PL-SQL

4.1. Основные понятия PL-SQL

PL/SQL - это процедурный язык пошагового программирования, инкапсулирующий язык **SQL**. В результате получается хорошо развитый язык программирования третьего поколения (**3GL**), подобный языку **C++**, **Pascal** и т. д. В своей сути **PL/SQL** блочно ориентирован.

PL/SQL имеет строгие правила области видимости переменных, поддерживает параметризованные вызовы процедур и функций и так же унаследовал от языка **ADA** такое средство, как пакеты (**package**).

PL/SQL предусматривает строгий контроль типов, все ошибки несовместимости типов выявляются на этапе компиляции и выполнения. Так же поддерживается явное и неявное преобразование типов.

PL/SQL - поддерживает сложные структуры данных, так же предусмотрена перегрузка подпрограмм, для создания гибкой среды прикладного программирования.

Язык **PL/SQL** - имеет элемент **Exception Handler** (обработчик исключительных ситуаций) для синхронной обработки ошибок на этапе выполнения кода **PL/SQL**.

Так же строго говоря, язык **PL/SQL** не является объектно-ориентированным, хотя имеет некоторые средства для создания и работы с объектами БД на уровне объектно-ориентированных языков программирования. Запутанно, правда? Но тем не менее, это так и есть и в дальнейшем вы в этом убедитесь! :) Что-то очень похожее на ООП в **PL/SQL** имеется.

PL/SQL - является машинно независимым языком программирования! И это действительно так! Например, не нужно изучать **PL/SQL** для **Windows** или для **Unix**! Сам код программных блоков **PL/SQL** не зависит от платформы, на которой они выполняются. Сам язык остается таким же как и он есть!!! Вот вам и мультиплатформенность!

PL/SQL - поддерживает стандартные интерфейсы работы с языками высокого уровня такими как **C**, **C++** - через предкомпиляторы поставляемые фирмой **Oracle**. Например, для работы с языком **C** есть такое средство как **OCI** (**Oracle Call Interface**).

Так же **PL/SQL** имеет ряд встроенных средств для работы с **Internet**, прямо из хранимых процедур. Имеет поддержку создания **HTTP** запросов так же непосредственно из хранимых процедур. Вот, что из себя представляет встроенный язык **PL/SQL** сервера **Oracle**. Однако прежде чем приступать, к изучению **PL/SQL** необходимо сказать несколько слов о его среде выполнения.

В типичной клиент/серверной среде обычно, самое узкое место это - сеть и то как она построена и отлажена вашим администратором сети. Кстати взаимодействие администратора сети и администратора БД, является немаловажным фактором!

Посмотрим на рисунок:

