МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
ДОНЕЦКОЙ НАРОДНОЙ РЕСПУБЛИКИ
ГОУ ВПО «Донецкий национальный университет»
Кафедра компьютерных технологий

БОНДАРЕНКО В.И.

СУБД ORACLE

УЧЕБНО-МЕТОДИЧЕСКОЕ ПОСОБИЕ

Донецк 2020

УДК 004.65(072)
ББК 3973.233-018.2р30
Б811

Рекомендовано к изданию Ученым советом
ГОУ ВПО «Донецкий национальный университет»
(протокол № 10 от 25.12.2020 г.)

Бондаренко В.И. СУБД Oracle: учебно-методическое пособие / В.И. Бондаренко – Донецк: ДонНУ, 2020. – 104 с.

Рецензенты:

Головчан А.В., кандидат физико-математических наук, заместитель директора по научной работе, ГУ «Донецкий физико-технический институт им. А.А. Галкина»;

Гридин С.В., кандидат технических наук, Доцент кафедры промышленной теплоэнергетики, ГОУ ВПО «Донецкий национальный технический университет».

В учебно-методическом пособии изложены теоретические основы архитектуры системы управления базами данных Oracle и языкового расширения PL/SQL. В издании приводится краткое описание каждой темы и несколько вариантов заданий к ней. Имеются примеры, разъясняющие теоретический материал, приведены четыре лабораторные работы. Учебно-методическое пособие содержит рисунки, схемы и таблицы, что облегчает восприятие материала.

Учебно-методическое пособие предназначено для студентов четвертого и третьего ускоренного курсов физико-технического факультета направления подготовки 09.03.01 «Информатика и вычислительная техника» всех форм обучения, изучающих систему управления базами данных Oracle.

УДК 004.65(072)
ББК 3973.233-018.2р30
Б811

ОГЛАВЛЕНИЕ

ПРЕДИСЛОВИЕ

Учебная дисциплина «СУБД Oracle» относится к вариативной части профессионального блока и состоит из двух содержательных модулей. Основывается на базе дисциплин: «Основы программирования», «Информатика и информационно-коммуникационные технологии», «Дискретная математика», «Базы данных».

Учебно-методическое пособие разбито на 4 раздела, отражающих основные темы курса, указанные в рабочей программе:

1. Установка и первый запуск. Настройки серверной и клиентской частей.

2. Основные понятия архитектуры. Основные объекты БД Oracle.

3. Процедурное языковое расширение PL/SQL.

4. Создание приложений в ORACLE APEX.

Теоретический курс, изложенный в данном учебном издании, предназначен для формирования у студентов знаний и навыков в области программирования на языках SQL, PL/SQL, администрирования баз данных в СУБД Oracle для выполнения разработки БД и дальнейшего сопровождения.

ВВЕДЕНИЕ

Выполнение лабораторных работ студентами направления подготовки 09.03.01 «Информатика и вычислительная техника» направлено на формирование знаний и умений обучающихся по архитектуре СУБД Oracle; получение навыков использования основных структур базы данных в СУБД Oracle; обучение студентов основным техническим приемам администрирования баз данных в СУБД Oracle; обучение языку SQL и процедурному языку PL/SQL.

Учебно-методическое пособие построено следующим образом. Вначале приводится краткая теоретическая часть, знание которой необходимо для выполнения лабораторной работы. После теоретической части формулируются задания в виде лабораторных работ. В конце лабораторной работы приводятся контрольные вопросы по соответствующей теме.

Выполнение лабораторной работы содержит этапы:

1) получение студентом у преподавателя задания на лабораторную работу;

2) изучение теоретической части работы, приведенной в учебно-методическом пособии, и самостоятельный анализ литературных источников;

3) выполнение этапов проектирования и написания программы;

4) тестирование и отладка программного продукта;

5) оформление и защита отчета по лабораторной работе.

Оформленный отчет по лабораторной работе должен содержать:

1) название темы;

2) цель лабораторной работы;

3) распечатку программы и результатов;

4) ответы на контрольные вопросы, приведенные в конце каждой лабораторной работы.

В процессе подготовки лабораторной работы студент *должен*:

1) закрепить знания и навыки, полученные при изучении теоретической части;

2) получить навыки самостоятельного решения конкретной задачи посредством использования языка программирования PL/SQL.

После выполнения всех лабораторных работ студент *должен приобрести следующие навыки:* создавать реляционные и объектно-реляционные базы данных; писать SQL-запросы, манипулировать реляционными данными; писать программы на языке PL/SQL; писать программы для работы с объектно-реляционными базами данных; выполнять настройку SQL; решать основные задачи администрирования Oracle; создавать хранимые процедуры, функции, пакеты, триггеры на языке PL/SQL в инструментах SQL*Plus и Oracle SQL Developer; выполнять настройку SQL с помощью SQL Developer и SQL*Plus; администрировать базу данных с помощью Oracle Enterprise Manager, разрабатывать ориентированные на использование базы данных веб-приложений с помощью технологии APEX.

ТЕМА 1. УСТАНОВКА И ПЕРВЫЙ ЗАПУСК. НАСТРОЙКИ СЕРВЕРНОЙ И КЛИЕНТСКОЙ ЧАСТЕЙ

1.1 Основные понятия и определения

Программное обеспечение (ПО) — программа или множество программ, используемых для управления компьютером. Программное обеспечение является одним из видов обеспечения вычислительной системы, наряду с техническим (аппаратным), математическим, информационным, лингвистическим, организационным, методическим и правовым обеспечением.

Редакцией (Ediition) называют выпуск программы, обладающий значительными отличительными характеристиками для пользователя. Редакция – это разновидность программы в пределах одной версии.

Дистрибутив – это форма распространения программного обеспечения. Дистрибутив обычно содержит программы для начальной инициализации системы. Наличие дистрибутивов – это следствие того, что форма программного обеспечения, используемая для его распространения, почти никогда не совпадает с формой программного обеспечения на работающей системе. Дистрибутив ПО — это комплект (как правило, набор файлов), приспособленный для распространения ПО. Может включать вспомогательные инструменты для автоматической или автоматизированной начальной настройки ПО (установщик). Так и при использовании дистрибутива программного обеспечения — устанавливаются только необходимые файлы, причём таким образом, чтобы их правильно видела операционная система. Также конфигурируются начальные параметры, язык, способ подключения, например, к Интернету.

База данных – это набор данных. Oracle позволяет сохранять данные и получать к ним доступ в соответствии с моделью, называемой реляционной. В связи с этим Oracle называют системой управления

реляционными базами данных (РСУБД). Чаще всего под базой данных подразумевают не только физические данные, но также и комбинацию физических объектов, объектов памяти и процессов.

Экземпляром (или сервером) БД называется набор структур памяти и фоновых процессов, обращающихся к группе файлов базы данных. К одной базе данных могут обращаться несколько экземпляров (свойство Real Application Clusters). Экземпляр может смонтировать и открыть только одну базу данных в каждый момент времени.

1.2 Установка Oracle

1.2.1 Отличия в редакциях Oracle

Oracle Enterprise Edition. Флагманский продукт, ориентирован на крупномасштабные проекты, нуждающиеся в полном наборе средств Oracle. Только Enterprise Edition поддерживает такие развитые механизмы обеспечения безопасности, как виртуальная частная база данных (Virtual Private Database, VPD), детальный аудит (FineGrained Auditing) и другие опции, включая Database Vault, Advanced Security и Label Security. Лишь в Enterprise Edition хранилища данных поддерживают сжатие повторяющихся значений, кроссплатформенные переносимые табличные пространства, управление жизненным циклом информации (Information Lifecycle Management, ILM), перезапись запросов с материализованными представлениями, а также секционирование (Partitioning), OLAP и добычу данных (Data Mining). К числу механизмов обеспечения высокой доступности, включенных в Enterprise Edition, относятся Data Guard, ретроспективные (flashback) базы, ретроспективные таблицы и ретроспективные транзакции. В Oracle Database llg добавлена опция сжатия Advanced Compression Option для любой рабочей нагрузки, в том числе для обработки транзакций, хранения больших объектов (Large Object, LOB) и

резервных копий; подсистема тестирования базы данных, которая называется Real Application Testing Option и включает в себя программы Database Replay и SQL Performance Analyzer, а также опция Total Recall Option, обеспечивающая режим архивации ретроспективных данных Flashback Data Archive, который сохраняет данные, необходимые для выполнения хронологических запросов (запросов с конструкцией AS OF, где задается дата в прошлом).

Oracle Standard Edition. Эта СУБД ориентирована на реализацию баз данных малого и среднего размера. Ее можно развернуть в серверной конфигурации, имеющей до 4 ЦП, на одном компьютере или на кластере с использованием подсистемы Real Application Clusters (RAC).

Standard Edition One. Ориентированная на небольшие проекты, эта СУБД поддерживает до двух ЦП и не поддерживает RAC. В остальном набор возможностей схож с реализованным в редакции Oracle Standard Edition.

Oracle Personal Edition. СУБД, используемая разработчиками-одиночками для создания кода, который будет выполняться в многопользовательской СУБД. В отличие от Express Edition, требует лицензии, но обладает всей функциональностью Enterprise Edition.

Oracle Express Edition. СУБД начального уровня, доступная для Windows и Linux бесплатно. Может использовать не более 1 Гбайт памяти и 4 Гбайт дискового пространства. Предоставляет часть функциональности, включенной в редакцию Standard Edition One. Отсутствуют такие функции, как виртуальная Java-машина, управляемое сервером резервное копирование и восстановление, а также подсистема Automatic Storage Management. Oracle Enterprise Manager не умеет управлять этой СУБД, однако ее можно развернуть так, что она будет доступна из административного интерфейса Oracle Application Express (бывший HTML-DB), позволяющего управлять несколькими пользователями.

1.2.2 Метрики лицензирования

Доступны 2 метрики лицензирования Oracle Database:

- Named User Plus – по пользователям, можно конкретно подсчитать точное количество пользователей системы
- Processor – по процессорной мощности серверов на неограниченное количество пользователей.

1.2.3 Виды дистрибутивов

Oracle Database предлагает лучшую на рынке производительность, масштабируемость, надежность и безопасность как в локальной, так и в облачной среде.

Oracle Database 19c — это текущий **долгосрочный выпуск**, который обеспечивает высочайший уровень стабильности выпуска и самые длительные сроки для поддержки и исправления ошибок.

Oracle Database 21c – **последняя версия**, которая позволяет опробовать многие улучшения и новые возможности. К ним относятся улучшенная поддержка нескольких моделей с помощью встроенного в базу данных Javascript и собственных таблиц блокчейнов, а также усовершенствования для совместной работы, такие как AutoML и улучшения сегментирования, которые будут включены в будущие долгосрочные выпуски.

1.2.4 Процесс установки Oracle Enterprise и Express Edition

Подготовка к установке

Проверяется, соответствует ли используемая операционная система минимальным требованиям для установки и работы Oracle Database 18c XE. Допустимы следующие версии ОС Windows:

- Windows 7 x64 – Professional, Enterprise, and Ultimate editions
- Windows 8.1 x64 – Pro and Enterprise editions
- Windows 10 x64 – Pro, Enterprise, and Education editions
- Windows Server 2012 x64 – Standard, Datacenter, Essentials, and Foundation editions
- Windows Server 2012 R2 x64 – Standard, Datacenter, Essentials, and Foundation editions
- Windows Server 2016 x64 – Standard, Datacenter, and Essentials editions

Для установки требуется минимум 2 Гб оперативной памяти, минимум 8,5 Гб дискового пространства для СУБД Oracle, 2 Гб дискового пространства для хранения временных файлов, и обладание правами администратора.

Если системные требования соответствуют, то скачивается установочный файл. Oracle Database Express Edition (XE) распространяется бесплатно и установочный файл можно скачать с официального сайта Oracle: https://www.oracle.com/database/technologies/xe-downloads.html

Для скачивания на портале Oracle необходимо наличие учетной записи с паролем. При ее отсутствии осуществляется регистрация новой учетной записи.

Пройдя по ссылке, выбирается версия Oracle Database 18c XE для вашей операционной системы: Oracle Database 18c Express Edition for Windows x64. Запускается скачивание zip архива (OracleXE184_Win64.zip). Время скачивания может быть достаточно большим. Объем архива 1,9 Гб.

Установка СУБД

Распаковывается скачанный архив. Среди извлеченных файлов ищется и запускается файл, под названием setup.exe. Запустится окно установщика (рис.1.1). Нажимаем Next.

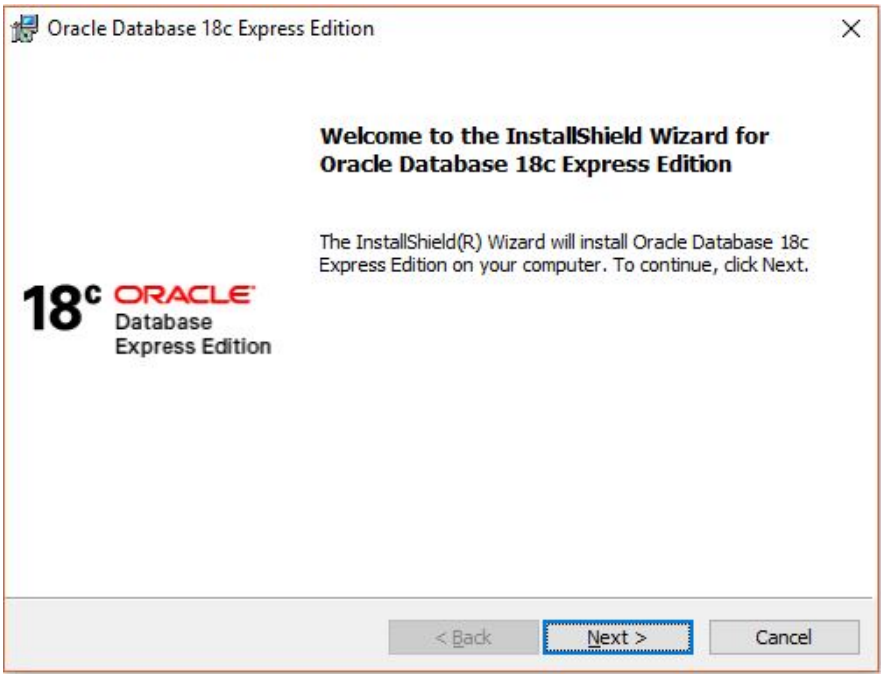


Рисунок 1.1 – Запуск установки Oracle

Дается согласие с лицензионным соглашением (I accept the terms in the license agreement). Нажимаем Next.

Выбирается каталог, в который будет установлена СУБД Oracle Database 18c XE. Можно оставить каталог по умолчанию, или же выбрать собственный. Нажимаем Next.

Далее указываются пароли для учетных записей SYS, SYSTEM и PDBADMIN, которые понадобятся для подключения к базе данных (БД) в дальнейшем (рис.1.2).

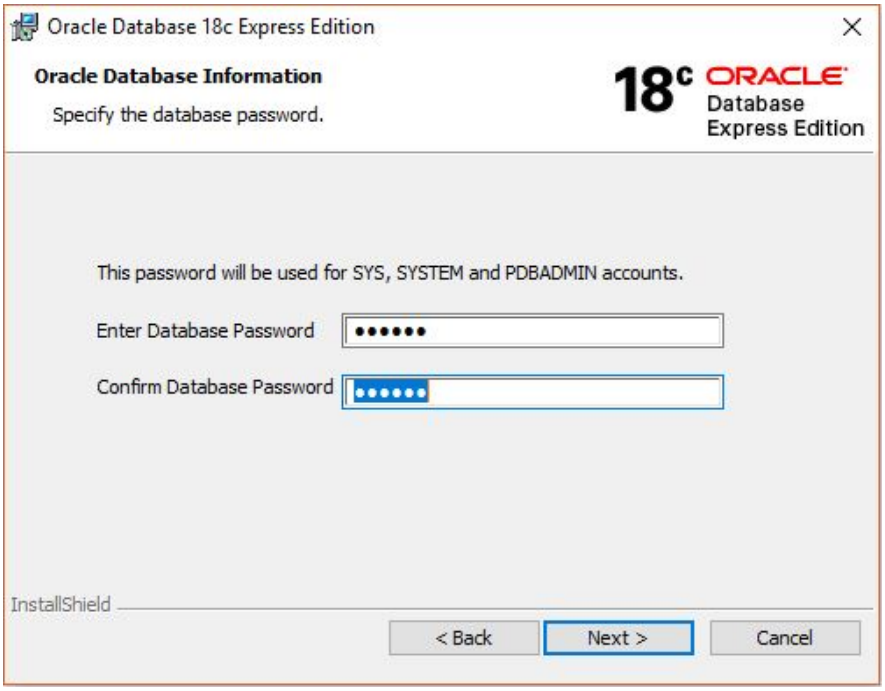


Рисунок 1.2 – Задание паролей

На следующем шаге проверяется (и корректируется опцией Back в случае необходимости) правильность введенной ранее информации и нажимается Install. Это запускает процесс установки СУБД.

Во время установки система запрашивает разрешение на доступ к сетям от Java платформы (рис.1.3). При отказе в доступе некоторые функции Oracle Database 18c XE не будут доступны.

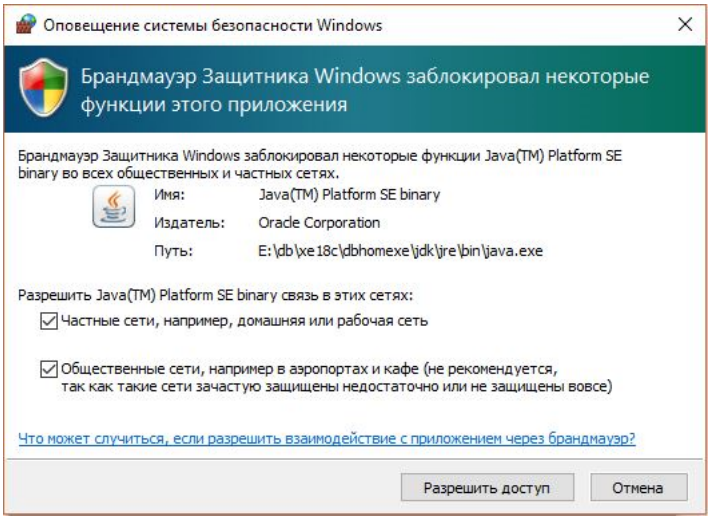


Рисунок 1.3 – Запрос на доступ к Java

Завершение установки зафиксирует в соответствующем окне необходимую информацию для подключения к установленной базе данных (рис.1.4).

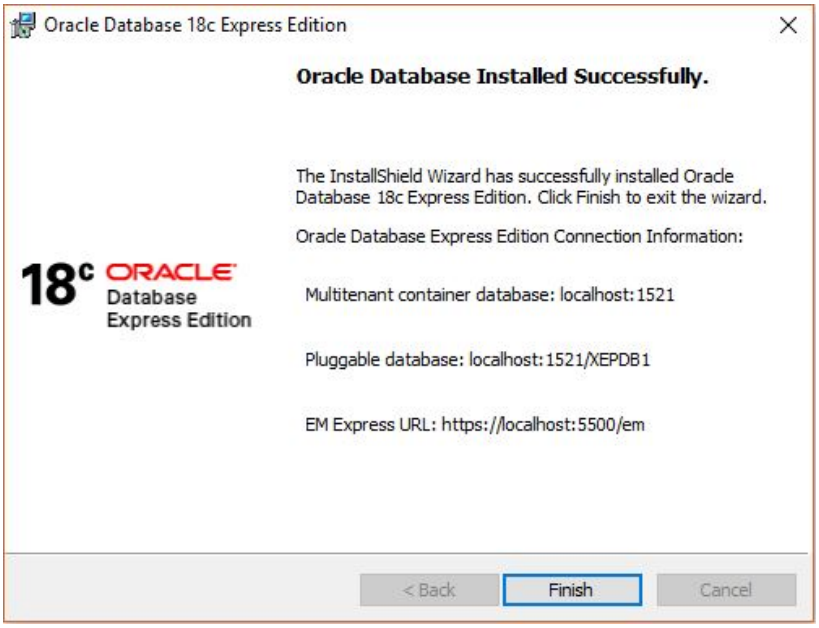


Рисунок 1.4 – Успешное завершение установки

Подключение к СУБД

После установки необходимо произвести подключение к БД, что позволит проверить работоспособность Oracle Database 18c XE. Подключиться можно с помощью SQL Developer и с помощью командной строки – SQLPlus.

Подключение с помощью SQL Developer.

Запускается SQL Developer и нажимается зеленый + в левом верхнем углу (рис 1.5).

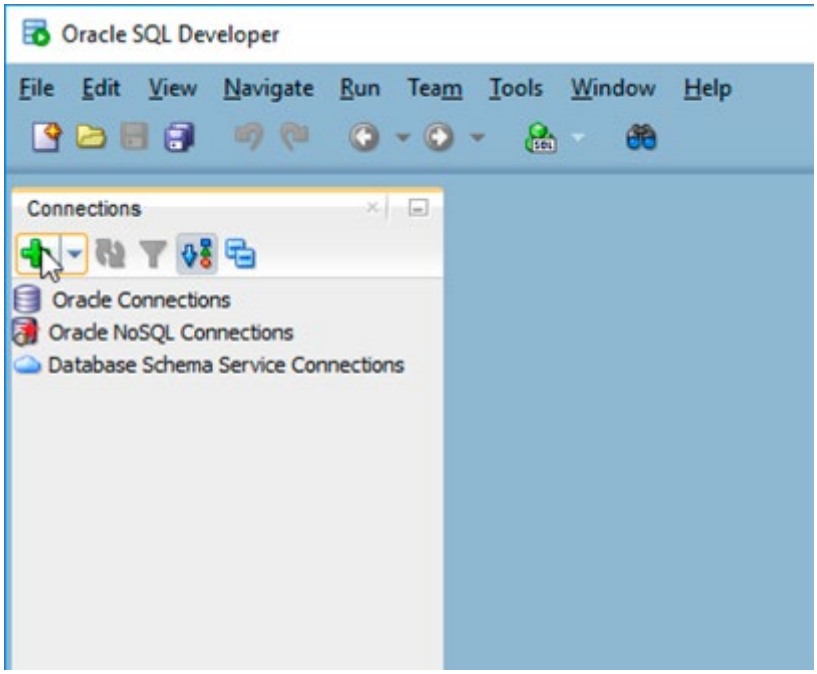


Рисунок 1.5 – Список подключений

Вводятся необходимые данные для подключения (рис. 1.6).

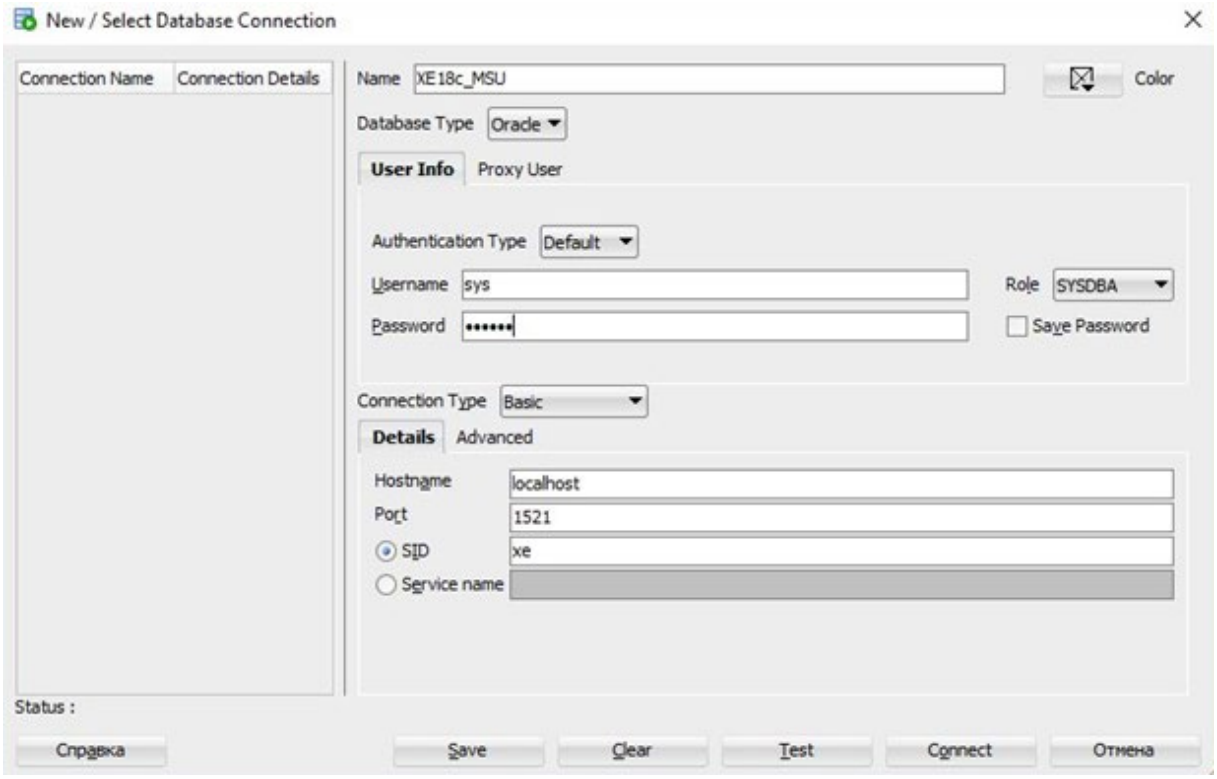


Рисунок 1.6 – Данные подключения

Все данные указывались во время установки базы данных. Если подключение осуществляется под пользователем SYS, необходимо в графе

роль выбрать SYSDBA.

Нажимается кнопка «Connect». Если все указано правильно, то SQL Developer открывает окно, которое позволяет выполнять SQL запросы к выбранной базе данных (рис.1.7).

