Специальность 10.05.01 «Компьютерная безопасность»,
Специализация «Математические методы защиты информации»
Уровень высшего образования – специалитет

Дисциплина: Основы построения защищенных баз данных.

Лабораторная работа №6. Администрирование Oracle. Принцип предоставления наименьших привилегий. Аудит.

1. Учебные цели:

- Отработать вопросы управления экземпляром Oracle в части настройки и управления аудитом базы данных.
- Освоить приемы включения и настройки различных вариантов аудита, которые реализованы СУБД и использования для целей аудита триггеров.

2. Требования к результатам обучения основной образовательной программы, достигаемые при проведении лабораторной работы:

- Уметь использовать возможности современных систем для решения задач администрирования и защиты баз данных.
- Владеть инструментами аудита базы данных для отслеживания действий пользователей и выявления подозрительных тенденций.

3. Перечень материально-технического обеспечения

ПЭВМ с проигрывателем виртуальных машин, виртуальная машина с установленной СУБД Oracle.

4. Краткие теоритические сведения и задания на исследование. Задания выделены рамками и синим шрифтом. Результаты лабораторной работы представляются в виде файла, содержащего копии экрана, показывающие этапы выполнения заданий.

Для выполнения лабораторной работы понадобятся некоторые сведения по использованию языка PLSQL.

PL/SQL

PL/SQL – разработанный фирмой Oracle язык четвертого поколения (fourth generation programming language – 4GL), который своими процедурными возможностями расширяет язык SQL. PL/SQL предоставляет общую среду программирования для баз данных и приложений независимо от операционной системы и аппаратной платформы.

В среде PL/SQL можно манипулировать данными с помощью команд SQL, а также управлять потоком выполнения программы, применяя такие процедурных конструкции, как IF-THEN, CASE и LOOP. Кроме того, можно объявлять константы и переменные, определять процедуры и функции, использовать коллекции и объектные типы, а также перехватывать ошибки в ходе выполнения. Предоставляется возможность вызова из кода PL/SQL программ, написанных на других языках, например C, C++ и Java.

PL/SQL также обеспечивает защиту данных. Чтобы выполнить вызов, вызывающая компонента (caller) не должна знать структуру читаемых и обрабатываемых данных. Ей также не нужны права доступа к этим объектам; все, что необходимо, это наличие у вызывающей компоненты права выполнения программы PL/SQL. Однако можно воспользоваться и другим режимом, в котором вызывающая компонента должна иметь такие права, которые позволяют успешно выполнить каждую команду вызываемой программы PL/SQL.

Так как код PL/SQL выполняется внутри базы данных, он очень эффективен для реализации операций, интенсивно использующих данные. Кроме того, при его применении уменьшается сетевой трафик приложений.

Дополнительную информацию о процедурных конструкциях и использовании PL/SQL см. в документе PL/SQL User's Guide and Reference.

Сопровождение объектов PL/SQL

Администратор базы данных обычно не несет ответственность за загрузку кода PL/SQL в базу данных и не обязан помогать разработчикам при поиске причин ошибок в их программах. Он также обычно не пишет код приложения, используя PL/SQL. Несмотря на все это, администратору следует быть знакомым с различными объектами PL/SQL, чтобы быть в состоянии дать рекомендации разработчикам приложений и найти проблемы, связанные с использованием объектов PL/SQL.

С помощью консоли Database Control можно сопровождать объекты PL/SQL. Ссылки на различные типы объектов находятся в секции Schema на странице, доступной по закладке Administration. После выбора типа объектов PL/SQL можно просматривать, модифицировать и создавать объекты выбранного типа.

Объекты PL/SQL

Спецификация пакета (package). Пакет – это совокупность логически связанных процедур и функций. Пакет обязательно содержит спецификацию (она хранится в БД с типом package), в которой описывается интерфейс к возможностям пакета. В спецификации объявляются доступные извне типы, переменные, константы, исключения, курсоры и подпрограммы.

Тело пакета (package body) реализует спецификацию и полностью определяет курсоры и подпрограммы. В теле пакета находятся скрытые от вызывающей программы (caller) детали реализации и объявления частных объектов.

Тело типа (type body) – это совокупность методов (процедур и функций), связанных с определяемыми пользователями типами данных. Дополнительную информацию см. в документе Oracle Database Application Developer's Guide – Object Relational Features

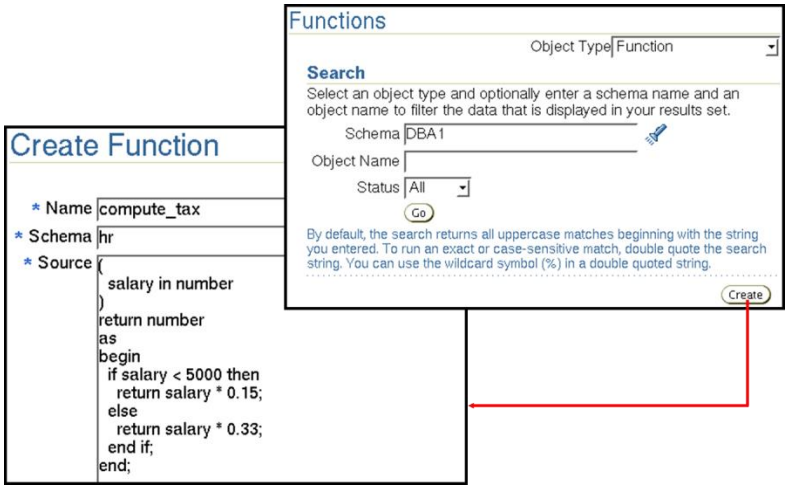
Процедура (procedure) – это блок PL/SQL, выполняющий заданные действия.

Функция (function) – блок PL/SQL, возвращающий единственное значение, указываемое в команде PL/SQL RETURN.

Триггер (trigger) – это блок PL/SQL, выполняемый при возникновении конкретного события в базе данных. События могут быть связаны с таблицей, например, вставка строки в таблицу. Они также могут также происходить на уровне базы данных, например, событие, возникающее сразу после подсоединения пользователя к БД.

Функции

Функции PL/SQL обычно используются для вычисления значения. Существует большое число встроенных функций, например, SYSDATE, SUM, AVG и TO_DATE. Кроме того, разработчики приложений создают свои собственные функции. В коде PL/SQL должна быть команда RETURN. На слайде показан пример создания функции в среде Enterprise Manager. Для новой функции вводится имя, схема, в которой она создается, а также исходный код.



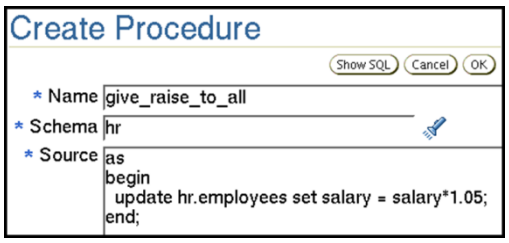
Ниже приведен полный код создания функции computetax, показанной на рисунке:

```
CREATE OR REPLACE FUNCTION compute_tax (salary NUMBER)
RETURN NUMBER
AS
BEGIN
  IF salary<5000 THEN
    RETURN salary*.15;
  ELSE
    RETURN salary*.33;
  END IF;
END;
/
```

Процедуры (Procedures)

Процедуры используются для выполнения заданных действий. Процедуры:

- Также как и функции, могут принимать входные значения и выполнять условные операторы (IF-THEN, CASE и LOOP)
- Вызываются по команде CALL

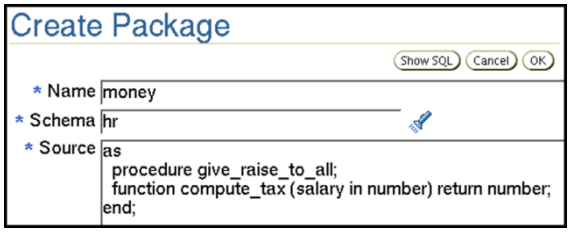


Пакеты

Пакеты группируют функции и процедуры. Такое объединение дает преимущества с точки зрения производительности и сопровождения кода компонентов пакета. Для каждого пакета следует создавать два отдельно компилируемых объекта базы данных: спецификацию пакета и тело пакета.

