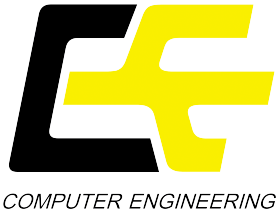

Laporan Praktikum Basis Data

JUDUL MODUL

Nama lengkap - NRP

Departemen Teknik Komputer
Institut Teknologi Sepuluh Nopember
Surabaya

Penulis: lab b201
2025

Daftar Isi

Implementasi Transaction dan ACID pada PostgreSQL **2**
1 Pendahuluan 2
1.1 Tujuan Praktikum 2
2 Konsep ACID dalam Transaksi Database 3
2.1 Atomicity (Atomisitas) 3
2.2 Consistency (Konsistensi) 3
2.3 Isolation (Isolasi) 3
2.4 Durability (Durabilitas) 4
3 Tingkat Isolasi Transaksi 4
3.1 READ UNCOMMITTED 4
3.2 READ COMMITTED (Default) 4
3.3 REPEATABLE READ 5
3.4 SERIALIZABLE 5
3.5 Anomali Konkurensi 5
4 Strategi Penanganan Konkurensi 6
4.1 Pessimistic Concurrency Control 6
4.2 Optimistic Concurrency Control 6
4.3 Multi-Version Concurrency Control (MVCC) 7
5 Referensi 7

Dasar Teori **8**

Hasil Praktikum **9**

Tugas Modul **10**

Tugas Asistensi **11**

Kesimpulan **12**

Implementasi Transaction dan ACID pada PostgreSQL

1 Pendahuluan

Transaksi (transaction) merupakan salah satu konsep fundamental dalam sistem basis data relasional yang menjamin integritas dan reliabilitas data. Dalam PostgreSQL, implementasi transaksi yang tepat dan pemahaman ACID properties sangat penting untuk mengembangkan aplikasi yang handal, terutama dalam lingkungan dengan konkurensi tinggi.

Transaksi adalah unit kerja yang mengubah data dari satu state ke state lainnya. Semua perubahan dalam transaksi harus berhasil secara lengkap, atau tidak ada yang dilakukan sama sekali (all-or-nothing). Konsep ini menjamin bahwa database tetap dalam keadaan konsisten meskipun terjadi kegagalan sistem.

1.1 Tujuan Praktikum

Dilaksanakannya praktikum transaction dan ACID, praktikan diharapkan mampu:

1. Memahami konsep ACID dalam transaksi database
2. Mengimplementasikan transaksi pada PostgreSQL
3. Mempelajari tingkat isolasi transaksi
4. Menganalisis fenomena konkurensi seperti dirty read, non-repeatable read, dan phantom read
5. Mengimplementasikan strategi penanganan konkurensi tinggi

2 Konsep ACID dalam Transaksi Database

ACID adalah akronim yang merepresentasikan empat properti penting dari transaksi database yang menjamin reliabilitas data meskipun terjadi kegagalan sistem, error, atau konkurensi akses.

2.1 Atomicity (Atomisitas)

1. Transaksi harus dijalankan sepenuhnya atau dibatalkan sepenuhnya
2. Tidak ada hasil parsial; jika satu operasi gagal, semua operasi dibatalkan
3. Implementasi menggunakan mekanisme COMMIT dan ROLLBACK
4. Contoh: Transfer uang antar rekening harus mengurangi saldo satu rekening dan menambah rekening lain, atau tidak melakukan perubahan sama sekali

2.2 Consistency (Konsistensi)

1. Transaksi harus membawa database dari satu state valid ke state valid lainnya
2. Semua aturan integritas data (constraints, cascade, triggers) harus tetap terpenuhi
3. Jumlah total saldo sebelum dan sesudah transaksi harus tetap sama
4. Melindungi aplikasi dari data yang tidak konsisten

2.3 Isolation (Isolasi)

1. Transaksi harus berjalan seolah-olah tidak ada transaksi lain yang berjalan secara bersamaan
2. Mencegah transaksi membaca data ”kotor” dari transaksi lain yang belum commit
3. PostgreSQL mengimplementasikan multi-version concurrency control (MVCC)
4. Terdapat beberapa tingkat isolasi yang bisa dipilih sesuai kebutuhan

2.4 Durability (Durabilitas)

1. Perubahan yang sudah di-commit harus tetap tersimpan, bahkan jika terjadi kegagalan sistem
2. PostgreSQL menggunakan Write-Ahead Logging (WAL) untuk menjamin durabilitas
3. Data yang sudah di-commit harus bertahan dari crash, power failure, dll
4. Memungkinkan pemulihan data setelah kegagalan sistem

3 Tingkat Isolasi Transaksi

PostgreSQL mendukung beberapa tingkat isolasi sesuai dengan standar SQL. Setiap tingkat isolasi menawarkan keseimbangan antara konsistensi dan performa.

