

## 16-1 Working With Sequences

- SQL poznaje proces automatskog generisanja jedinstvenih brojeva koji eliminišu pojavu duplikata; to se izvodi preko db objekte zvane SEQUENCE

### Sequence Object

- Do sada su pravljene dve vrste db objekata, TABLE i VIEW; treća vrsta objekata je SEQUENCE
- SEQUENCE je deljeni (shareable) objekat koji automatski generiše jedinstvene brojeve; njemu može prići istovremeno više korisnika
- Najčešće se koriste za kreiranje PK vrednosti
- Pk moraju biti jedinstveni za svaki red. Sekvenca se generiše i inkrementira (ili dekrementira) od strane interne Oracle rutine
- sekvenca brojeva je smeštena i generisana nezavisno od tabela pa se zato ista sekvenca može koristiti za više tabela
- Da bi se kreirala SEQUENCE:

```
CREATE SEQUENCE sequence
  [INCREMENT BY n]
  [START WITH n]
  [{MAXVALUE n | NOMAXVALUE}]
  [{MINVALUE n | NOMINVALUE}]
  [{CYCLE | NOCYCLE}]
  [{CACHE n | NOCACHE}];
```

- Iako se sekvenca može koristiti za više tabela, to se nepreporučuje, pošto svaki broj u sekvenci se koristi samo jednom, svaka tabela će imati nedostajuće vrednosti (one koje koriste druge table)

### Sequence Syntax

sequence	is the name of the sequence generator (object)
INCREMENT BY n	specifies the interval between sequence numbers where n is an integer (If this clause is omitted, the sequence increments by 1.)
START WITH n	specifies the first sequence number to be generated (If this clause is omitted, the sequence starts with 1.)

- Ime sekvence mora biti uključeno u CREATE SEQUENCE iskazu, a svi ostali izrazi su opcioni, ali se savetuje da se uključe svi izrazi

MAXVALUE n	specifies the maximum value the sequence can generate
NOMAXVALUE	specifies a maximum value of 10 <sup>27</sup> for an ascending sequence and -1 for a descending sequence (default)
MINVALUE n	specifies the minimum sequence value

- MINVALUE se koristi za opadajuću sekvencu

NOMINVALUE	specifies a minimum value of 1 for an ascending sequence and $-(10^{26})$ for a descending sequence (default)
CYCLE   NOCYCLE	specifies whether the sequence continues to generate values after reaching its maximum or minimum value (NOCYCLE is the default option.)
CACHE n   NOCACHE	specifies how many values the Oracle server pre-allocates and keeps in memory. (By default, the Oracle server caches 20 values.) If the system crashes, the values are lost.

### Creating Sequence

- Primer kreiranja sekvence od broja 1:

```
CREATE SEQUENCE runner_id_seq
  INCREMENT BY 1
  START WITH 1
  MAXVALUE 50000
  NOCACHE
  NOCYCLE;
```

- U prethodnom primeru se očekuje da neće biti potrebno više od 50000 brojeva
- Opcija NOCACHE sprečava vrednosti u SEQUENCE da budu keširane u memoriji, što u slučaju pada sistema sprečava brojeve koji su prealocirani i smešteni u memoriji da budu izgubljeni
- Korišćenje CACHE opcije daje blago poboljšanje performansi pošto su brojevi prealocirani i smešteni u keš memoriji. Ovo je brži način nego sa NOCACHE, koji poziva sledeću vrednost iz sekvence kada se to zatraži
- Opcija NOCYCLE sprečava brojanje da započne ponovo od 1 ako se pređe vrednost od 50000
- Ne koristiti CYCLE opciju ako sekvenca se koristi za generisanje PK vrednosti sem ako postoji siguran mehanizam koji briše stare redove brže nego što se dodaju novi redovi

### Confirming Sequences

- Da bi verifikovali da je sekvenca kreirana, pravi se upit USER\_OBJECTS nad DD
- Da bi se videle sve SEQUENCE postavke, pravi se upit USER\_SEQUENCES nad DD kao na sledećem primeru

```
SELECT sequence_name, min_value, max_value, increment_by, last_number
FROM user_sequences;
```

- Ako je NOCACHE specificirano, last\_number kolona u upitu pokazuje sledeći dostupan sekvencijalan broj
- Ako je CACHE specificirano, last\_number kolona prikazuje sledeći dostupan broj u sekvenci koji nije keširan u memoriji

