Вопросы к экзамену

Администрирование баз данных и приложений / Базы данных

1. Дистрибутивы СУБД Oracle. Установка СУБД Oracle 12с на Windows. Global Database Name и SID.

**Дистрибутивы:**

на oracle.com выбрать *Database 12c* > *Download* > 2 файла для вашей ОС

**Инсталляция:**

запуск Oracle Database 12c Installer

\* указ. email (необяз)

\* Skip software updates

\* Create and configure database

\* *Server class* / Desktop class

\* Single instance database installation

\* *Typical* / Advanced install

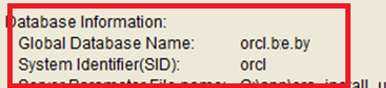
\* Create New Windows User : name, password

\* Oracle database name : orcl.be.by

Administrative password : ….

SID : orcl

***SID (System ID)*** – уник. имя, кот. однозначно идентиф. экземпляр/БД



1. Основные системные пользователи. Основные специальные привилегии. Роль DBA.

*Основные системные пользователи Oracle:*

**SYS** – предопределенный привилегированный пользователь ранга администратора базы данных, который является владельцем ключевых ресурсов БД Oracle

**SYSTEM** – предопределенный привилегированный пользователь, которому принадлежат ключевые ресурсы БД Oracle

**DBA –** предопределенная роль, которая автоматически создаётся для каждой базы данных Oracle и содержит все системные привилегии, кроме SYSDBA и SYSOPER

* SYSDBA и SYSOPER - специальные привилегии администратора, которые позволяют выполнять базовые задачи администрирования: запуск или остановка экземпляра БД; создание, удаление, открытие или монтирования базы и др.

1. Понятия базы данных и экземпляра базы данных.

**База данных** — это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе. База данных обычно управляется [системой управления базами данных (СУБД)](https://www.oracle.com/cis/database/what-is-database/#WhatIsDBMS).

**Oracle Database** или Oracle RDBMS — объектно-реляционная система управления базами данных компании Oracle.

Реляционная база данных представляет собой набор взаимосвязанных таблиц.

Экземпляр Oracle:

* запущенный сервер (программа) СУБД Oracle
* общая (глобальная) область памяти (SGA – system global area) и др. системные области памяти
* фоновые процессы, предназначенные для управления файлами базы данных

1. Запуск и останов экземпляра базы данных Oracle.

Старт экземпляра базы данных Oracle:

* STARTUP NOMOUNT

Запуск экземпляра Oracle без монтирования базы данных

Доступ пользователей запрещен

Используется для создания БД и пересоздания управляющих файлов

* STARTUP MOUNT

Запустить экземпляр, монтировать БД, но не запускать ее

Считан файл параметров

Происходит обращение к управляющим файлам

Проверяется состояние файлов БД

Доступ пользователей запрещен

Из состояния STARTUP NOMOUNT переводится ALTER DATABASE MOUNT

* STARTUP OPEN

Запустить экземпляр

Монтировать и открыть БД

Могут присоединяться пользователи

Из состояния STARTUP MOUNT переводится ALTER DATABASE OPEN

Можно стартовать в режиме только для чтения

ALTER DATABASE OPEN READ ONLY

ALTER DATABASE OPEN READ WRITE

 STARTUP RESTRICT - для пользователей, имеющих привилегию RESTRICTED SESSION

ALTER SYSTEM DISABLE(ENABLE) RESTRICTED SESSION

* STARTUP FORCE
* STARTUP RESTRICT

Останов экземпляра базы данных:

* SHUTDOWN NORMAL

Запрещено создавать новые сессии

Ожидается завершение работы всех пользователей

Самый безопасный и долгий способ останова

Никаких восстановительных работ при следующем старте не проводится

* SHUTDOWN TRANSACTIONAL

Запрещено создавать новые сессии

Запрещено запускать новые транзакции

Сервер дожидается завершения уже начатых транзакций и отключает пользователей, не имеющих активных транзакций

Применяется в случаях, когда нет возможности применить NORMAL

Никаких восстановительных работ при следующем старте не проводится

* SHUTDOWN IMMEDIATE

Запрещено создавать новые сессии

Запрещено запускать новые транзакции

Все незафиксированные транзакции откатываются

Применяется в случаях, когда нет возможности ждать

Никаких восстановительных работ при следующем старте не проводится

* SHUTDOWN ABORT

Применяется в крайних случаях, когда остальные режимы останова не приводят к результату

Все действия прекращаются

Все транзакции не фиксируются и не откатываются

Пользователей отсоединяют от БД

При следующем старте будет выполнено возможное восстановление

1. Словарь базы данных: назначение, применение, основные представления.

**Словарь Oracle** - набор таблиц и связанных с ними представлений, который представляет возможность отследить внутреннюю структуру базы данных и деятельность СУБД Oracle

* Создается при генерации базы данных
* Обновляется и обслуживается сервером Oracle в фоновом режиме после выполнения операторов DDL
* Позволяет запрашивать данные в виде представлений
* Содержит следующую информацию:

- Имена пользователей сервера Oracle

- Уровни привилегий пользователей

- Имена объектов базы данных

- Табличные ограничения

- Учетные данные

* Располагается в системном табличном пространстве SYSTEM
* Владелец: пользователь SYS, некоторые представления - SYSTEM
* Для доступа к словарю необходима специальная привилегия GRANT SELECT ANY DICTIONARY
* - USER Объекты, принадлежащие пользователю
* - ALL Объекты, к которым пользователь имеет доступ
* - DBA Все объекты базы данных (для администратора БД)
* - V$ Производительность сервера
* Вывод списка всех представлений словаря данных, доступных пользователю

SELECT \* FROM DICTIONARY;

1. Мультиарендная архитектура Oracle Multitenant.

* Oracle Multitenant - технология, позволяющая запустить несколько независимых баз данных в рамках одного экземпляра.
* Каждая база данных имеет свой набор табличных пространств и набор схем, но при этом у них общая SGA и один набор серверных процессов.
* Базы данных изолированы, друг о друге ничего не знают, не конфликтуют между собой.
* Словарь разбивается на две части: общую часть и локальную.
* CDB - container DB – контейнер базы данных
* PDB - pluggable DB – подключаемые базы данных
* Можно создавать несколько CDB – для разных версий программного обеспечения СУБД.
* Одну и ту же PDB можно переносить между CDB.
* В CDB создается главный контейнер Root. Root содержит метаданные CDB.
* В одной CDB можно создать до 252 PDB.

1. Файлы экземпляра Oracle. Файл параметров, управляющие файлы, файлы паролей, файлы трассировки.

управляющие файлы

Поскольку база данных Oracle является физическим набором связанных файлов данных, то для их синхронизации и контроля требуется особые методы. Для этих целей используются управляющие файлы.

База данных Oracle может иметь один или несколько управляющих файлов. Если имеется несколько управляющих файлов, все они должны быть абсолютно идентичными. При каждом запуске базы данных Oracle читает информацию управляющего файла, а при каждом изменении размещения или добавления новых файлов данных и журналов базы данных обновляет управляющий файл.

* Если надо изменить управляющий файл, то следует создать сценарий, откорректировать его и выполнить
* **Control files** – файлы, содержащие имена (местоположение) основных физических файлов базы данных и некоторых параметров

Файл параметров

Для того, чтобы запустить экземпляр базы данных, Oracle должен прочесть файл параметров и определить, какие параметры инициализации установлены для этого экземпляра. В файле параметров содержатся многочисленные параметры и их установленные значения. Oracle считывает файл параметров при запуске базы данных. Можно создать несколько файлов параметров, каждый будет соответствовать различным конфигурациям экземпляра.

* spfile - бинарный файл, который используется сервером Oracle при старте.
* pfile - текстовый файл с параметрами, будет использоваться при старте, если не будет найден spfile.
* Файл параметров предназначен для хранения параметров экземпляра



файлы паролей

* Предназначен для аутентификации администраторов базы данных

файлы трассировки

Когда возникает ошибка базы данных, может генерироваться файл трассировки (trace file). Они содержит подробную информацию о возникновении ошибки.

При работе базы данных события и ошибки регистрируются в текстовых файлах на сервере базы данных. Файл журнала предупреждений (alert log) нужен администратору базы данных для отслеживания важнейших действий с базой данных - наподобие открытия и закрытия базы данных, установления параметров загрузки базы данных и переключения оперативных журналов повтора. Также в эти файлы записываются многие ошибки базы данных для последующего расследования их причин. Любые структурные изменения базы данных также регистрируются в файле журнала предупреждений.

1. Файлы базы данных Oracle. Файлы данных, журналы, архивы.

Файлы данных

Все данные в базе данных Oracle сохраняются в файлах данных. Все таблицы, индексы, триггеры, последовательности, программы на PL/SQL, представления - все это находится в файлах данных. И хотя эти и другие объекты базы данных логически содержатся в табличных пространствах, в действительности они сохраняются в файлах на жестком диске компьютера.

В каждой базе данных Oracle имеется по крайней мере один файл данных (но обычно их бывает больше). Если вы создаете в Oracle таблицу и заполняете ее строками, Oracle помещает эту таблицу и строки в файл данных. Каждый файл данных может быть связан только с одной базой данных.

журналы,

Оперативные файлы журналов повтора - предназначены для записи всех изменений, выполненных над данными базы данных Oracle. Используется для хранения на диске информации для повторного выполнения операций.

Для компьютера выполнить задачи повторно - означает выполнить ее точно так, как она выполнялась в предыдущий раз. Поэтому назначение оперативного файла журнала повтора заключается в сохранении информации об изменениях в базе данных таким, образом, чтобы позже их можно было повторить.

Каждая база данных должна иметь не менее двух оперативных файлов журналов повтора. Текущий файл постепенно заполняется, после его заполнения (или переключения некоторыми командами), база данных приступает к записи в следующий файл. Эта операция называется переключением журналов.

Поскольку файлы повтора необходимы для выполнения восстановления базы данных и являются критичными, их объединяют в группы. Запись происходит одновременно в файлы одной группы.

* **Мультиплексирование журналов повтора –** поддержка несколько копий каждого журнала
* **Журналы повторного выполнения -** дисковыересурсы, в которых фиксируются изменения вносимых пользователями в базу данных;
* журнал представляет собой файл операционной системы;
* как минимум должно быть два файла;
* журналы применяются при восстановлении базы данных.