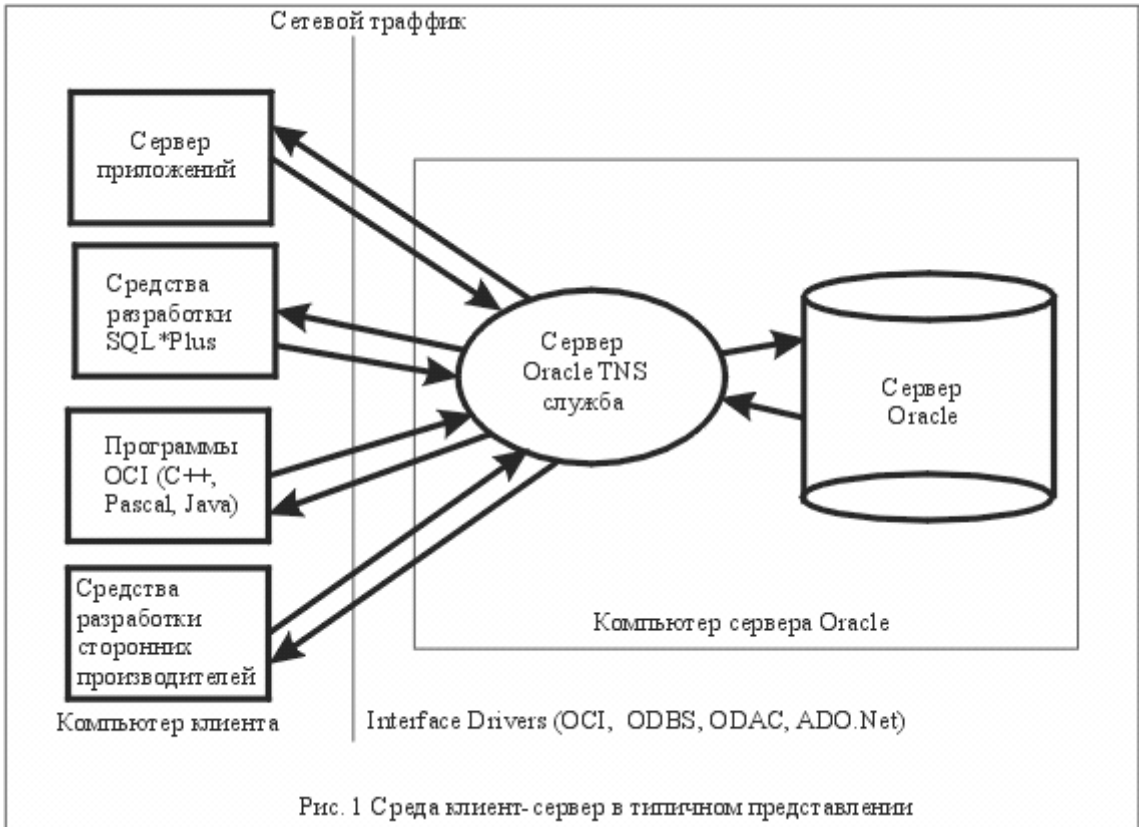

Рис. 1 Среда клиент-сервер в типичном представлении

Здесь хорошо видно, что используется при работе с сервером БД **Oracle**. Клиенты использующие программы на языках высокого уровня, с большим количеством единичных запросов **SQL**. Средства разработки **SQL*Plus**, серверы приложений, и т.д. Между клиентами и сервером, как правило, стоит некий посредник, так же определяющий скорость обработки запросов поступающих от клиента к серверу. На жаргоне БД админов, их еще называют "толстыми" или "тонкими" клиентами. Например, я на первом этапе работы с БД использовал сервер БД **InterBase**, а клиентов создавал на несгибаемом **Borland Delphi** с использованием "толстого" клиента **IBX** компонент. Затем я подрос и стал работать с сервером БД **Oracle**, а клиентские части разрабатывал уже на **Borland C++**, с использованием "тонкого" клиента **ODAC.NET** компонент, фирмы **CrLab** (www.crlab.com), сегодня для работы с **Oracle** сервером я использую **MS Visual Studio.Net**, а именно среду разработки **C#** с "толстым" клиентом **OraADO.NET** той же **CrLab**. Собственно получается довольно не плохо! Что такое "толстый" и "тонкий" клиенты, расскажу чуть позже. Пока надеюсь ясно как сервер **Oracle** реализует взаимодействие с внешним миром.

Идем дальше:

