Транзакции

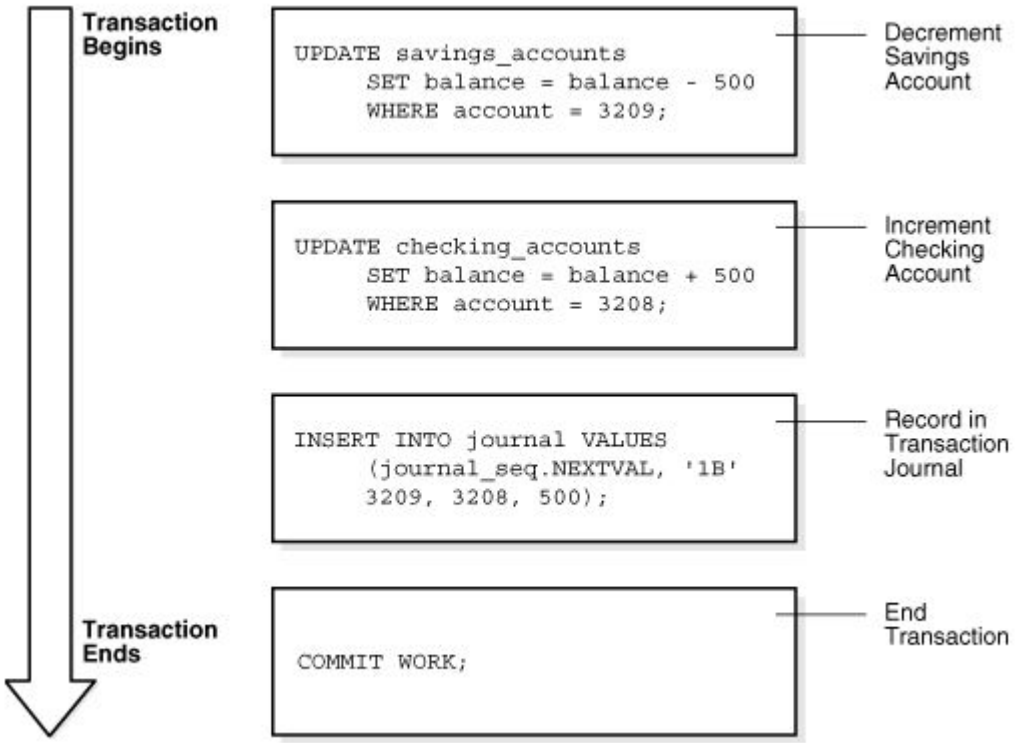
Графеева Н.Г.

2015

Определение

- Транзакция – логически неделимая последовательность операторов, переводящая базу из одного согласованного состояния в другое согласованное состояние.

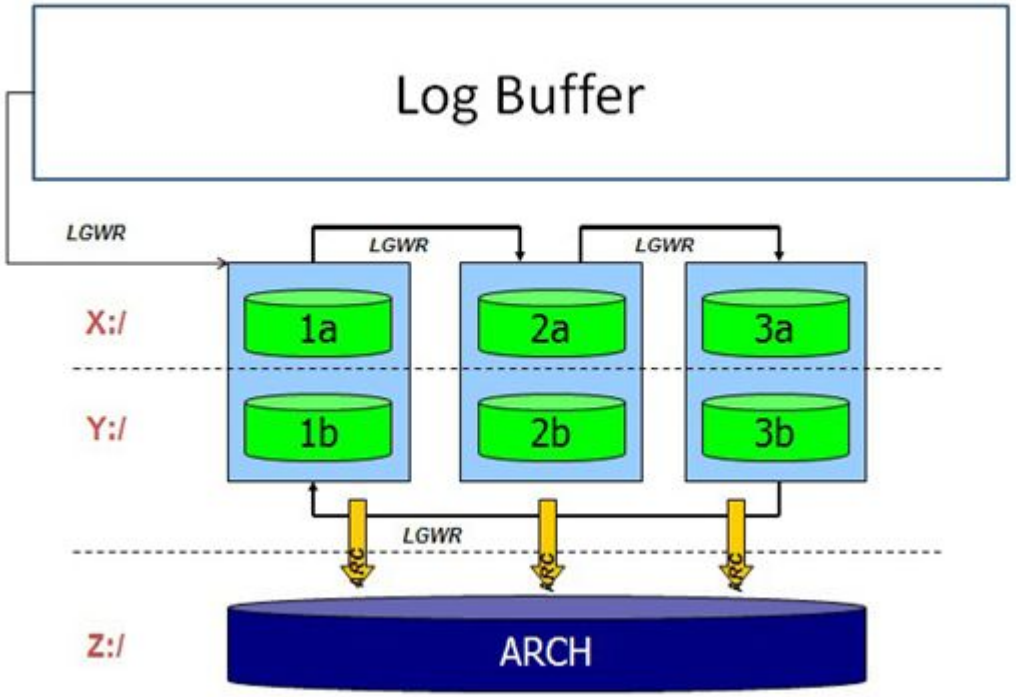
Пример(банковская транзакция)



