыбора.
В списке значения могут храниться значения разных типов, хотя можно наложить ограничение с помощью свойства ТипЗначения. Список значений создается в памяти и не сохраняется в базе данных, т.е. это временный набор данных.
Список значений можно представить себе как таблицу, содержащую следующие колонки:

- **Значение** — собственно хранимое значение.
- **Представление** — пользовательское представление значения, которое будет показано при выводе значения на экран, например, в диалоге выбора значения из списка. Если представление не указано, то оно формируется системой автоматически.
- **Пометка** — пометка (флажок), связанная со значением. Может быть использована для отметки некоторых значений из списка.
- **Картинка** — графическое изображение, связанное с данным значением.

При размещении в форме поля списка автоматически создается новый реквизит формы типа СписокЗначений. Через этот реквизит можно управлять содержимым списка, добавлять или удалять элементы, расставлять пометки, задавать картинки. Следует понимать, что ПолеСписка — это не список значений, *а элемент управления,* имеющий собственный набор свойств и методов.
Со списком значений также тесно связан элемент управления ПолеВыбора. Его свойство СписокВыбора представляет собой список значений, из которого пользователь может выбрать одно значение.

## Таблица значений

Таблица значений — это не сохраняемый в базе данных объект, предназначенный для создания динамических наборов данных. Таблица значений состоит из строк и колонок. В отличие от списка значений, таблица значений имеет структуру, которую определяет разработчик.

## Дерево значений

Дерево значений — это объект для хранения временных наборов данных, возникающих при работе программы.
Работа с деревом значений напоминает работу с таблицей значений, за исключением того, что в дереве могут быть представлены иерархические данные (по аналогии с иерархическим справочником). Это достигается тем, что каждая строка дерева значений может иметь подчиненные строки, а те, в свою очередь, тоже могут иметь подчиненные строки и т.д.

**Встроенный язык** позволяет алгоритмически определить поведение объектов конфигурации.
Средства встроенного языка 1С:Предприятия 8.0 позволяют управлять практически всеми аспектами поведения системы, работать с прикладными объектами, например, справочниками и документами, формировать печатные формы отчетов и выполнять другие самые разнообразные действия.
Важно понимать, что встроенный язык системы 1С:Предприятие не предназначен для написания отдельных выполняемых приложений, а является неотъемлемой частью платформы. Прикладное решение использует механизмы 1С:Предприятия и работает только под управлением платформы, оно не может быть использовано самостоятельно, как отдельное приложение.

### Программные модули

Программный модуль представляет собой текст на встроенном языке 1С:Предприятия, расположенный в определенном месте конфигурации.
В соответствии с этим различают следующие виды программных модулей:

**Общие модули**

Общие модули принадлежат всей конфигурации в целом, но в отличие от модуля приложения (см. ниже) их может быть несколько. Общие модули не могут содержать объявлений переменных, и в них нет раздела основной программы, таким образом, они состоят только из процедур и функций.

**Модуль приложения**

Это модуль, который относится ко всей конфигурации в целом и может быть только один. Он отвечает за пользовательскую сессию (сеанс) работы с 1С:Предприятием.

**Модуль внешнего соединения**

Если конфигурация запускается не в режиме клиентской сессии, а через COM-соединение, то вместо модуля приложения используется модуль внешнего соединения, который в конфигурации может быть только один.

**Модули прикладных объектов**

У некоторых прикладных объектов конфигурации могут быть собственные модули, например, модуль документа, модуль справочника. Не следует их путать с *модулями форм* этих объектов.

**Модули форм**

У каждой формы есть модуль, в котором определяется поведение формы и действия, выполняемые из нее, например, открытие других форм.

## Архитектура модуля

Любой программный модуль, за исключением общих модулей, состоит из следующих разделов:
• раздел объявления переменных,
• раздел процедур и функций,
• раздел основной программы.

У общих модулей есть только раздел процедур и функций.

В разделе объявления переменных определяются локальные переменные модуля и экспортируемые переменные, которые объявляются с ключевым словом Экспорт. Далее идут процедуры и функции, а затем раздел основной программы
Раздел процедур и функций содержит тела процедур и функций, написанных разработчиком вручную или сформированных конструктором. Некоторые процедуры и функции могут быть объявлены с ключевым словом Экспорт, тогда они дополняют контекст объекта и становятся видимыми вне объекта.

Процедуры и функции рекомендуется отделять комментарием с разделителем. В виде комментариев рекомендуется указывать назначение процедуры или функции, параметры и возвращаемое значение для функций.
В самом конце модуля, после всех процедур или функций, следует раздел основной программы, состоящий из выполняемых операторов. Эти операторы будут исполнены при инициализации модуля, например, для модуля справочника — при создании объекта типа

СправочникОбъект, для модуля формы — при создании объекта Форма. Обычно в этом блоке операторов производится инициализация переменных модуля и заполнение полей начальными значениями.

**Контекст**

Контекст — очень важное понятие при программировании на любом языке. В 1С:Предприятии контекст обозначает *окружение* модуля, т.е. какие ему будут доступны переменные, объекты, свойства, методы и события.
Можно выделить следующие виды контекстов, существующих в 1С:Предприятии:
Глобальный контекст, доступный во всех остальных контекстах, состоит из следующих частей:
- свойства, методы и события глобального контекста (например, свойство РабочаяДата),
- системные перечисления и системные наборы значений (например, КодВозвратаДиалога и Символы).

В контексте модуля приложения (или модуля внешнего соединения) доступны экспортируемые переменные, процедуры и функции общих модулей.
В контексте общего модуля доступны экспортируемые процедуры и функции других общих модулей. В этом контексте недоступны экспортируемые переменные, процедуры и функции модуля приложения.
В контексте модуля прикладного объекта есть доступ к реквизитам и табличным частям объекта, а также его методам и событиям. Например, в модуле документа РасходнаяНакладная доступны реквизиты документа и его табличные части, можно вызывать методы документа и обрабатывать события.
В контексте модуля формы доступны реквизиты формы, а также ее свойства, методы и события. Если у формы назначен основной реквизит, то в модуле формы становятся доступны свойства и методы прикладного объекта, используемого в качестве основного реквизита.
Необходимо помнить правила видимости экспортируемых переменных, процедур и функций различных модулей:
- В общем модуле недоступны экспортируемые переменные, процедуры и функции модуля приложения (модуля внешнего соединения).
- В модуле приложения (модуле внешнего соединения) доступны экспортируемые процедуры и функции общих модулей.
- В общих модулях доступны экспортируемые процедуры и функции других общих модулей.
- В модулях прикладных объектов и модулях форм доступны экспортируемые переменные, процедуры и функции модуля приложения (модуля внешнего соединения), а также экспортируемые процедуры и функции общих модулей.
- Если у формы назначен основной реквизит, то контекст модуля формы содержит дополнительные свойства и методы, связанные с основным реквизитом. Например, в модуле формы элемента справочника Номенклатура доступны свойства и методы объекта «СправочникОбъект.Номенклатура».

Система 1С:Предприятие обеспечивает два способа доступа к данным — объектный (для чтения и записи) и табличный (только для чтения). Объектный способ доступа к данным подразумевает обращение к объекту как к единому целому, т.е. доступны все реквизиты объекта, подчиненные объекты и его методы, табличный подразумевает доступ к данным сразу нескольких объектов, что удобно для обработки больших объемов данных, однако реквизиты и методы отдельных объектов в данном случае могут быть не доступны.

**Справочники**

Справочники являются основным механизмом хранения условно-постоянной информации. Справочники часто используются в тех случаях, когда необходимо исключить неоднозначный ввод информации. Каждый справочник представляет собой список однородных объектов: сотрудников, организаций, товаров и т.д. Такие объекты называются *элементами справочника.*
Для упрощения понимания можно считать, что справочник — это обычная таблица с заданными колонками. Но в 1С:Предприятии есть возможность создания *иерархических* справочников и *табличных частей* у справочников. Отсюда можно сделать вывод, что справочник — это не просто таблица, а достаточно сложный механизм хранения данных.
На этапе разработки можно описать, какими свойствами обладает справочник.
Во всех справочниках есть системные реквизиты Код и Наименование (если только у них не установлена нулевая длина).
Код элемента справочника может быть как числовым, так и строковым. Система 1С:Предприятие предоставляет широкие возможности по работе с кодами элементов справочника: автоматическое присвоение кодов новым элементам, контроль уникальности кода и т.д.

**Подчиненные справочники.**

Между справочниками может быть установлено отношение *подчиненности.* В этом случае каждый элемент подчиненного справочника будет связан с одним из элементов справочника-владельца. Часто можно сказать, что элементы одного справочника *принадлежат* элементам другого. В 1С:Предприятии у справочника может быть несколько владельцев, то есть справочник может быть подчинен сразу нескольким справочникам или другим объектам. Но каждый элемент подчиненного справочника имеет одного и только одного владельца, который хранится в одном из объектов-владельцев.
Такая схема позволит избавиться от дублирования кода и упростить конфигурацию.
Справочник может быть подчинен не только другим справочникам, но также планам видов характеристик и планам счетов. Назначение такого подчиненного справочника — хранение подчиненных объектов о конкретной характеристике или о конкретном счете.

**Иерархические справочники.**

Список элементов справочника в системе 1С:Предприятие 8.0 может быть иерархическим (или многоуровневым). Использование иерархических справочников позволяет сгруппировать элементы справочника по какому-либо признаку с нужной степенью детализации.
Иерархический справочник может быть двух видов:

- **Элементы и группы**

В первом случае все элементы иерархического справочника разделяются на «просто» элементы справочника и группы справочника. Группы могут отличаться по структуре от обычных элементов, т.е. могут содержать другой состав реквизитов. Для каждого реквизита указывается, относится ли он только к элементу, только к группе или к обоим.

- **Только элементы**

Во втором случае иерархический справочник состоит только из элементов. Любой элемент может выполнять функцию группы, т.е. он может быть родительским элементом для других элементов этого же справочника. Например, по такому принципу можно организовать справочник Подразделения. В любой момент времени какой-нибудь отдел может быть разделен внутри себя на несколько подотделов, и отразить это в справочнике не составит

никакого труда.
Максимально возможное количество уровней иерархии справочника задается в Конфигураторе.