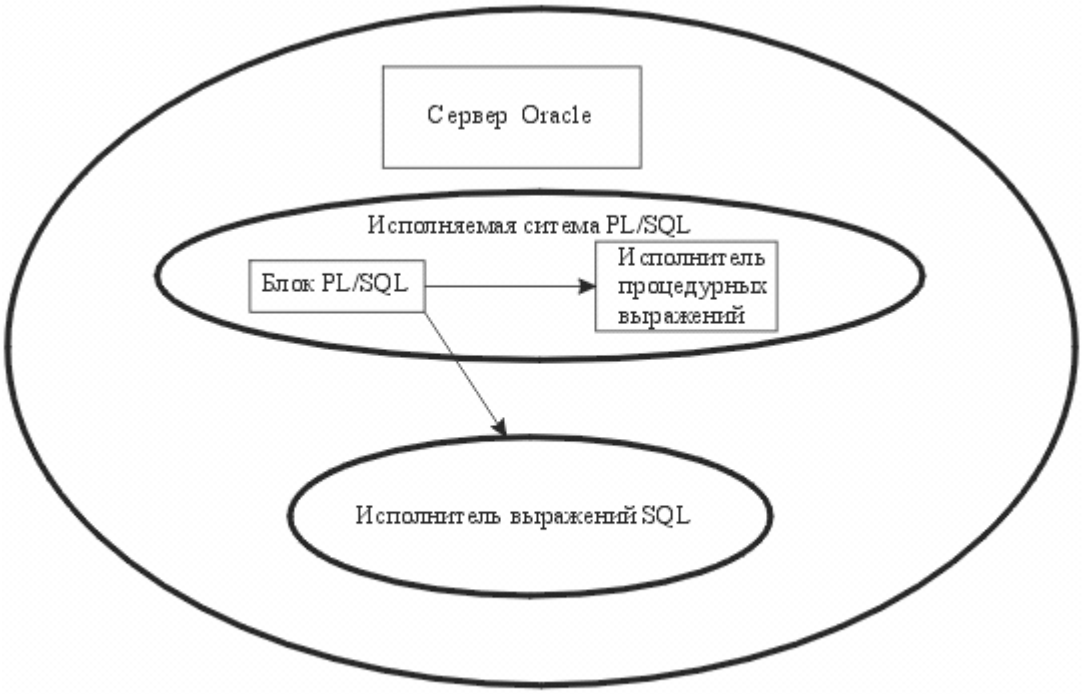

Рис. 2 Система исполнитель языка PL/SQL (является компонентом сервера БД Oracle

Программы, написанные на языке **PL/SQL**, выполняются системой-исполнителем языка, которая представляет собой часть сервера БД **Oracle**. Независимо от средства, с помощью которого формируется исполняемый код, он посылается на сервер **Oracle**. Система исполнитель языка **PL/SQL** сканирует, разбирает и компилирует код. После этого код, готов к выполнению. Выполняется код посредством передачи его **SQL Statement Executor** (системе исполнителю **SQL**-кода). Набор данных, полученных в результате исполнения запроса, поступает в систему исполнитель **PL/SQL** для дальнейшей обработки. Вот таким образом действует этот механизм. Далее давайте рассмотрим, в чем преимущество использования **PL/SQL** как процедурного языка.

Посмотрим на рисунок:

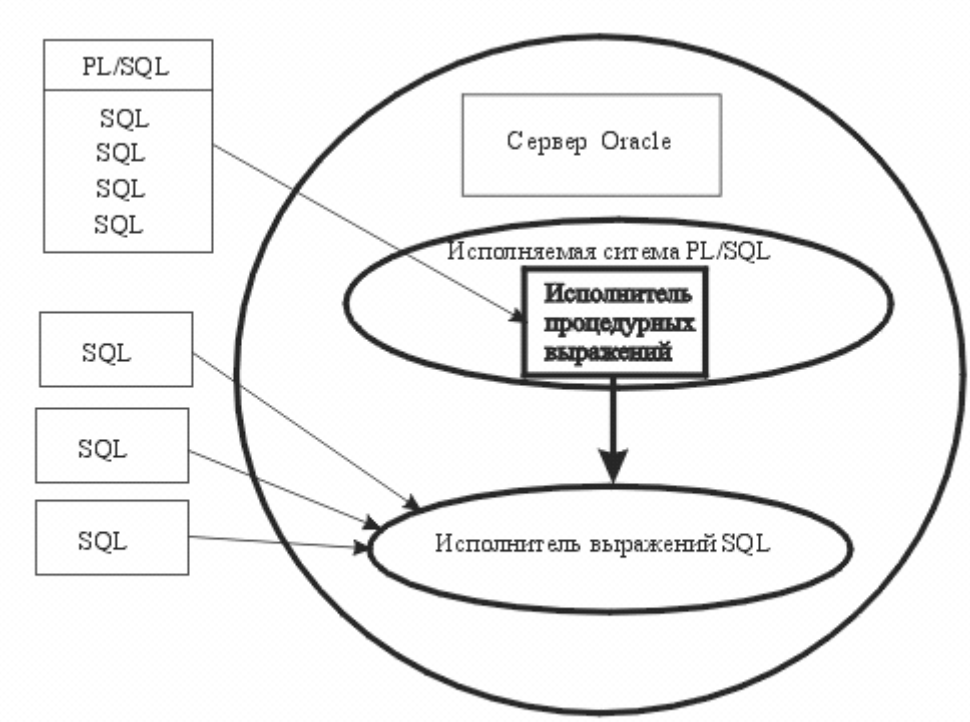

Рис. 3 Группировка SQL кода в единый блок языка PL/SQL, позволяет значительно уменьшить загрузку сети

Основное преимущество при работе с **PL/SQL** как языком БД является то, что на исполнение серверу посылается группа **SQL** предложений, а не единичные запросы сформированные, скажем какой-либо программой не использующей **PL/SQL** как средство работы с БД. Действительно, получив **PL/SQL** блок, сервер приступает к его обработке, а клиенту остается только ожидать результата операции. Тогда как единичные запросы к БД, порождают большой трафик и тормозят работу сети в целом. То есть, говоря точным языком, на сервере снижается число активных транзакций, вследствие того, что за одну активную транзакцию, обрабатывается большее число операторов **PL/SQL**! Вот таким образом, строится в целом работа с блоками **PL/SQL**.

