Przemysław Turek
Wydział Informatyki i Zarządzania
Wyższej Szkoły Informatyki w Łodzi

Promotor: dr hab. Adam Pelikant, prof. WSInf

REPLIKACJE W ORACLE

Streszczenie – W niniejszej pracy przedstawione zostały metody replikacji w bazach danych Oracle. Zostały przeprowadzone doświadczenia ilustrujące użyteczność replikacji prostej („snapshot" lub „materialized view") jak i typu multi-master oraz przedstawiające obraz okoliczności, w których poszczególne sposoby odświeżania sprawdzają się najlepiej. Z przeprowadzonych doświadczeń można też wywnioskować, że sam proces odświeżania w niewielkim stopniu zależy od łącza (niewielkie różnice w czasach odświeżania między siecią bezprzewodową 54Mbit/s, a siecią przewodową 100Mbit/s).

1. Wstęp

Replikacja danych jest to kopiowanie danych oraz utrzymanie wspólnego połączenia baz danych, które razem tworzą wspólne środowisko. Dane kopiowane są z jednej tabeli będącej źródłem danych do innej tabeli będącej miejscem docelowym. Także głównym celem przeprowadzania replikacji w bazach danych jest zintegrowanie wszystkich danych w jedną wspólną całość. Dzięki takiej operacji w różnych oddziałach firmy, np. bankach dane dla konkretnego klienta są identyczne. Każda operacja jest aktualizowana i możliwa do przeprowadzania w każdym oddziale firmy.

Dzięki replikacji zdecydowanie zmniejsza się czas dostępu do danych, ponieważ baza danych znajduje się w siedzibie konkretnego oddziału. Gdyby dla każdego oddziału banku była jedna baza danych, znajdująca się np. w głównej siedzibie firmy wymagająca każdorazowo łączenia, pobierania czy dodawania danych spowodowałoby to wydłużenie czasu dostępu oczekiwaniem w kolejce w przypadku dużej ilości połączeń. Innym zagrożeniem w takim przypadku byłyby awarie sieci łączące oddziały z siedzibą formy. Każda awaria na łączu

doprowadziłaby do uniemożliwienia załatwienia spraw w banku czy innej firmie.

Rozważmy inny problem, który rozwiązują replikacje. Jeżeli w oddziale firmy byłby pożar albo powódź i awarii uległyby serwery baz danych to dane zostaną utracone. Po wymianie sprzętu można dokonać replikacji danych z innych baz będących we wspólnym środowisku bazodanowym.

Niestety replikacja danych ma również swoje wady. Niezbędna jest aktualizacji replik, jeśli tabele źródłowe zostały zmienione, czyli synchronizacja wszystkich danych w bazach. W tym przypadku pojawiają się dwa problemy, w jaki sposób dokonać synchronizacji i kiedy ją przeprowadzić. Jeśli chodzi o sposoby synchronizacji to można ją dokonywać na dwa sposoby: synchroniczny i asynchroniczny. Replikacja synchroniczna polega na tym, że dane są aktualizowane zaraz po zmianie w tabelach źródłowych. Drugi rodzaj synchronizacji, czyli asynchroniczna może być przeprowadzona automatycznie przez system bądź na żądanie użytkownika korzystającego z baz danych (odświeżanie manualne). Jeśli synchronizację przeprowadza system, to należy określić mu, co jaki czas taka operacja ma się przeprowadzać, np. co 24 godziny. Odświeżanie replik może być pełne albo przyrostowe. Przy pełnym odświeżaniu kopiowana jest cała zawartość tabeli źródłowej. Natomiast przyrostowe polega na kopiowaniu danych z tabeli źródłowej zmienionych od momentu ostatniej synchronizacji.

2 Rodzaje replikacji

Bazy danych Oracle dopuszcza możliwość replikacji na wiele sposobów. Pierwszą metodą replikacji są migawki tylko do odczytu. Migawki tylko do odczytu realizują podstawową kopię tabeli źródłowej (występują także przypadki wykorzystywania zapytań wybierających tylko część danych z tabeli źródłowej). Tabela (baza danych) źródłowa nazywana jest tabelą (bazą danych) „master”. Cechą charakterystyczną prostych migawek tylko do odczytu jest to, że zapytanie wybierające może czerpać dane tylko z jednej tabeli. Może także używać tylko niektórych form podzapytań i funkcji specjalnych, na przykład funkcje agregujące nie są dozwolone. Złożona migawka może być wypełniana danymi z dowolnego zapytania. Najczęściej stosowane są migawki proste zasilane wszystkimi danymi istniejącymi w tym momencie w tabeli źródłowej. Migawki są okresowo wypełniane aktualnymi danymi. Proces ten nazywany jest „odświeżaniem” migawki. Innym rodzajem replikacji jest replikacja multi-master. Postaram się przedstawić cechy tego typu replikacji oraz zwrócić uwagę na zagadnienia związane z utrzymaniem tego typu konfiguracji. W replikacji typu multi-master można dokonywać replikacji nie tylko tabele, ale również indeksy, procedury, funkcje,

