Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра информационных систем и технологий

Специальность 1-40 05 01 «Информационные системы и технологии»

Специализация 1-40 05 01 03«Информационные системы и технологии» (издательско-полиграфический комплекс)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА**

по дисциплине Администрирование баз данных и приложений»

Тема: Бумажник Oracle Wallet: обеспечение безопасности для базы данных «Гостиница»

Исполнитель

студент 3 курса группы 2 \_\_\_\_\_\_\_\_\_\_ Т.А. Шихова

подпись, дата

Руководитель

Ассистент \_\_\_\_\_\_\_\_\_\_ О.Ю. Азаркевич

подпись, дата

Курсовая работа защищена с оценкой

Руководитель О.Ю. Азаркевич

подпись

Минск 2017

**Оглавление**

[Введение 3](#_Toc482647670)

[1. Алгоритмы решения 4](#_Toc482647671)

[1.1 Oracle Database 12c 4](#_Toc482647672)

[1.2 SQL Developer 5](#_Toc482647673)

[1.3 Бумажник Oracle Wallet 7](#_Toc482647674)

[2. Разработка архитектуры проекта 9](#_Toc482647675)

[2.1 Разработка модели базы данных 9](#_Toc482647676)

[2.2 Разработка необходимых объектов 11](#_Toc482647677)

[2.3 Описание использования технологии 15](#_Toc482647678)

[3. Руководство пользователя 19](#_Toc482647679)

[Заключение 21](#_Toc482647680)

[Список используемых источников 22](#_Toc482647681)

[Приложение А Создание таблиц, пользователей и назначение им привилегий 23](#_Toc482647682)

# **Введение**

Базы данных – это сложные многофункциональные программные системы, работающие в открытой распределенной среде. Они уже сегодня доступны для использования в деловой сфере и выступают не просто в качестве технических и научных решений, но как завершенные продукты, предоставляющие разработчикам мощные средства управления данными и богатый инструментарий для создания прикладных программ и систем.

В современной технологии баз данных предполагается, что создание базы данных, ее поддержка и обеспечение доступа пользователей к ней осуществляется централизованно с помощью специального программного инструментария – системы управления базами данных (СУБД).

Система управления базами данных (СУБД) – это совокупность языковых и программных средств, предназначенных для создания, ведения и совместного использования БД многими пользователями.

Современная СУБД содержит в своем составе программные средства создания баз данных, средства работы с данными и сервисные средства. С помощью средств создания БД проектировщик, используя язык описания данных (ЯОД), переводит логическую модель БД в физическую структуру, а на языке манипуляции данными (ЯМД) разрабатывает программы, реализующие основные операции с данными (в реляционных БД – это реляционные операции).

Цель курсового проекта: изучение и использование технологии Oracle Wallet в базе данных «Hotel».

# **Алгоритмы решения**

## **Oracle Database 12c**

В 2013 г. компания Oracle выпустила версию Oracle Database 12c (версия 12.1.0.1), основными достоинствами которой стали снижение стоимости хранения, высокая доступность данных, простота консолидации баз данных и защита доступа к данным.

Говоря чуть подробнее, в этой версии появилась архитектура Oracle Multitenant, которая существенно облегчает консолидацию баз данных, ускоряет развертывание баз данных и позволяет управлять многими базами данных как одним целым — вместо администрирования сотен баз данных по отдельности администратор работает с одной базой данных, управляя многими базами данных, как одной. Все это сделало версию Oracle Database 12c на момент ее выпуска самой подходящей системой управления базами данных для облачных вычислений, особенно для SaaS-приложений, где особенно актуально скоростное создание новых баз данных по требованию пользователей, которое при поддержке технологии Snapshot Cloning (тонкое клонирование) занимает несколько минут.

Кроме того, в Oracle Database 12.1.0.1 появилась автоматическая оптимизация данных, сочетающая технологию «умного сжатия», которая автоматически выявляет блоки данных, к которым редко обращались («холодные» данные), и сжимает их, и технологию автоматизации многоуровневого хранения данных, которая автоматически переносит «холодные» данные на более дешевый уровень хранения.

Еще одна новая технология Oracle Database 12c, которая называется Data Guard Far Sync, обеспечивает нулевую потерю данных на больших расстояниях и позволяет держать резервные копии баз данных на большом удалении от основной базы данных. Дополнительный специальный экземпляр базы данных, не имеющий файлов данных, принимает изменения от основной базы данных в синхронном режиме и асинхронно передает эти изменения удаленным экземплярам базы данных, что обеспечивает и надежность синхронного режима, и производительность асинхронного режима.

Технология Application Continuity позволяет повторять аварийно прерванные транзакции — решая тем самым одну из главных проблем работы веб-приложений с базами данных. Технология делает отказ экземпляра базы данных прозрачным для веб-приложения и позволяет определить состояние последней транзакции. Если транзакция не прошла, она будет выполнена, а если она уже выполнена, то технология Application Continuity не позволяет выполнить ее повторно

Технология динамического маскирования данных Data Redaction прозрачная для приложений и позволяет задавать политики доступа к данным внутри базы данных. Данные остаются неизменными, но, в зависимости от прав конечного пользователя, его роли, он будет видеть только те данные, на доступ к которым он авторизован. Это позволяет приложениям прозрачно работать с базой данных, политика будет выполняться для всех приложений.

Наконец, в Oracle Database 12.1.0.1 была реализована мощная система анализа взаимосвязи строк Pattern Matching, которая позволяет анализировать тренды и находить в них статистические закономерности с помощью конструкций языка SQL. И это — не считая еще более пятисот других модификаций [2].

