Tugas Basis Data Sesi 11

Aldi Maulana Iqbal – 20210801222

Mengapa tidak konsistenan data dan kehilangan data dapat terjadi. Mengapa penting untuk mengelola transaksi secara bersamaan?

Ketidakkonsistenan data dapat terjadi karena adanya kesalahan input, proses yang tidak sesuai, atau perubahan data yang tidak dilakukan dengan benar. Hal ini dapat menyebabkan adanya kekeliruan atau kontradiksi dalam data yang disimpan.

Kehilangan data dapat terjadi karena adanya kerusakan pada perangkat penyimpanan data, seperti hard drive yang rusak, atau karena adanya virus atau malware yang menghapus atau merusak data. Kehilangan data juga dapat terjadi karena adanya kesalahan manusia, seperti ketika seseorang secara tidak sengaja menghapus atau menimpa data yang ada.

Mengelola transaksi secara bersamaan penting karena memastikan bahwa semua transaksi yang dilakukan di sistem dapat dilakukan secara bersamaan dan tidak terjadi konflik atau kegagalan transaksi. Ini membantu menjaga konsistensi dan keandalan data yang disimpan dalam sistem. Selain itu, mengelola transaksi secara bersamaan juga memastikan bahwa tidak ada transaksi yang hilang atau terlewatkan dalam proses pemrosesan.

Jelaskan dan berikan contoh, jenis masalah yang dapat terjadi di lingkungan multi-user ketika akses bersamaan ke database diperbolehkan.

Di lingkungan multi-user, terdapat beberapa masalah yang dapat terjadi ketika akses bersamaan ke database diperbolehkan. Berikut ini adalah beberapa contoh masalah yang dapat terjadi:

1. Konflik akses: Ketika beberapa pengguna mencoba mengakses data yang sama pada saat yang bersamaan, maka dapat terjadi konflik akses yang dapat menyebabkan data tidak tersimpan dengan benar atau tidak dapat diakses oleh pengguna lain.
2. Data tidak sinkron: ketika beberapa pengguna mengubah data pada saat yang bersamaan, maka dapat terjadi data tidak sinkron, yaitu terdapat perbedaan antara data yang disimpan di database dan data yang diakses oleh pengguna lain.
3. Deadlock: ketika beberapa pengguna mencoba mengakses data yang sama secara bersamaan, maka dapat terjadi deadlock, yaitu terjadinya kemacetan di database sehingga tidak dapat diakses oleh pengguna lain.

Contoh: Ketika beberapa pengguna sedang mencoba mengubah data pada saat yang bersamaan, maka dapat terjadi konflik akses. Misalnya, pengguna A sedang mengubah data produk dengan ID 100, sedangkan pengguna B juga sedang mengubah data produk dengan ID 100. Jika tidak ada mekanisme yang menangani konflik akses ini, maka data yang diubah oleh pengguna A mungkin tidak tersimpan dengan benar atau tidak dapat diakses oleh pengguna B.

Tunjukkan bagaimana mekanisme mencegah masalah yang digambarkan terjadi. Diskusikan bagaimana mekanisme kontrol konkurensi berinteraksi dengan mekanisme transaksi.

Untuk mencegah masalah yang telah digambarkan terjadi, terdapat beberapa mekanisme yang dapat digunakan, yaitu:

1. Blocking: mekanisme ini menggunakan lock pada data yang sedang diakses oleh pengguna, sehingga pengguna lain tidak dapat mengakses data tersebut sampai lock dibuka.
2. Pembatasan akses: mekanisme ini menggunakan pembatasan akses pada data, sehingga hanya pengguna yang memiliki izin yang sesuai yang dapat mengakses data tersebut.
3. Transaksi: mekanisme ini menggunakan transaksi untuk mengelola perubahan data yang dilakukan oleh pengguna. Transaksi memungkinkan pengguna untuk melakukan beberapa perubahan data secara bersamaan sebagai satu kesatuan, sehingga jika terjadi masalah pada salah satu perubahan data, maka semua perubahan data lainnya dapat dibatalkan.