triggery, pakiety oraz typy zdefiniowane przez użytkownika, widoki, synonimy, operatory zdefiniowane przez użytkownika. Jak zwykle w takich przypadkach istnieją plusy i minusy wykorzystywania tego typu replikacji. Do pozytywnych aspektów wykorzystywania replikacji multi-master zaliczyć można replikowanie większej ilości obiektów (np. typy zdefiniowane przez użytkownika), możliwa jest zmiana struktury obiektów będących replikowanymi (dodanie kolumny do replikowanej tabeli po stronie „master", spowoduje zreplikowanie tej modyfikacji na pozostałe maszyny), możliwe jest replikowanie obiektów na dowolnej liczbie baz danych (dowolna maszyna „master" może replikować na inną maszynę typu „master", maszynę z migawkami tylko do odczytu lub migawkami z możliwością zmiany). Istnieją także negatywne aspekty wykorzystania tego typu replikacji. Należą do nich: wysokie zapotrzebowanie na przepustowość sieci (replikacja multi-master nie tylko wysyła i pobiera informacje z podłączonych baz danych, ale również wysyła potwierdzenia i całkiem dużą ilość danych administracyjnych), zmniejszona wydajność – replikacja multi-master wymaga użycia wyzwalaczy i procedur, które w zależności od ilości danych mogą wywołać negatywne skutki w postaci spadku wydajności. Kolejną wadą tego typu rozwiązania jest zwiększenie zaangażowania administratorów, a co za tym idzie zwiększa się podatność na „błędy ludzkie". Stosowanie replikacji multi-master może być problematyczne, gdy ilość transakcji na sekundę jest większa od 50 – 100 (w zależności od wydajności sieci i maszyny). Powyższe uwagi skłaniają do następującej refleksji: Nie należy wprowadzać replikacji na wyższym poziomie niż jest to konieczne. Przed dokonaniem wyboru replikacji należy przeanalizować, co będzie podlegało replikacji (jakie dane – często ze względu na ograniczenia przepustowości łącza, zmieniane są pierwotne założenia replikacji). W kolejnym kroku należy dokonać konfiguracji tabel (obiektów będących w obszarze zainteresowania). Należy upewnić się, czy replikowane tabele posiadają klucz podstawowy identyfikujący każdy z wierszy. Jest to zalecane ze względu na późniejsze rozwiązywanie problemów. W przypadku, gdy wiersze nie posiadają kolumny lub grupy kolumn, które można użyć do utworzenia klucza podstawowego, można skorzystać z sekwencji. Jednak wówczas powstaje problem integralności. Istnieją dwa rozwiązania. Pierwszym jest rozpoczynanie sekwencji od różnych wartości na kolejnych bazach danych dołączanych do grupy replikacyjnej (np. Na bazie danych DB1 sekwencja zaczynać się będzie od 1, na bazie danych DB2 sekwencja zaczynać się będzie od 100000).

```
CREATE SEQUENCE s1 START WITH 1; -- DB1
CREATE SEQUENCE s2 START WITH 100000; -- DB2
```

Takie rozwiązanie będzie prawidłowe dla pierwszych 99999 wierszy. Wiersz 100000 spowoduje powstanie konfliktu. Drugim rozwiązaniem jest rozpoczynanie sekwencji od tej samej liczby, lecz dodawanie do niej unikalnego dla danej bazy danych ciągu, np. Dla bazy danych DB1 sekwencja zaczynać się będzie od 1 lecz kluczem podstawowym będzie DB1_1, DB1_2 itd.… Analogicznie dla bazy danych DB2.

```
CREATE SEQUENCE db1_s1 START WITH 1; -- DB1
CREATE or REPLACE TRIGGER tr_db1
BEFORE INSERT
    ON example_table
    FOR EACH ROW
BEGIN
    :new.id := 'DB1_' || db1_s1.nextval;
END;
```

Takie rozwiązanie zapewnia wyeliminowanie ewentualnych konfliktów oraz umożliwia skalowalność (dodanie kolejnej bazy danych nie spowoduje konfliktu).

Kolejną ważną z punktu widzenia integralności danych rzeczą są klucze obce. W standardowych przypadkach jest tworzony indeks na kolumnach klucza obcego, aby nie wykonywać pełnego skanowania tabeli przy każdej zmianie tabeli rodzica. Dodając indeks klucza obcego do grupy replikacji bazy źródłowej nie ma potrzeby replikowania tabeli „dziecka" przy jednoczesnym zachowaniu integralności danych (zapewnionym przez klucze główne i obce). Zmiana danych w tabeli dziecku spowoduje zmianę indeksu klucza obcego, a ten zostanie zreplikowany na inne strony biorące udział w replikacji. Jeśli natomiast replikowana jest tabela dziecko, konieczne jest replikowanie tabeli rodzica w celu zachowania integralności.

Istnieją dwa podejścia do korzystania z replikacji multi-master: synchroniczne i asynchroniczne.

Do zalet podejścia synchronicznego zaliczyć można:

- Gwarancję pełnej synchronizacji pomiędzy stronami replikacji w dowolnym punkcie czasowym. Może być to dobre podejście do zastosowania przy budowie rozproszonych aplikacji. W środowiskach rozproszonych poprawność danych jest krytycznym czynnikiem. Mała ilość operacji DML (INSERT/UPDATE/DELETE) powoduje, że czas odpowiedzi nie stanowi problemu.
- Nie ma możliwości konfliktów. Podejście synchroniczne korzysta z zatwierdzania dwufazowego (two phase commit).

Wady podejścia synchronicznego to:

- Z powodu wykorzystywania zatwierdzania dwufazowego, jeśli jedna ze stron nie jest dostępna zmiana danych może nie dojść

do skutku. Ta metoda nie może zostać użyta w środowiskach wysokiej dostępności.
- Przy dużej ilości operacji DML powstają opóźnienia spowodowane oczekiwaniem (two phase commit).

Zalety podejścia asynchronicznego:
- Dzięki wykorzystaniu kolejek zmian, zmaterializowanych kolejek, logów migawek możliwe jest przechowywanie historii zmian, a co za tym idzie przy niedostępnej bazie docelowej, pozostałe mogą być używane. Co więcej, gdy baza stanie się dostępna, nastąpi synchronizacja danych (czas, w jakim zostanie to wykonane zależy od ustawień na serwerze źródłowym – okres opróżnienia zapobiega przepełnieniu kolejki po stronie źródłowej przez nie spropagowane dane).

Wady rozwiązania asynchronicznego:
- Główną wadą jest prawdopodobieństwo wystąpienia konfliktów DML. Kiedy następuje konflikt, te same dane są zmieniane w dwóch (lub więcej) bazach danych w przedziale czasu pomiędzy propagacją zmiany do innych baz danych. Nie istnieje technika eliminująca występowanie konfliktów, jak również pozwalająca na 100% rozwiązanie konfliktu.
- Kolejna wada wynika z zastosowania dbms_job, co jak wiadomo jest podatne na błędy, a także nie jest proste w administracji.