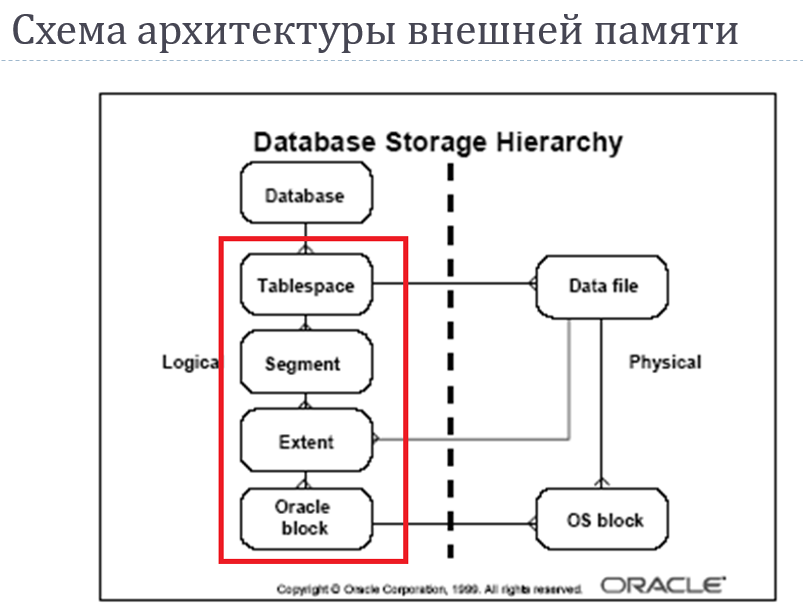
архивы

Как только оперативный файл журнала повтора (Redolog) оказывается заполнен, программное обеспечение сервера Oracle начинает запись в следующий файл. Эта операция повторяется, как следствие информация в оперативных файлах журнала (Redolog) многократно перезаписывается.

Если необходимо сохранить историю изменений, нужно, чтобы после переключения журналов сохранялась их копия. Для этого достаточно перевести работу базы данных в режим работы ARCHIVELOG.

ALTER DATABASE ARCHIVELOG

1. Абстрактная модель Oracle. Логическая структура внешней памяти.



СЕГМЕНТ:

* Сегмент – область на диске, выделяемая под объекты.
* Сегментытипизируются в зависимости от типа данных, хранящихся в них – сегменты таблиц, сегменты индексов, сегменты кластеров и т.д.(всего 10 типов).
* сразу после создания табличного пространства в нем нет сегментов
* Сегмент хранит только данные, поэтому он создается только при добавлении данных (в примере строки в таблицу).
* При удалении строк (delete) из таблицы, сегмент не удаляется.
* При удалении таблицы (drop table) изменяется имя сегмента, и информация об удалении записывается в словарь базы данных.
* Таблица и ее содержимое (сегмент) могут быть восстановлены с помощью механизма RECYCLEBIN.
* Для удаления RECYCLEBIN-сегмента применяется команда PURGE.

ЭКСТЕНТ:

* Экстент – непрерывный фрагмент дисковой памяти.
* Является единицей выделения вторичной памяти (выделяется целым числом экстентов).
* Когда экстент заполняется выделяется следующий.
* Размер экстента варьируется от одного блока до 2 Гб.
* Длину выделяемого экстента вычисляет СУБД (не меньше 64К).
* Если при создании табличного пространства задана опция UNIFORM, то все экстенты имеют одинаковую длину.
* Управление экстентами возможно локальное LOCAL и через словарь базы данных DICTIONARY.

БЛОКИ:

* Блок – минимальная единица объема памяти, применяемая при записи и чтении данных.
* Размер кратен 2К, и должен быть кратен величине блока операционной системы (2К, 4К, 8К, допустимы 16К, 32К).
* Устанавливается в файле параметров экземпляра при создании БД,
* В табличном пространстве все блоки одного размера.



1. Абстрактная модель Oracle. Физическая структура внешней памяти.

**Физические структуры базы данных Oracle (Oracle Database)** — это действительные файлы данных Oracle на уровне операционной системы. База данных Oracle состоит из следующих трех основных типов файлов.

* [Файлы данных](https://oracle-patches.com/oracle/begin/1272-%D1%84%D0%B0%D0%B9%D0%BB%D1%8B-%D0%B1%D0%B0%D0%B7%D1%8B-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-oracle). Эти файлы хранят данные таблиц и индексов.
* [Управляющие файлы](https://oracle-patches.com/oracle/prof/641-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%8F%D1%8E%D1%89%D0%B8%D0%B5-%D1%84%D0%B0%D0%B9%D0%BB%D1%8B-oracle). Эти файлы записывают изменения всех структур баз данных.
* [Файлы журналов повторного выполнения](https://oracle-patches.com/oracle/prof/642-%D1%84%D0%B0%D0%B9%D0%BB%D1%8B-%D0%B6%D1%83%D1%80%D0%BD%D0%B0%D0%BB%D0%BE%D0%B2-%D0%BF%D0%BE%D0%B2%D1%82%D0%BE%D1%80%D0%B5%D0%BD%D0%B8%D1%8F-%D1%82%D1%80%D0%B0%D0%BD%D0%B7%D0%B0%D0%BA%D1%86%D0%B8%D0%B9-oracle). Эти онлайновые файлы содержат изменения, проведенные в табличных данных.

1. Абстрактная модель Oracle. Структура SGA.



* Память различным пулам в SGA выделяется блоками, которые называются гранулами
* Одна гранула (granule) - это область памяти размером 4, 8 или 16 Мбайт
* Гранула является наименьшей единицей выделения памяти

БУФЕРНЫЙ ПУЛ

* Буферный пул (буферный кэш) – область SGA, которая содержит образы блоков, считанные из файлов данных или созданные динамически, чтобы реализовать модель согласованного чтения
* Поддерживается два списка блоков:
  + список грязных блоков (отличаются от своей копии на диске и должны быть записаны в табличное пространство)
  + список чистых блоков (не измененные блоки)
* Алгоритм LRU (least recently used)– первыми вытесняются блоки с наименьшим значением счетчика
* Запись грязных блоков на диск осуществляется в 4х случаях:
  + 1) истечение тайм-аута (3 сек);
  + 2) контрольная точка;
  + 3) превышение длины грязных блоков заданного лимита;
  + 4) процесс не может обнаружить свободный блок.



# **DEFAULT**

действует по механизму LRU

# **KEEP**

удерживает указанные блоки (тоже LRU; блокам назначаются наибольшие значения счётчика)

# **RECYCLE**

сразу пишет на диск

CACHE – помещение таблицы в конец LRU-списка (для малых таблиц) обычно в default pool

БУФЕР ЖУРНАЛА ПОВТОРНОГО ВЫПОЛНЕНИЯ

* Буфер журнала повторного выполнения предназначен для временного циклического хранения данных журнала повтора
* Позволяет ускорить работу сервера за счет буферизации
* Содержимое сбрасывается на диск (в журнал повтора) в 4-х случаях:
  + 1) каждые три секунды;
  + 2) при фиксации транзакции;
  + 3) при заполнении буфера на 1/3;
  + 4) если в буфере более 1m данных журнала повтора.

ФИКСИРОВАННАЯ ОБСЛАСТЬ SQA

* хранит переменные, указывающие на другие области памяти, значения параметров;
* представляет собой загрузочный бинарный код;
* размер области зависит от платформы, версии операционной системы;
* размером фиксированной области SGA управлять нельзя.

РАЗДЕЛЯЕМЫЙ

Разделяемый пул (shared pool) используется в базе данных Oracle для хранения следующей информации:

* информации о последних выполненных командах SQL
* информации из словаря данных, к которая недавно запрашивалась

Состоит из двух структур памяти, оказывающих существенное влияние на производительность:

* [библиотечный кэш](https://oracle-patches.com/oracle/begin/285-library-cache-%D0%B1%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D1%87%D0%BD%D1%8B%D0%B9-%D0%BA%D1%8D%D1%88);
* [кэш словаря данных](https://oracle-patches.com/oracle/prof/3260-%D0%BA%D1%8D%D1%88-%D1%81%D0%BB%D0%BE%D0%B2%D0%B0%D1%80%D1%8F-dictionary-cache-%D0%B2-%D0%B1%D0%B0%D0%B7%D0%B5-%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85-oracle-%D0%B2-%D0%B4%D0%B5%D1%82%D0%B0%D0%BB%D1%8F%D1%85).

БОЛЬШОЙ

* Большой пул - область памяти SGA, применяемая для хранения больших фрагментов памяти
* В этой области не применяется вытеснение по алгоритму LRU

JAVA ПУЛ

* Java-пул предназначен для работы Java-машины;

1. Абстрактная модель Oracle. Серверные процессы Oracle.

**Процесс** (process) – механизм ОС Windows, осуществляющий запуск и выполнение приложений

Процесс создается, когда запускается приложение

В общем случае выполняется в собственной области памяти

**Поток** (thread) – индивидуальная ветвь внутри процесса, выполняющая конкретные программные инструкции

* Серверные процессы – процессы, выполняющиеся на основании клиентского запроса
* Подчиненные процессы – аналогичны фоновым, но выполняют дополнительные действия для фонового или серверного процессов
* Oracle Net Listener – процесс на стороне сервера, прослушивающий входящие запросы клиента на соединение с экземпляром.
* Listener – это программа-сервер, прослушивающая TCP-порт, принимающая запросы на соединение с Oracle экземпляром от программ-клиентов.
* В результате успешной работы Listener устанавливается соединение между программой-клиентом и обработчиком запросов экземпляра.
* По умолчанию TCP-порт 1521

OracleNetListener – запрос на соединение

* 1)клиент выполняет запрос к Listener на соединение с сервисом экземпляра
* 2) Listener запрашивает соединение с сервером
* 3)сервер возвращает параметры соединения с обработчиком сервиса
* 4) Listener сообщает параметры соединения клиенту
* 5) клиент соединяется с обработчиком запросов сервиса для дальнейшей работы с сервисом в рамках соединения

1. Абстрактная модель Oracle. Фоновые процессы Oracle.

Фоновые процессы – запускаются вместе с базой данных и выполняют разнообразные задачи обслуживания

LREG

* Listener Registration Process **–** периодическая регистрация сервисов в процессе Listener
* Database Writer Process: (n=0,…,9, a,…,z; BWm, m=36,…,99) – фоновый процесс записывающий по LRU измененные блоки (грязные блоки) в файлы базы данных.
* CKPT- записывает информацию о контрольной точке в управляющие файлы
* Log Writer Process – только один,управляет буфером журналов повтора
* Archiver Process – копирует файлы журнала повтора после переключения группы журналов
* Process monitor – отвечает за очистку после ненормального закрытия подключений
* Recovery Process – разрешение проблем связанных с распределенными транзакциями
* Flashback Data Archiever – архивирование ретроспективных данных

1. Процесс-слушатель Oracle и его основные параметры.