### NEXTVAL and CURRVAL Pseudocolumns

- Pseudokolona NEXTVAL se koristi za ekstrakciju sukcesivne sekvence brojeva iz specificirane sekvence
- NEXTVAL se mora kvalifikovati sa imenom sekvence

- Kada se referencira sequence.NEXTVAL, novi broj sekvence se generiše i trenutni broj sekvence se smešta u CURRVAL
- Sledeći primer smešta novi sektor u DEPARTMENTS tabelu; koristi DEPARTMENTS\_SEQ sekvencu za generisanje novog broja sektora:

```
INSERT INTO departments
      (department_id, department_name, location_id)
VALUES (departments_seq.NEXTVAL, 'Support', 2500);
```

- Ako treba zaposliti radnike da popune novi sektor; INSERT iskaz se izvršava za sve nove zaposlene može uključiti kod:

```
INSERT INTO employees
      (employee_id, department_id, ...)
VALUES (employees_seq.NEXTVAL, dept_deptid_seq .CURRVAL, ...);
```

- Prethodni primer je pretpostavio da sekvenca EMPLOYEES\_SEQ je već kreirana za generisanje brojeva novih zaposlenih
- Pseudokolona CURRVAL u sledećem primeru se koristi za referisanje na broj sekvence koju je trenutni korisnik upravo generisao
- NEXTVAL mora da se koristi za generisanje broja sekvence u sesiji trenutnog korisnika pre nego se CURRVAL referencira
- Mora se CURRVAL kvalifikovati sa imenom sekvence
- Kada se sequence.CURRVAL referencira, poslednja vrednost generisana od tog procesa korisnika se vraća

### Using a Sequence

- Posle kreiranja sekvence, on generiše sekvencijalne brojeve za korišćenje u tabelama. Treba referencirati vrednosti sekvence korišćenjem NEXTVAL i CURRVAL pseudokolona
- Može se koristiti NEXTVAL i CURRVAL u sledećem kontekstu:
  - SELECT lista od SELECT iskaza koja nije deo podupita
  - SELECT lista od podupita u jednom INSERT iskazu
  - VALUES izraz od INSERT iskaza
  - SET izraz od UPDATE iskaza
- Ne može se koristiti NEXTVAL i CURRVAL u sledećem kontekstu:
  - SELECT lista od pogleda
  - SELECT iskaz sa DISTINCT službenom reči
  - SELECT iskaz sa GROUP BY, HAVING ili ORDER BY izrazima
  - podupit u SELECT, DELETE ili UPDATE iskazom
  - DEFAULT izraz u CREATE TABLE ili ALTER TABLE iskaz
- Za potrebe maratona napisana je sledeća tabela:

```
CREATE TABLE runners
(runner_id NUMBER(6,0) CONSTRAINT runners_id_pk PRIMARY KEY,
 first_name VARCHAR2(30),
 last_name VARCHAR2(30));
```

- Zatim se kreira sekvenca koja će generisati vrednosti za runner\_id PK kolonu

```
CREATE SEQUENCE runner_id_seq
  INCREMENT BY 1
  START WITH 1
  MAXVALUE 50000
  NOCACHE
  NOCYCLE;
```

- Korišćenje sledeće sintakse će omogućiti novim učesnicima da se unesu u tabelu runners
- Broj identifikacije trkača će biti generisan vraćanjem NEXTVAL iz sekvence

```
INSERT INTO runners
  (runner_id, first_name, last_name)
VALUES (runner_id_seq.NEXTVAL, 'Joanne', 'Everely');
```

```
INSERT INTO runners
  (runner_id, first_name, last_name)
VALUES (runner_id_seq.NEXTVAL, 'Adam', 'Curtis');
```

- Potvrđivanje da sekvenca radi korektno, radi se upit nad tabelom:

```
SELECT runner_id, first_name, last_name
FROM runners;
```

RUNNER_ID	FIRST_NAME	LAST_NAME
1	Joanne	Everely
2	Adam	Curtis

- Za videti trenutne vrednosti za runners\_id\_seq, CURRVAL se koristi
- Oracle APEX neće izvršiti ovaj upit ali treba videti kako on radi

```
SELECT runner_id_seq.CURRVAL
FROM dual;
```

