

# Baze de date

## Limbajul SQL

THE **INFORMATION** COMPANY

# ***Curs 10***

## ***Limbajul SQL***

# ***Limbajul SQL***

## **Alte obiecte din baza de date:**

- 1. Secvente**
- 2. Indecși**
- 3. Sinonime**

- O bază de date conține și alte obiecte decât cele cu care v-ați familiarizat în celelalte cursuri.
- Obiectele care pot exista într-o bază de date sunt:
  1. tabele
  2. vederi (views)
  3. secvențe
  4. indecși
  5. sinonime

# Cuprins

**1. Secvente**

**2. Indecsi**

**3. Sinonime**

# 1. Secvențele

De ce folosim secvențele?

- Aplicațiile au în general nevoie în cadrul tabelelor de a conține unele **numere de identificare unice per tabelă** cum este și cazul cheilor primare.
- Acestea pot fi calculate de către aplicație direct folosind anumiți algoritmi în funcție de înregistrările deja existente și apoi pot fi introduse în baza de date odată cu celelalte date.

# 1. Secvențele

- În acest caz însă apar mari probleme de sincronizare astfel încât dacă mai multe instanțe a aceleași aplicații accesează în același timp aproximativ datele deja existente în baza de date rezultatul cheilor va fi același.
- De asemenea sunt necesare de la una la mai multe cereri la baza de date pentru a determina care sunt valorile deja existente astfel încât aplicația consuma mai mult timp procesor pentru fiecare inserare și devine mult mai lentă.

- O soluție la aceste probleme sunt **secvențele**.
- *Secvențele sunt niște obiecte care pot fi împărțite între mai mulți utilizatori și care pot să genereze numere întregi unice.*
- Cea mai răspândită utilizare a lor este pentru cheile unice ale tabelelor.
- Secvențele sunt incrementate sau decrementate cu ajutorul unei **rutine interne** a sistemului de gestiune a bazelor de date (**Oracle**).



- *Secvențele nu sunt o proprietate a tabelelor* astfel încât ele *pot fi folosite de către mai multe tabele în același timp.*
- În practica uzuală *pentru fiecare tabelă în parte se atribuie o secvență separată* dar ele mai pot fi utilizate astfel încât mai multe tabele folosesc aceeași secvență sau o tabelă poate avea mai multe secvențe.

# Crearea secvențelor

O secvență poate fi creată utilizând următoarea sintaxa **SQL**:

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

## Parametrii utilizati:

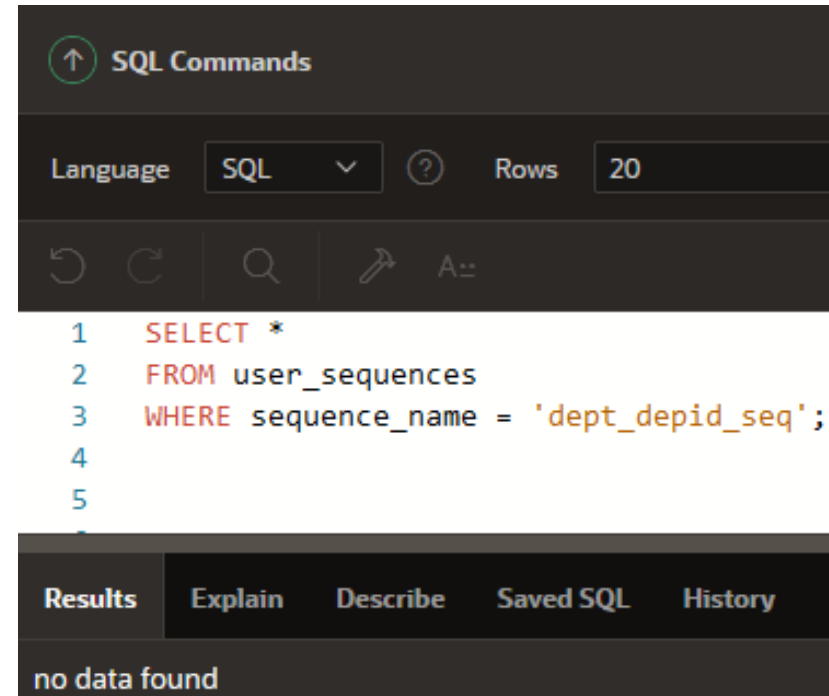
Parametru	Descriere
<b>nume_secventa</b>	este numele generatorului de secventa care trebuie sa fie unic in cadrul listei de secvente existente
<b>INCREMENT BY n</b>	aceasta optiune specifica un intreg n cu care ultima intrare in secventa va fi incrementata pentru obtinerea unei noi valori. (daca optiunea nu este specificata se foloseste 1)
<b>START WITH n</b>	precizeaza primul numar care va fi generat in cadrul secventei (daca acesta nu este specificat valoarea implicita este 1)
<b>MAXVALUE n</b>	specifica daca valoarea maxima pe care o secventa o poate genera
<b>NOMAXVALUE</b>	specifica o valoare maxima pe care o secventa o poate genera egala cu $10^{27}$ pentru o secventa incrementala sau -1 pentru o secventa decrementala. (aceasta este optiunea folosita daca nu este specificata o valoare maxima)
<b>MINVALUE n</b>	specifica valoarea minima pe care o secventa o poate genera.
<b>NOMINVALUE</b>	specifica o valoare minima de 1 pentru o secventa ascendenta sau de $-10^{26}$ pentru o secventa descendenta. (optiunea aceasta este folosita in cazul in care nu este specificat un minim)
<b>CYCLE   NOCYCLE</b>	specifica daca secventa va continua generarea de numere in cazul in care secventa si-a atins limitele (NOCYCLE este optiunea implicita)
<b>CACHE n   NOCACHE</b>	precizeaza cate valori sunt tinute in memorie de catre serverul Oracle (implicit valoarea este de 20)

# 1. Secventele

**Exemplul 1** - Crearea unei secvente in cadrul tabelii departamente pentru a fi folosita de cheia primara.

Sa ne asiguram ca aceasta secventa nu exista deja:

```
SELECT *  
FROM user_sequences  
WHERE sequence_name =  
      'dept_seq1';
```



```
SQL Commands  
Language SQL Rows 20  
1 SELECT *  
2 FROM user_sequences  
3 WHERE sequence_name = 'dept_seq1';  
4  
5  
Results Explain Describe Saved SQL History  
no data found
```

Acum sa o cream avand optiunile urmatoare:

- incepe la 120
- creste cu 10 unitati
- valoare maxima este 9999
- fara ciclu
- si fara cache

```
CREATE SEQUENCE dept_seq_1  
INCREMENT BY 1  
START WITH 70  
MAXVALUE 99  
NOCACHE  
NOCYCLE;
```

8	CREATE SEQUENCE dept_seq_1
9	INCREMENT BY 1
10	START WITH 70
11	MAXVALUE 99
12	NOCACHE
13	NOCYCLE;

## Pseudocoloanele **NEXTVAL** si **CURRVAL**

- Acestea se folosesc pentru a putea accesa valorile unei secvente.
- Denumirile lor sunt sugestive:
  1. **NEXTVAL** = valoarea urmatoare
  2. **CURRVAL** = valoarea curenta

**1. NEXTVAL** este o pseudocoloana care este utilizata pentru a putea extrage valoarea urmatoare dintr-o secventa.

- Aceasta se foloseste intr-un SELECT impreuna cu numele secventei din care se doreste extragerea valorii.
- Daca sintaxa cererii SQL contine formatul **nume\_secventa.NEXTVAL** atunci secventa este incrementata, valoarea curenta din NEXTVAL este plasata in CURRVAL si NEXTVAL va contine urmatorul numar din secventa.

