Universitatea Constantin Brâncuşi 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 şters din tabela.
2. Prevenirea ştergerii unei tabela în cazul în care există dependenţă 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 tabelei
FOREIGN KEY	stabileste o relatie de cheie straina intre coloana si coloana tabelei de referinta
CHECK	specifica o conditie care trebuie sa fie adevarata

Ghidul Constrângerilor

1. Toate constrângerile sunt cuprinse într-un dicţionar.
2. Este uşor să se facă referinţă la constrângeri dacă li se dă nume sugestive.
3. Numele unei constrângeri trebuie să urmeze un anumit standard.
4. Dacă nu se denumeşte 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** (dicţionarul tabelei).

Definirea Constrângerilor

Sintaxa

CREATE TABLE [schema.] table

(column datatype [DEFAULT expr]

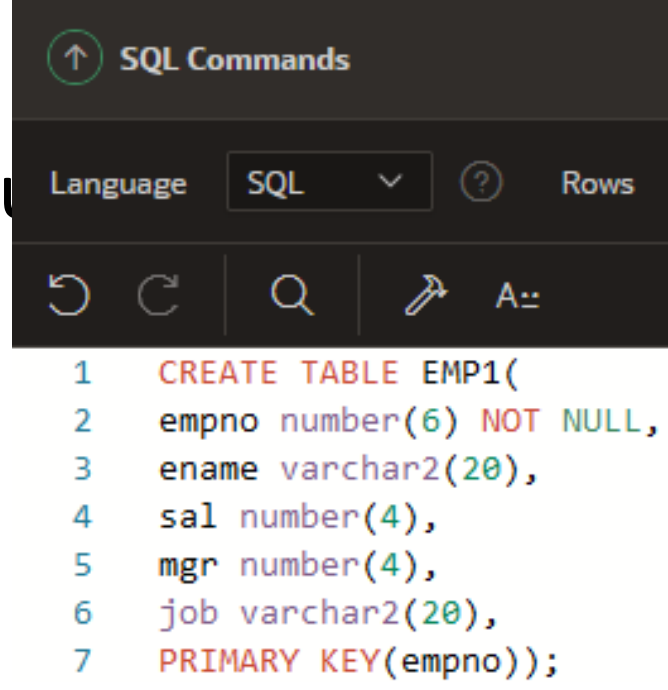
[column_constraint],

[table_constraint][...]);

În sintaxa prezentată anterior avem:

schema	este aceeasi ca si numele titularului
table	este numele tabelei
DEFAULT expr	specifica o valoare predefinita care sa fie utilizata daca o valoare este omisa in declaratia INSERT
column	este numele coloanei
datatype	este tipul datei si lungimea coloanei
column_constraint	este o **constrangere de integritate ca parte a definitiei coloanei**
table_constraint	este o **constrangere de integritate ca parte a definitiei tabelei**