Экземпляр может иметь несколько точек подключения

Точки подключения называются сервисами и имеют символические имена

При инсталляции автоматически создается два сервиса:

SYS$USERS (по умолчанию, указывается SID в параметрах соединения),

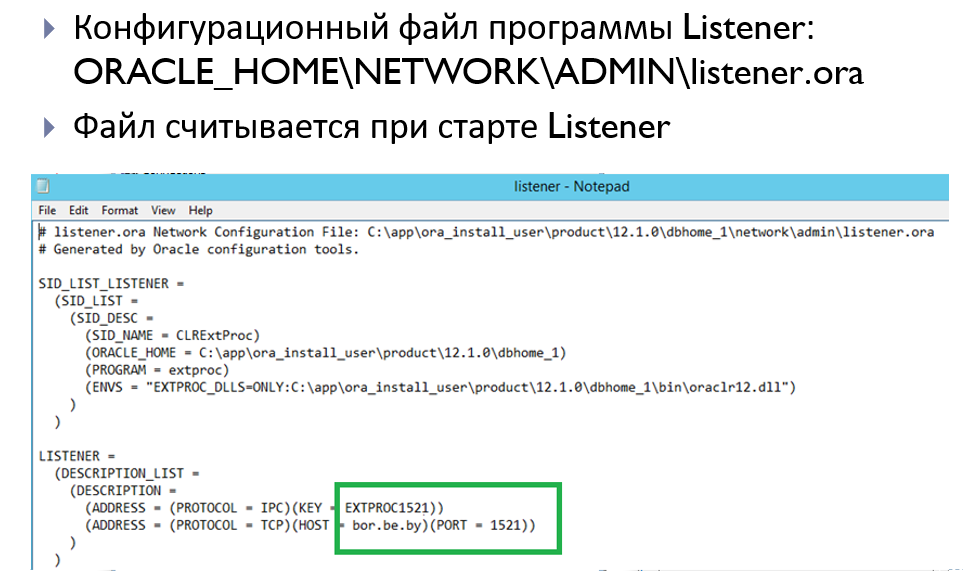
сервис с именем инстанса (указывается сервис)

При создании PDB для нее автоматически добавляется сервис с именем, совпадающим с PDB-именем

* Oracle Net Listener – процесс на стороне сервера, прослушивающий входящие запросы клиента на соединение с экземпляром.
* Listener – это программа-сервер, прослушивающая TCP-порт, принимающая запросы на соединение с Oracle экземпляром от программ-клиентов.

OracleNetListener – запрос на соединение

* 1)клиент выполняет запрос к Listener на соединение с сервисом экземпляра
* 2) Listener запрашивает соединение с сервером
* 3)сервер возвращает параметры соединения с обработчиком сервиса
* 4) Listener сообщает параметры соединения клиенту
* 5) клиент соединяется с обработчиком запросов сервиса для дальнейшей работы с сервисом в рамках соединения

****

1. Сетевые настройки Oracle. Установление соединения по сети.

Oracle поддерж.2 режма соед: dedicated(выделенный) и shared(разделяемый) server.

**Dedicated server** (режим по умолчанию) – для каждого клиента выделяется отдельный выделенный серверный процесс (обработчик запросов, dedicated server process) который называется shadow process (теневой процесс).

**Shared server**  – обрабатывает программа **dispatcher**:

1)получает запрос от клиента,

2)помещает их во входную очередь к разделяемым серверам;

3)незанятый разделяемый сервер извлекает и обрабатывает запрос;

4)после обработки разделяемый  сервер помещает результат обработки в выходную очередь;

5) из очереди результат извлекает диспетчер;

6) диспетчер пересылает результат клиенту

**Виды подключений**:

\*простое подключ (basic) – явно указ.все пар-ры соед.

CONNECT имя/пароль@[//]хост[:порт][/имя\_службы]

\*локальное именование (tns)

В tncnames (host, port, service\_name) -> oracle net manager

\*LDAP-соед – с помощью службы каталогов.

\*local/bequeath-соед

\*только на С

\*м.соед.с пом.sqlplus/sqldeveloper

\*без указ.пар-ров соед

\*только с выдел.С

\*listener не задействован

1. Табличные пространства СУБД Oracle и их основные параметры.

***TS*** – логич. структура хранения д-х, контейнер сегментов

граничный объект БД (файлы – лог)

*С одним TS связаны 1/неск файлов ОС, с каждым файлом ОС только 1 TS*

\* permanent – хранение постоянных объектов

\* temporary – хранение временных данных

\* undo – хранение сегментов отката, исп. всегда один (указ. в SPFILE.ORA)

**Параметры:**

create [temp] tablespace ..

datafile (tempfile) путь…

size

auto\_extent on next

maxsize

1. Роли и привилегии СУБД Oracle и их основные параметры.

***Привилегия*** – право выполнять конкр. тип предложений SQL, или право доступа к объекту другого юзера

\* grant / revoke

\* нельзя выдавать в одном предложении

| **системные** | **объектные** |
| --- | --- |
| на изменение с-мы: create table / trigger | на изм объекта: select |
| *WITH ADMIN OPTION* – дают право юзеру также назнач/отбир привил:  alter, analyze, audit, backup, create, drop, select (стараться редко исп)  any – для любого объекта;  all – для всех объектов | *WITH GRANT OPTION* – дают право юзеру также назначать/отбир привил (но объектные): alter, delete, execute, insert, update, select, references  снимает привил тот, кто их назначал |
| **объекты грантов:**  database, user, profile, tablespace, role, table, index, trigger, procedure, sequence, view | **объекты грантов:**  table, view, sequence, procedure |

***Роль*** – именованный набор привилегий

role, password\_required, authen\_type, common, oracle\_maintained

1. Пользователь СУБД Oracle и его основные параметры.

**Пользователь СУБД** – человек, у которого есть учетная запись, посредством которой определяются его полномочия и разрешения в пределах базы данных.create user .. identified by 12345

default tablespace .. quota unlimited on ..

temporary tablespace ..

profile ..

account unlock

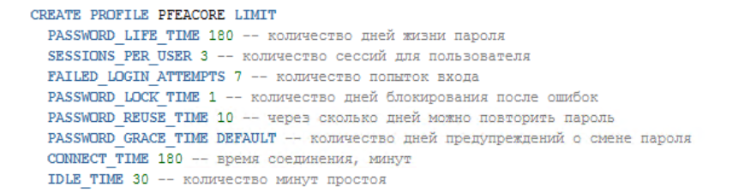
password expire

параметры пользователя, которые можно установить при его создании:

* Пароль
* Постоянное/временное табличное пространство, квота
* Профиль безопасности
* Тип аккаунта
* Действителен ли пароль

1. Профиль безопасности СУБД Oracle и его основные параметры.

***Профиль безопасности —*** Механизм позволяющий устанавливать ограничения на ресурсы, используемые каждым пользователем в базе данных, и осуществлять политику применения паролей для обеспечения безопасности.



1. Язык SQL. Основные операторы и их назначение.

Доступ к д-м осущ в виде запросов на языке SQL – декларативный (непроцедурный) язык: с его пом. можно сформул, что надо получить, но нельзя указать КАК.

\* DDL (definition) – create, alter, drop

\* DML (manipulation) – select, insert, update, delete

\* DCL (control) – grant, revoke

\* TCL – commit, rollback, savepoint

1. Таблица и ее основные параметры.

Таблица – основная структура сохранения информации в БД

Типы таблиц:

Традиционные таблицы (heap organized table)

Индекс-таблицы (index organized table)

Кластеризованные индекс-таблицы (index clustered table)

Кластеризованные хэш-таблицы (hash clustered table)

Отсортированные кластеризованные хэш-таблицы (sorted hash clustered table)

Вложенные таблицы (nested table)

Временные таблицы (temporary table)

Объектные таблицы

Внешние таблицы

* Может иметь до 1000 столбцов (<254)
* Может иметь неограниченное число строк
* Может иметь неограниченное число индексов
* Нет ограничения на число таблиц
* араметр PCTFREE – процент памяти блока, резервируемой для возможных обновлений строк, уже содержащихся в блоке
* Параметр PCTUSED – процент занятой части памяти блока
* Временные таблицы – механизм хранения данных в БД
* Состоит из столбцов и строк, как и обычная таблица
* Временные таблицы – глобальны
* Привилегии для создания временной таблицы CREATE TABLE
* Можно разместить временную таблицу в заданном табличном пространстве.
* Временные таблицы – это шаблон, хранящийся в словаре базы данных, для нее выделяется временный сегмент в (по умолчанию) TEMPORARY-табличном пространстве и для каждого пользователя свой.
* Каждый пользователь видит только свои данные (свой сегмент данных).
* Временные таблицы бывают:
  + ON COMMIT PRESERVE ROWS – на время сеанса, данные существуют только на время сеанса, возможны все DML-операторы, TCL-операторы
  + ON COMMIT DELETE ROWS – на время транзакции, данные существуют только на время транзакции, возможны все DML-операторы, после выполнения COMMIT или ROLLBACK таблица становится пустой
* Не могут быть индексно-организованными, нельзя секционировать, размещать в кластере.
* Данные повторного выполнения генерируются, но их количество пренебрежительно мало.



\* вложенные

\* временные (лок/глоб, удал когда юзер отсоед от С)

\* объектные (м. созд объекты и раб с ними)

\* внешние (м. раб с файлами как с таблицами)

1. Временные таблицы.

Вопрос 21

1. Ограничения целостности в таблицах.

**[NOT] NULL** Это ограничение задается для столбца и разрешает или запрещает наличие неопределенных значений в столбце при вводе или изменении его значений.

**UNIQUE** Это ограничение задается для столбца или совокупности столбцов и указывает, что значение столбца (совокупности столбцов) должно быть уникальным в пределах таблицы. Называется ограничением уникального ключа.

**PRIMARY KEY** Это ограничение указывает, что столбец или совокупность столбцов выступают в качестве первичного ключа, который уникально идентифицирует строки таблицы.

