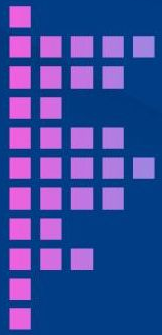


OTUS

SQL и реляционные СУБД. Введение в PostgreSQL

otus.ru

Проверить, идет ли запись

Меня хорошо видно && слышно?

Ставим “+”, если все хорошо
“-”, если есть проблемы

Правила вебинара

Активно
участвуем

Off-topic обсуждаем
в учебной группе ТГ

Задаем вопрос
в чат или голосом

Вопросы вижу в чате,
могу ответить не сразу

Маршрут вебинара

1. Реляционная модель и SQL
2. OLTP, ACID, MVCC, ARIES
3. Уровни изоляции транзакций
4. Современные РСУБД
5. Введение в PostgreSQL. Практика

Цели вебинара

К концу занятия вы сможете

1. Объяснить основу реляционной модели данных;
2. Объяснить назначение языка SQL и его основные конструкции;
3. Иметь представление о основных реляционных СУБД.

Смысл

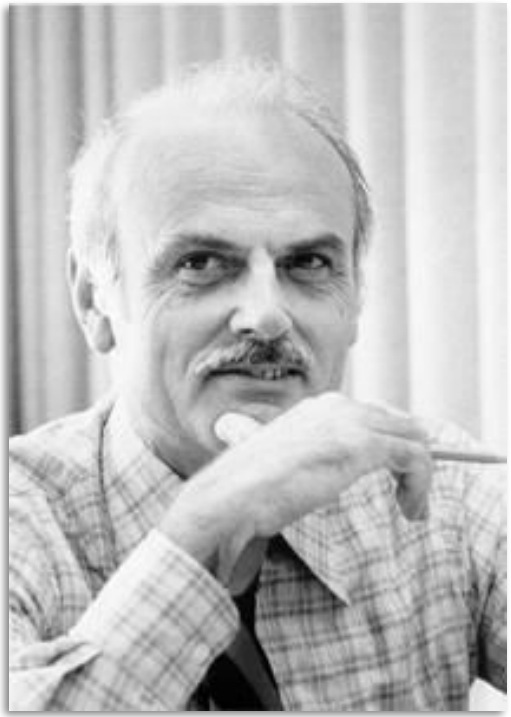
Зачем вам это уметь

1. Говорить на одном языке с разработчиками и архитекторами БД;
2. Понимать в каком направлении развивать свои навыки касательно работы со структурированными данными;
3. Научитесь минимальной работе с СУБД PostgreSQL

Реляционная модель и SQL

Реляционная модель

- начало 1970-х
- основоположник Эдгар Франк Кодд (Edgar Frank Codd)
- в рамках программы исследований IBM
  В 1970 году издал работу «A Relation Model of Data for Large Shared Data Banks», которая считается первой работой по реляционной модели данных
- NULL, view, нормализация, select и т.д.
- позднее предложил «12 правил Кодда»



Реляционная модель

- Нормализация отношений. Шесть нормальных форм - Habr
- Нормальная форма - Wikipedia
- минимизация логической избыточности
- сокращает объем хранимых данных
- увеличивает производительность
- в разумных пределах полезна для транзакционных БД
- вредна для аналитических БД

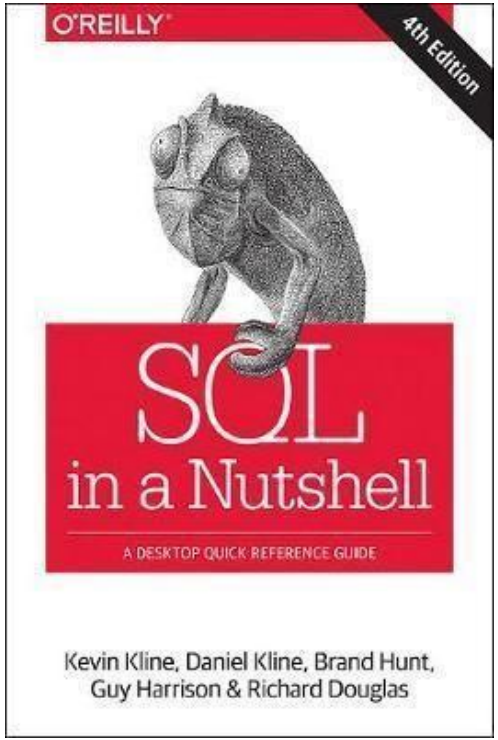
Реляционная модель и SQL

SQL

- разработан в IBM в начале 1970-х
- в рамках проекта СУБД IBM System R (потом стал DB2) и на основе работ Кодда
- SEQUEL, Structured English QUEry Language
- стал стандартом ANSI в 1986-м и ISO в 1987-м
- последний на данный момент времени стандарт SQL:2016, в 2019 внесена корректировка, пересматривается каждые 5 лет

