****

**Facultatea Calculatoare, Informatică si Microelectronică**

**Departamentul Ingineria Software și Automatică**

**Proiectarea unei baze de date**

**Nedalcord**

**(Denumirea BD)**

**si implementarea ei in SQL Server**

**Proiect de curs la disciplina Baze de date**

**Student: TI-184 Nedelea Tudor-Damian**

**Conducător:**

**Chişinău 2020**

**Cuprins:**

[DESCRIEREA PROBLEMEI: 3](#_Toc22753381)

[1.Proiectarea logica a bazei de date prin proceduri de normalizare: 3](#_Toc22753382)

[2. Definirea constrângerilor de integritate: 4](#_Toc22753383)

[3. Cerințe de administrarea a Bazei de Date 5](#_Toc22753384)

[4. Crearea și modificarea tabelelor 5](#_Toc22753385)

[5. Interogări SQL: 5](#_Toc22753386)

[6. Proceduri si funcții: 6](#_Toc22753387)

[7. Declanșatoare: 6](#_Toc22753388)

[8. Exportarea importarea datelor 6](#_Toc22753389)

9. Servicii de creare a rapoartelor .......................................................................................................................................6

10. Servicii de analiza dimensionala a datelor ....................................................................................................................7

[ANEXE: 7](#_Toc22753390)

# DESCRIEREA PROBLEMEI:

Problema propusă spre soluționare în cadrul proiectului constă în elaborarea unei **Baze de date care ar Automatiza evidența stocurilor de materie primă și materiale la Depozitul** unei companii care are domeniu de activitate producerea produselor de panificație.

Aprovizionarea cu materie prima se face de la furnizori în baza unei comenzi de aprovizionare cu materie primă și materiale, create de către un angajat al companiei de panificație. Furnizorii companiei sunt persoane juridice identificați prin cod unic de înregistrare, nume furnizor, adresa, cont bancar, telefon, e-mail. Furnizorul va întocmi Factura Fiscală pe baza comenzii făcute.

Factura fiscala cuprinde: numărul și data facturii, denumirea furnizorului, numărul contractului, descrierea operațiunii, codul și denumirea materiei prime, unitatea de măsură, cantitatea și prețul unitar al materiei prime facturate, costul total al materiei prime facturate (Anexa1).

Plata facturilor se face cu documentul Ordin de plata specificat prin: numărul și data documentului, numele furnizorului, suma plătită, explicația plății (Anexa 2).

Ieșirile de materii prime de la Depozit către Secțiile de fabricație în vederea obținerii de produse finite (pâine, cozonaci, colaci, s.a) se vor face pe baza Bonurilor de Consum ce cuprind: numărul și data bonului de consum, locul de consum (Secția 1, Secția 2), codul și denumirea materiei prime, unitatea de măsură, prețul unitar, cantitatea eliberată (Anexa 3).

Fiecare materie prima are un termen de valabilitate, pentru materia prima cu termenul expirat se creează Act de casare a materiei prime (Anexa 4).

Baza de date creată trebuie să ofere posibilitatea de înregistrare a materia prima care este procurată de la furnizori și este depozitată în Depozitul companiei, evidența ieșirilor de materie prima și oferirea în orișice moment de timp a informației despre stocul final al unei materii prime, evidenta plaților pentru materia prima procurată.

# 1.Proiectarea logica a bazei de date prin proceduri de normalizare:

* **Găsirea și organizarea informațiilor necesare**

Colectați toate tipurile de informații pe care doriți să le înregistrați în baza de date, cum ar fi: numele materie prima, nume furnizor, s.a

* **Împărțirea informațiilor în tabele**

Împărțiți elementele informaționale în entități sau subiecte majore, cum ar fi FURNIZORI sau MATERII\_PRIME. Fiecare subiect devine apoi un tabel.

* **Transformarea elementelor informaționale în coloane**

Decideți ce informații doriți să stocați în fiecare tabel. Fiecare element devine un câmp și este afișat sub formă de coloană în tabel. De exemplu, un tabel Angajați poate include câmpuri cum ar fi Numele de familie și Data de angajare.

* **Specificarea cheilor primare**

Alegeți cheia primară a fiecărui tabel. Cheia primară este o coloană care se utilizează pentru a identifica în mod unic fiecare rând. Un exemplu ar putea fi ID produs sau ID comandă.

* **Configurarea relațiilor în tabel**

Uitați-vă la fiecare tabel și decideți cum sunt corelate datele dintr-un tabel cu datele din alte tabele. Adăugați câmpuri la tabele sau creați tabele noi pentru a clarifica relațiile, după cum este necesar.