Oracle Database 12c доступна в трех выпусках и предлагает большой набор функций корпоративного класса для удовлетворения конкретных требований заказчиков в области производительности и доступности, безопасности и соответствия требованиям, хранения данных и аналитики, а также обработки неструктурированных данных:

* Oracle Database 12*c* Personal Edition поддерживает однопользовательскую разработку и развертывание приложений, полностью совместимых с редакциями Oracle Database Standard Edition One, Oracle Database Standard Edition и Oracle Database Enterprise Edition;
* Oracle Database 12c Standard Edition One (SE1) База данных для рабочей группы, подразделения, и веб-приложений на одном сервере с максимальной емкостью в два разъема;
* Oracle Database 12c Standard Edition (SE) Доступный, полнофункциональный базы данных для серверов с четырьмя разъёмами. Также включает в себя Oracle Real Application Clusters, повышающий работоспособность и обеспечивающий поддержку для одного контейнера базы данных для начального уровня облачных вычислений и консолидации;
* Oracle Database 12c Enterprise Edition (EE) Последнее поколение в мире база данных доступно на выбор платформ. Включает в себя архитектуру, которая упрощает процесс консолидации баз данных на облаке, что позволяет клиентам управлять многими базами данных, не меняя своих приложениях [3].

## **SQL Developer**

Oracle SQL Developer – бесплатный инструмент для написания SQL-запросов, разработки PL/SQL пакетов, процедур, функций, триггеров и т. п. Этот инструмент написан на языке Java и является кроссплатформенным, т. е. работает во всех операционных системах. Oracle SQL Developer интегрируется с Apex для разработки и администрирования приложений. Oracle SQL Developer позволяет выполнять экспорт и импорт данных и структур.

Фактически, Oracle SQL Developer – это намного больше, чем просто SQL-редактор, такой, например, как PL/SQL Developer, Toad и им подобные. Это мощный инструмент для разработки на PL/SQL с контролем версий, инструмент миграции с других баз данных и администрирования связи между различными СУБД, разработки отчетов и публикации в Oracle Apex и многое другое.

 Отметим его самые важные отличия:

* Oracle SQL Developer — полностью бесплатное приложение. Несмотря на это, по своим возможностям и удобству оно не уступает платным средствам (PL/SQL Developer, Toad, SQL Navigator);
* Oracle SQL Developer написан на языке Java и может использоваться (с одним и тем же графическим интерфейсом) как под Windows, так и под другими операционными системами, например, Linux;
* Oracle SQL Developer не требует установки на компьютер. Все пользовательские настройки в нем хранятся в файлах XML;
* Oracle SQL Developer может использоваться для создания запросов к другим системам управления базами данных, например, Microsoft SQL Server, Microsoft Access, MySQL и TimesTen. На многих предприятиях возможность использования единообразного программного средства для создания запросов к разным источникам может быть очень удобной. Правда, отметим, что Oracle SQL Developer знает про базы данных Microsoft далеко не все и лучше использовать его при создании простых запросов к базам данных, отличных от Oracle.

К недостаткам SQL Developer можно отнести несколько непривычный интерфейс (основанный на графических компонентах Java) и большую ресурсоемкость (особенно с точки зрения оперативной памяти). Кроме того, в Windows все настройки SQL Developer (включая, например, определения пользовательских отчетов и сниппетов) хранятся не в реестре, а в файлах XML в профиле пользователя, что также необходимо учитывать.

Наиболее важные возможности Oracle SQL Developer:

В Oracle SQL Developer реализованы возможности прямого редактирования записей в таблицах баз данных. Для этого нужно в контекстном меню для таблицы в окне Connections выбрать команду Edit, а затем перейти на вкладку Data.

В Oracle SQL Developer предусмотрены средства для отладки кода программных модулей PL/SQL (включая точки останова, пошаговое выполнение, просмотр значений переменных и т.п.). Эти средства в основном доступны из меню Debug.

Заранее заготовленные блоки программного кода доступны из окна Snippets (это окно можно открыть при помощи меню View). Такие блоки заготовлены как для встроенных функций Oracle SQL, так и для других синтаксических конструкций (операторов условного перехода, циклов, курсоров и т.п.). Набор сниппетов является расширяемым: вы вполне можете добавлять в него свои блоки программного кода.

Для того, чтобы было проще переносить программный код, написанный в расчете на выполнение в SQL Plus, в Oracle SQL Developer предусмотрена поддержка многих синтаксических конструкций SQL Plus (например, обращение к внешним файлам скриптов при помощи символов @ и @@). Однако поддерживаются далеко не все специальные команды SQL Plus.

Если объектов в базе данных много (обычная ситуация для реальных баз данных), и просматривать их все в окне Connection трудно, можно воспользоваться специальным окном поиска объектов. Для этого нужно в меню View воспользоваться командой Find DB Object, затем в открывшемся окне выбрать соединение, ввести имя объекта (можно использовать подстановочные символы) и нажать на Enter. Для объектов в списке результатов из контекстного меню доступны те же возможности, что и для объектов в окне Connections.

В Oracle SQL Developer предусмотрены средства переноса информации между различными источниками данных, например, из базы данных SQL Server в Oracle. Средства для выполнения такого переноса сосредоточены в меню Migration.

## **Бумажник Oracle Wallet**

БД Oracle не является замкнутой системой. СУБД вступает в контакт с участниками компьютерной сети, а данные базы, равно как и резервные копии, технически хранятся на внешних носителях. Хотя СУБД Oracle имеет собственную систему защиты данных, внешнее окружение, с которым она взаимодействует, вовсе не подконтрольно ей. Например, канал связи прикладной программы с СУБД может испытать постороннее вмешательство; существует риск постороннего же обращения к содержимому файлов с данными в обход Oracle. [4]

Проблема не связана исключительно с Oracle и носит общий характер для информационных систем как таковых. Стандартное решение состоит в использовании шифрования передаваемых данных и подсчете контрольной свертки (суммы). Более того, передача шифра (необходимого для расшифровки принимающей стороной) часто вдобавок вынуждено сопровождается ссылкой на «свидетельство подлинности», так называемый сертификат. Когда взаимодействие субъектов и объектов доступа в информационной системе идет активно, предъявлять в программе шифр или его свидетельство подлинности многократно и «вручную» и неудобно, и небезопасно. Гораздо легче запомнить эти сведения где-то один раз и «попросить» систему делать проверку по мере надобности самостоятельно.