**FOREIGN KEY** Это ограничение указывает, что столбец или совокупность столбцов принимают значения, которые должны совпадать со значениями столбца (совокупности столбцов) из другого связанной таблицы. Это так называемое референциальное ограничение целостности.

Референциальное ограничение целостности также включает правила, которые указывают, какие типы манипулирования данными допустимы на те значения, на которые делается ссылка, и как такие изменения влияют на зависимые данные. Правилами, связанными с референциальным ограничением целостности, являются следующие:

**CHECK** Описание сложных правил ограничений целостности.

Проще:

**not null** (запрет значений null): предотвращает появление в столбце значений null

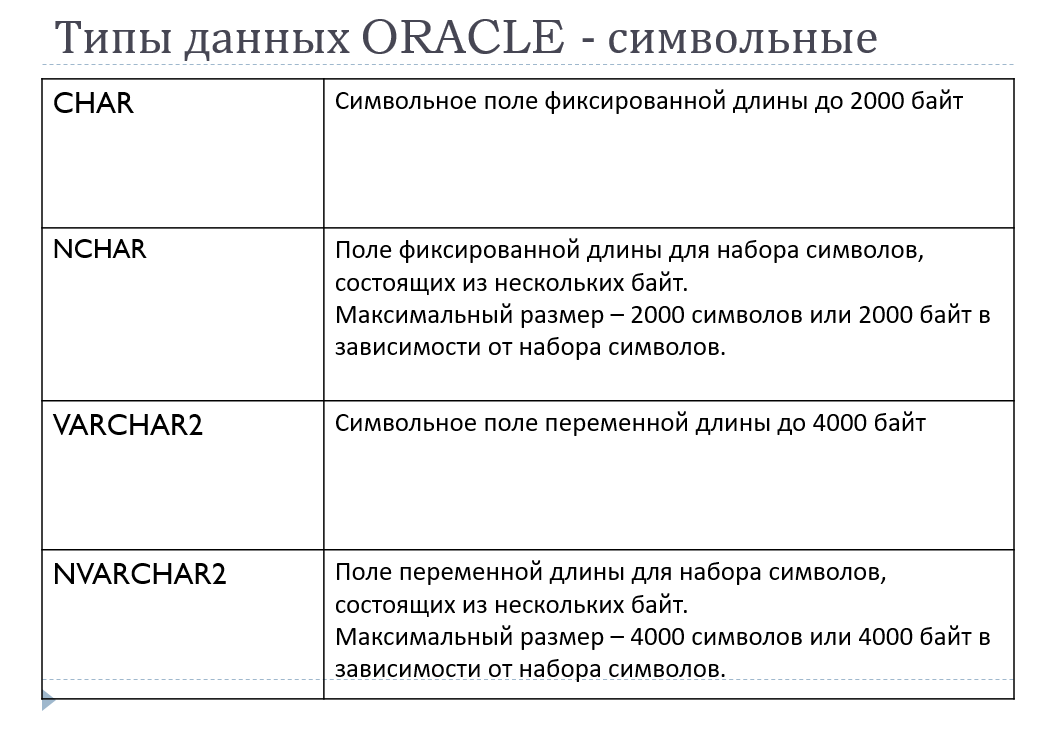
**primary key**(первичный ключ): предотвращает появление в столбце (группе столбцов) повторяющихся значений и пустого значения

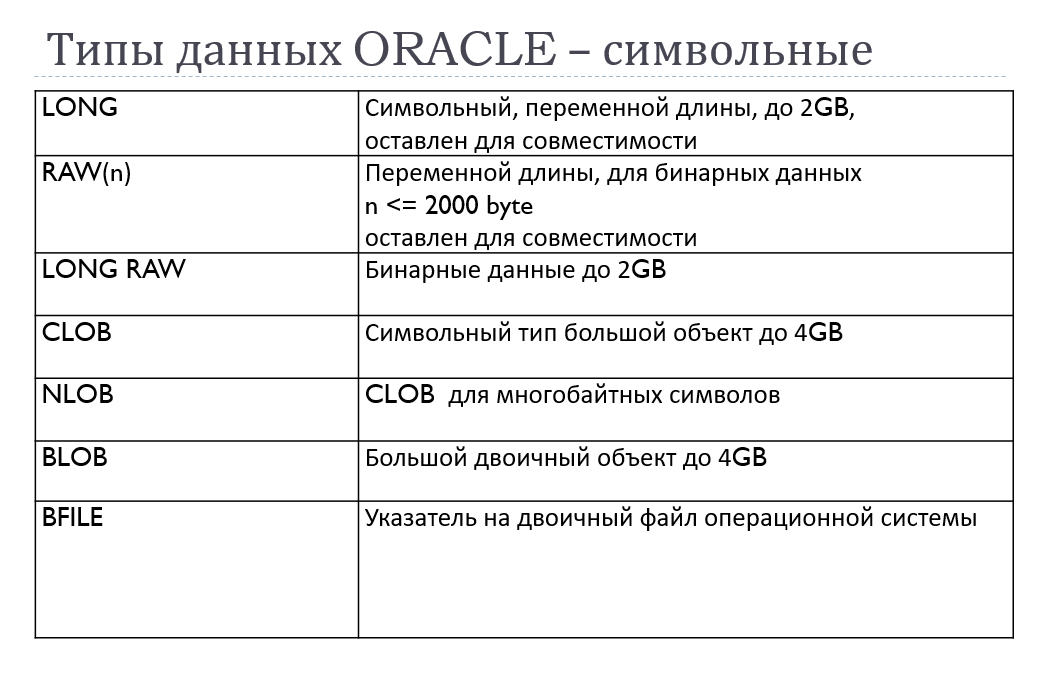
**foreign key**(внешний ключ): устанавливает связь между таблицей со столбцом, имеющим свойство foreign key (FK внешний ключ) и таблицей, имеющей столбец со свойством primary key (PK – первичный ключ); предотвращает несогласованные операции между PK и FK

**unique** (уникальное значение): аналогично  primary  key, но допускает пустые значения и не может быть использован для связи с foreign key

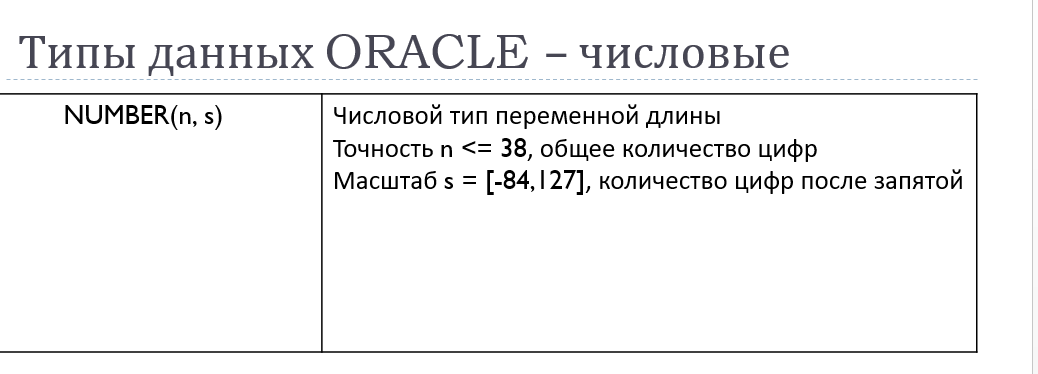
**check** (проверка значений): предотвращает  появление в  столбце значения, не удовлетворяющего логическому условию, записанному после check

1. Типы данных базы данных.









1. Индексы базы данных. Виды и особенности применения индексов.

Индекс – структура базы данных, используемая сервером для быстрого поиска строки в таблице

* B-Tree индекс предназначен для индексирования уникальных столбцов или столбцов с высокой селективностью. Селективность таблицы — значение, представляющее долю строк таблицы, удовлетворяющих определенному условию выбора.
* Bitmap индекс предназначен для индексирования столбцов с низкой селективностью. Селективность таблицы — значение, представляющее долю строк таблицы, удовлетворяющих определенному условию выбора.

Индексы могут относиться к нескольким типам, наиболее важные из которых перечислены ниже:

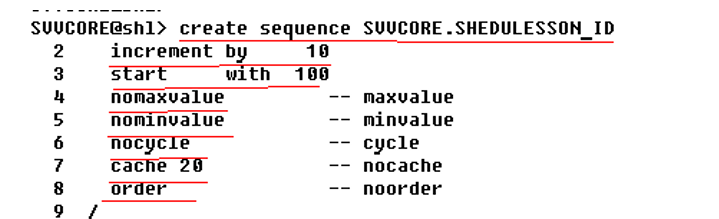
* Уникальные и неуникальные индексы. Уникальные индексы основаны на уникальном столбце – обычно вроде номера карточки социального страхования сотрудника. Хотя уникальные индексы можно создавать явно, Oracle не рекомендует это делать. Вместо этого следует использовать уникальные ограничения. Когда накладывается ограничение уникальности на столбец таблицы, Oracle автоматически создает уникальные индексы по этим столбцам.
* Первичные и вторичные индексы. Первичные индексы – это уникальные индексы в таблице, которые всегда должны иметь какое-то значение и не могут быть равны null. Вторичные индексы – это прочие индексы таблицы, которые могут и не быть уникальными.
* Составные индексы – индексы, содержащие два или более столбца из одной и той же таблицы.

1. Последовательность СУБД Oracle и ее параметры.

Последовательность – объект базы данных, предназначенный для генерации числовой последовательности

* GENERATED ALWAYS AS IDENTITY —значения определяются только сервером
* GENERATED BY DEFAULT AS IDENTITY — позволяет указать значение. Если значение не указано, то сервер назначит значение из последовательности

GENERATED BY DEFAULT ON NULL AS IDENTITY — позволяет указать значение. Если значение в столбце не указано или явно указать значение NULL



Параметры:

start with - значение, с которого начинается последовательность

minvalue - минимальное значение

maxvalue - максимальное значение

increment by - приращение/икремент

cache - определяет, сколько значений последовательности будут сохранены в памяти для быстрого доступа

Чтобы получить следующее значение,  нужно использовать nextval.

Чтобы получить текущее значение,  нужно использовать curval.