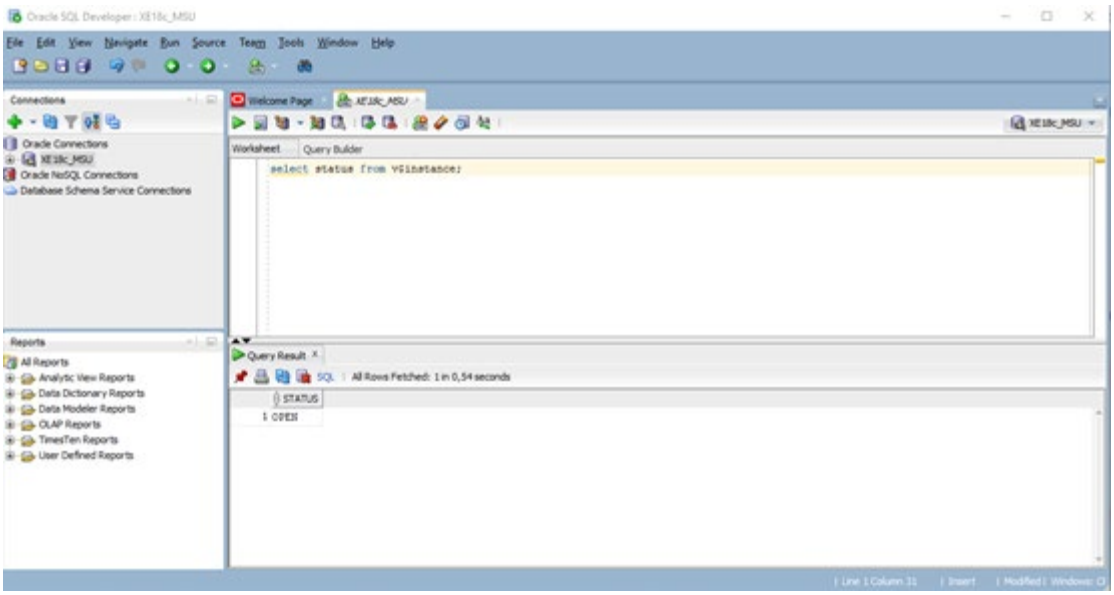


Рисунок 1.7 – Работа с подключенной БД

Подключение с помощью командой строки.

Для подключения с помощью командной строки – SQLPlus запускается командная строка и набирается команда sqlplus / as sysdba, которая подключает к БД под пользователем SYS (рис.1.8).

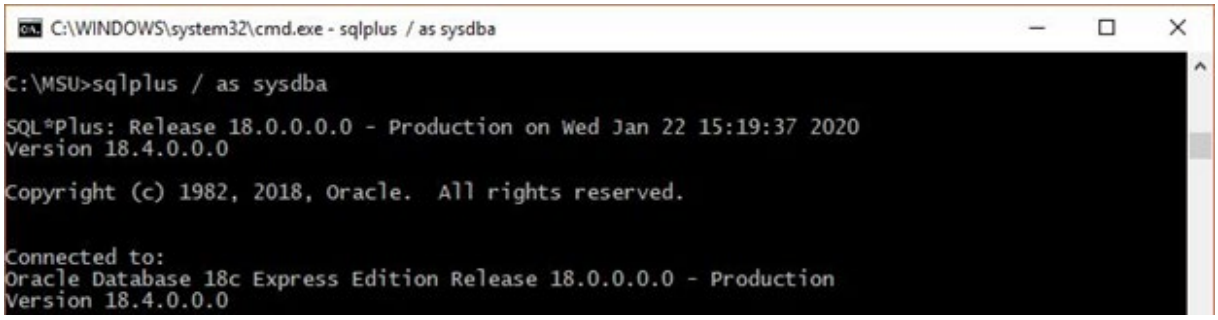


Рисунок 1.8 – Подключение с помощью командной строки

На этом установка и базовые настройки Oracle Database 18c Express Edition на ОС Windows завершены.

1.3 Первый запуск и настройка сервера и клиентских программ

1.3.1 Проверка установленного семейства программ

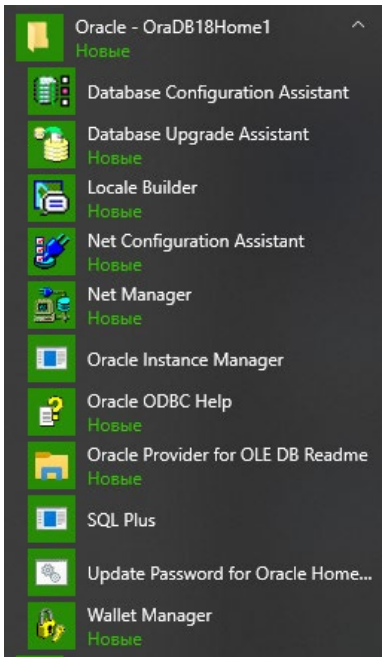


Рисунок 1.9 – Установленное семейство программ

1.3.2 Настройка сервисов

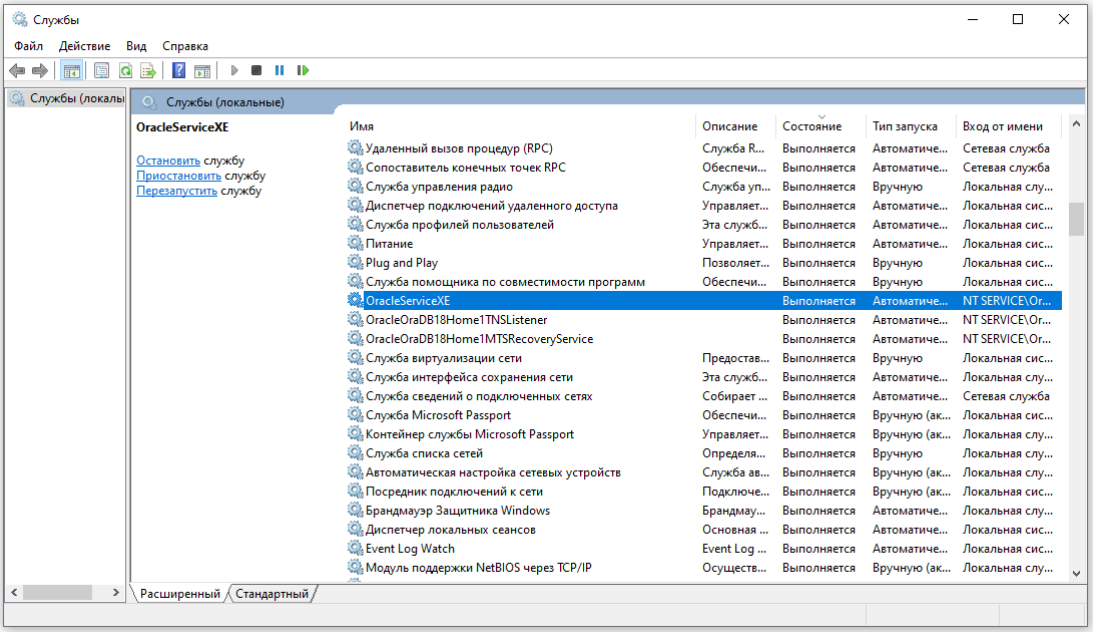


Рисунок 1.10 – Проверка служб

1.3.3 Создание пользователей и групп. Системные пользователи и привилегии

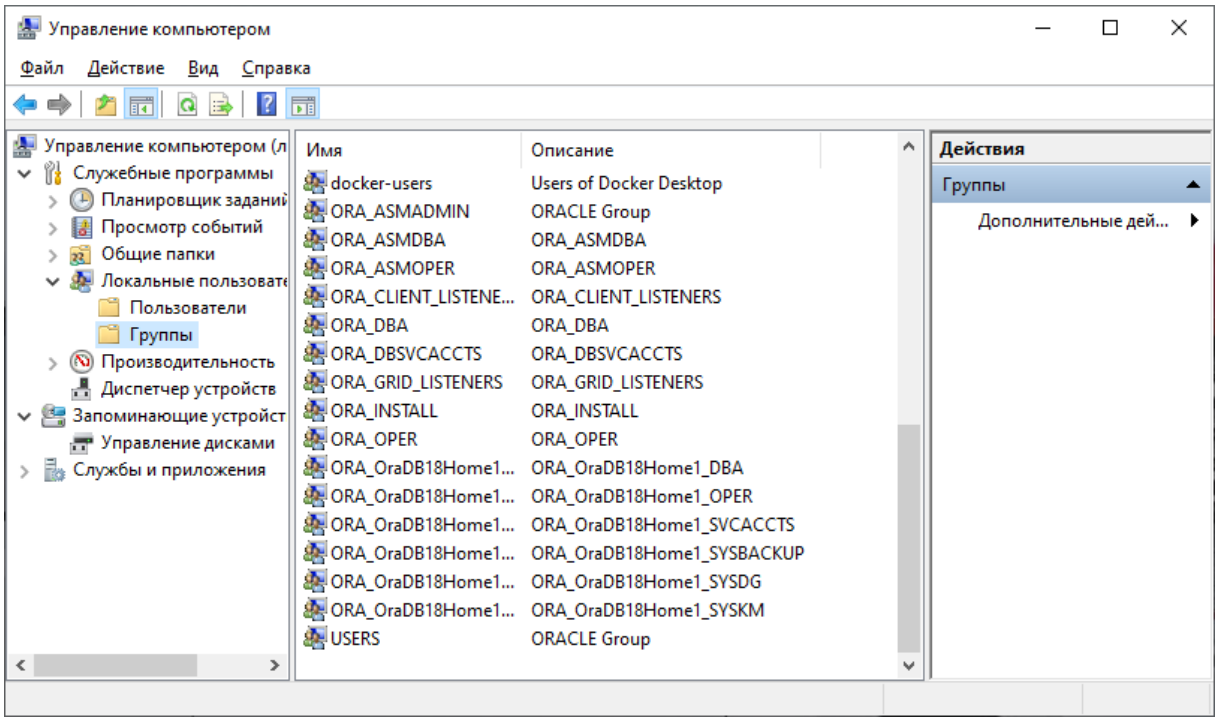


Рисунок 1.11 – Пользователи Oracle

- **SYSTEM** – генерируемый Oracle привилегированный пользователь, которому принадлежат ключевые ресурсы БД Oracle (представления словаря БД, репозиторий инструментов и пр.).

- **SYS** – генерируемый Oracle (предопределенный) привилегированный пользователь ранга администратора базы данных (АБД), который является владельцем ключевых ресурсов БД Oracle (модули Oracle, таблицы словаря БД, V$-представления словаря, ...)

- **DBA** – предопределенная роль, которая автоматически создаётся для каждой базы данных Oracle и содержит все системные привилегии, кроме SYSDBA и SYSOPER. Должна назначаться только администраторам, которым требуется полный доступ.

- **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.

- Роль DBA не включает SYSDBA и SYSOPER. Привилегии могут быть указаны при подключении (connect) пользователя к БД.

1.3.4 Экземпляры 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.

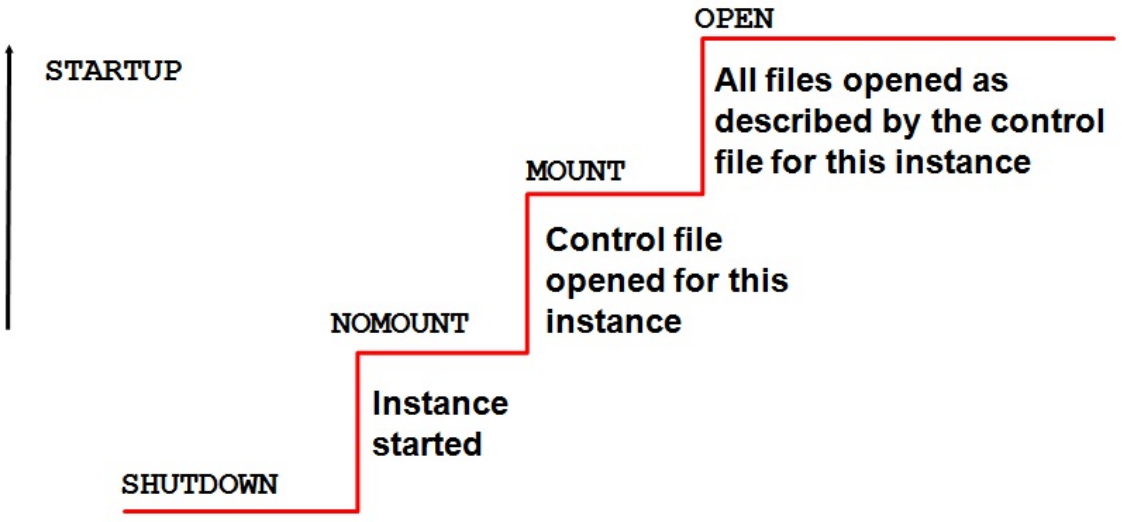


Рисунок 1.12 – Запуск экземпляра

1.4 Установка Oracle в контейнер docker

Sun Microsystems предоставляют возможность использования образа Docker. Одним из преимуществ использования Docker является быстрая и простая подготовка. Предварительно на компьютере должны быть установлены Docker и Git for Windows.

Открываем Git Bash и клонируем исходный код из git репозитория:

git clone https://github.com/oracle/docker-images.git

Переходим в каталог:

cd docker-images/OracleDatabase/SingleInstance/dockerfiles/

Собираем докер образ при помощи скрипта:

buildContainerImage.sh -v 18.4.0 -x

Создаем контейнер из созданного образа:

docker run --name oracle18xe -p 1521:1521 -p 5500:5500 -e oracle_pwd= WM7NFQy5 -v C:\Oracle\Docker\docker-demo-db\oracle18xe:/opt/oracle/oradata oracle/database:18.4.0-xe

1.5 Настройка сетевого соединения к серверам Oracle

1.5.1 Понятие Oracle Net Services

Oracle Net Services – набор служб, которые устанавливают подключение между сервером БД и пользователями БД (рис.1.13)

- Службы Oracle Net
- Oracle Net Listener
- Oracle Net Configuration Assistant
- Oracle Net Manager
- Oracle Connection Manager

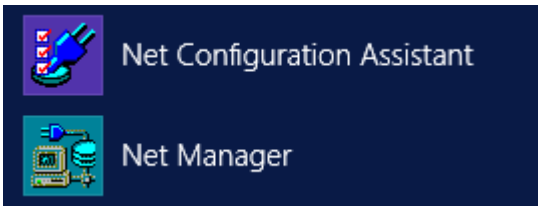


Рисунок 1.13 – Сетевые службы

Oracle Net – программный компонент, который инициализирует, устанавливает и поддерживает подключения между клиентом и сервером.

Должен быть установлен и на клиенте, и на сервере.

Состоит из двух компонентов:

- Oracle Network Foundation layer – отвечает за установку и поддержание подключений между клиентским приложением и сервером.
- Oracle Protocol Support – отвечает за отображение функциональности TNS (Transparent Network Substrate) на стандартные протоколы, используемые при подключении.

1.5.2 Дескрипторы, идентификаторы и строки соединений

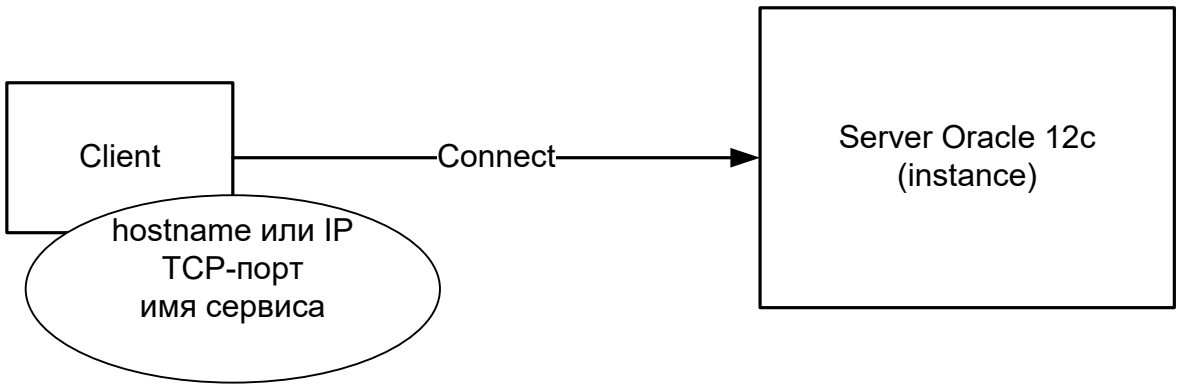


Рисунок 1.14 – Клиент-серверное соединение

Дескриптор соединения – объединенная спецификация двух обязательных компонентов подключения к базе данных:

- Имени службы базы данных
- Местоположения адреса базы данных

```
(DESCRIPTION
  (ADDRESS = (PROTOCOL = TCP)
             (HOST = имя_хоста)
             (PORT = 1521))
  (CONNECT_DATA =
             (SERVICE_NAME = имя_службы_базы данных)))

   (DESCRIPTION
     (ADDRESS = (PROTOCOL = TCP)
                (HOST = 192.168.1.225)(PORT = 1521))
     (CONNECT_DATA = (SERVICE_NAME = pdb_a.be.by)))
```

```
pdb_a = (DESCRIPTION
   (ADDRESS = (PROTOCOL = TCP)
                (HOST = 192.168.1.225)(PORT = 1521))
   (CONNECT_DATA = (SERVICE_NAME = pdb_a.be.by)))
```

Подключение к базе данных выполняется путем указания строки соединения:

```
CONNECT scott/tiger@(DESCRIPTION
   (ADDRESS = (PROTOCOL = TCP)
                (HOST = 192.168.1.225)(PORT = 1521))
   (CONNECT_DATA = (SERVICE_NAME = pdb_a.be.by)))


CONNECT scott/tiger@pdb_a
```

1.5.3 Настройка клиентских приложений Oracle

Oracle – клиент есть в поставке сервера. Дополнительно загружается с oracle.com. Не обязательно должен совпадать по версии с сервером, но желательно.

Виды подключений к Oracle:

- Простое подключение – Basic
- Локальное именование – TNS
- LDAP-соединение
- Local/bequeath-соединение
- Прочие

Basic – соединение. Явно указываются все параметры соединения

CONNECT имя/пароль@[//]хост[:порт][/имя_службы]

При BASIC соединении должны быть установлены Oracle Net Services. Необходима поддержка протокола TCP/IP – на сервере и клиенте. Нельзя использовать расширенные сетевые функциональные возможности Oracle.

TNS-соединение

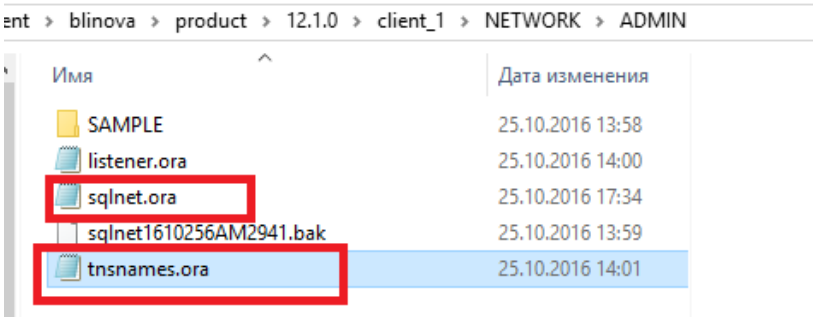


Рисунок 1.15 – Файлы параметров

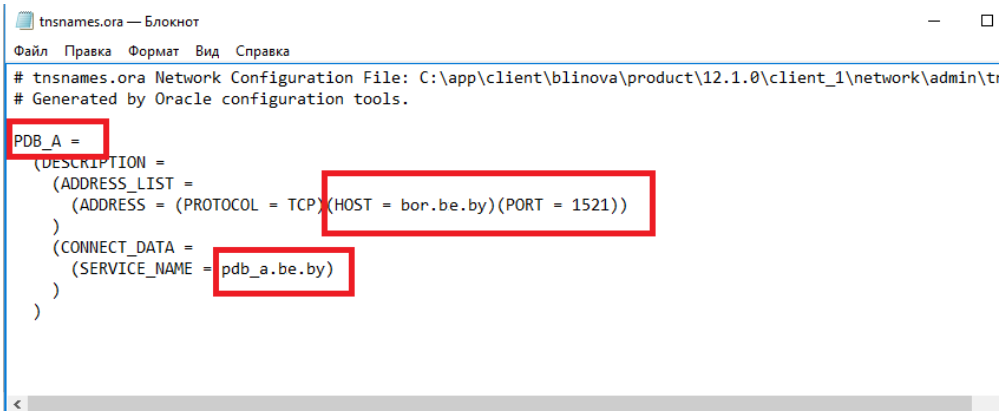


Рисунок 1.16 – Настройка соединения TNS

Утилита TNSPing

```
C:\Users\blinova>tnsping pdb_b

TNS Ping Utility for 64-bit Windows: Version 12.1.0.2.0 - Production on 25-OCT-2016 18:33:41

Copyright (c) 1997, 2014, Oracle.  All rights reserved.

Used parameter files:
C:\app\client\blinova\product\12.1.0\client_1\network\admin\sqlnet.ora


Used TNSNAMES adapter to resolve the alias
Attempting to contact (DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST = bor.b
e.by)(PORT = 1521))) (CONNECT_DATA = (SERVICE_NAME = pdb_b.be.by)))
OK (40 msec)
```

Рисунок 1.17 – Проверка соединения с помощью TNSPing

LDAP-соединение

Метод именования с помощью службы каталогов. OID – Oracle Internet Directory. LDAP – Lightweight Directory Access Protocol. Требуется наличие специального LDAP-сервера.

Local/bequeath-соединение

Только на сервере. Можно соединяться с помощью sqlplus или sqldeveloper без указания параметров соединения только с выделенным сервером (рис.1.18). Listener не задействован. Соединение со стандартным сервисом SYS$USERS

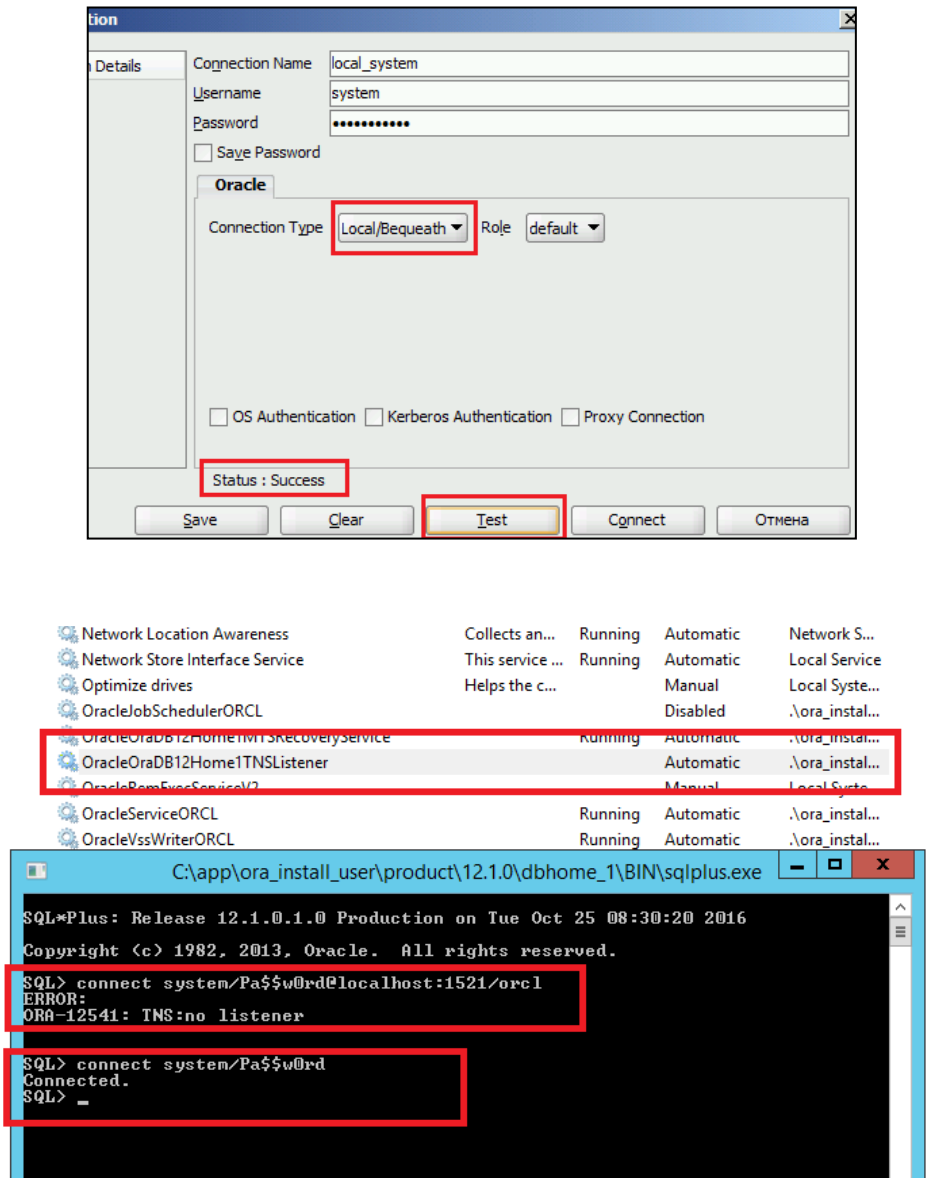


Рисунок 1.18 – Подключение Local/bequeath

1.5.4 Утилиты работы с БД Oracle

SQL*Plus позволяет выполнять команды SQL и блоки PL/SQL, а

также решать ряд других задач. С помощью SQL*Plus можно: вводить, редактировать, запоминать, загружать и выполнять команды SQL и блоки PL/SQL, форматировать, создавать, сохранять, печатать и публиковать в Web результаты выполнения запросов (отчеты); получать описание (имена и типы столбцов) любой таблицы и представления; обращаться к удаленным базам данных и копировать из них данные, посылать и принимать сообщения от конечных пользователей; администрировать базу данных. Работа осуществляется через командную строку. Поставляется вместе с Oracle.

PL/SQL Developer – это Integrated Development Environment, направленное на разработку программных единиц для Баз данных Oracle. Фокусируется на простоте использования, качестве кодирования и продуктивности, необходимых преимуществах при разработке приложений Oracle.