Популярным приемом является использовать для такого локализованного расположения данных параметров защищенного доступа «электронный бумажник» (ewallet). По сути это файл на компьютере (сервере или клиенте), где информация о параметрах доступа сама в свою очередь защищена.

Фирма Oracle предполагает использовать в качестве электронного бумажника собственное решение, Oracle Wallet, существующее в рамках расширения Advanced Security для Oracle Enterprise Edition или для клиента. Oracle Wallet позволяет хранить и обеспечивать использование системой следующие основные категории сведений:

* «главный ключ» (masterkey) - шифр для автоматического шифрования данных на диске: в требуемых столбцах таблиц или целиком в табличных пространствах (transparent data encryption, TDE), а также для шифрования результатов сохранного (резервного) копирования;
* сертификаты подлинности ключей шифрования, используемых при передаче данных по защищенным каналам и сами ключи;
* имена пользователя и пароли ради упрощенного («беспарольного») соединения программы с сервером по данным из бумажника клиента.

Бумажник Oracle Wallet представляет собой двоичный файл, составленный по промышленному стандарту PKCS #12 «синтаксиса обмена персональной информацией», что обеспечивает ему совместимость с ПО третьих фирм. Файл имеет имя ewallet.p12, и он должен располагаться в любом доступном СУБД каталоге файловой системы. Тем не менее, когда бумажник используется для хранения главного ключа шифрования, имеется местонахождение по умолчанию, для Windows это %ORACLE\_BASE%\admin\%ORACLE\_SID%\wallet. Оно разумно привязано к месту хранения рабочих файлов СУБД, так как несколько СУБД на одном компьютере не имеют права пользоваться общим бумажником.

Сам бумажник Oracle появился в версии 8.1, однако приводимое далее его применение оказалось возможным только в версии 10.2. Для работы со своим электронным бумажником фирма Oracle дает следующие средства:

* команды SQL;
* программу Oracle Wallet Manager (она же *owm);*
* программу *orapki;*
* программу *mkstore.*

# **Разработка архитектуры проекта**

## **Разработка модели базы данных**

Перед тем как приступить к созданию БД необходимо разработать схему данных. Схема данных – это структура базы, а, именно, структура или структуры основных таблиц. При создании таблиц нужно учесть то, какие будут использоваться связи между ними. Тип создаваемой связи зависит от полей, для которых определяется связь:

* связь один - ко - многим создается в том случае, когда только одно из полей является ключевым или имеет уникальный индекс, т.е. значения в нем не повторяются;
* связь один - к - одному создается в том случае, когда оба связываемых поля являются ключевыми или имеют уникальные индексы;
* связь многие - ко - многим фактически представляет две связи типа один - ко - многим через третью таблицу, ключ которой состоит, по крайней мере, из двух полей, общих для двух других таблиц.

Основными этапами проектирования базы данных информационной системы являются:

* концептуальное проектирование – сбор, анализ и редактирование требований к данным;
* логическое проектирование – преобразование требований к данным в структуры данных. На этом этапе проектируем непосредственно базу данных;
* физическое проектирование – определение особенностей хранения данных, методов доступа и т.д.

На первом этапе проектирования базы данных необходимо определить цель создания базы данных, основные ее функции и информацию, которую она должна содержать. То есть нужно определить основные темы таблиц базы данных и информацию, которую будут содержать поля таблицы.

В данной курсовой работе я использовала СУБД Oracle 12c для работы с моей базой данных. Таблицы, созданные в базе данных “Гостиница” представлены на рисунке 2.1.

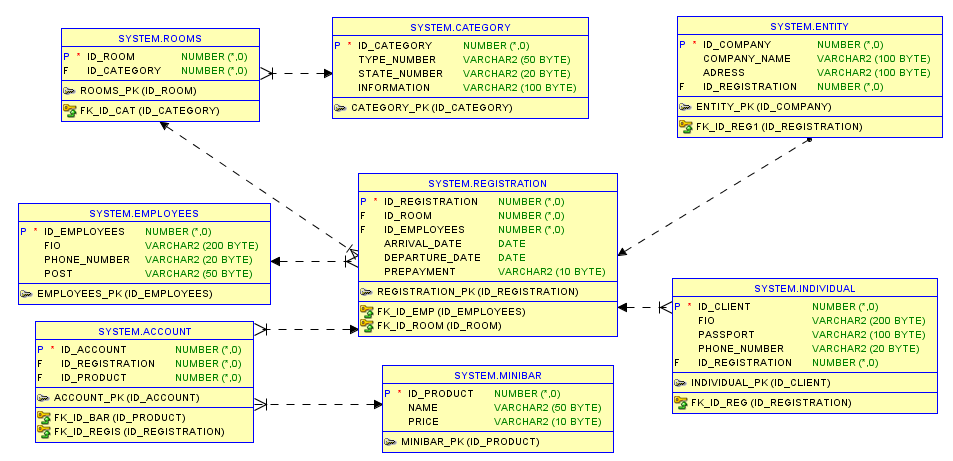


Рисунок 2.1 ­– Схема базы данных

База данных состоит из следующих таблиц:

Таблица CATEGORY, содержит информацию о всех номерах гостиницы:

* id\_category – номер комнаты (primary key);
* type\_number – тип номера;
* state\_number – состояние номера;
* information – дополнительная информация.