Mekanisme kontrol konkurensi dan mekanisme transaksi berinteraksi dengan cara sebagai berikut:

1. Mekanisme kontrol konkurensi, seperti blocking dan pembatasan akses, digunakan untuk mengontrol akses pengguna ke data, sehingga tidak terjadi konflik akses atau data tidak sinkron.
2. Mekanisme transaksi digunakan untuk mengelola perubahan data yang dilakukan oleh pengguna, sehingga jika terjadi masalah pada salah satu perubahan data, maka semua perubahan data lainnya dapat dibatalkan. Dengan demikian, mekanisme transaksi dapat membantu mengurangi risiko terjadinya deadlock.

Sebagai contoh, ketika pengguna A dan pengguna B sedang mencoba mengubah data produk dengan ID 100 pada saat yang bersamaan, maka mekanisme blocking dapat digunakan untuk mencegah konflik akses dengan cara mengunci data tersebut sehingga hanya pengguna A yang dapat mengaksesnya. Selain itu, mekanisme transaksi dapat digunakan untuk mengelola perubahan data yang dilakukan oleh pengguna A dan pengguna B, sehingga jika terjadi masalah pada salah satu perubahan data, maka perubahan data lainnya dapat dibatalkan.

Jelaskan konsep jadwal serial, nonserial, dan serializable.

Jadwal adalah urutan operasi transaksi yang terjadi di database. Terdapat beberapa jenis jadwal yang dapat digunakan untuk mengelola transaksi di database, yaitu:

1. Jadwal serial: jadwal ini menyatakan bahwa setiap transaksi harus dijalankan secara berurutan, sehingga tidak terjadi konflik akses pada data. Jadwal serial merupakan jadwal yang paling aman, karena tidak memungkinkan terjadinya konflik akses atau data tidak sinkron. Namun, jadwal serial juga merupakan jadwal yang paling lambat, karena setiap transaksi harus menunggu transaksi sebelumnya selesai sebelum dijalankan.

2. Jadwal nonserial: jadwal ini menyatakan bahwa transaksi dapat dijalankan secara bersamaan, sehingga dapat terjadi konflik akses pada data. Jadwal nonserial merupakan jadwal yang lebih cepat daripada jadwal serial, karena transaksi dapat dijalankan secara bersamaan. Namun, jadwal nonserial juga merupakan jadwal yang lebih rawan terhadap masalah, seperti konflik akses atau data tidak sinkron.
3. Jadwal serializable: jadwal ini menyatakan bahwa transaksi dapat dijalankan secara bersamaan, namun tidak memungkinkan terjadinya konflik akses pada data. Jadwal serializable merupakan jadwal yang menggabungkan kelebihan dari jadwal serial dan jadwal nonserial. Jadwal serializable cepat seperti jadwal nonserial, namun aman seperti jadwal serial, karena tidak memungkinkan terjadinya konflik akses atau data tidak sinkron.

Peran apa yang dimainkan oleh subsistem pengelola transaksi DBMS dalam sistem multi-pengguna?

Subsistem pengelola transaksi (Transaction Management Subsystem) merupakan salah satu subsistem yang terdapat pada DBMS (Database Management System). Subsistem ini bertugas untuk mengelola transaksi yang dilakukan oleh pengguna di sistem multi-pengguna.

Di sistem multi-pengguna, subsistem pengelola transaksi memiliki peran penting dalam menjamin integritas dan konsistensi data di database. Subsistem ini mengontrol akses pengguna ke data dan mengelola perubahan data yang dilakukan oleh pengguna agar tidak terjadi konflik akses atau data tidak sinkron.

Selain itu, subsistem pengelola transaksi juga bertugas untuk mengelola jadwal transaksi yang terjadi di sistem multi-pengguna. Subsistem ini dapat menggunakan jadwal serial, nonserial, atau serializable sesuai dengan kebutuhan sistem.

Diskusikan jenis masalah yang dapat terjadi dengan mekanisme berbasis penguncian untuk kontrol konkurensi dan tindakan yang dapat diambil oleh DBMS untuk mencegahnya.

Mekanisme berbasis penguncian adalah salah satu mekanisme yang dapat digunakan oleh DBMS untuk mengontrol konkurensi di sistem multi-pengguna. Mekanisme ini menggunakan lock pada data yang sedang diakses oleh pengguna, sehingga pengguna lain tidak dapat mengakses data tersebut sampai lock dibuka.