EXEMPLU - Adăugarea unei constrângeri 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));
```

Constrângere pentru coloană

Constrângere pentru tabela

- De obicei constrângerile sunt create în acelaşi timp cu tabela.
- Constrângerile pot fi adăugate tabelei 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:

constraint_name	este numele constrangerii
constraint_type	este tipul constrangerii

Constrângerea **NOT NULL**

- Constrângerea de tip **NOT NULL** asigură faptul că o coloană să nu conţină valoarea nulă.

- Ea poate fi specificată la nivel de coloană şi nu la nivel de tabela.

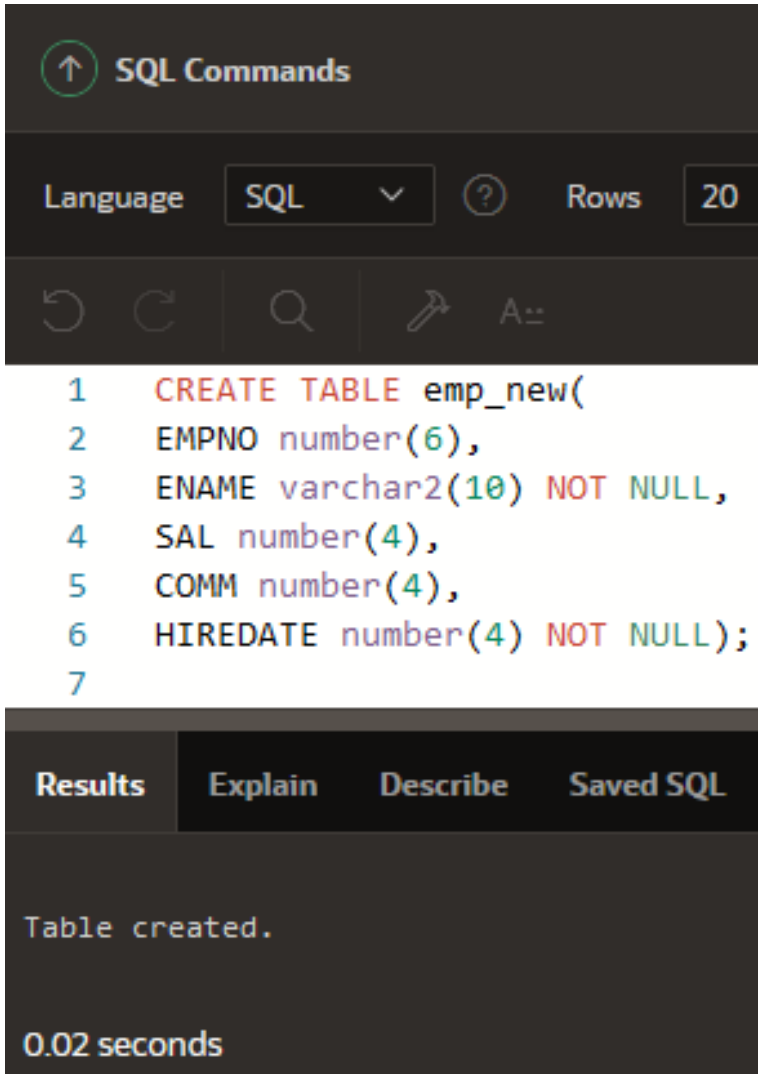
Exemplu

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

- Pentru coloana ENAME constrângerea nu este denumită astfel încât serverul **Oracle** o să creeze un nume pentru ea.

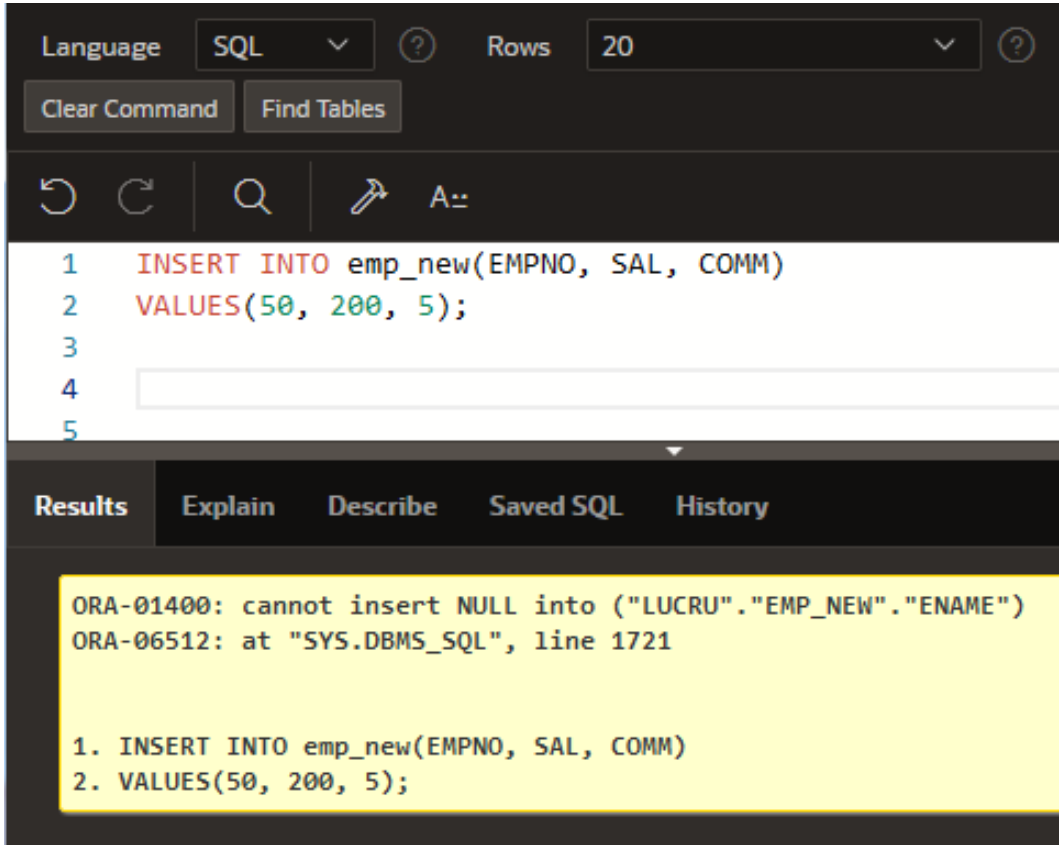
- Pentru coloana HIREDATE constrângerea este denumită: "**NOT NULL**".

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



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

INSERT INTO emp_new(EMPNO, SAL, COMM)
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 aceluiaşi tabele să nu aibă aceleaşi valori într-o anumită coloana sau set de coloane.*

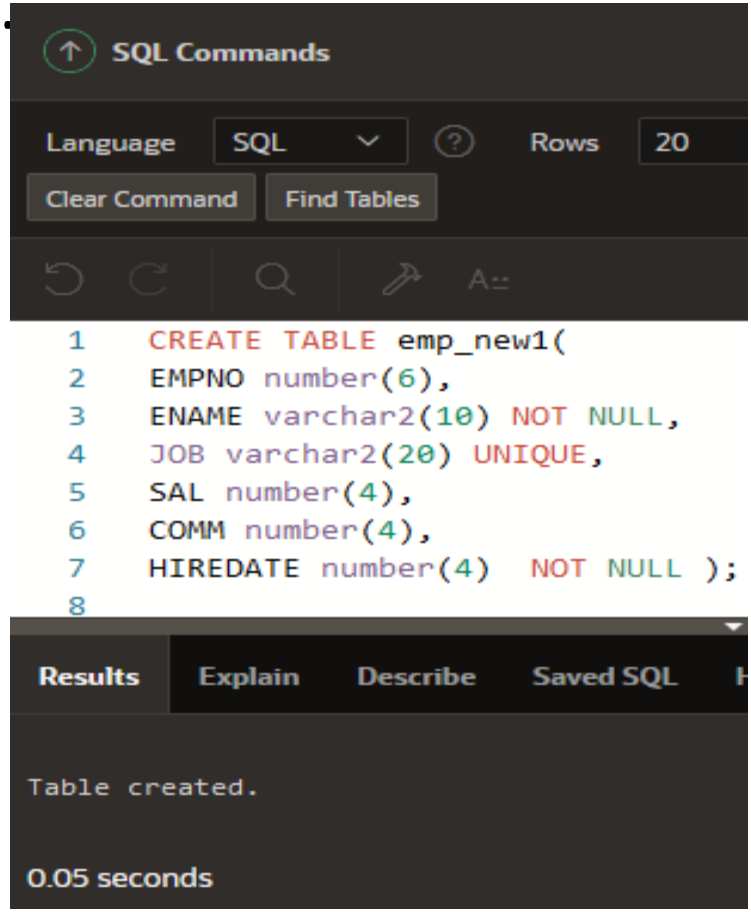
Permite includerea de valori nule numai dacă constrângerea **NOT NULL** nu este definită pentru aceeaşi coloană (valoarea nulă nu este considerată a fi echivalentă cu ceva).

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

Exemplu

- În exemplul de mai jos se aplică constrângerea **UNIQUE** coloanei JOB a tabelei **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 );
```



