VI. Przetwarzanie rozproszone w SZBD Oracle

Celem zajęć jest zapoznanie się z własnościami SZBD Oracle umożliwiającymi przetwarzanie rozproszone. Własności te obejmują:

- Transparenty dostęp do zdalnych danych.
- Rozproszone transakcje.

1. Utworzenie baz danych

W celu przygotowania środowiska do ilustracji własności przetwarzania rozproszonego należy utworzyć 2 bazy danych. Wykonaj poniższe kroki aby utworzyć dwie bazy danych, jedną o nazwie *RBD1* i drugą o nazwie *RBD2*.

1. Zaloguj się do maszyny wirtualnej jako użytkownik rbd używając hasła RBD#7102.
2. Otwórz okno terminala, który nazwiemy terminalem pomocniczym.
3. Pobierz plik manifestów za pomocą poniższego polecenia:
   wget www.cs.put.poznan.pl/jjezierski/RBDv2/ora1.yaml
4. Otwórz plik manifestów w celu jego przeglądnięcia za pomocą polecenia:
   less ora1.yaml
1. Rozpocznij wdrożenie komponentów z pliku manifestów za pomocą polecenia:
   kubectl apply -f ora1.yaml
2. Obserwuj postęp wdrożenia *StatefulSet* wykorzystując poniższe polecenie:
   kubectl get sts --watch
   Wdrożenie zajmie chwilę ponieważ obraz kontenera ma objętość ponad 3GiB.
3. Skopiuj plik manifestów do pliku ora2.yaml:
   cp ora1.yaml ora2.yaml
4. Użyj swojego ulubionego edytora tekstu wykonać następujące zmiany w pliku ora2.yaml [Raport]:
   - zamień wszystkie wystąpienia tekstu rbd1 na rbd2,
   - ustal w drugim manifeście wartość klucza spec.ports.port na 1522.
5. Wykonaj wdrożenie zawartości pliku ora2.yaml [Raport].
5. Pobierz skrypty SQL wykonując w terminalu pomocniczym polecenia:
   wget www.cs.put.poznan.pl/jjezierski/RBDv2/pracownicy.sql
   wget www.cs.put.poznan.pl/jjezierski/RBDv2/zespoly.sql
6. W terminalu pomocniczym skopiuj plik pracownicy.sql do katalogu /opt/oracle/oradata repliki ora-rbd1-0. Użyj poniższego polecenia:
   kubectl cp pracownicy.sql ora-rbd1-0:/opt/oracle/oradata
7. W terminalu pomocniczym skopiuj plik zespoly.sql do katalogu /opt/oracle/oradata repliki ora-rbd2-0. Użyj poniższego polecenia:
   kubectl cp zespoly.sql ora-rbd2-0:/opt/oracle/oradata
8. Otwórz dwie kolejne zakładki terminala o nazwach `ora-rbd1-0` i `ora-rbd2-0`. Pierwszy terminal o nazwie *rbd1* będzie służył do operowania na bazie danych *rbd1*, natomiast drugi terminal o nazwie *rbd2* będzie wykorzystywany do wykonywania operacji na bazie danych *rbd2*.

9. W terminalu `ora-rbd1-0` wykonaj poniższe polecenie aby uruchomić powłokę w replice `ora-rbd1-0`. Replika ta będzie obsługiwać bazę danych, którą będziemy dalej nazywać rbd1.
   kubectl exec -it ora-rbd1-0 -- /bin/bash
10. W powłoce repliki zrób kopie zapasowe plików konfiguracyjnych:
    cp -R /opt/oracle/oradata/dbconfig /opt/oracle/oradata/dbconfig.bak
11. W terminalu *rbd1* dopisz do pliku *tnsnames.ora* deskryptor połączenia do bazy danych *rbd1* i *rbd2*:

```
cat <<EOT >> /opt/oracle/oradata/dbconfig/XE/tnsnames.ora
RBD1 =(DESCRIPTION = (ADDRESS_LIST =
 (ADDRESS = (PROTOCOL = TCP)(HOST = ora-rbd1-lb)(PORT = 1521)))
 (CONNECT_DATA =(SERVICE_NAME = XEPDB1)))
RBD2 =(DESCRIPTION = (ADDRESS_LIST =
 (ADDRESS = (PROTOCOL = TCP)(HOST = ora-rbd2-lb)(PORT = 1522)))
 (CONNECT_DATA =(SERVICE_NAME = XEPDB1)))
EOT
```

