1. Что такое курсор?

Курсор – это средство извлечения данных из базы данных Oracle.

* Операторы INSERT, UPDATE, DELETE, MERGE, SELECT INTO – неявные курсоры.  выполнении INSERT, UPDATE и т.д. в "глобальной системной области", (применительно к Oracle) - всегда, открывается "курсор"

1. Что такое точная и неточная выборки?

Точная выборка – выборка, возвращающая одну строку.

Неточная выборка – выборка, возвращающая несколько строк.

1. Объясните действие конструкций WHEN OTHERS, WHEN TO\_MANY\_ROWS, WHEN NO\_DATA\_FOUND в секции исключения.

Когда неявный курсор не возвращает строк вообще, PL/SQL генерирует исключение NO\_DATA\_FOUND и передает управление в секцию исключений

Когда SELECT возвращает более одной строки, PL/SQL генерирует исключение TOO\_MANY\_ROWS и также передает управление в секцию исключений

**WHEN OTHERS**: Эта конструкция обрабатывает все исключения, которые не совпадают с явно указанными в блоке обработки исключений. Она действует как "поймать все", обрабатывая любые исключения, которые не были обработаны другими конструкциями WHEN.

1. Объясните назначение функций SQLERRM и SQLCODE.

Функция **SQLERRM** возвращает сообщение об ошибке, связанной с исключительной ситуацией.

Функция SQLCODE возвращает номер ошибки, связанной с исключительной ситуацией.

1. Что такое атрибут курсора? Перечислите все атрибуты курсора и объясните их назначение.

**Атрибуты курсора** — это свойства, которые позволяют получить информацию о текущем состоянии курсора в PL/SQL.

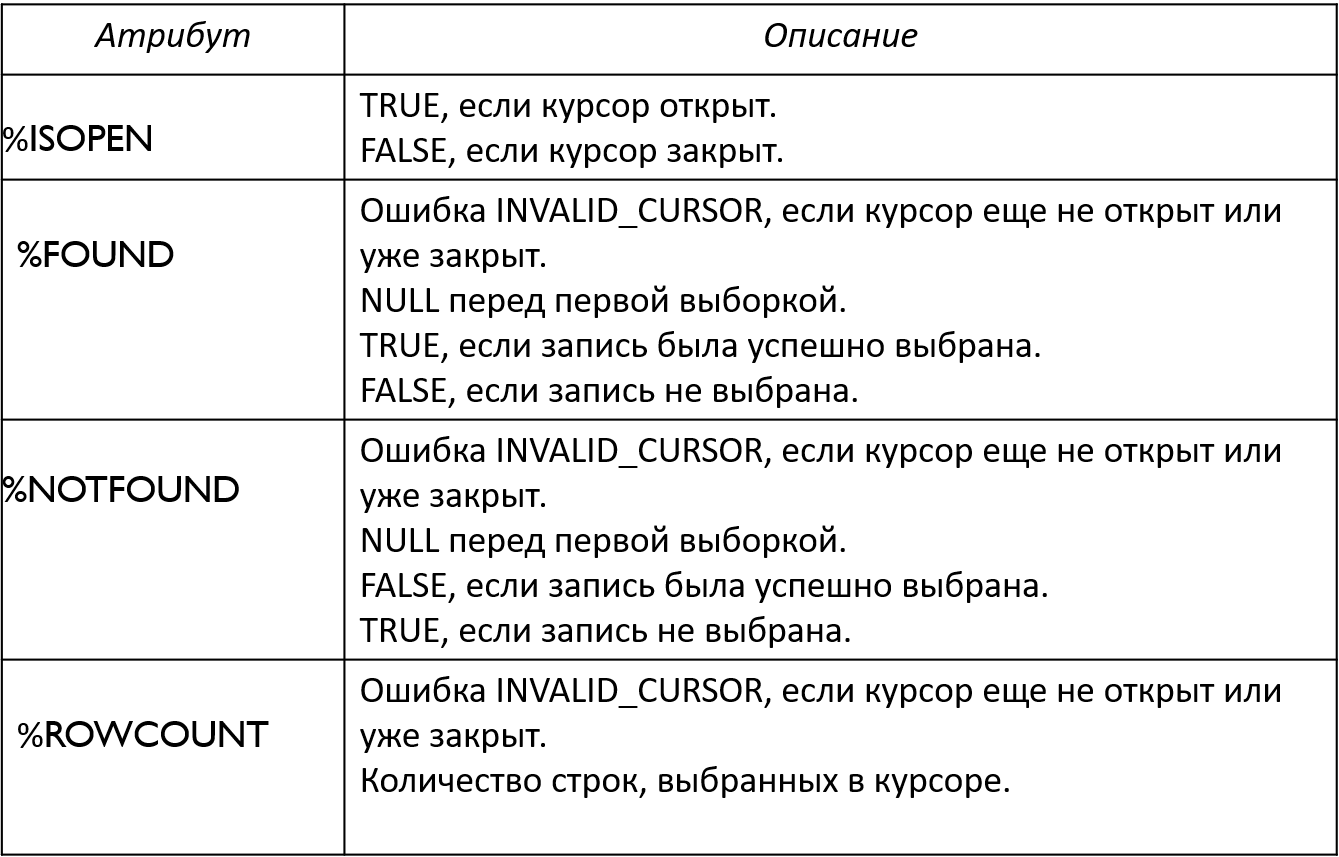
**%ISOPEN — возвращает значение TRUE, если курсор открыт**

