

Universitatea Constantin Brâncuși din Târgu Jiu

Facultatea: Inginerie

*Program de conversie profesională a cadrelor didactice din învățământul preuniversitar: Informatică,
Tehnologia Informației și a Comunicațiilor*

Baze de date

Limbajul SQL

Teams: TIC - Baze de date-2022/2023

Adrian Runceanu

THE **INFORMATION** COMPANY

Curs 8

Limbajul SQL

Limbajul SQL

Constrângeri

Serverul **Oracle** utilizează constrângeri pentru a preveni pătrunderea de date invalide în tabele.

Putem utiliza constrângeri pentru a realiza următoarele acțiuni:

1. Impune reguli datelor unei tabele ori de câte ori un rând este inserat, modificat sau șters din tabela.
2. Prevenirea ștergerii unei tabele în cazul în care există dependență de alte tabele
3. Furnizarea regulilor pentru instrumentele **Oracle**, cum ar fi **ORACLE DEVELOPER**.

Tipuri de constrângeri

Constrangere	Descriere
NOT NULL	specifica faptul ca o coloana nu poate avea valoarea nula
UNIQUE	specifica o coloana sau o combinatie de coloane a carei valori trebuie sa fie unice pentru toate randurile din tabel
PRIMARY KEY	identifica fiecare rand al tablei
FOREIGN KEY	stabileste o relatie de cheie straina intre coloana si coloana tablei de referinta
CHECK	specifica o conditie care trebuie sa fie adevarata

Ghidul Constrângerilor

1. Toate constrângerile sunt cuprinse într-un dicționar.
2. Este ușor să se facă referință la constrângeri dacă li se dă nume sugestive.
3. Numele unei constrângeri trebuie să urmeze un anumit standard.
4. Dacă nu se denumește constrângerea, server-ul **Oracle** generează un nume de forma **SYS_Cn**, unde **n** este un număr întreg astfel încât numele constrângerii este unic.
5. Constrângerile definite pentru o anumita tabela pot fi vizualizate în **USER_CONSTRAINTS** (dicționarul tablei).

Definirea Constrângerilor

Sintaxa

CREATE TABLE [schema.] table

(column datatype [DEFAULT expr]

[column_constraint],

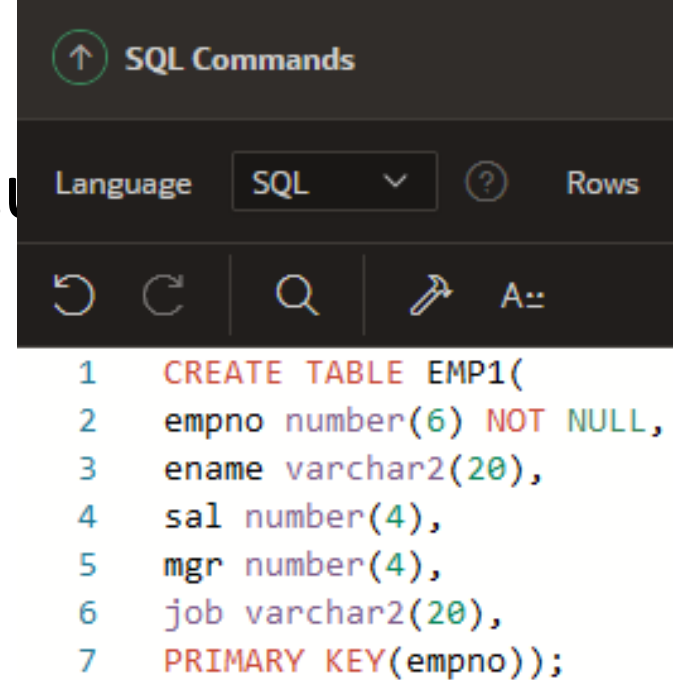
[table_constraint][...]);

În sintaxa prezentată anterior avem:

schema	este aceeași ca și numele titularului
table	este numele tabelului
DEFAULT expr	specifică o valoare predefinită care să fie utilizată dacă o valoare este omisă în declarația INSERT
column	este numele coloanei
datatype	este tipul datei și lungimea coloanei
column_constraint	este o constrangere de integritate ca parte a definiției coloanei
table_constraint	este o constrangere de integritate ca parte a definiției tabelului

EXEMPLU - Adăugarea unei
constrângerii unei tabele odată cu
crearea lui.

```
CREATE TABLE EMP1(  
empno number(6) NOT NULL,  
ename varchar2(20),  
sal number(4),  
mgr number(4),  
job varchar2(20),  
PRIMARY KEY(empno));
```



```
1 CREATE TABLE EMP1(  
2 empno number(6) NOT NULL,  
3 ename varchar2(20),  
4 sal number(4),  
5 mgr number(4),  
6 job varchar2(20),  
7 PRIMARY KEY(empno));
```

Constrângere
pentru coloană

Constrângere
pentru tabela

- De obicei constrângerile sunt create în același timp cu tabela.
- Constrângerile pot fi adăugate tablei după crearea ei.
- Constrângerile pot fi definite pe 2 nivele:

Nivelul constrangerii	Descriere
Coloana	Face referire la o singura coloana; poate defini orice tip de constrangere de integritate
Tabel	Face referire la una sau mai multe coloane; poate defini orice constrangere exceptand pe cea de tip NOT NULL

Sintaxa:

1. Constrângere la nivel de coloană

```
column [CONSTRAINT constraint_name] constraint_type
```

2. Constrângere la nivel de tabela

```
column,..  
[CONSTRAINT constraint_name] constraint_type  
(column,...),
```

- In sintaxa avem:

<i>constraint_name</i>	este numele constrangerii
<i>constraint_type</i>	este tipul constrangerii

Constrângerea **NOT NULL**

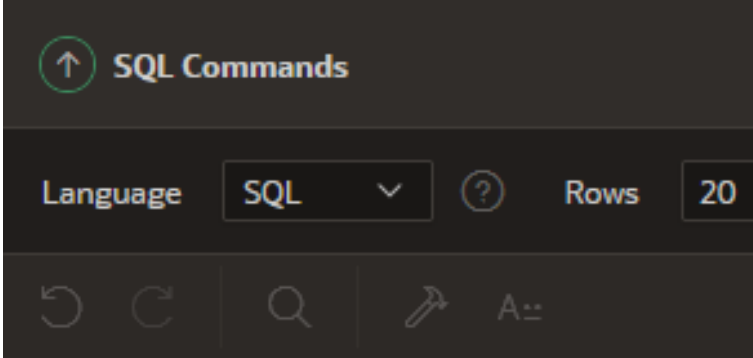
- Constrângerea de tip **NOT NULL** asigură faptul că o coloană să nu conțină valoarea nulă.
- Ea poate fi specificată la nivel de coloană și nu la nivel de tabela.