У справочника может быть несколько экранных форм. Даже если разработчик не создал ни одной экранной формы для справочника, то они будут сгенерированы «на лету» при выполнении программы. Конечно, автоматически созданная экранная форма может быть не самой удобной для пользователя, поэтому, когда это требуется, рекомендуется создавать собственные экранные формы с эргономичным интерфейсом.

Ниже поясняется назначение различных экранных форм справочника:
**Форма списка**
Отображает список элементов справочника, содержит средства сортировки, поиска и отбора элементов. Из списка элементов можно открыть форму элемента (группы) или редактировать данные прямо в табличном поле. Это регулируется свойством табличного поля СпособРедактирования.
**Форма элемента**
Отображает и позволяет редактировать сведения об одном элементе справочника. Если у справочника много реквизитов, то они могут быть распределены по закладкам.
**Форма группы**
Отображает и позволяет редактировать сведения о группе справочника. Группы могут иметь другой состав реквизитов по сравнению с элементами.
**Форма выбора элемента**
Предназначена для выбора элемента из списка. Такая форма вызывается при выборе значения, например, в поле ввода, связанного с реквизитом документа.
**Форма выбора группы**
Предназначена для выбора группы из списка. Выбор группы может понадобиться, например, для отчета по товарам, который позволяет вывести не все товары, а только товары заданной группы. Эта форма также используется при интерактивном переносе элемента в другую группу.
**Произвольная форма**
Данная форма не вызывается системой автоматически, как предыдущие типы форм. Назначение произвольной формы и точки ее вызова определяет разработчик. Основное отличие произвольной формы от других состоит в том, что для нее *не назначен* основной реквизит, следовательно, контекст формы не включает дополнительные свойства и методы, характерные для основного реквизита.

**Менеджеры справочников**

Работа со справочниками из программы производится с помощью нескольких объектов. Каждый объект имеет свое назначение, собственные свойства и методы.
Ниже описываются эти объекты и взаимосвязи между ними:
**СправочникиМенеджер**
Обеспечивает доступ ко всем справочникам конфигурации. Свойства этого объекта совпадают с именами справочников и содержат объекты типа СправочникМенеджер.
**СправочникМенеджер**
Обеспечивает доступ к операциям над справочником как множеством элементов. Через методы этого объекта можно осуществлять поиск, получать выборку, создавать новые элементы или группы, обращаться к формам и макетам справочника.
**СправочникСсылка**
Данный объект однозначно идентифицирует элемент (группу) справочника и позволяет обращаться к нему в режиме «только чтение». Через свойства и методы этого объекта можно

прочитать реквизиты элемента (группы), обратиться к его табличным частям. Значение этого типа хранится в реквизитах, ссылающихся на элементы данного справочника.
Продемонстрируем на примерах выполнение типичных операций со справочниками.

**Обращение к менеджеру справочника**

Обращение к менеджеру справочника производится следующим образом:

```
СпрСотр = Справочники.Сотрудники; СпрТовары =
Справочники["Номенклатура"];
```

В новой версии используется свойство глобального контекста Справочники, которое предоставляет доступ к объекту типа СправочникиМе-неджер (обратите внимание на множественное число). Данный объект через свои свойства предоставляет доступ к объектам типа СправочникМенеджер (единственное число).
Объект СправочникиМенеджер является коллекцией значений и позволяет обратиться к конкретному элементу коллекции через приведенные выше конструкции. Хотя эти две конструкции функционально эквивалентны, но нужно знать различные варианты их использования:
1. Первый вариант (через точку) применяется, когда имя справочника известно программисту и оно неизменно. Такая конструкция более простая и рекомендуется к применению в большинстве случаев.
2. Во втором варианте имя справочника задается строкой. Такая конструкция может быть использована, когда неизвестно имя справочника, с которым будут производиться действия, например, в универсальном отчете ПечатьСправочника, который выводит на печать содержимое любого заданного справочника.

**Понятие простого и обусловленного проведения документов. Модуль формы. Модуль объекта. Способы внесения данных в регистры накопления. Обращение к результатам регистров.**

Как мы уже говорили, документы предназначены для хранения информации обо всех событиях, происходящих на предприятии и имеющих смысл с точки зрения экономики. При помощи документов отражаются платежи с расчетного счета и операции по кассе, кадровые перемещения сотрудников и движения товаров, а также другие события.
Типичными примерами документов являются платежное поручение, счет, приходная накладная, расходная накладная, приходный кассовый ордер.
Данные, вводимые в реквизиты документа, обычно содержат информацию о событии, например, в расходной накладной — информацию о том, с какого склада, каких товаров и сколько отгружено; в приказе о приеме на работу — информацию о сотруднике, оклад и другие сведения; в договоре с клиентом — условия договора, график плановых платежей и т.д.
Кроме собственно записи, для документа весьма важным аспектом является его *проведение.* Именно при проведении документ обычно отражает зафиксированное им событие в регистрах.
В общем проведение документов можно разделить на два вида: безусловное и обусловленное.
При безусловном проведении документов, мы берем информацию из документа и пишем в регистр, т.е. для выполнения движений целиком и полностью достаточно информации, находящейся в самом документе.
я обусловленного проведения информации, содержащейся только в документе недостаточно, значит ее нужно откуда-то взять. Тогда возникает еще один вопрос, из каких же источников можно использовать информацию при проведении документов.
С точки зрения надежности и простоты конфигураций — при организации обусловленного проведения, в качестве «чужих» данных использовать данные только из регистров, причем записи этих регистров должны быть подчинены регистратору. При получении данных из других объектов придется контролировать связи, между этими объектами и саму информацию в текущем документе.
Обусловленное проведение можно разделить на два вида: простое и сложное. Простое можно рассмотреть на примере проведения документа «Расходная». При списании товара необходимо не только контролировать текущий остаток товара на складе, но и себестоимость этого товара при списании. Таким образом при проведении документа «Расходная», мы предварительно должны проанализировать регистр накопления для определения остатков по тому или иному товару на момент реализации (момент времени может определяться по регистратору).
К сложному обусловленному проведению есть смысл обращаться тогда, когда есть возможность сэкономить на стадии сбора информации из источников, т.е. кроме контроля остатков и себестоимости, нужно еще контролировать взаиморасчеты с контрагентами.
Самый быстродействующий способ получения разнородной информации (из разных источников) — это запрос.

Каждый прикладной объект конфигурации, данные которого могут быть модифицированы в режиме 1С:Предприятие, имеет свой модуль. Этот модуль исполняется при создании объекта встроенного языка, которые позволяет модифицировать данные объекта конфигурации. Соотвестствующий объект встроенного языка создается, например при вводе нового объекта, при копировании, при изменении данных существующего объекта и т.д. Для различных объектов конфигурации этот модуль имеет разное название, например для справочников, констант, документов и т.п. Он называется модуль объекта, а для последовательностей и регистров — модуль набора записей. Контекст модуля объекта образуется:

- глобальным контекстом, в том числе экспортируемыми функциями и процедурами общихмодулей (в зависимости от места создания объекта);
- экспортируемыми переменными, процедурами и функциями модуля приложения или модуля внешнего соединения (в зависимости от места создания объекта);
- свойствами и методами встроенного языка, контекст которого расширяется модулем;
- реквизитами объекта конфигурации, которому «принадлежит» модуль;
- локальным контекстом самого модуля.

Если переменные процедуры и функции модуля объекта определены как экспортируемые, то они будут доступны в качестве свойств и методов соотвестствующих объектов встроенного языка. Помимо описания переменных и основной программы модуль объекта может содержать описания процедур-обработчиков событий, связанных с самим объектом.
Каждая форма, определенная в конфигурации, имеет свой собственный модуль. Этот модуль исполняется при создании объекта Форма встроенного языка. Этот объект может создаваться при открытии формы прикладного объекта (например при открытии формы элемента справочника) либо явно получаться средствами встроенного языка

Форма = Справочники.Номенклатура.ПолучитьФормуСписка();

Контекст модуля формы образуется:

глобальным контекстом, в том числе экспортируемыми функциями и процедурами общих модулей (если для этих модулей установлено хотя бы одно изсвойств Клиент или Сервер);

экспортируемыми переменными,процедурами и функциями модуля приложения;

свойствами и методами объекта, который назначен основным реквизитом формы, включая экспортируемые переменные, процедуры и функции, определенные в модуле этого объекта;

свойствами и методами расширения формы, основным реквизитом формы;

свойствами и методами объекта Форма встроенного языка;

реквизитами формы, которой «принадлежит» модуль;

локальным контекстом самого модуля формы.

Если переменные, процедуры и функции модуля формы определены как экспортируемые, то они будут доступны как свойства и методы соотвестствующихобъектов Форма встроенного языка.
Помимо описания переменных и основной программы, модуль формы может содержать описание процедур-обработчиков событий, связанных с формой. Основными событиями, которые могут обрабатываться в модуле формы являются события открытия и закрытия окна формы.

**Модуль объекта.**

У некоторых объектов конфигурации могут быть собственные модули, например, модуль документа РасходнаяНакладная, модуль справочника Номенклатура. Не следует их путать с *модулями форм* этих объектов.

В модуле документа располагаются процедуры и функции, связанные с документом как объектом конфигурации, независимо от его экранных форм.
Например, там и только там располагаются обработчики следующих событий:

**ОбработкаПроведения**
Событие возникает в момент проведения документа. Обычно в нем содержится алгоритм формирования движений документа по регистрам. Событие имеет два параметра: Отказ и Режим. Если в обработчике события первому параметру присвоить значение Истина, тогда

документ не будет проведен. Второй параметр позволяет узнать режим проведения документа: оперативный или неоперативный. Перед началом выполнения данной процедуры, если документ раньше сформировал движения, то они обычно очищаются системой автоматически.

**ОбработкаУдаленияПроведения**

Событие возникает в момент отмены проведения документа. При этом движения документа, как правило, очищаются автоматически.

**ОбработкаЗаполнения**

Это событие возникает при вводе документа на основании другого документа (или, например, элемента справочника), который передается в параметр Основание. В данной процедуре программист должен предусмотреть заполнение реквизитов документа.