**%FOUND — определяет, найдена ли строка, удовлетворяющая условию**

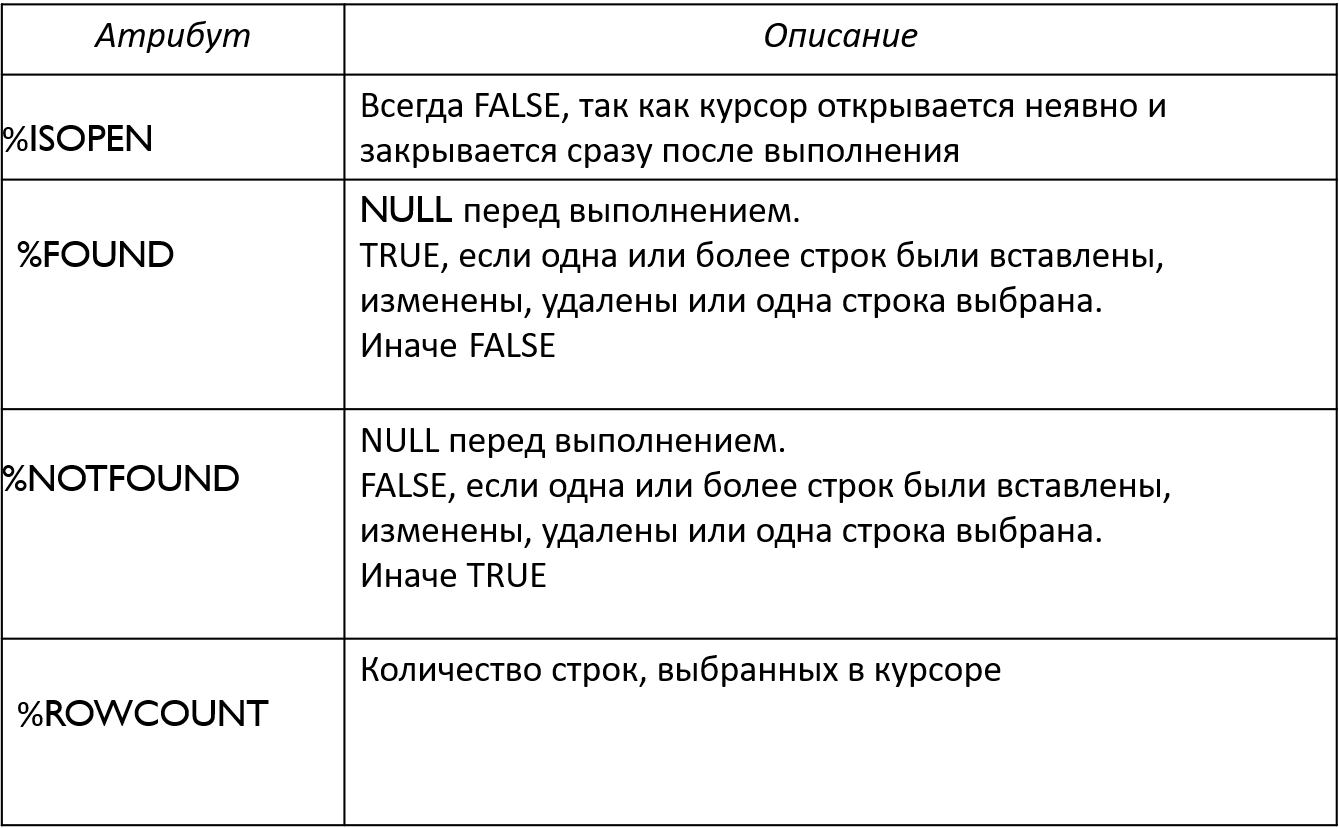
**%NOTFOUND — возвращает TRUE, если строка не найдена**