Exemplu

În exemplul următor constrângerea **NOT NULL** se aplică coloanelor ENAME și HIREDATE din tabela **emp_new**.

- Pentru coloana ENAME constrângerea nu este denumită astfel încât serverul **Oracle** o să creeze un nume pentru ea.
- Pentru coloana HIREDATE constrângerea este denumită: "**NOT NULL**".

```
CREATE TABLE emp_new(  
EMPNO number(6),  
ENAME varchar2(10) NOT NULL,  
SAL number(4),  
COMM number(4),  
HIREDATE number(4) NOT  
NULL);
```

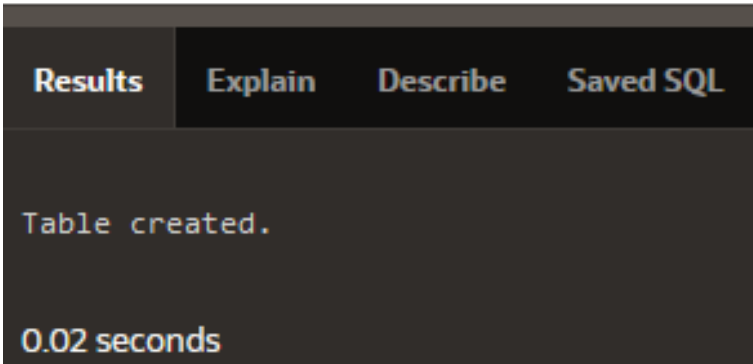


SQL Commands

Language SQL Rows 20

SQL Commands interface showing the SQL command being entered.

```
1 CREATE TABLE emp_new(  
2 EMPNO number(6),  
3 ENAME varchar2(10) NOT NULL,  
4 SAL number(4),  
5 COMM number(4),  
6 HIREDATE number(4) NOT NULL);  
7
```



Results Explain Describe Saved SQL

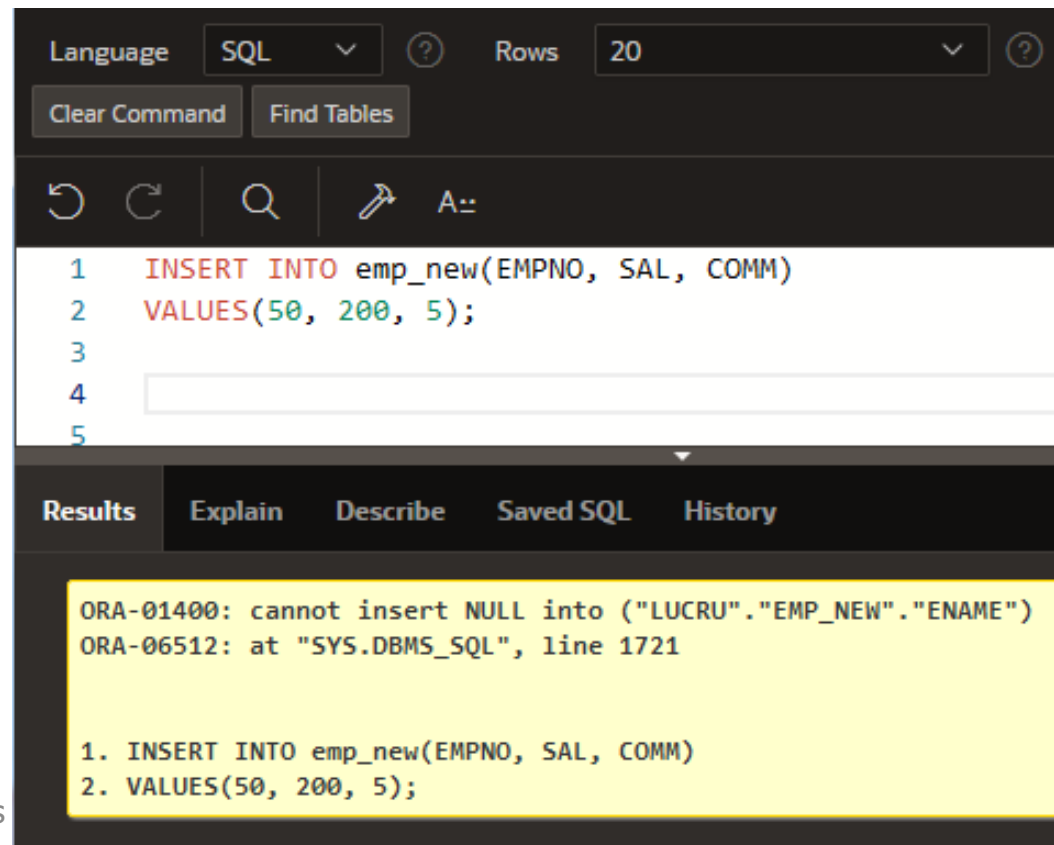
Table created.

0.02 seconds

Results interface showing the execution of the SQL command.

Acum vom încerca să inserăm valori doar în coloanele EMPNO, SAL, COMM, dar la execuție ne va da eroare pentru că ENAME și HIREDATE au valori nule iar constrângerile ne obligă să le atribuim o valoare.