**ПередЗаписью**

Это событие возникает перед выполнением записи объекта после начала транзакции, но до начала записи документа в базу данных.

**ПередУдалением**

Событие возникает перед удалением документа из базы, независимо от того, программно или интерактивно он удаляется.

**ПриУстановкеНовогоНомера**

Данное событие возникает в момент, когда выполняется установка нового номера документа.

**ПриКопировании**

Это событие возникает при создании документа копированием. В процедуре-обработчике можно произвести дополнительные действия, например, присвоить новую дату и новый номер, заполнить реквизит Автор и т.д.

**Замечание.** Все эти процедуры-обработчики событий вызываются при наступлении указанных событий независимо от того, как они были инициированы: программно или интерактивно. Даже если документ записывается методом Записать, то будут вызваны процедуры-обработчики ПередЗаписью и ПриЗаписи. Это следует учитывать при разработке конфигураций.

Кроме вышеперечисленных процедур, в модуле документа могут располагаться процедуры и функции, созданные разработчиком. Если их объявить с ключевым словом Экспорт, то они станут доступными *вне* документа, как обычные методы.

Часто таким образом производится печать документов.

**Приемы программирования**

С помощью встроенного языка 1С:Предприятия 8 можно выполнять различные операции над документами: создавать, изменять и удалять их, искать документ по номеру, перебирать их в цикле и т.д.

Работа с документами производится с помощью нескольких программных объектов (аналогичные объекты мы уже рассматривали для справочников):

**ДокументыМенеджер**

Обеспечивает доступ к менеджерам всех документов конфигурации. Свойства этого объекта совпадают по имени с именами документов и содержат объекты типа Документ-Менеджер.

**ДокументМенеджер**

Обеспечивает доступ к действиям над документами определенного вида как множеством объектов. Через методы этого объекта можно осуществлять поиск, получать выборку, создавать новый документ, обращаться к формам и макетам документа.

**ДокументСсылка**

Это специальный объект, однозначно идентифицирующий документ в базе данных и позволяющий обращаться к нему в режиме «только чтение». Значение данного типа хранится, например, в реквизитах других объектов, ссылающихся на документы данного вида.

Через свойства и методы этого объекта можно прочитать реквизиты документа, обратиться к его табличным частям.
**ДокументОбъект**
Предоставляет доступ к документу с возможностью записи и проведения. Данный объект содержит методы, изменяющие документ в базе данных, например, метод Записать.
**ДокументВыборка**
Предоставляет возможность обхода (перебора) документов определенного вида. Объект этого типа возвращается методом Выбрать объекта ДокументМенеджер.
**ДокументСписок**
Объект для управления списком документов, отображаемым в форме. Позволяет управлять колонками, отбором и сортировкой в списке. Этот объект не может быть создан программно, он создается системой автоматически при размещении в форме табличного поля, отображающего список документов.
Из одних объектов с помощью определенных свойств или методов можно получить другие объекты.
***Обращение к менеджеру документа***
Обращение к менеджеру документа производится следующим образом:
ДокСчет = Документы.Счет; ДокСчет = Документы["Счет"];
При обращении к менеджеру документа ошибкой будет применение ключевого слова Новый. Оно предназначено для создания новых объектов, например, списков значений. Следует помнить, что при работе с прикладными объектами всегда используется обращение через менеджер.
**Внимание!** С помощью ключевого слова Новый нельзя создавать новые документы. Для этого предназначен метод СоздатьДокумент объекта ДокументМенеджер, который возвращает объект типа ДокументОбъект.<ИмяДокумента>.

**У документов есть атрибут Движения, который предоставляет доступ к наборам записей этого документа по каждому регистру, поэтому при проведении документов сначала добавляются записи в набор, а затем набор записывается в базу данных.**

**Набор записей** позволяет оперировать сразу несколькими записями регистра. Набор записей можно целиком прочитать из базы данных, добавить в него записи или изменить их, целиком записать в базу данных. Важно понять, что это единственно возможный способ добавления и изменения записей регистра накопления.

Можно создать новый набор записей из объекта РегистрНакопленияМенеджер, если вызвать его метод СоздатьНаборЗаписей, например:

**Набор = РегистрыНакопления.Продажи.СоздатьНаборЗаписей();**
**Набор.Отбор.Регистратор.Значение = ВыбДок;**
**Движ = Набор.Добавить();**
**Движ.Регистратор = ВыбДок;**
**Движ.Номенклатура = ВыбТовар;**
**Движ. Контрагент = ВыбКонтрагент ,-**
**Движ.Период = РабочаяДата;**
**Движ.Количество = Количество;**
**Движ.Сумма = Сумма;**
**Набор.Записать(Ложь); //добавить к набору записей по документу**

**Внимание!** При записи набора методом Записать с параметром Истина сначала удаляются все записи с заданным отбором, а затем на их место записываются новые. Если передать параметр Ложь, то запись будет добавлена к уже существующим. Напомним, что, в отличие от регистра сведений, для регистра накопления две записи с одинаковыми полями — вполне

нормальная ситуация.

Через набор записей можно обращаться к уже существующим записям регистра накопления. Для этого нужно установить свойство Отбор и прочитать записи из базы данных. Свойство Отбор является объектом типа Отбор, свойства которого совпадают с именами измерений регистра и являются объектами типа ЭлементОтбора.
Например:

**Набор = РегистрыНакопления.Продажи.СоздатьНаборЗаписей();**
**Набор.Отбор.Регистратор.Значение = ВыбДок;**
**Набор.Прочитать();**
**Для Каждого движ Из Набор Цикл**
**Сообщить(движ.Сумма);**
**КонецЦикла;**

Записи набора можно выгружать в таблицу значений с помощью метода Выгрузить или, наоборот, загружать записи в набор из таблицы значений с помощью метода Загрузить. При выгрузке структура таблицы значений полностью соответствует структуре набора. При загрузке загружаются только те колонки, имена которых в наборе и таблице значений совпадают:

**Набор.Прочитать();**
**тзЗаписи = Набор.Выгрузить();**
**Набор.Загрузить(тзЗаписи); Набор.Записать();**

Можно выгружать в массив и загружать из массива только одну колонку набора записей, для чего предназначены методы Выгру-зитьКолонку и ЗагрузитьКолонку:
**МассивОстатков = Набор.ВыгрузитьКолонку("Количество");**
**Набор.ЗагрузитьКолонку(МассивОстатков,"Количество");**
Выгружаются и загружаются записи, имеющие тот же индекс, что и элементы массива (индексация записей набора и элементов массива начинается с 0).
Для удаления записей из набора существует два метода Удалить и Очистить. Первый метод удаляет запись с заданным индексом, но в качестве параметра можно передавать и саму запись (объект типа РегистрНакопленияЗапись). Второй метод полностью очищает набор, удаляя все записи. Обратите внимание, что записи удаляются из набора, а не из регистра накопления.
Для чтения записей или итогов регистра, кроме методов Выбрать, Остатки и Обороты, могут быть использованы запросы, о чем рассказано ниже в этой главе. О самом механизме запросов вы можете прочитать в главе «Запросы».

***Модуль набора записей***
*У* набора записей регистра накопления есть собственный модуль, который открывается из окна редактирования свойств регистра. Можно считать, что модуль набора записей является аналогом модуля любого другого прикладного объекта, например, модуля справочника или модуля документа.
В модуле набора записей регистра накопления могут располагаться процедуры-обработчики событий ПередЗаписью и ПриЗапи-си. Их назначение точно такое же, как у модуля набора записей регистра сведений, описанного в предыдущей главе.
***Выборка из регистра накопления***
Объект РегистрНакопленияВыборка предназначен для динамического обхода записей регистра. При этом записи считываются из базы данных порциями по мере выборки. Такая схема рационально использует память и позволяет перебрать множество записей регистра,

например:
**Выборка = РегистрыНакопления.Остатки.Выбрать(); Пока Выборка.Следующий() Цикл**
**Сообщить(СокрЛП(Выборка.Товар) + " " +**
**СокрЛП(Выборка.Склад) + " " +**
**СокрЛП(Выборка.Количество));**
**КонецЦикла;**
Выборку можно ограничить только нужными записями, используя параметры метода Выбрать.
Синтаксис метода Выбрать следующий:
**Выбрать(<Начало интервала>, <Конец интервала>, <0тбор>, <Порядок>)**
Назначение параметров метода Выбрать таково:
**Начало интервала, Конец интервала**
Позволяют задать границы интервала и могут иметь тип «дата», МоментВремени или Граница.
**Отбор**
Структура только с одним элементом, которая задает фильтр по полю. Ключ структуры описывает имя поля, а значение структуры — значение отбора по этому полю.
Приведем пример выборки записей регистра накопления за определенный период по заданному товару:
**Нач = Новый МоментВремени(ВыбДатаВремяНачала);**
**Кон = Новый Граница(ВыбДатаВремяОкончания,ВидГраницы.Включая);**
**Отбор = Новый Структура("Товар",ВыбТовар);**
**Выборка = РегистрыНакопления.ОстаткиТоваров.Выбрать(Нач,Кон,Отбор);**
**Кока Выборка.Следующий() Цикл**
**Сообщить(СокрЛП(Выборка.Товар) + " " +**
**СокрЛП(Выборка.Склад) + " " +**
**СокрЛП(Выборка.Количество));**
**КонецЦикла;**

**Регистр накопления и формы**
У регистра накопления могут быть формы для просмотра всех записей регистра или набора записей. У регистра накопления нет формы одной записи регистра.
Для вывода записей регистра в форме необходимо разместить табличное поле и задать ему свойство ТипЗначения.