Спецификация пакета (иногда ее называют заголовком пакета) имеет тип PACKAGE. В ней содержатся только определения процедур, функций и переменных пакета.

Тело пакета имеет тип PACKAGE BODY и содержит фактический код подпрограмм, определенных в спецификации пакета.

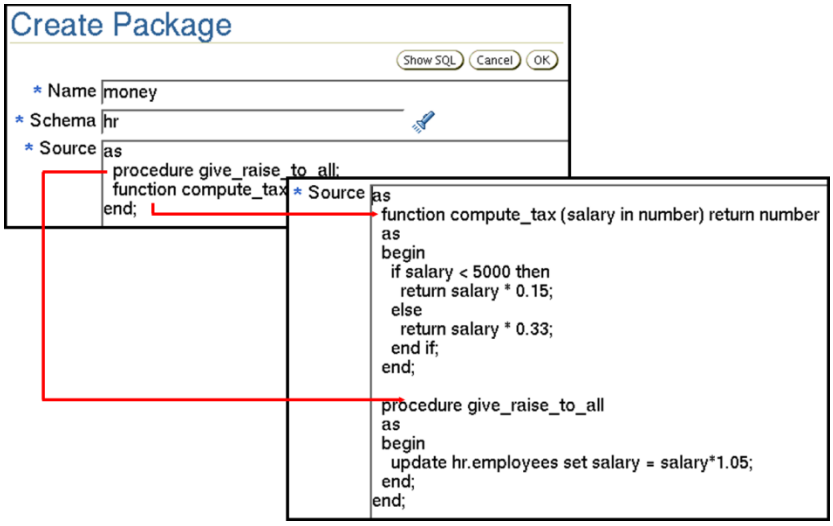


Пакетные процедуры и функции вызываются извне пакета следующим образом:

`имя_пакета.имя_процедуры` или `имя_пакета.имя_функции`

Примеры использования подпрограмм пакета:

```
SQL> SELECT money.compute_tax(salary) FROM hr.employees WHERE employee_id=107;
SQL> EXECUTE money.give_raise_to_all;
```



Встроенные пакеты

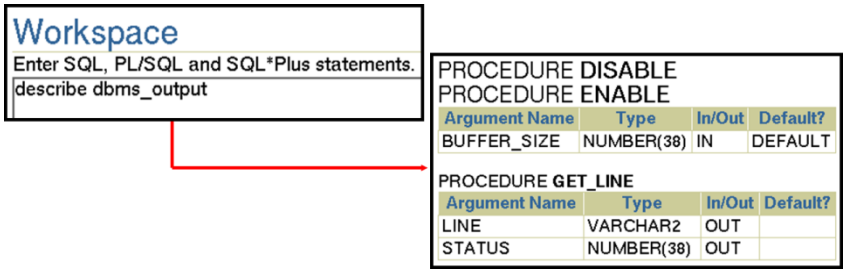
Встроенные PL/SQL-пакеты, поставляемые вместе с базой данных Oracle, предоставляют доступ к расширенным возможностям базы данных, например, обработке очередей (advanced queuing), шифрованию и вводу/выводу файлов. Эти пакеты также содержат утилиты администрирования и сопровождения.

Набор используемых пакетов зависит от вида приложений, выполняемых на сервере базы данных. Примеры наиболее часто используемых для администрирования и сопровождения пакетов:

- DBMS_STATS; сбор, просмотр и изменение статистик оптимизатора.
- DBMS_OUTPUT; формирование выходных данных в PL/SQL
- DBMS_SESSION; доступ из кода PL/SQL к командам ALTER SESSION и SET ROLE.
- DBMS_RANDOM; генерирование случайных чисел.
- DBMS_UTILITY; оценка времени выполнения, использования времени ЦП, получение сведений о версии базы данных; вычисление хэш-значения и выполнение многих других разнообразных функций.
- DBMS_SCHEDULER; планирование выполнения процедур и функций, вызываемых из кода PL/SQL
- DBMS_CRYPTO; шифрование и дешифрование информации базы данных.
- UTL_FILE; чтение и запись из PL/SQL файлов операционной системы.

Детальную информацию о встроенных пакетах см. в документе PL/SQL Packages and Types Reference.

Для просмотра подпрограмм пакета используйте команду DESCRIBE:

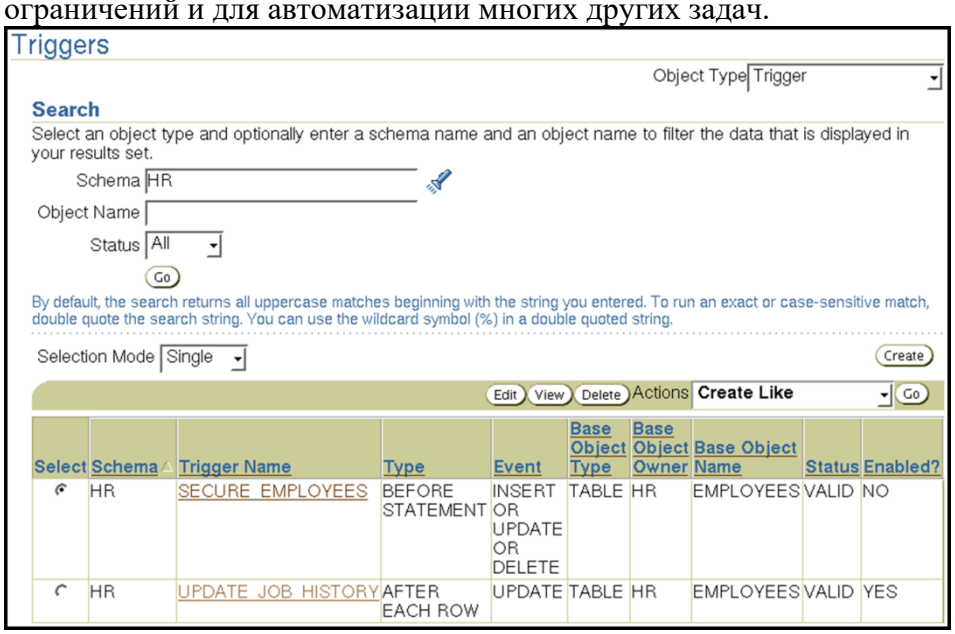


Триггеры

Триггеры – это хранимые в базе данных кодовые объекты PL/SQL, которые срабатывают и выполняются автоматически, когда что-то происходит. База данных Oracle позволяет обрабатывать с помощью триггеров многие события. Например, вставку в таблицу, подсоединение пользователя к БД, кем-то предпринимаемую попытку удаления таблицы или изменения параметров аудита.

Триггеры могут вызывать другие процедуры и функции (опытный разработчик обычно создает триггер с коротким кодом и выносит действия, программируемые длинным кодом в отдельный пакет).

АБД используют триггеры для аудита по значениям данных, для проверки сложных ограничений и для автоматизации многих других задач.