INSERT INTO emp_new(EMPNO, SAL, COMM)
VALUES(50, 200, 5);



The screenshot shows a SQL IDE interface. At the top, there are controls for 'Language' (set to SQL) and 'Rows' (set to 20). Below these are buttons for 'Clear Command' and 'Find Tables'. The main editor area contains the following SQL code:

```
1  INSERT INTO emp_new(EMPNO, SAL, COMM)
2  VALUES(50, 200, 5);
3
4
5
```

Below the editor, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active and displays the following error message:

```
ORA-01400: cannot insert NULL into ("LUCRU"."EMP_NEW"."ENAME")
ORA-06512: at "SYS.DBMS_SQL", line 1721
```

At the bottom of the results pane, the executed SQL statements are listed:

```
1. INSERT INTO emp_new(EMPNO, SAL, COMM)
2. VALUES(50, 200, 5);
```


Constrângerea **UNIQUE**

*Constrângerea **UNIQUE** de integritate impune ca fiecare valoare a unei coloane sau set de coloane să fie unică - două rânduri ale aceluiași tabele să nu aibă aceleași valori într-o anumită coloana sau set de coloane.*

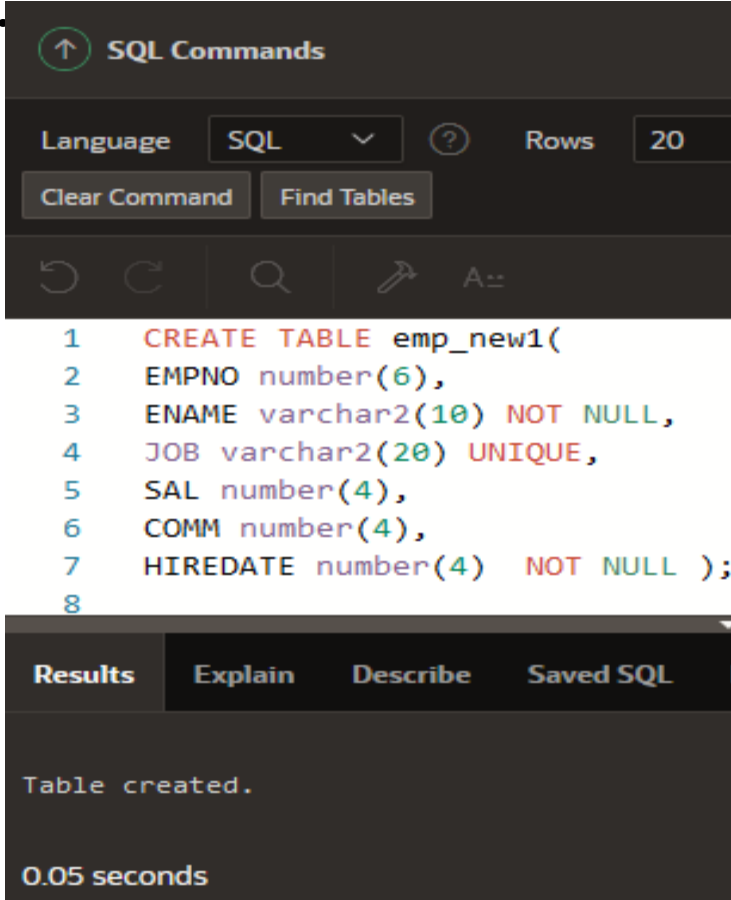
Permite includerea de valori nule numai dacă constrângerea **NOT NULL** nu este definită pentru aceeași coloană (valoarea nulă nu este considerată a fi echivalentă cu ceva).

Constrângerea **UNIQUE** poate fi definită atât la nivel de linie cât și la nivel de tabelă.

Exemplu

- În exemplul de mai jos se aplică constrângerea **UNIQUE** coloanei JOB a tabeli **emp_new1**.
- Numele constrângerii este **UNIQUE**.

```
CREATE TABLE emp_new1(  
EMPNO number(6),  
ENAME varchar2(10) NOT NULL,  
JOB varchar2(20) UNIQUE,  
SAL number(4),  
COMM number(4),  
HIREDATE number(4) NOT NULL );
```



```
SQL Commands  
Language SQL Rows 20  
Clear Command Find Tables  
1 CREATE TABLE emp_new1(  
2 EMPNO number(6),  
3 ENAME varchar2(10) NOT NULL,  
4 JOB varchar2(20) UNIQUE,  
5 SAL number(4),  
6 COMM number(4),  
7 HIREDATE number(4) NOT NULL );  
8  
Results Explain Describe Saved SQL  
Table created.  
0.05 seconds
```

Constrângerea Cheie Primară

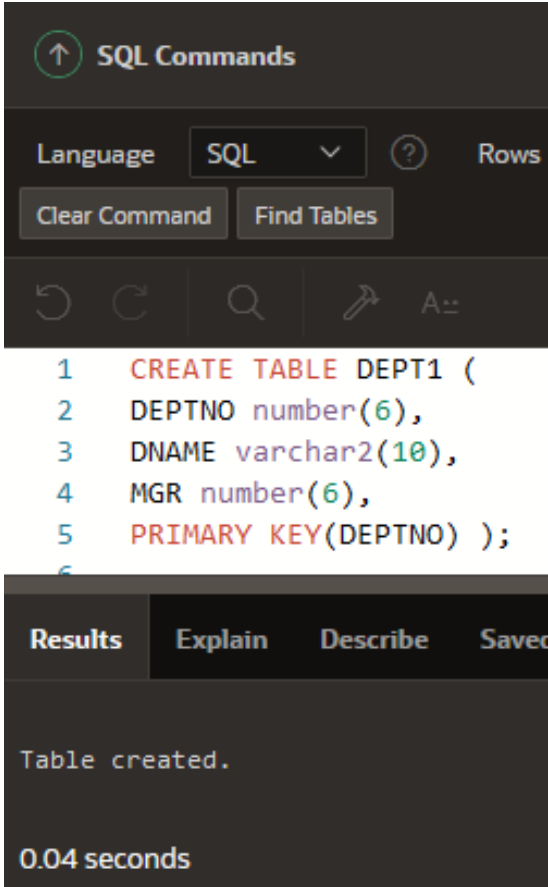
- Constrângerea **CHEIE PRIMARĂ (PRIMARY KEY)** creează o cheie primară pentru tabela.
- Numai o singură cheie poate fi creată pentru fiecare tabela.
- Această constrângere este o coloană sau un set de coloane care identifică în mod unic fiecare rând al tabelului.
- Nici o coloană care face parte din cheia primară nu poate conține valoarea nulă.
- Poate fi definită la nivel de coloană sau tabela.

*O tabela poate avea o singură cheie primară dar poate avea mai multe constrângeri de tip **UNIQUE**.*

Exemplu

În exemplul de mai jos este definită o constrângere de tip cheie primară la nivelul coloanei DEPTNO
Numele constrângerii este **PRIMARY KEY**.

```
CREATE TABLE DEPT1 (  
DEPTNO number(6),  
DNAME varchar2(10),  
MGR number(6),  
PRIMARY KEY(DEPTNO) );
```



```
SQL Commands  
Language SQL ? Rows  
Clear Command Find Tables  
1 CREATE TABLE DEPT1 (  
2 DEPTNO number(6),  
3 DNAME varchar2(10),  
4 MGR number(6),  
5 PRIMARY KEY(DEPTNO) );  
6  
Results Explain Describe Saved  
Table created.  
0.04 seconds
```

Constrângerea **FOREIGN KEY**

- Constrângerea **FOREIGN KEY** definește o coloană sau o combinație de coloane ca **foreign key** și *stabilește o relație între o cheie primară și una unică în aceeași tabelă sau în tabele diferite.*
- O valoare care apare într-o tabelă trebuie să se regăsească și în cea de-a 2-a tabelă, pe coloana unde formează cheia primară.
- Constrângerile de tip **FOREIGN KEY** pot fi definite la nivel de coloană sau tabelă.