TOAD for Oracle – мощное средство, предназначенное для повышения скорости разработки баз данных и приложений, а также упрощения ежедневных задач администрирования. Toad for Oracle позволяет разработчикам избегать утомительных задач отладки PL/SQL кода, обеспечивает удобную среду разработки приложений, экономя время для разработки и тестирования больших проектов.

1.5.5 SQL Developer

SQL Developer – средство для работы с запросами SQL и программными единицами PL/SQL.

Преимущества:

- официально бесплатен;
- разработан в самой Oracle;
- написан на языке Java, поэтому один и тот же графический интерфейс можно использовать как из-под Windows, так и под Unix;
- можно использовать также для написания и отладки запросов к

другим СУБД;

- не требует установки на компьютер. Все пользовательские настройки в нем хранятся в файлах XML, поэтому его достаточно скопировать на диск и при первом запуске указать путь к Java (не ниже 6 версии).

Недостатки:

- ресурсоемкость;
- отсутствие официальной поддержки со стороны службы поддержки Oracle;
- некоторая неустойчивость в работе.

1.6 Технология Oracle Application Express (APEX)

Oracle Application Express (APEX) – это low-code платформа для разработки функциональных, масштабируемых и безопасных web-приложений. На сегодняшний день последней актуальной версией APEX является Oracle APEX 20.2. Данная версия выпущена в октябре 2020 года. Для установки и работы с APEX 20.2 в Oracle Database 18c Express Edition необходимо понимать принцип работы новой опции Oracle Database – Multitenant. Начиная с Oracle Database 12c поддерживается новая архитектура – Multitenant, которая предоставляет возможность использовать множество баз данных для консолидации их в составе единой и главной базы данных. Такое объединение упрощает задачи администрирования баз данных. Единая и главная база данных используется в качестве платформы и называется контейнерная база данных (Container Database – CDB), а база данных из множества работающих в составе контейнерной базы данных называется подключаемой базой данных (Pluggable Database – PDB). Архитектура Multitenat позволяет создать в Oracle Database 18c Express Edition одну CDB базу и до трех PDB баз. По

умолчанию в Oracle Database 18c Express Edition APEX отсутствует. В связи с этим, для установки и работы с APEX необходимо подключиться к PDB и выполнять установку и настройку APEX.

1.6.1 Установка и настройка локальной версии APEX

Предполагается, что есть успешно установленная Oracle Database 18c Express Edition.

Установка APEX на Oracle Database 18c Express Edition

Oracle APEX распространяется бесплатно и установочный файл можно скачать с официального сайта Oracle – https://www.oracle.com/tools/downloads/apex-downloads.html (рис.1.19)

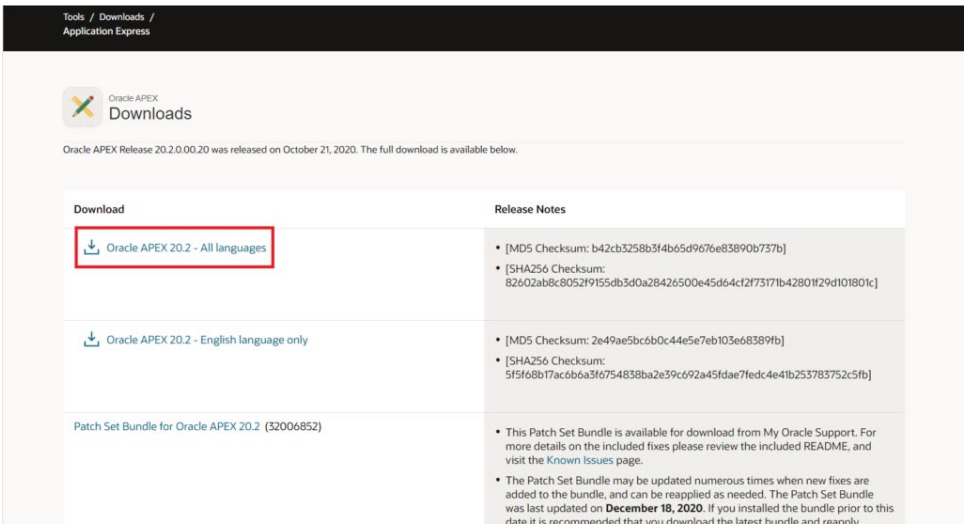


Рисунок 1.19 – Загрузка APEX

После ознакомления и принятия условий лицензирования необходимо поставить галочку в разделе I reviewed and accept the Oracle License Agreement (рис.1.20).

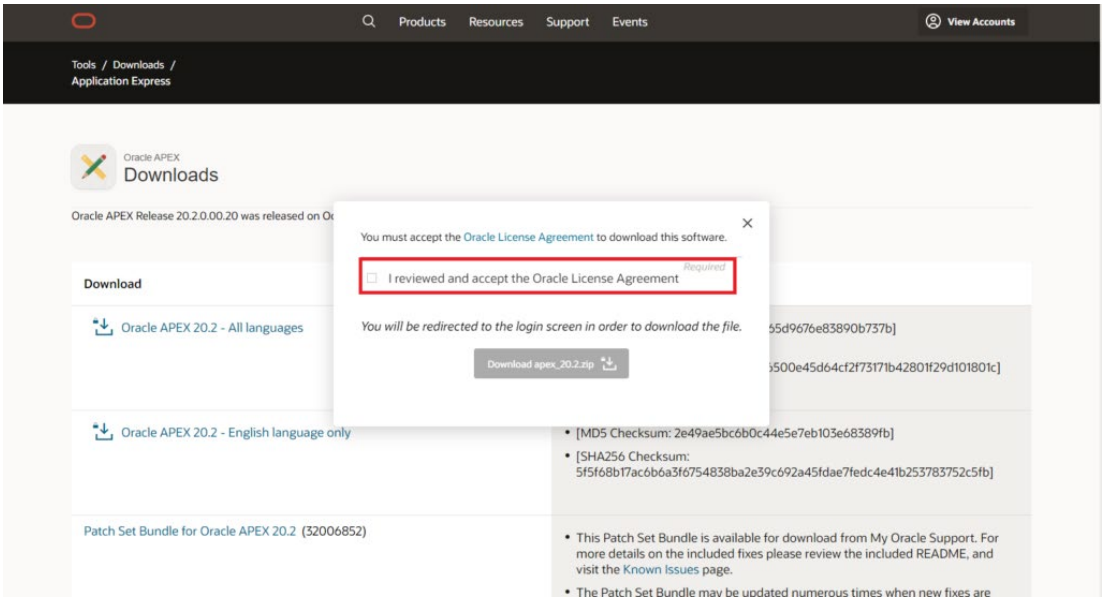


Рисунок 1.20 – Принятие условий лицензирования

После завершения скачивания архив apex_20.2.zip нужно извлечь в директорию C:\Oracle. После разархивирования создалась папка apex с множеством sql скриптов для создания и настройки APEX. Необходимо из директории со скриптами (C:\Oracle\apex) подключиться к PDB XEPDB1 и начать установку APEX (рис.1.21):

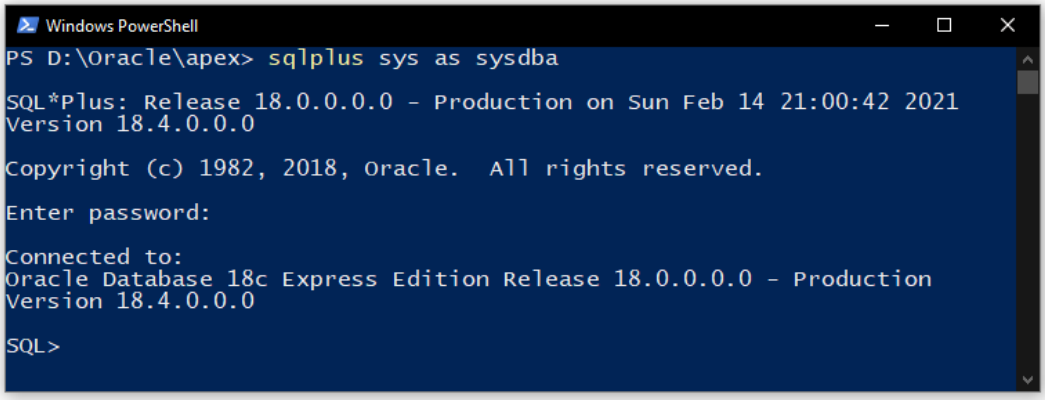


Рисунок 1.21 – Подключение к PDB XEPDB1

[oracle@ apex]$ sqlplus sys as sysdba
*SQL*Plus: Release 18.0.0.0.0 - Production on Sat Aug 8 11:21:28 2020*
Version 18.4.0.0.0
Copyright (c) 1982, 2018, Oracle. All rights reserved.
Enter password:
Connected to:
Oracle Database 18c Express Edition Release 18.0.0.0.0 - Production

```
Version 18.4.0.0.0

SQL> show pdbs;
 CON_ID  CON_NAME  OPEN MODE  RESTRICTED
 ----------------------------------------
   2    PDB$SEED  READ ONLY     NO
   3    XEPDB1    READ WRITE    NO
SQL> alter session set container = XEPDB1;
Session altered.
SQL> show pdbs;
 CON_ID  CON_NAME  OPEN MODE  RESTRICTED
 ----------------------------------------
   3    XEPDB1    READ WRITE    NO
 SQL>
```

Подключение к XEPDB1 прошло успешно. Запускается скрипт apexins.sql для установки APEX со следующим синтаксисом:

@apexins.sql tablespace_apex tablespace_files tablespace_temp images

где

- **tablespace_apex** – название табличного пространства для пользователя приложения APEX;
- **tablespace_files** – название табличного пространства для пользователя файлов APEX;
- **tablespace_temp** – название временного табличного пространства;
- **images –** виртуальная директория для изображений APEX.

Рассмотрим пример выполнения данного скрипта:

```
SQL> @apexins.sql SYSAUX SYSAUX TEMP /i/
...
<Эта часть лога не приведена>
...
#
Actions in Phase 3:
#
   ok 1 - BEGIN                                    |  0.00
   ok 2 - Computing Pub Syn Dependents             |  0.00
   ok 3 - Upgrade Hot Metadata and Switch Schemas  |  0.00
   ok 4 - Removing Jobs                            |  0.00
   ok 5 - Creating Public Synonyms                 |  0.03
   ok 6 - Granting Public Synonyms                 |  0.10
   ok 7 - Granting to FLOWS_FILES                  |  0.00
```

ok 8 - Creating FLOWS_FILES grants and synonyms | 0.02
ok 9 - Creating Jobs | 0.00
ok 10 - Creating Dev Jobs | 0.00
ok 11 - Installing FLOWS_FILES Objects | 0.00
ok 12 - Installing APEX$SESSION Context | 0.00
ok 13 - Recompiling APEX_200200 | 0.05
ok 14 - Installing APEX REST Config | 0.02
ok 15 - Set Loaded/Upgraded in Registry | 1.32
ok 16 - Removing Unused SYS Objects | 0.00
ok 17 - Validating Installation | 0.03
ok 3 - 17 actions passed, 0 actions failed | 1.57

PL/SQL procedure successfully completed.

Thank you for installing Oracle Application Express 20.2.0.00.20

Oracle Application Express is installed in the APEX_200200 schema.
The structure of the link to the Application Express administration services is as follows:

http://host:port/ords/apex_admin

The structure of the link to the Application Express development interface is as follows:
http://host:port/ords

timing for: Phase 3 (Switch)
Elapsed: 00:01:34.83
timing for: Complete Installation
Elapsed: 00:14:57.61

PL/SQL procedure successfully completed.
1 row selected.
...null1.sql
SYS>

Здесь приводится последняя часть лога установки APEX, так как система генерирует очень много строк лога. Лог показывает, что установка прошла успешно и заняла 14 мин 57 секунд времени.

Настройки и подключение к APEX

После успешного завершения установки необходимо создать пользователя с административными правами в APEX и назначить ему пароль, настроить ORDS и создать рабочее пространство для разработки приложений.

Для создания администратора и назначения ему пароля запускается

скрипт apxchpwd.sql в PDB. Данный скрипт в будущем также может быть использован для сброса пароля существующему администратору APEX.

После запуска скрипта система предложит ввести пользовательское имя для администратора (login), его email и пароль. По умолчанию система предлагает ADMIN для пользовательского имени администратора (login). Если необходимо оставить пользователя по умолчанию, то поле Enter the administrator’s username [ADMIN] следует оставить пустым и нажать Enter. В поле Enter ADMIN’s email [ADMIN] можно указать свою электронную почту. В поле Enter ADMIN’s password [] надо назначить пароль администратора, который будет удовлетворять следующим требованиям:

- Пароль должен содержать не менее 6 символов.
- Пароль должен содержать не менее одного буквенного символа.
- Пароль должен содержать не менее одного символа пунктуации: (!”#$%&()“*+,-/:;?_).
- Пароль должен содержать не менее одного заглавного буквенного символа.
- Пароль должен содержать не менее одного строчного буквенного символа.

Иначе система выдаст следующую ошибку:

```
ORA-20001: Password validation failed.
ORA-06512: at line 30
ORA-06512: at "APEX_200100.WWV_FLOW_FND_USER_INT", line 3744
ORA-06512: at line 20
```

Запускается скрипт для создания администратора с именем ADMIN и назначения его email и пароля:

```
SYS> @apxchpwd.sql
...set_appun.sql
================================================
This script can be used to change the password of an Application Express
```

instance administrator. If the user does not yet exist, a user record will be created.
==========
Enter the administrator's username [ADMIN]
User "ADMIN" does not yet exist and will be created.
*Enter ADMIN's email [ADMIN] test@***.tj*
Enter ADMIN's password []
Created instance administrator ADMIN.
SYS>

Пользователь успешно создан.

Также важно настроить учетную запись APEX_PUBLIC_USER для управления Oracle Application Express. Учетная запись APEX_PUBLIC_USER создается со случайным и недоступным паролем во время установки Oracle Application Express. Если же выполняется обновление (upgrade) с предыдущей версии Oracle Application Express, то приведенные ниже две команды не являются необходимыми:

SYS> ALTER USER APEX_PUBLIC_USER ACCOUNT UNLOCK;
SYS> ALTER USER APEX_PUBLIC_USER IDENTIFIED BY new_password;

Для новой установки APEX (а не обновление) необходимо запустить скрипт apex_rest_config.sql для настройки RESTful сервисов. После запуска потребуется назначить пароль для учетных записей APEX_LISTENER, APEX_REST_PUBLIC_USER. Ниже во время установки и настройки ORDS будет необходимо вводить пароли этих учетных записей.

SYS> @apex_rest_config.sql
*Enter a password for the APEX_LISTENER user [] *
Enter a password for the APEX_REST_PUBLIC_USER user []
...set_appun.sql
...setting session environment
...create APEX_LISTENER and APEX_REST_PUBLIC_USER users
SYS>

Настройка ORDS

В APEX версии 20.2 уже не поддерживается Web Listener с типом Embedded PL/SQL Gateway и Oracle HTTP Server. В связи с этим, необходимо установить и использовать Oracle REST Data Services (ORDS). Далее описывается процесс установки и базовой настройки APEX с ORDS версии 20.3.

Oracle ORDS распространяется бесплатно и установочный файл можно скачать с официального сайта Oracle: https://www.oracle.com/database/technologies/appdev/rest-data-services-downloads.html

Для скачивания на портале Oracle необходимо наличие учетной записи с паролем. При ее отсутствии осуществляется регистрация новой учетной записи.

По ссылке скачивается файл Oracle REST Data Services (рис.1.22).

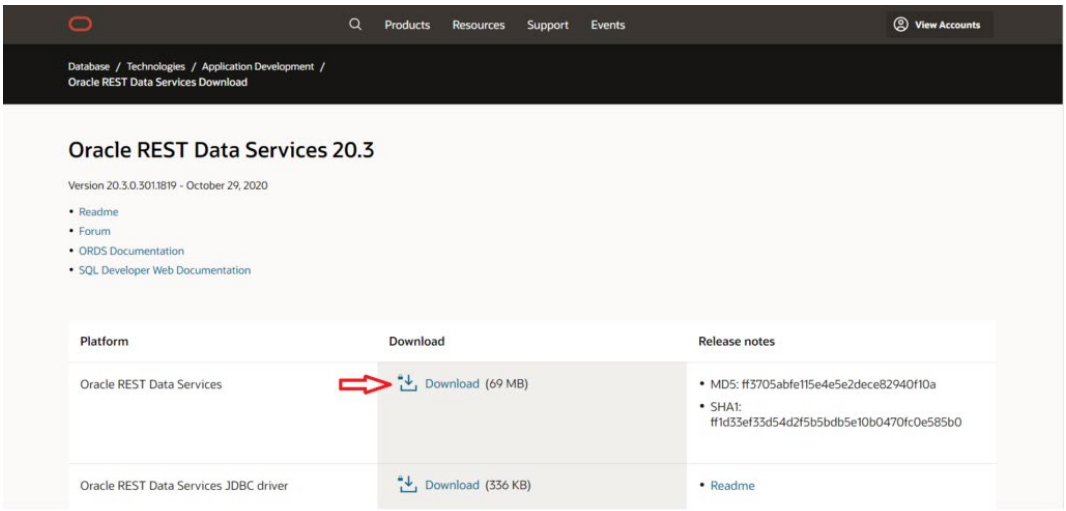


Рисунок 1.22 – Загрузка Oracle REST Data Services

После ознакомления и принятия условий лицензирования необходимо поставить галочку в разделе I reviewed and accept the Oracle License Agreement.

Нажать на «Download ords-*.zip».

После завершения скачивания необходимо создать директорию ords в C:\Oracle\ и архив ords-20.3.0.301.1819.zip нужно извлечь в созданную директорию C:\Oracle\ords.

При использовании ORDS для работы с APEX приложениями ORDS должен быть настроен для обслуживания статических файлов APEX. Необходимо скопировать директорию images (apex/images) из домашней директории с установочными файлами APEX в папку файловой системы, где установлены службы Oracle REST Data Services. В данном примере домашняя директория APEX находится в C:\Oracle\apex и директория установки ORDS в C:\Oracle\ords.

Проверяется текущая версия JDK.

```
[oracle@ ~]$ java -version
java version "15.0.1" 2020-10-20
Java(TM) SE Runtime Environment (build 15.0.1+9-18)
Java HotSpot(TM) 64-Bit Server VM (build 15.0.1+9-18, mixed mode, sharing)
[oracle@ ~]$
```

Результат команды показывает, что на этой операционной системе установлена и настроена 15-я версия JDK.

Есть вероятность, что при использовании старой версии JDK (например, 8 версия) запуск установки ORDS выдаст следующую ошибку.

Есть два варианта установки ORDS:

- Автономный (standalone) режим.
- На сервере приложений (Oracle WebLogic Server, Apache Tomcat).

Далее описывается установка ORDS в standalone режиме.

Для начала установки и настройки ORDS необходимо запустить следующую команду: **java -jar ords.war install advanced**

После запуска команды система потребует предоставить данные для множества параметров. Ниже приведены некоторые параметры и их значения во время установки. Остальные параметры оставлены по

умолчанию или не требуют дополнительного описания.

– **Enter the location to store configuration data:** указывается директория настроек ORDS. Можно указать любую директорию. В данном примере указана следующая директория: C:\Oracle\ords\config

– **Enter the administrator username:** необходимо указать логин и пароль учетной записи базы данных с правами sysdba.

На этом завершается установка и настройка APEX с ORDS. Можно подключиться к Oracle APEX 20.2 по следующему адресу: http://localhost:8080/ords

Вместо localhost можно указать адрес 127.0.0.1 или IP адрес хоста. В примере используется IP адрес хоста равный 192.168.0.1. Запускается веб-браузер и выполняется переход к следующему адресу: http://192.168.0.1:8080/ords

Для информации: по умолчанию, для того чтобы получить доступ к корню APEX через сервисы ORDS необходимо перейти по пути /ords. Если есть необходимость использовать путь /apex для доступа к APEX, то потребуется переименовать файл ords.war в apex.war перед началом установки и настройки ORDS.

После завершения установки автоматически создается рабочее пространство (workspace) для выполнения административных задач в APEX. Название workspace – INTERNAL. Выполняется подключение к административному workspace под пользователем ADMIN и его паролем, которые были созданы ранее при запуске скрипта apxchpwd.sql (рис.1.23).

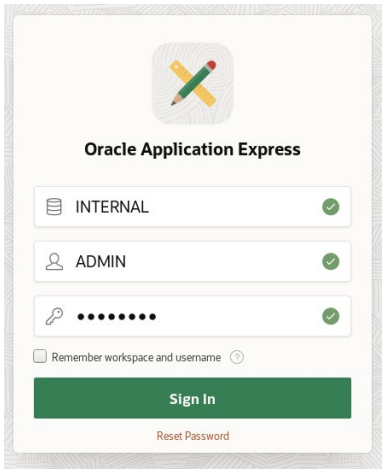


Рисунок 1.23 – Подключение к APEX

После успешного входа система запросит создать новое рабочее пространство для разработки приложений.

Далее задается название, идентификатор и описание рабочего пространства. В примере на рис.1.24 задается имя рабочего пространства и краткое описание. По умолчанию система сама генерирует идентификатор рабочего пространства.

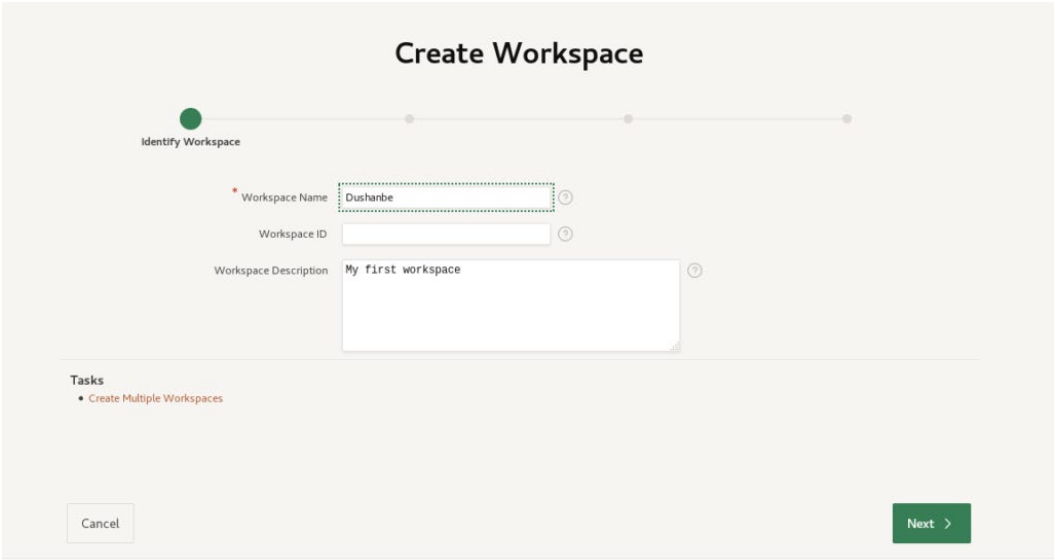


Рисунок 1.24 – Создание рабочего пространства APEX

На этапе Identify Schema (рис.1.25) надо будет указать название схемы (пользователя базы данных), к которой будет привязано новое рабочее пространство. Если в поле Re-use existing schema выбран No, то в поле

Schema Name необходимо указать название нового пользователя/схемы, в поле Schema Password задать ему пароль, а в поле Space Quota (MB) назначить ему квоту на использование пространства в табличном пространстве.

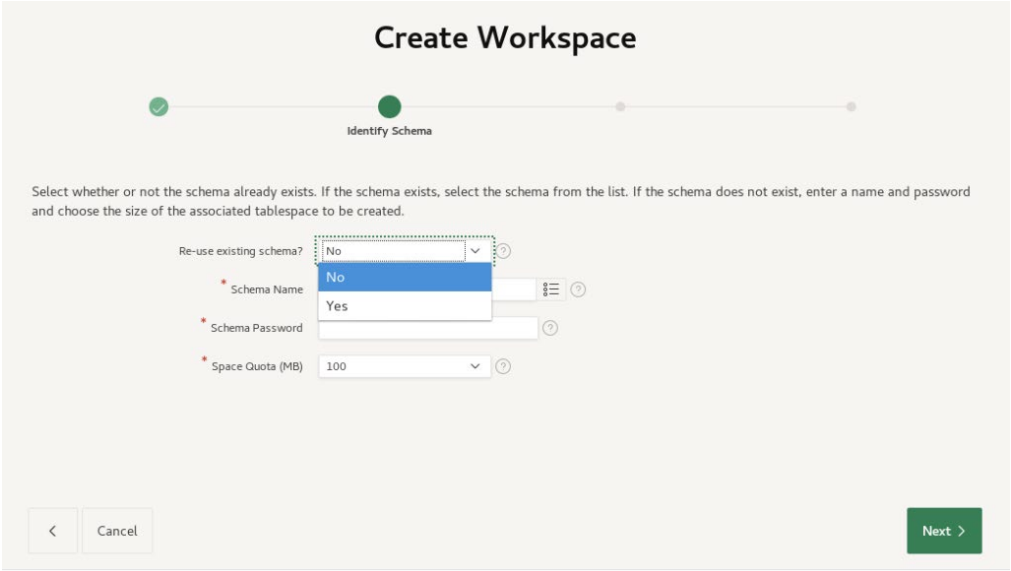


Рисунок 1.25 – Этап Identify Schema

В примере для поля Re-use existing schema выбирается значение Yes и в Schema Name указывается тестовая и учебная схема/пользователь hr (рис 1.26).