3 Konfigurowanie środowisk

W opisywanym projekcie wykorzystywane są dwie lokalizacje (fizyczne komputery):

Tabela. 1. Opis konfiguracji komputerów

Nazwa komputera	*Konfiguracja*
F1	System operacyjny: Linux Fedora 9. Baza danych: Oracle Database 11g Enterprise Edition Release 11.1.0.6.0
F2	System operacyjny: Windows XP Home Edition. Baza danych: Oracle Database 11g Enterprise Edition Release 11.1.0.6.0

F1 jest to lokalizacja „master”, czyli źródłowa baza danych zawierająca dane, na których możliwe są operacje takie jak wstawianie czy kasowanie (ingerujące w dane). Danymi w „naturalnych” warunkach manipuluje system wykorzystywany w przedsiębiorstwie (np. system sprzedażowy). W projekcie wykorzystywane są cztery tabele:

Tabela. 2. Opis zawartości tabel

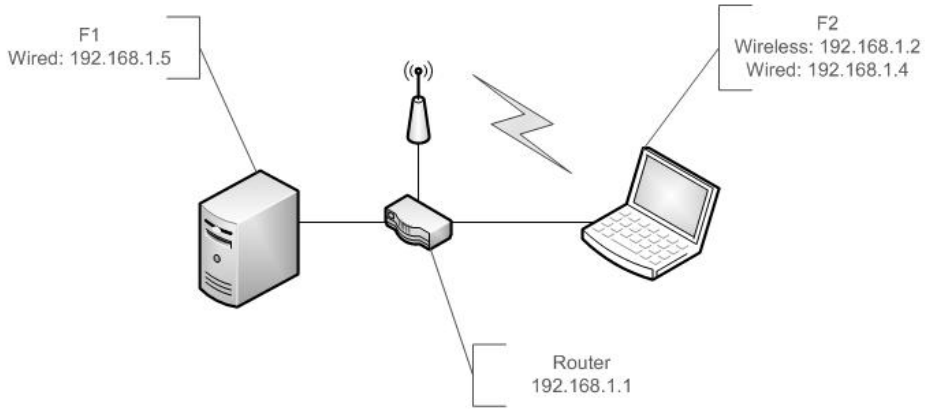
Nazwa tabeli	*Opis*
Clients	Tabela przechowująca dane klientów. Wypełniona losowymi danymi (10000 wierszy).
Items	Tabela przechowująca informacje na temat towarów. Wypełniona losowymi danymi (200000 wierszy)
Orders	Tabela przechowująca informacje o zamówieniach. Każdy wiersz określa zamówienie i powiązanego z tym zamówieniem klienta. Tabela wypełniona losowymi danymi (dla jednego klienta możliwe jest maksymalnie 10 wpisów, więc maksymalny rozmiar tej tabeli to 10000*10 = 100000 wpisów)
Order_items	Tabela przechowująca informacje o pozycjach (towarach) znajdujących się na zamówieniach. Każdy wiersz określa towar oraz zamówienie, na rzecz którego został wpisany. Tabela wypełniona jest losowymi danymi (dla jednego zamówienia możliwe jest maksymalnie 10 wpisów, więc maksymalny rozmiar tej tabeli to 100000*10 = 1000000 wpisów)

Na potrzeby replikacji w każdej bazie danych stworzeni zostali użytkownicy „repadmin” mający dostęp do obiektów replikacji. We właściwym systemie bazodanowym użytkownicy ci powinni ze względów bezpieczeństwa mieć dostęp jedynie do schematów objętych replikacją w danej lokalizacji („master” lub „snapshot”). W celu usprawnienia procesu przygotowania środowiska testowego stworzeni użytkownicy „repadmin” mają nadane uprawnienia do dowolnej replikowanej grupy obiektów („GRANT_ADMIN_ANY_SCHEMA”). Skrypt tworzący użytkownika repadmin i nadający mu odpowiednie uprawnienia znajduje się w rozdziale 3.1.2.1. Możliwe jest tworzenie grup replikacji, czyli grup

obiektów, które mają być replikowane w celu ułatwienia zarządzania powiązanymi obiektami bazy danych. Łączone są wówczas obiekty, które zwykle są dostępne dla danej bazy danych. Grupa replikacyjna może istnieć w wielu lokalizacjach replikacyjnych. Analogicznie do pojedynczych obiektów replikacyjnych grupy replikacji mogą istnieć w dwóch podstawowych lokalizacjach:

- „master" zawiera kompletną wersję wszystkich obiektów w grupie replikacyjnej;
- „snapshot" obsługuje proste migawki tylko do odczytu oraz migawki z możliwością zmiany danych powiązane z lokalizacją „master". Migawki w lokalizacjach „snapshot" mogą zawierać wszystkie lub niektóre z elementów grupy replikacyjnej.

Rysunek 1 przedstawia strukturę sieci wykorzystanej do przeprowadzenia doświadczeń.



Rys. 1. Konfiguracja sieci do prowadzenia testów

Sieć składa się z jednego routera bezprzewodowego obsługującego standard 802.11 b/g (maksymalna prędkość przesyłu danych to 54 Mbity/s). Router ten posłużył do połączenia komputera nazwanego na potrzeby doświadczeń „F1" z komputerem „F2". Przy czym komputer „F2" ma możliwość korzystania z dwóch połączeń sieciowych:

- Przewodowego 100 Mbit/s (oznaczonego na powyższym diagramie, jako Wired) o adresie IP: 192.168.1.4;
- Bezprzewodowego 54 Mbit/s (oznaczonego na powyższym diagramie jako Wireless) o adresie IP: 192.168.1.2.

4 Konfigurowanie baz danych

Na potrzeby projektu serwer master został oznaczony, jako „F1”. Bazę stanowiącą źródło danych nazywamy „master”. Poniższe kroki przedstawiają sposób przygotowania zarówno samego serwera (systemu operacyjnego) jak i bazy danych:

- przygotowanie pliku tnsnames.ora;
- utworzenie użytkownika repadmin;
- nadanie odpowiednich uprawnień użytkownikowi repadmin;
- utworzenie połączenia z bazą „snapshot” („F2”);
- utworzenie nowej przestrzeni tabel;
- utworzenie struktury tabel testowych i wypełnienie ich przykładowymi danymi;
- utworzenie logu migawki na tabelach testowych z użyciem utworzonej przestrzeni tabel.