**Запросы к регистрам накопления**
Основное предназначение регистра накопления — это быстрое получение остатков на любой момент времени или оборотов за заданный период. Для решения этих задач регистр накопления предоставляет три виртуальные таблицы Остатки, Обороты и комбинированную таблицу ОстаткиИОбороты.
Сведения о применимости таблиц-источников для каждого типа регистра приведены в следующей таблице («+» обозначает применимость, «-» обозначает неприменимость):

| **Таблица-источник** | **Регистр остатков** | **Регистр оборотов** |
|---|---|---|
| **Основная** | + | + |
| **Остатки** | + | - |
| **Обороты** | + | + |
| **ОстаткиИОбороты** | + | - |

| | | |
|---|---|---|
| | | |

*Основная таблица регистра накопления*
Основная таблица регистра накопления существует для обоих типов регистров (остатков и оборотов) и предоставляет следующие поля:
**<Имя измерения>**
Набор полей содержит значения измерений регистра. Имена полей соответствуют именам измерений.
**<Имя реквизита>**
Набор полей содержит значения реквизитов регистра.
**<Имя ресурса>**
Набор полей содержит значения ресурсов регистра.
**Активность**
Содержит признак активности записи, т.е. говорит о том, учитываются ли записи при получение итогов регистра.

**ВидДвижения**
Значение системного перечисления ВидДвиженияНакопле-ния, которое определяет вид движения записи — приход или расход.
**Регистратор**
Содержит ссылку на документ-регистратор движения.
**Период**
Содержит период (дату) записи регистра.
**МоментВремени**
Содержит момент времени записи регистра.
**НомерСтроки**
Содержит номер строки, определяемый как порядковый номер записи в наборе по данному регистратору.

*Таблица получения остатков*
Таблица получения остатков существует только у регистра остатков и предоставляет в распоряжение разработчику следующие поля:
**<Имя измерения>**
Набор полей содержит значения измерений регистра. Имена полей соответствуют именам измерений.
**<Имя ресурса>Остаток**
Набор полей содержит остатки ресурсов регистра. Имена полей соответствуют именам ресурсов с добавлением слова Остаток.
В языке запросов есть возможность получить не все имеющиеся остатки регистра, которых может быть очень много, а только остатки на заданный момент времени с возможностью отбора по любому измерению.
Для этого у таблицы получения остатков имеются следующие параметры:
**Период**
Указывается дата или момент времени, на который рассчитываются остатки. Если параметр не задан, итоги рассчитываются по самую последнюю запись.
**Условие**
Указывается условие на языке запросов, которое будет использовано для ограничения состава записей, по которым будут выбираться итоги. Если параметр не задан, анализируются все активные записи регистра.
Например, следующий запрос получает остатки конкретного товара на заданную дату на каждом складе:

*Таблица получения оборотов*

Таблица получения оборотов существует у регистра остатков и регистра оборотов. Эта таблица включает следующие поля:

**<Имя измерения>**

Набор полей содержит значения измерений регистра. Имена полей соответствуют именам измерений.

**<Имя ресурса>Оборот**

Набор полей содержит обороты по ресурсам регистра. Имена полей соответствуют именам ресурсов с добавлением слова Оборот. Для оборотных регистров оборот подсчитывается как сумма всех движений. Для регистров остатков оборот подсчитывается как сумма всех движений Приход со знаком + (плюс) и Расход со знаком - (минус).

**<Имя ресурса>Приход**

Набор полей содержит суммы всех движений типа «приход» по ресурсам регистра. Имена полей соответствуют именам ресурсов с добавлением слова Приход. Это поле существует только для регистров остатков.

**<Имя ресурса>Расход**

Набор полей содержит суммы всех движений типа «расход» по ресурсам регистра. Имена полей соответствуют именам ресурсов с добавлением слова Расход. Это поле существует только для регистров остатков.

**Период**

Содержит начальную дату и время периода, к которому относится оборот регистра. Существует в случае, если только используется разворот по периодам, т.е. параметр Периодичность имеет какое-либо значение, например, Год, Квартал, Запись и т.д.

**Регистратор**

Содержит ссылку на документ-регистратор движения. Существует только в случае, если параметр Периодичность имеет значение Регистратор или Запись.

**НомерСтроки**

Содержит номер строки, определяемый как порядковый номер записи в наборе записей. Это поле существует, если только параметр Периодичность имеет значение Запись.

Напомним, что для регистра остатков колонка <ИмяРесурса>Оборот рассчитывается как разность между колонками <ИмяРесурса>Приход и <ИмяРесурса>Расход.

Таблица получения оборотов для регистра накопления имеет следующие параметры вызова:

**Начало периода**

Указывается начало периода расчета итогов.

**Конец периода**

Указывается конец периода расчета итогов.

**Периодичность**

Указывается дополнительный разворот итогов по периодичности. Задается один из следующих вариантов: Период (не разворачивать), Регистратор, День, Неделя, Декада, Месяц, Квартал, Полугодие, Год.

**Условие**

Указывается условие на языке запросов. Условие может оперировать полями регистра накопления. Оно будет использовано для ограничения состава записей, по которым будут выбираться обороты. То есть условие будет применяться к исходным записям, а не к уже отобранным. Если параметр не задан, анализируются все активные записи регистра.

*Таблица получения остатков и оборотов*

В языке запросов системы 1С:Предприятие 8.0 существует возможность получения комбинированной информации по остаткам и оборотам. Для этого предназначена таблица-источник ОстаткиИОбороты, которая существует у регистра остатков и регистра оборотов. Таблица ОстаткиИОбороты предоставляет следующие поля:

- <Имя измерения>,
- <Имя ресурса>НачальныйОстаток,

- <Имя ресурса>КонечныйОстаток,
- <Имя ресурса>Оборот,
- <Имя ресурса>Приход, <Имя ресурса> Расход,
- Период (поле существует только, если в параметрах указана периодичность, не равная Период),
- Регистратор (если периодичность Регистратор или Запись),
- НомерСтроки (если периодичность Запись).

При вызове данной таблицы есть возможность указать следующие параметры:
• Начало периода,
• Конец периода,
• Периодичность,
• Метод дополнения,
• Условие.

Нуждается в пояснении параметр «Метод дополнения». Он задает один из следующих вариантов:

**Движения**

В этом случае будут выданы те периоды, в которых были движения.

**ДвиженияИГраницыПериода** (по умолчанию)

В этом случае выбираются периоды, в которых были движения, а также периоды на начало и конец заданного интервала.

Пример обращения к этой таблице приведен ниже.

**ВЫБРАТЬ**
**Номенклатура, Склад, Период,**
**ВЫРАЗИТЬ(КоличествоНачальныйОстаток КАК ЧИСЛО(15,3)) НачОстаток,**
**ВЫРАЗИТЬ(КоличествоПриход КАК ЧИСЛО(15,3)) Приход,**
**ВЫРАЗИТЬ(КоличествоРаоход КАК ЧИСЛО(15,3)) Расход,**
**ВЫРАЗИТЬ(КоличествоКонечныйОстаток КАК ЧИСЛО(15,3)) КонОстаток**
**ИЗ РегистрНакопления.УчетНоменклатуры.ОстаткиИОбороты(**
**&НачДата,ЬКонДата,Месяц,,Номенклатура=&ВыбТовар)**
**ИТОГИ**
**Сумма(НачОстаток),**
**Сумма(Приход),**
**Сумма(Расход),**
**Сумма(КонОстаток) ПО ОБЩИЕ**

Этот запрос показывает остатки и обороты для конкретной номенклатуры по всем складам с разворотом по месяцам. Дополнительно подсчитываются общие итоги, что потребовало приведения данных к числовому типу с помощью конструкции ВЫРАЗИТЬ.

**План видов характеристик**.

Механизм описания характеристик позволяет организовать хранение свойств объектов (справочников, документов и т.д.), которые еще не известны на момент разработки прикладного решения. Таким образом, например, для номенклатуры пользователь сможет самостоятельно вводить новые свойства: цвет, размер, габариты, мощность и т.д. Для каждой группы номенклатуры может быть создан свой набор свойств: для холодильников - объем морозильной камеры, число компрессоров, уровень шума; для компьютеров - объем оперативной памяти, объем жесткого диска; для одежды - размер, рост, цвет.

В дальнейшем на основе этих характеристик можно строить отчеты, анализировать объемы продаж, получать другую информацию для принятия решений.

Задача описания характеристик состоит из двух этапов: создания характеристик и хранения значений созданных характеристик. Например, чтобы указать, что конкретная модель одежды имеет 48 размер, 5 рост и черный цвет, сначала следует создать (если они еще не созданы в прикладном решении) следующие характеристики:
размер, которая будет иметь тип значения Число;
рост, которая будет иметь тип значения Число;
цвет, которая будет иметь тип значения СправочникЦвета.

После того, как нужные характеристики созданы, можно уже указать их конкретные значения для выбранной номенклатуры:
размер = 48;
рост = 5;
цвет = Черный.

Для реализации каждого из описанных этапов используются различные объекты прикладного решения: создание и хранение перечня характеристик, которые могут использоваться в прикладном решении, выполняется с помощью объектов План видов характеристик. А для хранения значений конкретных характеристик, указанных для некоторого объекта прикладного решения, используются регистры сведений.

**Структура плана видов характеристик**

По своей структуре план видов характеристик напоминает справочник: в плане видов характеристик хранятся элементы, - характеристики, которые могут иметь некоторый набор реквизитов и табличных частей и, также как и элементы справочника, могут образовывать иерархические структуры. В плане видов характеристик могут существовать предопределенные характеристики, заданные разработчиком.

Однако основное назначение плана видов характеристик заключается в том, чтобы для каждой характеристики хранить тип значения, который она может принимать:

Возможный перечень типов, которые могут принимать характеристики, указывается разработчиком в процессе создания прикладного решения:
Создавая новые характеристики (или редактируя существующие), пользователь сможет выбрать для них один из типов, входящих в этот перечень.

Однако не исключена ситуация, когда для создания очередной характеристики пользователю понадобится тип, не существующий в прикладном решении. Например, пользователь решит создать характеристику Запах, которая должна иметь значения справочника Запахи, но такого

справочника в прикладном решении нет.

Специально для таких ситуаций разработчик может создать специальный пустой справочник, и указать, что в нем будут храниться дополнительные значения характеристик:
Теперь, создавая характеристику Запах, пользователь сможет выбрать для нее тип значения этого специального справочника, а в самом справочнике создать нужные ему значения: Сладкий, Резкий, Кислый и т.д.

