**Tipuri LOB (Large Object)**

Voi crea o tabela ‘tabela’ aceasta avand o cheie primara id, si blob\_data, clob\_data, nclob\_data, bfile\_data pentru a putea testa fiecare tip de data.

CREATE TABLE tabela (

id NUMBER,

blob\_data BLOB,

clob\_data CLOB,

nclob\_data NCLOB,

bfile\_data BFILE

);

INSERT INTO tabela (id, blob\_data, clob\_data, nclob\_data, bfile\_data)

VALUES (

1,

EMPTY\_BLOB(), -- BLOB

'This is a CLOB value.',

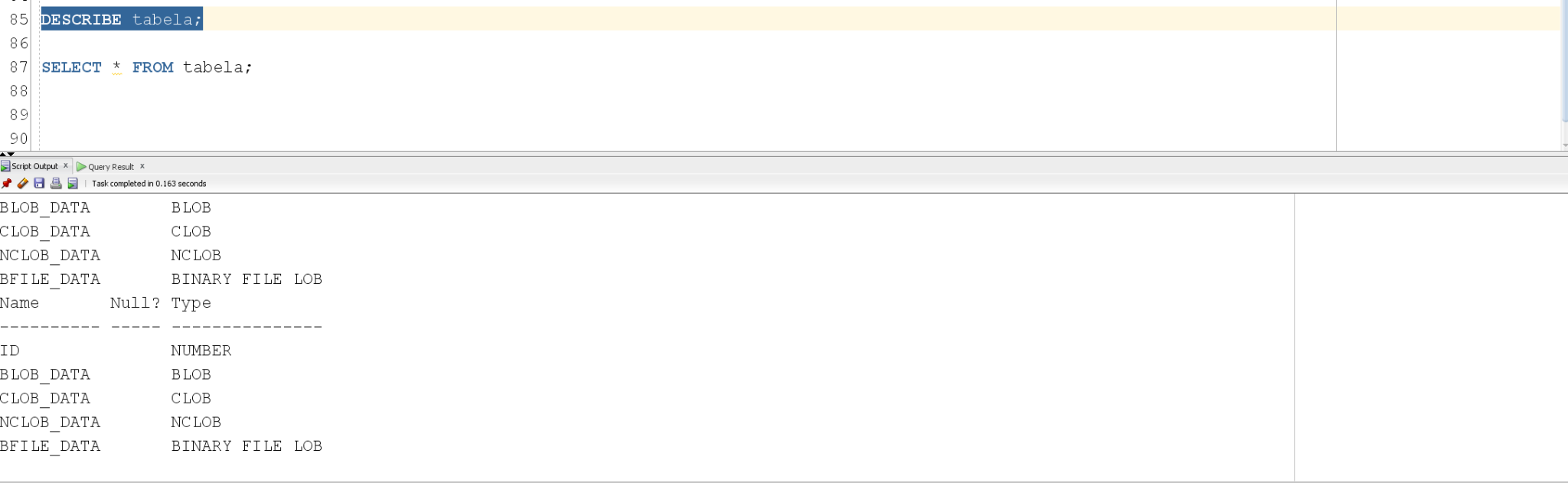
'Și asta este un NCLOB.',

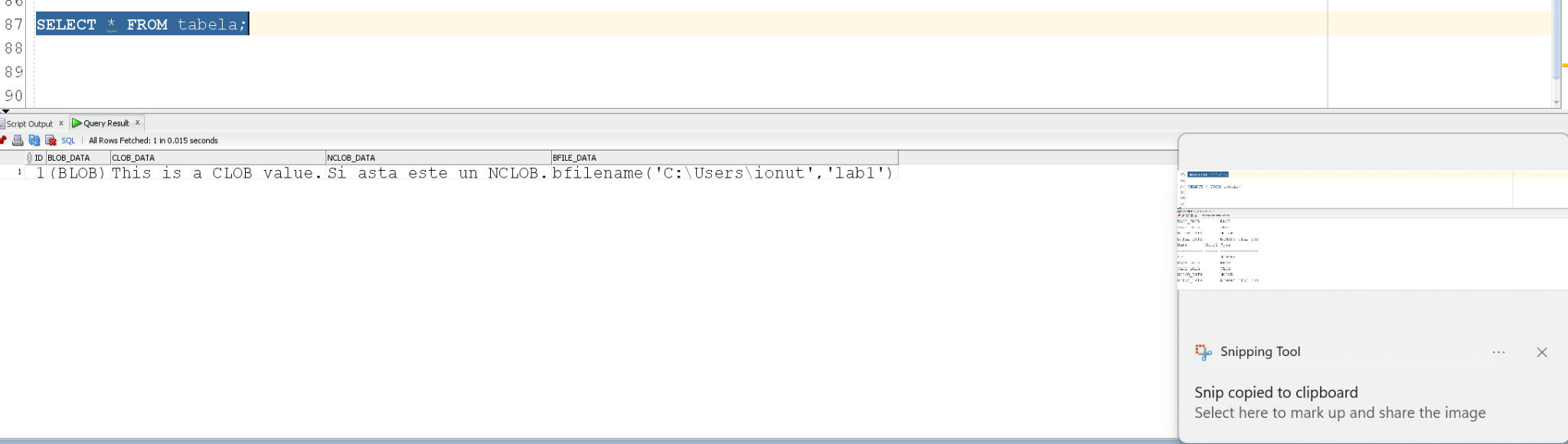
BFILENAME('C:\Users\ionut', 'lab1')

);

DESCRIBE tabela;

SELECT \* FROM tabela;





**BLOB:**

Setati pentru un camp din tabela o imagine jpg:

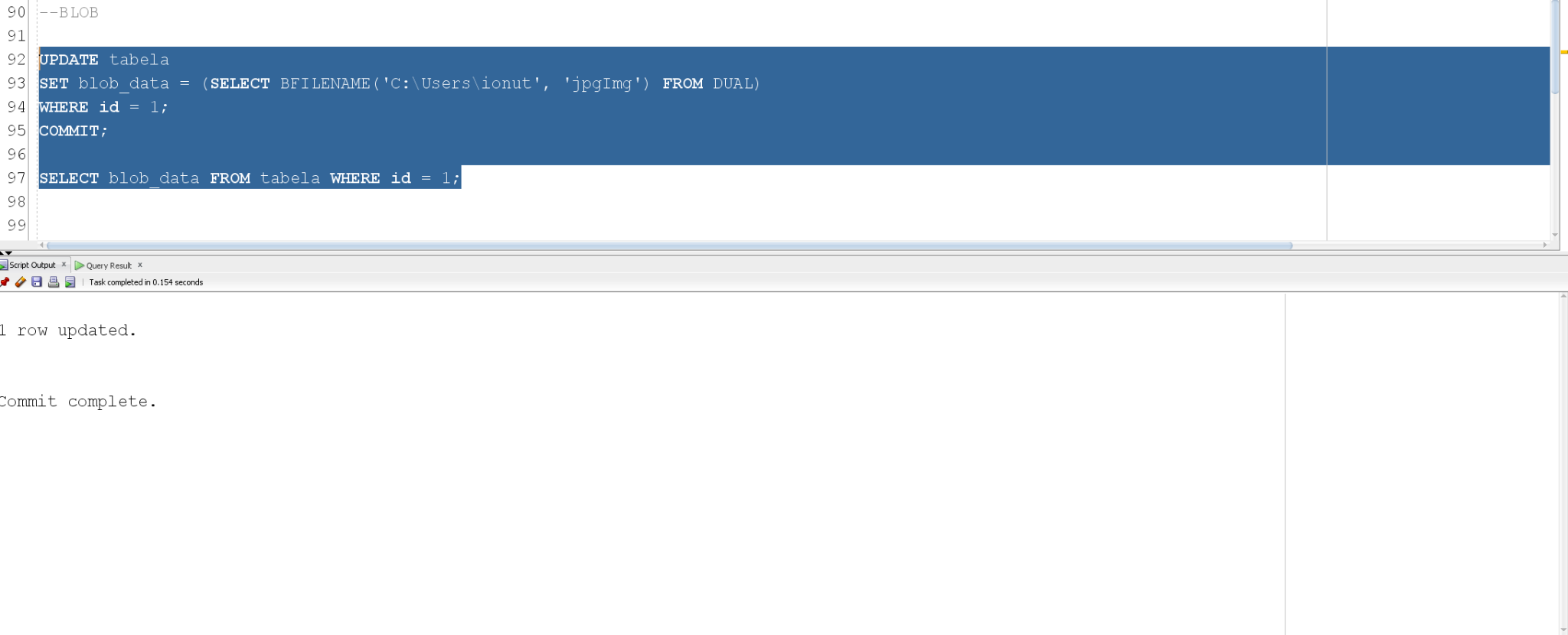
UPDATE tabela

SET blob\_data = (SELECT BFILENAME('C:\Users\ionut', 