Например, триггер SECURE_EMPLOYEES, показанный на слайде, разрешает выполнять команды DML, обрабатывающие данные таблицы EMPLOYEES, только во время рабочего дня.

События, по которым срабатывают триггеры

Существует много различных событий, которые могут быть использованы для создания триггеров. Эти события можно разделить на три категории:

DML-операции, вызывающие срабатывание триггеров, когда команды DML модифицируют данные.

DDL-операции, вызывающие срабатывание триггеров, когда команды создают или каким-либо образом модифицирую объект.

Определенные события, происходящие на уровне базы данных.

Тип события	Примеры событий
DML	INSERT, UPDATE, DELETE
DDL	CREATE, DROP, ALTER, GRANT, REVOKE, RENAME
Database	LOGON, LOGOFF, STARTUP, SHUTDOWN, SERVERERROR

Большинство триггеров срабатывает перед событием или после него. Триггеры DML могут быть созданы для событий, возникающих один раз на уровне команды или же для событий, возникающих при изменении каждой строки таблицы.

Аудит.

Даже пользователь, прошедший процедуру аутентификации и авторизации, может иногда попытаться выполнить несанкционированные действия. Выявление необычных операций в базе данных, например, внезапных больших запросов информации о кредитных карточках, результатах исследований или другой секретной информации, может быть первым шагом при обнаружении воровства информации. Oracle предоставляет богатый набор инструментов аудита для отслеживания действий пользователей и выявления подозрительных тенденций.

- Установка только необходимого программного обеспечения. Уменьшение количества пакетов программного обеспечения снижает затраты на сопровождение и обновление (upgrades), вероятность появления “дыр безопасности” и конфликтов в программном обеспечении.
- Запуск на машине только требуемых служб. При меньшем количестве служб меньше открытых портов и меньше направлений атак для злоумышленников.
- Предоставление доступа к ОС и базе данных только тем пользователям, которым доступ необходим. Меньше пользователей – меньше паролей и учетных записей. Снижается вероятность наличия учетных записей в открытом или просроченном состояниях. При меньшем количестве учетных записей администратору проще поддерживать их текущее состояние.
- Ограничение доступа к учетной записи root или администратора. Необходимо тщательно защищать, вести аудит и никогда не использовать совместно учетную запись администратора.
- Ограничение доступа к учетным записям с привилегиями SYSDBA и SYSOPER. Пользователи, которым необходимы эти роли, должны иметь собственные учетные записи и для них должен производиться аудит.
- Предоставление доступа пользователям только к таким объектам базы данных, которые необходимы для выполнения их заданий. Пользователи, которым предоставлен доступ к большему числу объектов, чем это необходимо, имеют возможность причинить вред.
- Пользователю следует предоставлять только такие привилегии, которые ему необходимы для эффективного решения задачи. Применение этого принципа уменьшает риск того, что пользователи смогут случайно или умышленно изменить или просмотреть данные, к которым им не должны были быть предоставлены привилегии доступа, позволяющие это сделать.

Защита словаря данных.

По умолчанию параметр O7_DICTIONARY_ACCESSIBILITY имеет значение FALSE. Не следует изменять этот параметр, если это не обосновано очень вескими причинами. Значение FALSE препятствует доступу пользователей с системными привилегиями ANY

TABLE к базовым таблицам словаря. Кроме того, значение FALSE не допускает подсоединения пользователя SYS без привилегии SYSDBA.

Ограничение предоставления привилегий на использование библиотек.

В число пакетов с большими возможностями, которыми можно злонамеренно воспользоваться, входят:

- UTL_SMTP; позволяет послать произвольное почтовое сообщение с помощью базы, используемой в качестве почтового сервера, работающего по протоколу SMTP (Simple Mail Transfer Protocol). Предоставление всем (PUBLIC) права применения этого пакета может привести к несанкционированному обмену почтовыми сообщениями.
- UTL_TCP; разрешает устанавливать серверу базы данных исходящие сетевые соединения с любыми принимающими и находящимися в ожидании сетевыми службами. Таким образом, произвольно выбранные данные могут быть посланы между сервером базы данных и ожидающей сетевой службой.
- UTL_HTTP; разрешает серверу базы данных запрашивать и принимать данные, передаваемые по протоколу HTTP. Выделение привилегии на этот пакет всем (PUBLIC) может позволить передать данные с помощью форм HTML на Web-сайт злоумышленников.
- UTL_FILE; если этот пакет неправильно сконфигурирован, отрывается доступ на текстовом уровне к любому файлу в операционной системе хоста. Даже при правильном конфигурировании пакет не различает вызывающие его приложения. В результате приложение, имеющее доступ к пакету UTL_FILE может писать произвольные данные в то же самое место, в которое пишет другое приложение.

Приведенные пакеты чрезвычайно полезны приложениям, которые в них нуждаются, но требуют надлежащего конфигурирования для безопасного использования. Отберите право их выполнения у PUBLIC и предоставьте его отдельным ролям, которым они необходимы.

Лишние привилегии, выданные всем (PUBLIC), должны быть отобраны, например:

```
REVOKE EXECUTE ON UTL_SMTP, UTL_TCP, UTL_HTTP, UTL_FILE
FROM PUBLIC;
```

Ограничение доступа пользователей к каталогам ОС.

Объект DIRECTORY, хранимый внутри базы данных, позволяет АБД отображать директорию БД в путь ОС и предоставлять привилегии на такие директории отдельным пользователям.

Сокращение числа пользователей с административными привилегиями.

Никогда не предоставляйте больших привилегий пользователям, чем это необходимо. Пользователям, не являющимся администраторами, не следует предоставлять роль DBA. Чтобы применить принцип наименьших привилегий, ограниченно предоставляйте следующие виды привилегий:

- позволяющие предоставлять системные и объектные привилегии;
- позволяющие соединяться с правами пользователя SYS (SYSDBA и SYSOPER);
- привилегии роли DBA, например, DROP ANY TABLE.

Ограничение на удаленную аутентификацию БД.

Параметр REMOTE_OS_AUTHENT по умолчанию имеет значение FALSE. Это значение не следует изменять, за исключением ситуации, когда всем клиентам можно доверить аутентификацию соответствующих пользователей.

В процессе удаленной аутентификации:

- производится внешняя (externally) аутентификация пользователя БД;
- аутентификация пользователя выполняется на удаленной системе;
- пользователь подсоединятся к БД без дальнейшей аутентификации.

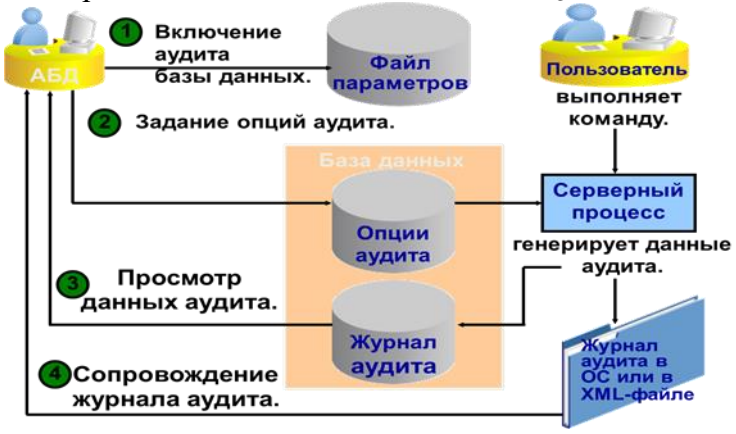
Наблюдение за подозрительными действиями.