**Формы плана видов характеристик**

Для того чтобы пользователь мог просматривать и изменять данные, содержащиеся в плане видов характеристик, система поддерживает несколько форм его представления. Система может автоматически генерировать все нужные формы плана видов характеристик. Наряду с этим разработчик имеет возможность создать собственные формы, которые система будет использовать вместо форм по умолчанию:

Для просмотра данных, содержащихся в плане видов характеристик, используется форма списка. Она позволяет выполнять навигацию по плану, добавлять, помечать на удаление и удалять характеристики и группы характеристик, перемещать характеристики и группы. Форма списка может представлять данные в иерархическом и не иерархическом виде и позволяет выполнять сортировку и отбор отображаемой информации по нескольким критериям:
Для просмотра и изменения данных отдельных характеристик используется форма элемента. Как правило, она представляет данные в удобном для восприятия и редактирования виде:
Кроме этого для планов видов характеристик, также как и для справочников, поддерживаются формы группы, выбора и выбора группы.

**Хранение значений характеристик**

Для хранения значений характеристик, которые задает пользователь для конкретных объектов прикладного решения, можно использовать регистры сведений.

Например, для того, чтобы хранить значения характеристик номенклатуры, можно использовать регистр сведений, измерениями которого являются номенклатура и характеристика, а ресурсом - значение характеристики.

**Запросы.Служебная конструкция запроса. Выборка, наложение условия, соединение. Внутреннее, внешнее, левое и правое соединение таблиц. Виртуальные таблицы — особенности использования параметров виртуальных таблиц.**

Запросы в системе 1С:Предприятие 8.0 предназначены для выборки информации из базы данных. По сути, запрос - это обращение к системе с просьбой выбрать определенную информацию из базы данных, а часто не только выбрать, но и произвести некоторую обработку: сгруппировать, отсортировать, вычислить итоги.

Например, с помощью запроса можно легко выбрать всех сотрудников, занимающих определенную должность, или можно узнать объем продаж каждого товара в течение года с детализацией до месяца.

В системе 1С:Предприятие 8.0 запросы предназначены только для чтения данных. Изменять данные в 1С с помощью запросов нельзя, и тем более нельзя изменять структуру базы данных, что возможно только в режиме «Конфигуратор».

Основная цель обычного языка программирования заключается в том, чтобы задать пошаговую программу для системы, то есть точно определить, *как* решать задачу. Цель же языка запросов состоит в том, чтобы сказать системе, *что* нужно получить, то есть потребовать результат. При этом нас не интересует, как именно система будет выполнять задание, т.е. план (алгоритм) выполнения запроса строится системой автоматически.

Рассмотрим пример решения задачи обоими методами: с помощью языка программирования и с помощью запроса. Допустим, необходимо вывести сотрудников с окладом больше 10000 руб.

1-й способ. Используется язык программирования:

```
Выборка = Справочники.Сотрудники.Выбрать{);
Пока Выборка.Следующий() Цикл
Если Выборка.Оклад > 10000 Тогда
Сообщить (Выборка. Наименование -*
КонецЕсли;
имеет оклад " + Выборка.Оклад);
КонецЦикла;
```

2-й способ. Используется механизм запросов:

```
Запрос = Новый Запрос("
ВЫБРАТЬ * ИЗ Справочник.Сотрудники
ГДЕ Оклад > 10000");
Выборка = Запрос.Выполнить().Выбрать(};
Пока Выборка.Следующий() Цикл
Сообщить(Выборка.Наименование + " имеет оклад *
КонецЦикла;
Выборка. Оклад ),
```

Обратите внимание на особенности второго варианта с использованием запроса:

• внутри цикла не нужно фильтровать записи, поскольку запрос выполнил всю работу за нас;

• вариант с запросом обычно выполняется быстрее, так как в клиент-серверном варианте работы запрос выполняется на сервере и не требуется передавать по сети весь справочник, который может быть очень большим. В клиент-серверном варианте работы 1С:Предприятия 8.0 запросы будут транслироваться в SQL для выполнения в среде MS SQL Server. Сервер баз данных предпримет необходимые действия для оптимизации запроса.

Даже если вы работает в файловом варианте, то вариант с запросом обычно эффективнее обычного цикла для обработки больших справочников, списков документов и извлечения данных из регистров.

Но необходимо иметь в виду, что при использовании запроса, результат целиком помещается в память, тогда как выборка, сформированная средствами встроенного языка, загружает информацию порциями и позволяет перебрать большие списки объектов, не требуя значительного объема памяти.

Написание запроса требует не *алгоритмического, а декларативного* типа мышления, когда разработчик говорит не *как* сделать задачу, а *что* нужно получить в результате. При этом производится обработка не одной переменной или одной текущей строки, а всей таблицы или столбца. Разработчик должен мыслить множествами, чтобы решить поставленную задачу с помощью одного или нескольких запросов.

Заметим, что в приведенном выше примере мы рассмотрели самое простое применение механизма запросов. С помощью запросов можно не только отбирать нужные записи по любому условию, но также группировать, сортировать, подсчитывать итоги.

Для выполнения запроса необходимо составить текст на специальном *языке запросов,* который сильно отличается от обычного языка программирования по синтаксису и назначению.

Язык запросов системы 1С:Предприятие 8.0 очень похож на стандартный SQL*, но имеет некоторые отличия. Все ключевые слова языка запросов 1С:Предприятие 8.0 имеют английские эквиваленты, совпадающие там, где это возможно, со стандартным SQL. Например, ключевое слово ВЫБРАТЬ может быть заменено на SELECT, а ключевое слово ИЗ на FROM.

В общем случае структура запроса строится по следующей схеме:

**ВЫБРАТЬ <Список полей | *>**

**[ИЗ <Список таблиц-источников:»]**

**[ГДЕ <Список условий>]**

**[УПОРЯДОЧИТЬ ПО <Список полей > ]**

**[СГРУППИРОВАТЬ ПО <Список полей>]**

**[ИТОГИ ПО <Список полей>]**

Рассмотрим основные конструкции языка запросов.

## ВЫБРАТЬ / SELECT

Предложение ВЫБРАТЬ позволяет указать список полей для выборки. Если вместо перечня полей указана звездочка («*»), тогда это означает, что нужно выбрать все поля таблицы. Указание конкретных полей позволяет выбрать только заданные колонки из исходной таблицы-источника запроса. В качестве источника данных для запроса можно использовать справочники, документы, журналы документов, регистры и другие таблицы-источники.

Примеры:

*ВЫБРАТЬ Наименование, Цена ИЗ Справочник.Товары*

*ВЫБРАТЬ * ИЗ Справочник.Сотрудники*

*ВЫБРАТЬ Номер, Дата, Представление ИЗ Документ.РасходнаяНакладная*

*ВЫБРАТЬ * ИЗ РегистрНакопления.Продажи*

В системе 1С:Предприятие 8.0 можно построить запрос без указания ключевого слова ИЗ, тогда список полей должен содержать *полные* имена таблиц, например : *ВЫБРАТЬ Справочник.Сотрудники.**

### ПСЕВДОНИМЫ ПОЛЕЙ (КАК/ AS)

Для поля может быть назначен псевдоним с помощью ключевого слова КАК. Это позволяет обращаться к полю по псевдониму при указании итогов и порядка сортировки, а также в результате запроса. В следующем запросе для полей Наименование и ЕдИзм назначаются псевдонимы:

```
ВЫБРАТЬ
Наименование КАК Товар,
Цена,
ЕдИзм КАК ЕдиницаИзмерения
ИЗ Справочник.Товары
УПОРЯДОЧИТЬ ПО Товар
```

Ключевое слово КАК необязательно и может быть опущено, но для повышения наглядности его рекомендуется указывать, особенно на первых порах.

```
ВЫБРАТЬ Наименование Товар,
Цена,
ЕдИзм ЕдиницаИзмерения
ИЗ Справочник.Товары
УПОРЯДОЧИТЬ ПО Товар
```

В языке запросов есть возможность указать псевдоним для таблицы-источника с помощью ключевого слова КАК. При этом можно обращаться к источнику через псевдоним, например, в списке полей конструкции ВЫБРАТЬ или при соединении таблиц, которое будет описано ниже.

Ниже приведен пример использования псевдонима таблицы-источника:

```
ВЫБРАТЬ Спр.Наименование,
Спр.Цена,
Спр.Страна ИЗ Справочник.Номенклатура КАК Спр
```

## ИЗ / FROM

Предложение ИЗ позволяет указать таблицы-источники для запроса и задать порядок их соединения, если таблиц несколько. Вот простые примеры запросов с предложением ИЗ:

```
ВЫБРАТЬ * ИЗ Справочник.Товары
ВЫБРАТЬ * ИЗ Документ.РасходнаяНакладная
```

Конструкция ИЗ необязательна, если в списке выбираемых полей указаны полные имена таблиц, например,

```
ВЫБРАТЬ Справочник.Товары.*
ВЫБРАТЬ Документ.РасходнаяНакладная.Дата, Документ.РасходнаяНакладная.Номер
```

Важной возможностью языка запросов системы 1С:Предприятие 8 является обращение сразу к нескольким таблицам. При этом их можно соединять определенным образом.

Например, необходимо выбрать все проданные товары и вывести их в отчет с указанием группы, к которой они относятся. Это делает представленный ниже запрос:

```
ВЫБРАТЬ Док.Номенклатура,
Спр.ЗакупочнаяЦена КАК Цена, Спр.Родитель КАК Группа
ИЗ Документ.РасходнаяНакладная.Состав КАК Док СОЕДИНЕНИЕ Справочник. Номенклатура КАК Спр ПО
Док.Номенклатура = Спр.Ссылка
```

Результат этого запроса показан ниже:

| Номенклатура | Цена | Группа |
|---|---|---|
| 1С:Бухгалтерия 7.7 | 35 | Программы |
| Клавиатура Keyboard PS/2 | 3 | Клавиатуры |
| Монитор 15" LG | 134,5 | Мониторы |
| Мышь 2-кноп A4Tech PS/2 | 1,2 | Мыши |
| Мышь LOGITECH M-S48 | 0,8 | Мыши |

В данном примере того же эффекта можно добиться, если просто обращаться к имени поля через точку, что называется *разыменованием ссылочных полей.* При этом соединение таблиц производится неявно.