'jpgImg') FROM DUAL)

WHERE id = 1;

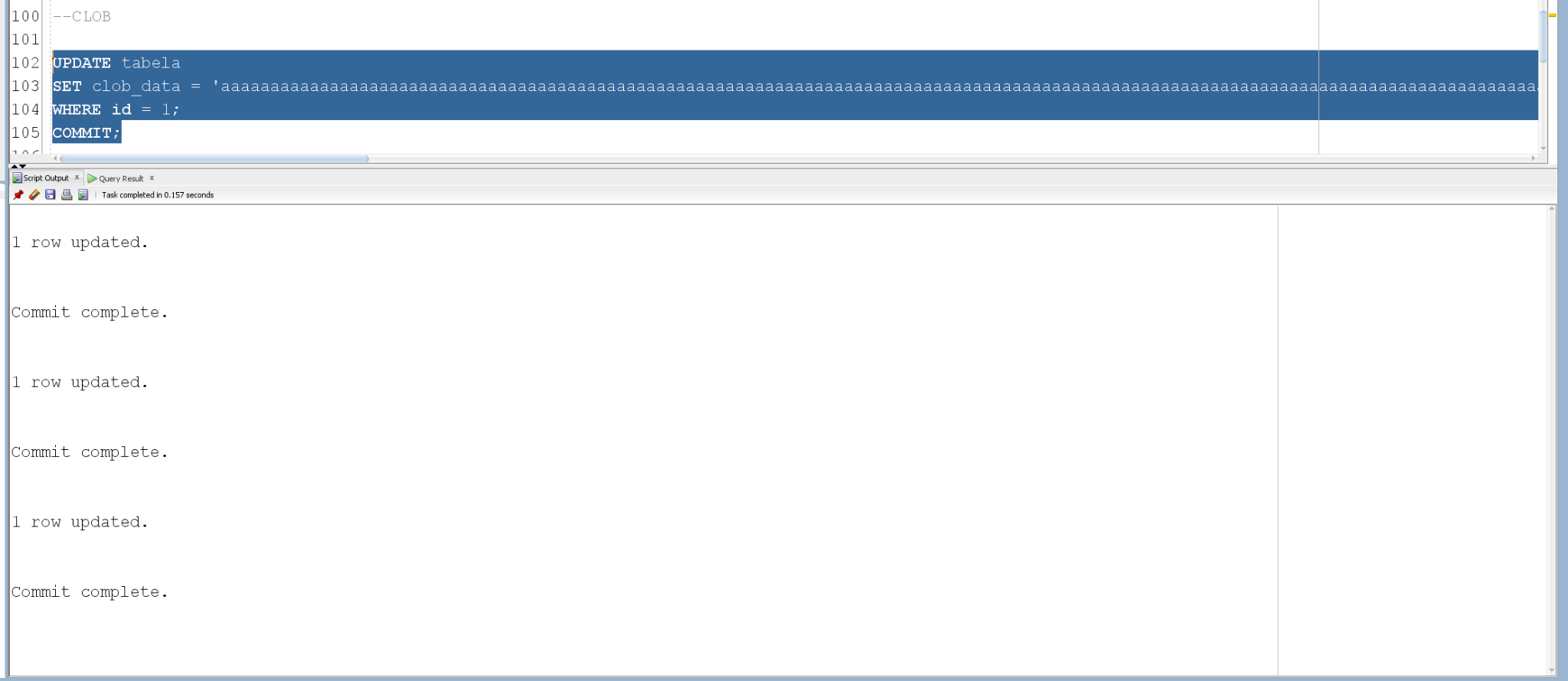
COMMIT;