Стандарт SQL выделяет 5 видов операций

- **DML - modification**
  - INSERT
  - UPDATE
  - DELETE
- **DDL - definition**
  - ALTER
  - CREATE
  - DROP
- **DCL - control**
  - GRANT
  - REVOKE
- **DQL - query**
  - SELECT
- **TCL - transaction control**
  - BEGIN
  - COMMIT
  - ROLLBACK



Реляционная модель и SQL

- SQL constraints
  - правила описывающие корректность данных
  - работают на уровне DML
  - любимы DBA
  - не любимы разработчиками
  - давайте им имена уж если вы их используете
    - ANSI:
    - PRIMARY KEY
    - FOREIGN KEY
    - UNIQUE
    - CHECK

OLTP, ACID, MVCC, ARIES

Множественная параллельная нагрузка

- реляционная теория и SQL позволяет абстрагироваться от конкретной реализации СУБД
- но есть одна непростая проблема
- как обеспечить **параллельную** работу множества сессий (**concurrency**)
- которые **модифицируют** данные
- так чтобы они **не мешали** друг другу
- ни с точки зрения **чтения** ни с точки зрения **записи**
- и обеспечивали **целостность** данных - т.н. **consistency**
- и их надежность - т.н. **durability**

ACID

ACID

- **Atomicity** — Атомарность
- **Consistency** — Согласованность
- **Isolation** — Изолированность
- **Durability** — Долговечность

Транзакция (**transaction**):

- называется множество операций, выполняемое приложением
- которое переводит базу данных из одного корректного состояния в другое корректное состояние (согласованность)
- при условии, что транзакция выполнена полностью (атомарность)
- и без помех со стороны других транзакций (изолированность)

ACID - хорошо, а как блокировки?

- **ARIES** - **A**lgorithms for **R**ecovery and **I**solation **E**xploiting **S**emantics
  - logging
  - undo
  - redo
  - checkpoints
- **MVCC** - **M**ulti**v**ersion **C**oncurrency **C**ontrol
  - copy-on-write
  - каждый пользователь работает со снимком БД
  - вносимые пользователем изменения не видны другим до фиксации транзакции

Современные РСУБД

Современные РСУБД

- поддержка SQL: 2016 как минимум
- ACID: ARIES/MVCC
- поддержка Linux
- наличие облачных сервисов (cloud managed services)
- работа в Docker и Kubernetes

Современные РСУБД

	Oracle Database	SQL Server	MySQL	PostgreSQL	MariaDB
Тип	закрытая	закрытая	открытая	открытая	открытая
Год основания	1979	1989	1995	1989	2009
Текущая версия	23c	2023	8.0	16	11
ANSI/ISO SQL	SQL:2016	SQL:2016	SQL:2016	SQL:2016	SQL:2016
Процедурное расширение	PL/SQL	T-SQL	SQL/PSM	PL/pgSQL, PL/Tcl, PL/Perl, PL/Python	SQL
ACID	ARIES/MVCC	ARIES, MVCC-ready	ARIES/MVCC	ARIES/MVCC	ARIES/MVCC
Linux	Oracle, RedHat, SuSE	RedHat, SuSE, Ubuntu	Any	Any	Any
Cloud	AWS, Oracle	AWS, Azure, GCP	AWS, Azure, GCP, Oracle	AWS, Azure, YC	AWS
Docker/Kubernetes	cloud	+	+	+	+
Цена	космос	чуть меньше	free	free	free

Введение в PostgreSQL. Практика

Введение в PostgreSQL

- автор: Майк Стоунбрейкер, Бёркли
- начало 1970-х: Ingres — INteractive Grafic REtrieval System
- середина 1980-х: Postgres - Post Ingres
- Postgres95 - добавлен SQL
- 1996: postgresql.org
- сейчас развивается PostgreSQL Global Development Group
- компании контрибьюторы:
  - 2ndQuadrant (в прошлом году куплен EnterpriseDB)
  - EnterpriseDB
  - Crunchy Data
  - Postgres Professional

release notes PostgreSQL

https://www.postgresql.org/docs/release/10.0/
Встречаем PostgreSQL 10. Перевод Release Notes / Блог компании Авито / Хабр

https://www.postgresql.org/docs/release/13.0/

PostgreSQL 13 Released!
https://info.crunchydata.com/blog/why-postgresql-13-is-a-lucky-release

https://pgpedia.info/postgresql-versions/postgresql-14.html

https://www.postgresql.org/docs/release/15.0/

https://www.postgresql.org/docs/release/16.0/

PostgreSQL Extensions

https://www.postgresql.org/docs/current/contrib.html

Документация PostgreSQL

https://www.postgresql.org/files/documentation/pdf/16/postgresql-16-A4.pdf

Практика

Создадим свою БД и несколько объектов в ней

Минитест

https://forms.gle/pFTbtRzvkgb5KNMF6

Уровни изоляции транзакций

Уровни изоляции транзакций

Стандарт SQL допускает четыре уровня изоляции, которые определяются в терминах аномалий, которые допускаются при конкурентном выполнении транзакций на этом уровне:

- **«Грязное» чтение (dirty read)**. Транзакция T1 может читать строки измененные, но еще не зафиксированные, транзакцией T2. Отмена изменений (ROLLBACK) в T2 приведет к тому, что T1 прочитает данные, которых никогда не существовало.