**%ROWCOUNT — возвращает номер текущей строки**

**Явные:**



**Неявные:**



1. Объясните назначение конструкции RETURNING в операторах INSERT, DELETE, UPDATE.

**RETURNING** в операторах INSERT, DELETE и UPDATE в SQL используется для получения значений столбцов, которые были изменены, вставлены или удалены в результате выполнения соответствующей операции.

1. В чем отличие явного и неявного курсоров.

Неявные курсоры - простой оператор SELECT ... INTO извлекает одну строку данных непосредственно в переменные локальной программы. Это удобный (и часто наиболее эффективный) способ доступа к данным, использование которого, однако, может приводить к необходимости повторного кодирования оператора SELECT (или похожих операторов) в нескольких местах программы.

Явные курсоры - вы можете явно объявить курсов в разделе объявлений (локального блока или пакета). В этом случае курсор можно будет открывать и извлекать данные в одной или нескольких программах, причем возможности контроля будут шире, чем при использовании неявных курсоров.

1. Объясните схему работы с явным курсором в PL/SQL-блоке.

Сначала необходимо объявить курсор в PL/SQL блоке с помощью ключевого слова CURSOR, указав запрос, который будет выполняться при открытии курсора. Курсор можно объявить как с параметрами, так и без них.

|  |
| --- |
| CURSOR cursor\_name IS  SELECT column1, column2  FROM table\_name  WHERE condition; |

После объявления курсора его нужно открыть с помощью оператора OPEN. Это действие запускает выполнение запроса, и курсор становится активным.

OPEN cursor\_name;

После открытия курсора можно использовать циклы или операторы FETCH для извлечения данных из курсора по одной строке за раз. Данные могут быть присвоены переменным или использованы для выполнения каких-либо действий.

|  |
| --- |
| LOOP  FETCH cursor\_name INTO variable1, variable2;  EXIT WHEN cursor\_name%NOTFOUND;  -- Выполнение действий с данными  END LOOP; |

По завершении работы с курсором его следует закрыть с помощью оператора CLOSE. Это освободит ресурсы, занятые курсором, и завершит его работу.

1. Что происходит по команде OPEN курсора?

При вызове OPEN происходит инициализация запроса, связанного с курсором.

После инициализации запроса он отправляется к базе данных для выполнения. База данных обрабатывает запрос и возвращает первую порцию данных, удовлетворяющих условиям запроса.

