Rozproszone bazy danych to zbiór lokalnych baz danych w różnych węzłach sieci które z punktu widzenia aplikacji stanowią jedną bazę danych.

-Łączniki baz danych - obiekty, które umożliwiają dostęp z lokalnej bazy danych do zdalnej
SCHEMAT:
create [public] database link {nazwa} connect to {użytkownik} identified by {hasło} using '{nazwa_bazy_danych}';

USUWANIE:
DROP [PUBLIC] DATABASE LINK C1;

INFORMACJE:
SELECT * FROM ALL_DB_LINKS;
SELECT * FROM USER_DB_LINKS;

-Perspektywy - obiekty które udostępniają podzbiory danych,znajdujące się w bazie danych; Stosuje się je do integracji danych pochodzących z różnych węzłów
SCHEMAT:
create [or replace] [force | noforce] view {nazwa_perpektywy}
as
{zapytanie SQL}
[with read only]
[with check option [constraint nazwa_ograniczenia]];

-Synonimy - obiekty wskazujące na inne obiekty najczęściej na tabelę lub perspektywę,ta sama perspektywa widziana jest pod różnymi nazwami,są stosowane do ukrywania lokalizacji danych rozproszonych
SCHEMAT:
create synonym {nazwa} for {nazwa_tabeli@nazwa_łącznika};
create synonym {nazwa} for {nazwa_użytkownika.nazwa_tabeli};

INFORMACJE:
select synonym_name, table_owner, table_name, db_link from user_synonyms/all_synonyms;

-Trigger (procedura wyzwalana) - to program w języku PL/SQL, który reaguje na zdarzenia zachodzące w bazie danych i wykonuje się po zajściu określonych warunków
SCHEMAT:
CREATE [OR REPLACE] TRIGGER nazwa
{BEFORE | AFTER | INSTEAD OF} {INSERT | UPDATE | DELETE}
ON { tabela | perspektywa }

[FOR EACH ROW [WHEN warunek]]
[DECLARE /* deklaracje zmiennych i kursorów */]
BEGIN
/* ciało procedury wyzwalanej */
END;

Procedura:
create procedure {nazwa} as
begin
{polecenie sql np. UPDATE SPRZEDAŻ_S SET CENA_JEDN= CENA_JEDN*10;}
end;

PRZYKŁADY TRIGGERÓW:
Trigger wyzwalający procedurę po update:
create trigger aktualizuj
after update on sprzedaż_s
begin
{nazwa_procedury}
end;

-Migawki (snapshots) - obiekty stanowiące w najprostszym przypadku kopię tabeli znajdującej się w zdalnej bazie danych,Istnieje możliwość uaktualniania zawartości migawki.Moment ,częstotliwość i sposób odświeżania migawki są określone w jej definicji.
SCHEMAT:
create snapshot {nazwa}
refresh {sposób_odświeżania}
start with {data_1_odświeżania}
next {częstotliwość}
with {typ}
as {zapytanie};

FUNKCJE:

- **refresh {fast|complete|force} {on demand|on commit}**
  start with {sysdate - w tym momencie/ sysdate + 1 - jutro/ sysdate + (1/(24*6)) - za 10 minut}
  next {sysdate/sysdate + (1/(24*60)) - co minutę)}

BRAK ODŚWIEŻANIA: create materialized view {nazwa} never refresh as {zapytanie};

- **build {immediate|deferred}**
- **with {primary key|rowid}**

Snapshot logs - tabele, które przechowują informacje o zmianach wprowadzonych na tabeli master, konieczne do refresh fast (odświeżania przyrostowego)
SCHEMAT:

create snapshot log
on {tabela_bazowa}
with {primary key|ROWID|primary key, ROWID}
{including new values|excluding new values};

create materialized view log on {tabela_bazowa} with {primary key|ROWID|primary key, ROWID} {including new values|excluding new values};

USUWANIE:
drop snapshot {nazwa}; - usunięcie migawki
drop snapshot log on {nazwa tabeli bazowej}; - usunięcie dziennika migawki