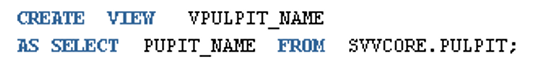
1. Кластер и его параметры

* Кластер – объект БД, который хранит значения общих столбцов нескольких таблиц
* Создание CREATE CLUSTER
* Привилегия CREATE CLUSTER
* Таблицы, с которыми часто работают совместно, можно физически хранить совместно.
* Для этого создается кластер, который будет их содержать
* Строки из отдельных таблиц сохраняются в одних и тех же блоках, поэтому объединяющие запросы выполняются быстрее
* Уменьшается количество операций ввода-вывода
* Производительность операций вставки, обновления и удаления может быть ниже, чем для обычных таблиц
* Связанные столбцы называются кластерным ключом
* Хэш-кластеры используют функции хэширования кластерного ключа строки для определения физической локализации места, где строку следует хранить
* Наибольшие преимущества – в запросах, использующих операции равенства

1. Представление и его параметры.

**Представление** (View) – это объект базы данных, представляющий собой *поименованный* SELECT-запрос, который хранится в базе данных. Представление создается с помощью оператора CREATE, удаляется с помощью оператора DROP и изменяется с помощью ALTER.

* Привилегия – CREATE VIEW
* Создание – CREATE (OR REPLACE) VIEW
* FORCE – создает представление, независимо от того, существуют ли таблицы и есть ли права
* NOFORCE – по умолчанию
* WITH CHECK OPTION – указывает, что будут вставлены или изменены строки, которые будут выбираться через это представление



1. Материализованное представление и его параметры.

* Материализованное представление — физический объект базы данных, содержащий результат выполнения запроса
* Привилегия – CREATE MATERIALIZED VIEW
* Создание – CREATE MATERIALIZED VIEW
* BUILD IMMEDIATE – создает представление в момент выполнения оператора
* START WITH – показывает, когда выполнится в первый раз (если не был построен сразу)
* NEXT– показывает, когда выполнится в следующий раз
* Далее – в разницу времени между START WITH и NEXT

Обновление:

* REFRESH
  + COMPLETE — полное обновление данных из базовых таблиц
  + REFRESH FAST – используются журналы фиксации изменений базовых таблиц
  + REFRESH FORCE – попытка быстрого обновления; если быстрое обновление невозможно, то выполняется полное обновление

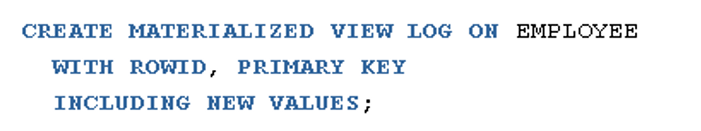
Обновление неявное по расписанию.

**REFRESH**

* **ON COMMIT** – обновление при COMMIT

**ON DEMAND** – обновление по требованию

* *REFRESH FAST*
* Создать журналы материализованного представления - вспомогательные таблицы, накапливающие сведения об изменениях, которые происходят в базовых таблицах



Отличие от обычного:

* Основное различие между представлением и материализованным представлением состоит в том, что представления физически **не сохраняются** на диске. С другой стороны, **материализованные представления хранятся на диске.**
* Материализованное представление отвечает **быстрее,** чем представление, поскольку материализованное представление предварительно вычисляется.
* Материализованное представление **использует** **пространство памяти в том** виде, в котором оно хранится на диске, тогда как представление - это просто **отображение,** следовательно, оно не требует места в памяти.
* Представление может быть определено как **виртуальная таблица,** созданная в результате выражения запроса. Однако Материализованное представление - это **физическая копия**, изображение или снимок базовой таблицы.
* Представление всегда **обновляется, так** как представление, создающее запрос, выполняется каждый раз, когда оно используется. С другой стороны, материализованное представление обновляется **вручную** или путем применения к нему **триггеров** .

1. Частные и публичные синонимы СУБД Oracle.

Синоним – способ обращаться к объекту базы данных без указания обязательной полной идентификации объекта (хост – экземпляр – владелец – объект).

Частный синоним принадлежит пользователю, который его создал.

Публичный синоним используется совместно всеми пользователями базы данных.

* Привилегия – CREATE (PUBLIC) SYNONYM
* Создание – CREATE (PUBLIC) SYNONYM
* Допустимость синонима не проверяется сервером при создании!
* Представление словаря dba.synonyms
* Может указывать на:
  + Таблицы,
  + Процедуры,
  + Функции,
  + Последовательности,
  + Представления
  + Пакеты
  + Объекты в локальной или удаленной базе данных

1. Основные характеристики языка PL/SQL.

* Procedure Language extensions to SQL  
  Основной язык для программирования хранимых процедур (stored procedure)  
  Интегрирован с БД Oracle  
  (быстрое переключ между движками PL/SQL и SQL)  
    
  Приложение мб проще в реализации при написании бизнес-логики на основе хранимых процедур  
  Нету накладных расходов на приведение типов  
  PL/SQL-функции можно вызвать из SELECT запросов  
    
  Взаимодействие с пользователем (user interaction);
* Внутренний язык (proprietary for Oracle);
* Cодержит элементы объектно-ориентированного программирования;
* Позволяет использовать объектные типы;
* Интерпретация (режим по умолчанию);
* Компиляция (промежуточный код на C и конечный объектный код процессора);

Среда выполнения: SQL\*Plus, SQL Developer, TOAD

* Модуль DBMS\_OUTPUT обеспечивает вывод информации для отладки
* Владелец – пользователь SYS.
* Принципы работы модуля DBMS\_OUTPUT следующий:
  + Операция PUT берет свои аргументы и помещает во внутренний буфер для хранения.
  + Операция GET считывает этот буфер и возвращает его содержимое процедуре в качестве аргумента.
  + Размер буфера устанавливается с помощью процедуры ENABLE.
* DBMS\_OUTPUT.put\_line();

1. Структура программы языка PL/SQL. Анонимные и именованные блоки.

заголовок  
IS  
Раздел объявл – не обяз. – объяв переменные  
BEGIN  
Исполняемый раздел – обяз., хоть один  
EXCEPTION  
Раздел исключений – не обяз.  
END;  
  
Анонимный блок  
} Не имеет секции заголовка  
} Используется как скрипт для выполнения PL/SQL выражений  
} Не может быть вызван из другого блока  
} Начинается с DECLARE или BEGIN  
} Варианты использования:  
} Триггер на стороне клиента (Oracle Development Tools)  
} Триггер базы данных (содержит АБ)  
} SQL-скрипт (описание процедур, функций и execute)  
} Откомпилированная программа (блок в execute команде, выполня-ющейся на сервере)  
  
Именованные блоки – процедуры и функции

1. Типы данных, основные операции, константы языка PL/SQL.

Типы данных:  
- скалярные символ/строка, число, булев, дата/время  
- ссылочные  
- составные  
- больште объекты объекты  
Константы: с пом constant  
  
Основные операции: DML операторы sql(insert,update,delete)

1. Поддержка национальных языков в СУБД Oracle. Наборы символов. Байтовая и символьная семантика символов.

* NLS - National Language Support, далее Globalization Support
* Можно хранить данные множества национальных языков, используя Unicode или специальные кодировки – наборы символов (character set)
* Символы хранятся как коды символов, зависящие от выбранного набора символов
* В одной БД могут использоваться два набора символов: основной (database character set) и дополнительный (national character set)
* Устанавливаются при создании БД
* Изменяются alter database (national) character set
* Кроме символов алфавита в набор включаются знаки препинания, числа, символы денежных единиц и пр.

} для Windows NLS-LANG может быть установлена в реестре и в переменных среды  
} проверить, установлена ли переменная среды  
HOST ECHO %NLS\_LANG%  
если ответ %NLS\_LANG%, то не установлена  
  
} рекомендуется устанавливать только в реестре  
  
  
осн набор символов для:  
- хран char, varchar2, clob, long  
- опис имен объектов, пер-ных  
- ввод и хран pl/sql модулей  
доп набор символов для:  
- хран nchar, nvarchar2, nlob  
  
пер-ная окружения NLS\_LANG = lang\_territory.charset  
lang – имена месяцев, дней, направл текста : по умолч american  
territory – настр календаря, формат даты, денег  
charset – отобр символов, заглавных букв,  
словари: nls\_ [session | instance | database] \_parameters  
  
Семантика символов:  
\* байтовая – рассм строки как посл-сть байтов (по умолч)  
\* символьная – рассм строки как посл-сть символов  
задается nls\_length\_semantics

1. Связанные объявления переменных: инструкция %TYPE, инструкция %ROWTYPE.

* Скалярная ссылка %TYPE для определения переменной на основе другой переменной или поля в таблице
* Ссылка на запись %ROWTYPE для определения структуры записи на основе таблицы или курсора

1. Локальные процедуры и функции языка PL/SQL.

Лок.прогр. модуль – процедура/ф-ция, опред.в секции декларации pl/sql блока  
Лок.процедуры и ф-ции:  
\*их объявление дб в конце секции декларир.после всех типов, записей, кур-соров, пер-ных и исключений  
\*мб исп-ны только в рамках блока, в кот.они объявл  
\*мб перегружены (разн.число пар, др.тип прогр.модуля,

др.семейство пар-ров)  
  
Лок.процедура:  
  
  
Лок.функция:

1. Использование записей в PL/SQL. Вложенные записи.

Запись – структура данных, составл.из полей.  
Типы записей:  
\*табличные  
\*курсорные  
\*программно-определенные  
Объявление записей:  
\*на осн.таблицы (%rowtype)  
declare  
one\_book books%rowtype;  
\*на осн.курсора (cursor + %rowtype)  
declate cursor myc is  
select \* from books where author like ‘%Fadeev%’;  
one\_curs myc%rowtype;  
\*запись опр-мая прогером:  
  
  
Вложенные записи – одно из полей внешней записи в действительности явл.полем другой записи.

1. Операторы управления, операторы цикла языка PL/SQL.

1) if … then … end if;  
2) if … then … else … end if;  
3) if … then … elseif …then …(elseif … then …) … else … end if;  
Оператор CASE  
case x  
when … then …;  
when … then …;  
when x between 13 and 20 then ….;  
else ….;  
end case;  
Отличие: в 1 выбир.знач и сравниваем его с чем-то. Во 2 – проверяем условия.  
  
Циклы loop, for, while  
\*for – нельзя измен.знач.в цикле  
\*loop – обяз.указ.когда вых. из цикла  
[\*while](https://vk.com/while) – вып. пока вып.усл.цикла  
  
Выход из цикла:  
\*exit – безусл.выход  
\*exit when – выход при вып.усл.  
\*goto – выход во внешн.контекст

1. Курсоры. Виды курсоров. Схемы обработки курсора.