**2. CURRVAL** este folosita pentru a putea extrage din cadrul secventei ultima valoare care a fost generata.

- Precum am precizat anterior trebuie utilizata pseudocoloana NEXTVAL pentru generarea valorii care va fi continuta in CURRVAL.
- CURRVAL este extras in acelasi mod in care se extrage si valoarea NEXTVAL printr-o cerere SQL SELECT impreuna cu numele secventei pentru care se doreste aflarea valorii.
- Cand cererea se face cu ajutorul sintaxei **nume\_secventa.CURRVAL** valoarea returnata va fi valoarea care a fost folosita ultima data in cadrul aceluiasi proces.



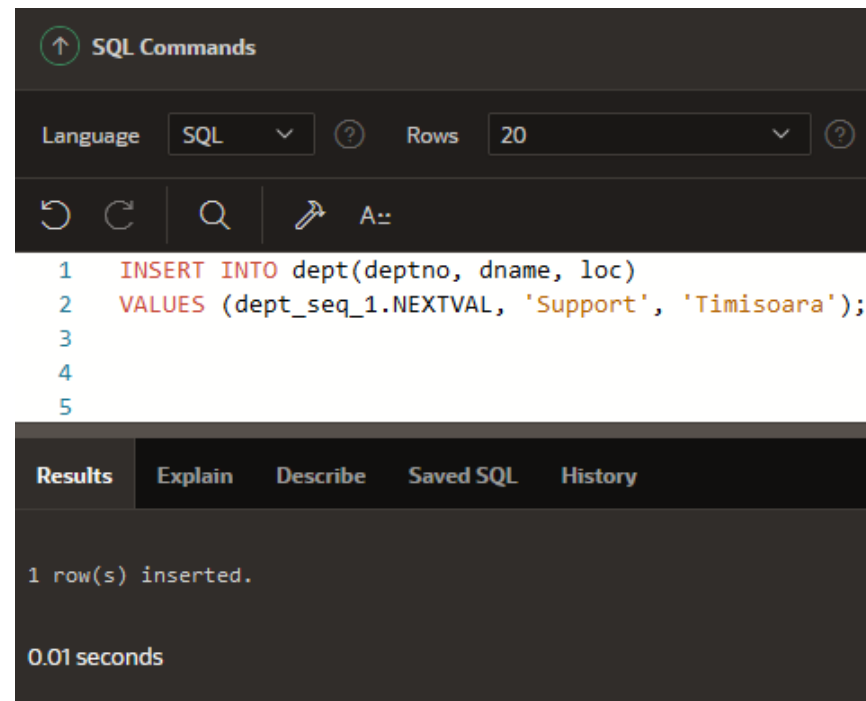
## Exista niste constrangeri la folosirea pseudocoloanelor **CURRVAL** si **NEXTVAL**:

- Cand cererea SELECT se refera la obiecte de tip vedere
- Cand in cererea SELECT se utilizeaza cuvantul cheie DISTINCT
- Cand cererea SELECT contine GROUP BY, HAVING sau ORDER BY
- Cand instructiunea SELECT este folosita intr-o sub-interogare.
- Cand este utilizata expresia DEFAULT impreuna cu CREATE TABLE sau cu ALTER TABLE

## Exemplu 2 - Utilizarea unei secvente:

- Pentru inserarea in baza de date in tabela de departamente vom folosi in cererea noastra SQL secventa pe care am creat-o mai inainte.
- Pentru inserare se va folosi cererea urmatoare:

```
INSERT INTO dept(deptno, dname, loc)  
VALUES (dept_seq_1.NEXTVAL, 'Support', 'Timisoara');
```



The screenshot shows a SQL command window with a dark theme. At the top, there's a tab labeled "SQL Commands". Below it, the "Language" is set to "SQL" and "Rows" is set to "20". The command being executed is:

```
1  INSERT INTO dept(deptno, dname, loc)
2  VALUES (dept_seq_1.NEXTVAL, 'Support', 'Timisoara');
3
4
5
```

Below the command window, there's a "Results" tab. The output shows:

```
1 row(s) inserted.
```

At the bottom, it indicates the execution time: "0.01 seconds".