Exemplu

În următorul exemplu se definește o constrângere de tip **FOREIGN KEY** coloanei DEPTNO a tabelii EMP_NEW2 utilizând sintaxa la nivel de tabela.

```
CREATE TABLE EMP_NEW2(
EMPNO number(6),
ENAME varchar2(10),
SAL number(4),
COM number(4),
JOB varchar2(25) UNIQUE,
DEPTNO number(4),
FOREIGN KEY(DEPTNO) REFERENCES DEPT (DEPTNO) );
```

The screenshot shows a SQL IDE interface with the following elements:

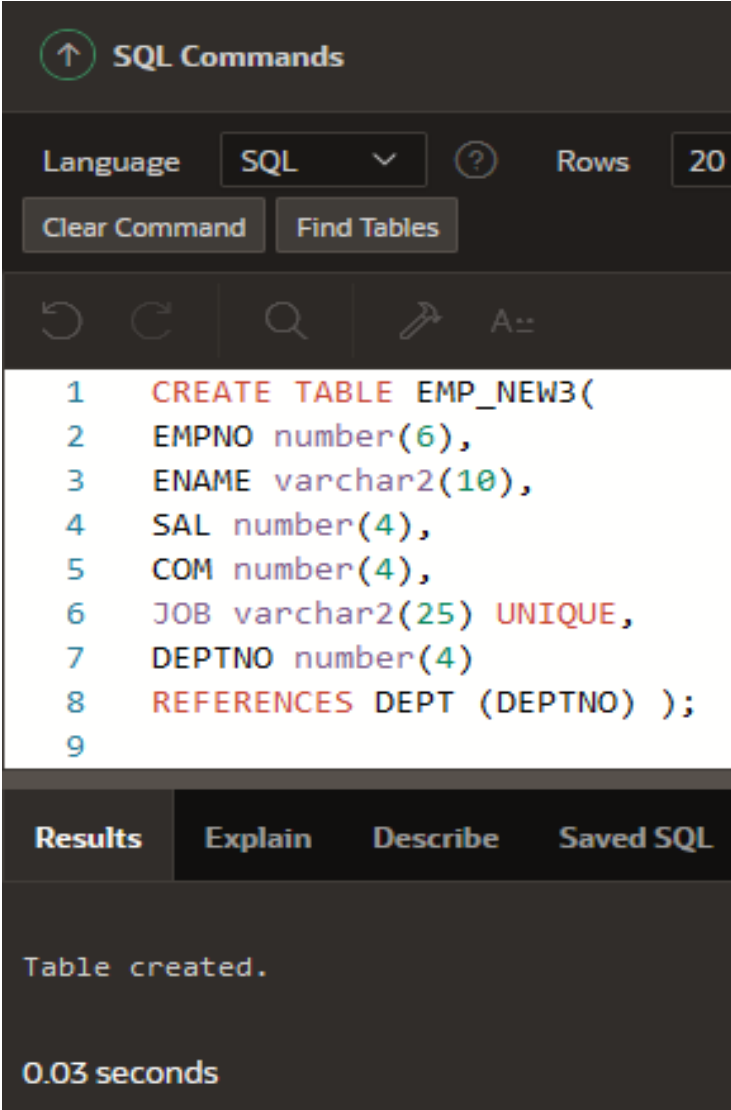
- SQL Commands** window title.
- Schema**: LUCRU
- Language**: SQL
- Rows**: 20
- Buttons**: Clear Command, Find Tables
- SQL Code**:


```
1 CREATE TABLE EMP_NEW2(
2 EMPNO number(6),
3 ENAME varchar2(10),
4 SAL number(4),
5 COM number(4),
6 JOB varchar2(25) UNIQUE,
7 DEPTNO number(4),
8 FOREIGN KEY(DEPTNO) REFERENCES DEPT (DEPTNO) );
9
```
- Results** tab selected, showing:


```
Table created.
0.04 seconds
```

Se poate defini și la nivel de coloană. Sintaxa este următoarea:

```
CREATE TABLE EMP_NEW3(  
EMPNO number(6),  
ENAME varchar2(10),  
SAL number(4),  
COM number(4),  
JOB varchar2(25) UNIQUE,  
DEPTNO number(4)  
REFERENCES DEPT (DEPTNO) );
```



The screenshot shows a SQL command interface with the following elements:

- SQL Commands** header with an upward arrow icon.
- Language** dropdown menu set to **SQL**.
- Rows** dropdown menu set to **20**.
- Clear Command** and **Find Tables** buttons.
- Navigation icons: back, forward, search, and refresh.
- SQL Command:**

```
1 CREATE TABLE EMP_NEW3(  
2 EMPNO number(6),  
3 ENAME varchar2(10),  
4 SAL number(4),  
5 COM number(4),  
6 JOB varchar2(25) UNIQUE,  
7 DEPTNO number(4)  
8 REFERENCES DEPT (DEPTNO) );  
9
```
- Results** tab selected, showing **Table created.**
- 0.03 seconds** execution time.

- După cum am observat a dispărut din sintaxa **FOREIGN KEY**.
- O constrângere de tip **FOREIGN KEY** *este definită într-o tabela copil, iar tabela care conține coloana la care se face referință este părintele.*

O **FOREIGN KEY** este definită utilizând o combinație a următoarelor cuvinte cheie:

- **FOREIGN KEY** este utilizată pentru a defini o coloană în tabelul copil la nivel de tabela
- **REFERENCES** identifică tabela și coloana în tabela părinte
- **ON DELETE CASCADE** indică faptul că atunci când rândul din tabela părinte va fi șters, rândul dependent din tabela copil va fi de asemenea șters.
- **ON DELETE SET NULL** convertește valorile **FOREIGN KEY** în valori nule atunci când valoarea părinte este ștersă.