* **Курсор** — объект БД, который позволяет приложениям работать с записями построчно
* Поддерживаются два типа курсоров:
  + явный — объявляется разработчиком;
  + неявный — не требует объявления
* Курсор может возвращать одну строку, несколько строк или ни одной строки
* Для повторного создания результирующего набора для других значений параметров курсор следует закрыть, а затем повторно открыть
* DECLARE — выполняет объявление явного курсора.
* OPEN — открывает курсор, создавая новый результирующий набор на базе указанного запроса.
* FETCH — выполняет последовательное извлечение строк из результирующего набора от начала до конца.
* CLOSE — закрывает курсор и освобождает занимаемые им ресурсы.

Ошибки неявного курсора:  
- no\_data\_found – не возвр строк вообще  
- too\_many\_rows – более 1 строки  
select into чтобы вернуть ровно 1 строку – точную выборку!

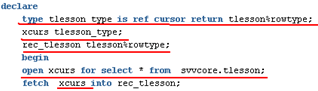
* PL/SQL позволяет создавать 2 вида курсоров:
  + 1) Статические курсоры, SQL выражение для которых определяется на этапе компиляции:
    - Используются для DML команд
    - Могут быть явно объявлены и именованы
  + 2) Динамические курсоры, SQL выражение для которых определяется на этапе выполнения:
    - Могут использоваться для любых SQL выражений, включая DDL и DCL команды
    - Реализуются с помощью оператора EXECUTE IMMEDIATE или пакета dbms\_sql
* Неявные курсор – выполнение SQL выражения в секции исполнения или в секции исключений блока
* Операторы INSERT, UPDATE, DELETE, MERGE, SELECT INTO
* Не требуют объявления
* Не требуют OPEN, FETCH, CLOSE

1. Курсоры. Атрибуты курсора. Курсоры с параметрами.

* %ISOPEN — возвращает значение TRUE, если курсор открыт
* %FOUND — определяет, найдена ли строка, удовлетворяющая условию
* %NOTFOUND — возвращает TRUE, если строка не найдена
* %ROWCOUNT — возвращает номер текущей строки

1. Курсоры. Курсорные переменные. Параметры инстанса, связанные с курсорами.

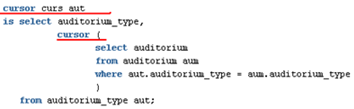
* Курсорная переменная, объявленная с помощью REF CURSOR без указания RETURN может быть связана с любым запросом
* Курсорная переменная, объявленная с помощью REF CURSOR с указанием RETURN может быть связана только с запросом, который возвращает результат точно соответствующий числу и типам данных в записи после фразы RETURN во время выполнения



Если курс. перем объяв с пом. REF CURSOR без **RETURN** – она мб связана с любым запросом, иначе – только с запросом, возвр результат точно соотв. числу и ТД в записи после фразы return

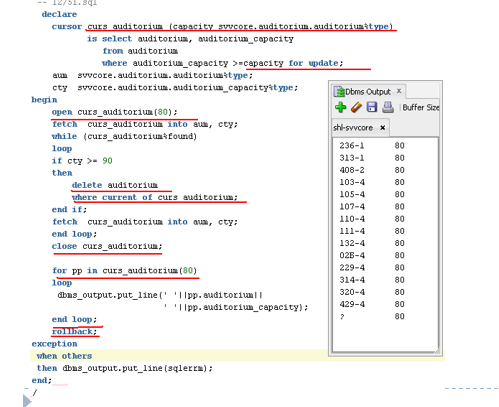
* **Курсорные переменные** - дополнительный уровень гибкости обеспечивают курсорные переменные (объявленные на основе типа REF CURSOR), которые позволяют передавать указатель на результирующее множество, полученное по запросу из одной программы в другую. Любая программа, имеющая доступ к такой переменной, сможет открывать и закрывать курсор, а также выбирать из него данные.

1. Курсоры. Курсорные подзапросы.



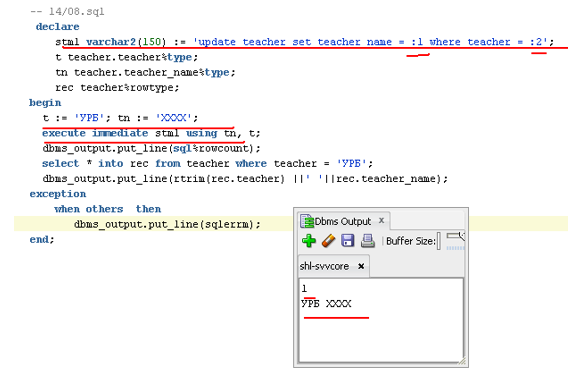
1. Курсоры. Использование конструкции CURRENT OF в курсорах.

Иногда, при выборке из курсора бывает ситуация, что какой-либо столбец или строки результирующего набора необходимо обновить. То есть, изменить их содержимое. Для того, чтобы это осуществить, непосредственно при объявлении курсора необходимо использовать конструкцию - **FOR UPDATE** (для обновления ..). А, так же конструкцию, **WHERE CURRENT OF** (где текущая строка ..) в операторах **UPDATE**, **DELETE**. Собственно конструкция **FOR UPDATE**, является частью оператора **SELECT** и объявляется последней:



1. Курсоры. Динамические курсоры.
   * + Динамические курсоры, SQL выражение для которых определяется на этапе выполнения:
       - Могут использоваться для любых SQL выражений, включая DDL и DCL команды
       - Реализуются с помощью оператора EXECUTE IMMEDIATE или пакета dbms\_sql

* EXECUTE IMMEDIATE - однострочные запросы и DDL команды
* OPEN FOR, FETCH и CLOSE - динамические многострочные запросы
* Синтаксис EXECUTE IMMEDIATE:
  + EXECUTE IMMEDIATE sql\_statement
  + [INTO {variable [,variable ...] | record}] [USING [IN | OUT | IN OUT] bind\_argument
  + [, [IN | OUT | IN OUT] bind\_argument . ..] ] [{RETURNING | RETURN} INTO bind\_argument [,bind\_argument]...];



1. Применение псевдостолбцов ROWID, ROWNUM в PL/SQL.

* **ROWID**-двухбайтовая величина, которая соответствует физическому положению любой строки БД. (**ROWID** (зашиф-й 64-битовым ключом уникальный адрес строки таблицы) - логический адрес записи)
* **ROWNUM**-используется для ограничения кол-ва выводимых записей.
* **ROWNUM** - логический номер записи. Позволяет управлять порядком и количеством извлекаемых записей. SELECT \* FROM employees WHERE ROWNUM < 10;

1. Обработка исключений в PL/SQL, стандартные исключения, генерация и обработка исключения.

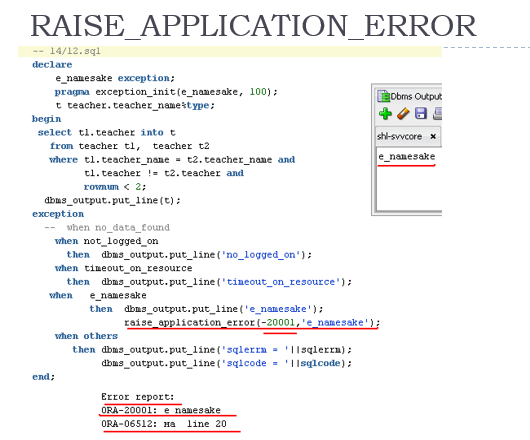
* Исключительная ситуация – событие, возникающее в программе и требующее незамедлительной обработки
* Два типа исключительных ситуаций:
  + 1) программно-определяемые исключения;
  + 2) предопределенные (стандартные) исключения.

\* sqlerrm – функция, возвр сообщ об ошибке

\* sqlcode – функция, возвр № ошибки

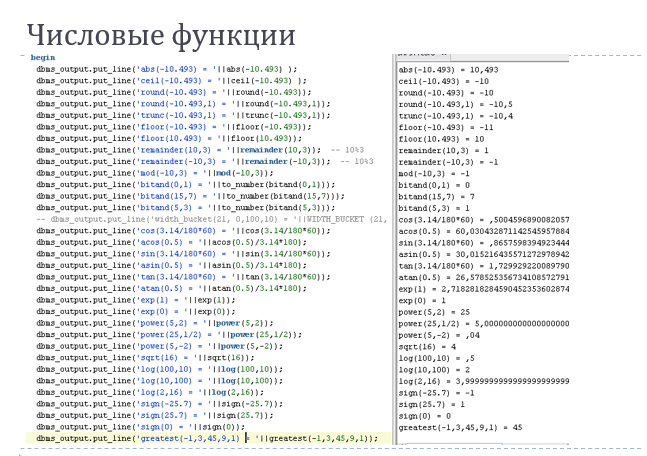
1. Принцип распространения исключений в PL/SQL. Инструкция RAISE\_APPLICATION\_ERROR.

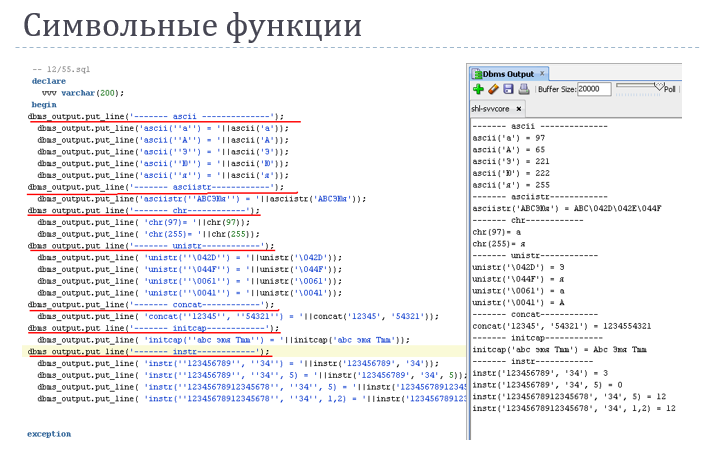
* **Инициировать исключение** — значит остановить выполнение текущего блока PL/SQL, оповещая исполняемое ядро об ошибке. Исключение может инициировать либо Oracle, либо ваш собственный программный код при помощи команды RAISE или процедуры RAISE\_APPLICATION\_ERROR.
* **Обработать исключение** — значит перехватить ошибку, передав управление обработчику исключения. Написанный программистом обработчик может содержать код, который в ответ на исключение выполняет определенные действия (например, записывает информацию об ошибке в журнал, выводит сообщение для пользователя или передает исключение во внешний блок).