Constrângerea **Cheie Primară**

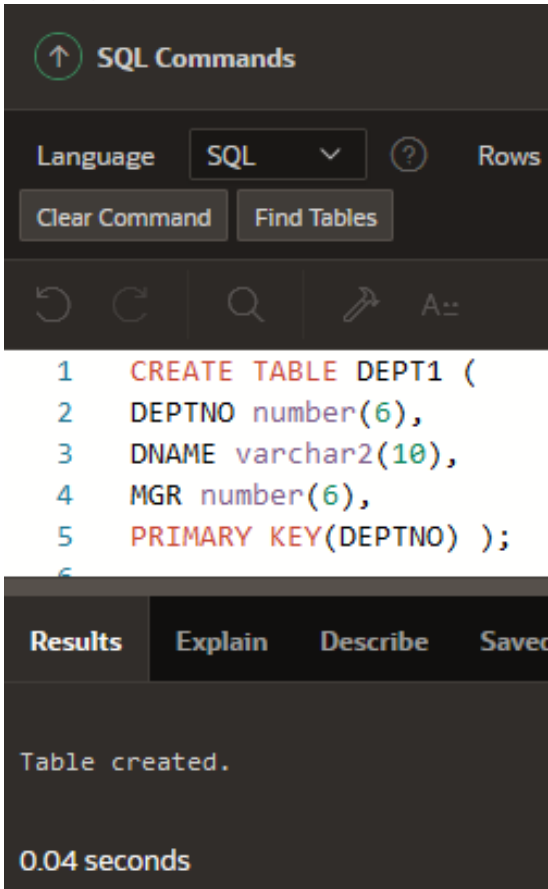
- Constrângerea **CHEIE PRIMARĂ(PRIMARY KEY)** creează o cheie primară pentru tabela.
- Numai o singură cheie poate fi creată pentru fiecare tabela.
- Această constrângere este o coloană sau un set de coloane care identifică în mod unic fiecare rând al tabelului.
- Nici o coloană care face parte din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



Constrângerea **FOREIGN KEY**

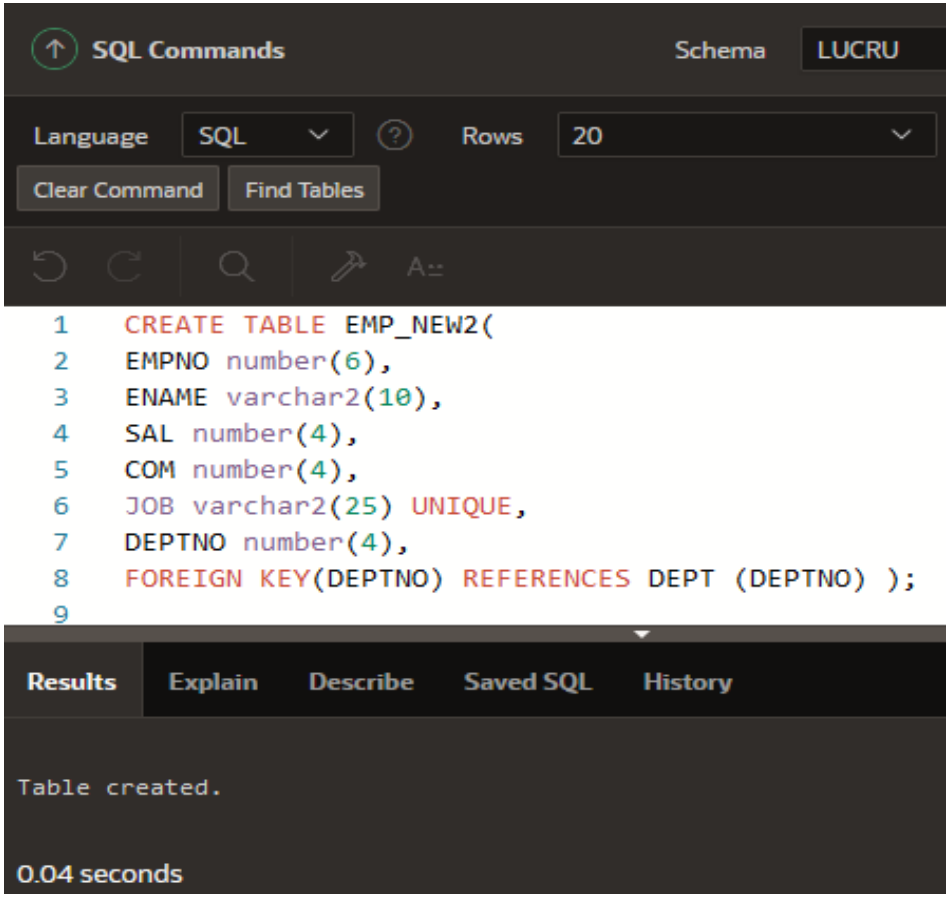
- Constrângerea **FOREIGN KEY** defineşte o coloană sau o combinaţie de coloane ca foreign key şi *stabileşte o relaţie între o cheie primară şi una unică în aceeaşi tabela sau în tabele diferite*.