Таблица ROOMS содержит информацию о том какой номер за кем зарегистрирован:

* id\_room – номер зарегистрированной комнаты (primary key);
* id\_category – номер комнаты (foreign key связанный с таблицей CATEGORY и колонкой id\_category).

Таблица REGISTRATION содержит информацию о регистрации клиента:

* id\_registration – код регистрации (primary key);
* id\_room – номер зарегистрированной комнаты (foreign key связанный с таблицей ROOMS и колонкой id\_ room);
* id\_employee – код сотрудника, зарегистрировавший посетителя (foreign key связанный с таблицей EMPLOYEES и колонкой id\_ employee);
* arrival\_date – дата заезда;
* departure\_date – дата выезда;
* prepayment – предоплата.

Таблица EMPLOYEES содержит информацию о сотруднике, который зарегистрировал клиента:

* id\_employee – код сотрудника, зарегистрировавший посетителя (primary key);
* fio – фамилия, имя, отчество сотрудника;
* phone\_number – номер телефона;
* post – занимаемая должность.

Таблица ACCOUNT представляет собой счет посетителя:

* id\_account – номер счета (primary key);
* id\_product – код продукта из мини-бара (foreign key связанный с таблицей MINIBAR и колонкой id\_ product);
* id\_registration – код регистрации (foreign key связанный с таблицей REGISTRATION и колонкой id\_ registration).

Таблица MINIBAR содержит информацию о продуктах в мини-баре:

* id\_ product – код продукта (primary key);
* name – наименование;
* price – цена.

Таблица INDIVIDUAL содержит информацию о физическом лице:

* id\_ client – код клиента (primary key);
* fio – фамилия, имя отчество;
* passport – номер и серия паспорта;
* phone\_number – номер телефона;
* id\_registration – код регистрации (foreign key связанный с таблицей REGISTRATION и колонкой id\_ registration).

Таблица ENTITY содержит информацию о юридическом лице:

* id\_ company – код компании (primary key);
* name – наименование компании;
* adress – адрес;
* id\_registration – код регистрации (foreign key связанный с таблицей REGISTRATION и колонкой id\_ registration).

## **Разработка необходимых объектов**

Неотъемлемой частью любой базы данных является создание необходимых объектов, таких как:

* создание ролей и назначение им системных привилегий. Для администратора назначены все привилегии, а для обычного пользователя только возможность посматривать таблицы и объекты (рисунок 2.2);

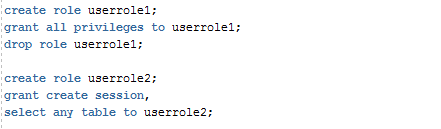


Рисунок 2.2 – Создание ролей.

* создание профилей безопасности пользователей, которые отличаются тем, сколько действителен пароль, администратор должен чаще его менять, и продолжительностью сессии (рисунок 2.3);



Рисунок 2.3 – Создание профилей

* создание пользователей и назначение им ролей и профилей (рисунок 2.4);

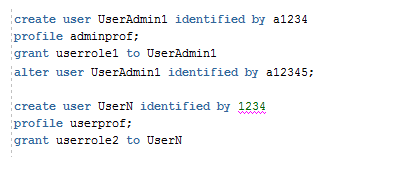


Рисунок 2.4 – Создание пользователей

* создание объектов базы данных (таблиц, индексов, представлений, функций, процедур, пакетов, и т.п.);

Было создано несколько представлений. На рисунке 2.5 представлены только два.

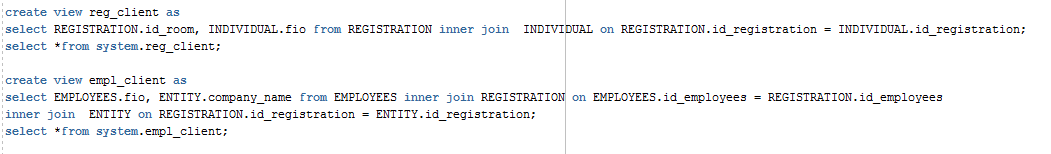


Рисунок 2.5 – Создание представление

Доступ к данным осуществляется через соответствующие процедуры, созданные для каждой таблицы. На рисунках 2.5 – 2.9 представлены процедуры для таблицы MINIBAR.

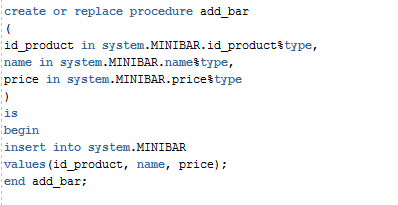


Рисунок 2.5 – Процедура добавления

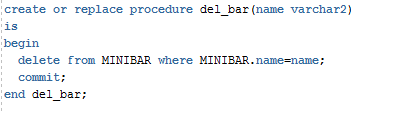


Рисунок 2.6 – Процедура удаления

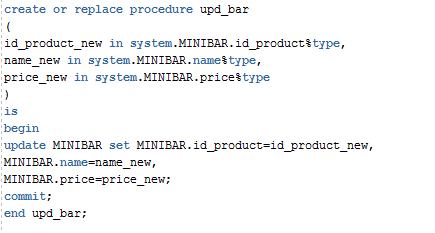


Рисунок 2.7 – Процедура обновления

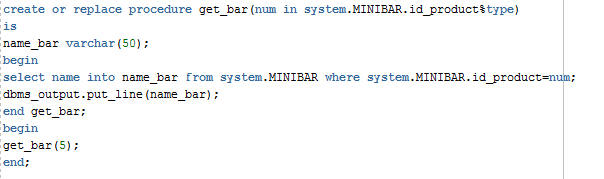


Рисунок 2.8 – Процедура вывода на экран одного значения из таблицы

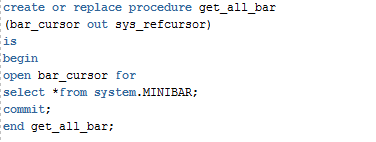


Рисунок 2.9 – Процедура вывода всех значений из таблицы

Также были созданы пакеты для каждой таблицы, объединяющие все процедуры (рисунок 2.10).