Аудит означает сбор и хранение информации о том, что происходит в системе. Его выполнение **увеличивает нагрузку на систему**. Поэтому аудит должен быть целенаправленным и отслеживать только такие события, сведения о которых необходимо собирать. Соответствующим образом сфокусированный аудит оказывает минимальное влияние на производительность. Чрезмерно обширный аудит может существенно повлиять на производительность.

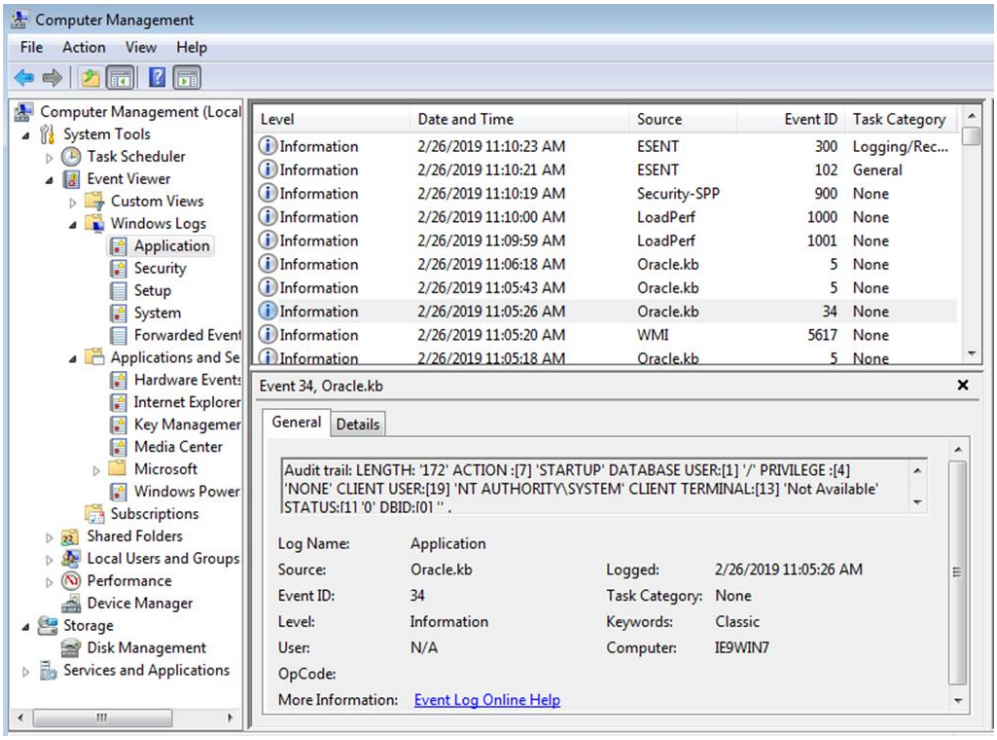
- Обязательный аудит. Все базы данных Oracle выполняют аудит определенных действий, независимо от других опций и параметров аудита. Обязательный аудит фиксирует в журнальных файлах некоторые операции, например startup и shutdown.
- Стандартный аудит базы данных. Задается на уровне системы с помощью параметра инициализации AUDIT_TRAIL. После включения аудита необходимо выбрать объекты и привилегии, за которыми необходимо вести наблюдение.
- Аудит по значениям (value-based auditing). Такой аудит расширяет стандартный аудит БД, фиксируя не только произошедшее событие, но и фактические значения, которые были вставлены, изменены или удалены. Он реализуется с помощью триггеров базы данных.
- Дифференцированный аудит (fine-grained auditing – FGA) расширяет стандартный аудит БД, регистрируя выполняемые команды SQL, а не только происходящие события.
- Аудит АБД. Разделяйте аудит, который обязан выполнять АБД, и аудит, производимый аудитором или системным администратором, ведущим мониторинг действий АБД с использованием журнала аудита операционной системы.

Стандартный аудит базы данных.

БД начинает собирать данные аудита после включения аудита базы данных и задания его опций, определяющих регистрацию событий входа в систему, использования системных и объектных привилегий, выполнения команд SQL.



Если значение AUDIT_TRAIL равно OS, тогда записи проводимого аудита сохраняются в журнале аудита, находящемся в операционной системе. В среде Windows это журнал событий (event log), в UNIX или Linux записи сохраняются в файле. Месторасположение этого файла задается в параметре AUDIT_FILE_DEST.



Найдите в журнале ОС Windows записи о запуске СУБД.

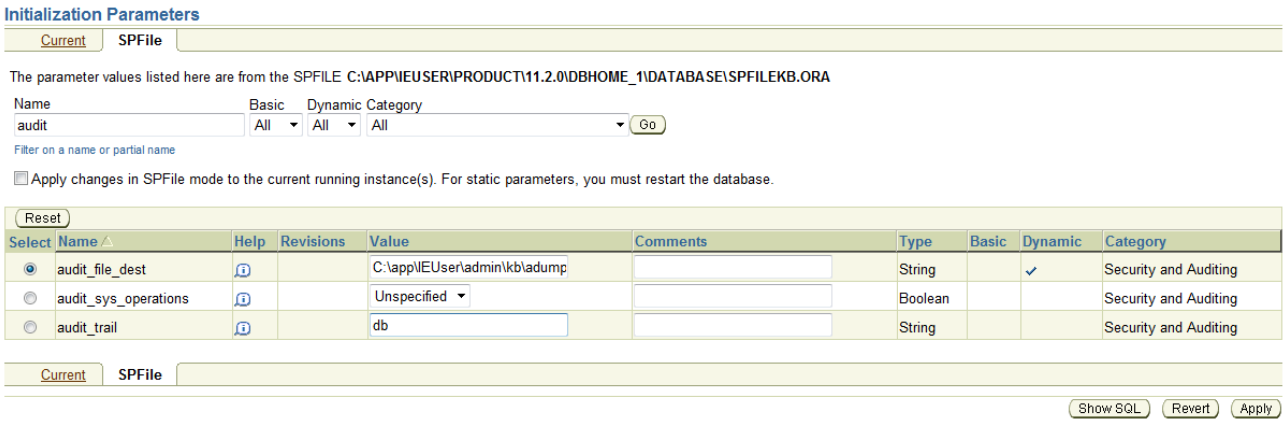
В случае, когда параметр AUDIT_TRAIL равен DB, записи аудита можно просмотреть в представлении DBA_AUDIT_TRAIL, принадлежащем схеме SYS.

Когда значение AUDIT_TRAIL равно XML или XML,EXTENDED, записи аудита записываются в XML-файлы в каталоге, на который указывает параметр AUDIT_FILE_DEST. Представление V$XML_AUDIT_TRAIL позволяет просмотреть все XML-файлы из этой директории.

Сопровождение журнала аудита – важная административная задача. В зависимости от фокусировки аудита, определяемого задаваемыми опциями, журнал с его результатами может расти очень сильно и чрезвычайно быстро. При отсутствии соответствующего сопровождения журнал может занять такое большое пространство, что это окажет влияние на производительность системы.

Включение аудита

Необходимо включить аудит перед тем, как задавать его установки.



Пример:

```
ALTER SYSTEM SET audit_trail=”XML” SCOPE=SPFILE;
```

Средствами Enterprise Manager включите аудит базы данных, измените параметры, как задано в командах ниже. Перезапустите базу данных.