- O valoare care apare într-o tabela trebuie să se regăsească şi în cea de-a 2-a tabela, pe coloana unde formează cheia primară.

- Constrângerile de tip **FOREIGN KEY** pot fi definite la nivel de coloană sau tabela.

Exemplu

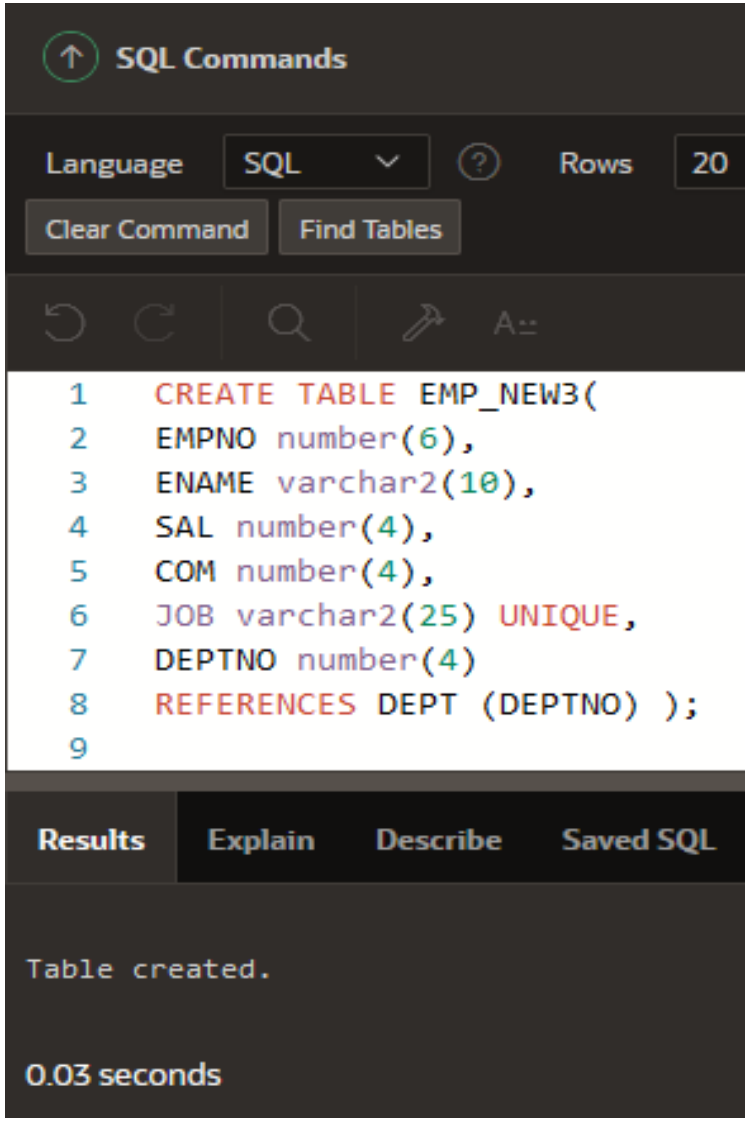
În următorul exemplu se defineşte o constrângere de tip **FOREIGN KEY** coloanei DEPTNO a tabelei EMP_NEW2 utilizând sintaxa la nivel de tabela.




```
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) );
```

Se poate defini şi 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));



- 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 conţine coloana la care se face referinţă este părintele*.

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

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

Constrangerea de tip **CHECK**

Constrângerea de tip **CHECK** *defineşte o condiţie pe care fiecare rând trebuie să o îndeplinească*.