SELECT blob\_data FROM tabela WHERE id = 1;

****

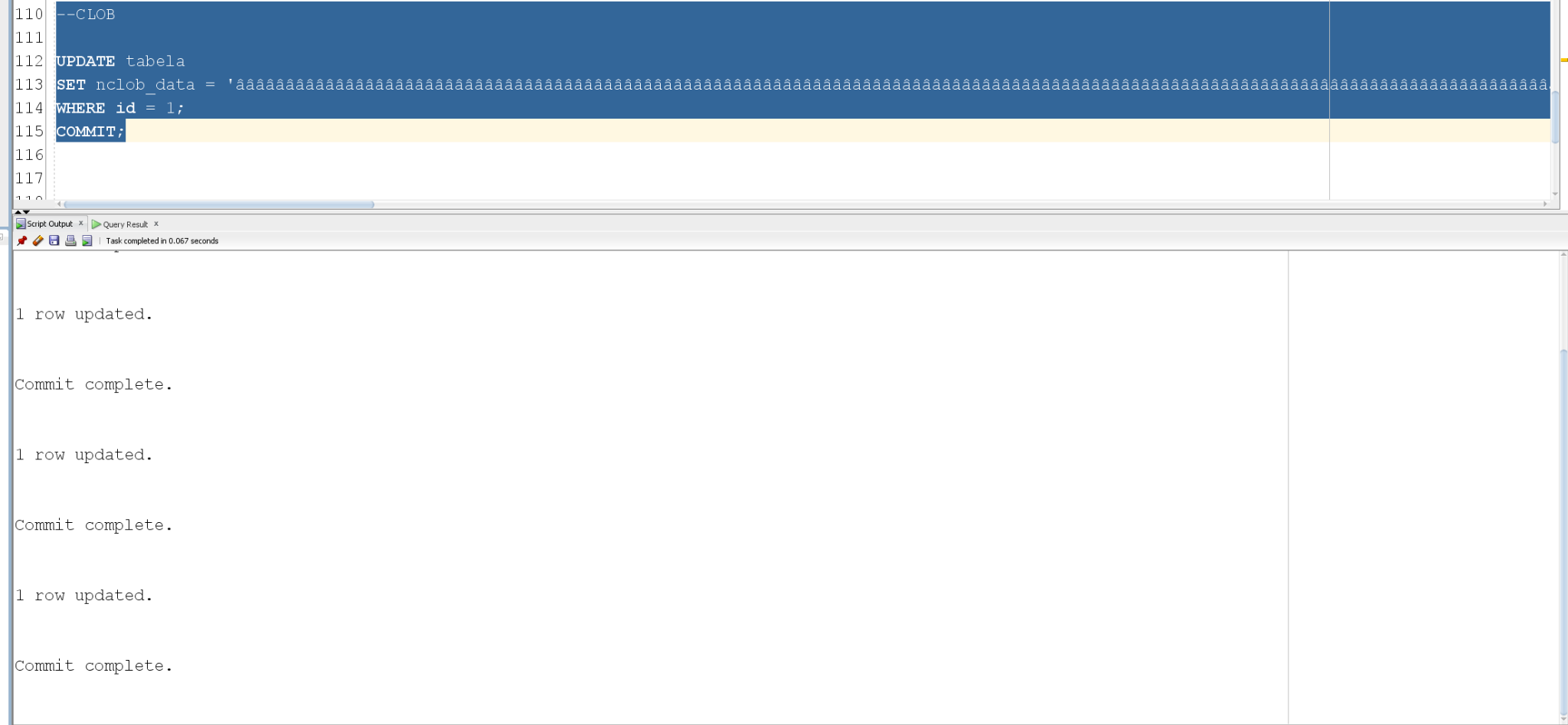
**CLOB:**Scrieti intr-un camp din tabela un text cu 3999 de caractere. (4000 e maxim pentru un query SQL, acest sir l-am luat dintr-un program C++:)  
  
for (int i = 0; i < 3999; i++)

g << “a”;



**NCLOB:**

Aceeasi cerinta ca pentru CLOB, doar ca in loc de a sa avem â.