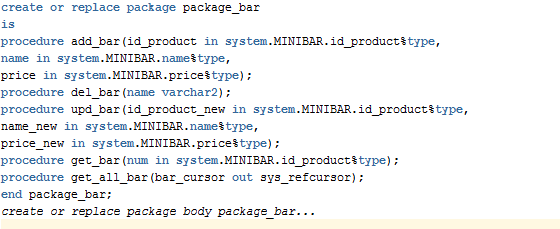


Рисунок 2.10 – Пакет для работы с таблицей MINIBAR

* назначение пользователям объектных привилегий.

Так как пользователь UserAdmin1 может работать с таблицами и соответствующими объектами, то ему были назначены привилегии для работы с пакетами (рисунок 2.11).

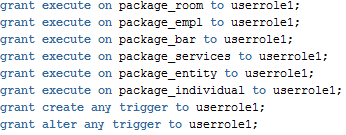


Рисунок 2.11 – Назначение привилегий администратору для работы с пакетами

## **Описание использования технологии**

Создать бумажник с главным ключом шифрования можно неявно командой в ALTER SYSTEM SET ENCRYPTION… или явно программами Oracle Wallet Manager и mkstore. Обратим

внимание, что главный ключ будет применяться СУБД для шифрования данных при размещении на диске и расшифровке при взятии с диска, поэтому дальнейшая работа предполагается на сервере, с серверным ПО.

К сожалению, для электронного бумажника в Oracle ПО написано недостаточно тщательно, так что несоблюдение описанного далее порядка действий способно привести вплоть до ошибки ORA-00600 и необходимости перезагрузки СУБД.

Проще завести бумажник неявно, командой SQL. Необходимо выдать команду SQL от имени SYS, в SQLPLUS. Мы можем убедиться, что бумажника пока не существует (рисунок 2.12)

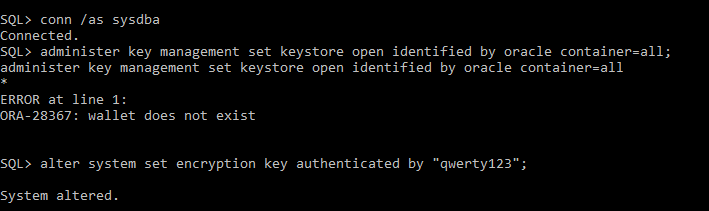


Рисунок 2.12 – Подключение к инстансу и проверяем наличие бумажника

Для начала нам нужно создать каталог wallet в директории D:\app\new\_user\admin\orcl\ в ней по умолчанию будет находиться «кошелёк». Далее в SQLPLUS создаем кошелек с паролем к нему. Приведенный пароль будет служить для последующего открытия бумажника и вообще доступа к его содержимому. Двойные кавычки неслучайны, так как в пароле различаются большие и малые буквы. Для смены пароля в будущем надо будет использовать программу Oracle Wallet Manager, обратиться к которой в Windows можно через меню запуска программ или из командной строки ОС, а в Unix - набором owm в командной строке ОС.

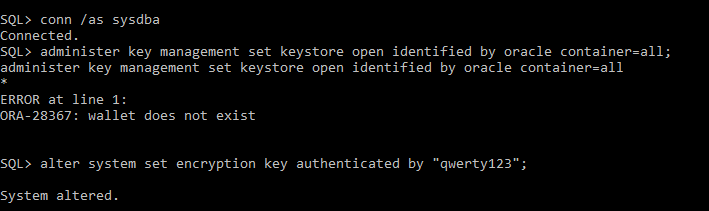


Рисунок 2.13 – Создание кошелька с паролем qwerty123

По указанной команде SQL в созданном каталоге появится файл бумажника с главным ключом доступа. К фактическому использованию главного ключа из бумажника мы еще не готовы, так как бумажник нужно открыть. Команды открытия и закрытия бумажника представлены на рисунке 2.14.

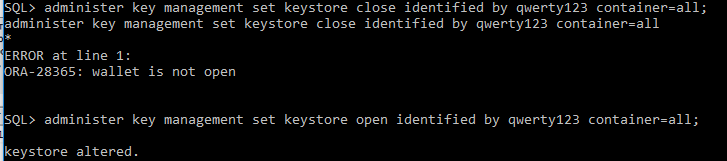


Рисунок 2.14 – Команды открытия и закрытия кошелька

Также мы можем создать бумажник явно с помощью Oracle Wallet Manager. Для этого просто нажимаем File->New. Далее необходимо указать пароль, он должен состоять как минимум из 8 символов, и тип бумажника (рисунок 2.15).

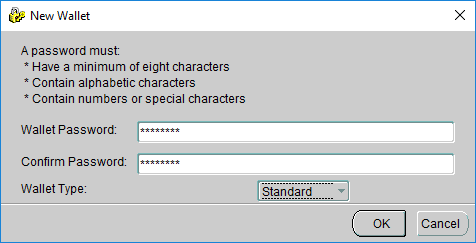


Рисунок 2.15 – Установка пароля бумажнику в Oracle Wallet Manager.

Бумажник не обязательно открывать и закрывать всякий раз по мере надобности вручную. На компьютере, где он располагается, можно сообщить ему режим «автооткрытия» - при запуске ОС. Сделать это можно опять-таки через Oracle Wallet Manager (рисунок 2.16).