Următoarele expresii nu sunt permise:

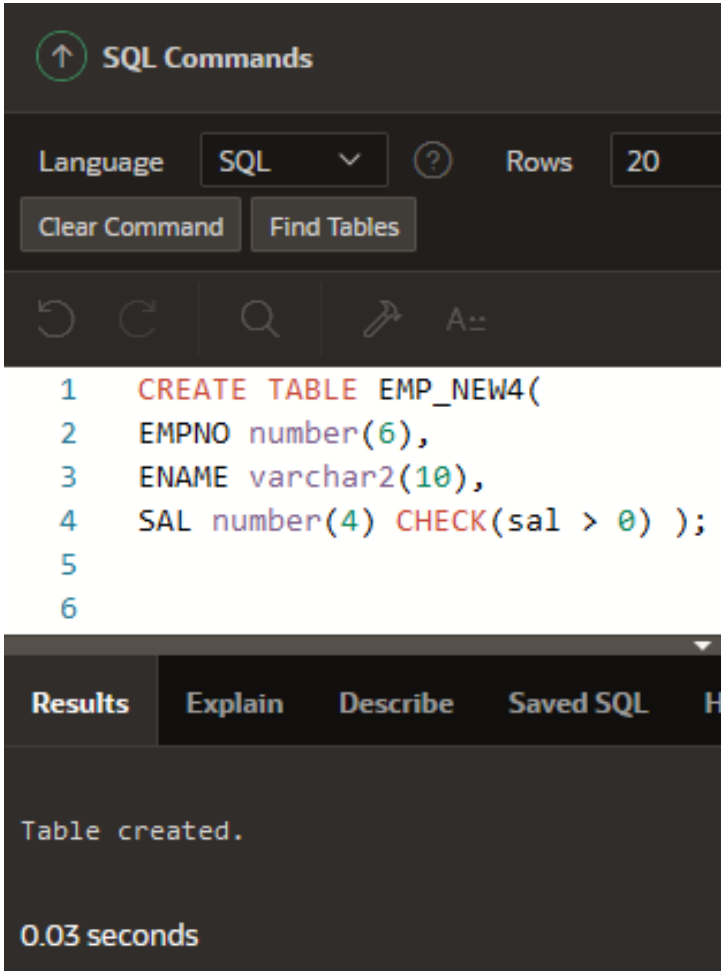
- Referinţe la pseudocoloanele **CURRVAL, NEXTVAL, LEVEL** şi **ROWNUM.**
- Apelul funcţiilor **SYSDATE, UID, USER,** şi **USERENV.**
- Cereri care se referă la alte valori ale altor rânduri

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 şi la nivel de tabela.

Exemplu

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



Adăugarea unei constrângeri

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

Sintaxa

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

În sintaxă avem:

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

Observaţii

- Se poate adăuga, şterge, 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 declaraţiei **ALTER TABLE**.

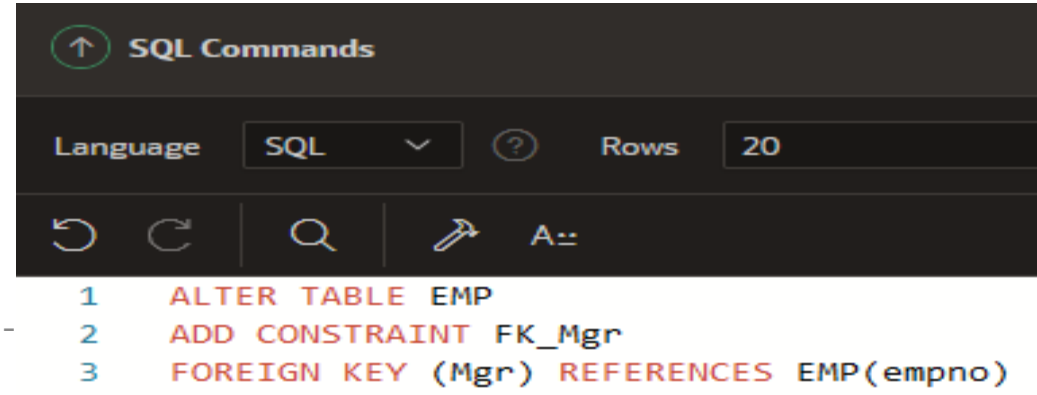
Exemplu

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

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

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

ALTER TABLE EMP
ADD CONSTRAINT FK_Mgr
FOREIGN KEY(Mgr) **REFERENCES** EMP (empno)



Ştergerea unei constrângeri

Pentru a şterge o constrângere trebuie utilizată declaraţia **ALTER TABLE** cu clauza **DROP**.

Opţiunea **CASCADE** a clauzei **DROP** face ca şi constrângerea dependentă să fie ştearsă.