INFORMACJE:
select {kolumny*} from {user_snapshots|all_snapshots|dba_snapshots}; - dla migawek
select {kolumny*} from {user_snapshot_logs|all_snapshot_logs|dba_snapshot_logs}; - dla dzienników migawek
select {kolumny*} to_char (last_refresh, 'dd:mm:yyyy:hh24:mm:ss') last_refresh from
{user_snapshot_refresh_times|all_snapshot_refresh_times|dba_snapshot_refresh_times}; - dla odświeżania migawek

***kolumny:**
name, table_name, master_owner, master, master_link, refresh_method, type (migawki)
log_owner, master, log_table, rowids, primary_key, filter_columns, current_snapshots, snapshot_id (dzienniki)
owner, name, master, master_owner (odświeżanie)

Łączniki baz danych

Create database link {nazwa} connect to {użytkownik} identified by {hasło} using '{nazwa_bazy_danych}';

np. : create database link c1
connect to baumgart identified by baumgart
using 'baza2';

Domyślnie tworzony jest łącznik prywatny.
Tworzenie łącznika publicznego wymaga uprawnienia
CREATE PUBLIC DATABASE LINK;
np:

np. : create public database link c1
connect to baumgart identified by baumgart
using 'baza2';

Sprawdzanie poprawności łącznika
Select * from tabela@c1;

UWAGA W BAZIE ZDALNEJ MUSI ISTNIEĆ IDENTYCZNY UŻYTKOWNIK Z IDENTYCZNYM HASŁEM.

Łącznik pozostaje otwarty do końca sesji lub jego jawnego zamkniecia poleceniem:

ALTER SESSION CLOSE DATABASE LINK C1;

Usuwanie łącznika:

DROP [PUBLIC] DATABASE LINK C1;

Informacje o utworzonych łącznikach :

SELECT * FROM ALL_DB_LINKS;

SELECT * FROM USER_DB_LINKS;

Lab 2

CELE PERSPEKTYW:

-Autoryzacja dostępu do danych – poprzez ograniczenie zbioru dostępnych danych dla użytkowników;

-Ułatwienie dostępu do danych – poprzez ukrycie złożonych zapytań w definicji perspektywy,

-Możliwość prezentowania tych samych danych w różny sposób – poprzez umieszczenie w definicji perspektywy np. funkcji konwersji danych, wyrażeń arytmetycznych i funkcji operacji na łańcuchach,

Logiczna niezależność danych – dzięki utworzeniu interfejsu z perspektyw pomiędzy aplikacjami a tabelami bazy danych, w przypadku zmiany schematu tabel bazowych, należy zmodyfikować wyłącznie definicję odpowiednich perspektyw, aby ich schemat pozostał taki, jak poprzednio, a aplikacje nie będą wymagały żadnych modyfikacji,

–Możliwość wprowadzenia dodatkowego poziomu ograniczeń integralnościowych – poprzez wprowadzenie klauzuli with check option w definicji perspektywy.

–Uniezależnienia aplikacji od fizycznej lokalizacji danych – poprzez ukrycie w definicji perspektywy lokalizacji danych

TWORZENIE PERSPEKTYWY

Create [or replace] [force | noforce] view nazwa_perpektywy
as
zapytanie SQL
[with read only]

[with check option [constraint nazwa_ograniczenia]];

Klauzula **or replace** umożliwia nad pisanie definicji istniejącej przez nową definicję.
Klauzula **force** umożliwia utworzenie perspektywy, nawet jeśli nie istnieją jej tabele bazowe.
Klauzula **with read only** zabrania kierowania do perspektywy poleceń DML.
Klauzula **with check option** umożliwia zdefiniowanie ograniczenia integralnościowymi.