После открытия курсора текущая позиция курсора устанавливается на первую строку результирующего набора данных. Таким образом, курсор готов начать извлечение данных с первой строки.

Первая порция данных из результирующего набора загружается в буфер курсора, и они могут быть использованы для последующей обработки.

1. Объясните особенность применения FOR-цикла при работе с явным курсором.

Сначала необходимо объявить курсор в PL/SQL блоке, как это делается обычно.

CURSOR cursor\_name IS

SELECT column1, column2

FROM table\_name

WHERE condition;

Затем курсор может быть использован внутри конструкции FOR-цикла для автоматической итерации по результирующему набору данных. В этом случае, после слова FOR, указывается итератор, переменные, в которые будут записываться значения из результирующего набора, и ключевое слово IN, за которым следует курсор.

FOR record IN cursor\_name LOOP

-- Обработка данных, например:

DBMS\_OUTPUT.PUT\_LINE('Column1: ' || record.column1 || ', Column2: ' || record.column2);

END LOOP;

При использовании FOR-цикла PL/SQL автоматически открывает курсор перед началом цикла и закрывает его после завершения работы с ним. Это упрощает код и обеспечивает автоматическое управление ресурсами. При проходе явного курсора при помощи for-цикла создаётся переменная при начале описания цикла, которая будет содержать строку,

берущуюся из курсора на каждой итерации цикла for

1. Для чего применяются параметры курсора? Где они указываются?

**Параметры** **курсора** используются для передачи значений в запрос, который используется в курсоре, что позволяет делать запросы более динамическими и универсальными. Они позволяют использовать один и тот же запрос с разными наборами параметров без необходимости изменения самого запроса. Параметры курсора указываются в объявлении курсора.

DECLARE

CURSOR cursor\_name (param1 VARCHAR2, param2 NUMBER) IS

SELECT column1, column2

FROM table\_name

WHERE condition\_column = param1

AND another\_condition\_column = param2;

1. Что такое курсорная переменная? Как ее можно объявить?

**Курсорная переменная** в PL/SQL представляет собой переменную, которая содержит ссылку на курсор. Это позволяет создавать динамические курсоры и управлять ими во время выполнения программы. Курсорная переменная объявляется с использованием специального типа данных REF CURSOR в PL/SQL.

1. **Что значит курсорный подзапрос?**

Курсорный подзапрос – это тоже курсор (второго уровня), указываемый при создании другого курсора (первого уровня), позволяющий обрабатывать отдельные значения результирующего набора курсора первого уровня в виде курсора второго уровня.

**Курсорный подзапрос** представляет собой подзапрос, который внешнему запросу возвращает курсор, а не просто набор значений. Этот курсор может затем использоваться внешним запросом для обработки данных.

Основная идея курсорного подзапроса состоит в том, что каждая строка, возвращаемая внешним запросом, вызывает выполнение подзапроса, который затем возвращает курсор для этой строки. Таким образом, каждая строка внешнего запроса имеет свой собственный набор данных, возвращаемый подзапросом.

1. **Объясните назначение конструкции CURRENT OF.**

CURRENT OF в SQL используется для обновления или удаления текущей строки, на которой находится курсор. Это позволяет обращаться к строке в результате курсора без явного указания условий в WHERE-предложении. Обычно конструкция CURRENT OF используется в командах UPDATE и DELETE, чтобы указать базе данных, что операция должна быть применена к текущей строке, на которой находится курсор. Оператор WHERE CURRENT OF позволяет обновить или удалить запись, которая была в курсоре последней