Sintaxa

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

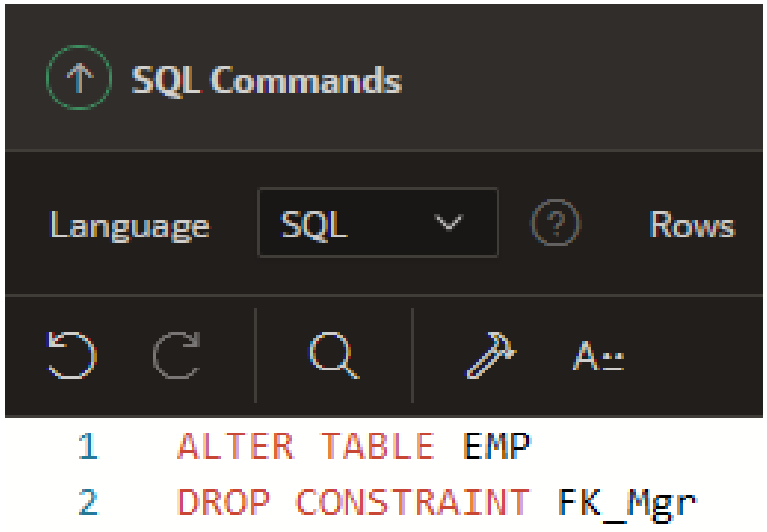
În sintaxa avem

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

Exemplu

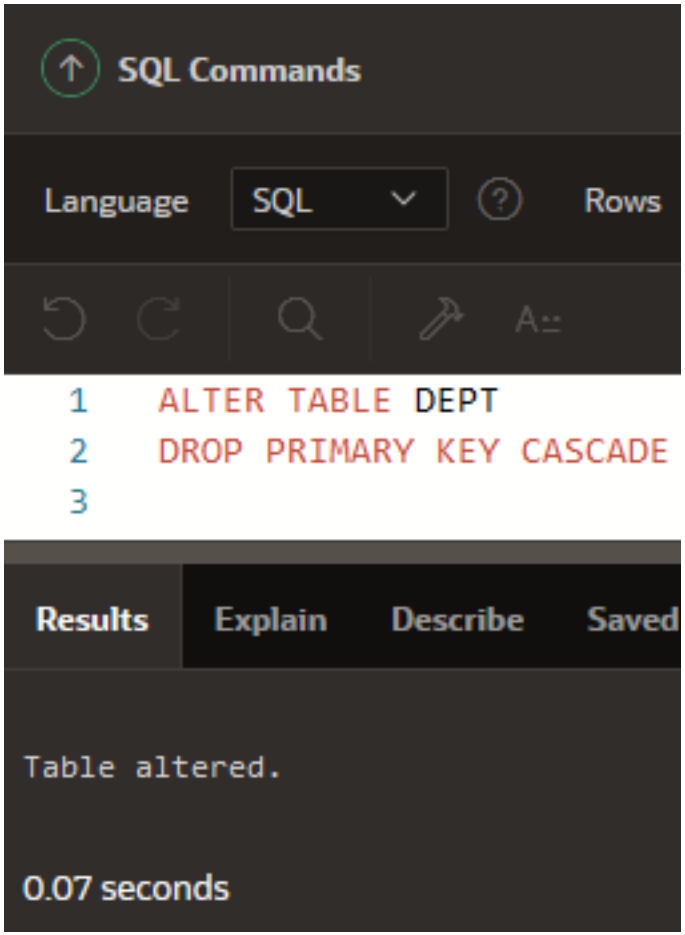
1. În exemplul următor vom şterge constrângerea la nivelul coloanei MGR din tabela.

ALTER TABLE EMP
DROP CONSTRAINT FK_Mgr



2. În următorul exemplu vom şterge constrângerea cheie primară din tabela DEPT şi cheia străină asociată coloanei DEPTNO din tabela emp_new2.

ALTER TABLE DEPT
DROP PRIMARY KEY CASCADE



Dezactivarea unei constrângeri

Dezactivarea constrângerii se efectuează cu declaraţia **ALTER TABLE** însoţită de clauza **DISABLE**.

Sintaxa

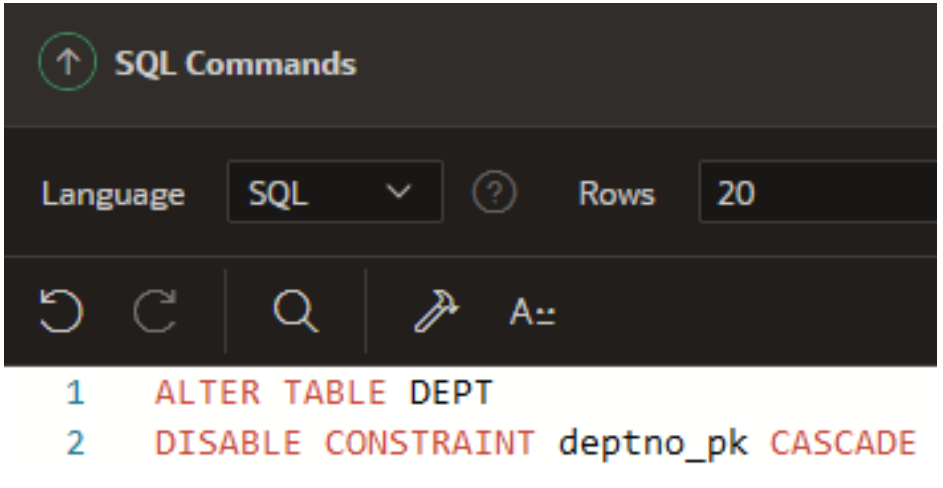
```
ALTER TABLE table
DISABLE CONSTRAINT constraint
[CASCADE];
```

În sintaxa avem:

table	este numele tabelului
constraint	este numele constrangerii

Observaţii

- Se poate utiliza clauza **DISABLE** atât în declaraţia **CREATE TABLE** cât şi 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 şterge sau recreea utilizând **ALTER TABLE** cu clauza **ENABLE**.

