**23. Объекты и концепции базы данных. Курсоры. Назначение и область применения.**

Особенности работы

Курсор — ссылка на контекстную область памяти. В некоторых реализациях языка программирования SQL (Oracle, Microsoft SQL Server) — получаемый при выполнении запроса результирующий набор и связанный с ним указатель текущей записи. бы сказал, что курсор — это виртуальная таблица которая представляет собой альтернативное хранилище данных. При этом курсор, позволяет обращаться к своим данным, как к данным обычного массива.

Курсоры прекрасно поддерживаются в хранимых процедурах, функциях и триггерах.

Синтаксис такой же, как и во внедренном SQL. Курсоры пока только для чтения, однонаправленные (т.е по набору можно ходить только вперед без возможности вернуться) и невосприимчивы. Невосприимчивость означает, что сервер может создавать копию результатирующей таблицы, а может и не создавать, формируя ее на лету .

Курсоры должны быть объявлены до их использования. Переменные с условиями объявляются прежде курсоров. Обработчики объявляются строго после объявления курсоров.

Объявление курсоров

DECLARE cursor\_name CURSOR FOR select\_statement

Это выражение объявляет курсор c именем cursor\_name. select\_statement указывает на конструкцию типа SELECT ... FROM ... Можно объявить много курсоров в подпрограмме, но каждый курсор в данном блоке должен иметь уникальное имя. Выражение SELECT не должно содержать указание INTO.

Открывание курсоров

OPEN cursor\_name

Выражение открывает ранее объявленный курсор

Выборка из курсора в переменную

FETCH cursor\_name INTO var\_name [, var\_name] ...

Это выражение выбирает следующую строку (если строка существует), используя указанный открытый курсор, и продвигает указатель курсора. Если более строк не доступно, происходит изменение значения переменной SQLSTATE в 02000. Для отлова этого события вы должны установить обработчик: HANDLER FOR SQLSTATE '02000'

Закрытие курсора

CLOSE cursor\_name

Закрывает курсор cursor\_name. Если явно не указано, то курсор закрывается автоматически при закрытии соответствующего блока подпрограммы.

Как использовать курсоры

Применение курсора в процедурах осуществляется путем последовательного выполнения следующих шагов:

При помощи оператора DECLARE объявляется курсор для отдельного оператора SELECT или для отдельной процедуры.

Оператором OPEN производится открытие курсора.

Используя оператор FETCH, осуществляется установление указателя на требуемую запись курсора. При этом значения полей текущей записи присваиваются переменным, указываемым в операторе FETCH. Обычно это конструкция помещается в итеративный элемент (проще говоря цикл), который прерывается по некоторому условию. См. пример выше.

В процессе перемещения указателя текущей записи курсора при выходе указателя за пределы курсора устанавливается значение SQLSTATE = 02000.

После того как курсор становится ненужным, он закрывается оператором CLOSE.

Примеры курсоров

Приведу еще один пример курсора. Курсор предназначен для выборки данных (идентификаторов записей) в строку с разделителем ввиде запятой по переданным параметрам. Курсор находится внутри хранимой функции get\_pedplan(). В нее передается три параметра: lip-номер лаборатории, ti-номер пары и dt - дата проведения занятия.

01: CREATE DEFINER = 'for\_spammers'@'zoonman.com' FUNCTION `get\_pedplan`(lip INTEGER(11), ti INTEGER(11), dt DATE)

02: RETURNS char(64) CHARSET latin1

03: DETERMINISTIC

04: CONTAINS SQL

05: SQL SECURITY INVOKER

06: COMMENT 'Функция возвращает список id из таблицы raspisanie'

07: BEGIN

08:

09: DECLARE done INT DEFAULT 0;

10: DECLARE a INT;

11: DECLARE retv CHAR(64);

12: DECLARE flg INT;

13:

14: DECLARE cur1 CURSOR FOR SELECT id FROM raspisanie WHERE timeintv=ti AND rdate=dt AND labip=lip ;

15: DECLARE CONTINUE HANDLER FOR SQLSTATE '02000' SET done = 1;

16: OPEN cur1;

17: SET retv:='';

18: SET flg:=0;

19: REPEAT

21: FETCH cur1 INTO a;

22: IF NOT done THEN

23: IF flg!=0 THEN

24: SET retv:= CONCAT(retv,',' , a);

25: ELSE

26: SET retv:=a;

27: SET flg:=1;

28: END IF;

29: END IF;

30: UNTIL done END REPEAT;

31: CLOSE cur1;

32: RETURN retv;

33: END;

Курсор определен на строке 14. Открыт на 16 строчке. С 19-й начат проход по выборке полученной курсором. На каждом шаге цикла происходит считывание записи (21-я). Затем, если не достигнут конец выборки (22-я), выполняется проверка флага на первую запись (23-я). Если запись первая, то присваиваем retv текущее значение выборки (26-я) и устанавливаем флаг (27-я), иначе объединяем значение retv с текущим значением выборки (строка 24). После прохода по курсору закрываем его (31-я) и возвращаем значение (32-я строка).

Начиная с 5-ой версии, MySQL имеет возможность создавать курсоры. Курсор — это виртуальная таблица, предоставляющая альтернативный способ просмотра табличных данных. Курсор не содержит данных, а лишь ссылается на них из других таблиц. Из этого вытекает ряд особенностей, но об этом ниже.

Обычный запрос выборки из таблицы выглядит примерно так:

SELECT `id`, `name`, `family`, `age`, `place`, `price` FROM `girls`;

Однако не всегда удобно использовать такой длинный запрос. Гораздо проще использовать курсор:

CREATE VIEW c\_girls AS

SELECT `id`, `name`, `family`, `age`, `place`, `price` FROM `girls`;

Теперь, чтобы выбрать все записи, достаточно запроса:

SELECT \* FROM c\_girls;

Ну или для тех, кто любит помоложе

SELECT \* FROM c\_girls WHERE age < 25;

Разумеется, нельзя использовать в выборке столбцы, не включенные в курсор.

В определение курсора можно включать инструкции сортировки, группировки и т. д. Например:

CREATE VIEW c\_girls AS

SELECT `id`, `name`, `family`, `age`, `place`, `price` FROM `girls` ORDER BY age LIMIT 7;

Что мне особенно нравится, так это возможность компоновки курсора из множества таблиц.

Как уже было упомянуто, курсор – это виртуальная таблица. Это значит, что в некоторых случаях курсор не обновляем. То есть нельзя модифицировать/удалять/добавлять записи через курсор. Это справедливо только тогда, когда курсор ссылается на несколько таблиц. В этом случае нельзя четко определить, к каким таблицам относятся изменения, а к каким нет. Курсор обновляем только когда он ссылается на одну таблицу (а не на производные выражения). Впрочем в своей практике я практически никогда не пользуюсь добавлением записей через курсор. Нафига? Мой ответ – Упрощение.

Ясно, что курсор не напишет код за вас. Но он упрощает работу с объектами реального мира, описанными в базе. Мы можем создать курсор main\_book\_info с основной информацией о товаре, и курсор full\_book\_info с более полной информацией. Такой прием сокращает код и позволяет быстрее ориентироваться в нём.

Сравните:

SELECT id, (SELECT author.name FROM author) as Author\_Name…

или так:

SELECT \* FROM main\_book\_info

Можно создавать курсоры для разных частей сайта: каталог, страница с описанием, рейтинг товаров и т. д.