1. **Для чего применяется псевдостолбец ROWID? Поясните его структуру.**

ROWID — это псевдостолбец, который является уникальным идентификатором строки в таблице и фактически описывает точное физическое расположение данной конкретной строки. На основе этой информации Oracle впоследствии может найти данные, связанные со строкой таблицы. При каждом перемещении, экспорте, импорте строки, а также при выполнении любых других операций, которые приводят к изменению ее местонахождения, изменяется ROWID строки, поскольку она занимает другое физическое положение. Для хранения данных ROWID требуется 80 бит (10 байт). Идентификаторы ROWID состоят из четырех компонентов: номера объекта (32 бита), относительного номера файла (10 бит), номера блока (22 бита) и номера строки (16 бит). Эти идентификаторы отображаются как 18-символьные последовательности, указывающие местонахождение данных в БД, причем каждый символ представлен в формате base-64, состоящем из символов A-Z, a-z, 0-9, + и /. Первые шесть символов – это номер объекта данных, следующие три – относительный номер файла, следующие шесть – номер блока, последние три – номер строки.

1. **Для чего применяется псевдостолбец ROWNUM?**

ROWNUM - логический номер записи в запросе

Псевдостолбец ROWNUM в SQL используется для ограничения количества возвращаемых строк в результирующем наборе данных. Он присваивает каждой возвращенной строке уникальный номер в порядке их выборки из таблицы или представления.

1. Объясните разницу между функциями floor(), ceil(), round() и trunc().

1. floor(число):

Назначение: Округляет число вниз до ближайшего целого числа.

2. ceil(число):

Назначение: Округляет число вверх до ближайшего целого числа.

3. round(число, [количество\_знаков]):

Назначение: Округляет число до указанного количества знаков после запятой. Если количество знаков не указано, округляет до ближайшего целого числа.

4. trunc(число, [количество\_знаков]):

Назначение: Округляет число, отбрасывая все цифры после запятой (если количество знаков не указано) или после указанного количества знаков.

1. Перечислите встроенные функции для работы с датами, исследованные в лабораторной работе.
2. Перечислите встроенные функции конвертации, исследованные в лабораторной работе.
3. Объясните назначение встроенных функций sqlcode и sqlerrm.
4. Поясните принцип обработки исключений в PL/SQL.
5. **REGEXP\_SUBSTR(строка, выражение, начало, [появление])**: Эта функция извлекает подстроку из строки, соответствующую регулярному выражению. Она принимает строку, регулярное выражение, начальную позицию поиска и, при необходимости, номер появления (если выражение встречается несколько раз в строке).
6. **SUBSTR(строка, начало, длина)**: Эта функция извлекает подстроку из строки, начиная с указанной позиции и заданной длины.

**EXTRACT(MONTH FROM SYSDATE)** - это функция, используемая в SQL для извлечения месяца из текущей даты

TO\_CHAR(SYSDATE, 'D') – день недели

TO\_CHAR(SYSDATE, 'DAY') – текстовое представление дня недели

TO\_CHAR(SYSDATE, 'DD') – день месяца

**add\_months(SYSDATE, 12)**: Эта функция добавляет 12 месяцев к текущей дате (**SYSDATE**)

**months\_between(add\_months(SYSDATE, 12), BIRTHDAY)**: Это выражение вычисляет разницу в месяцах между датой, которая наступит через 12 месяцев, и датой рождения (**BIRTHDAY**).

ROLLUP — это функция, расширяющая функционал GROUP BY. Она создаёт иерархическую группировку данных, добавляя итоговые строки на каждом уровне группировки.1

В данном случае ROLLUP применяется к двум столбцам: PULPIT.FACULTY и PULPIT.PULPIT. Это создаёт 3 уровня группировки:

Группировка по факультету и кафедре: Это базовый уровень, где вычисляется средняя зарплата для каждой комбинации факультета и кафедры.

Группировка только по факультету: ROLLUP добавляет строки, где PULPIT.PULPIT принимает значение NULL. Это означает, что данные агрегируются по каждому факультету, игнорируя конкретную кафедру.

Общая сумма: ROLLUP также добавляет строку, где и PULPIT.FACULTY, и PULPIT.PULPIT имеют значение NULL. Это представляет собой итоговую строку с общей средней зарплатой по всем факультетам и кафедрам.

1. Что такое локальные процедуры и функции PL/SQL?