PRZYKŁAD:

Create or replace view dobrze_zarabiajacy
as
Select nazwisko, placa
From pracownicy
Where placa >= 3000
with check option constraint zarobki;

Pozwala na edycje tylko pracowników którzy mają płacę większą niż 3000zł;

przeniesienie tabeli instytuty_baza2 do bazy danych case:
–utworzenie nowego łącznika wskazującego na instytuty_baza2
–zmiana definicji perspektywy z uwzgl. nowego łącznika
–aplikacje odwołujące się do perspektywy pozostają niezmienione

create or replace view inst_prac_baza2
as select nazwa, nazwisko
from instytuty_baza2@c2 i, pracownicy@c1 p
where i.idi=p.idi;

Synonim ukrywa łącznik:
create synonym s_inst_b2 for instytuty_baza2@c1;

select * from s_inst_b2;

INFORMACJE O SYNONIMACH:

select * from ALL_SYNONYMS;
select * from USER_SYNONYMS;

LAB4

Procedura wyzwalana (ang. trigger) to program w języku PL/SQL który reaguje na zdarzenia zachodzące w bazie danych i wykonuje się po zajściu określonych warunków.

Typy procedur wyzwalanych:
•BEFORE-uruchamiane przed wykonaniem polecenia INSERT, UPDATE, DELETE
•AFTER -uruchamiane po wykonaniu polecenia INSERT, UPDATE, DELETE
•INSTEAD OF –uruchamiane zamiast polecenia INSERT, UPDATE, DELETE
•systemowe –uruchamiane po zajściu określonego zdarzenia w schemacie lub bazie danych

INSTEAD OF: wyzwalacz może być zdefiniowany tylko na perspektywie •FOR EACH ROW: wyzwalacz wykonuje się dla każdej modyfikowanej krotki •WHEN: wyzwalacz wykonuje się tylko dla tych krotek, dla których jest spełniony warunek

PRZYKŁAD:
Poniższa procedura wyzwalana uruchamia się przed wstawieniem nowego pracownika i nadaje mu kolejny identyfikator pobierany z licznika (sekwencji)

```
CREATE OR REPLACE TRIGGER trig_id_prac
BEFORE INSERT ON pracownicy
FOR EACH ROW
BEGIN
        SELECT seq_pracownik.NEXTVAL
        IF (:NEW.id_prac IS NULL) THEN
        INTO :NEW.id_prac FROM DUAL;
        END IF;
END;
```

Wszystkie procedury wyzwalane związane z daną relacją można zablokować (odblokować) pojedynczym poleceniem:

```
ALTER TABLE nazwa_relacji
DISABLE [ENABLE] ALL TRIGGERS;
```

BLOKOWANIE WYZWALACZA:

```
ALTER TRIGGER nazwa DISABLE [ENABLE];
```

USUWANIE WYZWALACZA:

```
DROP TRIGGER nazwa;
```

WYSWIETLANIE WYZWALACZY:

SELECT TRIGGER_NAME, TRIGGER_TYPE,TRIGGERING_EVENT, TABLE_NAME, TRIGGER_BODY FROM USER_TRIGGERS;

PRZYKŁAD:

Trigger modyfikujący dane w tabeli przez perspektywę:

```
Begin
        if:new.miasto='Bydgoszcz'
                then update pracownicy_baza2
                        set stanowisko='profesor'
                        where tytul='profesor';
        Elsif:new.miasto='Poznań'
                then update pracownicy_case@c1
                        set stanowisko='profesor'
                        where tytul='profesor';
        endif;
End;
```

LAB5

Replikacja synchroniczna z wyzwalaczami trigger zmienia dane w replice:

UZUPEŁNIANIE SKLEPY_S WRAZ ZE ZMIANA SKLEPY:

```
create or replace trigger uzup_sklepy
        after insert on sklepy for each row
                begin
                insert into sklepy_s@c1
                values(:new.sklep_id id, : , new.nazwa nazwa, : , new.miasto miasto); );
        end
end;
```

ZMIANA WARTOŚCI DANYCH w sprzedaz_S :

```
create procedure zmien_dane
        AS
        BEGIN
        UPDATE sprzedaz_s SET CENA_jedn = CENA_jedn +
        (( CENA_jedn * 10) / 100);
END;
```

W PRZYPADKU ZMIANY W SPRZEDAŻ ZMIENIA DANE W SPRZEDAZ_S:

```
Create or replace trigger zmien
        after update on sprzedaz
```

begin
zmien_dane@c1;
end
end;