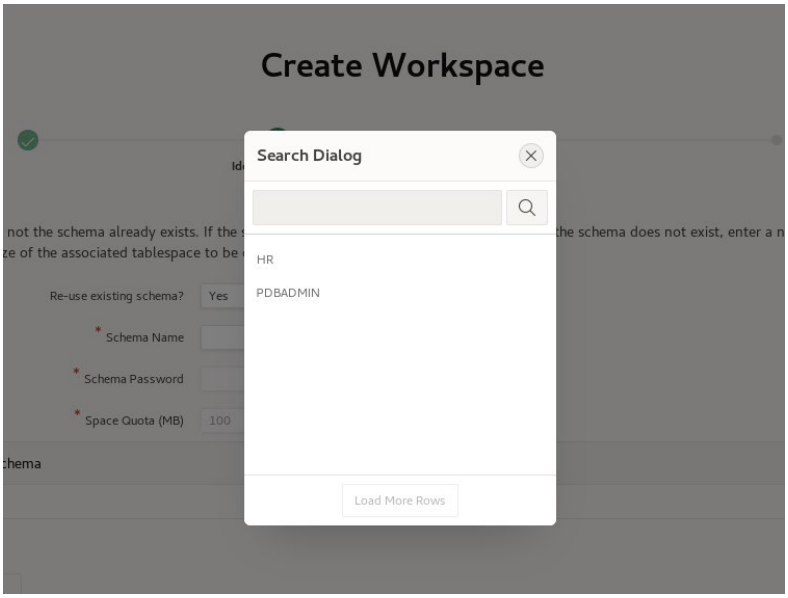


Рисунок 1.26 – Выбор учебной схемы

Это даст возможность использовать связанные таблицы с готовыми тестовыми данными, представления, триггеры, данные и другие объекты схемы hr во время создания приложений в APEX.

Нажимается Next и выполняется переход на шаг создания пользователя с административными правами для нового создаваемого пространства. Для нового пространства тоже создается пользователь с именем Admin. Указываются пароль, имя, фамилия и email администратора нового создаваемого пространства для разработки приложений (рис.1.27)

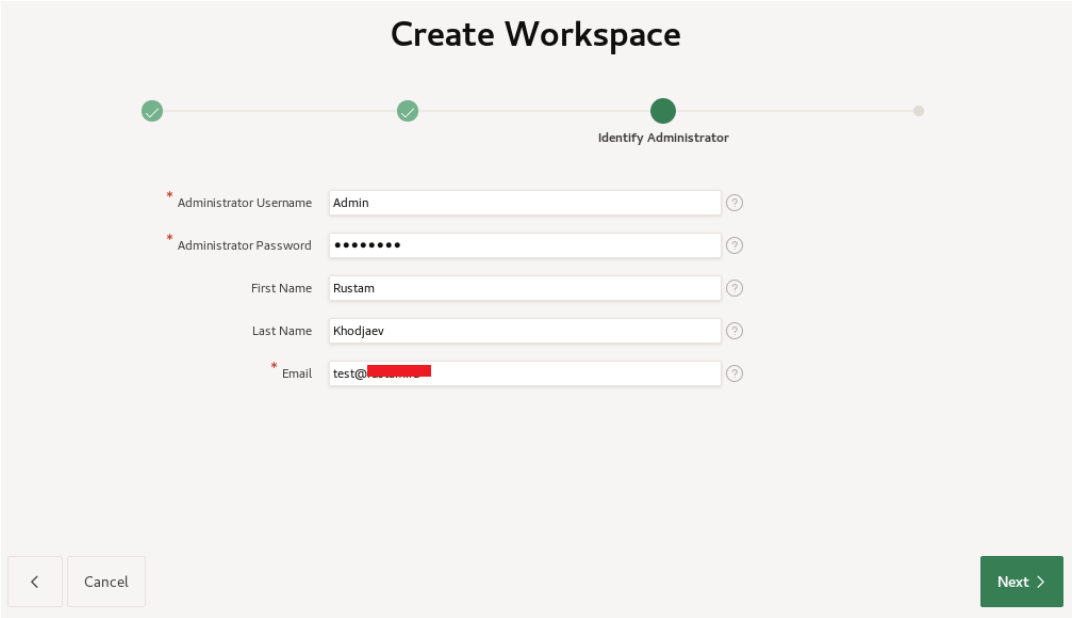


Рисунок 1.27 – Создание пользователя с административными правами

Нажимается Next и выполняется переход на следующий шаг – подтверждения настроек нового рабочего пространства:

После нажатия Create Workspace начнется создание рабочего пространства и при успешном завершении система выдаст сообщение об успешном создании рабочего пространства (рис.1.28)

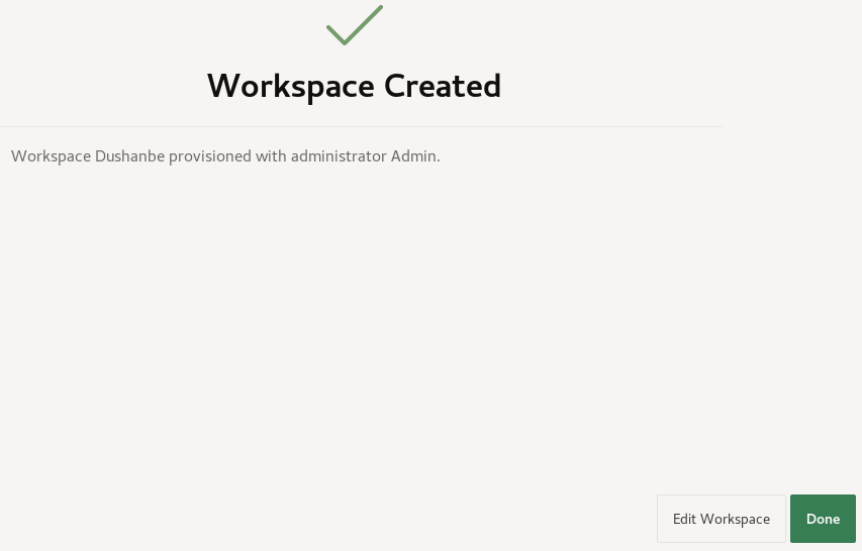


Рисунок 1.28 – Успешное завершение создания рабочего пространства

Это означает, что в APEX успешно создано новое рабочее пространство. Нажатие кнопки Done позволит пользователю ADMIN получить доступ к своему рабочему пространству (INTERNAL) и начать работу (рис.1.29).

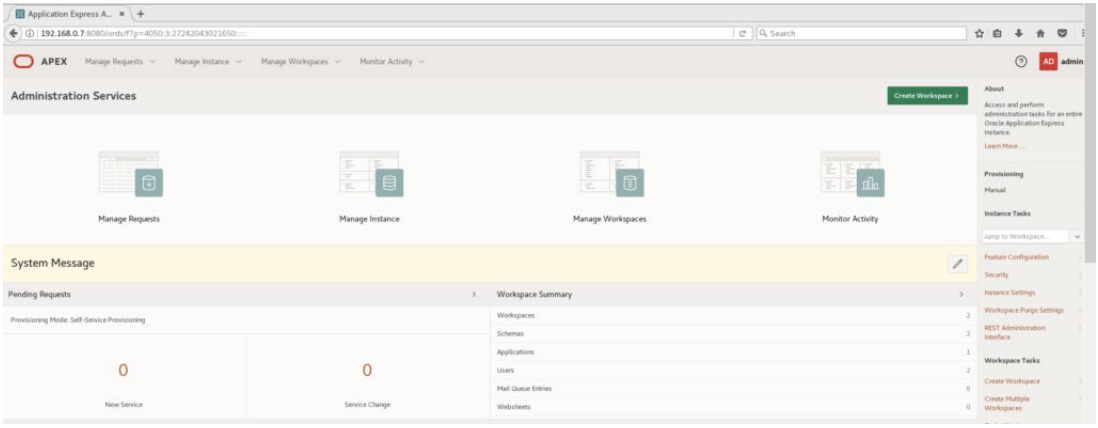


Рисунок 1.29 – Рабочее пространство APEX

Выполняется выход пользователя ADMIN из административного рабочего пространства (workspace) INTERNAL. Для этого в правом верхнем углу нажимается на имя пользователя и нажимается на Sign Out.

Нажимается на Return to Sign in Page и на новой странице указываются данные для подключения к новому рабочему пространству (workspace) (рис.1.30)

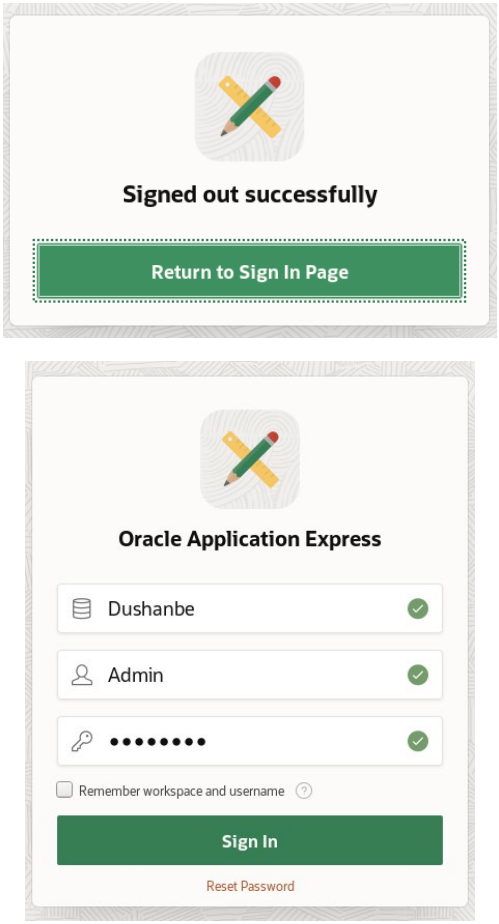


Рисунок 1.30 – Подключение к новому рабочему пространству

При первом входе администратору нового рабочего пространства (workspace) необходимо самостоятельно сменить пароль.

После успешной смены пароля администратор workspace получает доступ к среде. Рабочее пространство готово для начала разработки веб-приложений.

Необходимо учитывать, что после перезагрузки операционной системы сервис будет недоступен. Для запуска ORDS необходимо вручную запустить следующую команду java -jar ords.war standalone из директории

ords.

1.6.2 Подключение и настройка WEB-версии APEX на сайте Oracle

Кроме установки и использования платформы Oracle APEX на локальном компьютере, также есть возможность установить и использовать ее для разработки веб-приложений на бесплатном сервисе Always Free Service облака Oracle или же запросить и настроить свое рабочее пространство в бесплатном облачном сервисе apex.oracle.com (рис.1.31).

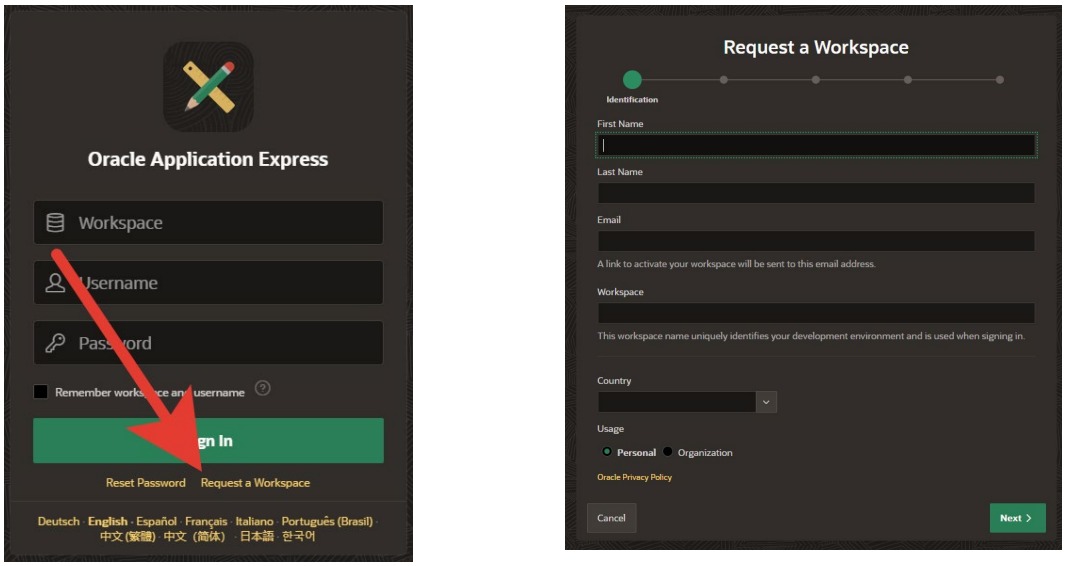


Рисунок 1.31 – Получение рабочего пространства в облачном сервисе

Через несколько минут на указанную Вами почту придёт письмо с ссылкой на продолжение создания окружения:

ЛАБОРАТОРНАЯ РАБОТА №1
СОЗДАНИЕ РАБОЧЕГО ПРОСТРАНСТВА ORACLE

Цель работы

1. Получение навыков установки и запуска СУБД Oracle Database Express Edition.

Знания, необходимые для выполнения лабораторной работы

1. Способы установки программ в ОС Windows.

Задание

1. Выполнить установку СУБД Oracle Database Express Edition на свой компьютер.
2. Установить платформу APEX на свой компьютер.

Контрольные вопросы

1. Что такое база данных?
2. Что такое экземпляр (сервер) базы данных?
3. Назовите ключевые отличия в редакциях Oracle Database.
4. Какие метрики лицензирования доступны в Oracle Database?
5. Что такое Oracle APEX?

ТЕМА 2. ОСНОВНЫЕ ПОНЯТИЯ АРХИТЕКТУРЫ. ОСНОВНЫЕ ОБЪЕКТЫ БД ORACLE

2.1 Понятие базы данных и экземпляров Oracle

В Oracle термином база данных описывается физическое хранилище информации, а термином экземпляр - программное обеспечение, работающее на сервере и предоставляющее доступ к информации в базе данных. Экземпляр исполняется на конкретном компьютере или сервере; база данных хранится на дисках, подключенных к этому серверу.

База данных – физическая сущность: она состоит из файлов, хранящихся на дисках.

Экземпляр – сущность логическая: он состоит из структур в оперативной памяти и процессов, работающих на сервере. Например, Oracle использует область разделяемой памяти System Global Area (SGA, системная глобальная область) и области памяти в каждом процессе - Program Global Area (PGA, программная глобальная область). Экземпляр может быть частью одной и только одной базы данных. Напротив, с одной базой данных может быть ассоциировано несколько экземпляров. Время жизни экземпляров ограничено, тогда как база данных при должном обслуживании может существовать вечно.

Пользователи не имеют прямого доступа к информации, хранящейся в базе данных Oracle; они должны запрашивать информацию у экземпляра Oracle.

2.2 Табличные пространства

Любые данные, хранящиеся в базе Oracle, должны находиться в каком-то табличном пространстве. Табличное пространство (tablespace) – это логическая структура. Каждое табличное пространство состоит из физических структур, называемых файлами данных (datafiles). В одном табличном пространстве может быть один или несколько файлов данных,

тогда как каждый файл данных принадлежит ровно одному табличному пространству. При создании таблицы можно указать, в какое табличное пространство ее поместить.

2.3 Словари данных

Словарь Oracle – набор таблиц и связанных с ними представлений, который предоставляет 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.

2.4 Мультиарендная архитектура

Oracle Multitenant - технология, позволяющая запустить несколько независимых баз данных в рамках одного экземпляра (рис.2.1). 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.

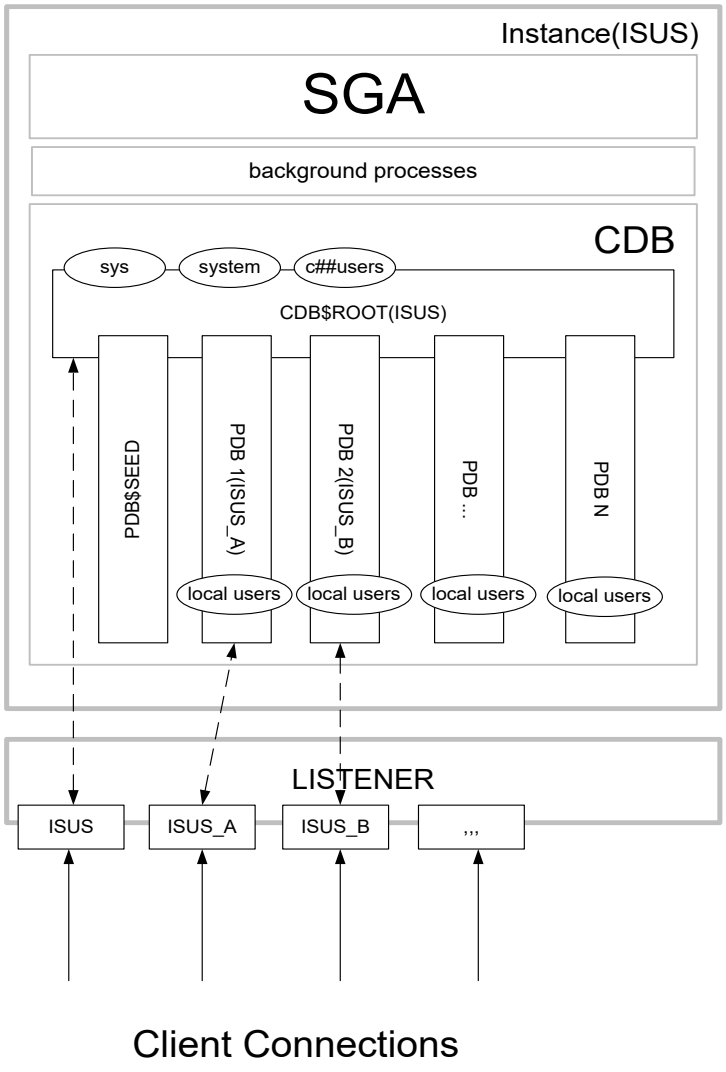


Рисунок 2.1 – Экземпляр базы данных

CDB – container DB – контейнер базы данных, а 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.

2.5 Разработка инфраструктуры базы данных

Разработка инфраструктуры базы данных состоит из следующих этапов:

- создание табличных пространств;
- создание ролей;
- назначение ролям системных привилегий;
- создание профилей безопасности;
- создание пользователей;
- назначение пользователям ролей;
- создание объектов базы данных;
- назначение ролям объектных привилегий.

2.6 Основные объекты базы данных Oracle

- Пользователи и схемы
- Привилегии и роли
- Таблицы, столбцы, ограничения и типы данных (в том числе абстрактные типы данных)
- Последовательности
- Кластеры и хэш-кластеры
- Индексы
- Синонимы
- Представления
- Моментальные снимки и материализованные представления
- Связи баз данных
- Секции
- Триггеры
- Процедуры, функции, пакеты

2.6.1 Пользователи и схемы

- Учетная запись пользователя не является физической структурой
- Пользователям принадлежат объекты
- Схема – набор объектов, принадлежащий учетной записи

пользователя

- Объекты создаются с правами учетных записей пользователей
- Учетные записи пользователей можно связать с учетными записями в ОС
- Двухкомпонентные имена – имя схемы.имя объекта

2.6.2 Привилегии и роли

- GRANT / REVOKE
- CREATE / ALTER / DROP - DATABASE / USER / PROFILE / TABLESPACE / ROLE / TABLE / INDEX / TRIGGER / PROCEDURE / SEQUENCE / VIEW
- WITH ADMIN OPTION
- ANY
- INSERT / UPDATE / DELETE / SELECT / EXECUTE / index / references - имя объекта
- WITH GRANT OPTION
- COLUMN
- CASCADE / RESTRICT

2.6.3 Таблицы, столбцы, ограничения и типы данных (в том числе абстрактные типы данных)

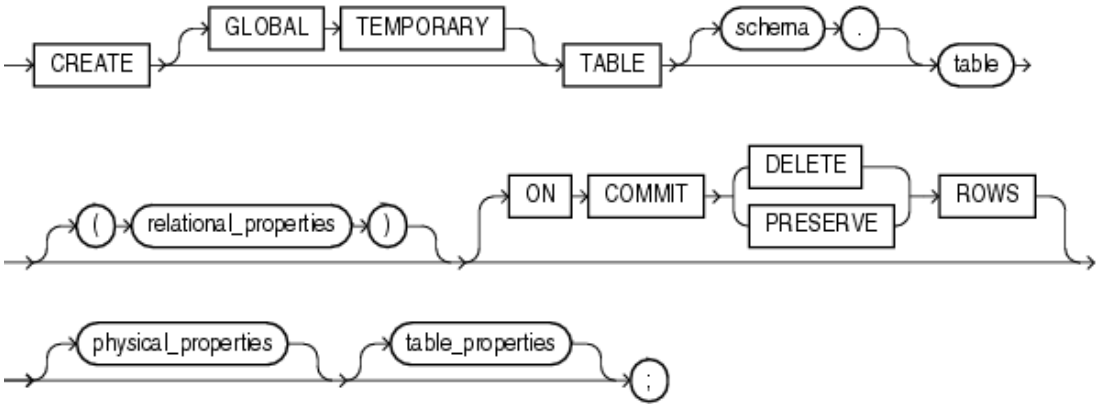
Таблица – основная структура сохранения информации в БД. Различают следующие типы таблиц:

- 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

Таблица может иметь до 1000 столбцов, содержать неограниченное число строк и иметь неограниченное число индексов. Количество таблиц не ограничено.

Запросы на создание таблиц строятся по схеме:



Параметр PCTFREE устанавливает процент памяти блока, резервируемой для возможных обновлений строк, уже содержащихся в блоке. Как только блок данных будет заполнен до процента PCTFREE, в этот блок невозможно будет вставить новые строки до тех пор, пока процент памяти, используемой в этом блоке, не упадет ниже значения PCTUSED. Параметр PCTUSED задает нижнюю границу, достижение которой вызывает возврат блока данных в список свободных областей.

Устанавливая разные варианты значений для этих параметров, можно оптимизировать использование дискового пространства.

Oracle оперирует со следующими типы данных:

- CHAR / NCHAR
- VARCHAR2 / NVARCHAR2
- DATE
- INTERVAL DAY TO SECOND / INTERVAL YEAR TO MONTH

- TIMESTAMP
- TIMESTAMP WITH TIME ZONE / TIMESTAMP WITH LOCAL TIME
- NUMBER (A, B)
- LONG RAW/ LONG / RAW
- BLOB / CLOB / NCLOB
- ROWID / UROWID

Если точность не указана, в столбце хранятся значения, как указано. Если масштаб не указан, масштаб равен нулю. Oracle гарантирует переносимость чисел с точностью до или менее 38 цифр. Вы можете указать масштаб и без точности: имя столбца NUMBER (*, масштаб). В этом случае точность равна 38, и указанный масштаб сохраняется.

Если вы укажете отрицательную шкалу, Oracle Database округляет фактические данные до указанного числа мест слева от десятичной точки. Например, указание (7, -2) означает округление базы данных Oracle до ближайших сотых.

Столбцы, определенные как LONG, могут хранить символьные данные переменной длины, содержащие до 2 гигабайт информации. LONG данные – это текстовые данные, которые должны быть соответствующим образом преобразованы при перемещении между различными системами. LONG-столбцы типов данных используются в словаре данных для хранения текста определений представлений. Вы можете использовать LONG-столбцы в SELECT-списках, SET-предложениях UPDATE- и VALUES-предложениях INSERT.

Типы больших объектов для символьных данных – это CLOB и NCLOB. Они могут хранить до 8 терабайт данных символов (CLOB) или данных национальных символов (NCLOB)

RAW-типы данных используются для данных, которые не должны быть истолкованы (не преобразованных при передаче данных между

различными системами) с помощью базы данных Oracle. Эти типы данных предназначены для двоичных данных или байтовых строк. Например, LONG RAW может использоваться для хранения графики, звука, документов или массивов двоичных данных. Интерпретация зависит от использования.

2.6.4 Последовательности

Последовательность – объект базы данных, предназначенный для генерации числовой последовательности (рис.2.2).

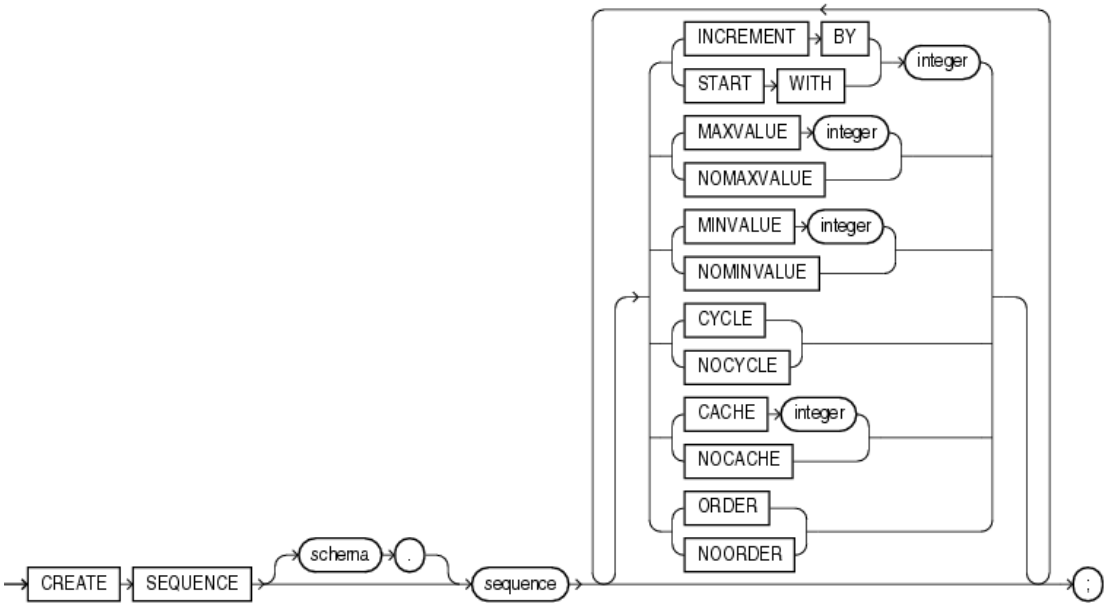


Рисунок 2.2 – Создание последовательности

После создания последовательности к ней можно обращаться через CURRVAL (возвращает текущее значение) и NEXTVAL (выполняет приращение последовательности и возвращает ее следующее значение). Текущее и следующее значения последовательности пользователи базы данных получают, выполняя команду SELECT. Последовательности – не таблицы, а простые объекты, генерирующие целые числа с помощью виртуальных столбцов, поэтому нужна общедоступная таблица словаря данных DUAL, из которой будут извлекаться данные виртуальных столбцов.

Первое обращение к NEXTVAL возвращает начальное значение последовательности. Последующие обращения к NEXTVAL изменяют значение последовательности на приращение, которое было определено, и возвращают новое значение. Любое обращение к CURRVAL всегда возвращает текущее значение последовательности, а именно, то значение, которое было возвращено последним обращением к NEXTVAL.

