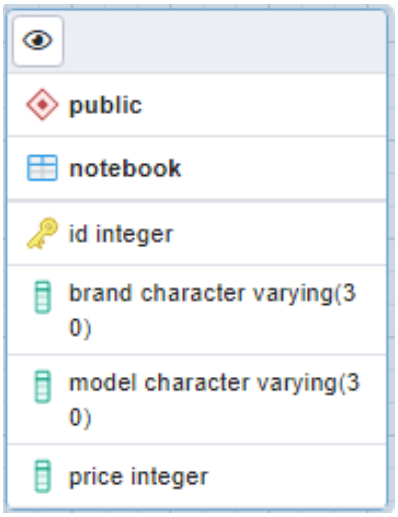
Отчет по ЛР 7 “Транзакции”

Выполнил студент группы ПМИ-34-20

Семенов Сергей

Дата сдачи: 02.12.2022




Для удобства была создана бд с одной табличкой с информацией о ноутбуках.

id

brand – название производителя

model – название модели

price – цена за ноутбук

**Транзакция** — это набор действий с данными, объединенный в логическую единицу. Она либо выполняется целиком, либо нет.

- Транзакции есть во всех СУБД.
- PostgreSQL на самом деле обрабатывает каждый SQL-оператор как транзакцию. Если мы не вставим команду BEGIN, то каждый отдельный оператор будет неявно окружён командами BEGIN и COMMIT
- Промежуточные состояния внутри последовательности не видны другим транзакциям, и, если что-то помешает успешно завершить транзакцию, ни один из результатов этих действий не сохранится в базе данных

Транзакции обладают четырьмя основными свойствами – ACID:

1. **Atomicity — Атомарность.**
   *Атомарность гарантирует, что каждая транзакция будет выполнена полностью или не будет выполнена совсем.*
2. **Consistency — Согласованность**
   *каждая успешная транзакция по определению фиксирует только допустимые результаты*.
3. **Isolation — Изолированность**
   *Во время выполнения транзакции параллельные транзакции не должны оказывать влияния на её результат. Параллельная работа с несколькими транзакциями может привести к нежелательным эффектам. Подробнее будут описаны в отдельном пункте.*
4. **Durability — Надёжность**
   *Если пользователь получил подтверждение от системы, что транзакция выполнена, он может быть уверен, что сделанные им изменения не будут отменены из-за какого-либо сбоя.*

В разных СУБД эти свойства могут быть реализованы по-разному, но результат работы свойств ACID всегда один. Транзакция либо отрабатывает корректно, либо завершается с соответствующим сообщением об ошибке.

Уровни изоляции и возможные проблемы параллельного доступа

| Уровень изоляции | «Грязное» чтение | Неповторяемое чтение | Фантомное чтение | Аномалия сериализации |
|---|---|---|---|---|
| Read uncommited - Не поддерживается в PostgreSQL (Чтение незафиксированных данных) | Допускается, но не в PG | Возможно | Возможно | Возможно |
| Read committed (Чтение зафиксированных данных) | Невозможно | Возможно | Возможно | Возможно |
| Repeatable read (Повторяемое чтение) | Невозможно | Невозможно | Допускается, но не в PG | Возможно |
| Serializable (Сериализуемость) | Невозможно | Невозможно | Невозможно | Невозможно |

Проблемы параллельного доступа:

1. **Грязное чтение**

   *Транзакция читает данные, записанные параллельной незавершённой транзакцией (т.е. это такое чтение, при котором могут быть считаны добавленные или изменённые данные из другой транзакции, которая впоследствии не подтвердится). Данная проблему невозможно встретить в PostgreSQL т.к. в этой СУБД уровень изоляции Read uncommited не реализован, а все остальные уровни изоляции исключают эту проблему.*

   *Проверим. Далее различные транзакции будут обозначены как транзакция 1 – A, транзакция 2 – B.*

--A

```
begin;
update notebook set price = price + 1
where brand = 'dell';
select * from notebook where brand = 'dell';
```

| | id [PK] integer | brand character varying (30) | model character varying (30) | price integer |
|---|---|---|---|---|
| 1 | 4 | dell | s27 | 75001 |
| 2 | 6 | dell | s-30 | 82001 |

--B

```
select * from notebook where brand = 'dell';
```

| | id [PK] integer | brand character varying (30) | model character varying (30) | price integer |
|---|---|---|---|---|
| 1 | 4 | dell | s27 | 75000 |
| 2 | 6 | dell | s-30 | 82000 |

Фактический результат совпал с ожидаемым. Транзакция не прочитала данные, измененные параллельной незавершенной транзакцией.