- Kada se jednom uzme NEXTVAL iz sekvence, više nema "session" i APEX više ne zna šta je u CURRVAL. Ako iskaz pokuša povratiti CURRVAL, izveštaj o grešci je generisan: ORA-08002: sequence SEQ\_D\_SONGS\_SQ.CURRVAL nije još definisan u ovoj sesiji
- Keš sekvence u memoriji daju brži pristup vrednostima sekvence
- Keš je popunjen prvi put kada se obratimo sekvenci
- Svaki zahtev za sledećim vrednosti u sekvenci se vraća iz keš sekvence
- Pošto se iskoristi poslednja vrednost sekvence, sledeći zahtev za sekvencom vuče sledeći keš od sekvanca u memoriju
- 20 je difolt broj sekvence brojeva koji su keširani

### Nonsequential Numbers

- Iako generator sekvenci daje sekvencijalne brojeve bez pauza, ova akcija se dešava nezavisno od db commit ili rollback
- Praznine (gaps – nesekvencijalni brojevi) se mogu generisati sa:
  - rolling back iskaz koji sadrži sekvencu, broj je izgubljen
  - pad sistema; ako sekvenca kešira vrednosti u memoriju a sistem padne, te vrednosti su izgubljene
  - ista sekvenca s ekoristi na više tabela; tada svaka tabela može imati gaps

## Viewing the Next Value

- Ako je sekvenca kreirana sa NOCACHE, moguće je videti sledeću dostupnu sekvencijalnu vrednost bez inkrementiranja sa upitom USER\_SEQUENCES tabele

```
SELECT sequence_name, min_value, max_value, last_number AS "Next number"
FROM USER_SEQUENCES
WHERE sequence_name = 'RUNNER_ID_SEQ';
```

SEQUENCE_NAME	MIN_VALUE	MAX_VALUE	Next number
RUNNER_ID_SEQ	1	50000	3

## Modifying a Sequence

- Kao i sa drugim db objektima, SEQUENCE se može menjati korišćenjem ALTER SEQUENCE iskaza
- Šta ako je na maraton prijavljeno više od 50000 trkača i mora se dodati još brojeva ?
- Sekvenca se može promeniti da bi se povećao broj MAXVALUE bez menjanja postojećeg redosleda brojeva

```
ALTER SEQUENCE runner_id_seq
    INCREMENT BY 1
    MAXVALUE 999999
    NOCACHE
    NOCYCLE;
```

- Neka validacija se izvrši kada se promeni sekvenca
- Npr, nova MAXVALUE koja je manja od trenutnog broja sekvence ne može se izvršiti

```
ALTER SEQUENCE runner_id_seq
    INCREMENT BY 1
    MAXVALUE 90
    NOCACHE
    NOCYCLE;
```

**ERROR at line 1:**  
**ORA-04009: MAXVALUE cannot be made to be less than the current value**

## ALTER SEQUENCE Guidelines

- Nekoliko uputstava pri izvršavanju ALTER SEQUENCE iskaza:
  - mora se biti vlasnik ili imati ALTER privilegije za sekvence da bi se moglo modifikovati
  - samo budući brojevi sekvenci su pod efektom ALTER SEQUENCE iskaza
  - opcija START WITH se ne može promeniti korišćenjem ALTER SEQUENCE. Sekvenca s emora odstraniti i ponovo napraviti da bi se restartovala sekvenca sa drugačijim brojem

## Removing a Sequence

- Da bi se odstranila sekvenca iz DD, koristiti DROP SEQUENCE iskaz
  - Mora se biti vlasnik sekvence ili imati DROP ANY SEQUENCE privilegije za odstranjivanje
  - Jednom odstranjeno, sekvenca se ne može ponovo referencirati
- ```
DROP SEQUENCE runner_id_seq;
```

## 16-2 Indexes and Synonyms

- Za svaki upit izvrši se potpuno skeniranje cele tabele što može i da potraje
- Oracle koristi indekse za ubrzavanje vraćanja redova