# 1. Secventele

- Pentru a putea sa vedem valoarea inserata se poate folosi urmatoarea cerere:

```
SELECT dept_seq_1.NEXTVAL  
FROM dual;
```

The screenshot shows a SQL IDE interface. At the top, there's a 'Language' dropdown set to 'SQL' and a 'Rows' dropdown set to '20'. Below these are buttons for 'Find Tables' and 'Clear Command'. The main area displays the SQL query: `SELECT dept_seq_1.NEXTVAL FROM dual;`. Below the query, there's a 'Results' tab with sub-tabs for 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, showing a table with one column named 'NEXTVAL' and one row with the value '71'. At the bottom, it says '1 rows returned in 0.00 seconds' and has a 'Download' button.

NEXTVAL
71

1 rows returned in 0.00 seconds [Download](#)

- Folosirea secventelor cu incarcare in prealabil in memorie are avantajul ca viteza de raspuns creste din partea bazei de date.
- Din pacate apar cu ocazia aceasta si unele probleme:
  - *In momentul unei erori fizice a masinii* pe care ruleaza baza de date informatiile care sunt pastrate in memorie sunt pierdute si apar asa numitele *gauri in secventa*.
  - Astfel de *gauri in secventa* mai pot aparea in cazul in care **se sterg inregistrari din baza de date**, in cazul in care **o cerere a esuat si sistemul trebuie sa se intoarca la valorile precedente** (actiune de Rollback) sau in momentul in care **o secventa este utilizata pentru mai multe tabele**.

# Modificarea secventelor

O secventa poate fi modificata folosind urmatoarea interogare generica:

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

unde **nume\_secventa** este numele secventei care se doreste a fi modificata. Aceasta metoda este utila de exemplu pentru cazul in care secventa atinge valoarea maxima si in cazul dorintei de a o folosi in continuare ea intoarce o eroare.

Exista unele **restrictii** in cazul alterarii unei secvente deja existe.

Acestea sunt:

1. Userul care executa actiunea de alterare trebuie sa aiba drepturile necesare sau sa fie proprietarul secventei.
2. Numai numerele care vor fi generate in viitor vor fi afectate.
3. Secventa va trebui distrusa si recreata pentru ca valoarea de inceput sa poata fi modificata
4. Anumite validari logice de genul ca valoarea maxima nou setata trebuie sa nu fie mai mica decat valoarea care exista deja, trebuie indeplinite.

# Stergerea secventelor

- Stergerea unei secvente se poate efectua cu ajutorul urmatoarei cereri SQL:

**DROP SEQUENCE nume\_secventa**

- unde **nume\_secventa** este numele secventei care se doreste distrusa.

# Cuprins

**1. Secvente**

**2. Indeksi**

**3. Sinonime**



## 2. Indecșii

- *Indecșii sunt niste obiecte de tip schema care imbunatatesc timpul de cautare si acces la inregistrările din baza de date.*
- Indecșii sunt creati in mod explicit sau automat in anumite conditii.

## De ce folosim indecsii?

- *Indecsii ofera o metode de acces direct catre inregistrările care se doresc extrase.*
- Scopul lor este de a diminua operatiile de intrare/iesire catre disc prin folosirea unei metode de indexare a cailor catre acele date.
- Indecsii sunt utilizati si mentinuti de catre serverul **Oracle** in mod transparent pentru utilizator care nu implica nici o alta actiune ulterioara.