Sintaxa

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

În sintaxa avem:

table	este numele tabelului
constraint	este numele constrangerii

Observaţii

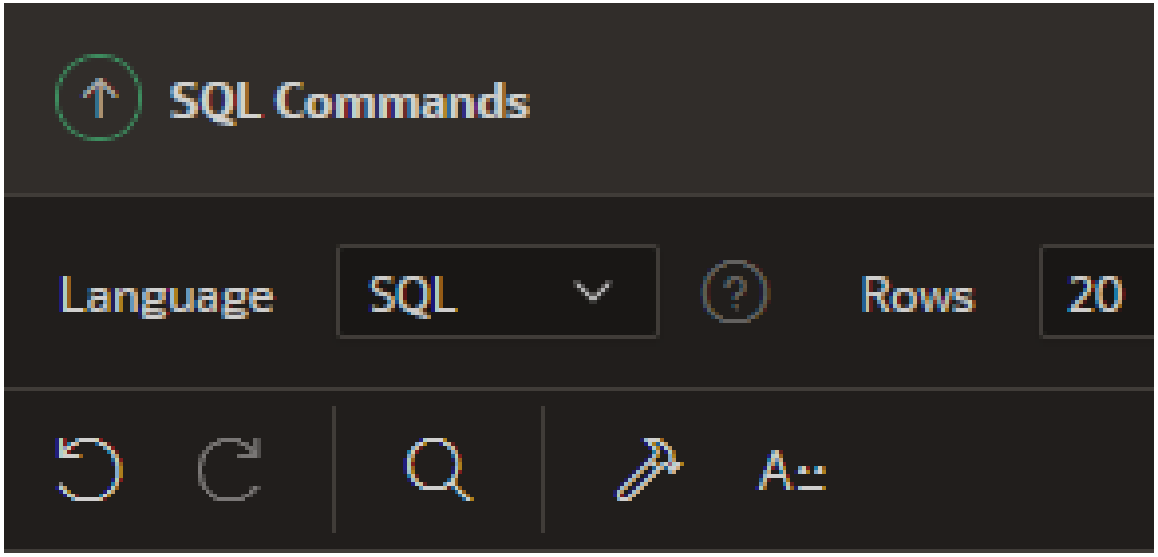
1. Dacă se activează o constrângere, această constrângere se aplică la toate datele din tabela.

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 declaraţii **CREATE TABLE** cât şi **ALTER TABLE.**

Exemplu

ENABLE CONSTRAINT deptno_pk



Constrângeri Cascadate

- Constrângerile de tip **CASCADE** sunt utilizate cu clauza **DROP COLUMN**.
- Constrângerea **CASCADE** *şterge toate constrângerile de integritate ce se referă la cheile primare şi unice definite în coloanele şterse*.
- Şterge de asemenea toate constrângerile multicoloane definite în coloanele şterse.

Vizualizare Constrângeri

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

Exemplu

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

Exerciţii Propuse

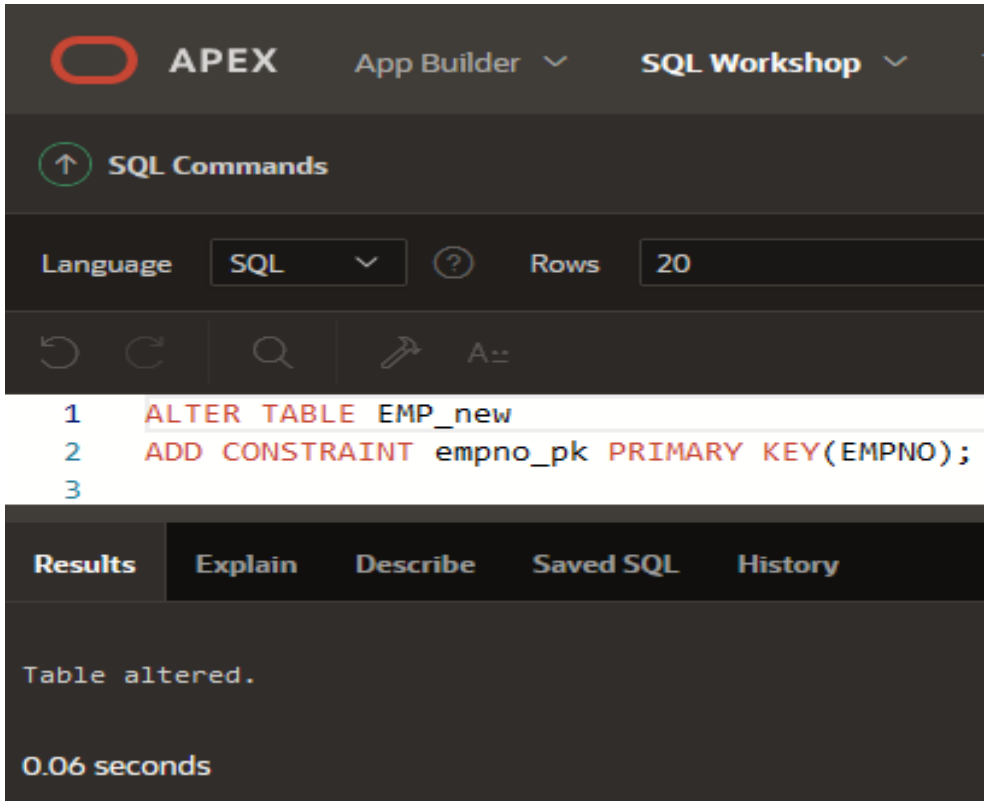
1. Adăugaţi o constrângere **PRIMARY KEY** tabelei **EMP_NEW** pe coloana **EMPNO**. Constrângerea trebuie unică la creare. Numele constrângerii este empno_pk.