### Indexes

- Oracle Server indeks je šema objekat koji može ubrzati vraćanje redova korišćenjem pointera. Indeksi se mogu napraviti eksplicitno ili automatski
- Ako ne postoji indeks na koloni koja je izabrana, onda se dešava potpuno skeniranje tabele
- Indeks omogućava direktan i brz pristup redovima u tabeli
- Njegova uloga je da smanji neophodnost I/O diska korišćenjem indeksiranog puta za brzo lociranje podatka
- Indeks se koristi i održava automatski od Oracle Servera. Kada je jednom indeks kreiran, nikakva direktna aktivnost nije tražena od korisnika
- ROWID je osnovni 64 string reprezentacija adrese reda koja sadrži blok identifikatora, lokaciju reda u bloku i db fajl identifikator
- Indeksi koriste ROWID-e pošto su oni brz način za pristup bilo kojem redu
- Indeksi su logički i fizički nezavisni od tabele koju indeksiraju
- To znači da mogu biti kreirani ili odstranjeni u bilo kom momentu i da nemaju nikakav efekat na osnovnu tabelu ili druge indekse

| COUNTRY_ID | COUNTRY_NAME                | CAPITOL        | REGION_ID |
|------------|-----------------------------|----------------|-----------|
| 1          | United States of America    | Washington, DC | 21        |
| 2          | Canada                      | Ottawa         | 21        |
| 3          | Republic of Kazakhstan      | Astana         | 143       |
| 7          | Russian Federation          | Moscow         | 151       |
| 12         | Coral Sea Islands Territory | -              | 9         |
| 13         | Cook Islands                | Avarua         | 9         |
| 15         | Europa Island               | -              | 18        |
| 20         | Arab Republic of Egypt      | Cairo          | 15        |
| ...        | ...                         | ...            | ...       |

- Kada odstraniš (drop) tabelu, odgovarajući indeksi su takođe odstranjeni

### Types of Indexes

- Dva tipa indeksa se mogu napraviti:
  - Unique index: Oracle Server automatski pravi ovaj indeks kada se definiše kolona u tabeli da ima PK ili UNIQUE KEY ograničenje (ime indeksa je ime dato ograničenju; iako se može manuelno kreirati jedinstveni indeks, preporučuje se da se kreira jedinstvenio ograničenje u tabeli koje implicitno kreira jedinstveni indeks)
  - Nonunique index: Ovo je indeks koji korisnik može napraviti za ubrzanje pristupa redovima (npr, za optimizovanje joins, može se napraviti indeks na FK koloni, koji ubrzava potragu za identičnim redovima sa PK kolonom)

### Creating an Index

- Kreirati indeks na jednoj ili više kolona izdavanjem CREATE INDEX iskaza:

```
CREATE INDEX index_name
ON table_name( column...,column)
```

- Za kreiranje indeksa u šemi, mora se imati CREATE TABLE privilegija

- Za kreiranje indeksa u bilo kojoj šemi, treba CREATE ANY INDEX privilegije ili CREATE TABLE privilegije na tabeli na koj se želi kreirati indeks. Null vrednosti nisu uključene u indeks
- Npr, za poboljšanje brzine pristupa upitu u REGION\_ID koloni u WF\_COUNTRIES tabeli:

```
CREATE INDEX wf_cont_reg_id_idx
ON wf_countries(region_id);
```

### When to Create an Index

- Indeks treba napraviti samo ako:
  - kolona sadrži širok opseg vrednosti
  - kolona sadrži veliki broj null vrednosti
  - jedna ili više kolona su često korišćene zajedno u WHERE izrazu ili join uslovu
  - tabela je velika i od dosta upita se očekuje da vrate manje od 2-4% redova

### When Not to Create an Index

- Kada se odlučuje da li ili ne kreirati indeks, više nije i uvek bolje
- Svaka DML operacija (INSERT, UPDATE, DELETE) koje se izvode na tabeli s aindeksima znači da indeksi moraju da se updejtuju
- Što je više indeksa povezano sa tabelom, više napora treba da bi se updejtivali svi indeksi posle DML operacija
- Obično ne treba praviti indekse ako:
  - tabela je mala
  - kolone se ne koriste često kao uslov u upitu
  - najviše upita se očekuje da vrate više od 2-4% redova u tabeli
  - tabela se često updejtuje
  - indeksirane kolone su referencirane kao deo izraza

### Composite Index

- Kompozitni indeks (concatenated index) je indeks koji se kreira na više kolona u tabeli
- Kolone u kompozitnom indeksu mogu da se pojave u bilo kojem redosledu i ne trebaju biti grupisane u tabeli
- Kompozitni indeksi mogu ubrzati povratak podataka za SELECT iskaze u kojima WHERE izraz referencira sve ili veći deo kolona u kompozitnom indeksu

```
CREATE INDEX emps_name_idx
ON employees(first_name, last_name);
```