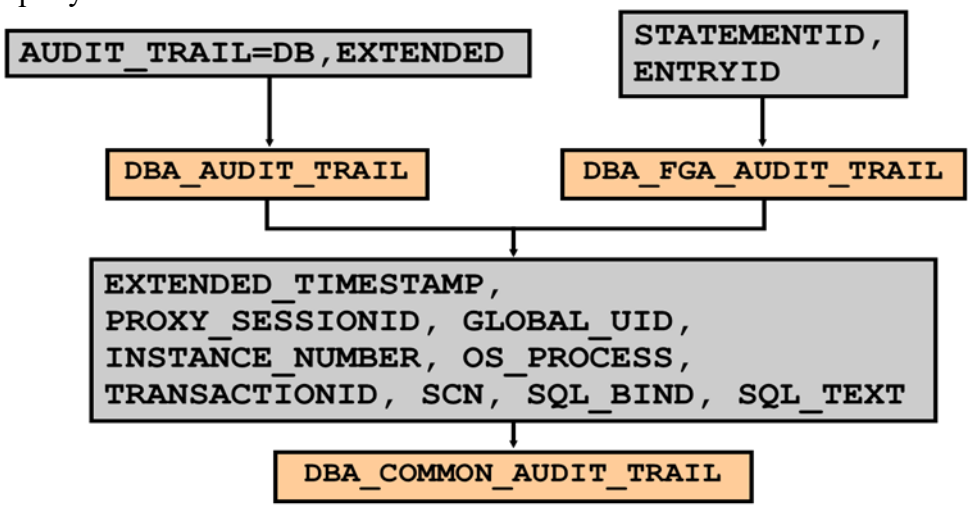
ALTER SYSTEM SET audit_trail=" DB_EXTENDED" SCOPE=SPFILE;
ALTER SYSTEM SET audit_sys_operations=true SCOPE=SPFILE;

После модификации статических параметров инициализации необходимо перезапустить базу данных.

Унифицированный журнал аудита

Для того чтобы воспользоваться аудитом, прежде всего, необходимо установить нединамический параметр AUDIT_TRAIL, указывающий на то, где будут храниться записи аудита. Аудит включается в результате установки этого параметра.

База данных Oracle отслеживает данные стандартного и дифференцированного аудита в похожих полях журналов. Это позволяет проще анализировать действия в базе данных. Журналы стандартного аудита и дифференцированного аудита имеют дополняющие друг другу атрибуты.



Дополнительные данные, собираемые **стандартным** аудитом, включают:

• Системный номер изменения (system change number, SCN), который записывается при каждом изменении в системе.

• Точный код SQL, выполненный пользователем, и связанные переменные, использованные в тексте кода SQL. Эти столбцы заполняются только при условии, что AUDIT_TRAIL=DB_EXTENDED.

Дополнительные данные, собираемые **дифференцированным** аудитом, включают:

• Серийный номер каждой записи аудита.

• Номер команды, связывающий несколько записей аудита, созданных при выполнении одной и той же команды.

Общие атрибуты включают:

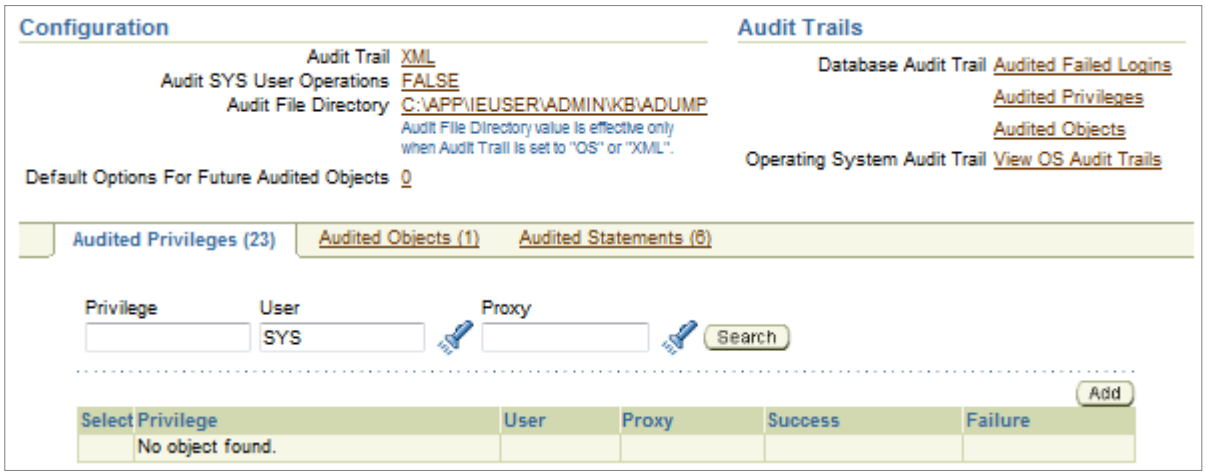
• Глобальная временная метка в виде универсального синхронизированного времени (среднее время по Гринвичу, Universal Time Coordinates, UTC). Это поле полезно для мониторинга серверов, располагаемых в различных географических точках и временных зонах.

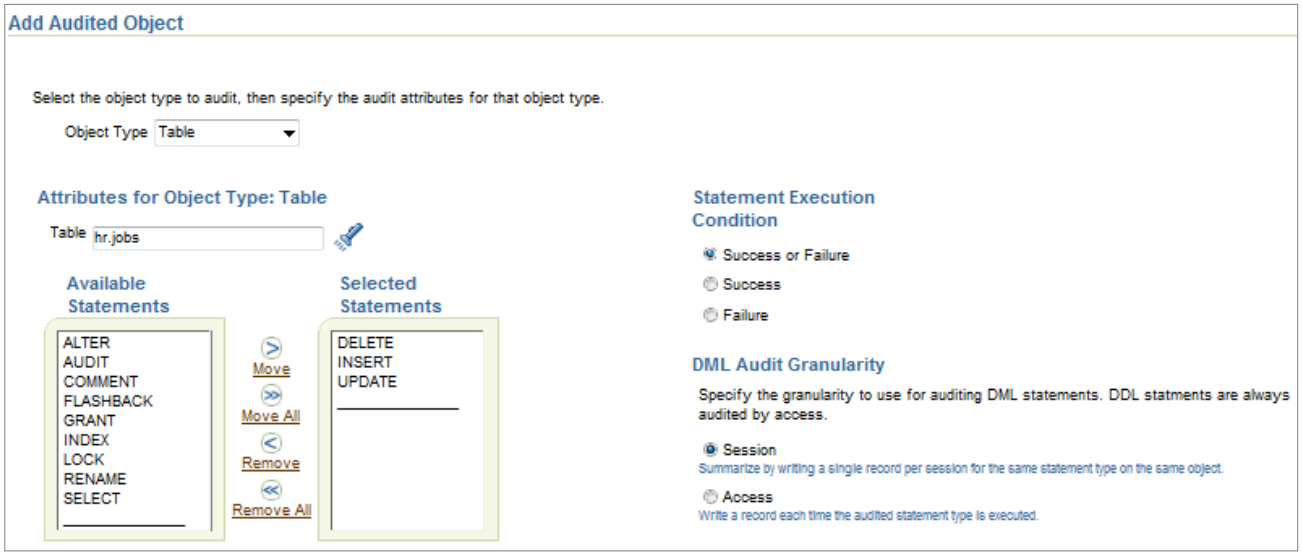
• Номер экземпляра, значение которого уникально для каждого экземпляра Real Application Clusters (RAC).

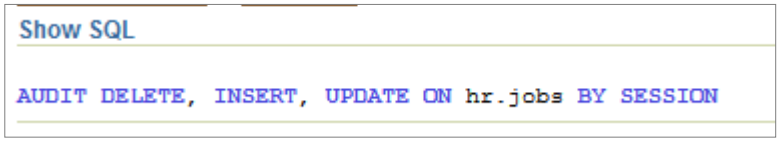
• Идентификатор транзакции, помогающий группировать записи аудита одной и той же транзакции.