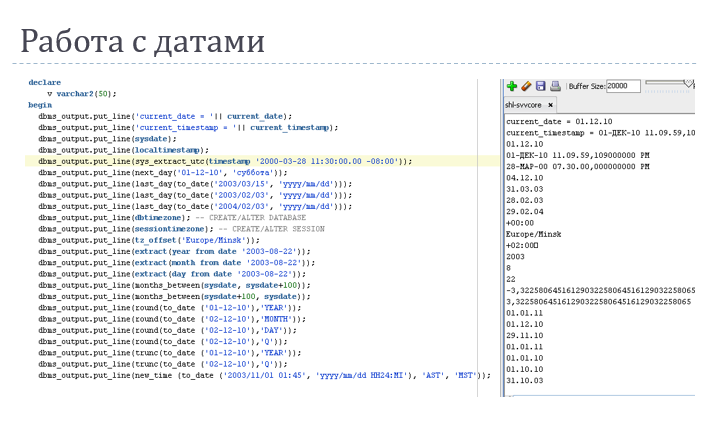
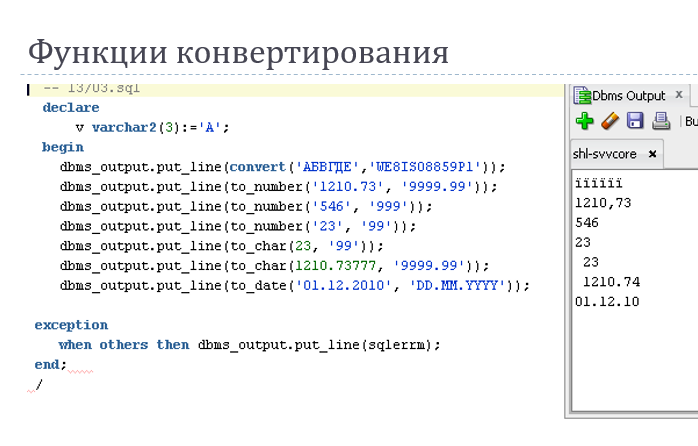


1. Встроенные функции языка PL/SQL. Функции работы с датами, текстом и числами.

* Числовые функции
* Символьные функции
* Функции по работе с датами
* Конвертирование
* Функции обработки ошибок







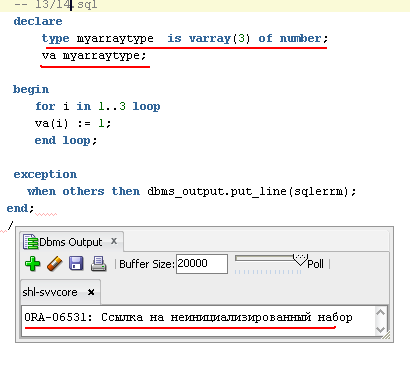
1. Встроенные функции языка PL/SQL. Функции регулярных выражений

Регулярные выражения - формальный язык поиска и осуществления манипуляций с подстроками в тексте, основанный на использовании метасимволов

* REGEXP\_LIKE выбирает все строки, соответствующие заданному шаблону
  + REGEXP\_LIKE (source\_char, pattern [, match\_parameter ] )
* REGEXP\_INSTR определяет местоположение вхождения шаблона в строку
  + REGEXP\_INSTR(source\_char, pattern [, start\_position [, nth\_appearance [, return\_option [, match\_parameter [, sub\_expression ] ] ] ] ] )
* REGEXP\_REPLACE заменяет шаблон выражения на заданный
  + - REGEXP\_REPLACE(source\_char, pattern [, replacement\_string [, start\_position [, nth\_appearance [, match\_parameter ] ] ] ] )
* REGEXP\_SUBSTR выделяет из строки шаблон
  + REGEXP\_SUBSTR(source\_char, pattern [, start\_position [, nth\_appearance [, match\_parameter [, sub\_expression ] ] ] ] )
* REGEXP\_COUNT определяет количество вхождений
  + REGEXP\_COUNT (source\_char, pattern [, position [, match\_param]])
* \*REGEXP\_LIKE выбирает все строки, соответствующие заданному шаблону
* \*REGEXP\_INSTR определяет местоположение вхождения шаблона в строку
* \*REGEXP\_REPLACE заменяет шаблон выражения на заданный
* \*REGEXP\_SUBSTR выделяет из строки шаблон
* \*REGEXP\_COUNT определяет количество вхождений

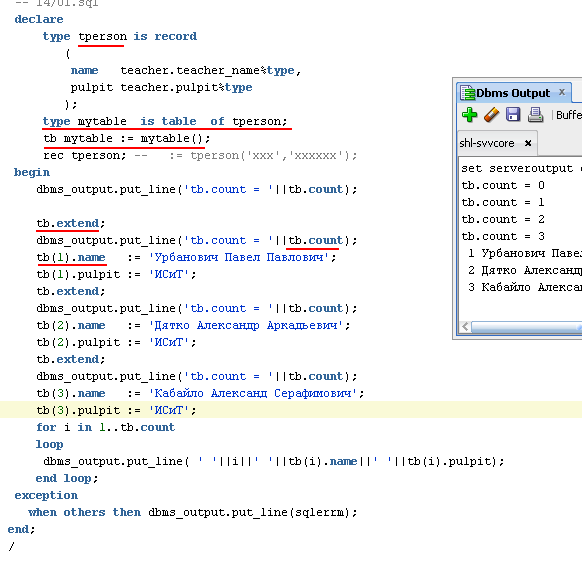
1. Коллекции. Массивы переменной длины.

* Коллекция – структура данных, содержащая элементы одного типа
* Элементом коллекции может быть как скалярная величина, так и композитные данные
* Элементы коллекций можно сравнивать между собой на эквивалентность
* Можно передавать параметром
* Одномерная, но можно создавать коллекции коллекций
  + Массив переменной длины VARRAY
  + Вложенная таблица (nested tables)
  + Ассоциативный массив (associative array)
* Коллекция состоит из **набора** элементов, причем каждый элемент находится в определенной позиции (имеется **индекс** элемента)
* Необходимо объявить **тип коллекции** – командой TYPE
* Необходимо объявить **коллекцию** – переменную этого типа для дальнейшего использования
* Коллекция называется **ограниченной**, если заранее определены границы возможных значений индексов ее элементов, иначе **неограниченной**
* Коллекции типа VARRAY всегда ограничены
* Вложенные таблицы и ассоциативные массивы неограничены
* Массивы переменной длины – одномерные, связанные коллекции однотипных элементов
* Доступны в рамках PL/SQL и в БД
* Являются плотными



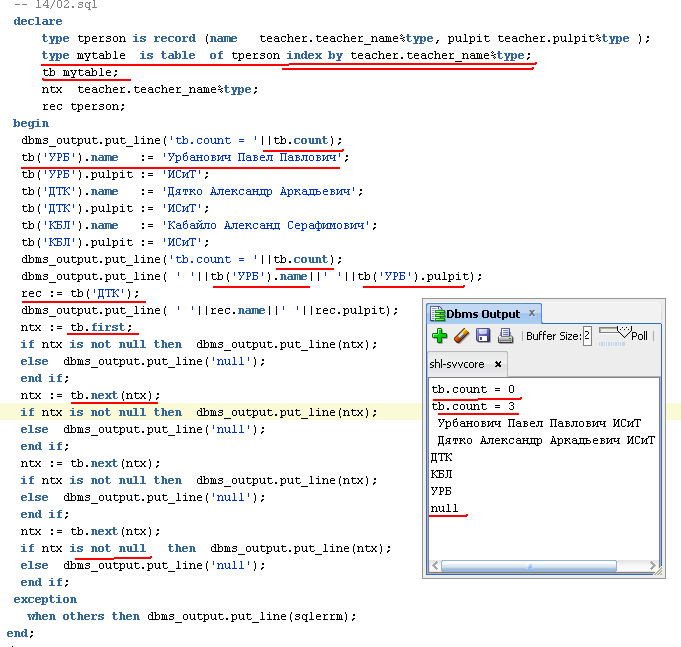
1. Коллекции. Вложенные таблицы.

* Коллекция – структура данных, содержащая элементы одного типа
* Элементом коллекции может быть как скалярная величина, так и композитные данные
* Элементы коллекций можно сравнивать между собой на эквивалентность
* Можно передавать параметром
* Одномерная, но можно создавать коллекции коллекций
  + Массив переменной длины VARRAY
  + Вложенная таблица (nested tables)
  + Ассоциативный массив (associative array)
* Коллекция состоит из **набора** элементов, причем каждый элемент находится в определенной позиции (имеется **индекс** элемента)
* Необходимо объявить **тип коллекции** – командой TYPE
* Необходимо объявить **коллекцию** – переменную этого типа для дальнейшего использования
* Коллекция называется **ограниченной**, если заранее определены границы возможных значений индексов ее элементов, иначе **неограниченной**
* Коллекции типа VARRAY всегда ограничены
* Вложенные таблицы и ассоциативные массивы неограничены
* Вложенные таблицы – одномерные, несвязанные коллекции однотипных элементов
* Доступны в рамках PL/SQL и как поля таблицы в БД
* Изначально являются плотными, но могут впоследствии становиться разреженными



1. Коллекции. Ассоциативные массивы.

* Коллекция – структура данных, содержащая элементы одного типа
* Элементом коллекции может быть как скалярная величина, так и композитные данные
* Элементы коллекций можно сравнивать между собой на эквивалентность
* Можно передавать параметром
* Одномерная, но можно создавать коллекции коллекций
  + Массив переменной длины VARRAY
  + Вложенная таблица (nested tables)
  + Ассоциативный массив (associative array)
* Коллекция состоит из **набора** элементов, причем каждый элемент находится в определенной позиции (имеется **индекс** элемента)
* Необходимо объявить **тип коллекции** – командой TYPE
* Необходимо объявить **коллекцию** – переменную этого типа для дальнейшего использования
* Коллекция называется **ограниченной**, если заранее определены границы возможных значений индексов ее элементов, иначе **неограниченной**
* Коллекции типа VARRAY всегда ограничены
* Вложенные таблицы и ассоциативные массивы неограничены
* Ассоциативные массивы – одномерные, неограниченные (по максимальному количеству элементов при создании) коллекции элементов
* Доступны только в рамках PL/SQL
* Изначально являются разреженными, индекс может принимать непоследовательные значения



1. Процедурные объекты. Хранимые процедуры. Вызов процедур. Входные и выходные параметры, позиционный и параметрический форматы передачи фактических параметров. Значения параметров по умолчанию.