Aby z poziomu bazy danych można było wykonywać wyrażenia DML na zdalnej bazie danych niezbędne jest utworzenie połączenia między bazami. Będzie ono omówione w dalszej części. Utworzenie takiego połączenia jest możliwe jedynie w przypadku, gdy zdalna baza danych jest „znana”, tzn. znane są parametry pozwalające na uzyskanie połączenia z tą bazą danych. Plik ten należy zmienić tak, aby zawierał następujące wpisy:

```
LISTENER_ORCL =
  (ADDRESS = (PROTOCOL = TCP)(HOST = F1)(PORT =
1522))


F1 =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = F1)(PORT =
1522))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = orcl)
    )
  )

F2 =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = F2)(PORT =
1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
```

```
        (SERVICE_NAME = orcl)
      )
    )
```

Zapewnią one możliwość połączenia z serwera „F1” do baz danych znajdujących się zarówno na lokalnej maszynie, jak i na maszynie nazwanej „F2”. Połączenia między bazami danych określane są jako dblink. Połączenia takie pozwalają na wykonywanie zapytań DML na zdalnej (odległej) bazie danych. Poniższe zapytanie tworzy połączenie z serwerem „F2” o nazwie „F2”. Nazwa ta będzie wykorzystywana w zapytaniach korzystających z obiektów znajdujących się na zdalnej maszynie.

```
CREATE DATABASE LINK F2 CONNECT TO repadmin
IDENTIFIED BY repadmin using 'F2';
```

Na potrzeby projektu serwer docelowy został oznaczony jako „F2”. Bazę docelową nazywamy „snapshot”. Tabele w lokalizacji „master” zawierają dane, które podczas odświeżenia migawek przesyłane są do lokalizacji „snapshot”. Odświeżenie migawek może nastąpić w sposób automatyczny (w odstępach czasu określonych podczas tworzenia migawki). Poniższe kroki przedstawiają sposób przygotowania zarówno samego serwera (systemu operacyjnego) jak i bazy danych:

- przygotowanie pliku tnsnames.ora;
- utworzenie użytkownika repadmin;
- nadanie odpowiednich uprawnień użytkownikowi repadmin;
- utworzenie połączenia z bazą „master” („F1”);
- utworzenie przestrzeni tabel;
- utworzenie struktury tabel testowych (utworzenie migawek na podstawie tabel na serwerze „F1”);
- „ręczne” odświeżenie migawek.

Plik tnsnames.ora powinien umożliwiać połączenie z serwerem „F1” (zawierającym dane źródłowe). Na serwerach „F1” i „F2” pliki tnsnames.ora są identyczne (usługi są rozwiązywane za pośrednictwem nazw serwerów). Połączenia między bazami danych określane są jako dblink. Połączenia takie pozwalają na wykonywanie zapytań DML na zdalnej (odległej) bazie danych. Poniższe zapytanie tworzy połączenie z serwerem „F1” o nazwie „F1”. Nazwa ta będzie wykorzystywana w zapytaniach korzystających z obiektów znajdujących się na zdalnej maszynie.

```
CREATE DATABASE LINK F1 CONNECT TO repadmin
IDENTIFIED BY repadmin using 'F1';
```

Po stronie bazy docelowej “snapshot” tworzone są migawki (inaczej perspektywy zmaterializowane) będące odwzorowaniem tabel znajdujących się po stronie źródłowej bazy danych. Każda z tabel źródłowych posiada swój odpowiednik po stronie bazy docelowej. Dla potrzeb doświadczenia utworzone zostały jedynie migawki proste, czyli bazujące na jednej tabeli, ponadto zapytanie generujące daną migawkę nie zawiera agregatów, operatora DISTINCT, podzapytań, operacji na zbiorach, klauzul GROUP BY oraz CONNECT BY tak jak operacji połączenia. Takie ograniczenia pozwalają na wykonywanie odświeżeń tych migawek w trybie przyrostowym (FAST). Podczas tworzenia migawek możliwe jest podanie przestrzeni tabel zarówno dla migawki jak i dla indeksu (podobnie jak dla tabel). Poniżej, na podstawie tabeli items pokazano metodę tworzenia migawek tak, aby migawki znajdowały się w przestrzeni tabel tab oraz wykorzystywały indeksy umieszczając je w przestrzeni tabel idx. Podobnie należy postąpić dla wszystkich replikowanych tabel.

```
CREATE SNAPSHOT items
    TABLESPACE idx
 AS SELECT * FROM items@F1;
```

Podczas tworzenia migawek następuje automatyczne ich odświeżenie (czas tworzenia migawek jest zależny od ilości danych). Odświeżenie migawki polega na zaktualizowaniu danych w lokalizacji docelowej danymi (aktualnymi na moment odświeżenia) z lokalizacji źródłowej. Odświeżenia mogą odbywać się w sposób asynchroniczny: w ustalonych odstępach czasu (definiowane jest to podczas tworzenia migawki), na żądanie użytkownika (manualnie) lub w sposób asynchroniczny: natychmiast po modyfikacji danych źródłowych. W celu obserwacji procesu odświeżania wykorzystywana będzie metoda manualna. Manualne odświeżanie migawki realizowane jest za pomocą procedury REFRESH znajdującej się w pakiecie dbms_snapshot. Poniżej znajduje się składnia polecenia używanego do odświeżania migawek.

```
EXECUTE dbms_snapshot.refresh(nazwy_migawek,
rodzaj_odświeżenia);
```

Jako rodzaj_odświeżenia możliwe są następujące wartości:

- C lub c – odświeżenie pełne (Complete);
- F lub f – odświeżenie przyrostowe (Fast);
- ? – odświeżenie określone w definicji migawki.

Istnieje również możliwość odświeżenia więcej niż jednej migawki za pomocą jednego polecenia, wówczas w miejsce nazwa_migawki należy wpisać listę migawek oddzielonych przecinkami, natomiast w miejscu rodzaj_odświeżenia należy wpisać metody, z jakimi mają wykonać się poszczególne odświeżenia (bez separatorów). Poniżej znajduje się przykład odświeżenia dwóch tabel, pierwszej w trybie przyrostowym, drugiej w trybie pełnym.

5 Konfiguracja dla różnych modeli

W celu przygotowania odpowiednich użytkowników do modelu niezaufanego, należy przygotować użytkowników repadmin i repprop tak jak w przypadku modelu zaufanego. Następnie należy utworzyć grupy replikacji i odbiorcę („receiver”). Bardzo ważne jest, aby w momencie przypisywania ról istniały grupy replikacji. Tworzenie grup replikacji (nazwy i połączenia przedstawione zostały w rozdziale “Model niezaufany”):