Constrangerea de tip **CHECK**

Constrângerea de tip **CHECK** *definește o condiție pe care fiecare rând trebuie să o îndeplinească.*

Următoarele expresii **nu sunt permise**:

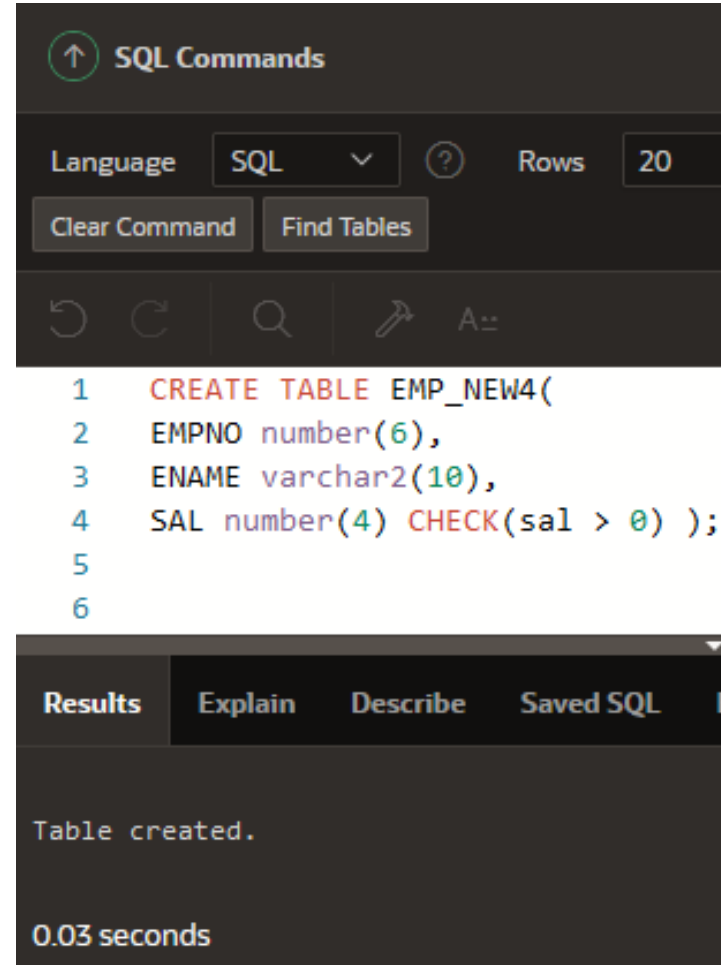
- Referințe la pseudocoloanele **CURRVAL**, **NEXTVAL**, **LEVEL** și **ROWNUM**.
- Apelul funcțiilor **SYSDATE**, **UID**, **USER**, și **USERENV**.
- Cereri care se referă la alte valori ale altor rânduri

Nu există un număr limitat de constrângeri de tip **CHECK** pe care să le definim pe o coloană.

Constrângerea de tip **CHECK** poate fi definită atât la nivel de coloană cât și la nivel de tabela.

Exemplu

```
CREATE TABLE EMP_NEW4(  
EMPNO number(6),  
ENAME varchar2(10),  
SAL number(4) CHECK(sal > 0) );
```



The screenshot shows a SQL IDE interface. At the top, there is a "SQL Commands" header with an upward arrow icon. Below this, there are controls for "Language" (set to "SQL"), "Rows" (set to "20"), and buttons for "Clear Command" and "Find Tables". A toolbar with icons for undo, redo, search, and refresh is also visible. The main area displays the SQL command being executed, with line numbers 1 through 6 on the left. The command is: `1 CREATE TABLE EMP_NEW4(`, `2 EMPNO number(6),`, `3 ENAME varchar2(10),`, `4 SAL number(4) CHECK(sal > 0));`, `5`, `6`. Below the command, there are tabs for "Results", "Explain", "Describe", and "Saved SQL". The "Results" tab is active, showing the message "Table created." and the execution time "0.03 seconds".

Adăugarea unei constrângeri

Se poate adăuga o constrângere pentru o tabela existent utilizând **ALTER TABLE** și clauza **ADD**.

Sintaxa

```
ALTER TABLE table  
ADD [CONSTRAINT constraint] type(column);
```

În sintaxă avem:

table	este numele tabelului
constraint	este numele constrangerii
type	este tipul constrangerii
column	este numele coloanei afectate de constrangere

Observații

- Se poate adăuga, șterge, activa sau dezactiva o constrângere, dar nu-i putem modifica structura.
- Se poate adăuga o constrângere **NOT NULL** la o coloană existentă utilizând clauza **MODIFY** a declarației **ALTER TABLE**.

Exemplu

Se poate adăuga o constrângere și unei tabele existent (nu numai odată cu crearea lui).

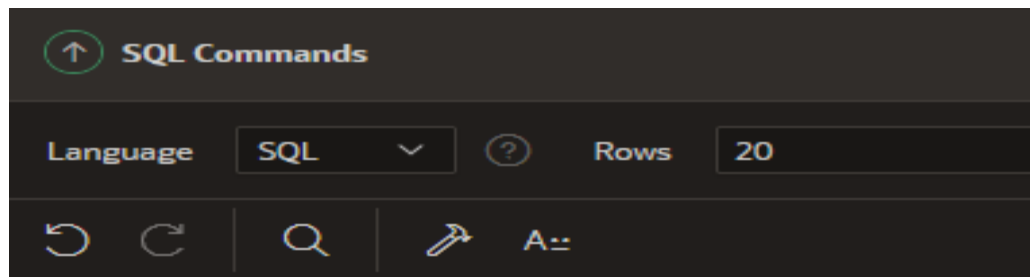
În următorul exemplu vom crea o constrângere **FOREIGN KEY** în tabela EMP.

Constrângerea asigură existența unui manager dacă există angajat în tabela EMP.

ALTER TABLE EMP

ADD CONSTRAINT FK_Mgr

FOREIGN KEY(Mgr) **REFERENCES** EMP (empno)



The screenshot shows a dark-themed SQL command window. At the top, it says "SQL Commands" with an upward arrow icon. Below that, there are controls for "Language" (set to "SQL") and "Rows" (set to "20"). At the bottom of the window, there are icons for undo, redo, search, and execute. The SQL commands are displayed in a list:

Curs 8 -

```
1 ALTER TABLE EMP
2 ADD CONSTRAINT FK_Mgr
3 FOREIGN KEY (Mgr) REFERENCES EMP(empno)
```

Ștergerea unei constrângeri

Pentru a șterge o constrângere trebuie utilizată declarația **ALTER TABLE** cu clauza **DROP**.