**BFILE:**

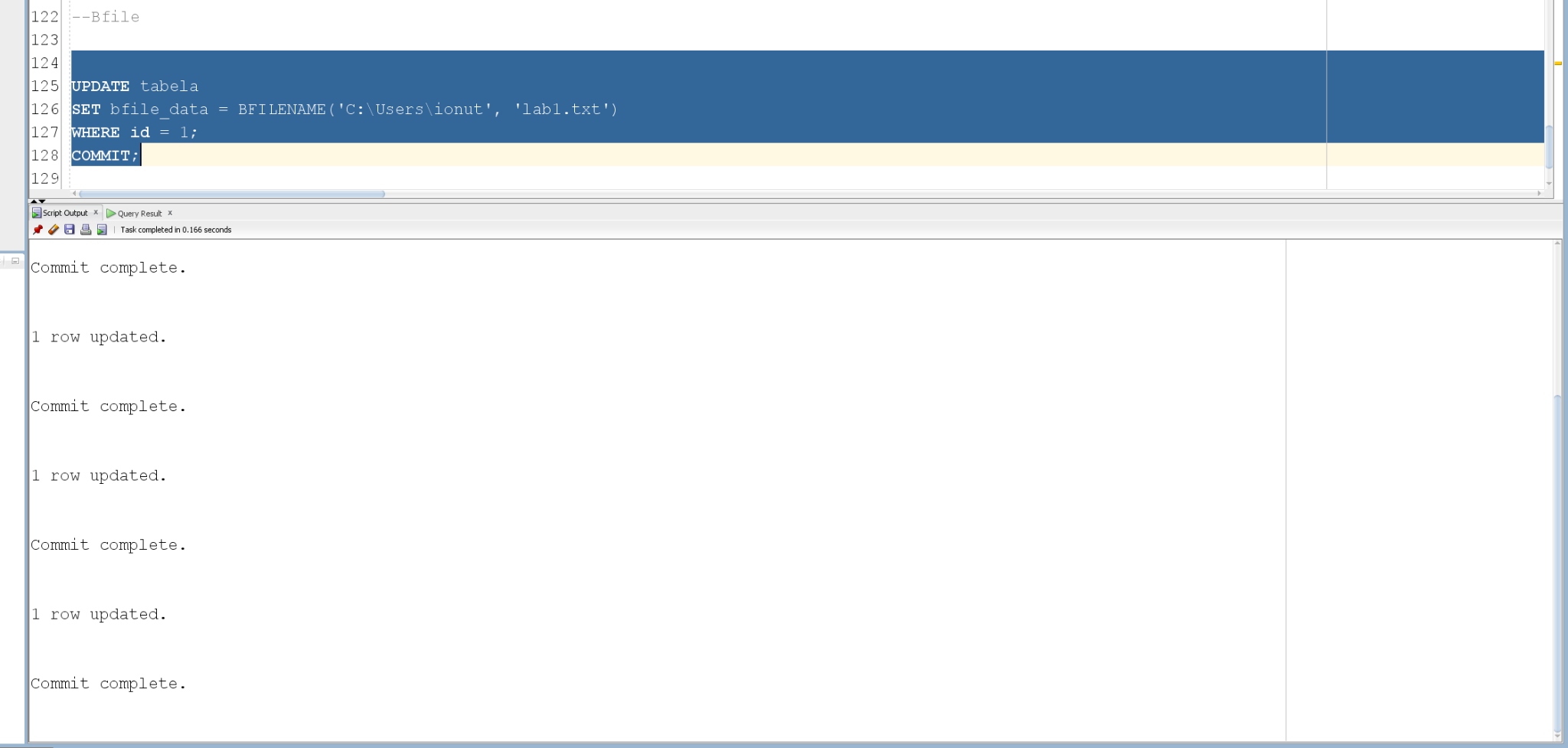
Updatati un camp din tabela, acesta sa aiba continutul unui fisier extern.