## 2. Indecșii

- Modul în care *indecsii* sunt concepuți îi fac independent fizic și logic de baza de date ceea ce se exprimă prin faptul că ei *pot să fie creați sau distruși în orice moment fără a afecta structura bazei de date sau asupra celorlalți indecși.*
- De asemenea din partea de management transparent face parte și faptul că *dacă o baza de date este distrusă atunci și indecsii care au fost creați pentru ea sunt de asemenea distruși.*

# Crearea indecșilor

Indecsii sunt creati in doua feluri:

- 1. Automat:** In momentul in care se creaza o cheie primara asupra tabelelor, in momentul in care se creaza o restrictie ca inregistrarile din baza de date sa fie unice si in alte situatii asemanatoare.
- 2. Manual:** Se pot defini indecsi manuali care nu trebuie sa contina o restrictie cum este cazul indecsilor unici.

- Indecsii sunt creati folosind urmatoarea cerere SQL la baza de date:

```
CREATE INDEX numele_index  
ON numele_tabelei (camp1[, camp2].... )
```

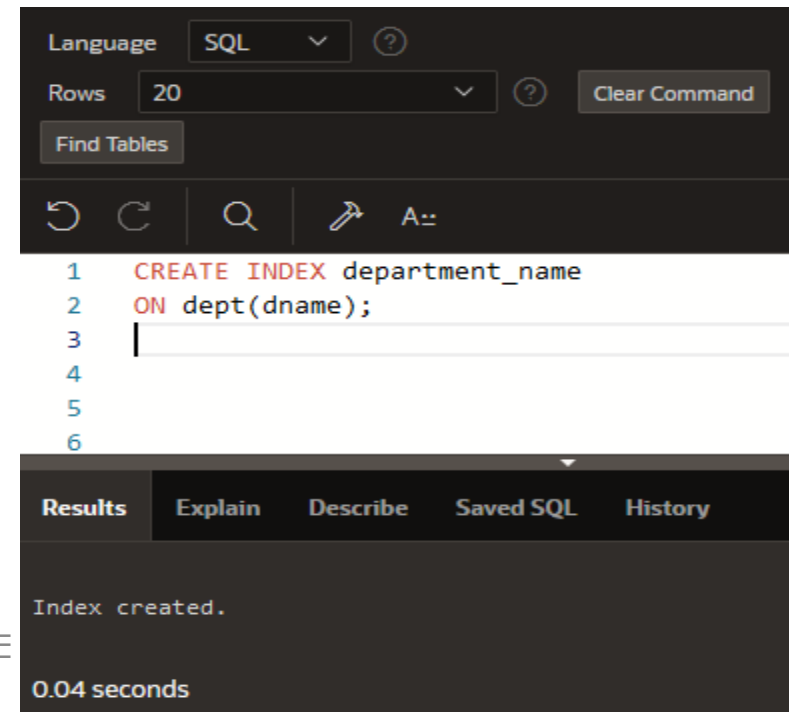
- unde **numele\_index** este numele pe care il are indexul, **numele\_tabelei** este numele tabelei din baza de date pentru a carei campuri indexul va fi creat si **camp1, camp2...** sunt numele campurilor din tabela pentru care se creaza acest index.
- *Un index poate fi creat pentru una sau mai multe coloane din aceeasi tabela.*

## 2. Indecșii

Exemplu 1 - crearea unui index:

O sa cream un index pentru departamente care ne va ajuta sa extragem foarte repede o inregistrare in functie de numele departamentului.

**CREATE INDEX department\_name  
ON dept(dname);**



The screenshot shows a SQL IDE interface. At the top, there's a 'Language' dropdown set to 'SQL' and a 'Rows' dropdown set to '20'. Below these are buttons for 'Find Tables' and 'Clear Command'. The main area displays the SQL command: `1 CREATE INDEX department_name`  
`2 ON dept(dname);`  
`3`  
`4`  
`5`  
`6`. Below the command area, there's a tabbed interface with 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, showing the message 'Index created.' and the execution time '0.04 seconds'.

```
Language SQL ?
Rows 20 ? Clear Command
Find Tables

1 CREATE INDEX department_name
2 ON dept(dname);
3
4
5
6

Results Explain Describe Saved SQL History

Index created.

0.04 seconds
```

## Cand se creaza indecșii?

Exista mai multe situatii care necesita crearea indecsilor in baza de date.