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

Про перегрузку:

* 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

1. Чем отличаются процедуры от функций?



1. Что такое хранимые процедуры и функции?

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

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

1. Что такое параметрическая, позиционная и смешанная формы передачи параметров?

Позиционный – каждое значение в списке аргументов вызова ставится в соответствие формальному параметру по порядку.

*Empid\_to\_name(23, name, surname);*

Именованный – явно связывает аргументы при вызове с параметрами по именам. *Empid\_to\_name(in\_id =>23, out\_name=> name, out\_surname =>surname);*

Можно комбинировать оба метода, пока позиционные аргументы стоят слева.

*Empid\_to\_name(23, name, out\_surname =>surname);*

1. Что такое пакет?

Пакеты - коллекция PL/SQL объектов, сгруппированных вместе.

1. Что может входить в пакет?

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

1. Что значит «пакет сохраняет состояние переменных, объявленных в спецификации, на время сессии»?

Спецификация пакета (package) – обязательна, содержит список объектов для общего доступа из других модулей или приложения

Реализация пакета (package body) – содержит весь программный код для реализации процедур и функций и спецификации, приватные объекты и секцию инициализации

Состоянием пакета (package state) называется совокупность текущих значений его глобальных и локальных переменных и констант, а также текущих состояний курсоров (курсор открыт или закрыт, а также значения атрибутов курсора, отражающие в том числе сколько строк было считано из открытого курсора к настоящему времени).

Состояния пакетов сохраняется в течение всей жизни сессии пользователя. Очень важно то, что состояния пакетов у каждой сессии пользователя свое. Нельзя «подсмотреть» или «подправить» значения пакетных переменных или считать строку курсора из состояния пакета другой сессии. Во многих приложениях на этом построены специфические системы управления доступом.

1. Перечислите типы триггеров, поддерживаемых Oracle.

**Триггер –** особый вид процедур, которые срабатывают по запускающему их событию

**По привязанному объекту:**  
На таблице

На представлении - instead of trigger

**По событиям запуска:**  
Вставка записей - insert

Обновление записей - update

Удаление записей - delete

**По области действия:**   
Уровень оператора - statement level triggers

Уровень записи - row level triggers

Составные триггеры - compound triggers

**По времени срабатывания:**  
Перед выполнением операции – before

После выполнения операции - after

1. Поясните правило: триггер является частью транзакции.

По умолчанию триггеры DML участвуют в транзакциях, из которых они запущены. Это означает, что:

* если триггер инициирует исключение, будет выполнен откат соответствующей части транзакции;
* если триггер сам выполнит команду DML (например, вставит запись в таблицу- журнал), она станет частью главной транзакции;
* в триггере DML нельзя выполнять команды COMMIT и ROLLBACK.

Если триггер DML определен как [автономная транзакция PL/SQL](https://oracle-patches.com/db/sql/3544-%D0%B0%D0%B2%D1%82%D0%BE%D0%BD%D0%BE%D0%BC%D0%BD%D1%8B%D0%B5-%D1%82%D1%80%D0%B0%D0%BD%D0%B7%D0%B0%D0%BA%D1%86%D0%B8%D0%B8), то все команды DML, выполняемые внутри триггера, будут сохраняться или отменяться (командой COMMIT или ROLLBACK) независимо от [основной транзакции](https://oracle-patches.com/db/sql/3543-%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5-%D1%82%D1%80%D0%B0%D0%BD%D0%B7%D0%B0%D0%BA%D1%86%D0%B8%D1%8F%D0%BC%D0%B8-%D0%B2-pl-sql).

1. Перечислите привилегии необходимые для работы с триггерами.