Основные свойства транзакций (ACID)

- Неделимость (**Atomicity**). Транзакция либо выполняется полностью, либо не выполняется.
- Согласованность (**Consistency**). Транзакция переводит базу данных из одного согласованного состояния в другое.
- Изолированность (**Isolation**). Результаты транзакции становятся доступны для других транзакций только после ее фиксации.
- Продолжительность (**Durability**). После фиксации транзакции изменения становятся постоянными.

Как реализуется механизм транзакций?



Транзакции стартуют

- Автоматически после редактирования данных базы (но не во всех СУБД…).
- После явного объявления начала транзакции с помощью операторов объявления транзакции.

Транзакции завершаются

- После явного объявления о завершении транзакции соответствующим оператором.
- После завершения сеанса пользователя транзакцию фиксирует среда или инструментальное средство (но не всегда…).

Операторы управления транзакцией (ORACLE)

- COMMIT [WORK]
- ROLLBACK [WORK]
- SAVEPOINT <точка сохранения>
- ROLLBACK TO <точка сохранения>
- SET TRANSACTION

Пример

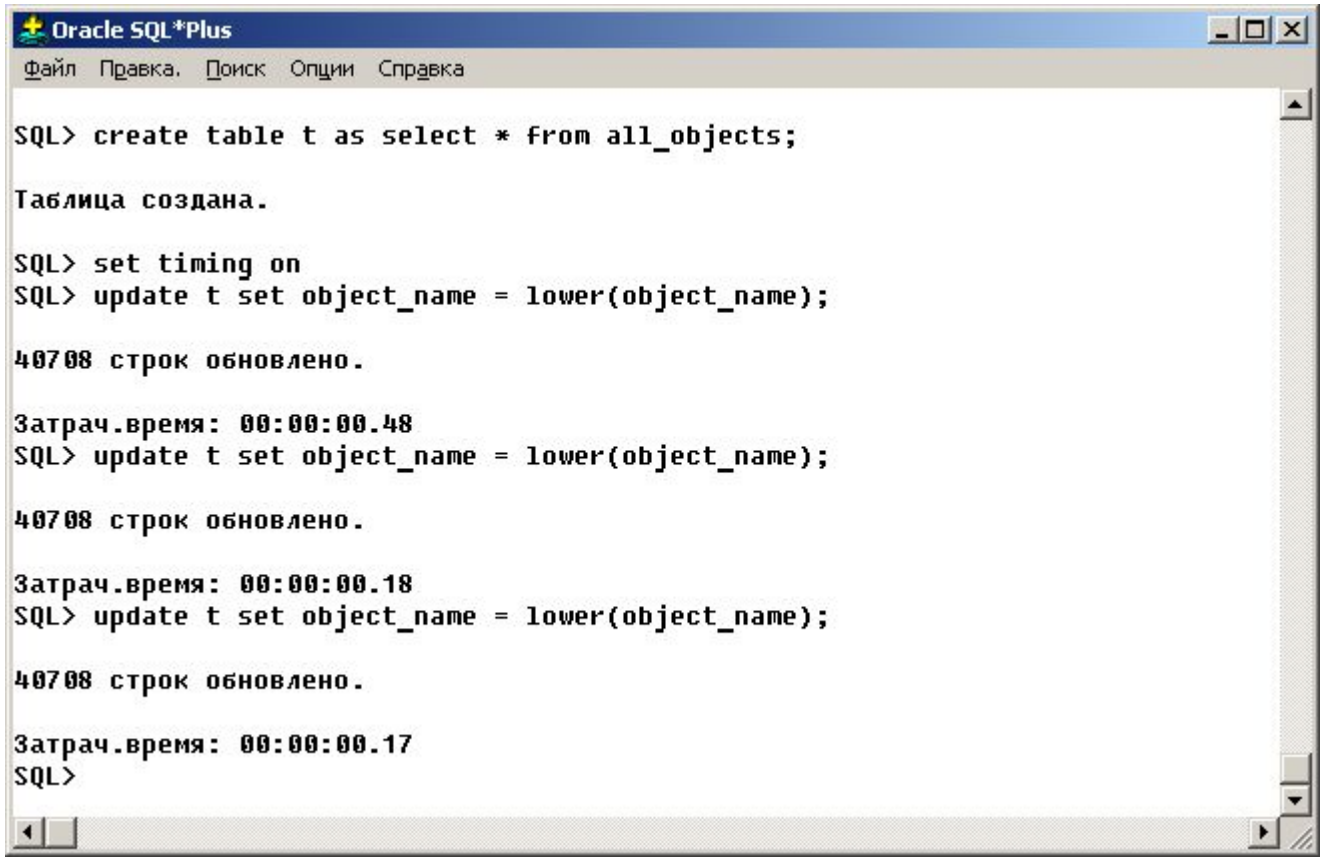
```
BEGIN
   COMMIT;

   SAVEPOINT POINT1;
      UPDATE EMP SET SAL = 3000 WHERE EMPNO = 7902;

   SAVEPOINT POINT2;
      SELECT SUM(SAL) INTO varSum FROM EMP;
      DBMS_OUTPUT.PUT_LINE('varSum1=' || varSum);

      UPDATE EMP SET SAL = SAL + 1000 WHERE EMPNO = 7788;

      SELECT SUM(SAL) INTO varSum FROM EMP;
      DBMS_OUTPUT.PUT_LINE('varSum2=' || varSum);

     IF varSum > 34000 THEN ROLLBACK TO POINT2; END IF;

   COMMIT;
     SELECT SUM(SAL) INTO varSum FROM EMP;
     DBMS_OUTPUT.PUT_LINE('varSum3=' || varSum);

END
```