Opțiunea **CASCADE** a clauzei **DROP** face ca și constrângerea dependentă să fie ștearsă.

Sintaxa

```
ALTER TABLE table  
DROP PRIMARY KEY| UNIQUE (column)|  
CONSTRAINT constraint [CASCADE];
```


În sintaxa avem

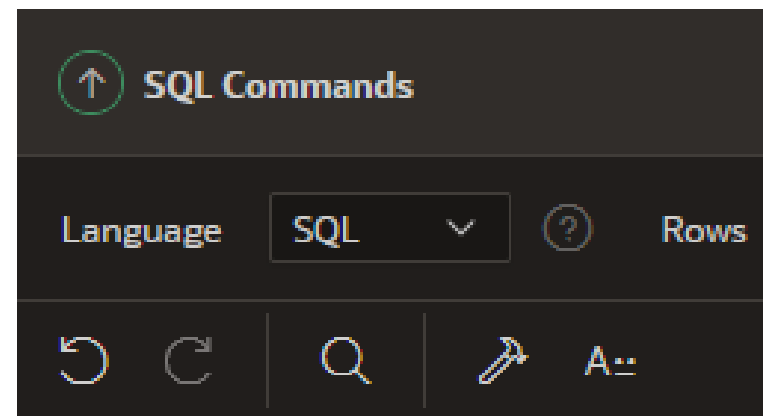
table	este numele tabelului
constraint	este numele constrangerii
column	este numele coloanei afectate de constrangere

Exemplu

1. În exemplul următor vom șterge constrângerea la nivelul coloanei MGR din tabela.

```
ALTER TABLE EMP
```

```
DROP CONSTRAINT FK_Mgr
```

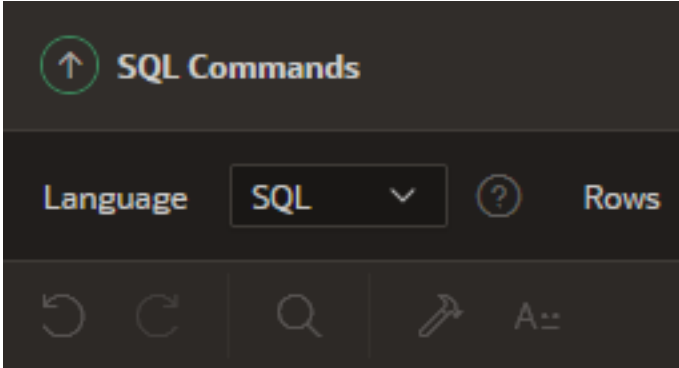


```
1 ALTER TABLE EMP
```

```
2 DROP CONSTRAINT FK_Mgr
```

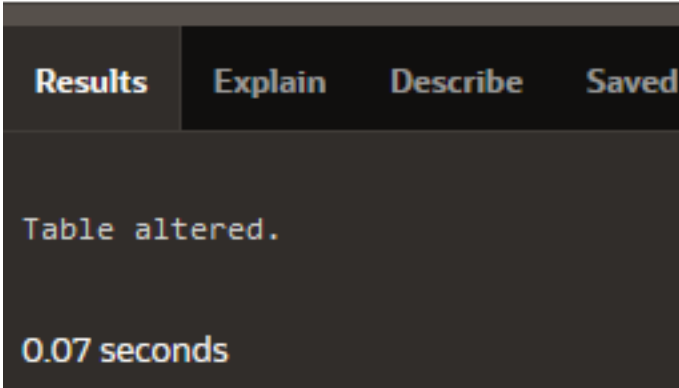
2. În următorul exemplu vom șterge constrângerea cheie primară din tabela **DEPT** și cheia străină asociată coloanei DEPTNO din tabela emp_new2.

**ALTER TABLE DEPT
DROP PRIMARY KEY CASCADE**

A screenshot of a SQL interface. At the top, it says "SQL Commands" with an upward arrow icon. Below that, there's a "Language" dropdown menu set to "SQL" and a "Rows" label. There are also several icons: a refresh icon, a search icon, a hammer icon, and a keyboard icon labeled "A++".

```
↑ SQL Commands  
Language SQL ? Rows  
↻ 🔍 🛠️ A++
```

```
1 ALTER TABLE DEPT  
2 DROP PRIMARY KEY CASCADE  
3
```

A screenshot of the "Results" tab in the SQL interface. It shows the message "Table altered." and the execution time "0.07 seconds".

```
Results Explain Describe Saved  
Table altered.  
0.07 seconds
```

Dezactivarea unei constrângeri

Dezactivarea constrângerii se efectuează cu declarația **ALTER TABLE** însoțită de clauza **DISABLE**.

Sintaxa

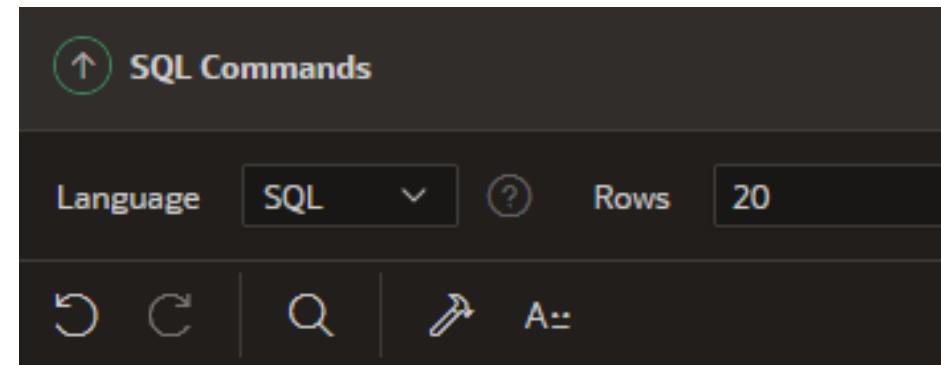
```
ALTER TABLE table  
DISABLE CONSTRAINT constraint  
[CASCADE];
```

În sintaxa avem:

<i>table</i>	este numele tabelului
<i>constraint</i>	este numele constrangerii

Observații

- Se poate utiliza clauza **DISABLE** atât în declarația **CREATE TABLE** cât și în **ALTER TABLE**.
- Clauza **CASCADE** dezactivează constrângeri de integritate dependente



Exemplu

ALTER TABLE DEPT

DISABLE CONSTRAINT deptno_pk CASCADE

Activarea unei constrângeri

Se poate activa o constrângere fără a o șterge sau recreea utilizând **ALTER TABLE** cu clauza **ENABLE**.