Следующий запрос эквивалентен предыдущему и использует разыменование полей:

```
ВЫБРАТЬ Номенклатура,
Номенклатура.ЗакупочнаяЦена КАК Цена,
Номенклатура.Родитель КАК Группа ИЗ Документ.РасходнаяНакладная.Состав
```

Возможность разыменования полей в 1С:Предприятии 8.0 допускает обращение к свойствам объектов через несколько точек, например, «Номенклатура.Поставщик.Страна». Это позволяет значительно упростить написание запросов.

Рассмотренное соединение относится к классу внутренних. В языке запросов системы 1С:Предприятие 8.0 существует возможность *внешних* соединений, которые могут быть *левыми* (LEFT OUTER), *правыми* (RIGHT OUTER) и *полными* (FULL OUTER).

# Предложение ГДЕ / WHERE

Предложение ГДЕ позволяет задать *условие отбора* данных из исходных таблиц-источников запроса. В запросе будут выбраны только те записи, для которых выполняется заданное условие.

Например, выберем товары с ценой, большей или равной определенному значению:

```
ВЫБРАТЬ Наименование, ЗакупочнаяЦена КАК Цена ИЗ Справочник.Номенклатура ГДЕ ЗакупочнаяЦена >= 1300
```

# Предложение УПОРЯДОЧИТЬ ПО / ORDER BY

Часто результат запроса требуется отсортировать по алфавиту или по числовому полю. В общем случае в качестве значения упорядочивания может быть выражение.

Например, представим список товаров, упорядоченный по алфавиту. При сортировке сначала идут товары, начинающиеся на цифры, затем на английские буквы, а затем на русские буквы:

```
ВЫБРАТЬ Код, Наименование ИЗ Справочник.Номенклатура УПОРЯДОЧИТЬ ПО Наименование ВОЗР
```

| Код | Наименование |
|---|---|
| 00014 | 1С:Бухгалтерия ПРОФ версия 7.7 |
| 00016 | 1С:Торговля и Склад 7.7 Проф |
| 00009 | Windows XP Home Edition Russian CD |
| 00010 | Windows XP Home Edition Russian UPG CD |
| 00011 | Windows XP Professional Russian CD |
| 00041 | Доставка |
| 00042 | Инсталляция ПО |
| 00018 | Клавиатура Apple Pro Keyboards |

Сортировка часто применяется с ключевым словом ПЕРВЫЕ. Например, следующий запрос сортирует товары по убыванию цены и показывает 5 самых дорогих товаров:

```
ВЫБРАТЬ ПЕРВЫЕ 5 Код, Наименование, ЗакупочнаяЦеяа КАК Цена ИЗ Справочник.Номенклатура
упорядочить по цена убыв
```

Результат запроса показан в таблице:

| Код | Наименование | Цена |
|---|---|---|
| 00035 | Ноутбук Rover Computers Explorer | 1326 |
| 00039 | Сист. блок IBM NetVista M41 | 1 222 |
| 00034 | Ноутбук Rover Computers Navigator KT7 | 1 118 |
| 00038 | Сист. блок IBM NetVista A22p | 1 111 |
| 00032 | Лазерный принтер HP LaserJet 2200 | 720 |

# Агрегатные функции в запросе

Часто требуется не просто выбрать отдельные записи из базы данных, а получить сводную информацию, например, для ответа на следующие вопросы:

- Каков общий объем продаж за период?
- Какова средняя стоимость заказа в каждом филиале?
- Сколько сотрудников работает в штате?
- Какова наименьшая и наибольшая цена продажи каждого товара?

В системе 1С:Предприятие 8.0 такие запросы можно создавать с помощью агрегатных функций, группировок и предложения ИМЕЮЩИЕ (HAVING). Ниже будут описаны агрегатные функции языка запросов.

Любая агрегатная функция принимает в качестве аргумента какой-либо столбец, а возвращает единственное значение. Например, агрегатная функция СУММА (SUM) принимает в качестве аргумента столбец чисел и вычисляет его сумму.

В языке запросов 1С:Предприятия 8.0 существуют следующие агрегатные функции:

**СУММА (SUM)**

Вычисляет сумму всех значений, содержащихся в столбце.

**МАКСИМУМ (MAX)**

Находит наибольшее значение в столбце.

**МИНИМУМ (MIN)**

Находит наименьшее значение в столбце.

**СРЕДНЕЕ (AVG)**

Вычисляет среднее арифметическое значение по столбцу.

**КОЛИЧЕСТВО (COUNT)**

Подсчитывает количество значений, содержащихся в столбце. Если в качестве параметра данной функции передать звездочку («*»), то функция подсчитает количество строк в таблице результата запроса.

Ниже приведен пример запроса с несколькими агрегатными функциями:

```
ВЫБРАТЬ
СУММА(Оклад) КАК ФондОплатыТруда,
МИНИМУМ(Оклад! КАК МинОклад,
МАКСИМУМ(Оклад) КАК МаксОклад,
СРЕДНЕЕ(Оклад) КАК СреднийОклад,
КОЛИЧЕСТВО(*) КАК Количество ИЗ Справочник.Сотрудники
```

Результат запроса будет содержать всего одну строку:

| **ФондОплаты Труда** | **МинОклад** | **Макс Оклад** | **Средний Оклад** | **Коли-чество** |
|---|---|---|---|---|
| **1500000** | **6000** | **17000** | **9000** | **30** |

Рассмотрим более подробно функцию КОЛИЧЕСТВО / COUNT. Эта функция подсчитывает количество значений параметра, попавших в выборку.

В отличие от других агрегатных функций она допускает три варианта использования:

• Позволяет узнать количество *строк* в результате запроса. Для этого в качестве параметра функции надо указать звездочку («*»). Это наиболее часто встречающийся вариант использования функции КОЛИЧЕСТВО. Даже если в строке все поля содержат NULL, то такая строка тоже будет посчитана.

• Позволяет подсчитать количество *значений* указанного поля, не являющихся NULL-значениями. В качестве параметра функции можно указывать ссылки на поля, содержащие значения любого типа, при этом NULL-значения игнорируются.

• Позволяет узнать количество *различных значений* указанного поля. Для этого перед спецификацией поля надо указывать ключевое слово РАЗЛИЧНЫЕ / DISTINCT, при этом NULL-значения игнорируются.

Например, с помощью функции КОЛИЧЕСТВО можно ответить на следующие вопросы:

• Сколько сотрудников, у которых оклад больше заданной величины?

```
ВЫБРАТЬ КОЛИЧЕСТВО)*) КАК Количество ИЗ Справочник.Сотрудники ГДЕ Оклад > &ВыбОклад
```

• Сколько различных клиентов купили хоть что-нибудь за заданный период?

```
ВЫБРАТЬ КОЛИЧЕСТВО(РАЗЛИЧНЫЕ Контрагент) КАК Количество ИЗ Документ.РасходнаяНакладная ГДЕ Дата
МЕЖДУ &НачДата И &КонДата
```

# Предложение СГРУППИРОВАТЬ ПО / GROUP BY

Очень часто запрос делается с целью не просто выбрать записи из таблицы, но также сгруппировать их определенным образом. Под словом «сгруппировать» имеется в виду не распределить записи по группам, а *свернуть* по группировочным полям, вычислив агрегатные функции по каждой группе.

Например, если требуется узнать объем продаж каждого товара за период, тогда в запросе понадобится группировка по товару. Ниже приведен запрос с группировкой к документам Расходная-Накладная:

```
ВЫБРАТЬ Номенклатура, СУММА(Сумма) КАК ОбъемПродаж ИЗ Документ.РасходнаяНакладная.Состав КАК
ДокСостав ГДЕ ДокСостав.Ссылка.Дата МЕЖДУ &НачДата И &КонДата СГРУППИРОВАТЬ ПО Номенклатура
АВТОУПОРЯДОЧИВАНИЕ
```

В данном примере использована агрегатная функция СУММА для поля Сумма табличной части Состав. Запрос группирует все продажи по товарам и подсчитывает объем продаж по каждому това-

| Номенклатура | ОбъемПродаж |
|---|---|

| | |
|---|---|
| 1С:Торговля и Склад 7.7 Проф | 1540 |
| Windows XP Home Edition Russian CD | 1 360 |
| Windows XP Home Edition Russian UPG CD | 1 105 |
| Windows XP Professional Russian CD | 2480 |
| Доставка | 40 |
| Инсталляция ПО | 60 |
| Клавиатура Apple Pro Keyboards | 5890 |
| Клавиатура Keyboard PS/2 | 384 |

Таким образом, в большинстве случаев группировки используются совместно с агрегатными функциями. Если взять пример из предыдущего параграфа и добавить группировку по подразделению, то можно легко детализировать информацию до подразделений и получить ценную информацию для анализа:

```
ВЫБРАТЬ
Подразделение,
СУММА(Оклад) КАК ФондОплатыТруда
МИНИМУМ(Оклад) КАК МинОклад,
МАКСИМУМ(Оклад) КАК МаксОклад,
СРЕДНЕЕ(Оклад) КАК -СреднийОклад,
КОЛИЧЕСТВО)*) КАК КоличествоЧеловек ИЗ Справочник.Сотрудники СГРУППИРОВАТЬ ПО Подразделение
```

| **Подразделение** | **Фонд Оплаты Труда** | **Мин Оклад** | **Макс Оклад** | **Средний Оклад** | **Коли-чество Человек** |
|---|---|---|---|---|---|
| **Бухгалтерия** | **30000** | **6000** | **16000** | **10000** | **9** |

| | | | | | |
|---|---|---|---|---|---|
| **Маркетинг** | **40000** | **6500** | **15000** | **11000** | **7** |

| **Подразделение** | **Фонд Оплаты Труда** | **Мин Оклад** | **Макс Оклад** | **Средний Оклад** | **Коли-чество Человек** |
|---|---|---|---|---|---|
| **Снабжение** | **30000** | **7000** | **12000** | **9000** | **8** |
| **Руководство** | **50000** | **9000** | **17000** | **14000** | **6** |

В языке запросов можно группировать данные по нескольким полям, при этом будут подсчитаны агрегатные функции для каждой комбинации группировок:

```
ВЫБРАТЬ Номенклатура,
ДокСостав.Ссылка.Контрагент КАК Контрагент,
СУММА(Сумма) КАК Продажи
ИЗ Документ.РасходнаяНакладная.Состав КАК ДокСостав ГДЕ ДокСостав.Ссылка,Дата МЕЖДУ
&НачДата И &КонДата СГРУППИРОВАТЬ ПО Номенклатура, Контрагент АВТОУПОРЯДОЧИВАНИЕ
```

# Предложение ИТОГИ / TOTALS

Язык запросов системы 1С:Предприятие 8.0 имеет очень мощную возможность расчета итогов, чего нет в стандартном языке SQL. Данный механизм дает возможность включить в результат запроса дополнительные строки, содержащие общие и промежуточные итоги по заданным полям и группировкам.