2. **Неповторяемое чтение**

*Ситуация, когда при повторном чтении в рамках одной транзакции ранее прочитанные данные оказываются изменёнными*

Проверим:

--A

```
begin;
select * from notebook where brand = 'asus';
```

| | id [PK] integer | brand character varying (30) | model character varying (30) | price integer |
|---|---|---|---|---|
| 1 | 2 | asus | rog-2 | 50000 |
| 2 | 5 | asus | rog-3 | 65000 |
| 3 | 8 | asus | rog-5 | 100000 |

--B

update notebook set model = 'ultra-' || model

where brand = 'asus';

| | id [PK] integer | brand character varying (30) | model character varying (30) | price integer |
|---|---|---|---|---|
| 1 | 2 | asus | ultra-rog-2 | 50000 |
| 2 | 5 | asus | ultra-rog-3 | 65000 |
| 3 | 8 | asus | ultra-rog-5 | 100000 |

--A

select * from notebook where brand = 'asus';

| | id [PK] integer | brand character varying (30) | model character varying (30) | price integer |
|---|---|---|---|---|
| 1 | 2 | asus | ultra-rog-2 | 50000 |
| 2 | 5 | asus | ultra-rog-3 | 65000 |
| 3 | 8 | asus | ultra-rog-5 | 100000 |

Незавершенная транзакция прочитала информацию, которая изменилась в завершенной транзакции. Исправить это можно, установив уровень изоляции транзакции A repeatable read и выше.

После begin; пишем

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;

И в таком случае, результатом второго селекта в транзакции А будет такая же, как и после первого:

| | id [PK] integer | brand character varying (30) | model character varying (30) | price integer |
|---|---|---|---|---|
| 1 | 2 | asus | rog-2 | 50000 |
| 2 | 5 | asus | rog-3 | 65000 |
| 3 | 8 | asus | rog-5 | 100000 |

3. **Фантомное чтение**

*Ситуация, когда при повторном чтении в рамках одной транзакции одна и та же выборка дает разные множества строк (От неповторяющегося чтения оно отличается тем, что результат повторного обращения к данным изменился не из-за изменения/удаления самих этих данных, а из-за появления новых (фантомных) данных.)*

Проверим:

--A

begin;

select * from notebook where brand = 'logitech';

| | id [PK] integer | brand character varying (30) | model character varying (30) | price integer |
|---|---|---|---|---|
| 1 | 15 | logitech | g2 | 80000 |

--B

insert into notebook values(default, 'logitech', 'g3', 85000);

--A

select * from notebook where brand = 'logitech';

| | id [PK] integer | brand character varying (30) | model character varying (30) | price integer |
|---|---|---|---|---|
| 1 | 15 | logitech | g2 | 80000 |
| 2 | 19 | logitech | g3 | 85000 |

Незавершенная транзакция прочитала информацию, которая добавилась в завершенной транзакции. Исправить это можно, установив уровень изоляции транзакции A repeatable read и выше.

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;

И в таком случае, результатом второго селекта в транзакции A будет такая же, как и после первого:

| | id [PK] integer | brand character varying (30) | model character varying (30) | price integer |
|---|---|---|---|---|
| 1 | 15 | logitech | g2 | 80000 |

Что мы и хотели получить

4. **Аномалия сериализации**
   *возникает, когда две транзакции читают одну и ту же строку таблицы, затем одна транзакция обновляет эту строку, а после этого вторая транзакция тоже обновляет ту же строку, не учитывая изменений, сделанных первой транзакцией*