12. W terminalu *rbd1* sprawdź komunikację procesami nasłuchu (ang. listner) baz danych RBD1 i RBD2:
    tnsping RBD1
    tnsping RBD2
13. W terminalu rbd1 przetestuj za pomocą narzędzia sqlplus zdalne przyłączenie do bazy danych RBD2:
    sqlplus system/rbd2@RBD2
    Opuść narzędzie sqlplus za pomocą polecenia exit.
14. W terminalu rbd1 przyłącz się za pomocą narzędzia sqlplus do bazy danych RBD1:
    sqlplus system/rbd1@RBD1
15. W bazie danych *rbd1* utwórz użytkownika *rbd* z hasłem *RBD#7102*:
    create user rbd identified by "RBD#7102"
    quota unlimited on users;
    grant dba to rbd;
16. W bazie danych *rbd1* przełącz się na użytkownika *rbd*:
    connect rbd/RBD#7102@rbd1
17. Wykonaj odpowiednio zmodyfikowane kroki od 6 do 16 dla bazy danych RBD2. Zwróć uwagę, że loadbalancer dla usługi bazy danych RBD1 nasłuchuje na porcie 1521.

2. Transparentny dostęp do zdalnych danych

Celem tego punktu jest zaprezentowanie mechanizmów dostępu do zdalnych danych za pomocą łączników bazy danych oraz synonimów celu zapewnienia transparentnego dostępu do danych.

1. W bazie danych *rbd1* utwórz tabelę *pracownicy* uruchamiając skrypt w narzędziu *psql* poleceniem @ /opt/oracle/oradata/pracownicy.sql.
2. W bazie danych *rbd2* utwórz tabelę *zespoły* uruchamiając skrypt /opt/oracle/oradata/zespoly.sql.
3. Utwórz w bazie danych *rbd1* łącznik do bazy danych *rbd2* za pomocą poniższego polecenia:
   CREATE DATABASE LINK RBD@RBD2

CONNECT TO rbd IDENTIFIED BY RBD#7102
USING 'rbd2';

4. W bazie danych *rbd1* wykonaj poniższe polecenie testujące zdalny dostęp do tabeli *zespoly*:
   SELECT * FROM zespoly@RBD@RBD2;
5. W bazie danych *rbd1* utwórz synonim do zdalnej tabeli *zespoly:*
   CREATE SYNONYM zespoly FOR zespoly@RBD@RBD2;
6. W bazie danych *rbd1* przetestuj transparentny dostęp do tabeli *zespoly*:
   SELECT nazwisko, nazwa FROM pracownicy natural join zespoly;
7. Utwórz w bazie danych RBD2 odpowiednie obiekty umożliwiające zdalny dostęp do tabeli *pracownicy* znajdującej się w bazie danych RBD1. [Raport]

3. Rozproszone transakcje

Celem zadania jest przedstawienie rozproszonych transakcji, czyli takich transakcji, które modyfikują dane więcej niż w jednej bazie danych. System Oracle do zapewnienia atomowości operacji zatwierdzenia transakcji wykorzystuje Two Phase Commit (2PC) z pewną modyfikacją. Modyfikacja ta polega na wprowadzeniu wyróżnionego węzła zatwierdzania transakcji (ang. commit point site). Zapoznaj się z dokumentacją dotyczącą węzła zatwierdzania.

3.1. Rozproszona transakcja zakończona sukcesem

1. W terminalu *rbd1* za pomocą narzędzia *sqlplus* wykonaj w bazie danych RBD1 poniższą transakcję rozproszoną:
   update zespoly set adres='PIOTROWO 1' where id_zesp=10;
   update pracownicy set placa_pod=999 where id_prac=100;
   commit;
2. Sprawdź w terminalu rbd2 stan tabel *pracownicy* i *zespoly*. Czy posiadają one zmiany wprowadzone przez transakcję z poprzedniego punktu? [Raport]

3.2. Rozproszona transakcja zakończona awarią

System Oracle posiada narzędzie diagnostycznych w celu wywołania awarii podczas wykonywania operacji COMMIT. Do tego celu służy polecenie *commit comment 'ORA-2PC-CRASH-TEST-n'*, gdzie n przybiera następujące wartości:

1. Crash commit point site after collect
2. Crash non-commit point site after collect
3. Crash before prepare (non-commit point site)
4. Crash after prepare (non-commit point site)
5. Crash commit point site before commit
6. Crash commit point site after commit
7. Crash non-commit point site before commit
8. Crash non-commit point site after commit
9. Crash commit point site before forget
10. Crash non-commit point site before forget