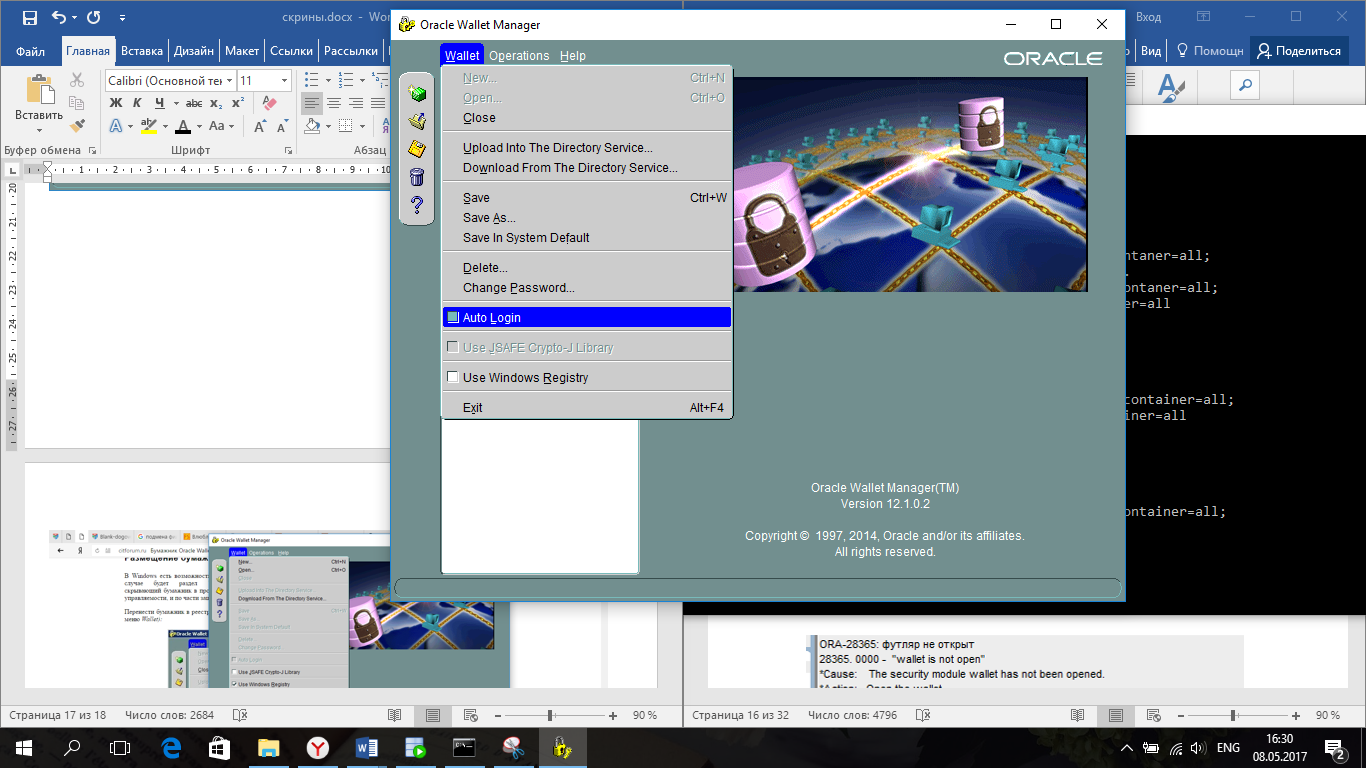


Рисунок 2.16 – Установка режима автоотрытия бумажника.

В Windows есть возможность поместить бумажник не в файл, а в реестр. Местонахождением в этом случае будет раздел \\HKEY\_CURRENT\_USER\SOFTWARE\ORACLE\WALLETS, волей-неволей скрывающий бумажник в профиле пользователя; то есть сохранение его в реестре выгоднее и по части управляемости, и по части защищенности.

Перенести бумажник в реестр можно средствами Oracle Wallet Manager (пометка Use Windows Registry в меню Wallet) (рисунок 2.17).