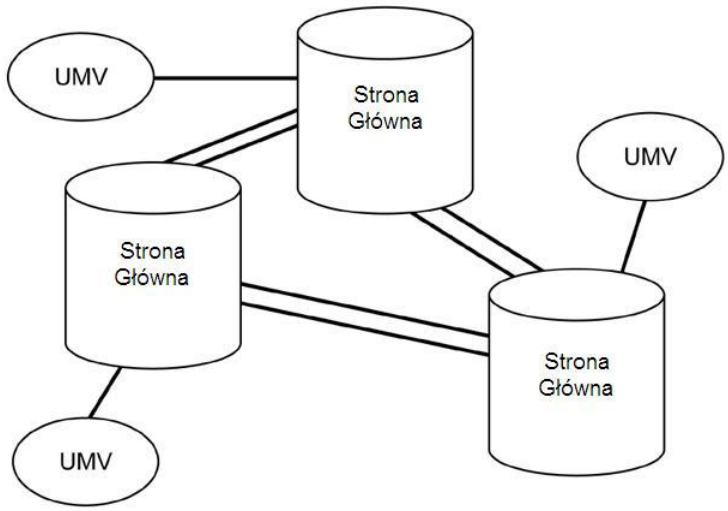
```
execute dbms_repcat.create_master_repgroup(
gname=> 'REP_GROUP2');
execute dbms_repcat.create_master_repgroup(
gname=> 'REP_GROUP3');
```

W tym przypadku replikowany będzie schemat TAB. Należy zatem utworzyć połączenia pomiędzy bazami danych. W przypadku replikacji multi-master należy utworzyć połączenia w każdym kierunku (podczas gdy replikacja za pomocą migawek wymagała połączenia jedynie w jednym kierunku).. W przypadku korzystania z modelu zaufanego i niezaufanego, należy ponadto utworzyć połączenia z lokalnego propagatora do odbiorcy po stronie zdalnej.

```
connect repprop/repprop@F1
-- PO STRONIE F1
create database link F2 connect to reprecv
identified by reprecv using ’F2’;
```

```
connect repprop/repprop@F2;
-- PO STRONIE F2
create database link F1 connect to reprecv
identified by reprecv using 'F1';
```

Jest wystarczającym aby każda strona główna (“master site”) była połączona z jedną stroną główną. Nie jest konieczne tworzenie połączeń „każdy-z-każdym”. Wszystkie zmiany, jakie zostały dokonane lokalnie mają zastosowanie po stronie lokalnej ale jednocześnie umieszczane są w kolejce transakcji „odroczonych”. Na poniższym rysunku przedstawione są połączenia pomiędzy stronami głównymi („master sites”). Połączenia te są jednostronne. Oznaczenia UMV oznacza „Updatable Materialized View” czyli migawkę z możliwością zmian.



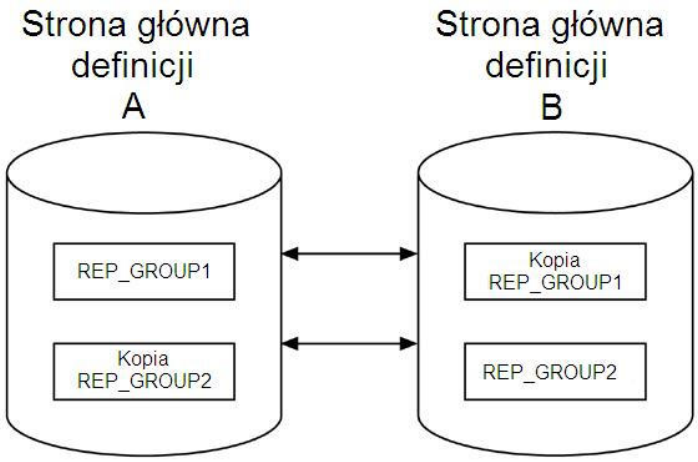
Rys. 2. Konfiguracja replikacji dla modelu niezaufanego

Potrzebne jest zadanie, które pozwoli na zastosowanie zmian czekających w kolejce na inne strony główne. Zadanie takie musi być uruchamiane z pewną częstotliwością (zależną od typu i rozmiaru danych oraz typu aplikacji). Konieczne jest również stworzenie zadania usuwającego zmiany z kolejki transakcji „odroczonych”, po tym jak zostaną one przesłane do odpowiednich stron głównych. Powyższe dwa zadania realizowane będą poprzez mechanizm Oracle Jobs. W celu

zwiększenia efektywności zadania zostały rozdzielone. Zadanie wysyłające zmiany musi działać najszybciej i najbardziej efektywnie jak to tylko możliwe, natomiast zadanie usuwające niepotrzebne wpisy z kolejki powinno być uruchamiane na tyle często, aby umożliwiać zarządzanie kolejką. Poniżej przedstawiono sposób zarejestrowania zadania wysyłającego („schedule_push") oraz czyszczącego („schedule_purge").

```
-- Po stronie F1
connect repadmin/repadmin@F1
begin
 dbms_defer_sys.schedule_push(
 destination => 'F2',
 interval => 'SYSDATE + 1/(60*24)',
 next_date => sysdate,
 stop_on_error => FALSE,
 delay_seconds => 0,
 parallelism => 1);
end;
/
begin
 dbms_defer_sys.schedule_purge(
 next_date => sysdate,
 interval => 'sysdate + 1/24',
 delay_seconds => 0,
 rollback_segment => '');
end;
/
-- Po stronie F2
connect repadmin/repadmin@F2
begin
 dbms_defer_sys.schedule_push(
```

```
 destination => 'F1',
 interval => 'SYSDATE + 1/(60*24)',
 next_date => sysdate,
 stop_on_error => FALSE,
 delay_seconds => 0,
 parallelism => 1);
end;
/
begin
 dbms_defer_sys.schedule_purge(
 next_date => sysdate,
 interval => 'sysdate + 1/24',
 delay_seconds => 0,
 rollback_segment => '');
end;
```



Rys. 3. Konfiguracja replikacji o dwóch stronach głównych

Następnie należy zdefiniować stronę główną definicji („master definition site”) na F1 i dokonać replikacji na F2. Można to wykonać korzystając z linii poleceń lub z narzędzia Oracle Enterprise Manager.