1. W bazie danych *RBD1* wykonaj rozproszoną transakcję:
   update zespoly set adres='PIOTROWO 2' where id_zesp=10;
   update pracownicy set placa_pod=888 where id_prac=100;
   commit comment 'ORA-2PC-CRASH-TEST-4';
2. W bazie danych *RBD2* stan tabel *pracownicy* i *zespoly*. Czy posiadają one zmiany wprowadzone przez transakcję z poprzedniego punkty? Dlaczego? [Raport]
3. W bazie danych *RBD1* wykonaj rozproszoną transakcję:
   update zespoly set adres='PIOTROWO 3' where id_zesp=10;
   update pracownicy set placa_pod=777 where id_prac=100;
   commit comment 'ORA-2PC-CRASH-TEST-6';
4. W bazie danych *RBD2* stan tabel *pracownicy* i *zespoly*. Czy posiadają one zmiany wprowadzone przez transakcję z poprzedniego punktu? Dlaczego? [Raport]

3.3. Rozproszona transakcja zakończona awarią – ręczne odtwarzanie

W scenariuszach awarii z poprzedniego punktu, skutki awarii były automatycznie naprawiane. W taki sposób system będzie się zachowywał gdy zostanie przywrócona komunikacja między uczestnikami rozproszonej transakcji. W ramach poniższych scenariuszy awarii, komunikacja między uczestnikami transakcji nie będzie przywrócona i wystąpi potrzeba ręcznego odtworzenia transakcji.

1. W bazie danych *RBD1* ustaw parametr COMMIT_POINT_STRENGTH:
   alter system set COMMIT_POINT_STRENGTH=10 scope=spfile;
2. W bazie danych *RBD2* ustaw parametr COMMIT_POINT_STRENGTH:
   alter system set COMMIT_POINT_STRENGTH=20 scope=spfile;
3. Zrestartuj bazę danych *RBD1* wykonując w terminalu pomocniczym następujące polecenie:
   kubectl rollout restart sts ora-rbd1
4. Nawiąż ponownie połączenie do bazy danych RBD1 z wykorzystaniem terminala ora-rdb1-0 i programu sqlplus. Użyj danych uwierzytelnienia użytkownika rbd. Nie zapomnij o ustawieniu zmiennych środowiskowych.
5. Zastosuj odpowiednio poprzednie 2 punkty dla bazy danych RBD2.
6. Która baza danych będzie węzłem zatwierdzania rozproszonych transakcji? Dlaczego? [Raport]
7. W bazie danych *RBD1* wyłącz automatyczne odtwarzanie transakcji, które zakończyły się awarią, tę operację trzeba będzie wykonać w bazie danych XE, która jest kontenerem dla bazy danych RDB1:
   connect system/rbd1@XE
   ALTER SYSTEM DISABLE DISTRIBUTED RECOVERY;
   connect connect rbd/RBD#7102@rbd1
8. W bazie danych *RBD2* wyłącz automatyczne odtwarzanie transakcji.
9. W bazie danych *RBD1* wykonaj rozproszoną transakcję, zakończoną symulacją błędu w trakcie wykonywania algorytmu 2PC:
   update zespoly set adres='PIOTROWO 4' where id_zesp=10;
   update pracownicy set placa_pod=666 where id_prac=100;
   commit comment 'ORA-2PC-CRASH-TEST-4';
10. W bazie danych *RBD2* sprawdź informację o zespole 10. [Raport]

11. W bazie danych *RBD1* spróbuj zmodyfikować informację o pracowniku 100. (To już będzie nowa transakcja, poprzednia została zatwierdzona.) Co się stało? Dlaczego? [Raport]
12. Wycofaj nieudane zmiany z poprzedniego punktu.
13. W bazie danych *RBD1* wyświetl niedokończone rozproszone transakcje.
    select LOCAL_TRAN_ID, GLOBAL_TRAN_ID, STATE from DBA_2PC_PENDING;
14. W bazie danych *RBD1* wyświetl uczestników rozproszonej transakcji.
    select LOCAL_TRAN_ID, INTERFACE, DATABASE from DBA_2PC_NEIGHBORS;
15. Sprawdź w dokumentacji jakie dane są wyświetlane w kolumnie INTERFACE i co oznaczają w kontekście interesującej transakcji. [Raport]
16. Wykonaj punkty 13 i 14 w bazie danych *RBD2*.
17. Należy wycofać zmiany rozproszonej transakcji. Dlaczego? [Raport]
18. W bazie danych *rbd1* wycofaj lokalną część rozproszonej transakcji. Uwaga: użyj odpowiedniego identyfikatora lokalnej transakcji.
    rollback force '6.29.817';
19. Wykonaj punkty 13 i 14 jeszcze raz w bazie danych *RBD2*.
20. W bazie danych *RBD1* spróbuj zmodyfikować informację pracowniku 100. Co się stało? Dlaczego? [Raport]
21. Czy istnieje potrzeba wykonania kroku 17 w bazie danych RBD2? Dlaczego? [Raport]
22. Zrealizuj powyższy scenariusz dla awarii *Crash commit point site after commit*. [Raport]
23. W obu bazach danych włącz automatyczne odtwarzanie rozproszonych transakcji.