**CREATE TRIGGER** - создавать, удалять, изменять в своей подсхеме

**CREATE ANY TRIGGER** - создать любой триггер в любой схеме, кроме SYS, не рекомендуется для словаря, не разрешает менять текст триггера

**ALTER ANY TRIGGER** - разрешать, запрещать, изменять, компилировать, любые, кроме SYS-триггеров, триггеры

**DROP ANY TRIGGER** - удалять любой триггер, кроме SYS-триггеров

**ADMINISTER DATABASE TRIGGER** - создавать, изменять, удалять системные триггеры, должен иметь привилегию CREATE TRIGGER или CREATE ANY TRIGGER

* Назначаются напрямую USERу, а не через роль

1. Перечислите события, на которые могут срабатывать DML-триггеры.

|  |  |
| --- | --- |
| INSERT | Событие возникает, когда добавляется строка (строки) в таблицу или представление. |
| UPDATE | Событие возникает, когда выполняется операция UPDATE над данными в таблице или представлении.  Можно дополнительно задавать выражение OF для указания полей, при изменении которых срабатывает триггер. |
| DELETE | Событие возникает, когда удаляется строка (строки) из таблицы или представления.  Триггер не срабатывает при выполнении команды TRUNCATE table. |

1. Поясните принцип применения INSTEADOF-триггеров в Oracle.

**Триггеры замещения** (ins­tead of) можно создавать **только для представлений**. В отличие от триггеров DML, которые выполняют­ся в дополнение к операторам DML, триггеры замещения выполняются вместо операторов DML, вызывающих их срабатывание. Триггеры заме­щения должны быть **строковыми триггерами**.

* FOR EACH ROW (для каждой строки) - срабатывает для каждой измененной строки
* ПО УМОЛЧАНИЮ (операторный уровень) - срабатывает один раз на тригтерное событие

1. Что такое секционирование таблиц?

**Секционирование** (partitioning) — это способ логического разделения крупной таблицы на более мелкие части для облегчения обработки запросов, операций DML и управления базой данных. Секционирование приводит к повышению производительности потому, что базе данных в ответ на запрос приходится выполнять поиск только в определенных разделах таблицы.

1. В каких случаях целесообразно применять секционирование?

• При управлении большими объемами данных.

• Для оптимизации операций загрузки и выгрузки данных.

• Для улучшения управляемости и обработки часто обновляемых данных.

1. Объясните принцип секционирования для всех типов секционирования, которые использовались в заданиях лабораторной работы.

**Секционирование по диапазонам ключей** (ключи - стобцы секционирования)

Диапазонное секционирование используется для **данных, которые разделяются на диапазоны на основе некоторого критерия**.

При секционировании по диапазонам принято последним устанавливать раздел “остальные” (catchall). Когда такой раздел создается, он содержит значения, меньшие максимального (**maxvalue**), которое просто является значением, большим значений в предпоследнем разделе.

**Интервальное секционирование**

Интервальное секционирование — это расширение традиционного метода секционирования по диапазону ключей. Чтобы реализовать интервальное секционирование, сначала потребуется специфицировать минимум один диапазонный раздел таблицы. Используете вы минимальный однодиапазонный раздел или многодиапазонные разделы, максимальное значение ключа диапазонного секционирования называется **точкой перехода** (transition point). После того, как данные пересекают точку перехода, база данных автоматически создает интервальные разделы.

Ниже перечислено, что следует знать об интервальном секционировании.

* Чтобы создать таблицу, секционированную по интервалам, укажите в операторе CREATE TABLE **конструкцию INTERVAL**.
* Прежде чем специфицировать интервальные разделы, определите, как минимум, один диапазонный раздел с помощью **конструкции PARTITION.**
* Можно использовать ключ секционирования, который включает более одного столбца.
* Ключ секционирования должен быть типа **NUMBER или DATE**.