Представление DBA_COMMON_AUDIT_TRAIL объединяет строки журналов стандартного и дифференцированного аудита.

Настройка параметров аудита средствами Enterprise Manager.







Варианты задания аудита

1) Аудит команд SQL

Пример:

```
AUDIT table;
```

Команда вызовет аудит любой команды DDL, связанной с таблицей: CREATE TABLE, DROP TABLE, TRUNCATE TABLE и т.д. Такой вид аудита можно сфокусировать на

конкретном пользователе и удачном (SUCCESSFUL) или неудачном (NOT SUCCESSFUL) завершении операции.
Пример:
```
SQL> AUDIT TABLE BY hr WHENEVER NOT SUCCESSFUL;
```

2) Аудит использования системных привилегий (несфокусированный и сфокусированный)

Пример:
```
AUDIT select any table, create any trigger;
AUDIT select any table BY hr BY SESSION;
```

Используется для сбора данных об использовании какой-либо системной привилегии (например, DROP ANY TABLE). Он может быть сфокусирован на конкретном пользователе и удачном или неудачном завершении операции. По умолчанию каждое использование системной привилегии генерирует запись аудита. Записи можно группировать таким образом, чтобы только одна запись генерировалась для сеанса (так при изменении пользователем 100 000 строк в таблице другого пользователя получается только одна запись аудита). Фразу BY SESSION (аудит на уровне сеанса) необходимо задавать, так как по умолчанию действует BY ACCESS (аудит на уровне доступа). Рекомендуется использовать аудит на уровне сеанса, чтобы ограничить влияние аудита системных привилегий на производительность и расходование памяти.

3)Аудит использования объектных привилегий (несфокусированный и сфокусированный)

Пример:
```
AUDIT ALL on hr.employees;
AUDIT UPDATE, DELETE on hr.employees BY ACCESS;
```

Применяется для наблюдения за операциями с таблицами, представлениями, процедурами, последовательностями, объектами directory и пользовательскими типами данных. Такой аудит может быть сфокусирован на удачных или неудачных операциях и фиксироваться на уровне сеанса или каждого обращения. В отличие от аудита системных привилегий по умолчанию действует аудит на уровне сеанса. Если необходимо получить в журнале записи по каждому использованию объекта, следует указывать фразу BY ACCESS.

Подсказка по наилучшему практическому использованию: поскольку аудит увеличивает системную нагрузку, отключите все, что не используется.

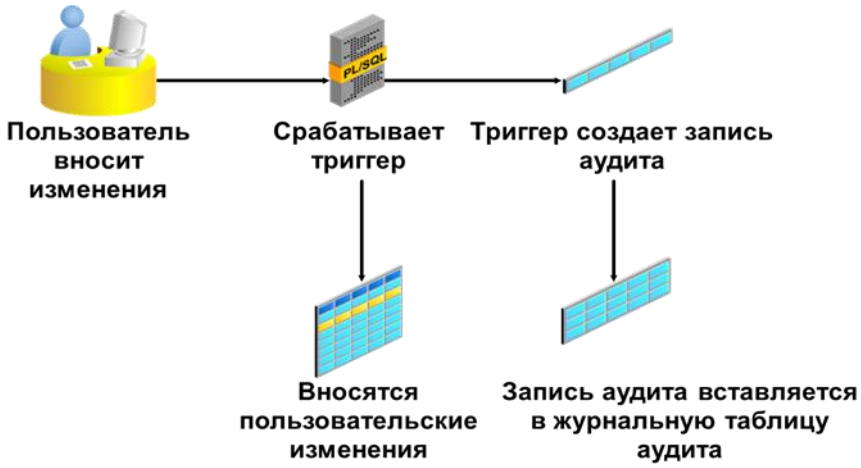
Средствами Enterprise Manager настройте аудит SQL-команд Select, Update, Insert для таблицы Jobs пользователя HR.

Войдите в базу данных под учетной записью JGOODMAN (была создана в лабораторной работе №4).
Сделайте выборку из таблицы Jobs пользователя HR.
Измените данные в одной из строк таблицы Jobs пользователя HR (или добавьте строку).
Найдите записи в журнале аудита средствами Enterprise Manager.

От имени sys выполните запрос вида:
```
Select AUDIT_TYPE, DB_USER, EXTENDED_TIMESTAMP, SQL_TEXT
  from DBA_COMMON_AUDIT_TRAIL
  where DBA_COMMON_AUDIT_TRAIL.OBJECT_NAME = 'JOBS';
```

Средствами SQL Developer найдите представление DBA_COMMON_AUDIT_TRAIL, посмотрите, какие еще данные попадают в таблицу аудита.
Найдите информацию о действиях пользователя SYS.

Аудит по значениям.



Аудит базы данных не может регистрировать значения столбцов для операций вставки, изменения и удаления при выполнении аудита объектов. *Аудит по значениям* (*value-based auditing*) расширяет аудит базы данных, отслеживая изменения фактических значений. Такой аудит может быть выполнен с помощью триггеров базы данных (срабатывающие по событиям конструкции PL/SQL).

При вставке, изменении и удалении данных из таблицы в фоновом режиме прорабатывает соответствующий триггер данной таблицы. Он записывает перехватываемые, измененяемые данные в таблицу, созданную для хранения результатов аудита. Аудит по значениям может вызвать большее снижение производительности, чем аудит базы данных, так как триггер должен прорабатывать при каждой операции вставки, изменения или удаления. Степень снижения производительности зависит от эффективности кода триггера. Аудит по значениям должен использоваться только в ситуациях, когда стандартный аудит базы данных предоставляет недостаточные данные.

Аудит по значениям использует триггеры, подобные приведенному ниже:

```
CREATE OR REPLACE TRIGGER sys.hrsalary_audit
 AFTER UPDATE OF salary
 ON hr.employees
 REFERENCING NEW AS NEW OLD AS OLD
 FOR EACH ROW
BEGIN
 IF :old.salary != :new.salary THEN
 INSERT INTO sys.audit_employees
 VALUES (sys_context('userenv','os_user',40),
 sysdate,
 sys_context('userenv','ip_address',20),
 :new.employee_id || ' salary changed from '||:old.salary||
 ' to '||:new.salary);
 END IF;
END;
```

Аудит, выполняемый в триггере, сфокусирован на перехвате изменений оклада в таблице hr.employees. Когда строка изменяется, триггер проверяет значение столбца salary.

Если старое значение не совпадает с новым, триггер вставляет запись аудита в таблицу audit_employees, созданную ранее отдельной операцией в схеме SYS, командой вида:

```
create table audit_employees (
user_os varchar2(40),
aud_date date,
user_ip varchar2(20),
change_data varchar2(200)
);
```

Запись аудита включает имя пользователя, адрес IP компьютера клиента, с которого были произведены изменения, главный ключ, однозначно определяющий измененную строку, и значения старого и нового окладов.

Триггеры базы данных могут также использоваться для сбора информации о соединениях пользователей, если стандартный аудит базы данных не регистрирует достаточный объем информации. С помощью триггеров logon администратор может собирать данные о пользователе, подсоединяющемся к базе данных, например:

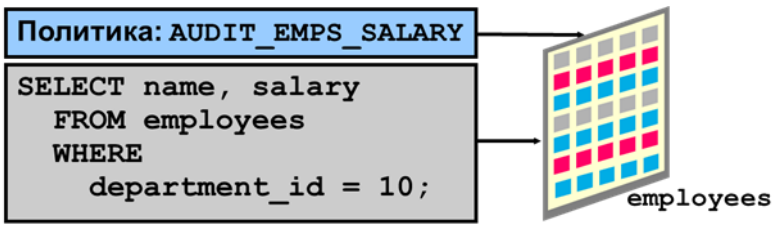
- адрес IP, с которого подсоединился пользователь;
- первые 48 символов имени программы, используемой для соединения пользователя с экземпляром;
- наименование терминала, используемого для соединения с экземпляром.

