4 Oracle Multiuser Concurrency

4.1 Inleiding

- Database systemen moeten data integriteit beschermen.
    - kan problemen geven als verschillende users op hetzelfde moment de data gebruiken.
- Dit is een probleem dat niet kan opgelost worden door de resources te vermeerderen.
- Als dit probleem niet opgelost wordt door de database, dan moeten ontwikkelaars zelf een oplossing voorzien voor dit probleem.

4.2 Basics of concurrent action

4.2.1 Transactions

Een transactie is een atomair deel werk dat data verandert.

1. Commit: aanpassingen worden uniform aangebracht in de database
2. Rollback: data die aangepast is wordt uniform teruggezet naar de initiële staat

Veranderingen worden pas opgeslagen als de transactie gecommit wordt.

4.2.2 Locks

- Worden gebruikt om ervoor te zorgen dat transacties elkaar niet beïnvloeden.

- Voorkomt dat de ene transactie veranderingen van een andere transactie overschrijft.
- Voorkomt dat gebruikers data aanpassen.
- 2 types:
    1. Exclusive locks = write lock
        - Bestaat zolang er veranderingen gebeuren aan de data.
        - Opgegeven als er een commit of rollback gebeurt.
        - Kan maar door 1 user per keer bestaan.
    2. Shared locks = read lock
        - Meerdere users kunnen dit soort lock hebben.
        - Voorkomt een write lock op de data die gelezen wordt.

Bij Oracle wordt een read lock enkel toegekend als hier specifiek om gevraagd wordt.

4.2.3 Contention

- Storing veroorzaakt door conflicterende locks
- Een transactie kan problemen veroorzaken door een lock te plaatsen op data waardoor andere transacties niet voltooid kunnen worden.

4.2.4 integriteit problemen

Als de isolatie levels van de transacties niet goedgekozen zijn dan kunnen er problemen ontstaan.

1. Lost updates

- De ene transactie overschrijft de verandering van de andere.

- Write lock

2. Dirty reads

- Er kan data gelezen worden van uncommitted changes.
- Deze verandering kan in een rollback terechtkomen, dus de gelezen data kan fout zijn.
- Veel databases staan dit wel toe voor performantie redens.

3. Inconsistente analyse
4. Fuzzy reads

> - Vragen om zelfde data kan verschillende resultaten opleveren.

2. Phantom reads

> - Dezelfde query retourneerd verschillende resultaten omdat data gewijzigd wordt terwijl men de query uitvoert.

4.2.5 Serializatie

Een serialiseerbare transactie lijkt een exclusive lock te hebben op de database voor de duur van de transactie.

De resultaten van een serialiseerbare transactie zijn voorspelbaar en reproduceerbaar.

4.3 Oracle's isolation levels

4.3.1 Read committed

- Ieder statement krijgt een consistent beeld van hoe de data eruit zag bij het begin van dat statement.
- Als een transactie uit meerdere statements bestaat kunnen er phantom reads optreden en fuzzy reads tot gevolg hebben.

Als een lock opgeheven wordt dan wordt er geprobeerd om de operatie opnieuw uit te voeren.

4.3.2 Serializable

- Ieder statement krijgt hetzelfde consistent beeld van de data als toen de transactie begon.

Als er een lock is op de data wordt er een error teruggegeven. Het programma moet terug naar het begin van de transactie en opnieuw beginnen.

4.3.3 Read only

- Verhinderd alle write operaties en voorziet een accuraat beeld van alle data als toen de transactie begon.

4.4 Oracle en concurrent user access

- Oracle lost dit probleem op via Multiversion read consistency (=**MVRC**).
- De gebruiker ziet een consistent beeld van de opgevraagde data.
    - Als er data wordt verandert, wordt er een nieuwe versie gecreëerd zodat de informatie van de eerste user hetzelfde blijft.
- De geretourneerde data zal enkel de gecommitte transacties van bij de start van de query reflecteren.

Dit heeft 2 gevolgen:

1. Geen lock op data voor read operaties

- Een read operatie blokkeert nooit een write operatie.

2. De gebruiker krijgt accurate info terug vanop het moment de query begon

4.5 Oracle concurrency features

4.5.1 Rollback segmenten

- Structuren die de "undo" informatie bevatten.
- Deze informatie kan de database herstellen naar de staat voor de transactie begon.
- Als een transactie data begint te veranderen wordt een image van de oude data weggeschreven in een rollback segement.
- Verschilt van de redo log

4.5.2 SCN

- logische tijdstempel die de volgorde van de transacties bijhoudt.
- Wordt gebruikt in de redo log om terug te keren naar een staat voor de transactie.
- Wordt gebruikt om overbodige rollback segmenten te verwijderen.

4.5.3 Locks in data blocks

Iedere lock wordt bijgehouden in het geheugen van de database.

4.6 Afhandelen van conflicterende write operaties

1. A zendt een update naar het server proces.
2. Server bekomt een SCN voor het statement en leest het data block die de rij bevat.
3. Server slaat row lock informatie op in het datablock.
4. Server schrijft veranderingen in de redo log bugger.
5. Server kopiëerd een image van de oude data naar een rollback segment.
6. B veranderd dezelfde data en stuurt een update naar de server.
7. Server haalt de SCN op en leest de data block met de data.
8. Server ziet dat er een lock is op het data block. Afhankelijk van het isolation level zal nu de gepaste actie ondernomen worden.
9. A commit de transactie en krijgt van de server confirmatie dat dit is gelukt.
10. Als het isolation level van B read committed was wordt de actie van B nu

uitgevoerd.

4.7 Afhandelen van read operaties

1. A zendt een select naar het server proces.
2. Server proces haalt SCN op voor het statement en begint de gevraagde data uit te lezen.

- Voor iedere gelezen block wordt de SCN van de select en de SCN's van iedere transactie vergeleken.
- Als er een later SCN is voor de select dan voor de transacties dan wordt er een consisten read versie gemaakt van de data.

3. B zendt een update naar het server proces naar een data block die nog niet uitgelezen is door A, maar die A wel nodig heeft.
4. B commit de veranderingen en het server proces voltooid deze operatie. Hierbij wordt data die nodig is om de SCN te bepalen ook toegevoegd.
5. Het server proces voor A komt aan het block dat net gewijzigd is door B. Het server proces ziet dat dat de SCN later is dan dat van het select statement dus gaat het zoeken in de header achter het rollback segment die de data bevat van toen het select statement begon.

4.8 Concurrent access en performance

Doordat Oracle geen gebruik maakt van locks is er de database contention zeer laag.