## *Общие итоги*

Рассмотрим сначала общие итоги, как более простые для понимания. Следующий запрос выбирает из регистра накопления Продажи все записи за заданный период и рассчитывает общий объем продаж:

```
ВЫБРАТЬ Номенклатура, Сумма ИЗ РегистрНакопления.Продажи ИТОГИ СУММА(Сумма) ПО Общие
```

В результате запроса появляется дополнительная итоговая строка:

| Номенклатура | Сумма |
|---|---|
| | 445 |
| 1С:Бухгалтерия 7.7 Базовая версия | 140 |
| 1С:Бухгалтерия 7.7 Стандартная версия | 280 |
| Мышь OK-720 Mouse A4Tech PS/2 | 3 |

| | |
|---|---|
| Клавиатура Keyboard PS/2 | 22 |

При обходе результата запроса итоговые строки можно отличить от обычных с помощью метода ТипЗаписи().

Например, выведем результат данного запроса в окно сообщений, а итоговую строку выделим заглавными буквами:

```
Запрос = Новый Запрос(" I ВЫБРАТЬ Номенклатура, Сумма I ИЗ РегистрНакопления.Продажи ИТОГИ
СУММА(Сумма) ПО Общие");

Выборка = Запрос.Выполнить О.Выбрать (); Пока Выборка.Следующий() Цикл

Если Выборка.ТипЗаписи() = ТипЗаписиЗапроса.ОбщийИтог Тогда

Сообщить("ОБЩИЙ ИТОГ: " + Выборка.СуммаПродажи); Иначе

Сообщить("Товар/услуга: " + Выборка.Номенклатура +

"Сумма: " + Выборка.СуммаПродажи); КонецЕсли; КонецЦикла;
```

## *Итоги по группировкам*

При расчете *итогов по группировкам* вычисляются значения агрегатных функций по выборкам с одинаковыми значениями полей, по которым производится группировка.

Например, в следующей таблице значения группировок выделены жирным шрифтом:

**Номенклатура**

**Вилы**

**Грабли**

Грабли

Грабли

**Лопата**

Лопата

Простой запрос с итогами по группировкам выглядит следующим образом:

```
ВЫБРАТЬ Номенклатура, Период, Сумма ИЗ РегистрНакопления.Продажи ИТОГИ СУММА(Сумма) ПО Номенклатура
АВТОУПОРЯДОЧИВАНИЕ
```

Результат запроса включает в себя обычные записи из регистра и итоги по каждому товару:

| Товар | Период | Сумма |
|---|---|---|
| 1С:Аспект 7.7 | | 720 |
| 1С: Аспект 7.7 | 11.01.200221:56:07 | 90 |
| 1С: Аспект 7.7 | 24.02.2002 12:00:00 | 180 |
| 1С: Аспект 7.7 | 26.02.2002 12:00:00 | 180 |
| 1С: Аспект 7.7 | 04.08.2002 12:00:00 | 270 |

| | | |
|---|---|---|
| 1С:Бухгалтерия 7.7 | | 350 |
| 1 С:Бухгалтерия 7.7 | 10.01.2002 12:00:01 | 140 |
| 1С:Бухгалтерия 7.7 | 04.08.2002 12:00:00 | 210 |

Обратите внимание, что запрос с итогами по группировкам отличается от обычной группировки (свертки) с помощью предложения СГРУППИРОВАТЬ ПО с агрегатными функциями. В последнем случае в результат запроса не включаются исходные записи, а остаются только итоговые строки.

Например, следующий запрос выводит объем продаж, сгруппированный по каждой номенклатуре, и не выводит детальные записи регистра накопления:

```
ВЫБРАТЬ Номенклатура, СУММА(Сумма) КАК Сумма ИЗ РегистрНакопления.Продажи СГРУППИРОВАТЬ ПО
Номенклатура АВТОУПОРЯДОЧИВАНИЕ
```

| Номенклатура | Сумма |
|---|---|
| 1С:Аспект 7.7 | 720 |
| 1С:Бухгалтерия 7.7 Базовая версия | 350 |
| 1С:Бухгалтерия 7.7 Стандартная версия | 280 |
| 1С:Бухгалтерия ПРОФ версия 7.7 | 1320 |
| 1С:Торговля и Склад 7.7 Проф | 1 540 |

Можно рассчитать итоги по комбинации группировок, перечислив их через запятую:

```
ВЫБРАТЬ Контрагент, Номенклатура, Сумма ИЗ РегистрНакопления.Продажи ИТОГИ СУММА(Сумма)
ПО Контрагент, Номенклатура АВТОУПОРЯДОЧИВАНИЕ
```

В данном запросе будет рассчитан объем продаж по каждой комбинации подразделения и номенклатуры. Кроме того, в запрос будут включены исходные записи из регистра накоплений:

| Контрагент | Номенклатура | Сумма |
|---|---|---|
| Алекс-2002 | | 780 |

| | | |
|---|---|---|
| Алекс-2002 | 1С:Аспект 7.7 | 270 |
| Алекс-2002 | 1С: Аспект 7.7 | 90 |
| Алекс-2002 | 1С: Аспект 7.7 | 180 |
| Алекс-2002 | Windows XP Home | 510 |
| Алекс-2002 | Windows XP Home | 170 |
| Алекс-2002 | Windows XP Home | 340 |
| Эльбрус | | 85 |
| Эльбрус | 1С:Аспект 7.7 | 85 |
| Эльбрус | 1С: Аспект 7.7 | 20 |
| Эльбрус | 1С: Аспект 7.7 | 25 |
| Эльбрус | 1С:Аспект 7.7 | 40 |

Порядок группировок в тексте запроса имеет довольно важное значение, поскольку сначала рассчитываются итоги по первому группировочному полю, затем по второму и т.д.

Если в предыдущем примере поменять местами контрагента и номенклатуру, то получим следующий результат:

```
ВЫБРАТЬ Контрагент, Номенклатура, Сумма ИЗ РегистрНакопления.Продажи ИТОГИ СУММА(Сумма)
ПО Номенклатура, Контрагент
АВТОУПОРЯДОЧИВАНИЕ
```

Результат запроса показан ниже:

| Номенклатура | Контрагент | Сумма |
|---|---|---|
| 1С:Аспект 7.7 | | 355 |

| | | |
|---|---|---|
| 1С: Аспект 7.7 | Алекс-2002 | 270 |
| 1С: Аспект 7.7 | Алекс-2002 | 90 |
| 1С: Аспект 7.7 | Алекс-2002 | 180 |
| 1С:Аспект 7.7 | Эльбрус | 85 |
| 1С:Аспект 7.7 | Эльбрус | 20 |
| 1С:Аспект 7.7 | Эльбрус | 25 |
| 1С: Аспект 7.7 | Эльбрус | 40 |
| Windows XP Home Edition | | 510 |
| Windows XP Home Edition | Алекс-2002 | 510 |
| Windows XP Home Edition | Алекс-2002 | 170 |
| Windows XP Home Edition | Алекс-2002 | 340 |

В 1С:Предприятии 8.0 реализована возможность обхода группировок в произвольном порядке. То есть из одного результата запроса возможно получать отчеты с различной последовательностью группировок. Например, в отчете на основе одного результата запроса могут выводиться сначала группировки по товарам, а затем по поставщикам.

## *Итоги по иерархии*

Если группировочное поле является ссылкой на справочник, то для расчета итогов по группам справочника (или родительским элементам, если справочник состоит из одних элементов) необходимо указать ключевое слово ИЕРАРХИЯ. В этом случае в результат будут добавлены записи с итогами для уровней иерархии справочника.

Например, выберем записи справочника Номенклатура, рассчитаем объем продаж по каждой позиции и по группам справочника:

```
ВЫБРАТЬ Номенклатура, Сумма
```

```
ИЗ РегистрНакопления.Продажи
ИТОГИ СУММА(Сумма) ПО Номенклатура ИЕРАРХИЯ
АВТОУПОРЯДОЧИВАНИЕ
```

Заметьте, что в результат запроса включены группы справочника, и по ним подсчитаны итоги:

| Номенклатура | Сумма |
|---|---|
| Клавиатуры | 675 |
| Клавиатура Apple Pro Keyboards | 600 |
| Клавиатура Apple Pro Keyboards | 225 |
| Клавиатура Apple Pro Keyboards | 375 |
| Клавиатура Keyboard PS/2 | 75 |
| Клавиатура Keyboard PS/2 | 25 |
| Клавиатура Keyboard PS/2 | 50 |
| Мониторы | 1460 |
| Монитор 15' LG Studioworks 575N | 620 |
| Монитор 15' LG Studioworks 575N | 465 |
| Монитор 15' LG Studioworks 575N | 155 |
| Монитор 17' Philips 107S20 | 840 |
| Монитор 17' Philips 107S20 | 840 |

## *Несколько итогов в запросе*

В языке запросов допускается совместное использование различных итогов в одном запросе, например, общих, иерархических и итогов по группировкам.