* **Rafinarea proiectării**

Analizați proiectarea pentru a detecta eventuale erori. Creați tabelele și adăugați câteva înregistrări de date eșantion. Vedeți dacă puteți obține rezultatele dorite din tabele. Faceți ajustări la proiectare, după cum este necesar.

* **Aplicarea regulilor de normalizare**

Aplicați reguli de normalizare a datelor pentru a vedea dacă tabelele sunt structurate corect. Faceți ajustări la tabele, după cum este necesar.

# 2. Definirea constrângerilor de integritate:

***Exemple in baza problemei descrise:***

***CI1****: Stocul final al unui material se calculează conform formulei:*

***Stocul final =Stocul inițial(Cantitatea inițială) +Cantitatea Intrată-Cantitatea Ieșita***

***CI2****:* ***Stocul final la sfârșitul perioadei precedente=******Stocul inițial la începutul perioadei următoare***

***CI3****: Costul unei materii prime se determina prin înmulțirea Cantității la Preț*

***Suma (Costul) = Cantitatea \*Prețul materii prime ;***

***CI4****: Unitatea de măsură pentru materiale poate fi una din următoarele șiruri de caractere : “t”, “kg”, “buc”, “l”, “gr.” ;*

***CI5****: Stocul de materii prime trebuie sa fie >=0; (control)*

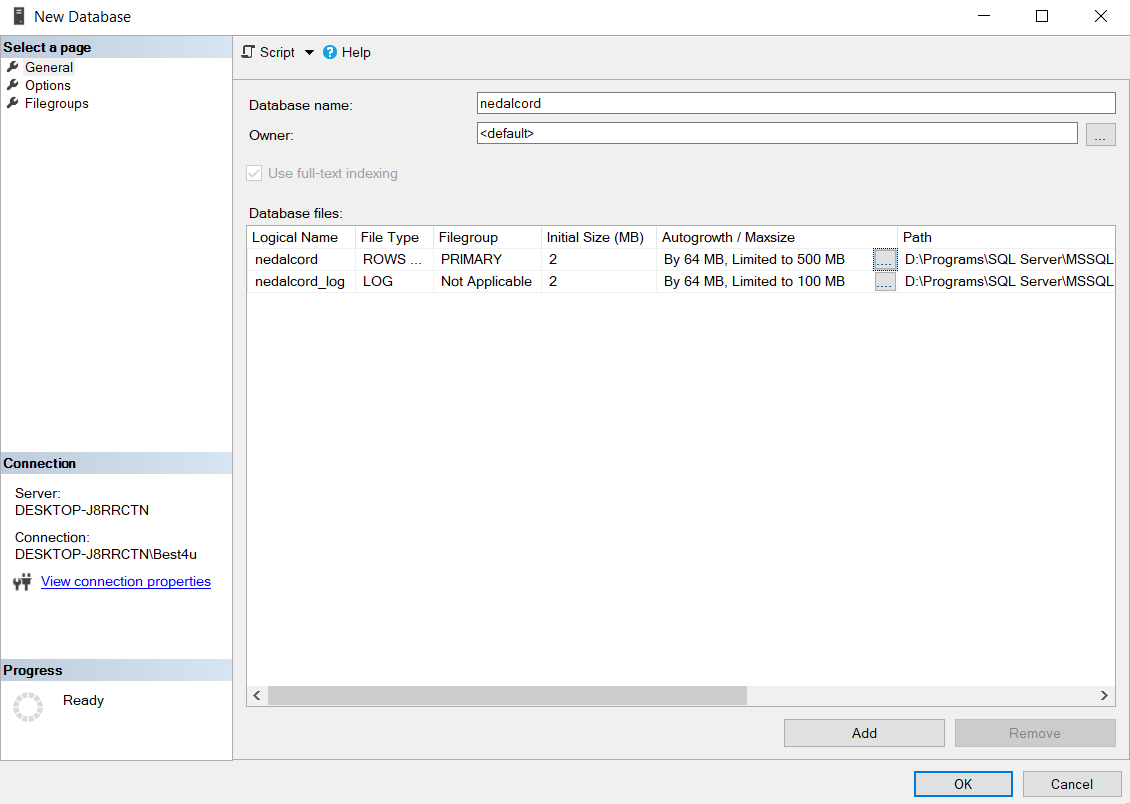
***CI6****: Prețul materii prime trebuie sa fie >0;*