* *Процедура – именованный модуль, который выполняет одно или несколько выражений и может принимать или возвращать значения через список параметров*

**Типы парам:**

\* in

\* out – при выполнении устан в null

\* in out – при выполн ост. неизменными

**Значение по умолч:**

in, in out – можно не задавать при вызове

**Передача парам:**

* Позиционный – каждое значение в списке аргументов вызова ставится в соответствие формальному параметру по порядку. *Empid\_to\_name(23, name, surname);*
* Именованный – явно связывает аргументы при вызове с параметрами по именам. *Empid\_to\_name(in\_id =>23, out\_name=> name, out\_surname =>surname);*

\* можно комбин оба метода, пока позиционные стоят слева

**Вызов процедуры:**

conn -> procedures -> выбрать -> Run

exec

1. Процедурные объекты. Хранимые функции. Параметры функции. Вызов функций. Понятие детерминированной функции. Понятие pipeline функции. Значения параметров по умолчанию.

* *Функция – именованный модуль, который выполняет ноль или более выражений через фразу Return*

**Вызов:**

- в присвоении нач. значения пер-ной

- в выражении присвоение

- в булевом выраж

- в SQL запросе

- как аргум в списке парам др. функции / процедуры

* **DETERMINISTIC – функция детерминирована, если она возвращает одно и то же значение при вызове с теми же параметрами**
* **Конвейерная функция или pipelined-функция** – табличная функция, которая поставляет свои результаты клиенту по мере их подготовки. Не материализуют весь результирующий набор.
* От обычной функции отличается ключевой конструкцией pipelined.
* Обязательно должна возвращаться коллекция
* Её заполнение происходит по мере выполнения через pipe row (в pipe row указывается элемент коллекции)
* Как только программа достигает участка с pipe row, результат будет передан в вызывающую среду. При этом вызывающая сторона уже может что-то делать с этой строкой.

1. Процедурные объекты. Пакеты. Спецификация и реализация пакета.

Пакеты - коллекция PL/SQL объектов, сгруппированных вместе.

Преимущества:

* Скрытие информации
* Объектно-ориентированный дизайн
* Постоянство объектов в транзакциях
* Улучшенная производительность

Можно включать в пакет: процедуры, функции, константы, исключения, курсоры, переменные, TYPE выражения, записи, REF курсоры

* Спецификация пакета (package) – обязательна, содержит список объектов для общего доступа из других модулей или приложения
* Реализация пакета (package body) – содержит весь программный код для реализации процедур и функций и спецификации, приватные объекты и секцию инициализации

1. Процедурные объекты. Триггеры. Виды триггеров. Классификация, порядок выполнения и предикаты триггеров. Триггеры замещения. Привилегии для создания триггеров. Включение/отключение триггеров. Псевдозаписи old и new.

*Триггер* – особый вид процедур, кот. срабат по запускающему их событию

**Виды триггеров:**

\* *по привяз. объекту:* на табл, на предст

\* *по событиям запуска:* insert, update, delete

\* *по обл д-вия:* уровень оператора, ур. записи, составные

\* *по t сраб:* before (до записи в журнал), after

**Уровни триггеров:**

\* for each row – сраб для каждой измененной строки

\* по умолч (операторный ур) – сраб 1 раз на триггерное событие

**Порядок вып:**

\* операторные BEFORE

\* для к. строки BEFORE

\* вып. оператор

\* для к. строки AFTER

\* операторный AFTER

**Предикаты триггеров:**

create or replace trigger ..

after insert or update or delete on ..

begin

if inserting then …

elsif updating then …

elsif deleting then …

**Триггеры замещения:** INSTEAD OF – только для предст! только уровня строки!

**Привилегии:**

- create (any) trigger

- alter any trigger

- drop any trigger

- administer database trigger – созд/изм/уд системные триггеры

**Вкл / откл триггеров:**

**\*** alter trigger { disable | enable }

**\*** всех д/табл : alter table .. { enable | disable } all triggers;

**\*** компиляция триггера : alter trigger .. compile

**\*** переименование триггера

**Псевдозаписи new, old:**

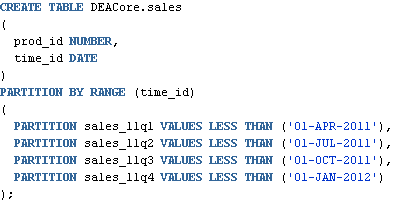
| операция срабатывания | OLD.column | NEW.column |
| --- | --- | --- |
| insert | null | новое знач |
| update | старое знач | новое знач |
| delete | старое знач | null |

1. Секционирование таблиц. Виды секционирования.

Метод, позволяющий хранить сегмент данных, такой как таблица, в виде нескольких сегментов, сохраняя логическую монолитную структуру

*USER, ALL и DBA представления*

* …\_TABLES – информация о таблицах
* …\_PART\_TABLES - информация о секционированных таблицах
* …\_ TAB\_PARTITIONS - информация о табличных секциях
* …\_PART\_KEY\_COLUMNS – информация о ключах секционирования
* …\_SEGMENTS - информация о сегментах хранения
* …\_OBJECTS - информация о объектах БД
* Разные сегменты могут находиться в разных ТП , а значит на разных дисках
* Различные секции-сегменты при общности логической структуры могут иметь собственные физические атрибуты
* Отдельные ТП находиться в состоянии OFFLINE, не нарушая работоспособности всей таблицы



**Диапазонное (range) –** секционир., при кот.для кажд.секции опред.диап.знач. ключа секционирования.

\*ключ м.приним.знач даты и времени, числа, текста

! при загрузке новых д-х, необх.посто.расщеплять секцию maxvalue;

\*для задания диап.исп.ключ.слово *less than*

Пример:

create table T\_RANGE( docnum number) partition by range(docnum)

(partition n1 values less than(10) tablespace t1,

partition n2 values less than(20) tablespace t2,

partition n3 values less than(30) tablespace t3);

**Интервальное –** при загрузке новых д-х в табл.авто созд.новые секции для нового диап.ключей.

\*новые секции будут созд.авто (по 1ой оп-ции insert, не попад.в диап.секций)

create table T\_INTERVAL( t\_date date) partition by range(t\_date)

interval(NUMTOYMINTERVAL(3, 'MONTH'))

(partition d1 values less than (to\_date('01.01.2016', 'DD/MM/YYYY')),

partition d2 values less than (to\_date('01.03.2016', 'DD/MM/YYYY')));

**Хэш-секц** позвол.равномерн.распред.стоки между секциями, т.е.разбросать строки по разным секциям и сделать эти секции равновеликими.

!не означ, что строки распред.по секцияям случайным образом

create table T\_HASH( doctype varchar2(5))

partition by hash(doctype)

(partition p1 tablespace th1,

partition p2 tablespace th2)

**Списочное** – позвол.разбить таблицу по списку конкр.знач.

\*ключ спис.секц.мб только 1столбцовым

\*default – опис.все знач, не попавшие в другие списки

\*null – мб ключ.знач.

create table T\_LIST (l1 char(10)) partition by list(l1)

(partition ch1 values('a', 'b') tablespace tl1,

partition ch2 values('c') tablespace tl2);

**Композитное -** слишком большие таблицы – секционирование по 2 ур. (секции и подсекции)

1. Транзакции. Виды транзакций. Понятие автономной транзакции.

**Транзакция** – неск. послед. инструкций SQL, кот. рассм как единое целое

**Операторы:**

\* **COMMIT** – фиксир все измен для тек. транз

\* **ROLLBACK** – откат незафикс. изменений: читает инфу из сегментов отката и восст блоки д-х в сост, в кот. они находились до нач. транз, освобожд блокировки

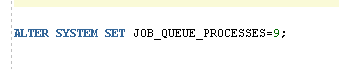
\* **SAVEPOINT** – в одной транз мб неск, созд возможность отмены только части работы, продел. в транз. Освоб. блокировки, кот. были установлены отмененным оператором

\* **SET TRANSACTION** – устан атриб транз – уровень изолир, будет исп д/чтения или записи

**Автономные транз** – позв созд *подтранзакции*, кот. м сохр/отмен изменения вне завис. от родительской транз

1. Обработка заданий. Системные пакеты обработки заданий в Oracle.

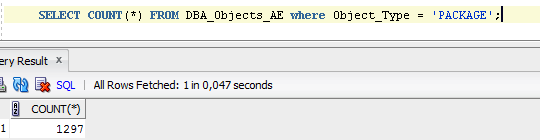
* DBMS\_JOB – поддержка управления заданиями
* Задание – процедура, Pl-SQL блок, внешняя процедура
* Выполняется в фоновом режиме, надо задать количество одновременно выполняемых процессов



* SUBMIT – создание задания
* ISUBMIT – создание задания с номером
* REMOVE – удаление задания из очереди
* RUN – немедленное выполнение задания в пользовательском сеансе
* BROKEN – разрушение задания (16)
* INSTANCE – указание экземпляра
* NEXT\_DATE – изменение времени выполнения
* INTERVAL – изменение интервала выполнения
* CHANGE – изменение параметров задания
* WHAT – изменение задания
* Представления словаря:
  + DBA\_JOBS
  + USER\_JOBS
  + USER\_SCHEDULER\_SCHEDULES
  + USER\_SCHEDULER\_PROGRAMS
  + USER\_SCHEDULER\_JOBS
  + USER\_SCHEDULER\_JOB\_LOG

1. Системные пакеты Oracle.

* Встроенные системные пакеты:
* Устанавливаются во время установки Oracle
* Используются для расширения основных функциональных возможностей Oracle
* Владелец пакетов - SYS (обычно, есть и другие)
* Написаны на C или на PL/SQL



**Пакеты APEX:**

- apex – oracle application express – среда разработки веб-прил

- apex\_custom\_auth – проверка подлинности управл сеансом

- apex\_application – использ глоб пер-ных

- apex\_item – созд эл-тов форм на основе sql-запроса

- apex\_util – различ утилиты состояния сеанса, файлов, авториз …

**Пакеты DBMS:**

- dbms\_advanced\_rewrite – перехват и замена sql-запросов

- dbms\_advisor – часть набора экспертной с-мы для реш проблем произв-сти, связ. с компонентами сервера БД

- dbms\_sqltune – сбор статистики, исп при анализе произв. операторов

- dbms\_appl\_info – присвоение имени процессу для удобства мониторинга и отладки