3.1 READ UNCOMMITTED

1. PostgreSQL tidak benar-benar mengimplementasikan tingkat ini
2. Dalam PostgreSQL, READ UNCOMMITTED sama dengan READ COMMITTED
3. Tidak mungkin membaca perubahan yang belum di-commit (dirty read)

3.2 READ COMMITTED (Default)

1. Tingkat isolasi default di PostgreSQL
2. Transaksi hanya dapat membaca data yang sudah di-commit
3. Mencegah dirty read, tetapi memungkinkan non-repeatable read dan phantom read
4. Membaca yang sama dalam satu transaksi dapat mengembalikan hasil berbeda jika transaksi lain melakukan commit

3.3 REPEATABLE READ

1. Transaksi hanya melihat data yang sudah di-commit sebelum transaksi dimulai
2. Mencegah dirty read dan non-repeatable read
3. Melindungi dari perubahan data yang dibaca, tetapi masih memungkinkan phantom read
4. Semua query dalam satu transaksi melihat snapshot database yang sama

3.4 SERIALIZABLE

1. Tingkat isolasi tertinggi
2. Mencegah semua anomali konkurensi (dirty read, non-repeatable read, phantom read)
3. Transaksi berjalan seolah-olah dieksekusi secara serial (satu per satu)
4. Dapat menyebabkan serialization error yang memerlukan retry
5. Memberikan konsistensi tertinggi dengan trade-off performa

Tingkat Isolasi	**Dirty Read**	**Non-repeatable Read**	**Phantom Read**
READ UNCOMMITTED*	Tidak	Ya	Ya
READ COMMITTED	Tidak	Ya	Ya
REPEATABLE READ	Tidak	Tidak	Ya**
SERIALIZABLE	Tidak	Tidak	Tidak

Gambar 1: Tingkat Isolasi dan Anomali Konkurensi di PostgreSQL
Sama dengan READ COMMITTED di PostgreSQL
*Di PostgreSQL, REPEATABLE READ juga mencegah phantom read, yang melebihi standar SQL

3.5 Anomali Konkurensi

Berikut adalah penjelasan lebih detail mengenai anomali yang dapat terjadi:

1. **Dirty Read**: Membaca data yang belum di-commit oleh transaksi lain

2. **Non-repeatable Read**: Transaksi membaca data dua kali, tetapi antara dua pembacaan tersebut, data diubah dan di-commit oleh transaksi lain

3. **Phantom Read**: Transaksi mengeksekusi query yang mengembalikan himpunan baris, tetapi transaksi lain insert/delete baris yang memenuhi kondisi query

4 Strategi Penanganan Konkurensi

Untuk menangani konkurensi dengan baik, PostgreSQL menyediakan beberapa strategi:

4.1 Pessimistic Concurrency Control

1. Menggunakan locking untuk mencegah konflik

2. Row-level locking dengan FOR UPDATE, FOR SHARE, dll

3. Table-level locking dengan LOCK TABLE

4. Mencegah konflik sebelum terjadi

5. Dapat menyebabkan deadlock

4.2 Optimistic Concurrency Control

1. Mengasumsikan konflik jarang terjadi

2. Tidak mengunci data, tetapi memeriksa apakah data berubah sebelum commit

3. Biasanya menggunakan version number atau timestamp

4. Cocok untuk lingkungan dengan read-heavy workload

5. Memerlukan retry jika terjadi konflik

4.3 Multi-Version Concurrency Control (MVCC)

1. PostgreSQL menggunakan MVCC untuk implementasi isolasi
2. Setiap transaksi melihat snapshot database pada titik waktu tertentu
3. Memungkinkan reader tidak mengunci writer dan sebaliknya
4. Meningkatkan konkurensi karena read tidak menghalangi write
5. Trade-off adalah overhead penyimpanan untuk multiple versions

5 Referensi

- PostgreSQL Documentation: Transactions
- PostgreSQL: Concurrency Control
- PostgreSQL: Explicit Locking
- PostgreSQL: Administrative Functions
- PostgreSQL: Monitoring Database Activity

Dasar Teori

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Hasil Praktikum

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Tugas Modul

1. Bandingkan performa dengan dan tanpa index untuk setiap query Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

2. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Tugas Asistensi

1. Bla bla bla bla Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

2. Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Kesimpulan

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.