Полный список параметров пользователя см. в разделе “SYS_CONTEXT” документа *Oracle Database SQL Reference*.

Под учетной записью SYS создайте таблицу `audit_employees` и триггер `hrsalary_audit`, как показано выше.
Измените данные в столбце `salary` таблицы `employees` пользователя HR.
Просмотрите таблицу `audit_employees` с данными аудита.

Дифференцированный аудит.

- Основанный на контексте мониторинг доступа к данным
- Аудит команд SELECT, INSERT,UPDATE,DELETE и MERGE
- Может быть связан с таблицей или представлением, с одним или несколькими столбцами
- Может вызвать срабатывание процедуры
- Используется пакет DBMS_FGA



Аудит БД фиксирует только операцию, но не всегда сохраняет информацию о команде. Дифференцированный аудит (*fine grained auditing – FGA*) позволяет регистрировать команды запросов и манипулирования данных языка SQL. FGA – это наиболее точно сфокусированный вид аудита по сравнению с аудитом по значениям и стандартным аудитом БД.

Параметры FGA позволяют сфокусироваться на отдельных столбцах таблицы или представления, задать условия (определяемые администратором спецификации) выполнения аудита. Политика FGA может быть определена на основе нескольких столбцов. По умолчанию аудит выполняется, когда команда SQL содержит любой из наблюдаемых столбцов (`audit_column_opts = DBMS_FGA.ANY_COLUMNS`). Можно также определить аудит,

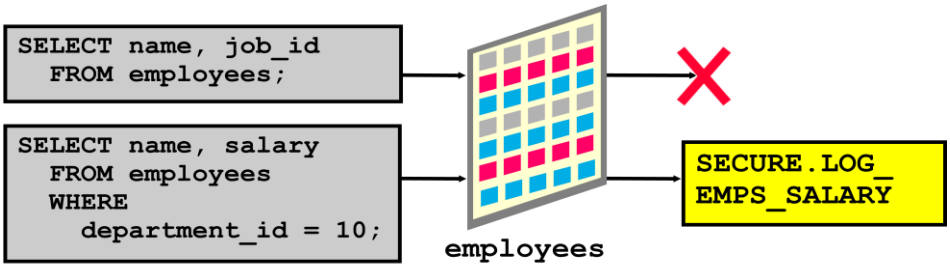
регистрирующий действие, если команда SQL содержит все наблюдаемые столбцы (`audit_column_opts = DBMS_FGA.ALL_COLUMNS`).

Используйте пакет PL/SQL DBMS_FGA, чтобы создать *политику аудита* (*audit policy*) для таблицы или представления. Когда строки блока запроса содержат наблюдаемые столбцы и удовлетворяют условиям аудита, срабатывает событие аудита, в результате которого создается и сохраняется в журнале строка аудита. Дополнительно событие аудита может вызвать выполнение процедуры. FGA автоматически фокусируется на уровне команды, поэтому одна команда SELECT, возвращающая тысячи строк генерирует только одну запись аудита.

Политика FGA

Пример:

```
dbms_fga.add_policy (
 object_schema      =>   'hr',
 object_name        =>   'employees',
 policy_name        =>   'audit_emps_salary',
 audit_condition    =>   'dept_id=10',
 audit_column       =>   'salary,commission_pct',
 handler_schema     =>   'secure',
 handler_module     =>   'log_emps_salary',
 enable             =>   TRUE,
 statement_types    =>   'select' );
```



Пример показывает политику FGA, созданную с помощью процедуры DBMS_FGA.ADD_POLICY, которая имеет следующие аргументы:

- Policy Name (наименование политики)

Каждой политике FGA, когда она создается, присваивается имя. В примере это имя – AUDIT_EMPS_SALARY:

policy_name => 'audit_emps_salary'

- Audit Condition (условия аудита)

Условие аудита – это предикат на SQL, определяющий, когда срабатывает событие аудита. В примере ведется наблюдение за всеми строками отдела продаж (sales), номер которого 10:

audit_condition => 'department_id = 10'

- Statement Type (тип команды)

Аудит можно вести за командой SELECT или же за заданными списком командами INSERT,UPDATE,DELETE.

- Audit Column (наблюдаемые столбцы)

Параметр audit_column указывает, за какими данными ведется наблюдения. Событие аудита происходит, когда столбец входит в команду SELECT или условие аудита допускает выборку наблюдаемых данных. В примере на слайде определяется аудит двух столбцов:

audit_column => 'SALARY, COMMISION_PCT'

Этот аргумент необязательный. Если он не задан, только AUDIT_CONDITION определяет условие срабатывания события аудита.

- Объект

Объектом аудита может быть таблица или представление. Он определяется с помощью двух аргументов:

- схема, содержащая объект;
- имя объекта.

В примере выше задается аудит таблицы hr.employees:

```
object_schema => 'hr'
object_name => 'employees'
```

- Обработчик

Необязательная PL/SQL-процедура обрабатывает событие и выполняет дополнительные действия в ходе аудита. Например, обработчик события может послать сигнальное сообщение администратору. Когда обработчик не задан, только запись о событии аудита вставляется в журнал аудита. Когда же обработчик задан, кроме вставки записи о событии аудита выполняется обработчик события.

Запись (вход; entry) события аудита содержит название политики, имя пользователя, выполнившего команду SQL, а также саму команду SQL и ее связанные переменные.

Обработчик событий определяется двумя аргументами:

- схема, содержащая программную единицу PL/SQL;
- имя программной единицы PL/SQL.

Пример содержит ссылку на процедуру SECURE.LOG_EMPS_SALARY в качестве обработчика прерываний:

```
handler_schema => 'secure'
handler_module => 'log_emps_salary'
```

По умолчанию информация о тексте и связанных переменных SQL (SQL_TEXT и SQL_BIND) всегда пишется в журнал в столбцы с типом LOB. Поведение по умолчанию может быть изменено (например, если в системе наблюдается деградация производительности).

- Статус

Статус определяет, включена ли политика FGA.

Пример:

```
enable => TRUE
```

Указания по аудиту команд DML

В соответствие с определенной для команд DML политикой FGA аудит регистрирует команду DML, когда обрабатываемые строки (как старые, так и новые) удовлетворяют условию, заданному в предикате политики.

Однако, если в политике определены также наблюдаемые столбцы, аудит регистрирует команду, если данные удовлетворяют условию, заданному в предикате политики FGA и если в команде есть ссылки на наблюдаемые столбцы.

Для команд DELETE не имеет смысла задавать наблюдаемые столбцы при определении политики, поскольку команда DELETE затрагивает все столбцы таблицы. Поэтому аудит команды DELETE выполняется всегда независимо от столбцов.

FGA позволяет вести наблюдение за командами MERGE. При выполнении такого аудита регистрируются базовые команды INSERT и UPDATE, когда они удовлетворяют политикам FGA, определенным для команд INSERT или UPDATE.