2. Ştergeţi constrângerea creată mai sus.

3. Vizualizati constrangerile din tabela **EMP_NEW**.

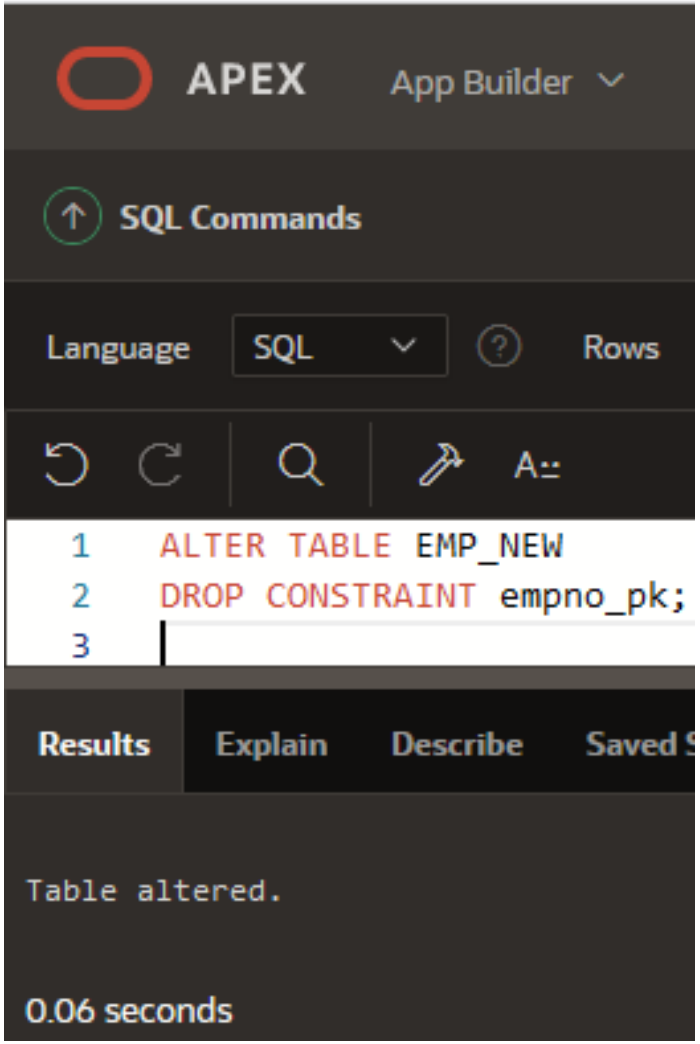
Soluţie 1:

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



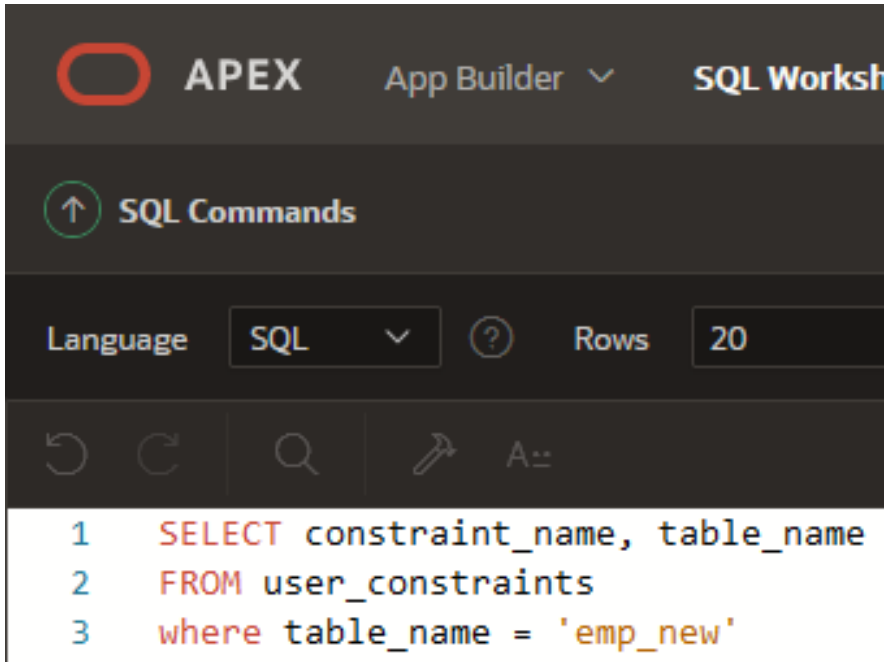
Soluţie 2:

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



Soluţie 3:

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



Întrebări?