Meskipun mekanisme berbasis penguncian dapat membantu mengurangi risiko terjadinya konflik akses atau data tidak sinkron, namun masih ada beberapa masalah yang dapat terjadi dengan mekanisme ini, seperti:

1. Deadlock: terjadi ketika beberapa pengguna sedang mencoba mengakses data yang sama secara bersamaan, sehingga terjadi kemacetan di database sehingga tidak dapat diakses oleh pengguna lain.
2. Starvation: terjadi ketika beberapa pengguna tidak dapat mengakses data yang diinginkan karena terus menerus dikunci oleh pengguna lain.

Untuk mencegah masalah tersebut, DBMS dapat mengambil beberapa tindakan seperti:

1. Menggunakan jadwal serializable: jadwal ini menyatakan bahwa transaksi dapat dijalankan secara bersamaan, namun tidak memungkinkan terjadinya konflik akses pada data. Dengan menggunakan jadwal serializable, DBMS dapat mencegah terjadinya deadlock dan starvation.
2. Melakukan deteksi dan pemulihan deadlock: DBMS dapat melakukan deteksi terhadap terjadinya deadlock dan mengelola deadlock dengan cara membatalkan salah satu transaksi yang terlibat dalam deadlock.
3. Memberikan prioritas akses: DBMS dapat memberikan prioritas akses kepada pengguna tertentu, seperti pengguna yang sedang melakukan transaksi penting atau pengguna yang sudah lama menunggu akses data. Dengan demikian, DBMS dapat mencegah terjadinya starvation pada pengguna yang memiliki prioritas lebih rendah.

Mengapa penguncian dua fase tidak menjadi skema kontrol konkurensi yang sesuai untuk indeks? Diskusikan skema penguncian yang lebih tepat untuk indeks berbasis tree

Penguncian dua fase (Two-Phase Locking, atau 2PL) adalah skema penguncian yang digunakan untuk mengontrol akses terhadap data dalam sistem database atau sistem manajemen basis data (DBMS). Skema ini memastikan bahwa tidak ada konflik akses antara transaksi yang sedang berlangsung, sehingga dapat mencegah terjadinya kerusakan atau korupsi data. Namun, meskipun penguncian dua fase merupakan skema penguncian yang efektif untuk mengontrol konkurensi pada data yang disimpan dalam tabel relasional, ia tidak sesuai untuk digunakan pada indeks yang berbasis tree.

Salah satu alasan mengapa penguncian dua fase tidak sesuai untuk indeks berbasis tree adalah karena skema ini tidak dapat menangani konflik akses secara efektif pada struktur data yang memiliki tingkat kedalaman yang bervariasi. Misalnya, pada sebuah indeks berbasis tree, sebuah transaksi mungkin membutuhkan akses ke node yang terletak di tingkat yang lebih rendah dari tree, sementara transaksi lain mungkin membutuhkan akses ke node yang terletak di tingkat yang lebih tinggi. Jika kedua transaksi tersebut dijalankan secara bersamaan, maka akan terjadi konflik akses yang tidak dapat diatasi oleh penguncian dua fase.

Untuk mengatasi masalah ini, skema penguncian yang lebih tepat untuk indeks berbasis tree adalah penguncian tiga fase (Three-Phase Locking, atau 3PL). Skema ini memungkinkan transaksi untuk mengunci seluruh node yang diakses secara bersamaan, sehingga menghindari konflik akses pada indeks berbasis tree. Selain itu, penguncian tiga fase juga memungkinkan transaksi untuk mengunci sebagian node saja, sehingga dapat meningkatkan efisiensi sistem dan mengurangi waktu tunggu untuk akses data.

Meskipun penguncian tiga fase lebih efektif daripada penguncian dua fase untuk mengontrol konkurensi pada indeks berbasis tree, skema ini masih memiliki beberapa kelemahan. Misalnya, skema ini dapat meningkatkan overhead sistem karena membutuhkan lebih banyak komunikasi antar transaksi untuk menentukan konflik akses.