Strona główna definicji jest „podstawą” dla grupy głównej. Zawiera obiekty, które mają być replikowane na strony zdalne („remote sites”). W tym przypadku stroną główną definicji będzie F1, natomiast stroną zdalną będzie F2. Raz utworzona strona główna definicji będzie automatycznie replikowała się na nowe strony główne. Może jednak istnieć tylko jedna strona główna definicji dla danego replikowanego obiektu. Wszystkie zmiany na replikowanym obiekcie są stosowane na stronie głównej definicji, po czym są propagowane do innych stron głównych. W środowisku replikacyjnym może być więcej niż jedna strona główna definicji. Przykład poniżej:

Rysunek 2 ilustruje możliwość istnienia dwóch głównych stron definicji, ale na różnych obiektach. Na przykład schemat PUBS może być replikowany poprzez grupę REP_GROUP1 ze strony głównej definicji A, natomiast schemat SCOTT może być replikowany przez grupę REP_GROUP2 ze strony głównej definicji B. Główna grupa replikacji zawiera obiekty, które będą replikowane. Jeśli grupa główna ma wspierać również widoki zmaterializowane z możliwością zmian, to należy upewnić się, że logi widoków zmaterializowanych istnieją na tych tabelach przed dodaniem ich do grupy głównej. Poniżej znajdują się instrukcje umożliwiające stworzenie głównej grupy replikacji.

```
connect repadmin/repadmin@F1
BEGIN
 DBMS_REPCAT.CREATE_MASTER_REPGROUP(
 gname => '"GROUP1"',
 qualifier => '',
 group_comment => '');
END;
/
/* Do głównej grupy replikacji należy dodać
tabele, które mają zostać zreplikowane. W tym
przypadku są to tabele: items oraz clients.
Poniżej znajduje się polecenie dodające te tabele
(znajdują się one w schemacie PUBS).*/
connect repadmin/repadmin@F1
BEGIN
 DBMS_REPCAT.CREATE_MASTER_REPOBJECT(
```

```
 gname => '"GROUP1"',
 type => 'TABLE',
 oname => '"ITEMS"',
 sname => '"TAB"');
END;
/
BEGIN
 DBMS_REPCAT.CREATE_MASTER_REPOBJECT(
 gname => '"GROUP1"',
 type => 'TABLE',
 oname => '"CLIENTS"',
 sname => '"TAB"');
END;
```

W przypadku korzystania z tabel, które nie mają zdefiniowanego klucza głównego należy użyć funkcji SET_COLUMNS. Oto sposób użycia tej procedury w odniesieniu do tabeli autor w schemacie PUBS:

```
BEGIN
 DBMS_REPCAT.SET_COLUMNS(
 sname => '"PUBS"',
 oname => '"AUTOR"',
 column_list => '"AUTOR_KEY"');
END;
```

Odpowiada to zdefiniowaniu ekwiwalentu (substytutu) klucza głównego. Będzie on występował tak długo jak długo będzie występowała definicja replikacji. W przypadku naruszenia tego klucza wystąpi konflikt, który trzeba będzie rozwiązać. Należy zauważyć, że nie jest to typowy klucz główny. Nie jest tworzony indeks i możliwe jest wstawienie duplikujących się rekordów do tabeli. Wskazanym jest zatem tworzenie standardowych kluczy głównych kiedy tylko jest to możliwe.

6 Część eksperymentalna – badanie wydajności

Eksperyment polega na utworzeniu odświeżeniu migawek prostych wykorzystując istniejącą infrastrukturę sieciową (połączenie przewodowe oraz bezprzewodowe). Dodatkowym doświadczeniem będzie zasymulowanie przerwania łączności podczas synchronizacji (odświeżania) tabel. W celu przeprowadzenia eksperymentu wykorzystane zostały skrypty, dzięki którym możliwe było wielokrotne powtarzanie danej części w celu uzyskania miarodajnych wyników. Doświadczenia można podzielić na następujące elementy składowe:

- usuwanie i tworzenie migawek prostych (sieć bezprzewodowa i przewodowa - porównanie);
- częściowe odświeżanie migawek (sieć bezprzewodowa i przewodowa - porównanie);
- całkowite odświeżanie migawek (sieć bezprzewodowa i przewodowa - porównanie);
- symulacja przerwania łączności podczas odświeżania migawek.

Analogiczne doświadczenia zostaną przeprowadzone dla replikacji multi-master. Replikacja tego typu może odbywać się z wykorzystaniem wielu stron głównych jednak w tym przypadku wykorzystana będzie istniejąca infrastruktura (dwie bazy danych). Dane z tabel wykorzystywanych w poprzednim doświadczeniu (replikacja prosta) posłużą również do przeprowadzenia testów multi-master. Podobnie jak dla replikacji prostej tabele zostaną wypełnione danymi (z wykorzystaniem tych samych skryptów), po czym nastąpi replikacja na drugą bazę danych. Doświadczenie można podzielić na następujące części:

- usuwanie i tworzenie grup replikacji z wypełnionymi tabelami (sieć bezprzewodowa i przewodowa – porównanie);
- częściowa replikacja grupy obiektów (sieć bezprzewodowa i przewodowa - porównanie);
- całkowita replikacja grupy obiektów (sieć bezprzewodowa i przewodowa – porównanie;)
- symulacja przerwania łączności podczas replikacji;
- replikacja grup obiektów z F1 na F2 i jednocześnie z F2 na F1 (replikacja w dwóch kierunkach jednocześnie).

Ta część doświadczenia polegała na naprzemiennym usuwaniu i tworzeniu migawek z tabel źródłowych. Tabele źródłowe w momencie tworzenia migawek wypełnione były danymi. Poniższa tabela przedstawia uśrednione (do 1 sek.) czasy usuwania i tworzenia migawek z wykorzystaniem infrastruktury przewodowej oraz bezprzewodowej.

Tabela. 3. Uśrednione czasy usuwania i tworzenia migawek

Tabela	*Czas odświeżenia (połączenie przewodowe) [sek.] usuwanie/tworzenie*	*Czas odświeżenia (połączenie bezprzewodowe) [sek.] usuwanie/tworzenie*
Items	82/17	82/18
Clients	73/5	73/5
Orders	92/21	92/23
Order_items	95/51	95/55

Ta część doświadczenia polegała na naprzemiennym usuwaniu i tworzeniu grup replikacji złożonych z tabel źródłowych. Tabele źródłowe w momencie tworzenia wypełnione były danymi. Poniższa tabela przedstawia uśrednione (do 1 sek.) czasy usuwania i tworzenia grup z wykorzystaniem infrastruktury przewodowej oraz bezprzewodowej.

Tabela. 4. Uśrednione czasy usuwania i tworzenia grup

Tabela	*Czas odświeżenia (połączenie przewodowe) [sek.] usuwanie/tworzenie*	*Czas odświeżenia (połączenie bezprzewodowe) [sek.] usuwanie/tworzenie*
Items	84/18	84/19
Clients	73/6	73/6
Orders	95/23	95/24
Order_items	98/53	98/55

Ta część doświadczenia polegała na częściowym odświeżaniu migawek zwiększając liczbę rekordów w tabelach źródłowych. Tabele źródłowe w momencie tworzenia migawek nie były wypełnione danymi. Poniższa tabela przedstawia uśrednione czasy częściowego odświeżania migawek z wykorzystaniem infrastruktury przewodowej oraz bezprzewodowej dla ¼ , ½ oraz wszystkich danych.