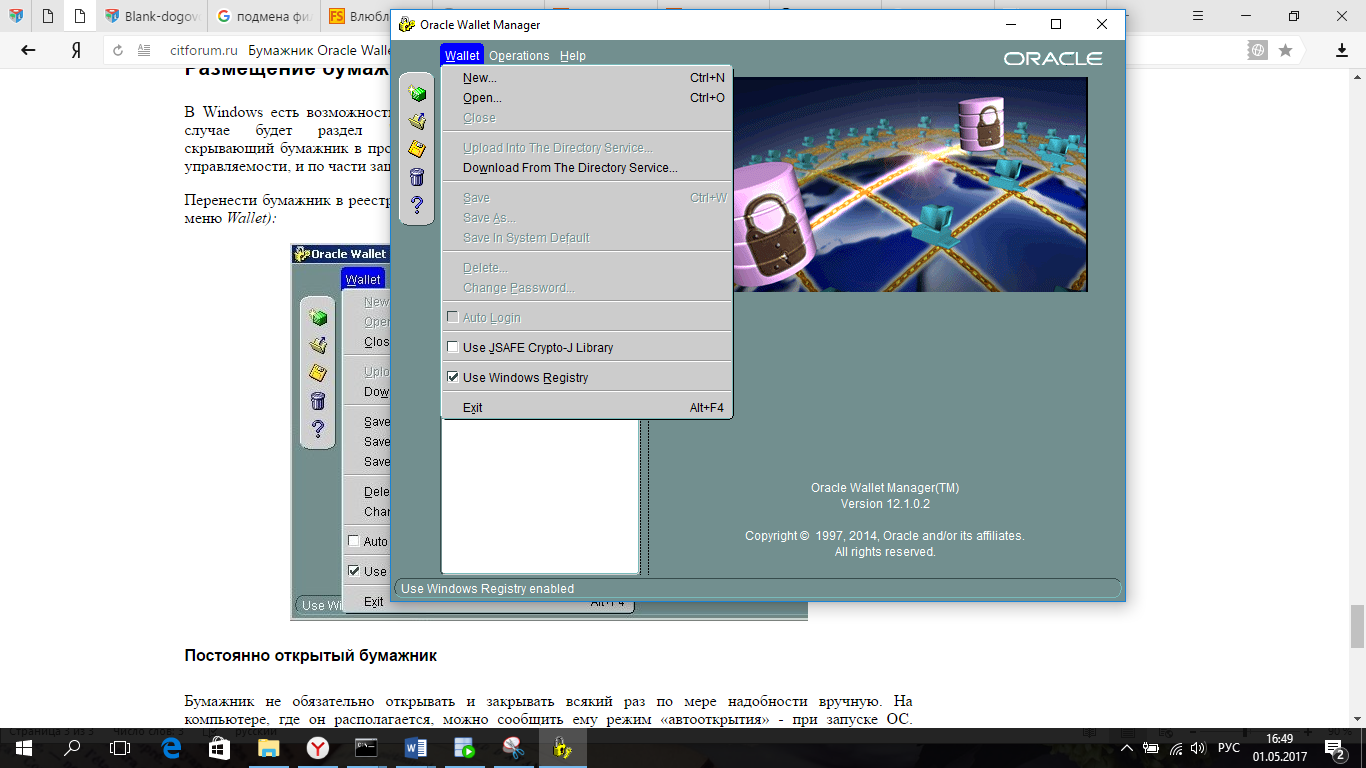


Рисунок 2.17 – Перенос бумажника в реестр.

# **Руководство пользователя**

После того как мы проинициализировали бумажник и открыли его, мы можем убедиться в этом (рисунок 3.1). Если бумажник будет закрыт, то вместо метки OPEN будет стоять CLOSE.

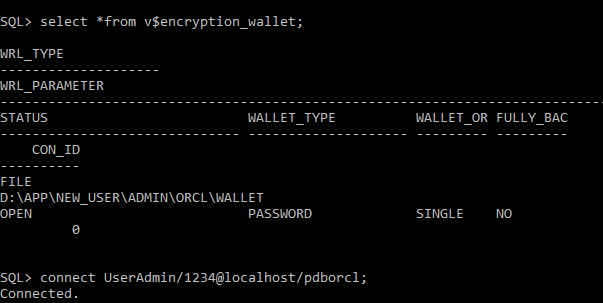


Рисунок 3.1 – Состояние кошелька

Теперь мы можем зашифровать наши данные. Для этого необходимо зайти под главным пользователем, т.е. UserAdmin1 и назначить шифрование столбцам (рисунок 3.2). Также назначить шифрование столбцам мы можем и в SQL Developer, для этого надо будет зайти под администратором.

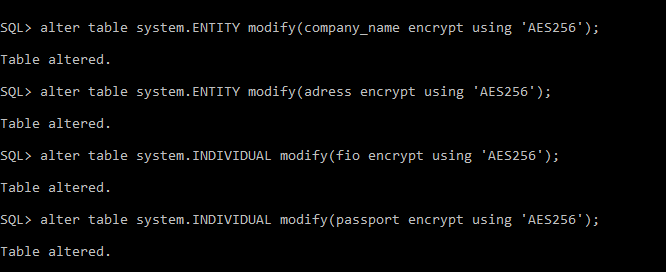


Рисунок 3.2 – Шифрование столбцов

Чтобы проверить зашифровались ли наши столбцы, нужно ввести в SQLPLUS команду desc <таблица>. Около наших столбцов будет стоять метка ENCRYPT (рисунок 3.3).

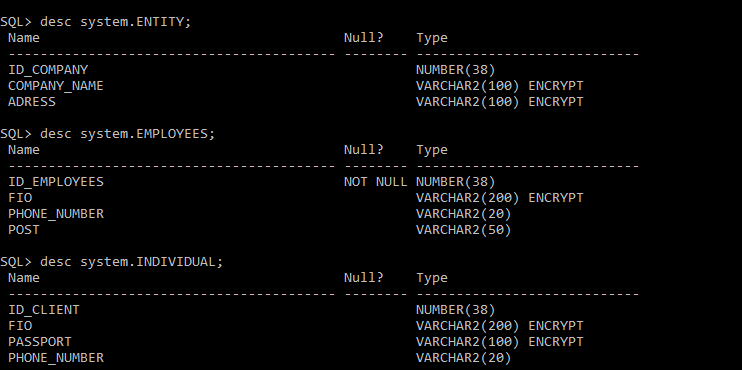


Рисунок 3.3 – Проверка наличия шифрования

Удостоверимся в том, что данные будут защищены в случае, если кто-то похитит нашу базу данных. Для этого закроем наш бумажник и попробуем вывести данные из зашифрованных таблиц. На рисунке 3.4 показано как это будет выглядеть в SQL Developer.

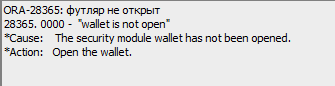


Рисунок 3.4 – Вывод значений из таблицы при закрытом бумажнике

На рисунке 3.5 показано как это выглядит в SQLPLUS.

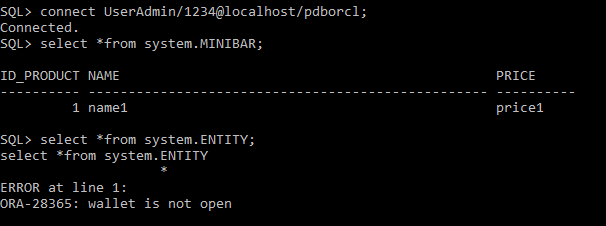


Рисунок 3.5 – Вывод значений из таблиц при закрытом бумажнике

# **Заключение**

Проблема защиты от данных от посторонних всегда стояла перед администратором БД. Помимо надежности. Бумажник Oracle Wallet должен обеспечивать минимальные неудобства пользователям, а сам процесс его использования должен быть максимально прозрачным с их точки зрения.