- Null vrednost nisu uključene u kompozitne indekse
- Za optimizovanje joins, može se kreirati indeks na FK koloni, koji ubrzava pretragu za odgovarajućim redovima sa PK kolonom
- Optimizator ne koristi indeks ako WHERE iskaz sadrži IS NULL izraz

### Conforming Indexes

- Potvrdi postojanje indeksa iz USER\_INDEXES DD pogleda
- Takođe se može proveriti uključene kolone u indeks davanjem upita nad USER\_IND\_COLUMNS pogledom

- Upit prikazan na sledećem primeru je join između USER\_INDEXES tabele (imena indeksa i njihove jedinstvenosti) i USER\_IND\_COLUMNS (imena indeksa, imena tabela i imena kolona) tabele

```
SELECT DISTINCT ic.index_name, ic.column_name,
                ic.column_position, id.uniqueness
FROM user_indexes id, user_ind_columns ic
WHERE id.table_name = ic.table_name
AND ic.table_name = 'EMPLOYEES';
```

| INDEX NAME        | COLUMN NAME   | COL POS | UNIQUENESS |
|-------------------|---------------|---------|------------|
| EMP_EMAIL_UK      | EMAIL         | 1       | UNIQUE     |
| EMP_EMP_ID_PK     | EMPLOYEE_ID   | 1       | UNIQUE     |
| EMP_DEPARTMENT_IX | DEPARTMENT_ID | 1       | NONUNIQUE  |
| EMP_JOB_IX        | JOB_ID        | 1       | NONUNIQUE  |
| EMP_MANAGER_IX    | MANAGER_ID    | 1       | NONUNIQUE  |
| EMP_NAME_IX       | LAST_NAME     | 1       | NONUNIQUE  |
| EMP_NAME_IX       | FIRST_NAME    | 2       | NONUNIQUE  |

### Function-based indexes

- Indeksi bazirani na funkcijama smeštaju indeksirane vrednosti i koriste indekse bazirane na SELECT iskazima za vraćanje podataka
- Indeksi bazirani na funkcijama su indeksi bazirani na izrazima
- Indeks izraz se pravi od tabele kolona, konstanti, SQL funkcija i korisnički definisanih funkcija
- Indeksi bazirani na funkcijama su korisni kada se ne zna u slučaju je podatak smešten u db
- Npr, može se kreirati indeks bazirana funkcija koja se može koristiti sa SELECT iskazom korišćenjem UPPER u WHERE izrazu
- Indeks će se kroistiti u ovoj pretrazi:

```
CREATE INDEX upper_last_name_idx
ON employees (UPPER(last_name));
```

```
SELECT *
FROM employees
WHERE UPPER(last_name) = 'KING';
```

- FBI definisani sa UPPER(column\_name) ili sa LOWER(column\_name) službenim rečima dopuštaju pretrage zavisne od veličine slova
- Ako se ne zna kako se unelo prezime zaposlenog u db, može se koristiti indeks unosom uppercase u SELECT iskazu
- Kada se upit modifikuje korišćenjem izraza u WHERE iskazu, indeks ga neće koristiti sem ako se kreira FBI za odgovarajući izraz
- Npr, sledeći iskaz dopušta pretrage sa različitom veličinom slova korišćenjem indeksa:

```
CREATE INDEX upper_last_name_idx
ON employees (UPPER(last_name));
```

```
SELECT *
FROM employees
WHERE UPPER(last_name) LIKE 'KIN%';
```



- Da bi se osiguralo da OracleServer koristi indeks pre nego izvodi puno skeniranje tabele, treba biti siguran da vrednost funkcije nije null u subsequent upitima
- Npr, sledeći iskaz garantovano koristi indeks, lai bez WHERE iskaza Oracle Server može izvesti potpuni sken tabele:

```
SELECT *
FROM employees
WHERE UPPER (last_name) IS NOT NULL
ORDER BY UPPER (last_name);
```

- Oracle Server tretira indekse sa kolonama markiranim DESC kao FBI
- Kolone markirane DESC su sortirane u opadajućem redosledu
- Svi ovi primeri koriste UPPER i LOWER funkcije, lai treba приметiti da dok ove dve su česte u FBI, Oracle db nije limitirana na njih
- Bilo koja validna Oracle ugrađena funkcija (npr, TO\_CHAR) se može koristiti
- Primer: tabela employees je pod upitom u potrazi za bilo kojim zaposlenim u 1987

```
SELECT first_name, last_name, hire_date
FROM employees
WHERE TO_CHAR(hire_date, 'yyyy') = '1987'
```