**Хеш-секционирование**

Все, что потребуется сделать — выбрать количество разделов, и алгоритм хеширования Oracle назначит хеш-значение ключу раздела каждой строки, после чего поместит ее в соответствующий раздел. Вам ничего не нужно знать о распределении данных в таблице, помимо того, что данные не должны попадать в один, легко определяемый диапазон. Все, что необходимо предоставить — это ключ раздела:

**Секционирование по списку значений ключа**

Секционирование по списку значений ключа, или списковое секционирование, —предпочтительный способ по отношению к диапазонному или хеш-секционированию,когда данные распределены среди множества **дискретных значений**. Например, может потребоваться группирование данных о продажах компании по регионам, а не по кварталам.

1. Перечислите названия типов секционирования, которые не использовались в заданиях лабораторной работы.

Если две таблицы связаны друг с другом, можно воспользоваться преимуществом этого отношения, выполнив секционирование этих двух таблиц на основе **существующего отношения “родительский–дочерний”.** Это отношение задается ограничениями первичного и внешнего ключа. Если две таблицы разделяют отношение “родительский–дочерний”, нужно лишь **формально секционировать родительскую таблицу**. Тем самым **исключается дублирование ключевых столбцов**. Любые операции обслуживания разделов на родительской таблице автоматически распространятся на дочернюю таблицу.

Ниже приведен простой пример для прояснения концепции ссылочного секционирования. Таблицы SalesR и SalesR\_details связаны друг с другом на основе столбца fact\_id из обеих таблиц. Это отношение зафиксировано ссылочным ограничением sales\_details\_fk. Родительская таблица — SalesR — секционирована по столбцу time\_id с использованием схемы диапазонного секционирования.

Системное секционирование — уникальный метод секционирования таблиц, при котором расположением данных управляет приложение, а не база данных. База данных просто позволяет разделить таблицу на разделы, не имея никакого представления относительно того, что будет содержать каждый из разделов. Приложение управляет тем,что попадает в каждый раздел. При вставке данных в таблицу с системным секционированием **необходимо явно специфицировать раздел**. Таким образом, если попытаться вставить данные в таблицу с системным секционированием, не указывая раздела, в который эти данные следует поместить, то вставка закончится неудачей.

Иногда простого секционирования по диапазону, хешу или списку оказывается недостаточно. Для более тонкого контроля над размещением данных можно выполнить дальнейшее секционирование крупной таблицы на подразделы. Oracle предлагает несколько типов составного секционирования.

• Секционирование по виртуальному столбцу: Этот тип секционирования использует виртуальные столбцы, которые вычисляются на основе значений других столбцов. Секционирование по виртуальному столбцу позволяет определять секции с использованием значений, вычисленных на лету из других столбцов.

1. Объясните действие оператора ALTER TABLE MERGE.

Оператор ALTER TABLE MERGE используется для объединения смежных секций в одну большую секцию в секционированной таблице.

Все данные из двух объединяемых секций переносятся в одну новую секцию, а затем оригинальные секции удаляются.

1. Объясните действие оператора ALTER TABLE SPLIT.

Оператор ALTER TABLE SPLIT используется для разделения одной секции таблицы на две или более новых секции.

Оператор ALTER TABLE SPLIT выбирает существующую секцию в таблице для разделения на две или более новых секции. Для каждого нового раздела определяется граница значений ключевого столбца, на основе которой произойдет разделение данных из исходной секции. Данные из исходной секции перераспределяются между новыми секциями в соответствии с их границами значений ключевого столбца.

1. Объясните действие оператора ALTER TABLE EXCHANGE.

Оператор ALTER TABLE EXCHANGE используется для обмена содержимым секции с содержимым некоторой внешней таблицы. Это позволяет перемещать данные между таблицами, сохраняя структуру и связи секций в секционированных таблицах.

Оператор ALTER TABLE EXCHANGE выбирает одну секцию из секционированной таблицы и одну внешнюю таблицу для обмена данными. Данные из выбранной секции перемещаются во внешнюю таблицу, а данные из этой внешней таблицы перемещаются в выбранную секцию. Важно, чтобы структура обоих таблиц (типы столбцов, индексы, ограничения) была совместима перед выполнением оператора ALTER TABLE EXCHANGE.