Прежде чем обращаться к CURRVAL в текущем сеансе работы, необходимо хотя бы один раз выполнить обращение к NEXTVAL.

2.6.5 Кластеры и хэш-кластеры

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 запросы, объединяющие эти таблицы, выполняются быстрее. Данные сохраняются вместе в кластере, что уменьшает количество операций ввода-вывода и повышает производительность. 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: *SELECT name FROM student WHERE id = 999;*

Схема запроса на создание кластера показан на рис.2.3.

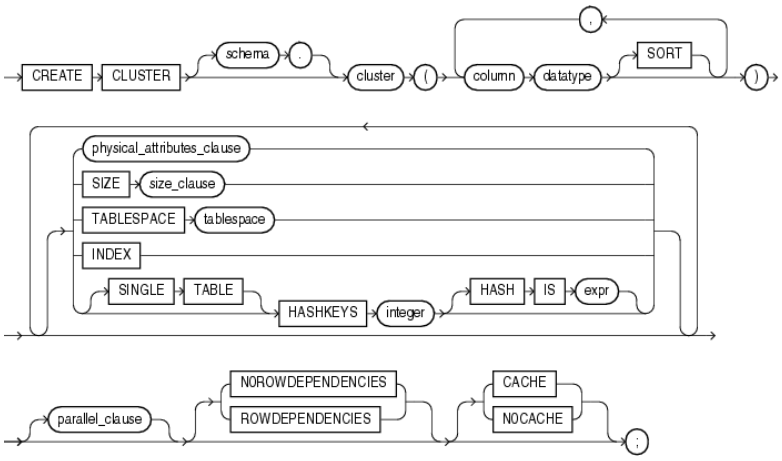


Рисунок 2.3 – Схема создания кластера

2.6.6 Индексы

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

Типы индексов:

- табличный индекс;
- битовый индекс;
- функциональный индекс;
- кластерный индекс.

Табличный индекс (B^+ Tree) структурирован в виде сбалансированного дерева. Листовой блок содержит индексированные значения столбца и соответствующий ему идентификатор строки (RowId) и предназначен для индексирования уникальных столбцов или столбцов с высокой селективностью

B^+-дерево имеет древовидную структуру, в которой все самые нижние узлы (листья) находятся на расстоянии одинакового количества уровней от вершины (корневого узла) дерева (рис.2.4). Это свойство поддерживается даже тогда, когда в индексированный столбец добавляются или удаляются данные.

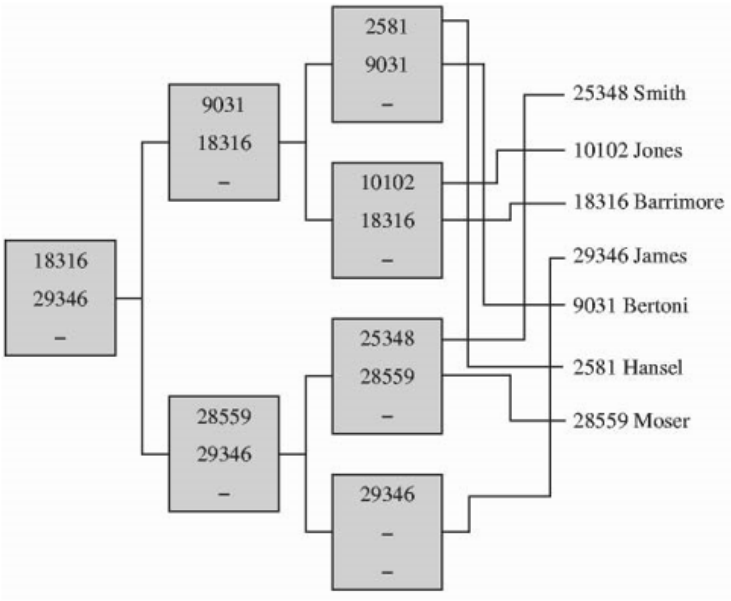


Рисунок 2.4 – B+-дерево

Битовый индекс создает битовые карты для каждого возможного значения столбца, где каждому биту соответствует строка, а значение бита 1 (0) означает, что соответствующая строка содержит (не содержит) индексируемое значение (рис.2.5).

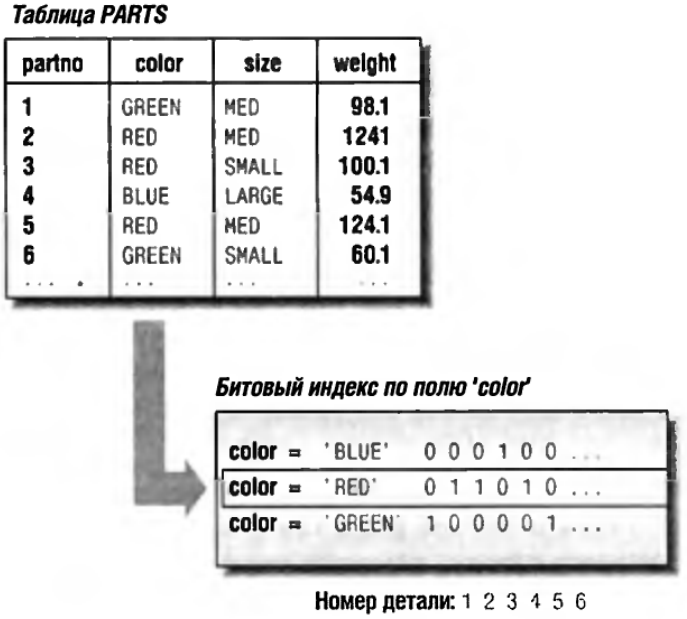


Рисунок 2.5 – Битовый индекс

Битовый индекс предназначен для индексирования столбцов с низкой

селективностью, не подходит для таблиц с частым обновлением, хорошо подходит для хранилищ данных.

Функциональный индекс – предварительно вычисляют значения функции по заданному столбцу и сохраняют результат в индексе.

2.6.7 Синонимы

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.

Синоним может указывать на:

- Таблицы,
- Процедуры,
- Функции,
- Последовательности,
- Представления
- Пакеты
- Объекты в локальной или удаленной базе данных

2.6.8 Представления

Представление – это хранимый запрос. К нему можно обращаться, как к обычной таблице. Синонимы добавляют уровень защиты данных, скрывают сложность данных, скрывают имена столбцов таблиц

2.6.9 Моментальные снимки и материализованные представления

- 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

```
CREATE TABLE SVVCORE.TEACHER
(
 TEACHER       CHAR(10),
 TEACHER_NAME  VARCHAR2(50),
 PULPIT        CHAR(10),
 CONSTRAINT PK_TEACHER PRIMARY KEY(TEACHER)
)

CREATE TABLE SVVCORE.PULPIT
(
 PULPIT       CHAR(10),
 PULPIT_NAME  VARCHAR2(100),
 FACULTY      CHAR(10),
 CONSTRAINT PK_PULPIT PRIMARY KEY(PULPIT)
 )
 CACHE
 COMPRESS

 CREATE TABLE SVVCORE.FACULTY
 (
   FACULTY      CHAR(10),
   FACULTY_NAME VARCHAR2(50),
   CONSTRAINT PK_FACULTY PRIMARY KEY(FACULTY)
 )
 ORGANIZATION INDEX
 NOMONITORING
 LOGGING


DROP MATERIALIZED VIEW COUNT_PFS

CREATE MATERIALIZED VIEW COUNT_PFS
BUILD IMMEDIATE
REFRESH COMPLETE START WITH TO_DATE('07102100','DDMMYYYY') NEXT TO_DATE('14102100','DDMMYYYY')
AS
SELECT P.PC, F.FC, T.TC
   FROM (SELECT COUNT(*) PC  FROM PULPIT)  P,
        (SELECT COUNT(*) FC  FROM FACULTY) F,
        (SELECT COUNT(*) TC  FROM TEACHER) T
```

2.6.10 Временные таблицы

Временные таблицы – механизм хранения данных в БД. Она состоит из столбцов и строк, как и обычная таблица.

ЛАБОРАТОРНАЯ РАБОТА №2
СОЗДАНИЕ ОБЪЕКТОВ БАЗЫ ДАННЫХ В ORACLE

Цель работы

1. Научиться производить первоначальную настройку пакета Oracle Database Express Edition.
2. Научиться создавать базы данных и совокупность связанных таблиц, принадлежащих указанной базе данных.
3. Научиться производить заполнение БД данными.

Знания, необходимые для выполнения лабораторной работы

1. Представление об экземплярах базы данных.
2. Синтаксис языка SQL базы данных Oracle.
3. Знать свойства основных объектов базы данных.

Задание

1. Создать базу данных из набора таблиц и связей между ними согласно своему варианту.
2. Внести несколько записей в каждую таблицу.

Контрольные вопросы

1. Поясните понятие «экземпляр базы данных».
2. Опишите основные виды файлов БД Oracle.
3. Что такое SGA?
4. Как связаны табличные пространства БД Oracle и файлы данных?
5. В чем заключается различие выделенных и разделяемых серверов БД Oracle?

ТЕМА 3. ПРОЦЕДУРНОЕ ЯЗЫКОВОЕ РАСШИРЕНИЕ PL/SQL

PL/SQL (Procedural Language / Structured Query Language) — язык программирования, процедурное расширение языка SQL, разработанное корпорацией Oracle.

3.1 Модуль DBMS_OUTPUT

Модуль DBMS_OUTPUT обеспечивает вывод информации для отладки. 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.

3.2 Блоки PL/SQL

Блока PL/SQL состоит из заголовка, раздела объявлений, исполняемого раздела и раздела исключений (рис.3.1).

Анонимные блоки (блоки без секции заголовка) используются как скрипт для выполнения PL/SQL выражений, но не могут быть вызваны из другого блока. Варианты использования анонимных блоков:

- 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, выполняющейся на сервере).

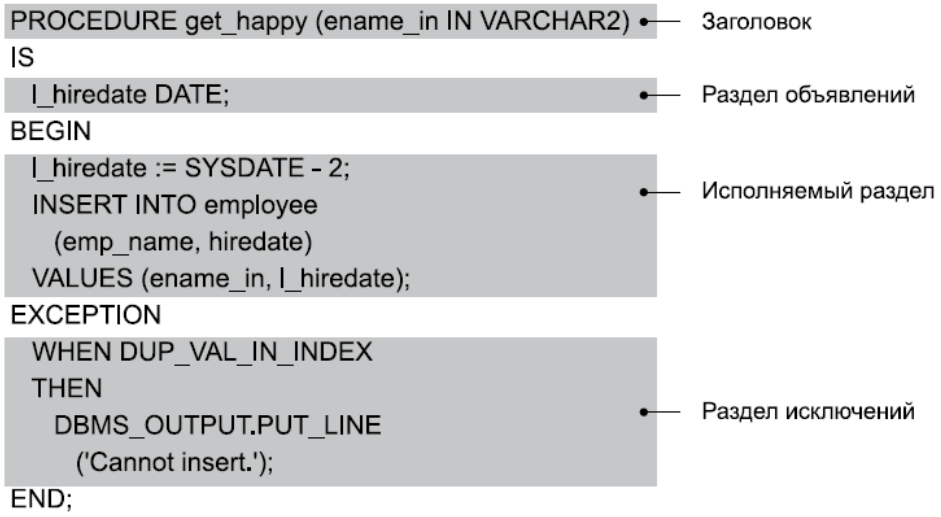


Рисунок 3.1 – Блок PL/SQL

Запрос на создание именованных процедур и функций осуществляется по схеме, приведенной на рис.3.2.

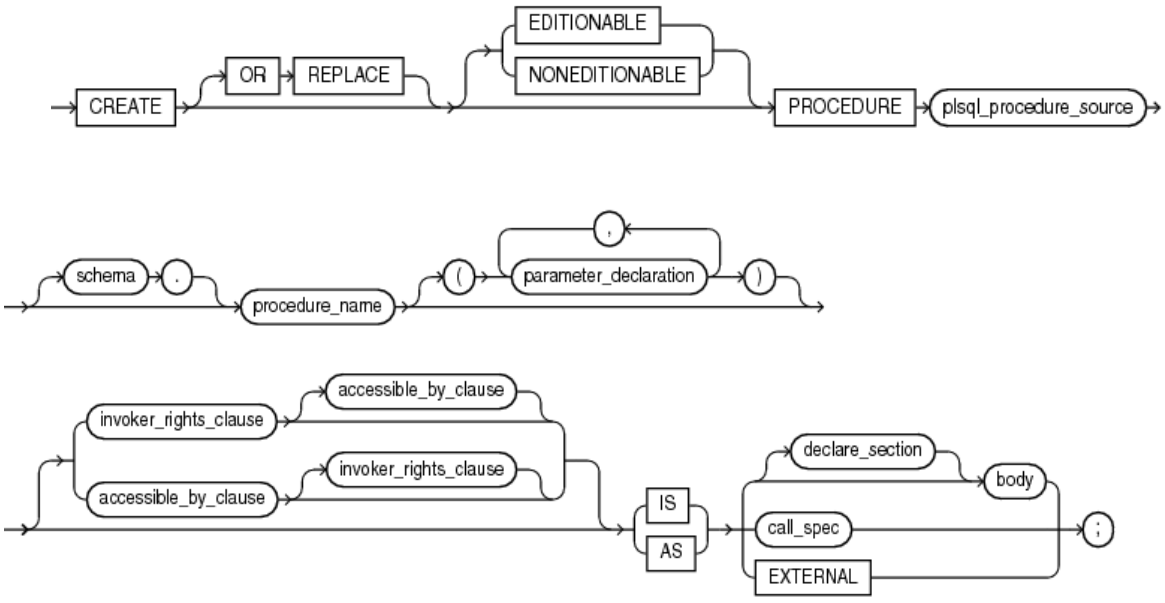


Рисунок 3.2 – Схема создания процедур и функций

В секции исключительных ситуаций доступны специальные функции SQLERRM и SQLCODE. SQLERRM возвращает сообщение об ошибке, связанной с исключительной ситуацией, а SQLCODE – номер ошибки,

связанной с исключительной ситуацией. Секция исключительных ситуаций может содержать несколько блоков when. Также в этой секции можно определить свои исключения.

При написании кода процедуры/функции можно использовать вложенные блоки для ограничения области видимости переменных.

При компиляции выдаются предупреждения. Уровень предупреждений настраивается при помощи переменной plsql_warnings:

- ALL (все);
- PERFOMANCE (производительность);
- INFORMATIONAL (информационные);
- SEVERE (логика программы);
- Specific error (ошибка).

3.3 Идентификаторы, литералы и метки

Идентификатор – это наименование объекта PL/SQL. Идентификаторами могут быть: константы, скалярные переменные, составные переменные (структуры и коллекции), исключения, пакеты, процедуры и функции, типы, курсоры, зарезервированные слова, метки.

Ограничения и свойства идентификатора:

- Не более 30 символов
- Начинается с буквы
- Не содержит пробелов
- Может включать символы «$», «_», «#».
- Компилятор приводит идентификаторы к верхнему регистру
- “идентификатор” регистрозависим

Литерал – это значение идентификатора:

- Number – 123, 21.6, NULL
- String – ‘sentence’, ’01-01-2017’ , NULL

- Boolean – true, false, NULL
- ANSI date – DATE '2016-11-01'

Метка – способ именовать определенную часть программы. Синтаксис метки: << label>>. Используется для:

- Именования блока на время выполнения
- Улучшение читаемости кода
- Необходимость ссылаться во вложенном блоке на переменную с таким же именем из внешнего блока
- Для перехода по GOTO

Таблица 3.1 – Символы специального назначения

;	Разделитель выражений
-- /* abcd */	Однострочный и многострочный комментарий
% и _	Множественный и одиночный групповой символ в Like
@	Индикатор удаленного объекта
<> != ^= ~=	Не равно
||	Конкатенация
<<label>>	Метка
<= и =>	Меньше или равно / Больше или равно
:=	Присваивание

3.4 Типы данных

Таблица 3.2 – Типы данных PL/SQL

CHAR	Символьное поле фиксированной длины до 2000 байт

NCHAR	Поле фиксированной длины для набора символов, состоящих из нескольких байт. Максимальный размер – 2000 символов или 2000 байт в зависимости от набора символов.
VARCHAR2	Символьное поле переменной длины до 4000 байт
NVARCHAR2	Поле переменной длины для набора символов, состоящих из нескольких байт. Максимальный размер – 4000 символов или 4000 байт в зависимости от набора символов.
LONG	Символьный, переменной длины, до 2GB, оставлен для совместимости
RAW(n)	Переменной длины, для бинарных данных n <= 2000 byte оставлен для совместимости
LONG RAW	Бинарные данные до 2GB
CLOB	Символьный тип большой объект до 4GB
NLOB	CLOB для многобайтных символов
BLOB	Большой двоичный объект до 4GB
BFILE	Указатель на двоичный файл операционной системы
DATE	7 байтовое поле фиксированной длины, используемое для хранения даты и времени
INTERVAL DAY TO SECOND	11 байтовое поле фиксированной длины для интервала времени: Дни, часы, минуты, секунды
INTERVAL YEAR TO MONTH TIMESTAMP	5 байтовое поле фиксированной длины для интервала времени: Годы и месяцы
TIMESTAMP WITH TIME ZONE	13 байтовое поле фиксированной длины Дата, время и настройки, связанные с часовым поясом.
TIMESTAMP WITH LOCAL TIME	7-11 байтовое поле переменной длины Дата и время, приведенные к часовому поясу базы данных
NUMBER (n, s)	Числовой тип переменной длины

	Точность n <= 38, общее количество цифр Масштаб s = [-84,127], количество цифр после запятой
ROWID	64 byte уникальный адрес строки в таблице, псевдостолбец
UROWID(n)	n <= 4000, логический адрес строки в индексно-организованной таблице

Таблица 3.3 – Неявные преобразования типов данных

VARCHAR2 CHAR	DATE
DATE	VARCHAR2
VARCHAR2 CHAR	ROWID
ROWID	VARCHAR2
VARCHAR2 CHAR	NUMBER
NUMBER	VARCHAR2

3.5 Поддержка национальных языков

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, а изменяются при помощи команды: alter database (national) character set.

Основной набор символов используется для хранения символьных типов char, varchar2, clob и long, описания имен объектов, переменных,

ввода и хранения PL/SQL модулей. Дополнительный набор символов используется для: хранения символьных типов nchar, nvarchar2, nclob. Кроме символов алфавита в набор включаются знаки препинания, числа, символы денежных единиц и пр.

3.6 Оператор условного управления IF

Условным управлением называется оператор IF - THEN (если-то), который говорит о том, что при истинности условия (или наоборот, если оно ложно) нужно выполнить следующий программный блок.

Использование этого оператора позволяет разработчику структурировать код программы таким образом, что определенные операторы будут или не будут выполняться — это определяется достоверностью операции сравнения. Причем логика этой конструкции, следующая: если (IF) сравнение истинно, то (THEN) выполнить следующее. В PL/SQL существуют еще две дополнительные конструкции ELSE (иначе) и ELSIF (иначе если). Логика оператора ELSE: "в противном случае сделать то, что указано в конструкции ELSE ", а логика оператора ELSIF: "в противном случае, если...". Оператор ELSIF позволяет вам исключить дополнительное вложение оператора IF внутрь конструкции ELSE, поэтому позволяет реализовать операцию взаимоисключения более четко. Приведем простой пример:

```
WHEN_NEW_FORM_INSTACE:
 BEGIN
  IF TO_CHAR (TO_DATE (SYSDATE, 'DAY')='SATURDAY' or
     TO_CHAR (TO_DATE(SYSDATE, 'DAY')='SUNDAY'
  THEN
   set_block_property('Block1',update_allowed,
property_false);
   set_block_property('Block1',insert_allowed,
property_false);
   set_block_property('Block1',delete_allowed,
property_false);
  ELSE
   message ('Сегодня '||
     TO_CHAR (SYSDATE,'DAY')||' удачного рабочегодня');
```

```
  end if;
 end;
```

В вышеприведенном примере перед запуском формы будет проверено, выходной день сегодня или рабочий: если условие истинно, то есть сегодня выходной, то запретить какие-либо операции, связанные с манипуляцией данными, если же условие ложно, то в строке состояния появится сообщение: "Сегодня понедельник, удачного рабочего дня". Теперь рассмотрим пример с использованием оператора ELSE:

```
Declare
 but_no Varchar2(1):= :System.Mouse_Button_Pressed ;
Begin
 If but_no = '1' Then
  Message('Нажата левая кнопка');
 Else If LN$Btn = '2' Then
  Message('Нажата правая кнопка' ); End if ;
 End if;
End ;
```

Рассмотрим тот же случай, только с использованием оператора ELSIF:

```
Declare
 but_no Varchar2(1) := :System.Mouse_Button_Pressed ;
Begin
 If but_no = '1' Then
  Message('Нажата левая кнопка');
 ELSIF LN$Btn = '2' Then
  Message('Нажата правая кнопка' );
 End if;
End ;
```

Как видим, построенная конструкция IF-THEN с помощью оператора ELSIF менее громоздка и более понятна.

3.7 Циклы loop, for, while

LOOP-EXIT - это один из самых простых циклов PL/SQL. Ключевое слово LOOP указывает на начало повторяемого программного блока, а END LOOP - на его конец. Ключевое слово EXIT, указываемое отдельно, означает, что цикл нужно прервать, а ключевое слово EXIT WHEN – что предлагаемый оператор сравнения нужно проверить на необходимость завершения выполнения операторов.

Рассмотрим пример с применением такого цикла:

```
Declare
i number:=0;
Begin
 LOOP i:=C_ount+1
  insert into EXEM1 (SUM) values (i);
  IF i=10 THEN EXIT;
  end if;
 end loop;
end;
```

После выполнения этого цикла в таблицу EXEM1 будет вставлено 10 записей, причем здесь наглядно видно, что как только наше условие будет истинно, то есть значение счетчика i будет равным 10, - цикл прервется.

Теперь рассмотрим этот же пример, только с использованием оператора EXIT-WHEN:

```
Declare
i number:=0;
Begin
LOOP i:=C_ount+1;
 insert into EXEM1 (SUM) values (i);
 EXIT WHEN i=10;
end loop;
end;
```

Из примера видно, что такая конструкция, в отличие от предыдущей, дает возможность разработчику писать менее громоздкий код.

WHILE -LOOP - цикл этого типа похож на цикл с условием LOOP-EXIT WHEN, разница заключается в том, что в WHILE -LOOP проверка условия выхода осуществляется в начале цикла. Если сравнивать два этих цикла, то по гибкости, конечно, выигрывает LOOP-EXIT, т. к. позволяет указывать условие EXIT в любом месте цикла, а если сравнивать по элегантности и краткости оформления кода, то WHILE -LOOP выигрывает.

Рассмотрим тот же самый пример, что и в предыдущем случае, только с использованием конструкции WHILE -LOOP:

```
Declare
i number:=0;
Begin
 while i<=10
  LOOP i:=C_ount+1;
   insert into EXEM1 (SUM) values (i);
```

```
  end loop;
end;
```

В этом примере мы еще больше упростили наш цикл, т. к. не использовали оператор EXIT.

FOR_LOOP - этот цикл отличается от двух других, т. к. позволяет указать конкретное количество выполнений программы до ее прерывания. Данная циклическая конструкция состоит из счетчика и диапазона циркуляции цикла. Счетчик в операторах FOR -LOOP является встроенным и автоматически увеличивает значение на единицу. Диапазон циркуляции определяют нижняя и верхняя граница, причем границы должны иметь целочисленный тип. Нижняя граница диапазона циркуляции цикла может начинаться не только с единицы, но и с любого другого целого числа.