Sintaxa

```
ALTER TABLE table  
ENABLE CONSTRAINT constraint [CASCADE];
```

În sintaxa avem:

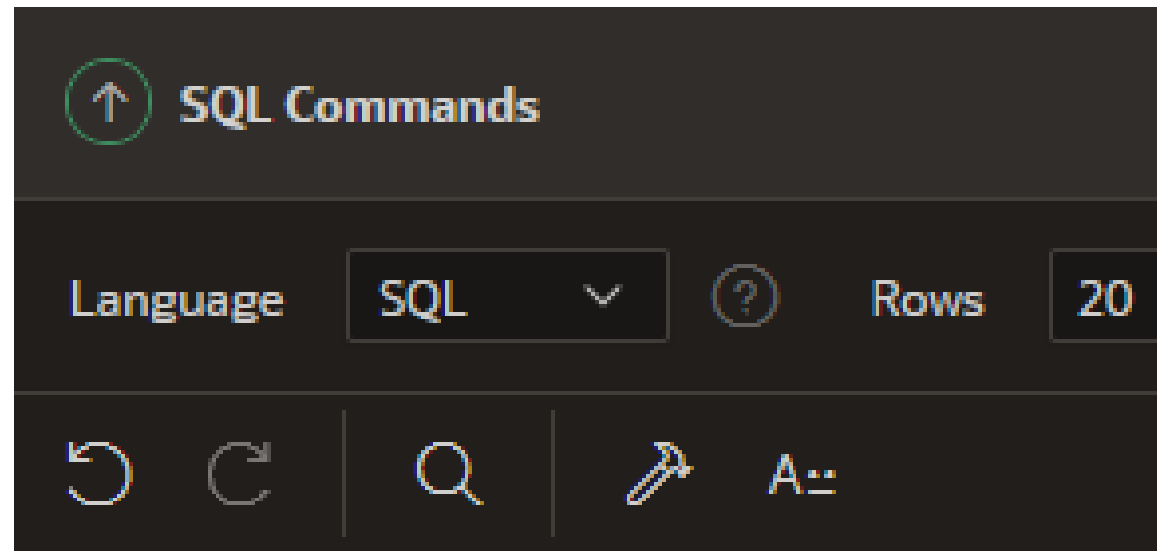
<i>table</i>	este numele tabelului
<i>constraint</i>	este numele constrangerii

Observații

1. Dacă se activează o constrângere, această constrângere se aplică la toate datele din tabela.
1. Dacă se activează o constrângere **UNIQUE** sau **PRIMARY** se creează automat un index **UNIQUE** sau **PRIMARY**.
1. Clauza **ENABLE** se poate utiliza în ambele declarații **CREATE TABLE** cât și **ALTER TABLE**.

Exemplu

ENABLE CONSTRAINT deptno_pk



```
1 ENABLE CONSTRAINT deptno_pk
```

Constrângeri Cascadate

- Constrângerile de tip **CASCADE** sunt utilizate cu clauza **DROP COLUMN**.
- Constrângerea **CASCADE** *șterge toate constrângerile de integritate ce se referă la cheile primare și unice definite în coloanele șterse.*
- Șterge de asemenea toate constrângerile multicolore definite în coloanele șterse.

Vizualizare Constrângeri

- După ce creem o tabela putem verifica existența lui utilizând o comanda **DESCRIBE**.
- Singura constrângere ce se poate verifica este constrângerea **NOT NULL**.
- Pentru a se vizualiza toate constrângerile din tabela trebuie interogată tabela **USER-CONSTRAINTS**.

Exemplu

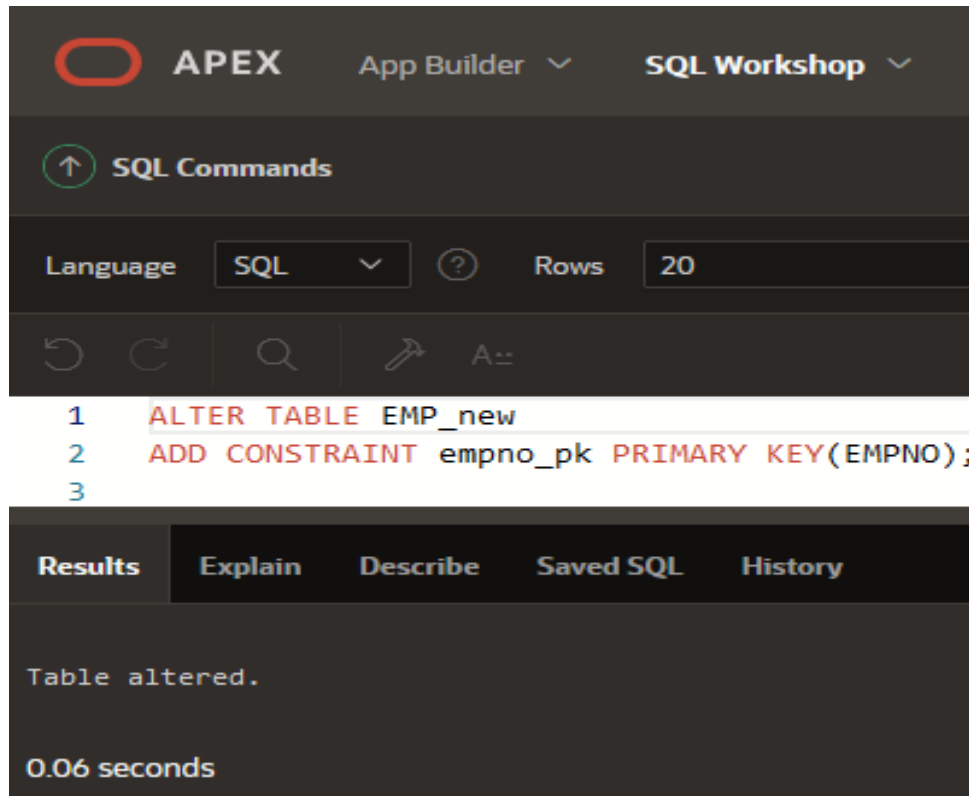
```
SELECT constraint_name, table_name  
FROM user_constraints  
WHERE table_name = 'emp_new'
```

Exerciții Propuse

1. Adăugați o constrângere **PRIMARY KEY** tablei **EMP_NEW** pe coloana **EMPNO**. Constrângerea trebuie unică la creare. Numele constrângerii este empno_pk.
2. Ștergeți constrângerea creată mai sus.
3. Vizualizați constrângerile din tabela **EMP_NEW**.