Tabela. 5. Uśrednione czasy częściowego odświeżania migawek

Tabela	*Czas odświeżenia (połączenie przewodowe) [sek.]*	*Czas odświeżenia (połączenie bezprzewodowe) [sek.]*
¼ ilości danych		
Items	2	2
Clients	1	1
Orders	2	2
Order_items	3	3
½ ilości danych		
Items	6	8
Clients	4	4
Orders	24	27
Order_items	14	15
wszystkie dane		
Items	10	12
Clients	6	6
Orders	49	55
Order_items	58	68

Ta część doświadczenia polegała na częściowym odświeżaniu migawek zwiększając liczbę rekordów w tabelach źródłowych. Tabele źródłowe w momencie tworzenia migawek nie były wypełnione danymi. Poniższa tabela przedstawia uśrednione czasy częściowego odświeżania migawek z wykorzystaniem infrastruktury przewodowej oraz bezprzewodowej dla ¼ , ½ oraz wszystkich danych.

Tabela. 6. Uśrednione czasy częściowego odświeżania migawek z wykorzystaniem infrastruktury

Tabela	***Czas odświeżenia (połączenie przewodowe) [sek.]***	***Czas odświeżenia (połączenie bezprzewodowe) [sek.]***
¼ ilości danych		
Items	2	2
Clients	1	1
Orders	2	2
Order_items	3	3
½ ilości danych		
Items	6	8
Clients	4	4
Orders	25	27
Order_items	16	18
wszystkie dane		
Items	11	12
Clients	6	6
Orders	51	56
Order_items	61	72

Ta część doświadczenia polega na całkowitym odświeżaniu migawek z tabel źródłowych. Tabele źródłowe w momencie tworzenia migawek wypełnione były danymi. Poniższa tabela przedstawia uśrednione czasy całkowitego odświeżania migawek z wykorzystaniem infrastruktury przewodowej oraz bezprzewodowej dla wszystkich danych.

Tabela. 7. Uśrednione czasy całkowitego odświeżania migawek z wykorzystaniem infrastruktury

Tabela	*Czas odświeżenia (połączenie przewodowe) [sek.]*	*Czas odświeżenia (połączenie bezprzewodowe) [sek.]*
Items	8	9
Clients	5	5
Orders	42	48
Order_items	52	61

Ta część doświadczenia polega na całkowitym odświeżaniu obiektów znajdujących się na jednej stronie głównej na drugą stronę główną. Tabele źródłowe w momencie przeprowadzania replikacji wypełnione były danymi. Poniższa tabela przedstawia uśrednione czasy całkowitego odświeżania obiektów z wykorzystaniem infrastruktury przewodowej oraz bezprzewodowej dla wszystkich danych.

Tabela. 8. Uśrednione czasy całkowitego odświeżania obiektów z wykorzystaniem infrastruktury

Tabela	*Czas odświeżenia (połączenie przewodowe) [sek.]*	*Czas odświeżenia (połączenie bezprzewodowe) [sek.]*
Items	8	9
Clients	6	6
Orders	44	49
Order_items	55	63

Ta część doświadczenia polega na całkowitym odświeżaniu migawek z tabel źródłowych (lub przeprowadzeniu replikacji multi-master), jednak podczas odświeżania następuje przerwanie połączenia ze zdalnym serwerem (na którym znajdują się tabele źródłowe). Tabele źródłowe w momencie rozpoczęcia doświadczenia wypełnione były danymi.

Podczas wykonania odświeżenia następuje przerwa w komunikacji (awaria routera, przecięcie kabla, itp.). Wówczas replikacja nie dochodzi do skutku. Dane, które zostały już przesłane nie zostają wprowadzone do tabeli docelowej. Jeśli odświeżenia wykonywane są z poziomu zadań

Oracle (Oracle jobs), to są one automatycznie ponawiane w określonych odstępach czasu i określoną ilość razy. Po wyczerpaniu limitu powtórzeń zadanie takie jest oznaczane, jako niepowodzenie. W przypadku ręcznego uruchamiania odświeżania nie istnieją mechanizmy pozwalające na powtórzenie procesu odświeżenia.

7 Monitorowanie replikacji

Ze względu na swoją złożoność i ogromne możliwości konfiguracji zadania związane z monitorowaniem replikacji zazwyczaj sprawiają najwięcej problemów. Akcje podejmowane w celu wyeliminowania tych problemów przez firmy, w których niezbędna jest replikacja zakrojone są na bardzo szeroką skalę. Często zostaje do tego celu wyznaczona osoba, której zadaniem jest monitorowanie i utrzymywanie wysokiego poziomu niezawodności replikacji. W dłuższym rozrachunku inwestycja w czas przeznaczony na konfigurację i wdrożenie replikacji oraz utrzymywanie dodatkowych zasobów ludzkich jest opłacalna (biorąc pod uwagę straty, na jakie narażona zostałaby firma funkcjonując bez odpowiednich administratorów, konfiguracji czy też replikacji).

Narzędziem ułatwiającym monitorowanie i ewentualne usuwanie problemów jest replication suport. Poniżej znajduje się przykład utworzenia wsparcia replikacji dla tabeli ORDERS. Zostaną utworzone obiekty, procedury, wyzwalacze i inne, wymagane do wsparcia replikacji. Analogiczne działania należy wykonać dla pozostałych artykułów replikacji.

```
BEGIN
DBMS_REPCAT.GENERATE_REPLICATION_SUPPORT(
sname => '"TAB"',
oname => '"ORDERS"',
type => 'TABLE',
min_communication => TRUE,
generate_80_compatible => FALSE);
END;
```

W powyższych przykładach parametry min_communication oraz generate_80_compatible odpowiadają za kompatybilność wsteczną replikacji. Jeśli którakolwiek baza danych strony głównej jest w wersji 7.3, wówczas parametr min_communication powinien przyjąć wartość FALSE. Jeśli wszystkie bazy danych stron głównych są w wersji 8.1.5 lub wyższej, wówczas parametr generate_80_compatible powinien

przyjąć wartość TRUE. W widoku dba_repcatlog możliwe jest sprawdzenie ewentualnych błędów. Jeśli ilość wierszy w tym widoku nie zmniejsza się (powinno być 0), lub zmniejsza się wolno, oznaczać to może:

- ustawienie parametru JOB_QUEUE_INTERVAL na zbyt dużą wartość lub brak zdefiniowania tego parametru;
- niepoprawne zdefiniowanie zadania wysyłającego „push job” lub brak jego definicji;
- okresy pomiędzy wykonaniami zadania wysyłającego są zbyt duże;
- nie ma połączeń pomiędzy bazami danych (dblink).