LAB6

Kopia tabel znajdujących się w odległych bazach danych
•standardowo tylko do odczytu
•przywileje:
–CREATE SNAPSHOT, CREATE TABLE, CREATE VIEW
–CREATE ANY SNAPSHOT
•rodzaje migawek

- PRIMARY KEY
  - - tabela master musi posiadać włączone ograniczenie PRIMARY KEY
  - klauzula SELECT musi zawierać wszystkie atrybuty wchodzące w skład klucza podstawowego tabeli master
- ROWID

•migawka-> tabela(+ perspektywa) + indeksy

SPOSOBY ODŚWIEŻANIA MIGAWEK

- –REFRESH FAST -> odświeżanie przyrostowe
  - dla migawek prostych
  - musi istnieć SNAPSHOT LOG dla tabeli master
- –REFRESH COMPLETE -> odświeżanie pełne
- –REFRESH FORCE -> automatyczny wybór metody odświeżania; jeżeli możliwe to Oracle wybiera FAST

•OKRESY ODŚWIEŻANIA

- –START WITH -> data pierwszego odświeżenia
- –NEXT -> wyrażenie określające częstotliwość odświeżania

- on demand–tworzona jest migawka odświeżana manualnie
- on commit można stosować jedynie, gdy:
  - –zapytanie korzysta z tabel lokalnych–migawek opartych o jedną tabelę, bez wyliczania agregatów
  - –migawek, których zapytanie wyznacza agregaty w oparciu o pojedynczą tabelę
  - –migawek których zapytanie wykorzystuje łączenie tabel, ale bez wyliczania agregatów

BEZ ODŚWIEŻANIA :

create materialized view mv_test
never refresh
as select *from user1.sklepy@dbl1;

create snapshot nazwa_migawki
refresh sposób_odświeżania
start with data_pierwszego_odświeżenia
next częstotliwość_odświeżania
with typ_migawki
as zapytanie;

PRZYKŁAD:

create snapshot mv_sprzedaz_1
build deferred
refresh force start with sysdate+ (1/(24*6)) //odświeżanie po 10 min
next sysdate+(1/(24*60))
with primary key as select produkt_id, l_sztuk, cena_jedn, data, sklep_id
from user1.sprzedaz@c1
where sklep_id=1;

MODYFIKOWANIE MIGAWKI:

ALTER SNAPSHOT [schemat.]migawka
[REFRESH { FAST | COMPLETE | FORCE}]
[{on demand | on commit]}
[WITH PRIMARY KEY]
[START WITH 'data']
[NEXT 'data']
;

DZIENNIK TWORZYMY W BAZIE 2 A MIGAWKĘ W CASE:
TWORZENIE DZIENNIKA MIGAWKI:

create snapshot log
on tabela_bazowa[with{primary key|
ROWID|
primarykey, ROWID |
ROWID(lista_kolumn_filtrujących) |primary_key(lista_kolumn_filtrujących)}]
[{ includingnewvalues| excludignnewvalues}];

EDYCJA DZIENNIKA MIGAWKI:

alter snapshot log
on tabela_bazowa
add{primarykey|
ROWID|
ROWID(lista_kolumn_filtrujących) **|**
primary_key(lista_kolumn_filtrujących)}
[{including new values| excluding new values}];

USUWANIE MIGAWKI

DROP SNAPSHOT [schemat.]migawka;

USUWANIE DZIENNIKA MIGAWSKI:

DROP SNAPSHOT LOG ON [schemat.]tabela;

INFORMACJE O MIGAWKACH:

SELECT * FROM USER_SNAPSHOTS
SELECT * FROM ALL_SNAPSHOTS
SELECT * FROM DBA_SNAPSHOTS

INFORMACJE O DZIENNIKACH MIGAWEK:

SELECT * FROM USER_SNAPSHOT_LOGS,
SELECT * FROM ALL_SNAPSHOT_LOGS,
SELECT * FROM DBA_SNAPSHOT_LOGS