Для примера создадим таблицу EXEM1 с одним столбцом типа number - SUM, который содержит различные числовые значения. Теперь выполним цикл:

```
DECLARE
C_OUNT NUMBER;
BEGIN
 for i in 1..10 loop
  insert into EXEM1 (SUM)
  values (i);
  C_OUNT:=C_OUNT+i;
 end loop;
end;
```

После выполнения этого цикла в таблицу EXEM1 будет вставлено 10 записей с значением от 1 до 10. В рассмотренном нами цикле встроенным счетчиком будет переменная i, а диапазоном циркуляции будет интервал от 1 до 10, где 1 — это нижняя граница диапазона, а 10 - верхняя. Если рассмотреть пример, который мы приводили в двух предыдущих случаях, но с применением операторов FOR-LOOP, то вы увидите, что использование этой конструкции делает код элегантнее:

```
Declare
 i number:=0;
Begin
 for i in 1..10 LOOP
  insert into EXEM1 (SUM) values (i);
```

```
 end loop;
end;
```

FOR -LOOP также позволяет исполнять цикл в другом направлении с помощью оператора REVERSE, причем при изменении направления цикла вам не надо как-то иначе задавать границы диапазона, т. к. Oracle сделает это за вас:

```
Declare
 i number:=0;
Begin
 for i in REVERSE 1..10 LOOP
  insert into EXEM1 (SUM)
  values (i);
 end loop;
end;
```

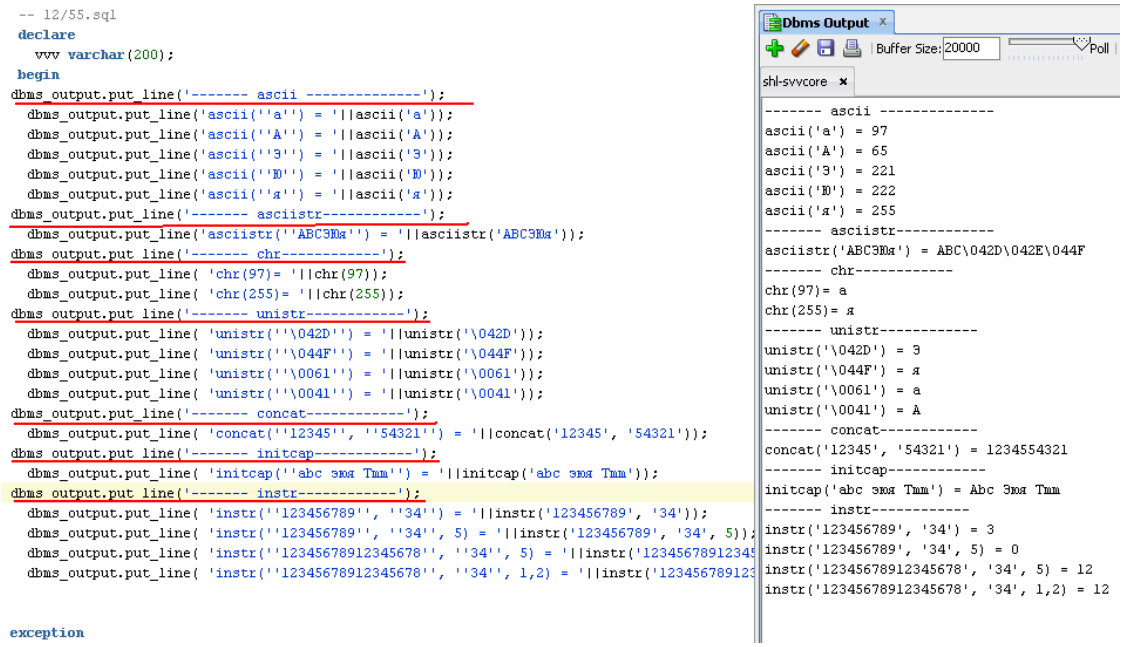
3.8 Встроенные функции

Рассмотрим использование встроенных функций PL/SQL на примерах ниже.

```
begin
  dbms_output.put_line('abs(-10.493) = '||abs(-10.493) );
  dbms_output.put_line('ceil(-10.493) = '||ceil(-10.493) );
  dbms_output.put_line('round(-10.493) = '||round(-10.493));
  dbms_output.put_line('round(-10.493,1) = '||round(-10.493,1));
  dbms_output.put_line('trunc(-10.493,1) = '||trunc(-10.493,1));
  dbms_output.put_line('floor(-10.493) = '||floor(-10.493));
  dbms_output.put_line('floor(10.493) = '||floor(10.493));
  dbms_output.put_line('remainder(10,3) = '||remainder(10,3));  -- 10%3
  dbms_output.put_line('remainder(-10,3) = '||remainder(-10,3));  -- 10%3
  dbms_output.put_line('mod(-10,3) = '||mod(-10,3));
  dbms_output.put_line('bitand(0,1) = '||to_number(bitand(0,1)));
  dbms_output.put_line('bitand(15,7) = '||to_number(bitand(15,7)));
  dbms_output.put_line('bitand(5,3) = '||to_number(bitand(5,3)));
  -- dbms_output.put_line('width_bucket(21, 0,100,10) = '||WIDTH_BUCKET (21,
  dbms_output.put_line('cos(3.14/180*60) = '||cos(3.14/180*60));
  dbms_output.put_line('acos(0.5) = '||acos(0.5)/3.14*180);
  dbms_output.put_line('sin(3.14/180*60) = '||sin(3.14/180*60));
  dbms_output.put_line('asin(0.5) = '||asin(0.5)/3.14*180);
  dbms_output.put_line('tan(3.14/180*60) = '||tan(3.14/180*60));
  dbms_output.put_line('atan(0.5) = '||atan(0.5)/3.14*180);
  dbms_output.put_line('exp(1) = '||exp(1));
  dbms_output.put_line('exp(0) = '||exp(0));
  dbms_output.put_line('power(5,2) = '||power(5,2));
  dbms_output.put_line('power(25,1/2) = '||power(25,1/2));
  dbms_output.put_line('power(5,-2) = '||power(5,-2));
  dbms_output.put_line('sqrt(16) = '||sqrt(16));
  dbms_output.put_line('log(100,10) = '||log(100,10));
  dbms_output.put_line('log(10,100) = '||log(10,100));
  dbms_output.put_line('log(2,16) = '||log(2,16));
  dbms_output.put_line('sign(-25.7) = '||sign(-25.7));
  dbms_output.put_line('sign(25.7) = '||sign(25.7));
  dbms_output.put_line('sign(0) = '||sign(0));
  dbms_output.put_line('greatest(-1,3,45,9,1) = '||greatest(-1,3,45,9,1));
```

```
abs(-10.493) = 10,493
ceil(-10.493) = -10
round(-10.493) = -10
round(-10.493,1) = -10,5
trunc(-10.493,1) = -10,4
floor(-10.493) = -11
floor(10.493) = 10
remainder(10,3) = 1
remainder(-10,3) = -1
mod(-10,3) = -1
bitand(0,1) = 0
bitand(15,7) = 7
bitand(5,3) = 1
cos(3.14/180*60) = ,5004596890082057
acos(0.5) = 60,030432871142545957884
sin(3.14/180*60) = ,8657598394923444
asin(0.5) = 30,015216435571272978942
tan(3.14/180*60) = 1,729929220089790
atan(0.5) = 26,578525356734108572791
exp(1) = 2,718281828459045235360287 4
exp(0) = 1
power(5,2) = 25
power(25,1/2) = 5,000000000000000000
power(5,-2) = ,04
sqrt(16) = 4
log(100,10) = ,5
log(10,100) = 2
log(2,16) = 3,999999999999999999999
sign(-25.7) = -1
sign(25.7) = 1
sign(0) = 0
greatest(-1,3,45,9,1) = 45
```

Рисунок 3.3 – Числовые функции

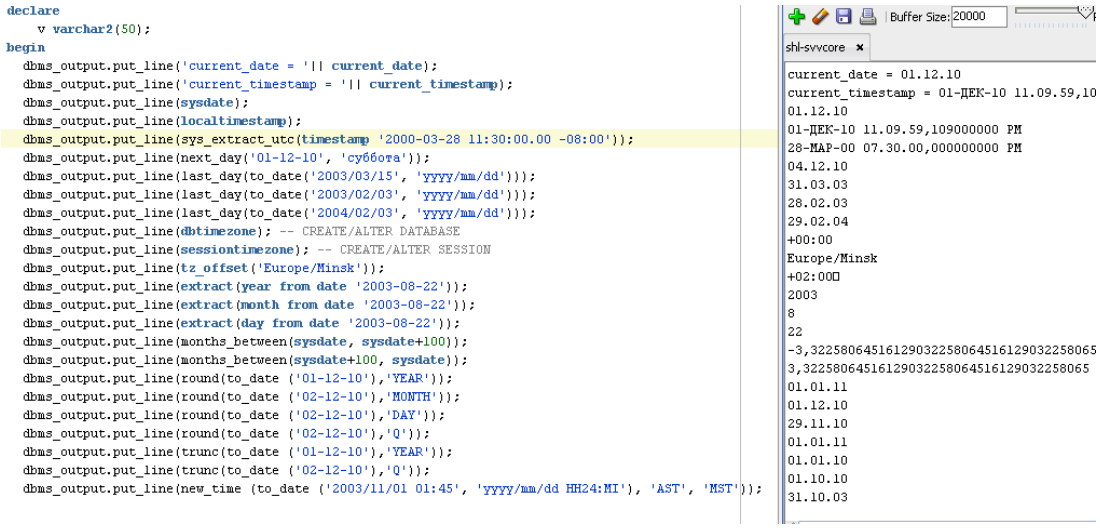


```
-- 12/55.sql
declare
  vvv varchar(200);
begin
dbms_output.put_line('------- ascii --------------');
  dbms_output.put_line('ascii(''a'') = '||ascii('a'));
  dbms_output.put_line('ascii(''A'') = '||ascii('A'));
  dbms_output.put_line('ascii(''Э'') = '||ascii('Э'));
  dbms_output.put_line('ascii(''Ю'') = '||ascii('Ю'));
  dbms_output.put_line('ascii(''я'') = '||ascii('я'));
dbms_output.put_line('------- asciistr------------');
  dbms_output.put_line('asciistr(''ABCЭЮя'') = '||asciistr('ABCЭЮя'));
dbms_output.put_line('------- chr------------');
  dbms_output.put_line( 'chr(97)= '||chr(97));
  dbms_output.put_line( 'chr(255)= '||chr(255));
dbms_output.put_line('------- unistr------------');
  dbms_output.put_line( 'unistr(''\042D'') = '||unistr('\042D'));
  dbms_output.put_line( 'unistr(''\044F'') = '||unistr('\044F'));
  dbms_output.put_line( 'unistr(''\0061'') = '||unistr('\0061'));
  dbms_output.put_line( 'unistr(''\0041'') = '||unistr('\0041'));
dbms_output.put_line('------- concat------------');
  dbms_output.put_line( 'concat(''12345'', ''54321'') = '||concat('12345', '54321'));
dbms_output.put_line('------- initcap------------');
  dbms_output.put_line( 'initcap(''abc эюя Tmm'') = '||initcap('abc эюя Tmm'));
dbms_output.put_line('------- instr------------');
  dbms_output.put_line( 'instr(''123456789'', ''34'') = '||instr('123456789', '34'));
  dbms_output.put_line( 'instr(''123456789'', ''34'', 5) = '||instr('123456789', '34', 5));
  dbms_output.put_line( 'instr(''12345678912345678'', ''34'', 5) = '||instr('12345678912345
  dbms_output.put_line( 'instr(''12345678912345678'', ''34'', 1,2) = '||instr('12345678912


exception
```

```
Dbms Output
Buffer Size: 20000
shl-svvcore
------- ascii --------------
ascii('a') = 97
ascii('A') = 65
ascii('Э') = 221
ascii('Ю') = 222
ascii('я') = 255
------- asciistr------------
asciistr('ABCЭЮя') = ABC\042D\042E\044F
------- chr------------
chr(97)= a
chr(255)= я
------- unistr------------
unistr('\042D') = Э
unistr('\044F') = я
unistr('\0061') = a
unistr('\0041') = A
------- concat------------
concat('12345', '54321') = 1234554321
------- initcap------------
initcap('abc эюя Tmm') = Abc Эюя Tmm
------- instr------------
instr('123456789', '34') = 3
instr('123456789', '34', 5) = 0
instr('12345678912345678', '34', 5) = 12
instr('12345678912345678', '34', 1,2) = 12
```

Рисунок 3.4 – Символьные функции



```
declare
    v varchar2(50);
begin
  dbms_output.put_line('current_date = '|| current_date);
  dbms_output.put_line('current_timestamp = '|| current_timestamp);
  dbms_output.put_line(sysdate);
  dbms_output.put_line(localtimestamp);
  dbms_output.put_line(sys_extract_utc(timestamp '2000-03-28 11:30:00.00 -08:00'));
  dbms_output.put_line(next_day('01-12-10', 'суббота'));
  dbms_output.put_line(last_day(to_date('2003/03/15', 'yyyy/mm/dd')));
  dbms_output.put_line(last_day(to_date('2003/02/03', 'yyyy/mm/dd')));
  dbms_output.put_line(last_day(to_date('2004/02/03', 'yyyy/mm/dd')));
  dbms_output.put_line(dbtimezone); -- CREATE/ALTER DATABASE
  dbms_output.put_line(sessiontimezone); -- CREATE/ALTER SESSION
  dbms_output.put_line(tz_offset('Europe/Minsk'));
  dbms_output.put_line(extract(year from date '2003-08-22'));
  dbms_output.put_line(extract(month from date '2003-08-22'));
  dbms_output.put_line(extract(day from date '2003-08-22'));
  dbms_output.put_line(months_between(sysdate, sysdate+100));
  dbms_output.put_line(months_between(sysdate+100, sysdate));
  dbms_output.put_line(round(to_date ('01-12-10'),'YEAR'));
  dbms_output.put_line(round(to_date ('02-12-10'),'MONTH'));
  dbms_output.put_line(round(to_date ('02-12-10'),'DAY'));
  dbms_output.put_line(round(to_date ('02-12-10'),'Q'));
  dbms_output.put_line(trunc(to_date ('01-12-10'),'YEAR'));
  dbms_output.put_line(trunc(to_date ('02-12-10'),'Q'));
  dbms_output.put_line(new_time (to_date ('2003/11/01 01:45', 'yyyy/mm/dd HH24:MI'), 'AST', 'MST'));
```

```
Buffer Size: 20000
shl-svvcore
current_date = 01.12.10
current_timestamp = 01-ДЕК-10 11.09.59,10
01.12.10
01-ДЕК-10 11.09.59,109000000 PM
28-МАР-00 07.30.00,000000000 PM
04.12.10
31.03.03
28.02.03
29.02.04
+00:00
Europe/Minsk
+02:00
2003
8
22
-3,32258064516129032258064516129032258065
3,3225806451612903225806451612903225806 5
01.01.11
01.12.10
29.11.10
01.01.11
01.01.10
01.10.10
31.10.03
```

Рисунок 3.5 – Функции по работе с датами

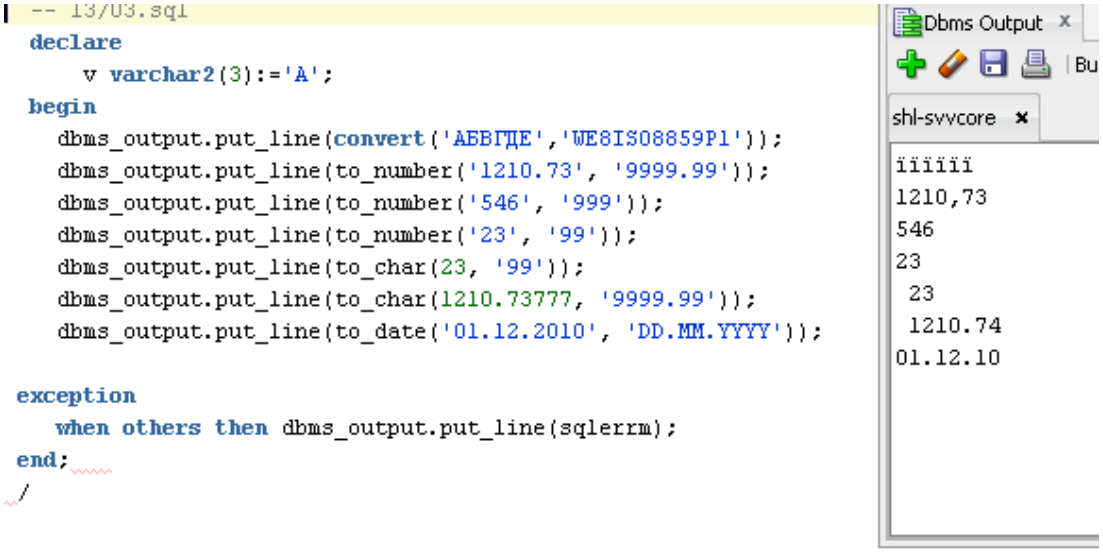


Рисунок 3.6 – Конвертирование типов

Функции обработки ошибок

Функция SQLERRM возвращает сообщение об ошибке, связанной с исключительной ситуацией, а функция SQLCODE возвращает номер ошибки, связанной с исключительной ситуацией.

3.9 Коллекции, записи, переменные 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 в процедуру или функцию. Можно создавать коллекции коллекций.

Запись – структура данных, составленная из нескольких частей информации, называемых полями. Для объявления записи вначале надо определить как тип, а потом объявить переменную типа «запись».

Типы записей:

- Табличные
- Курсорные
- Программно-определенные

3.10 Процедуры

Локальный программный модуль – это процедура или функция, определенная в секции декларации PL/SQL блока. Объявление локальных процедур и функций должно размещаться в конце секции декларации после всех типов, записей, курсоров, переменных и исключений. Локальные процедуры и функции могут быть использованы только в рамках блока, в котором они объявлены.

Локальные процедуры и функции могут быть перегружены. Параметры должны отличаться семейством (number, character, datetime, boolean). Тип программного модуля должен отличаться – можно перегружать процедуру и функцию с одинаковым именем и списком параметров. Число параметров должно быть разным.

Локальные процедуры

Например, если перед вами стоит задача запретить удаление строки, и если она единственная, то вместо того чтобы несколько раз писать один и тот же блок кода, можно обернуть его в процедуру.

```
PROCEDURE DONT_DELETE_LAST_RECORD
IS
BEGIN
 IF :System.Last_Record = 'TRUE'
 AND :System.Cursor_Record ='1' THEN
  Message('Последняя запись не может быть удалена');
 ELSE
  Delete_Record;
 END IF;
END;
```

Локальные функции

Фраза " RETURN … IS " на рис.3.7 указывает на тип возвращаемого значения функцией.

```
-- 13/09.sql
declare
  x number(3):= 4;
  y number(3):= 5;
  z number(3);
  s number(5);
  function  summod5 (x1 number, x2 number, x3 out number)
    return number  is
        z number(3):= 5;
    begin
      x3 := mod(x1+ x2,z);
      return (x1+x2);
    end  summod5;
begin
    s := summod5(x,y,z);
    dbms_output.put_line('s = '||s);
    dbms_output.put_line('z = '||z);

 exception
   when others then dbms_output.put_line(sqlerrm);
end;
/
```

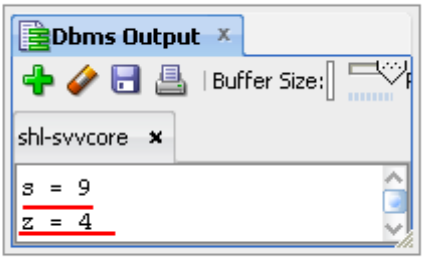


Рисунок 3.7 – Пример локальной функции

Ассоциативные массивы (индексные таблицы) – одномерные, неограниченные (по максимальному количеству элементов при создании) коллекции элементов. Доступны только в рамках PL/SQL. Изначально являются разреженными, индекс могут принимать непоследовательные значения.

Вложенные таблицы (nested tables) – одномерные, несвязанные коллекции однотипных элементов. Доступны в рамках PL/SQL и как поля таблицы в БД. Изначально являются плотными, но могут впоследствии становиться разреженными.

Массивы переменной длины (VARRAY) – одномерные, связанные коллекции однотипных элементов(рис.3.8). Доступны в рамках PL/SQL и в БД. Являются плотными.

```
-- 13/14.sql
declare
    type myarraytype  is varray(3) of number;
    va myarraytype := myarraytype(null,null,null);

begin
   for i in 1..3
   loop
     va(i) := i;
   end loop;
   for i in 1..3
   loop
    dbms_output.put_line(va(i));
   end loop;

exception
  when others then dbms_output.put_line(sqlerrm);
end;
/
```

```
Dbms Output
Buffe
shl-svvcore
1
2
3
```

```
-- 13/18.sql
declare
    type myarraytype  is varray(3) of number not null;
    va myarraytype := myarraytype();

begin
   dbms_output.put_line('va.count = '||va.count);

   va.extend;

   if va(1) is null
   then  dbms_output.put_line('va(1) = null '||va(1));
   else dbms_output.put_line('va(1) = '||va(1));
   end if;
   dbms_output.put_line('va.count = '||va.count);

   va.extend;
   dbms_output.put_line('va.count = '||va.count);

   va.extend;
   dbms_output.put_line('va.count = '||va.count);

   va.extend;
   dbms_output.put_line('va.count = '||va.count);
exception
  when others then dbms_output.put_line(sqlerrm);
end;
/
```

```
Dbms Output
Buffer Size: 20000
shl-svvcore
va.count = 0
va(1) = null
va.count = 1
va.count = 2
va.count = 3
ORA-06532: Индекс превышает пределы
```

Рисунок 3.8 – Массивы переменной длины

3.11 Триггеры

Триггер уровня DML является объектом реляционной базы данных (специальный вид хранимой процедуры), который активизирует выполнение хранимой (или встроенной) PL/SQL-процедуры при изменении пользователем данных в таблице. Событие, управляющее запуском триггера, описывается в виде логических условий. Например, попытка модифицировать данные в таблице активизирует триггер, соответствующий

данной команде манипулирования данными. Число триггеров на таблицу базы данных не ограничено.

Обычно триггеры используют для реализации ограничений ссылочной целостности, для предотвращения несогласованных изменений в базе данных (поддержка целостности базы данных), для выполнения скрытых операций при модификации, а также для снижения сетевого трафика за счет передачи обработки на сервер. Операции завершения транзакции выполняются после обработки триггеров.

Триггер уровня DML является объектом реляционной базы данных (специальный вид хранимой процедуры), который активизирует выполнение хранимой (или встроенной) PL/SQL-процедуры при изменении пользователем данных в таблице. Событие, управляющее запуском триггера, описывается в виде логических условий. Например, попытка модифицировать данные в таблице активизирует триггер, соответствующий данной команде манипулирования данными. Число триггеров на таблицу базы данных не ограничено. Oracle поддерживает три вида триггеров:

- предваряющие (BEFORE),
- замещающие (INSTEAD OF)
- завершающие (AFTER).

Предваряющие триггеры вызываются перед обработ-кой запроса на вставку, обновление или удаление, замещающие - вместо него, а завершающие - после обработки запроса. Всего имеется девять воз-можных типов триггеров: предваряющий триггер вставки, обновления и удаления, замещающий триггер вставки, обновления и удаления и завер-шающий триггер вставки, обновления и удаления.

Обычно триггеры используют для реализации ограничений ссылочной целостности, для предотвращения несогласованных изменений в базе данных (поддержка целостности базы данных), для выполнения скрытых операций при модификации, а также для снижения сетевого

трафика за счет передачи обработки на сервер. Операции завершения транзакции выполняются после обработки триггеров.

При выполнении команды UPDATE с помощью триггера можно проверить, что модифицируемые данные удовлетворяют ограничениям целостности базы данных до выполнения операции (при этом возможен доступ к новым данным!). После выполнения операции с помощью триггера можно выполнить скрытую обработку данных с учетом поступивших изменений (старые данные также могут быть доступны).

При выполнении команды INSERT также можно проверить данные до вставки в таблицу на допустимость ограничениям целостности, а после - выполнить операции над только что вставленными данными.

При выполнении команды DELETE можно проверить данные до их удаления или восстановить данные после удаления.

Синтаксис команды CREATE TRIGGER

Для создания триггера предусмотрена специальная команда SQL CREATE TPIGGER. Эта команда создает триггер на таблице, которой владеет пользователь (рис.3.9).

Рисунок 3.9 – Схема создания триггера

Ключевое слово OR REPLACE указывает на безусловное замещение старого теста триггера. Если оно не указывается, а триггер определен в базе данных, то замещения старого триггера не происходит, и возвращается сообщение об ошибке.

Определение триггера состоит из нескольких частей:

- задание имени триггера;
- указание команды SQL, к которой относится триггер;

- указание таблицы или представления, для которой определяется триггер;
- задание ограничений триггера;
- задание действия в теле триггера.

Если имя схемы опущено, то триггер создается в схеме текущего пользователя.

{BEFORE|AFTER} - время действия триггера- до или после выполнения команды манипулирования данными. Нельзя определить два триггера на одну и ту же операцию с одинаковым временем действия.

При создании триггера необходимо указывать, к какой команде манипулирования данными он относится - INSERT, DELETE или UPDATE. Для последней можно указывать конкретные колонки, указав фразу OF имя_колонки [, имя_колонки ...] в предложении UPDATE.

Ключевое слово ON задает имя таблицы или представления, для которого создается триггер.

Необязательное ключевое слово ON EACH ROW определяет триггер как строчный, т.е. запускаемый для каждой строки результирующего множества команды SQL. Если оно опущено, то триггер запускается только один раз в начале обработки команды. Таким образом, условие "для каждой строки" активизируется, только когда есть строки (например, предложение WHERE дает истинное значение условий поиска), в то время как для условия "для каждой команды" триггер сработает и в этом случае.

Дополнительные условия, сужающие область действия триггера, могут быть заданы в предложении WHEN. Условия, задаваемые в этом предложении, являются стандартными для SQL условиями, должны содержать корреляционные имена и не могут содержать запрос. Это предложение может быть указано только для строчного триггера.

Необязательное ключевое слово ON EACH ROW определяет триггер как строчный, т.е. запускаемый для каждой строки результирующего

множества команды SQL. Если оно опущено, то триггер запускается только один раз в начале обработки команды. Таким образом, условие "для каждой строки" активизируется, только когда есть строки (например, предложение WHERE дает истинное значение условий поиска), в то время как для условия "для каждой команды" триггер сработает и в этом случае.

Дополнительные условия, сужающие область действия триггера, могут быть заданы в предложении WHEN. Условия, задаваемые в этом предложении, являются стандартными для SQL условиями, должны содержать корреляционные имена и не могут содержать запрос. Это предложение может быть указано только для строчного триггера.

Участие в транзакциях

По умолчанию триггеры DML принимают участие в транзакциях, из которых они запускаются, т.е. если триггер инициирует исключение, то соответствующая часть транзакции будет отменена. Если триггер сам выполняет какие-то операторы DML (например, вставляет запись в журнальную таблицу), то такие операторы DML становятся частью главной транзакции. Внутри триггера DML нельзя использовать операторы COMMIT и ROLLBACK. Если использовать триггер DML как автономную транзакцию, то любые команды DML, исполняемые внутри триггера, будут сохранены или отменены посредством явно использованного оператора COMMIT или ROLLBACK, при этом главная транзакция затрагиваться не будет

Псевдозаписи NEW и OLD

При запуске триггера уровня строки исполняющее ядро PL/SQL создаёт и заполняет две структуры данных, которые работают подобно записям. Это псевдозаписи NEW и OLD (приставка «псевдо» объясняется тем, что они обладают не всеми свойствами настоящих записей PL/SQL). OLD хранит начальные значения записи, обрабатываемой триггером, а NEW — новые значения. Эти записи имеют такую же структуру, как запись,

объявленная при помощи атрибута %ROWTYPE на основе таблицы, к которой относится триггер.

Правила при работе с псевдозаписями NEW и OLD:

Для триггеров, относящихся к командам INSERT, структура OLD не содержит никаких данных, «старого» набора значений не существует. Для триггеров, относящихся к командам DELETE, структура NEW не содержит никаких данных, т.к. речь идёт об удалении записи. Изменение значений полей структуры OLD запрещено, попытка такого изменения приведёт к возникновению ошибки ORA-04085. Изменение значений полей структуры NEW допустимо.

Структура NEW Или OLD не может передаваться как параметр типа запись в процедуру или функцию, вызываемую внутри триггера. Можно передавать только отдельные поля псевдозаписей. При ссылке на структуру NEW или OLD внутри анонимного блока триггера необходимо предварять двоеточием соответствующие ключевые слова, например: IF :NEW.salary > 10000 THEN ...