W calu uzyskania informacji o liczbie zapytań administracyjnych, błędów występujących podczas zapytań administracyjnych, liczbie opóźnionych niespropagowanych par transakcja-cel (każda transakcja odroczona może mieć wiele celów, do których powinna zostać spropagowana), liczbie błędów podczas propagowania transakcji opóźnionej, liczbie poprawnych transakcji znajdujących się jeszcze w kolejce (transakcje te powinny zostać usunięte z kolejki) należy wykonać poniższe zapytanie:

```
SELECT G.GLOBAL_NAME, D.ADMIN_REQUESTS, E.STATUS,
DT.TRAN, DE.ERRORS, C.COMPLETE
   FROM (SELECT GLOBAL_NAME FROM GLOBAL_NAME) G,
   (SELECT COUNT(ID) ADMIN_REQUESTS FROM
DBA_REPCATLOG) D,
   (SELECT COUNT(STATUS) STATUS FROM
DBA_REPCATLOG WHERE STATUS = 'ERROR') E,
   (SELECT COUNT(*) TRAN FROM DEFTRANDEST) DT,
   (SELECT COUNT(*) ERRORS FROM DEFERROR) DE,
   (SELECT COUNT(A.DEFERRED_TRAN_ID) COMPLETE
FROM DEFTRAN A
  WHERE A.DEFERRED_TRAN_ID NOT IN (
   SELECT B.DEFERRED_TRAN_ID FROM DEFTRANDEST B))
C;
```

Zapytanie to może wykonać użytkownik posiadający uprawnienia sysdba.

Poniższe zapytanie przedstawia strony główne dla każdej grupy głównej wraz z informacją na temat strony głównej definicji (Master definition site).

```
SELECT GNAME, DBLINK, MASTERDEF
    FROM DBA_REPSITES
    WHERE MASTER = 'Y'
    AND GNAME NOT IN (SELECT GNAME FROM
DBA_REPSITES WHERE SNAPMASTER = 'Y')
    ORDER BY GNAME;
```

Podzapytanie ma na celu wyeliminowanie grup widoków zmaterializowanych z wyników (po usunięciu warunku z podzapytaniem otrzymamy wszystkie strony główne).

Korzystając na podobnych zasadach z analogicznych perspektyw systemowych możemy uzyskać wiele innych informacji o replikacjach np.:

- dotyczące grup głównych;
- dotyczące widoków zmaterializowanych;
- dotyczące tabel logów widoków zmaterializowanych;
- i wielu innych.

8 Podsumowanie

W niniejszej pracy przedstawione zostały metody replikacji w bazach danych Oracle. Zostały przeprowadzone doświadczenia ilustrujące użyteczność replikacji prostej („snapshot” lub „materialized view”) jak i typu multi-master oraz przedstawiające obraz okoliczności, w których poszczególne sposoby odświeżania sprawdzają się najlepiej. Z przeprowadzonych doświadczeń można też wywnioskować, że sam proces odświeżania w niewielkim stopniu zależy od łącza (niewielkie różnice w czasach odświeżania między siecią bezprzewodową 54Mbit/s, a siecią przewodową 100Mbit/s). Na podstawie obserwacji utylizacji zasobów maszyny, na którą następowało replikowanie, można także zauważyć, że istotnym jest, aby maszyna docelowa dysponowała możliwie największym czasem procesora. Tutaj nasuwa się sugestia, aby replikację przeprowadzać wówczas, gdy serwer jest najmniej obciążony, czyli np. w nocy. Z przeprowadzonych doświadczeń wynika również, że w przypadku dużych, mało zmieniających się tabel najlepszym rozwiązaniem jest replikacja przyrostowa, dzięki której odświeżenie trwa zdecydowanie krócej. W przypadku małych tabel z często zmieniającymi się danymi uzasadnione jest użycie odświeżenia

pełnego. Reasumując, jeśli w systemach rozproszonych baz danych konieczne jest skrócenie czasu dostępu do części danych można rozwiązać ten problem poprzez replikację. Zaletą będzie uniezależnienie się od przepustowości i aktualnego obciążenia sieci (serwer docelowy „snapshot" znajduje się w lokalnej podsieci) oraz od czasowej niedostępności węzłów i awarii sieci (w przypadku klastrów baz danych). Wadą replikacji jest konieczność dokonywania odświeżeń replik. Wówczas, gdy rozproszony system bazy danych jest odpowiednio zaprojektowany można wyeliminować tę wadę sprowadzając ją tylko do potwierdzenia poprawności wykonania odświeżenia. Replikacja staje się wówczas potężnym narzędziem pozwalającym na wyraźne zwiększenie wydajności rozproszonej bazy danych.

Literatura

[1] Andrew Hudson, Paul Hudson, „Fedora 7. Księga eksperta.", Helion 2008

[2] Rick Greenwald, Robert Stackowiak, Jonathan Stern; „Oracle Database 11g. To co najważniejsze"; PWN 2009

[3] Michael McLaughlin; „Oracle Database 11g. Programowanie w języku PL/SQL"; Helion 2009

[4] Robert Wrembel, Bartosz Bębel; „Oracle. Projektowanie rozproszonych baz danych"; Helion 2003

[5] Kevin Loney; „Oracle Database 11g Kompendium administratora"; Helion 2010

[6] Rick Greenwald, Robert Stackowiak, Jonathan Stern; "Oracle Database 11g To co najważniejsze"; PWN 2008

[7] Adam Pelikant; „Programowanie serwera Oracle 11g SQL i PL/SQL"; Helion 2009

[8] Dokumentacja techniczna baz Oracle - http://www.oracle.com/pls/db111/homepage

REPLIKACJE W ORACLE

Summary – The paper presents replication methods used In Oracle database. Some experiments were executed to illustrate simple replication ass well as multi-master replication usability. On the basis of executed experiments it is possible to state that refreshing process depends not much on the connection (small diferences exist between wireless LAN 54Mbit/s, and wire LAN 100Mbit/s).