Прозрачное шифрование базы данных — технология, применяемая, например, в продуктах Microsoft и Oracle для шифрования и дешифрования ввода-вывода файлов [БД](https://ru.wikipedia.org/wiki/%D0%91%D0%94). Данные шифруются перед записью на диск и дешифруются во время чтения в память, что решает проблему защиты «неактивных» данных, но не обеспечивает сохранность информации при передаче по каналам связи или во время использования. Преимуществом TDE является то, что шифрование и дешифрование выполняются прозрачно для приложений, то есть их модификация не требуется. Помимо шифрования данных на уровне столбцов также существуют TDE на уровне табличных пространств, а также путем шифрования резервных копий. Microsoft использует технологию шифрования файловой, которая обеспечивает шифрование на уровне файлов. Каждый объект шифруется с помощью уникального ключа шифрования файлов, защищенного сертификатом пользователя.

В данной курсовой работе был продемонстрирован способ шифрования на уровне столбцов. Были выполнены все условия присуще этому способу шифрования:

* должен быть создан wallet;
* в wallet должен быть создан мастер-ключ;
* wallet с мастер-ключом должен быть «открыт», т. е. сделан доступным для сервера БД;
* колонки таблиц должны быть зашифрованы.

# **Список используемых источников**

1. Oracle для профессионалов. Архитектура, методики программирования и особенности версий 9i, 10g и 11g, 2-е издание, издательский дом «Вильямс», М:, 2011
2. Oracle Database 12c [Электронный ресурс] – <http://www.interface.ru/home.asp?artId=34401>
3. Новые возможности Oracle Database 12.1.0.2 [Электронный ресурс] – <https://habrahabr.ru/company/oracle/blog/279333/>
4. Использование бумажника Oracle Wallet [Электронный ресурс] – <http://citforum.ru/database/oracle/wallet1/>

# **Приложение А Создание таблиц, пользователей и назначение им привилегий**

---------------------------ТАБЛИЦЫ----------------------------

create table CATEGORY(

id\_category int primary key,

type\_number varchar2(50),

state\_number varchar2(20),

information varchar2(100)

);

create table ROOMS(

id\_room int primary key,

id\_category int, constraint fk\_id\_cat foreign key(id\_category) references CATEGORY(id\_category)

);

create table EMPLOYEES(

id\_employees int primary key,

fio varchar2(200),

phone\_number varchar2(20),

post varchar2(50)

);

create table MINIBAR(

id\_product int primary key,

name varchar2(50),

price varchar2(10)

);

create table REGISTRATION(

id\_registration int primary key,

id\_room int, constraint fk\_id\_room foreign key(id\_room) references ROOMS(id\_room),

id\_employees int, constraint fk\_id\_emp foreign key(id\_employees) references EMPLOYEES(id\_employees),

arrival\_date date,

departure\_date date,

prepayment varchar2(10)

);

create table ACCOUNT(

id\_account int primary key,

id\_registration int, constraint fk\_id\_regis foreign key(id\_registration) references REGISTRATION(id\_registration),

id\_product int, constraint fk\_id\_bar foreign key(id\_product) references MINIBAR(id\_product)

);

create table INDIVIDUAL(

id\_client int primary key,

fio varchar2(200),

passport varchar2(100),

phone\_number varchar2(20),

id\_registration int, constraint fk\_id\_reg foreign key(id\_registration) references REGISTRATION(id\_registration)

);

create table ENTITY(

id\_company int primary key,

company\_name varchar2(100),

adress varchar2(100),

id\_registration int, constraint fk\_id\_reg1 foreign key(id\_registration) references REGISTRATION(id\_registration)

);

-------------------------ПОЛЬЗОВАТЕЛИ-------------------------

create role userrole1;

grant all privileges to userrole1;

grant execute on package\_room to userrole1;

grant execute on package\_empl to userrole1;

grant execute on package\_bar to userrole1;

grant execute on package\_services to userrole1;

grant execute on package\_entity to userrole1;

grant execute on package\_individual to userrole1;

grant create any trigger to userrole1;

grant alter any trigger to userrole1;

create role userrole2;

grant create session,

select any table to userrole2;

select \*from dba\_roles;

create profile userprof limit

password\_life\_time 30

sessions\_per\_user 1

failed\_login\_attempts 3

password\_lock\_time 1

password\_grace\_time default

password\_reuse\_time 5

connect\_time 60

IDLE\_time 30;

create profile adminprof limit

password\_life\_time 3

sessions\_per\_user 1

failed\_login\_attempts 3

password\_lock\_time 1

password\_grace\_time default

password\_reuse\_time 5

connect\_time 180

IDLE\_time 30;

drop profile adminprof;

create user UserAdmin1 identified by a12345

profile adminprof;

grant userrole1 to UserAdmin1

grant execute on package\_rooms to UserAdmin1;

grant execute on package\_account to UserAdmin1;

grant execute on package\_regist to UserAdmin1;

alter user UserAdmin1 identified by a1234;

create user UserN identified by 1234

profile userprof;

grant userrole2 to UserN

select \*from all\_users

alter table EMPLOYEES modify(fio encrypt using 'AES256');

alter table ENTITY modify(company\_name encrypt using 'AES256');

alter table ENTITY modify(adress encrypt using 'AES256');

alter table INDIVIDUAL modify(fio encrypt using 'AES256');

alter table INDIVIDUAL modify(passport encrypt using 'AES256');