- Ovaj upit daje rezultat u Full Table scan, koji može biti skupa operacija ako je tabela velika
- Čak ako hire\_date kolona je indeksirana, indeks nije korišćen zbog TO\_CHAR izraza
- Kada se jednom koristi sledeći FBI, može se startovati isti upit, ali sada izbegavajući Full Table Scan

```
CREATE INDEX emp_hire_year_idx
ON employees (TO_CHAR(hire_date, 'yyyy'));
```

```
SELECT first_name, last_name, hire_date
FROM employees
WHERE TO_CHAR(hire_date, 'yyyy') = '1987'
```

| FIRST_NAME | LAST_NAME | HIRE_DATE   |
|------------|-----------|-------------|
| Steven     | King      | 17-Jun-1987 |
| Jennifer   | Whalen    | 17-Sep-1987 |

- Sada, Oracle može koristiti indeks na hire\_date koloni

### Removing an Index

- Ne može se modifikovati indeks
- Da bi se promenio indeks, mora se ispustiti (drop) a zatim ponovo napraviti
- Otklanjanje indeks definicije iz DD izdavanjem DROP INDEX iskaza
- Da bi ispustio indeks, mora se biti vlasnik indeksa ili imati DROP ANY INDEX privilegiju
- Ako se ispusti tabela, indeksi i ograničenja su automatski ispušteni, ali pogledi i sekvence ostaju

```
DROP INDEX upper_last_name_idx;
```

```
DROP INDEX emps_name_idx;
```

```
DROP INDEX emp_hire_year_idx;
```

## SYNONYM

- U SQL, kao u jeziku, sinonim je reč ili izraz koji je dozvoljena zamena za drugu reč
- Sinonimi se koriste za pojednostavljenje pristupa objektima kreiranjem drugog imena za objekat
- Sinonimi mogu napraviti referisane na tabelu koja je vlasništvo drugog korisnika lakšim i skratiti dugačka imena objekata
- Npr, referisanje na amy\_copy\_employees tabelu u tvojoj classmate šemi, možeš prefiks tabelu imena sa imenom korisnika koji je napravio praćenu prazninom a onda imenom tabele kao USMA\_SBHS\_SQL01\_S04.amy\_copy\_employees
- Kreiranje sinonima eliminiše potrebu za kvalifikacijom imena objekta sa šemom i daje alternativu ienu za tabelu, pogled, sekvencu, proceduru ili drugi objekat
- Ovaj metod može biti posebno efikasan sa dugim imenima objekata, kao što su pogledi
- DBA može napraviti javni sinonim pristupačan svim korisnicima i može posebno dozvoliti CREATE PUBLIC SYNONYM privilegiju za bilo kojeg korisnika, a taj korisnik može napraviti javni sinonim
- Sintaksa:

```
CREATE [PUBLIC] SYNONYM synonym
FOR object;
```

- PUBLIC: kreira sinonim dostupan svim korisnicima
- synonym: ime sinonima koji će biti kreiran
- object: identifikuje objekat za koji je kreiran sinonim

```
CREATE SYNONYM amy_emps
FOR amy_copy_employees;
```

- Nedovoljne privilegije greška će se desiti kada se koristi CREATE PUBLIC SYNONYM u APEX. CREATE SYNONYM je dopušten
- Za kreiranje SYNONYM za tabele koje nisu u tvojoj šemi, treba prefiks tablename sa imenom šeme i tačkom, npr:  
CREATE SYNONYM amy\_emps  
FOR USMA\_SBHS\_SQL01\_S04.amy\_copy\_employees;

## SYNONYM Guidelines

- Uputstva: objekat ne može biti u paketu; privatno ime sinonima mora biti odvojeno od ostalih objekata čiji vlasnik je isti korisnik
- Za otklanjanje sinonima:

```
DROP [PUBLIC] SYNONYM name_of_synonym
```

```
DROP SYNONYM amy_emps;
```

## Confirming a SYNONYM

- Postojanje sinonima se može potvrditi upitom USER\_SYNONYMS DD pogleda

| Column Name  | Contents                                       |
|--------------|------------------------------------------------|
| Synonym_name | Name of the synonym.                           |
| Table_name   | Owner of the object referenced by the synonym. |
| Table_owner  | Name of the object referenced by the synonym.  |
| Db_link      | Database link referenced in a remote synonym.  |