255

Разработка в системе 1С:Предприятие 8.0

Ниже представлен запрос, который выбирает данные из расходных накладных и дополнительно подсчитывает общие итоги, итоги по контрагентам, по каждому товару и группе товаров.

```
ВЫБРАТЬ
НаклСостав.Ссылка.Контрагент КАК Контрагент,
НаклСостав.Номенклатура КАК Номенклатура,
НаклСостав.Ссылка.Номер,
НаклСостав.Сумма КАК Продажи
ИЗ Документ.РасходнаяНакладная.Состав КАК НакяСостав ИТОГИ СУММА(Продажи) ПО
ОБЩИЕ,
Контрагент,
Номенклатура ИЕРАРХИЯ
```

| **Контрагент** | **Номенклатура** | **Номер** | **Продажи** |
|---|---|---|---|
| | | | **4520** |
| **Автохозяйство** | | | **1280** |
| **Автохозяйство** | **Клавиатуры** | | **20** |
| **Автохозяйство** | **Клавиатура LK-601** | | **20** |
| **Автохозяйство** | **Клавиатура LK-601** | **00023** | **20** |
| **Автохозяйство** | **Мониторы** | | **1260** |
| **Автохозяйство** | **Монитор 17' Philips** | | **210** |
| **Автохозяйство** | **Монитор 17' Philips** | **00023** | **210** |
| **Автохозяйство** | **Монитор 19' Hitachi** | | **1050** |

| | | | |
|---|---|---|---|
| **Автохозяйство** | **Монитор 19' Hitachi** | **00023** | **1 050** |
| **Алекс-2002** | | | **3240** |
| **Алекс-2002** | **Клавиатуры** | | **3240** |
| **Алекс-2002** | **Клавиатура Apple Pro** | | **3225** |
| **Алекс-2002** | **Клавиатура Apple Pro** | **00012** | **3000** |
| **Алекс-2002** | **Клавиатура Apple Pro** | **00013** | **225** |
| **Алекс-2002** | **Клавиатура PS/2** | | **15** |
| **Алекс-2002** | **Клавиатура PS/2** | **00008** | **15** |

# Встроенные функции языка запросов

В языке запросов есть встроенные функции, которые могут быть использованы в списке полей выборки предложения ВЫБРАТЬ и в условии отбора предложения ГДЕ.

Если параметр функции является значением NULL, то возвращается тоже NULL. Для функций работы с датой следует учитывать, что тип «дата» включает в себя дату и время с точностью до секунды.

В языке запросов существуют следующие функции:

**ПОДСТРОКА (<Строка>, <Позиция>, <ЧислоСимволов>)**

Данная функция предназначена для выделения подстроки из строки.

**ГОД** (<дата>)

Выделяет год из даты. Возвращает число от 1 до 9999.

**МЕСЯЦ (<дата>)**

Выделяет месяц из даты (от 1 до 12).

**ДЕНЬ (<дата>)**

Выделяет число из даты (от 1 до 31).

**ЧАС (<дата>)**

Выделяет часы из даты (от 0 до 23).

**МИНУТА (<дата>)**

Выделяет минуты из даты (0 до 59).

**СЕКУНДА (<дата>)**

Выделяет секунды из даты (0 до 59).

**КВАРТАЛ (<дата>)**

Определяет номер квартала по дате (от 1 до 4).

**НЕДЕЛЯ (<дата>)**

Определяет номер недели в году по дате (от 1 до 53).

**ДеньГода (<дата>)**

Определяет номер дня в году по дате (от 1 до 366).

**ДеньНедели (<дата>)**

Определяет номер дня недели по дате (от 1 до 7).

**НачалоПериода ( <дата> ,<ТипПериода> )**

Возвращает начало периода по заданной дате. В качестве параметра ТипПериода передается Год, Месяц, Неделя, День, Час и т.д.

**КонецПериода (<дата>,<ТипПериода>)**

Возвращает конец периода по заданной дате.

Например, следующий запрос разбирает дату приема сотрудника на составляющие без учета времени и выводит первые 2 буквы от его ФИО, используя функцию ПОДСТРОКА:

```
ВЫБРАТЬ Наименование, ДатаПриема
ПОДСТРОКА(Наименование,1,2) КАК Сокращение, ДЕНЬ(ДатаПриема) КАК Число, МЕСЯЦ(ДатаПриема) КАК Месяц,
ГОД(ДатаПриема) КАК Год ИЗ Справочник.Сотрудники
```

| Наименование | ДатаПриема | Сокращение | Число | Месяц | Год |
|---|---|---|---|---|---|
| Иванов Иван Иванович | 25.12.2003 | Ив | 25 | 12 | 2003 |
| Петров Петр Петрович | 05.07.2000 | Пе | 5 | 7 | 2000 |
| Сидоров Иван Николаевич | 15.12.1995 | Си | 15 | 12 | 1995 |

## *Левое внешнее соединение*

Конструкция ЛЕВОЕ [ВНЕШНЕЕ] СОЕДИНЕНИЕ означает, что в результат запроса надо включить комбинации записей из обеих исходных таблиц, которые соответствуют указанному условию. Но, в отличие от внутреннего соединения, в результат запроса надо включить еще и записи из *первого* источника (указанного слева от слова СОЕДИНЕНИЕ), для которых не найдено соответствующих условию записей из второго источника.

Таким образом, в результат запроса будут включены *все записи из первого источника',* они будут соединены с записями из второго источника при выполнении указанного условия. Строки результата запроса, для которых не найдено соответствующих условию записей из второго источника, будут содержать значение NULL в полях, формируемых на основании записей из этого источника.

Обратите внимание, что NULL-значения не являются нулем или пустой строкой. Это специальные маркеры, обозначающие неуказанные (отсутствующие) значения или значения, не имеющие смысла.

Например, нужно показать курсы всех валют, которые хранятся в регистре сведений КурсыВалют. Возможно, что для некоторой валюты не будет найдено соответствующей записи в регистре сведений, но она также должна попасть в отчет (запросы к регистрам сведений и таблица СрезПоследних описаны в главе «Регистры сведений»):

```
ВЫБРАТЬ Спр.Наименование, Per.Курс ИЗ Справочник.Валюты КАК Спр ЛЕВОЕ ВНЕШНЕЕ СОЕДИНЕНИЕ
РегистрСведений.КурсыВалют.СрезПоследних КАК Per ПО Спр.Ссылка = Per.Валюта
```

Результат запроса показан в следующей таблице:

### *Правое внешнее соединение*

Конструкция ПРАВОЕ [ВНЕШНЕЕ] СОЕДИНЕНИЕ означает, что в результат запроса надо включить комбинации записей из обеих исходных таблиц, которые соответствуют указанному условию. Кроме того, в результат запроса надо включить еще и записи из *второго* источника (указанного справа от слова СОЕДИНЕНИЕ), для которых не найдено соответствующих условию записей из первого источника.

Таким образом, в результат запроса будут включены *все записи из второго источника;* они будут соединены с записями из первого источника при выполнении указанного условия. Строки результата запроса, для которых не найдено соответствующих условию записей из первого источника, будут содержать значение NULL в полях, формируемых на основании записей из этого источника.

Разработка в системе 1С:Предприятие 8.0

Правое внешнее соединение полностью аналогично левому, за исключением того, что таблицы поменялись местами. Например, представленный ниже запрос эквивалентен предыдущему, но вместо левого, используется правое внешнее соединение:

```
ВЫБРАТЬ Спр.Наименование, Per.Курс
ИЗ РегистрСведений.КурсыВалют.СрезПоследнихО КАК Per ПРАВОЕ ВНЕШНЕЕ СОЕДИНЕНИЕ Справочник.Валюты КАК
Спр ПО Спр.Ссылка = Per.Валюта
```

### *Полное внешнее соединение*

Конструкция ПОЛНОЕ [ВНЕШНЕЕ] СОЕДИНЕНИЕ означает, что в результат запроса надо включить комбинации записей *из обеих исходных таблиц,* которые соответствуют указанному условию. Кроме того, в результат запроса надо включить также еще и те записи из обоих источников, для которых не найдено соответствий.

Таким образом, в результат запроса будут включены *все записи из обоих источников;* они будут соединены друг с другом при выполнении указанного условия. Строки результата запроса, для которых не найдено соответствующих условию записей из какого-либо источника, будут содержать NULL в полях, формируемых на основании записей из этого источника.

# Обработка результата запроса

Напомним общую схему выполнения запроса:

1. Создание объекта Запрос и передача ему текста запроса.

2. Установка параметров запроса с помощью метода Установить-Параметр.

3. Выполнение запроса, получение результата запроса.

4. Получение выборки из результата запроса или выгрузка результата запроса в таблицу значений / дерево значений. Также есть возможность использовать результат запроса в качестве источника данных для сводной таблицы.

5. Обход выборки из результата запроса или обработка таблицы значений/дерева значений.

Ниже приведен пример выполнения простого запроса и получения выборки из результата запроса:

```
Запрос = Новый Запрос;
ТекстЗапроса = "ВЫБРАТЬ * ИЗ Справочник.Сотрудники";
Запрос.Текст = ТекстЗапроса;
РезультатЗапроса = Запрос . Выполнить () ,-
ВыборкаИзРезультатаЗапроса = РезультатЗапроса.Выбрать О;
```

Тот же самый фрагмент можно записать короче. В приведенном ниже примере используется конструктор объекта Запрос с параметром, через который передается текст запроса. Далее запрос выполняется, и сразу же производится выборка из результата запроса:

```
Запрос = Новый Запрос{"ВЫБРАТЬ * ИЗ Справочник.Сотрудники"); ВыборкаИзРезультатаЗапроса =
Запрос.Выполнить().Выбрать!);
```

Перед получением выборки можно проверить результат запроса на наличие хотя бы одной записи. Для этого предназначен метод Пустой:

```
Запрос = Новый Запрос("ВЫБРАТЬ * ИЗ Справочник.Сотрудники"); Результат =
Запрос.Выполнить!); Если НЕ Результат.Пустой() Тогда
ВыборкаИзРезультатаЗапроса = Результат.Выбрать(); КонецЕсли;
```

Виртуальные таблицы.

Предоставляют разработчику сервис обращения к данным таблиц регистров (накопления, бухгалтерии, расчетов) в сводном и упорядоченном виде. При трансляции запроса вместо одной виртуальной таблицы формируется сложный вопрос, включающий в себя соединение таблицы с самой собой. Важным условием использования виртуальных таблиц является передача внутри вируальной таблицы параметров, накладывающих фильтр выборки данных. Если не передать эти параметры, то время отклика системы на выполнение этого запроса может увеличиться в 10,100 или 1000 раз, в зависимости от размера таблицы.