Выполнение операций уровня записи для структур NEW и OLD не поддерживается. Например, подобный код вызовет ошибку при компиляции триггера: BEGIN :NEW := NULL; END;

Oracle предлагает набор функций (называемых также операционными директивами), которые позволяют определить, какое DML-действие вызвало запуск текущего триггера. Каждая такая функция возвращает TRUE или FALSE.

- ***INSERTING*** возвращает TRUE, если триггер был запущен в ответ на вставку в таблицу, с которой связан триггер, и FALSE — в противном случае.
- ***UPDATING*** возвращает TRUE, если триггер был запущен в ответ на обновление таблицы, с которой связан триггер, и FALSE — в

противном случае.

- ***DELETING*** возвращает TRUE, если триггер был запущен в ответ на удаление из таблицы, с которой связан триггер, и FALSE — в противном случае.

Используя эти директивы, можно создать один общий триггер, который будет объединять действия, необходимые для различных видов операций.

Oracle позволяет определить триггеры, которые будут запускаться в ответ на исполнение операторов языка DDL (Data Definition Language — язык определения данных). Оператор DDL — это любой оператор SQL, используемый для создания или изменения объекта базы данных, такого как таблица или индекс.

Каждый из этих операторов приводит к созданию, изменению или удалению объекта базы данных.

Несколько примеров операторов DDL:

- CREATE TABLE
- ALTER INDEX
- DROP TRIGGER

Синтаксис создания таких триггеров практически совпадает с синтаксисом создания триггеров DML, отличие лишь в запускающих их событиях и в том, что триггеры DDL не применяются к отдельным таблицам.

Создание триггера DDL

Для создания (или пересоздания) триггера DDL используйте такою конструкцию:

CREATE [OR REPLACE] TRIGGER имя_триггера
{ BEFORE | AFTER } {DDL-событие } ON { DATABASE | SCHEMA }
[WHEN (...) }

DECLARE

Объявления переменных

```
BEGIN
... код ...
END;
```

Таблица 3.4 – Триггерные события DDL

CREATE	Событие возникает, когда создается объект базы данных.
ALTER	Событие возникает, когда выполняется команда ALTER над объектом базы данных.
DROP	Событие возникает, когда выполняется команда удаления объекта DROP.
SERVERERROR	Событие возникает, когда сервер генерирует ошибку. Допустимы только AFTER триггеры.
LOGON	Событие возникает, когда создается соединение пользователя с базой данных. Допустимы только AFTER триггеры.
LOGOFF	Событие возникает, когда завершается соединение пользователя с базой данных. Допустимы только BEFORE триггеры.
STARTUP	Событие возникает, когда база данных открывается для работы. Допустимы только AFTER триггеры.
SHUTDOWN	Событие возникает, когда база данных закрывается. Допустимы только BEFORE триггеры.
BEFORE GRANT AFTER GRANT	При выполнении команды grant
BEFORE REVOKE AFTER REVOKE	При выполнении команды revoke
BEFORE DDL AFTER DDL	Срабатывает на большинство команд DDL, кроме: alter database, create control file, create database
BEFORE TRUNCATE AFTER TRUNCATE	При выполнении команды truncate

К объектам события относятся таблицы, пакеты и другие объекты базы данных, которые можно найти в системном представлении

ALL_OBJECTS. Может применяться к отдельной схеме или базе данных в целом.

Триггеры событий базы данных

Эти триггеры представляют собой средство автоматизации процесса администрирования базы данных и обеспечения детального контроля над базой данных.

Существует ряд ограничений, накладываемых на использование атрибутов **BEFORE** и **AFTER** для определённых событий. Некоторые ситуации представляются просто бессмысленными:

Не бывает триггеров **BEFORE STARTUP, AFTER SHUTDOWN, BEFORE LOGON**, **AFTER LOGOFF, BEFORE SERVERERROR**. Попытки создания таких триггеров приводит к появлению сообщений об ошибке подобных следующей:

ORA-30500: database open triggers and server error triggers cannot have BEFORE type.

Синтаксис, используемый для создания такого триггера, очень похож на синтаксис создания триггера DDL:

```
CREATE [ OR REPLACE ] TRIGGER имя_триггера { BEFORE | AFTER } {
событие_базы_данных } ON  { DATABASE | SCHEMA }
DECLARE
Объявление переменных
BEGIN
... код ...
END;
```

ЛАБОРАТОРНАЯ РАБОТА №3
СОЗДАНИЕ ХРАНИМЫХ ПРОЦЕДУР И ФУНКЦИЙ В ORACLE

Цель работы

1. Научиться создавать хранимые процедуры и функции

Знания, необходимые для выполнения лабораторной работы

1. Знание синтаксиса и схем создания хранимых процедур и функций.

Задание

1. Создать хранимые процедуры согласно своему варианту.

Контрольные вопросы

1. Перечислите системные имена пользователей, автоматически генерируемые СУБД Oracle. Опишите роль DBA.
2. Что такое табличное пространство? Опишите основные типы табличных пространств.
3. Перечислите основные объекты БД Oracle.
4. Опишите типы данных параметров процедур Oracle. Как происходит передача параметров?
5. Опишите синтаксис и вызов процедур.

ТЕМА 4. СОЗДАНИЕ ПРИЛОЖЕНИЙ В ORACLE APEX

4.1 Темы для интерфейса

Разработчики могут настроить макет страницы своего приложения базы данных, используя глобальную страницу, настраивая регионы и редактируя атрибуты элементов.

Темы — это наборы шаблонов, которые позволяют разработчикам определять макет и стиль всего приложения.

Темы предоставляют разработчикам полный набор шаблонов, которые учитывают каждый шаблон пользовательского интерфейса, который может потребоваться в приложении. Шаблоны упорядочиваются сначала по типу шаблона, а затем по классу шаблона. Типы шаблонов включают страницу, регион, отчет, список, кнопку, метку и всплывающий список значений.

Каждый тип шаблона имеет несколько классов шаблонов. Класс шаблона определяет цель шаблона в типе шаблона. Например, шаблон области можно классифицировать как шаблон области формы, шаблон области отчета и т. Д. Эти классификации позволяют Oracle Application Express сопоставлять шаблоны между темами, упрощая быстрое изменение всего внешнего вида приложения.

Администраторы могут добавлять темы в репозиторий тем следущим образом:

- Темы рабочего пространства – администраторы рабочего пространства могут создавать темы, доступные всем разработчикам в рабочем пространстве.
- Общедоступные темы – администраторы экземпляров могут создавать общедоступные темы, добавляя их в службы администрирования Oracle Application Express. После добавления эти общедоступные темы доступны всем разработчикам во всех рабочих областях экземпляра

Универсальная тема – отличается адаптивным дизайном и позволяет разработчикам создавать веб-приложения без глубоких знаний HTML, CSS или JavaScript.

Универсальная тема отличается гибким дизайном, универсальными компонентами пользовательского интерфейса и легко настраивается.

Приложения, которые вы создаете с помощью мастера создания приложений, используют универсальную тему. Преимущества универсальной темы:

- **Адаптивный дизайн** – разработан для работы на устройствах с маленьким экраном (например, смартфоны и планшеты) так же хорошо, как и на устройствах с большим экраном (включая ноутбуки и настольные компьютеры). Компоненты пользовательского интерфейса в универсальной теме работают с различными разрешениями экрана, сохраняя при этом одинаковые или похожие функции. Кроме того, Universal Theme в полной мере использует преимущества сверхвысокого разрешения экрана, используя векторную графику, где это возможно, и полагаясь на функции CSS3 для стилизации пользовательского интерфейса.
- **Универсальный пользовательский интерфейс** – предоставляет все компоненты и строительные блоки, необходимые для создания пользовательского интерфейса практически любого типа бизнес-приложений.
- **Легкая настройка** – легко настраивайте и полностью управляйте внешним видом ваших приложений, не становясь экспертом в области дизайна пользовательского интерфейса, HTML, CSS или JavaScript. Используя Theme Roller и параметры шаблона, вы можете легко настроить свое приложение в соответствии с брендом вашей компании и настроить внешний вид различных компонентов с помощью параметров шаблона.
- **Поддержка стилей тем** – универсальная тема включает поддержку стилей тем. Стиль темы — это таблица стилей CSS, которая

добавляется к базовому CSS. Разработчики могут изменить внешний вид приложения, изменив стиль темы с помощью утилиты Theme Roller.

4.2 Разработка элементов навигации

Используйте навигационную цепочку в качестве второго уровня навигации вверху каждой страницы, дополняя другие элементы пользовательского интерфейса, такие как вкладки и списки.

Навигационная цепочка – это иерархический список ссылок, который указывает, где находится пользователь в приложении с иерархической точки зрения (рис.4.1). Пользователи могут щелкнуть конкретную ссылку навигации, чтобы мгновенно просмотреть страницу.

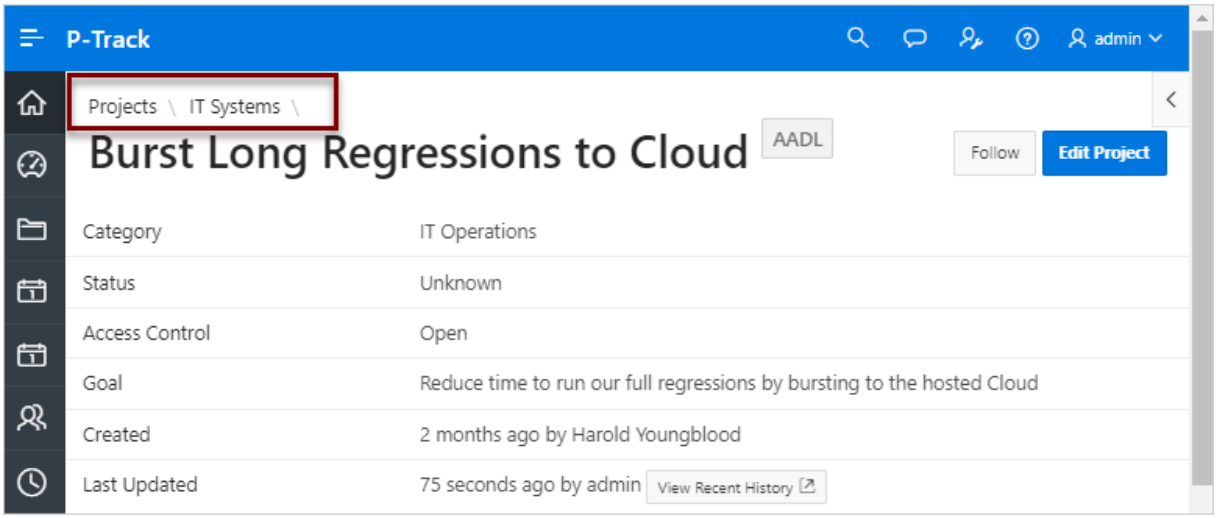


Рисунок 4.1 – Навигационная цепочка ссылок

Создание цепочки навигации при создании страницы

Создайте навигационную цепочку при создании новой страницы.

Чтобы создать навигационную цепочку при запуске мастера создания страницы:

1. Запустите мастер создания страницы, чтобы добавить новую страницу.

Во время работы мастера появляется опция «breadcrumbs». Фактическая страница, на которой отображается этот список, зависит от типа создаваемой вами страницы.

2. В списке «Breadcrumb» выберите «**Breadcrumb»** или выберите существующую breadcrumb (если применимо).

3. Если вы выберете Breadcrumb:
   a. Родительская запись - выберите родительскую страницу (если применимо) или выберите **No parent entry**.
   b. Имя записи - введите имя для «breadcrumb».

4. Следуйте инструкциям на экране.

4.3 Создание шаблонов форм

Разработчики могут создавать множество различных форм приложений вручную или с помощью мастеров. Некоторые формы позволяют пользователям обновлять одну строку в таблице, а другие формы позволяют пользователям обновлять несколько строк в таблице.

При создании приложения базы данных разработчики могут создать четыре основных типа форм: редактируемая интерактивная сетка, форма, отчет с формой или основная деталь.

Редактируемая интерактивная сетка

Интерактивная сетка представляет пользователям набор данных в настраиваемом отчете с возможностью поиска. В редактируемой интерактивной сетке пользователи также могут добавлять, изменять и обновлять набор данных непосредственно на странице (рис.4.2). Функционально интерактивная сетка включает в себя большинство возможностей настройки, доступных в интерактивных отчетах, а также возможность изменять порядок отчета в интерактивном режиме с помощью мыши. Ниже приведен пример редактируемой интерактивной сетки.

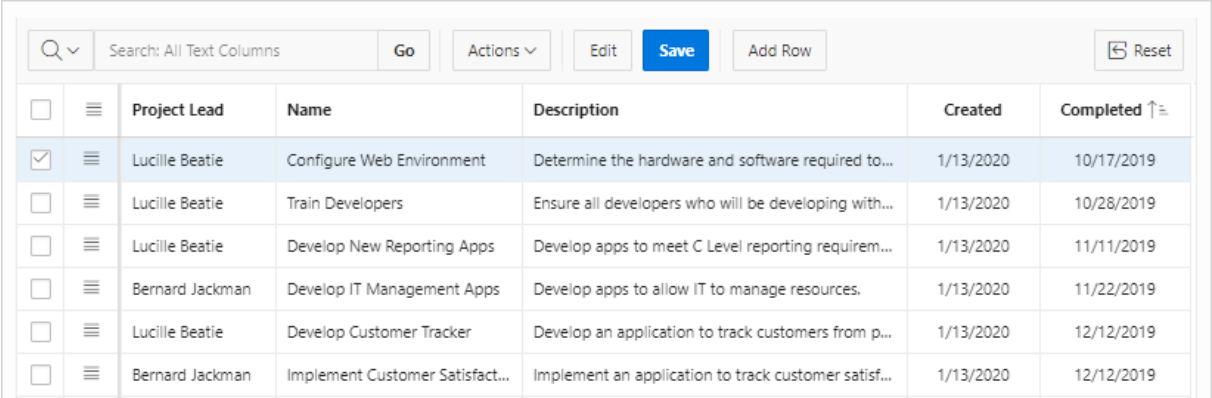


Рисунок 4.2 – Интерактивная сетка

Форма

Form создает форму, которая позволяет пользователям обновлять одну строку в таблице базы данных. На рис.4.3 приведен пример формы в таблице.

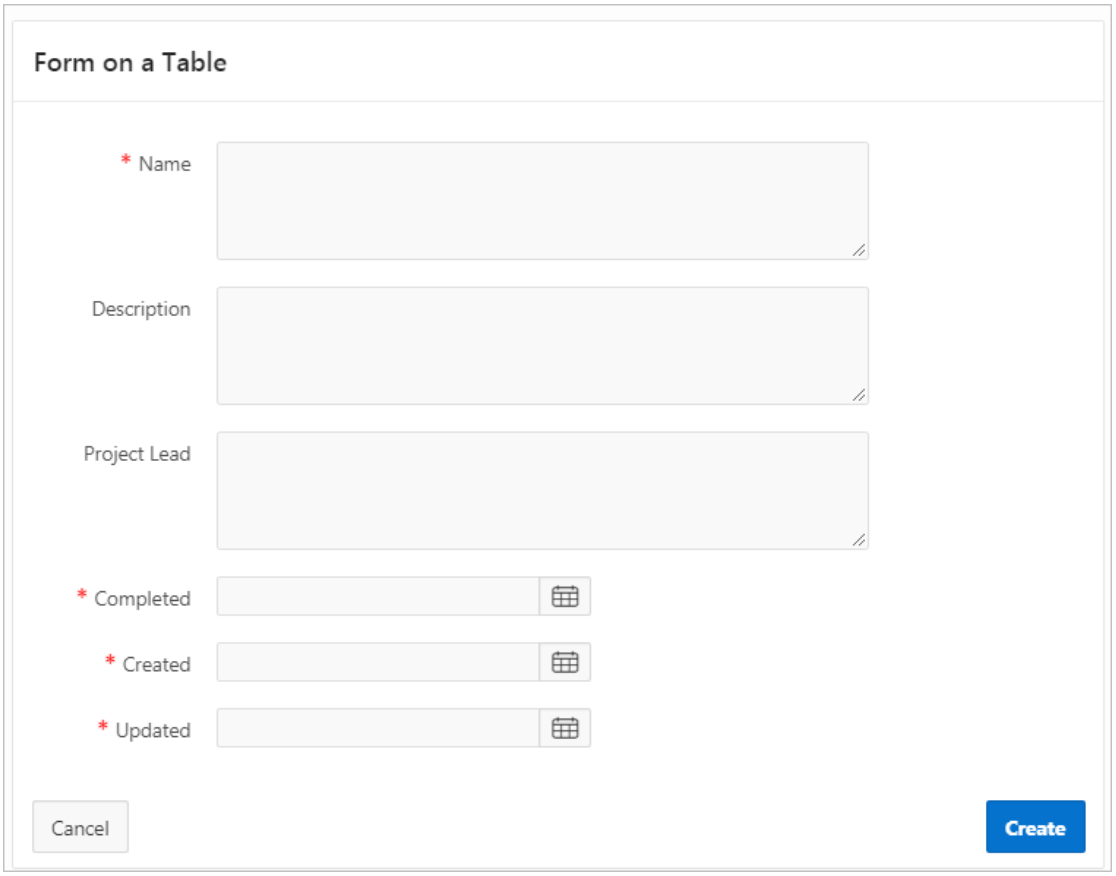


Рисунок 4.3 – Форма записи БД

И мастер создания приложений, и мастер создания страниц

поддерживают создание простой формы. Основное различие между этими двумя мастерами заключается в том, что мастер создания страниц предлагает больше параметров настройки и источников данных. С помощью мастера создания страницы разработчик может выбрать источник данных (то есть локальную базу данных, службу SQL с поддержкой REST или источник данных REST). Кроме того, разработчик также может указать, следует ли включать и настраивать панель навигации или меню навигации, а также выбирать столбцы и порядок, в котором они отображаются.

Отчет с формой

Отчет с формой создает отчет (то есть интерактивную сетку, интерактивный отчет или классический отчет) и форму в таблице. Разработчики выбирают тип отчета, чтобы определить, является ли отчет интерактивной сеткой, интерактивным отчетом или классическим отчетом. Пользователи щелкают значок редактирования, чтобы открыть форму.

И мастер создания приложения, и мастер создания страницы поддерживают создание комбинации отчета и формы. Основное различие между этими двумя мастерами заключается в том, что мастер создания страниц предлагает больше параметров настройки и источников данных. С помощью мастера создания страниц разработчик выбирает тип отчета (то есть интерактивную сетку, интерактивный отчет или классический отчет), а затем источник данных (то есть Локальная база данных, служба SQL с поддержкой REST или источник данных REST). Кроме того, разработчик также может указать, следует ли включать и настраивать панель навигации или меню навигации, а также выбирать столбцы и порядок, в котором они отображаются.

На рис.4.4 приведен пример интерактивного сеточного отчета с формой.

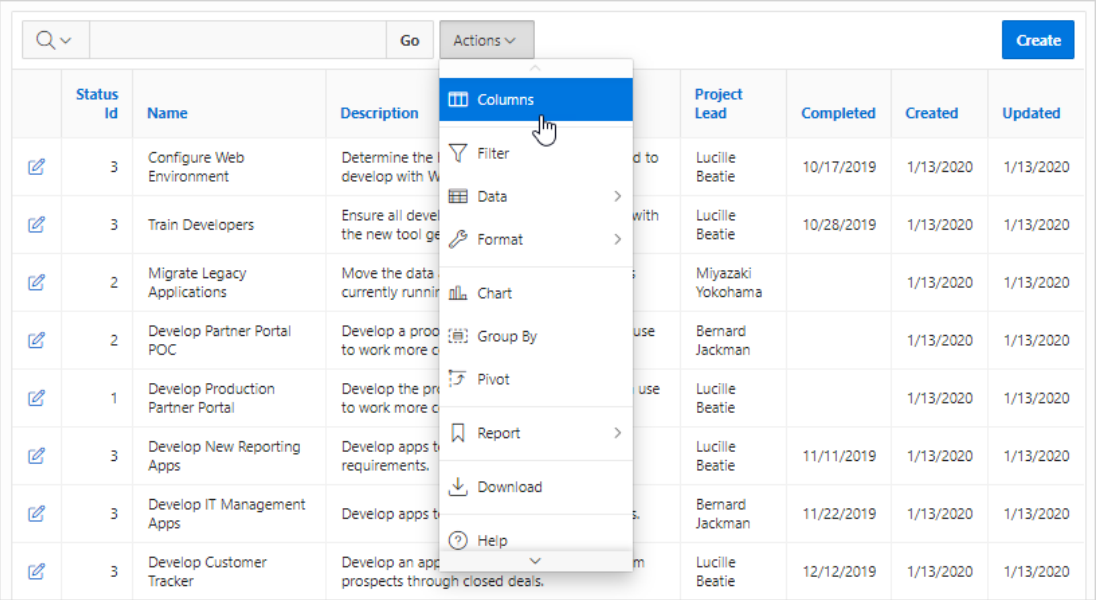


Рисунок 4.4 – Интерактивный сеточный отчет

Основные подробные формы

Форма основных сведений отражает отношение «один ко многим» между двумя таблицами в базе данных. Формы основных сведений позволяют пользователям вставлять, обновлять и удалять значения из двух таблиц или представлений. Как правило, основная форма сведений отображает основную строку и несколько строк сведений в одной HTML-форме. Разработчики могут создать одну или две страницы эталонной детали. Вы выбираете таблицы, на которых будет построена основная и детализированная области. Параметры формы Master Detail включают:

- **С накоплением** - создание единой главной детали с редактируемыми интерактивными сетками.
- **Параллельно** - создает главную деталь на одной странице (или рядом) с главной таблицей и таблицей деталей. Слева находится главный список для перехода к основной записи. Правая сторона содержит выбранную основную запись и связанный подробный отчет.
- **Drill Down** - создает двухстраничную (или детализированную) основную деталь. Первая страница содержит интерактивный отчет для

главной таблицы. На второй странице представлена стандартная форма для эталона и интерактивная сетка для деталей.

4.4 Настройка отчётов

В Oracle Application Express отчет – это форматированный результат SQL-запроса. Вы можете сгенерировать SQL-запрос, выбрав таблицу или представление в мастере или определив SQL-запрос вручную.

Запустите мастер создания приложения, чтобы создать новое приложение, содержащее одну или несколько страниц. Поддерживаемые страницы отчетов включают карточки, фасетный поиск, интерактивную сетку, интерактивный отчет и классический отчет.

Доступные отчеты при создании приложений

Мастер создания приложений поддерживает типы отчетов, приведенные в таблице 4.1.

Таблица 4.1 – Типы отчетов

Тип отчета	**Описание**
Открытки	Страница карточек состоит из отдельных ящиков, которые напоминают каталожные карточки, разложенные на странице. Каждая карточка отображает три части информации: заголовок карточки, столбец описания и дополнительный столбец текста. Сначала вы выбираете таблицу или представление, на основе которых будет построена страница. Во-вторых, вы выбираете заголовок карточки, столбец описания и столбец дополнительного текста.
Фасетный поиск	Создает сегментированную область поиска и отчет. Сначала вы выбираете тип отчета (**Отчет** или **Карты**). Во-вторых, вы выбираете таблицу или представление, на основе которых будет построена область фасетного поиска и отчет. При выборе **таблицы** фасеты автоматически обнаруживаются с помощью кэша словаря данных APEX. Если вы выбираете **View**, фасеты не обнаруживаются автоматически. Разработчики могут создать страницу фасетного отчета на основе представления, но область фасетов будет содержать только поле поиска, которое выполняет поиск по VARCHAR2 столбцам.
Интерактивная сетка	Интерактивная сетка представляет пользователям набор данных в настраиваемом отчете с возможностью поиска. Функционально интерактивная сетка включает в себя большинство возможностей

	настройки, доступных в интерактивных отчетах, а также возможность изменять порядок отчета в интерактивном режиме с помощью мыши. Сначала вы выбираете источник страницы (то есть **таблицу, представление** или **SQL-запрос**). Во-вторых, вы определяете, доступна ли интерактивная сетка для редактирования, выбрав «Разрешить **редактирование»** или «**Только чтение»**. Если вы выберете «**Разрешить редактирование»**, пользователи смогут добавлять, изменять и обновлять набор данных прямо на странице. Функционально интерактивная сетка включает в себя большинство возможностей настройки, доступных в интерактивных отчетах, а также возможность интерактивного переупорядочивания отчета с помощью мыши или клавиатуры. Вы выбираете стол для построения интерактивной сетки.
Интерактивный отчет	Создает страницу, содержащую форматированный результат SQL-запроса. Сначала вы выбираете источник страницы (то есть **таблицу, представление** или **SQL-запрос**). Во-вторых, вы выбираете тип отчета, **Интерактивный отчет**. Чтобы включить страницу формы для создания или обновления записей, выберите «**Включить форму»**. Выберите «**Включить форму»,** чтобы включить страницу формы для создания или обновления записей. Если отчет основан на таблице, которая имеет ограничения внешнего ключа для другой таблицы, разработчик также может определить столбцы поиска. Используйте **столбцы подстановки,** чтобы заменить идентификаторы отображаемым столбцом, например отображать название отдела вместо номера отдела.
Несколько отчетов	Создает несколько страниц отчета, содержащих интерактивный отчет с формой. Для каждой страницы вы выбираете таблицу, на которой будет построен интерактивный отчет с формой. Щелкните «**Изменить»,** чтобы изменить параметры по умолчанию (например, предоставить настраиваемый **запрос SQL**), указав другой тип отчета (например, выбрав **классический отчет** вместо **интерактивного отчета**). Выберите «**Включить форму»,** чтобы включить страницу формы для создания или обновления записей.
Классический отчет	Создает страницу, содержащую форматированный результат SQL-запроса. Сначала вы выбираете источник страницы (то есть **таблицу, представление** или **SQL-запрос**). Во-вторых, вы выбираете тип отчета **Classic Report**). Чтобы включить страницу формы для создания или обновления записей, выберите «**Включить форму»**. Если отчет основан на таблице, которая имеет ограничения внешнего ключа для другой таблицы, разработчик также может определить столбцы поиска. Используйте **столбцы подстановки,** чтобы заменить идентификаторы отображаемым столбцом, например отображать название отдела вместо номера отдела.