Стандартное заблуждения при работе с транзакциями

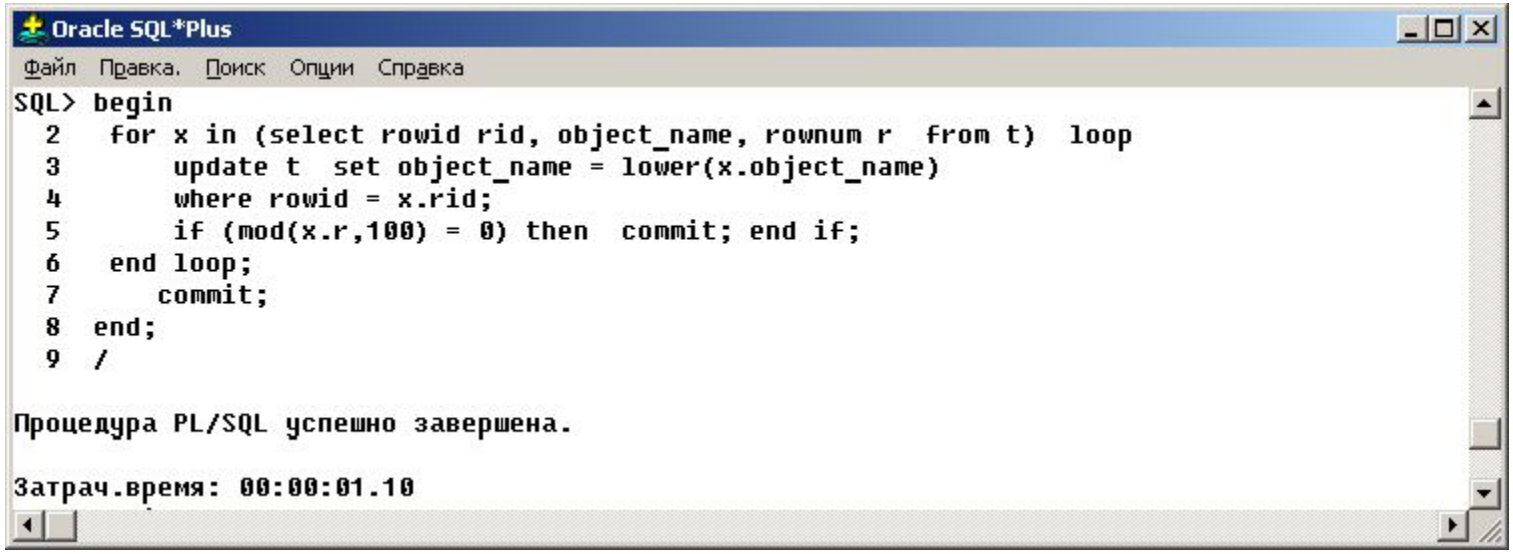
- Любую длинную транзакцию надо разбивать на части меньшего размера, чтобы она быстрее исполнялась

- Это не всегда так!!!!

Пример (длинная транзакция)



Пример (много коротких транзакций)



Когда разбиение на меньшие транзакции оправдано?

- При необходимости провести массовое обновление данных.
- При операциях вставки новых записей, если в таблице имеется первичный ключ или ограничение целостности, определяющее уникальный атрибут.
- При операциях удаления (если новые значения не зависят от старых).

Транзакции, ODBC и JDBC драйвера

- Эти функциональные интерфейсы по умолчанию выполняют "автоматическую
- фиксацию транзакции". Например, если JDBC драйвер выполняет команды:
- **update accounts set balance = balance - 1000 where account_id = 123;**
- **update accounts set balance = balance + 1000 where account_id = 456;**

- Фиксация транзакции будет происходить после каждой команды.

Как вернуть контроль над транзакциями?

- Отказаться от фиксации транзакции по умолчанию. Например:

- **connection conn11G = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:teacher", "scott", "tiger");**

- **conn11G.*setAutoCommlt (false);***

Автономные транзакции

- Автономные транзакции позволяют создать новую транзакцию в пределах текущей,
- так что можно фиксировать или откатывать ее изменения независимо от родительской
- транзакции. Они позволяют приостановить текущую транзакцию, начать новую, выполнить ряд действий, зафиксировать их или откатить, не влияя на состояние текущей транзакции.

Использование автономных транзакций

- Для фиксации динамических ошибок.
- Для аудита запрещенных попыток изменения данных.
- И т.д.

Пример (создание процедуры с автономной транзакцией)

```
create or replace procedure LogInfo
      ( inInfoMessage   in varchar2,  inSource in varchar2 )
is
      PRAGMA AUTONOMOUS_TRANSACTION;
begin
      insert into debug_log(id, LogTime, Message)
      values (seq_debug_log.nextval, sysdate, inInfoMessage, inSource);
      commit;
exception
   when others then
         return;
end LogInfo;
```

Домашнее задание 15(10 баллов)

- Создать приложение, демонстрирующее как разбиение транзакции (любого содержания, например добавление 100000 записей) на части меньшего размера приводит к изменению времени запроса (построить график зависимости времени запроса от количества разбиений).

Результат отправьте по адресу N.Grafeeva@spbu.ru. Тема письма – DB_Applcation_2015_job15.

Примечание:задание должно быть отправлено в течение 2 недель. За более позднее отправление будут сниматься штрафные баллы (по баллу за каждые 2 недели).

За сдачу 21 ноября – дополнительные 5 баллов.