Если политику FGA, приведенную выше, переопределить для команды UPDATE, тогда первая из приведенных ниже команд будет регистрироваться аудитом, а вторая нет.

```
1)UPDATE hr.employees
SET salary = 10
WHERE commission_pct = 0.25;

2)UPDATE hr.employees
SET salary = 10
WHERE employee_id = 111;
```

Создайте политику расширенного аудита (пользователь SYS):

```
begin dbms_fga.add_policy (
 object_schema      =>   'hr',
 object_name        =>   'employees',
 policy_name        =>   'audit_emps_salary',
 audit_condition    =>   'department_id=80',
 audit_column       =>   'salary,commission_pct',
 enable             =>   TRUE,
 statement_types    =>   'update' ); end;
```

Добавьте прав пользователю ora1 на изменение таблицы EMPLOYEES пользователя HR.

От имени ora1 выполните запросы на обновление таблицы EMPLOYEES пользователя HR:

```
UPDATE hr.employees SET salary = 700 WHERE commission_pct = 0.25;
UPDATE hr.employees SET salary = 700 WHERE employee_id = 111;
commit;
```

В представлении DBA_FGA_AUDIT_TRAIL должна появиться только одна запись о первом обновлении (для второго не выполняется условие 'department_id=80').
Просмотрите это представление средствами SQL Developer.

От имени SYS сделайте выборку из этого представления.
Select DB_USER, TIMESTAMP, SQL_TEXT from DBA_FGA_AUDIT_TRAIL where OBJECT_NAME = 'EMPLOYEES';

При использовании FGA для наблюдения за командами SELECT, регистрируется сама команда, но не фактически выбираемые строки. Однако, используя совместно FGA и Flashback Query, можно воспроизвести строки в том виде, в котором они существовали в момент выборки данных.

- Для аудита всех команд используется неопределенной (null) условие.
- Имена политик должны быть уникальными.
- Наблюдаемые таблица или представление должны существовать в момент создания политики.
- Если в синтаксисе условия аудита есть ошибки, во время доступа к наблюдаемому объекту возбуждается исключение ORA-28112.
- Если наблюдаемый столбец не существует в таблице, аудит строк не выполняется.
- Когда обработчик события не существует, запись о ведении аудита создается и сообщение об ошибке не возвращается.

Аудит АБД

Пользователи с привилегиями SYSDBA и SYSOPER могут подсоединяться к закрытой базе данных.

- Журнал аудита должен храниться вне БД.
- Всегда ведется аудит установления подсоединений вида as SYSDBA или as SYSOPER.
- С помощью параметра audit_sys_operations можно включить дополнительный аудит действий пользователей с привилегиями SYSDBA и SYSOPER.
- Для сопровождения журнала аудита используется параметр audit_file_dest.

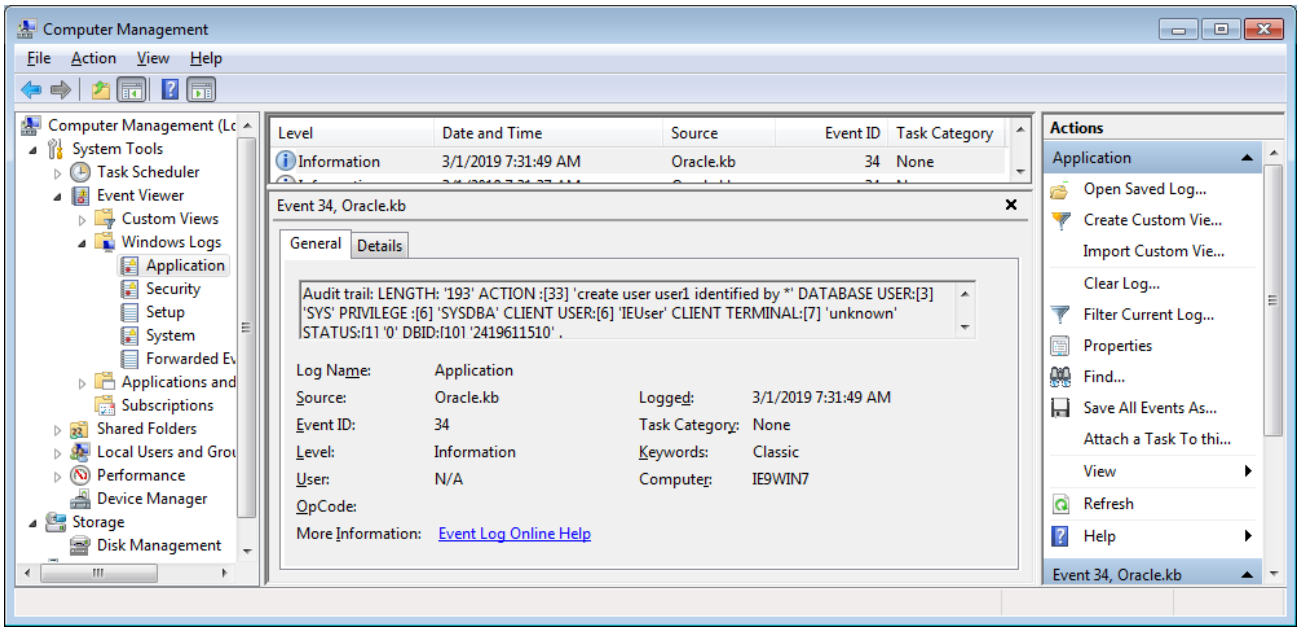
Привилегии SYSDBA и SYSOPER позволяют пользователю запускать и останавливать базу данных. Поскольку эти привилегии могут использоваться при закрытой базе данных,

журнал аудита должен храниться вне базы данных. Oracle автоматически регистрирует установление соединений с привилегиями SYSDBA и SYSOPER. Такой метод отслеживания санкционированных и несанкционированных действий с привилегиями SYSDBA и SYSOPER может предоставить полезные сведения, но только, если просматривать журнал аудита в ОС.

База данных не регистрирует ничего кроме событий входа в систему, если только аудит не включен специально. Для включения аудита пользователей с привилегиями SYSDBA и SYSOPER устанавливается параметр инициализации:

audit_sys_operations=TRUE (по умолчанию FALSE.)

Когда ведется аудит операций пользователя SYS, параметр инициализации audit_file_dest контролирует месторасположение записей аудита. На платформе Windows журнал аудита по умолчанию располагается в *журнале событий* (*event log*) Windows. На платформах UNIX и Linux записи аудита сохраняются в $ORACLE_HOME/rdbms/audit.



Параметр audit_sys_operations был включен в задании на стр.5.

Найдите в журналах Windows записи о действиях администратора БД.

Сопровождение журнала аудита

Необходимо сопровождать каждый тип журнала аудита. Основные действия по сопровождению включают просмотр и удаление старых записей из базы данных и операционной системы. Журнал аудита может вырасти и заполнить доступное пространство хранения. Полное заполнение файловой системы может привести к проблемам производительности или аварийному отказу системы. Если журнал аудита в базе данных полностью заполнит табличное пространство, действия, для которых ведется аудит, не завершаются. Полное заполнение журналом аудита табличного пространства system влияет на производительность других операций еще до остановки операции аудита.

Журнал стандартного аудита хранится в таблице AUD$. Журнал FGA находится в таблице FGA_LOG$. По умолчанию обе эти таблицы созданы в табличном пространстве SYSTEM. Эти таблицы можно переместить в другое табличное пространство с помощью утилит экспорта и импорта.

Примечание: перемещение таблиц аудита за пределы табличного пространства SYSTEM не поддерживается.

Записи аудита могут быть потеряны при удалении строк из таблиц аудита.

Подсказка по наилучшему практическому приему работы: используйте экспорт на основе временной метки, а затем удаляйте строки из журнала аудита, основываясь на той же самой временной метке.

Время на выполнение лабораторной работы – 2 часа.