Soluție 1:

```
ALTER TABLE EMP_NEW  
ADD CONSTRAINT empno_pk PRIMARY KEY(EMPNO);
```



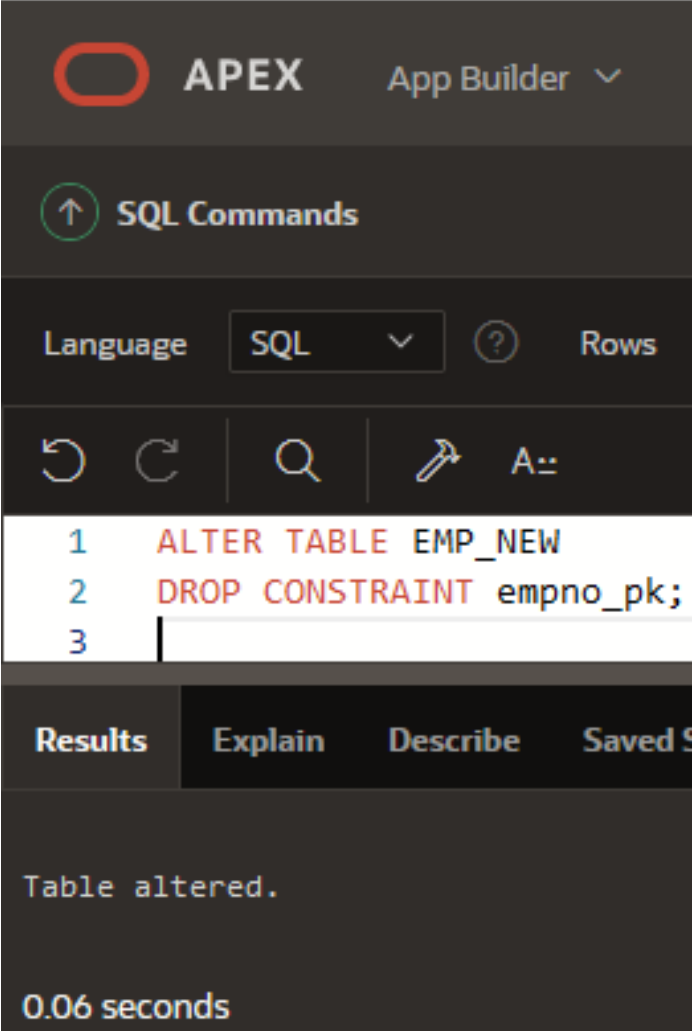
The screenshot shows the Oracle SQL Workshop interface. At the top, there are tabs for 'APEX App Builder' and 'SQL Workshop'. Below the tabs, there is a section for 'SQL Commands' with a language dropdown set to 'SQL' and a 'Rows' limit of '20'. The main area displays the following SQL command:

```
1 ALTER TABLE EMP_new  
2 ADD CONSTRAINT empno_pk PRIMARY KEY(EMPNO);  
3
```

Below the command, there is a 'Results' section with tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected, showing the message 'Table altered.' and the execution time '0.06 seconds'.

Soluție 2:

```
ALTER TABLE EMP_NEW  
DROP CONSTRAINT empno_pk;
```



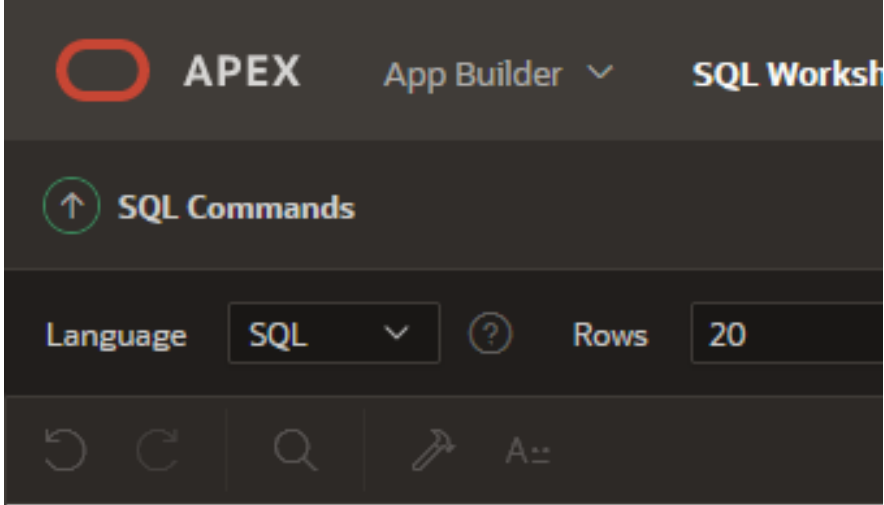
The screenshot shows the APEX SQL Commands interface. The top bar displays the APEX logo and 'App Builder'. Below it, the 'SQL Commands' section is active. The 'Language' dropdown is set to 'SQL'. The command editor contains the following SQL code:

```
1 ALTER TABLE EMP_NEW  
2 DROP CONSTRAINT empno_pk;  
3
```

Below the editor, there are tabs for 'Results', 'Explain', 'Describe', and 'Saved'. The 'Results' tab is selected, showing the message 'Table altered.' and the execution time '0.06 seconds'.

Soluție 3:

```
SELECT constraint_name, table_name  
FROM user_constraints  
WHERE table_name = 'emp_new'
```



The screenshot shows the APEX SQL Workshop interface. At the top, there is a navigation bar with the APEX logo, 'App Builder', and 'SQL Worksh'. Below this is a section for 'SQL Commands' with an upward arrow icon. The interface includes a 'Language' dropdown menu set to 'SQL', a help icon, and a 'Rows' dropdown menu set to '20'. At the bottom, there is a toolbar with icons for undo, redo, search, and a keyboard shortcut 'A::'. The main area displays the SQL query:

```
1  SELECT constraint_name, table_name  
2  FROM user_constraints  
3  where table_name = 'emp_new'
```

Întrebări?