Уровни изоляции транзакций

Стандарт SQL допускает четыре уровня изоляции, которые определяются в терминах аномалий, которые допускаются при конкурентном выполнении транзакций на этом уровне:

- **«Грязное» чтение (dirty read)**. Транзакция T1 может читать строки измененные, но еще не зафиксированные, транзакцией T2. Отмена изменений (ROLLBACK) в T2 приведет к тому, что T1 прочитает данные, которых никогда не существовало.
- **Неповторяющееся чтение** (non-repeatable read). После того, как транзакция T1 прочитала строку, транзакция T2 изменила или удалила эту строку и зафиксировала изменения (COMMIT). При повторном чтении этой же строки транзакция T1 видит, что строка изменена или удалена.

Уровни изоляции транзакций

Стандарт SQL допускает четыре уровня изоляции, которые определяются в терминах аномалий, которые допускаются при конкурентном выполнении транзакций на этом уровне:

- **«Грязное» чтение (dirty read)**. Транзакция T1 может читать строки измененные, но еще не зафиксированные, транзакцией T2. Отмена изменений (ROLLBACK) в T2 приведет к тому, что T1 прочитает данные, которых никогда не существовало.
- **Неповторяющееся чтение** (non-repeatable read). После того, как транзакция T1 прочитала строку, транзакция T2 изменила или удалила эту строку и зафиксировала изменения (COMMIT). При повторном чтении этой же строки транзакция T1 видит, что строка изменена или удалена.
- **Фантомное чтение** (phantom read). Транзакция T1 прочитала набор строк по некоторому условию. Затем транзакция T2 добавила строки, также удовлетворяющие этому условию. Если транзакция T1 повторит запрос, она получит другую выборку строк.

Уровни изоляции транзакций

Стандарт SQL допускает четыре уровня изоляции, которые определяются в терминах аномалий, которые допускаются при конкурентном выполнении транзакций на этом уровне:

- **«Грязное» чтение (dirty read)**. Транзакция T1 может читать строки измененные, но еще не зафиксированные, транзакцией T2. Отмена изменений (ROLLBACK) в T2 приведет к тому, что T1 прочитает данные, которых никогда не существовало.
- **Неповторяющееся чтение** (non-repeatable read). После того, как транзакция T1 прочитала строку, транзакция T2 изменила или удалила эту строку и зафиксировала изменения (COMMIT). При повторном чтении этой же строки транзакция T1 видит, что строка изменена или удалена.
- **Фантомное чтение** (phantom read). Транзакция T1 прочитала набор строк по некоторому условию. Затем транзакция T2 добавила строки, также удовлетворяющие этому условию. Если транзакция T1 повторит запрос, она получит другую выборку строк.
- **Аномалия сериализации** - Результат успешной фиксации группы транзакций оказывается несогласованным при всевозможных вариантах исполнения этих транзакций по очереди.

Уровни изоляции транзакций

	«грязное» чтение	неповторяющееся чтение	фантомное чтение	аномалия сериализации
Read Uncommitted	Допускается, но не в PG	да	да	да
Read Committed	-	да	да	да
Repeatable Read	-	-	Допускается, но не в PG	да
Serializable	-	-	-	-

На всех уровнях не допускается потеря зафиксированных изменений
https://habr.com/ru/company/otus/blog/501294/
https://www.postgresql.org/docs/current/transaction-iso.html

Уровни изоляции транзакций

Аномалия сериализации

class | value

-------+-------

1 | 10

1 | 20

2 | 100

2 | 200

сериализуемая транзакция A вычисляет:

`SELECT SUM(value) FROM mytab WHERE class = 1` ; добавляем запись с суммой и class 2

сериализуемая транзакция B вычисляет:

`SELECT SUM(value) FROM mytab WHERE class = 2` ; добавляем запись с суммой и class 1 что произойдет при попытке фиксации изменений?

Уровни изоляции транзакций

ОШИБКА: не удалось сериализовать доступ из-за зависимостей чтения/записи между транзакциями

Если бы одна из транзакций была бы repeatable read, все бы успешно зафиксировалось.

Уровни изоляции транзакций. Практика

Рефлексия

Рефлексия

Кто что запомнил за сегодня?

Что узнали нового?

Домашнее задание

ДЗ сдаем в виде мини отчета в markdown в гите
Критерии оценивания:
Выполнение ДЗ: 10 баллов
плюс 2 балла за красивое решение
минус 2 балла за рабочее решение, и недостатки указанные преподавателем не устранены

Заполните, пожалуйста, опрос о занятии по ссылке в чате

Спасибо за внимание!

Приходите на следующие вебинары

Ведущий разработчик СУБД

Специалист в области разработки и проектировании витрин данных в PostgreSQL, а также в области разработки хранимых процедур в таких СУБД как PostgreSQL и Oracle