*Un index este creat daca:*

- o singura coloana contine o varietate foarte mare de valori si un numar mare de valori.
- campul din baza de date contine multe inregistrari de tip *null*
- campurile sunt folosite foarte frecvent in conditile de selectare a inregistrarilor sau in cazuri de JOIN

## Cand se creaza indecșii? (continuare)

- tabela pe care se creaza indexul este foarte mare si cererile in mod frecvent nu extrag o cantitate mare de date (2-4% din cantitatea totala de informatii din tabela)
- in momentul in care anumite chei sunt create in tabela ca si cheile unice si atunci **Oracle** genereaza automat un index unic.



## Cand nu se creaza indecsii si de ce?

- In general nu tot timpul este necesar si bine sa fie creati indexi in baza de date.
- O creare excesiva de indecsi in baza de date are si multe neajunsuri.
- In momentul in care sunt *adaugati indecsi*, *baza de date face o repunere la zi a lor pentru fiecare operatie care este executata.*
- In momentul in care sunt creati prea multi indecsi *operatile de extragere din baza de date isi amelioreaza viteza doar cu un raport mic* dar *pentru fiecare operatiune de inserare sau stergere timpul necesar creste simtitor precum si resursele folosite.*

Deci se pune problema cand nu sunt necesari indecsii:

- ✓ Cand tabela pe care se doreste adaugarea este mica.
- ✓ Cand coloanele pe care se adauga indecsii nu sunt folosite frecvent pentru executarea de extrageri de inregistrari
- ✓ Cand cererile de tip SELECT in general extrag un numar mare de inregistrari (mai mare de 2-4%)
- ✓ Cand operatiuni de UPDATE sunt foarte des utilizate.
- ✓ Cand coloanele indexate sunt referentiate ca parte integranta a unor expresii.

## 2. Indecșii

### Confirmarea indecsilor

- O lista completa de indecsi poate fi extrasa din vederea **USER\_INDEXES** care contine numele indexului si daca acesta are o restrictie de unicitate.

## 2. Indecșii

Urmatorul query poate fi folosit pentru tabela 'dept'.

```
SELECT ic.index_name, ic.column_name,  
       ic.column_position col_pos, ix.uniqueness  
FROM   user_indexes ix, user_ind_columns ic  
WHERE  ic.table_name = ix.index_name  
AND    ic.table_name = "dept";
```

# Indecșii bazați pe expresii

- Indecsi mai pot fi adaugati nu numai pe campuri simple din baza de date.
- Ei pot sa fie folositi de asemenea pe anumite expresii.
- Aceste expresii pot fi construite din campuri din tabela, constante, functii SQL sau functii definite de catre utilizator.
- In felul acesta se pot optimiza cererile de extragere care contin conditii complexe bazate pe calcule.

## 2. Indecșii

Exemplu 2: - **Index bazat pe o functie SQL**

- Indexul urmator este folosit pentru o tabela de angajati care este foarte mare si in general in aplicatii se folosesc extragerile de o singura inregistrare din baza de date.
- Daca presupunem ca cea mai frecventa metoda de filtrare a cererii SQL este dupa nume si prenume putem crea un index pe cele doua campuri reunite:

```
CREATE INDEX employees_name  
ON emp( CONCAT(ename, job) )
```

# Distrugerea indecșilor

- Indecșii pot fi distrusi relativ usor pentru ca nu este afectata structura sau datele bazei de date.
- Singura conditie necesara este ca userul care incearca sa execute aceasta actiune trebuie sa fie proprietarul indexului.
- De asemenea trebuie ca utilizatorul sa aiba dreptul de DROP ANY INDEX in cadrul sistemului de securitate.
- Distrugerea indecșilor este realizata utilizand urmatoarea interogare **SQL**:

**DROP INDEX** *nume\_index*;

unde *nume\_index* reprezinta numele indexului.