альнейшим развитием процесса разработки программного кода и уменьшение трафика сети при работе с **PL/SQL**, является использование откомпилированных и именованных блоков **PL/SQL**. А именно процедур и функций. Понятие "именование", означает хранение имени подпрограммы, включая ее код. В **PL/SQL**, присутствуют так же неименованные блоки. До них мы еще доберемся. Посмотрим на рисунок:

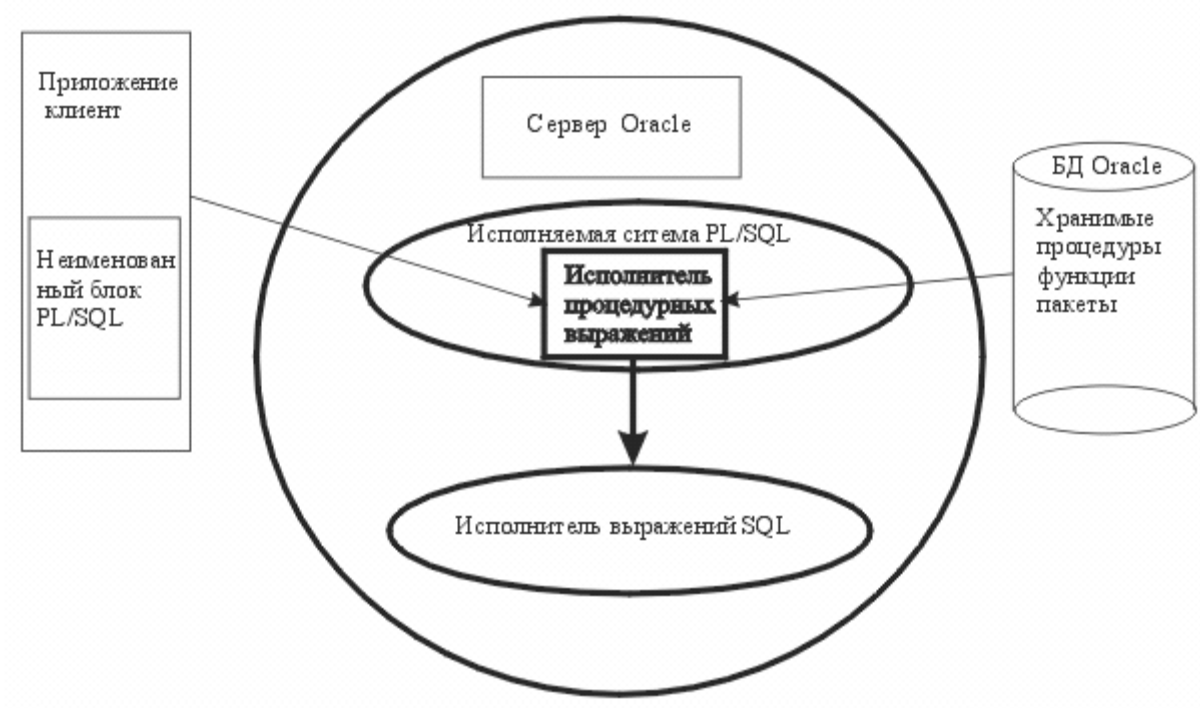

Рис.1 Система-исполнитель языка PL/SQL при выполнении хранимых процедур

Здесь показано, как выполняются хранимые процедуры и функции, вызываемые неименованными блоками. Подпрограммы могут выполнять достаточно сложные вычисления, а так же обрабатывать ошибки. Такой вызов подпрограмм еще называют "удаленный вызов процедур" (**Remote Procedure Call**). Либо возможно так же вызовы одной подпрограммы другой! Использование хранимых процедур и функций позволяет вам добиться максимальной производительности сервера БД, а так же эффективного повторного использования ранее созданного кода. Все блоки **PL/SQL**, в процессе компиляции хранятся в системном КЭШе - называемом **SGA** (**System Global Area**), там же находятся все таблицы схемы, переменные и т.д. А вот теперь запоминайте хорошенько! Пространство, выделенное для каждого отдельного блока **PL/SQL**, называется **"КУРСОР"**! Все **PL/SQL** блоки сохраняются в **SGA** посредством алгоритма **Least Recently Used** (звучит как - использованные наиболее давно). Любой **SQL** код внутри блока **PL/SQL**, так же получает собственные разделяемые области **SQL**. Так же по завершении компиляции подпрограммы запоминаются в словаре данных. Код всех подпрограмм является - реентерабельным! То есть к подпрограмме может обращаться несколько схем согласно прав использования подпрограмм. Все подпрограммы загруженные в область **SGA** являются разделяемыми. Вот пока на этом остановимся и поговорим на тему средств создания программ на **PL/SQL**. Обычно для этого достаточно блокнота, который встроен в **Windows** и среды **SQL*Plus**. Все можно приступать к написанию программ на **PL/SQL**! Просто и со вкусом, для начала я вам советую так и делать.