UPDATE tabela

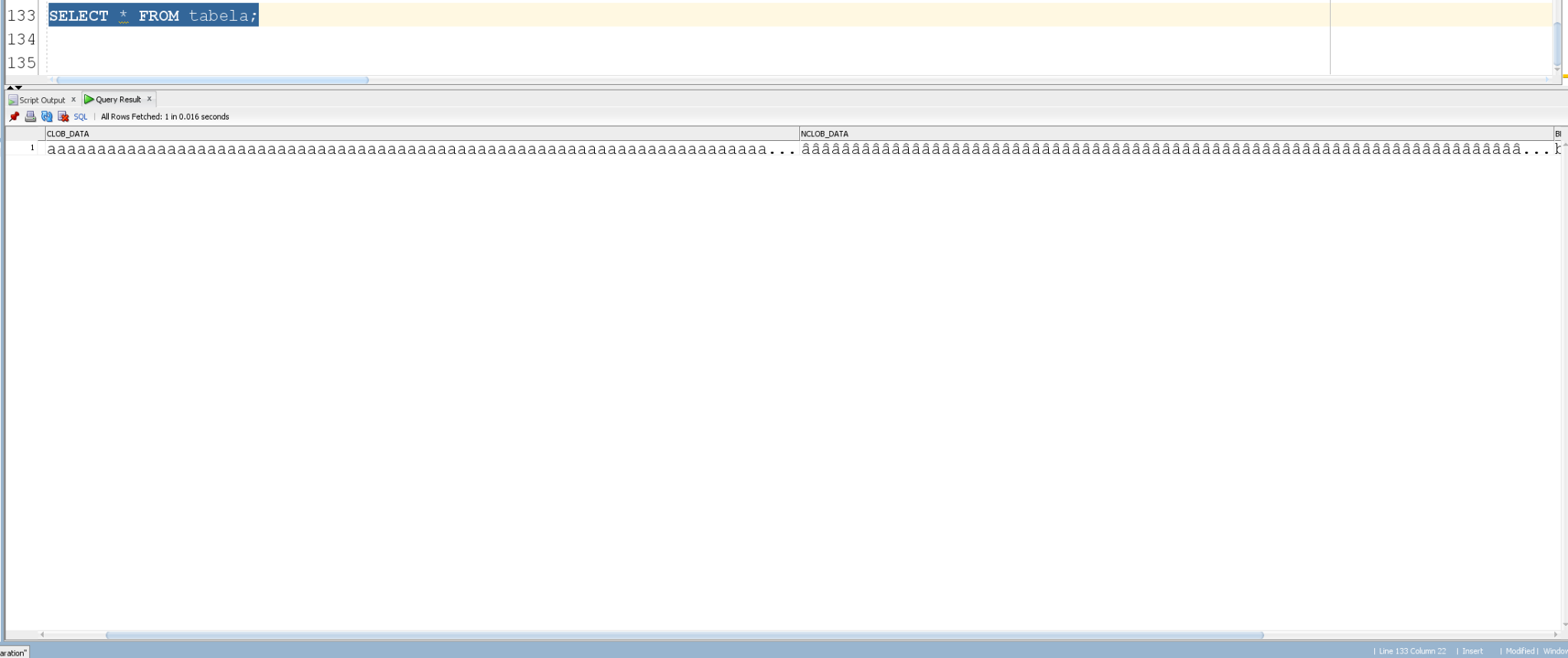
SET bfile\_data = BFILENAME('C:\Users\ionut', 'lab1.txt')

WHERE id = 1;

COMMIT;



SELECT \* FROM tabela;



**CONCLUZIE:**Asadar:

BLOB putem folosi pentru a stoca date imense de binar, de exemplu am vazut pentru un simplu jpg.

CLOB si NCLOB sunt folosite pentru a stoca date imsense de tipuri de caractere, pentru NCLOB si diacritice.

BFILE este folosit pentru a stoca date imense intr-un fisier extern.

Cu aceste tipuri de date se pot obtine query-uri extrem de complexe dar si cu o valoare practica.