ЛАБОРАТОРНАЯ РАБОТА №4
СОЗДАНИЕ ПРОСТОГО ПРИЛОЖЕНИЯ В ORACLE APEX.

Цель работы

1. Научиться создавать простые приложения в среде разработки Oracle Application Express при помощи App Builder.

Знания, необходимые для выполнения лабораторной работы

1. Уметь пользоваться документацией по Oracle Application Express.
2. Знать ключевые концепции конструктора приложений App Builder.

Задание

1. Создать простое приложение на платформе Oracle APEX согласно своей предметной области, реализовать процедуры и функции в виде страниц или отчетов.

Контрольные вопросы

1. Что такое Oracle APEX?
2. Какие преимущества у Oracle APEX?
3. Что такое App Builder?
4. Назовите разделы App Builder, которые вы использовали для выполнения лабораторной работы.
5. В каких случаях применяется SQL Workshop?
6. Что такое Quick SQL?

ЗАКЛЮЧЕНИЕ

Предлагаемое учебно-методическое пособие содержит необходимый теоретический и практический материал, соответствующий государственному образовательному стандарту по направлению подготовки 09.03.01 «Информатика и вычислительная техника» и рабочей программе дисциплины «СУБД ORACLE»

Учебное издание имеет теоретическую и практическую направленность. В издании нашли отражение особенности проектирования и разработки баз данных Oracle, в том числе, и для разработки веб-приложений.

Тринадцать вариантов предметных областей предоставляют необходимый теоретический и практический материал для выполнения студентами лабораторных работ и самостоятельной работы студентов.

С методической точки зрения учебное издание полностью отвечает требованиям преподавания дисциплины «СУБД ORACLE».

С помощью этого пособия читатель, уже знакомый с какой-либо реляционной СУБД, сможет повысить свои теоретические знания по администрированию современных СУБД, а также научиться эффективно использовать Oracle для хранения, обработки и защиты информации любого объёма.

Автор надеется, что полученные с помощью данного пособия знания и навыки использования СУБД Oracle для решения задач в области информатики и вычислительной техники будут использованы студентами в будущей профессиональной деятельности.

СПИСОК ЛИТЕРАТУРЫ

1. Хоббс, Л. Oracle: Разработка и эксплуатация хранилищ баз данных / Лилиан Хоббс, Сьюзан Хилсон, Шилпа Лоуенд ; Вступ. слово Чак Розуот ; Пер. с англ. С. М. Лунин. - М. : Кудиц-образ, 2014. - 585 с.
2. Урман, С. Oracle: Программирование на языке PL/SQL / С. Урман ; Пер. О. Труфонов ; Науч. ред. А. Головко. - М. : ЛОРИ, 2016. - 528 с.
3. Томас Кайт, Кун Дарл. Oracle для профессионалов. Технологии и решения для достижения высокой производительности и эффективности. – М: Вильямс, 2015 – 960с.
4. Рик Гринвальд, Роберт Стаковьяк, Джонатан Стерн. Oracle 11g. Основы. – М: Символ-Плюс, 2009 – 464с.
5. Смирнов, Сергей Николаевич. Обработка документов средствами ORACLE : Практикум по XML и JOBC : Учеб. пособие по специальности в информ. безопасности / С. Н. Смирнов. - М. : Гелиос АРВ, 2014. - 187 с.
6. Заккар, Меградж. Разработка приложений для электронной коммерции на Oracle8i и Java на примерах : Учеб. пособие / Меградж Заккар ; Пер. с англ. В. А. Коваленко. - М. : Вильямс, 2000. - 336 с.
7. Maqsood Alam, Aalok Muley, Chaitanya Kadaru, Ashok Joshi. Oracle NoSQL Database: Real-Time Big Data Management for the Enterprise. - McGraw-Hill Osborne Media, 2019 – 256c.
8. Крёнке Д. Теория и практика построения баз данных. 9-е изд. – СПб.: Питер, 2005 – 859 с.
9. Пирогов В.Ю. Информационные системы и СУБД Oracle: организация и проектирование: учеб. пособие. – СПб: БХВ-Петербург, 2009. – 528 с.

ИНФОРМАЦИОННЫЕ РЕСУРСЫ

1. Ссылки на электронные материалы курса. URL: http://donnu.ru/phys/kt/bondarenko (дата обращения 10.11.2020 г.)
2. Курс «СУБД Oracle» в репозитории электронных курсов ДОННУ URL: http://dl.donnu.ru/course/view.php?id=90 (дата обращения 10.11.2020 г.)
3. Сайт корпорации Oracle. URL: http://www.oracle.com (дата обращения 10.11.2020 г.)
4. Самоучитель по синтаксису SQL и по его расширению PL/SQL. С примерами и описаниями. URL: http://www.firststeps.ru/sql/oracle/ (дата обращения 10.11.2020 г.)
5. Документация на русском языке, посвященная технологиям Oracle. URL: http://www.citforum.ru/database/oracle.shtml (дата обращения 10.11.2020 г.)
6. Документация по языку SQL. URL: http://www.sql.ru (дата обращения 10.11.2020 г.)
7. Oracle Application Express Documentation. URL: https://docs.oracle.com/en/database/oracle/application-express (дата обращения 14.12.2020 г.)
8. Туториал по Oracle Application Express. Обзор IDE. URL: https://habr.com/ru/post/445128/ (дата обращения 14.02.2021 г.)
9. Oracle APEX Tutorial. URL: https://www.youtube.com/playlist?list=PL_c9BZzLwBRJXk2hEP-U_OUMJAf_TYEiZ (дата обращения 14.12.2020 г.)

ПРИЛОЖЕНИЕ А. ВАРИАНТЫ ИНДИВИДУАЛЬНЫХ ЗАДАНИЙ

1. База данных хроники восхождений в альпинистском клубе.

В базе данных должны записываться даты начала и завершения каждого восхождения, имена и адреса участвовавших в нем альпинистов, название и высота горы, страна и район, где эта гора расположена. Дайте выразительные имена таблицам и полям, в которые могла бы заноситься указанная информация. Написать пакет, состоящий из процедур и функций, которые позволил бы выполнить следующие действия с базой данных:

1) для каждой горы показать список групп, осуществлявших восхождение, в хронологическом порядке;
2) предоставить возможность добавления новой вершины, с указанием названия вершины, высоты и страны местоположения;
3) предоставить возможность изменения данных о вершине, если на нее не было восхождения;
4) показать список альпинистов, осуществлявших восхождение в указанный интервал дат;
5) предоставить возможность добавления нового альпиниста в состав указанной группы;
6) показать информацию о количестве восхождений каждого альпиниста на каждую гору;
7) показать список восхождений (групп), которые осуществлялись в указанный пользователем период времени:
8) предоставить возможность добавления новой группы, указав ее название, вершину, время начала восхождения;
9) предоставить информацию о том, сколько альпинистов побывали на каждой горе.

Предусмотреть разработку триггеров, обеспечивающих каскадные изменения в связанных таблицах.

2. База данных медицинского кооператива

Базу данных использует для работы коллектив врачей. В таблицы должны быть занесены имя, пол, дата рождения и домашний адрес каждого пациента. Всякий раз, когда врач осматривает больного, явившегося к нему на прием, или сам приходит к нему на дом, он записывает дату и место, где проводится осмотр, симптомы, диагноз и предписания больному, проставляет имя пациента, а также свое имя. Если врач прописывает больному какое-либо лекарство, в таблицу заносится название лекарства, способ его приема, словесное описание предполагаемого действия и возможных побочных эффектов. Создать пакет, состоящий из функций и процедур, позволяющих:

1) по заданной дате определить количество вызовов в этот день;
2) определить количество больных, заболевших данной болезнью;
3) по заданному лекарству определить его побочный эффект;
4) предоставить возможность добавления нового лекарства с описанием его свойств з БД.

Предусмотреть разработку триггеров, обеспечивающих каскадные изменения в связанных таблицах.

3. База данных Городского совета.

В базе хранятся имена, адреса, домашние и служебные телефоны всех членов Совета. В Думе работает порядка сорока комиссий, все участники которых являются членами Совета. Каждая комиссия имеет свой профиль, например, вопросы образования, проблемы, связанные с жильем и так далее. Данные по каждой из комиссий включают: ее нынешний состав и председатель, прежние председатели и члены этой комиссии,

участвовавшие в ее работе за прошедшие 10 лет, даты включения и выхода из состава комиссии, избрания ее председателей. Члены Совета могут заседать в нескольких комиссиях. В базу заносятся время и место проведения каждого заседания комиссии с указанием депутатов и служащих Совета, которые участвуют в его организации. Создать пакет с процедурами и функциями, которые позволяют выполнять следующие действия:
1) показать список комиссий, для каждой ее состав и председателя;
2) предоставить возможность добавления нового члена комиссии;
3) показать список членов муниципалитета, для каждого из них список комиссий, в которых он участвовал и/или был председателем;
4) предоставить возможность добавления новой комиссии, с указанием председателя;
5) для указанного интервала дат и комиссии выдать список членов с указанием количества пропущенных заседаний;
6) предоставить возможность добавления нового заседания, с указанием присутствующих;
7) по каждой комиссии показать количество проведенных заседаний в указанный период времени.
Предусмотреть разработку триггеров, обеспечивающих каскадные изменения в связанных таблицах.

4. База данных рыболовной фирмы.

Фирме принадлежит небольшая флотилия рыболовных катеров. Каждый катер имеет "паспорт", куда занесены его название, тип, водоизмещение и дата постройки. Фирма регистрирует каждый выход на лов, записывая название катера, имена и адреса членов команды с указанием их должностей (капитан, боцман и т.д.), даты выхода и возвращения, а также вес пойманной рыбы отдельно по сортам (например, трески). За время одного рейса катер может посетить несколько банок. Фиксируется дата прихода на каждую банку и дата отплытия, качество выловленной рыбы (отличное, хорошее, плохое). На борту улов не взвешивается. Написать запросы, осуществляющие операции:

1) для каждого катера вывести даты выхода в море с указанием улова;
2) предоставить возможность добавления выхода катера в море с указанием команды;
3) для указанного интервала дат вывести для каждого сорта рыбы список катеров с наибольшим уловом;
4) для указанного интервала дат вывести список банок, с указанием среднего улова за этот период;
5) предоставить возможность добавления новой банки с указанием данных о ней;
6) для заданной банки вывести список катеров, которые получили улов выше среднего;
7) вывести список сортов рыбы и для каждого сорта список рейсов с указанием даты выхода и возвращения, количества
улова;
8) для выбранного пользователем рейса и банки добавить данные о сорте и количестве пойманной рыбы;
9) предоставить возможность пользователю изменять характеристики выбранного катера;
10) предоставить возможность добавления нового катера;
11) для указанного сорта рыбы и банки вывести список рейсов с указанием количества пойманной рыбы. Предусмотреть разработку триггеров, обеспечивающих каскадные изменения в связанных таблицах.

5. База данных фирмы, проводящей аукционы.

Фирма занимается продажей с аукциона антикварных изделий и произведений искусства. Владельцы вещей, выставляемых на проводимых фирмой аукционах, юридически являются продавцами. Лица, приобретающие эти вещи, именуются покупателями. Получив от продавцов партию предметов, фирма решает, на котором из аукционов выгоднее представить конкретный предмет. Перед проведением очередного аукциона каждой из выставляемых на нем вещей присваивается отдельный номер лота, играющий ту же роль, что и введенный ранее шифр товара. Две вещи, продаваемые на различных аукционах, могут иметь одинаковые номера лотов.

В книгах фирмы делается запись о каждом аукционе. Там отмечаются дата, место и время его проведения, а также специфика (например, выставляются картины, написанные маслом и не ранее 1900 г.). Заносятся также сведения о каждом продаваемом предмете: аукцион, на который он заявлен, номер лота, продавец, отправная цена и краткое словесное описание. Продавцу разрешается выставлять любое количество вещей, а покупатель имеет право приобретать любое количество вещей. Одно и то же лицо или фирма может выступать и как продавец, и как покупатель. После аукциона служащие фирмы, проводящей аукционы, записывают фактическую цену, уплаченную за проданный предмет, и фиксируют данные покупателя.

Создать пакет, состоящий из процедур и функций, позволяющий осуществить следующие операции:

1) для указанного интервала дат вывести список аукционов с указанием наименования, даты и места проведения;

2) добавить на указанный пользователем аукцион на продажу предмет искусства с указанием начальной цены;

3) вывести список аукционов, с указанием суммарного дохода от продажи, отсортированных по доходу;

4) для указанного интервала дат, вывести список предметов, которые были проданы на аукционах в этот период времени;

5) предоставить возможность добавления факта продажи на указанном аукционе заданного предмета;

6) для указанного интервала дат вывести список продавцов с указанием общей суммы, полученной от продажи предметов в этот промежуток времени;

7) вывести список покупателей, которые сделали приобретения в указанный интервал дат;

8) предоставить возможность добавления записи о проводимом аукционе (место, время);

9) для указанного места, вывести список аукционов;

10) для указанного интервала дат вывести список продавцов, которые принимали участие в аукционах, проводимых в этот
период времени;

11) предоставить возможность добавления и изменения информации о продавцах и покупателях;

12) вывести список покупателей с указанием количества приобретенных предметов в указанный период времени. Предусмотреть разработку триггеров, обеспечивающих каскадные изменения в связанных таблицах.

6. База данных библиотеки.

Разработать информационную систему обслуживания библиотеки, которая содержит следующую информацию: названия книг, ФИО авторов, наименования издательств, год издания, количество страниц, количество иллюстраций, стоимость,

название филиала библиотеки или книгохранилища, в которых находится книга, количество имеющихся в библиотеке экземпляров конкретной книги, количество студентов, которым выдавалась конкретная книга, названия факультетов, в учебном процессе которых используется указанная книга.

Необходимо составить пакет из процедур и функций, который позволяет:

1) для указанного филиала посчитать количество экземпляров указанной книги, находящихся в нем;

2) для указанной книги посчитать количество факультетов, на которых она используется в данном филиале, и вывести названия этих факультетов;

3) предоставить возможность добавления и изменения информации о книгах в библиотеке;

4) предоставить возможность добавления и изменения информации о филиалах;

5) предусмотреть разработку триггеров, срабатывающих на пользовательские исключительные ситуации; Предусмотреть разработку триггеров, обеспечивающих каскадные изменения в связанных таблицах.

7. База данных по учету успеваемости студентов.

База данных должна содержать данные о контингенте студентов (фамилия, имя, отчество, год поступления, форма обучения (дневная/вечерняя/заочная), номер или название группы); об учебном плане (название специальности, дисциплина, семестр, количество отводимых на дисциплину часов, форма отчетности (экзамен/зачет)); о журнале успеваемости студентов (год/семестр, студент, дисциплина, оценка).

Разработать пакет, состоящий из процедур и функций, позволяющий:

1) для указанной формы обучения посчитать количество студентов этой формы обучения;

2) для указанной дисциплины получить количество часов и формы отчетности по этой дисциплине;

3) предоставить возможность добавления и изменения информации о студентах, об учебных планах, о журнале успеваемости при этом предусмотреть курсоры, срабатывающие на некоторые пользовательские исключительния;

4) предоставить возможность добавления и изменения информации о журнале успеваемости.

Предусмотреть разработку триггеров, обеспечивающих каскадные изменения в связанных таблицах.

8. База данных для учета аудиторного фонда университета

База данных должна содержать следующую информацию об аудиторном фонде университета.

Наименование корпуса, в котором расположено помещение, номер аудитории, расположение аудитории в корпусе, ширина и длина аудитории в метрах, назначение и вид помещения, подразделение университетаэ за которым закреплено помещение. В базе данных также должка быть информация о высоте потолков в помещениях в зависимости от места расположения помещений в корпусе. Следует также учитывать, что структура подразделений университета имеет иерархический вид. когда одни подразделения входят в состав других (факультет, кафедра, лаборатория).

Помимо SQL запросов для создания таблиц базы данных, разработать пакет, состоящий из процедур и функций, позволяющий:

1) рассчитать данные о площадях и объемах каждого помещения;

2) для указанного корпуса получить количество факультетов, их названия и структуру, находящиеся в этом корпусе;

3) предоставить возможность добавления и изменения информации о корпусах в университете, при этом предусмотреть
курсоры, срабатывающие на некоторые пользовательские исключительные ситуации;

4) предоставить возможность добавления и изменения информации о комнатах в корпусах университета, при этом предусмотреть курсоры, срабатывающие на некоторые пользовательские исключительные ситуации.

Предусмотреть разработку триггеров, обеспечивающих каскадные изменения в связанных таблицах,

9. База данных для регистрации происшествий.

Необходимо создать базу данных для регистрации происшествий. База данных должна содержать данные для регистрации сообщений о происшествиях (регистрационный номер сообщения, дата регистрации, краткая фабула (тип происшествия); информацию о принятом по происшествию решении (отказано в возбуждении дел, удовлетворено ходатайство о возбуждении уголовного дела с указанием регистрационный номера заведенного дела, отправлено по территориальному признаку); информацию о лицах, виновных или подозреваемых в совершении происшествия (регистрационный номер лица, фамилия, имя, отчество, адрес, количество судимостей), отношение конкретных лиц к конкретным происшествиям (виновник, потерпевший, подозреваемый, свидетель...).

Помимо SQL запросов для создания таблиц базы данных, разработать пакет, состоящий из процедур и функций, позволяющий:

1) рассчитать данные о количестве происшествий в указанный промежуток времени;

2) для указанного лица получить количество происшествий, в которых он зарегистрирован;

3) предоставить возможность добавления и изменения информации о происшествиях, при этом предусмотреть курсоры,
срабатывающие на некоторые пользовательские исключительные ситуации;

4) предоставить возможность добавления и изменения информации о лицах, участвующих в происшествиях, при этом предусмотреть курсоры, срабатывающие на некоторые пользовательские исключительные ситуации.

Предусмотреть разработку триггеров, обеспечивающих каскадные изменения в связанных таблицах.

10. База данных для обслуживания работы конференции.

База данных должна содержать справочник персоналий участников конференции (фамилия, имя, отчество, ученая степень, ученое звание, научное направление, место работы, кафедра (отдел), должность, страна, город, почтовый индекс, адрес, рабочий телефон, домашний телефон, e-mail), и информацию, связанную с участием в конференции (докладчик или участник, дата рассылки 1-го приглашения, дата поступления заявки, тема доклада, отметка о поступлении тезисов, дата рассылки 2-го приглашения, дата поступления оргвзноса, размер поступившего оргвзноса, дата приезда, дата отъезда, потребность в гостинице).

Помимо SQL запросов для создания таблиц базы данных, разработать пакет, состоящий из процедур и функций, позволяющий:

1) для указанной даты 1-ой рассылки вывести список приглашенных и посчитать их количество;

2) предоставить возможность добавления приглашенных на конференцию с указанием оргвзноса и даты его уплаты;

3) вывести список приглашенных, с указанием даты об уплате оргвзноса;

4) для указанной интервала дат, вывести список участников, уплативших оргвзнос в этом диапазоне;

5) для указанного города вывести название тезисов докладов, поступивших из этого города;

6) для указанного города, вывести список нуждающихся в гостинице.

Предусмотреть разработку триггеров, обеспечивающих каскадные изменения в связанных таблицах.

11. База данных для обслуживания склада.

База данных должна обеспечить автоматизацию складского учета, В ней должны содержаться следующие данные:

1) информация о "единицах хранения" — номер ордера, дата, код поставщика, балансный счет, код сопроводительного документа по справочнику документов, номер сопроводительного документа, код материала по справочнику материалов, счет материала, код единицы измерения, количество пришедшего материала, цена единицы измерения;

2) информация о хранящихся на складе материалах (справочник материалов — код класса материала, код группы материала, наименование материала);

3) информация о единицах измерения конкретных видов материалов — код материала, единица измерения (метры, килограммы, литры и т.д.);

4) информация о поставщиках материалов — код поставщика, его наименование, ИНН, юридический адрес (индекс, город, улица, дом), адрес банка (индекс, город, улица, дом), номер банковского счета.

Помимо SQL запросов для создания таблиц базы данных, разработать пакет, состоящий из процедур и функций, позволяющий:

1) посчитать количество поставщиков данного материала;

2) предоставить возможность добавления единицы хранения с указанием всех реквизитов;

3) вывести список поставщиков с указанием всех реквизитов данного материала на склад;

4) для указанного адреса банка посчитать количество поставщиков склада, пользующихся услугами этого банка. Предусмотреть разработку триггеров, обеспечивающих каскадные изменения в связанных таблицах.

12. База данных фирмы.

Фирма отказалась от приобретения некоторых товаров у своих поставщиков, решив самостоятельно наладить их производство. С этой целью она организовала сеть специализированных цехов, каждый из которых принимает определенное участие в технологическом процессе. Каждому виду выпускаемой продукции присваивается, как обычно, свой шифр товара, под которым он значится в файле товарных запасов. Этот же номер служит и шифром продукта. В записи с этим шифром указывается, когда была изготовлена последняя партия этого продукта, какова ее стоимость, сколько операций потребовалось.

Операцией считается законченная часть процесса производства, которая целиком выполняется силами одного цеха в соответствии с техническими требованиями, перечисленными на отдельном чертеже. Для каждого продукта и для каждой операции в базе данных фирмы заведена запись, содержащая описание операции, ее среднюю продолжительность и номер чертежа, по которому можно отыскать требуемый чертеж. Кроме того, указывается номер цеха, обычно производящего данную операцию. В

запись, связанную с конкретной операцией, заносятся потребные количества расходных материалов, а также присвоенные им шифры товара. Расходными называют такие материалы, как, например, электрический кабель, который нельзя использовать повторно. Когда, готовясь к выполнению операции расходный материал забирают со склада, регистрируется фактически выданное количество, соответствующий шифр товара, номер служащего, ответственного за выдачу, дата и время выдачи, номер операции и номер наряда на проведение работ, который будет обсуждаться ниже. Реально ззтраченное количество материала может не совпадать с расчетным, из-за того, например, что часть изготовленной продукции бракуется.

Каждый из цехов располагает многочисленными инструментами и приспособлениями. При выполнении некоторых операций их все же не хватает, и цех вынужден обращаться в центральную инструментальную за недостающими. Каждый тип инструмента снабжен отдельным номером и на него заведена запись со словесным описанием. Кроме того, там отмечено, какое количество инструментов этого типа выделено цехам и какое осталось в инструментальной. Экземпляры инструмента конкретного типа, например гаечные ключи одного размера, различаются по своим индивидуальным номерам. На фирме для каждого типа инструмента имеется запись, содержащая перечень всех индивидуальных номеров. Кроме того, указаны даты их поступления на склад. По каждой операции в фирме отмечают типы и количества инструментов этих типов, которые должны использоваться при ее выполнении. Когда инструменты действительно берутся со склада фиксируется индивидуальный номер каждого экземпляра, указываются номер заказавшего их цеха и номер наряда на проведение работ. И в этом случае затребованное количество не всегда совпадает с заказанным.

Наряд на проведение работ по форме напоминает заказ на приобретение товаров, но, в отличие от последнего, он направляется не поставщику, а в один из цехов. Оформляется этот наряд после того, как руководство фирмы сочтет необходимым выпустить партию некоторого продукта. В наряд заносятся шифр продукта, дата оформления наряда, срок, к которому должен быть выполнен заказ, а также требуемое количество продукта.

Разработайте структуру таблиц базы данных, подберите имена таблиц и полей, в которых могла бы разместиться вся эта информация.

Помимо SQL запросов для создания таблиц базы данных, разработать пакет, состоящий из процедур и функций, позволяющий:для выбранного цеха выдать список операций, выполняемых им;

1) для каждой операции — список расходных материалов, с указанием количества;

2) показать список инструментов и предоставить возможность добавления нового;

3) выдать список используемых инструментов;

4) для указанного интервала дат, вывести список нарядов;

5) показать список операций и предоставить возможность добавления новой операции;

6) выдать список расходных материалов, используемых в различных нарядах;

7) выдать список товаров, с указанием используемых инструментов;

8) показать список нарядов и предоставить возможность добавления нового;

9) выдать отчет о производстве товаров различными цехами, указав наименование цеха, название товара и его количество.

Предусмотреть разработку триггеров, обеспечивающих каскадные изменения в связанных таблицах.

13. База данных музыкального магазина.

Таблицы базы данных содержат информацию о музыкантах, музыкальных произведениях и обстоятельствах их исполнения. Нескольких музыкантов, образующих единый коллектив, называются ансамблем. Это может быть классический оркестр, джазовая группа, квартет, квинтет и т.д. К музыкантам причисляют исполнителей (играющих на одном или нескольких инструментах), композиторов, дирижеров и руководителей ансамблей.

Кроме того, в базе данных хранится информация о дисках, которыми магазин торгует. Каждый диск, а точнее, его лэйбл, идентифицируется отдельным номером, так что всем копиям, отпечатанным с матрицы в разное время, присвоены одинаковые номера. На диске может быть записано несколько исполнений одного и того же произведения — для каждого из них в базе заведена отдельная запись. Когда выходит новый диск, регистрируется название выпустившей его компании (например, EMI), а также адрес оптовой фирмы, у которой магазин может приобрести этот диск. Не исключено, что компания-производитель занимается и оптовой продажей своих дисков. Магазин фиксирует текущие оптовые и розничные цены на каждый диск, дату его выпуска, количество экземпляров, проданных за прошлый год и в нынешнем году, а также число еще не распроданных дискок.

Помимо SQL запросов для создания таблиц базы данных, разработать пакет, состоящий из процедур и функций, позволяющий получить:

1) количество музыкальных произведений заданного ансамбля;
2) список названий всех дисков заданного ансамбля;
3) список лидеров продаж текущего года, то есть названия дисков, которые чаще всего покупали в текущем году
4) предусмотреть изменения данных о дисках и ввод новых данных;
5) предусмотреть ввод новых данных об ансамблях.

Предусмотреть разработку триггеров, обеспечивающих каскадные изменения в связанных таблицах.