## 2. Indecșii

**Indecșii nu pot modificați.** Din aceasta cauza în momentul în care este necesară modificarea unui index este necesară distrugerea lui prealabilă.

**Exemplu** - Distrugerea indexului creat

- Pentru distrugerea indexului pe care l-am creat într-un exemplu anterior va trebui să folosim următoarea cerere:

**DROP INDEX employees\_name;**



# Cuprins

- 1. Secvente**
- 2. Indecsi**
- 3. Sinonime**

### 3. Sinonime

- *Sinonimele sunt niste aliasuri pe care le putem adauga oricarui obiect din baza de date pentru a simplifica accesul la aceasta resursa.*
- Sinonimele sunt folosite pentru cazul in care exista in diferite scheme din baza de date tabele cu mai multe denumiri, cand denumirile obiectelor sunt foarte lungi si/sau greu de tinut minte de catre programatorul care trebuie sa le foloseasca si in alte cazuri.

### 3. Sinonime

- Principalul avantaj al utilizarii lor este pentru micșorarea timpului necesar din partea utilizatorilor pentru a-si crea cererile **SQL**.
- În felul acesta tabelele, vederile, secvențele, procedurile sau alte obiecte au un nume alternativ pentru acces.

### 3. Sinonime

Pentru crearea sinonimelor este necesara o cerere **SQL** care are urmatorul format:

```
CREATE [PUBLIC] SYNONYM nume_sinonim  
FOR nume_obiect;
```

<b>PUBLIC</b>	Defineste faptul ca acest sinonim este accesibil de catre toti utilizatorii. Daca nu este precizat atunci sinonimul este disponibil doar pentru utilizatorul curent. Sinonimele publice pot fi create doar de catre utilizatorii care au drepturi administrative.
<b>nume_sinonim</b>	Numele sinonimului care va fi creat.
<b>nume_obiect</b>	Identifica obiectul pentru care se doreste crearea sinonimului

### 3. Sinonime

- Dacă dorim să accesăm o tabelă pe care un alt utilizator a creat-o în prealabil și careia îi dorim asignarea unui sinonim suntem obligați să introducem înaintea numelui tabelii și numele utilizatorului care a creat tabela, în fața numelui tabelii, despartite de un punct.

Exemplu 1 - În acest exemplu vom încerca să creăm un sinonim pentru tabela departamente care are un nume destul de lung.

```
CREATE PUBLIC SYNONYM dep  
FOR dept;
```

### 3. Sinonime

In felul acesta nu vom mai fi obligati sa accesam in cererile noastre **SQL** tabela de departamente prin scrierea numelui tablei 'department' si este suficienta doar utilizarea denumirii noi create: 'dep'

```
SELECT * FROM dep;
```

### 3. Sinonime

Exista unele restrictii in folosirea sinonimelor:

- obiectele respective nu pot sa fie continute intr-un pachet.
- numele sinonimului privat trebuie sa fie unic in cadrul grupului de obiecte pentru care acel user este proprietar.

## 3. Sinonime

- Un sinonim nu poate fi schimbat odata creat.
- El poate sa fie doar distrus.
- Distrugerea unui sinonim nu implica si distrugerea obiectului de care aliasul este atasat.
- Distrugerea unui sinonim este facuta folosind o interogare **SQL** de tipul urmator:

**DROP [PUBLIC] SYNONYM nume\_sinonim**

unde **nume\_sinonim** este numele sinonimului care sa fie distrus.



### 3. Sinonime

- Un sinonim de tip public poate sa fie distrus doar de catre un utilizator administrativ.
- Sinonimele private trebuie distruse de utilizatorul care le-a creat acestea nefiind disponibile in cadrul listei numelor de obiecte a celorlalti utilizatori.

# Bibliografie

Florin Rădulescu: Oracle SQL, PL/SQL, Editura  
Printech, ISBN 973-718-203-02005

# Întrebări?