***CI7****: Materia prima cu termenul expirat nu poate fi transmis în producere; (control)*

***CI8****: Cu fiecare furnizor se încheie contracte anual, în ultima zi a anului trebuie sa apară înștiințare despre expirarea termenului de valabilitate a contractului*

# 3. Cerințe de administrarea a Bazei de Date

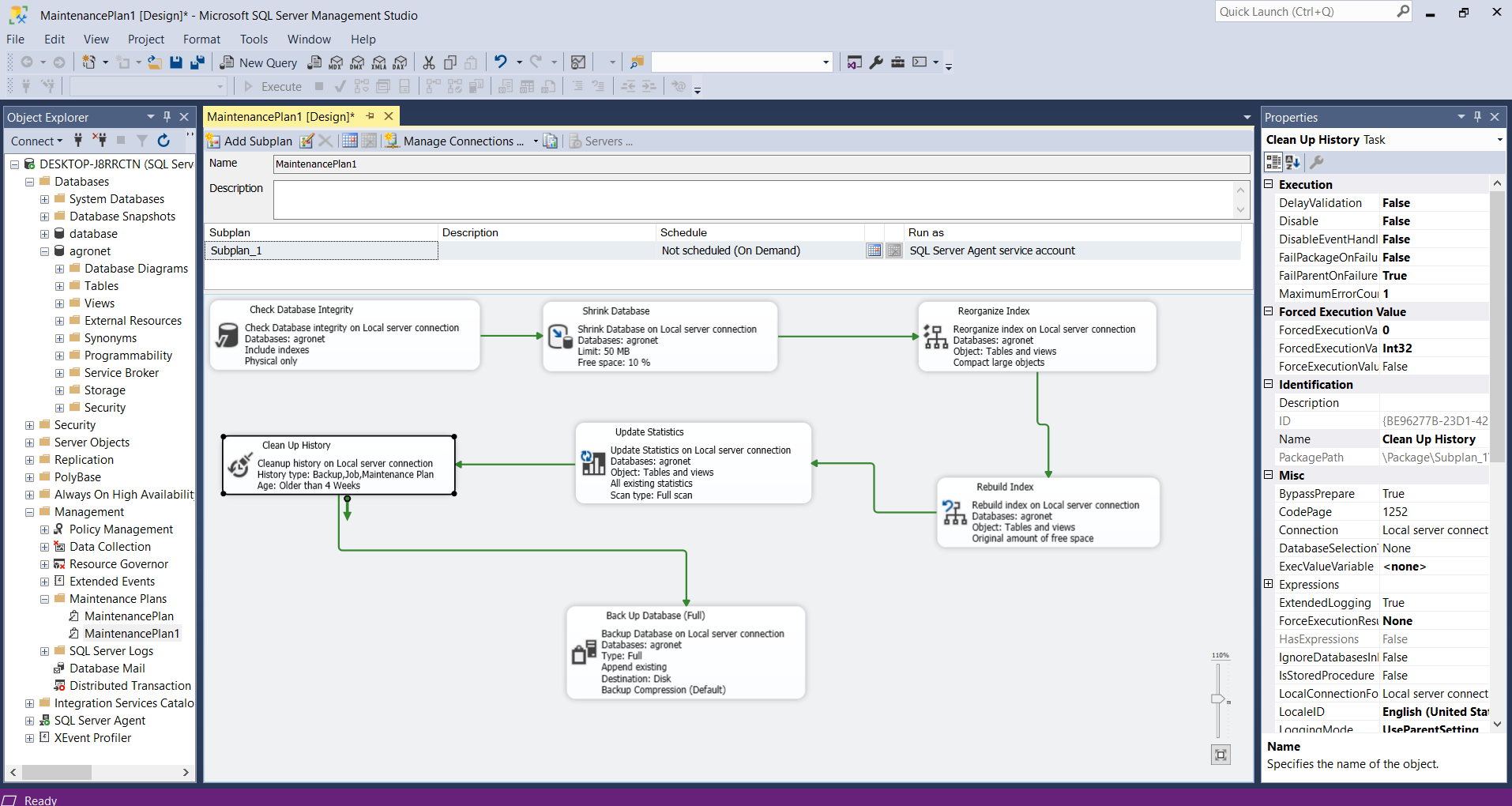
Exemplu:



Baza de date va avea o creștere a fișierului primar a bazei de date de 2MB cu limita de creștere de 500 MB și a logului de 2 MB cu limita de creștere de 100MB. Locul amplasare a fesierilor BD să este pe discul D.

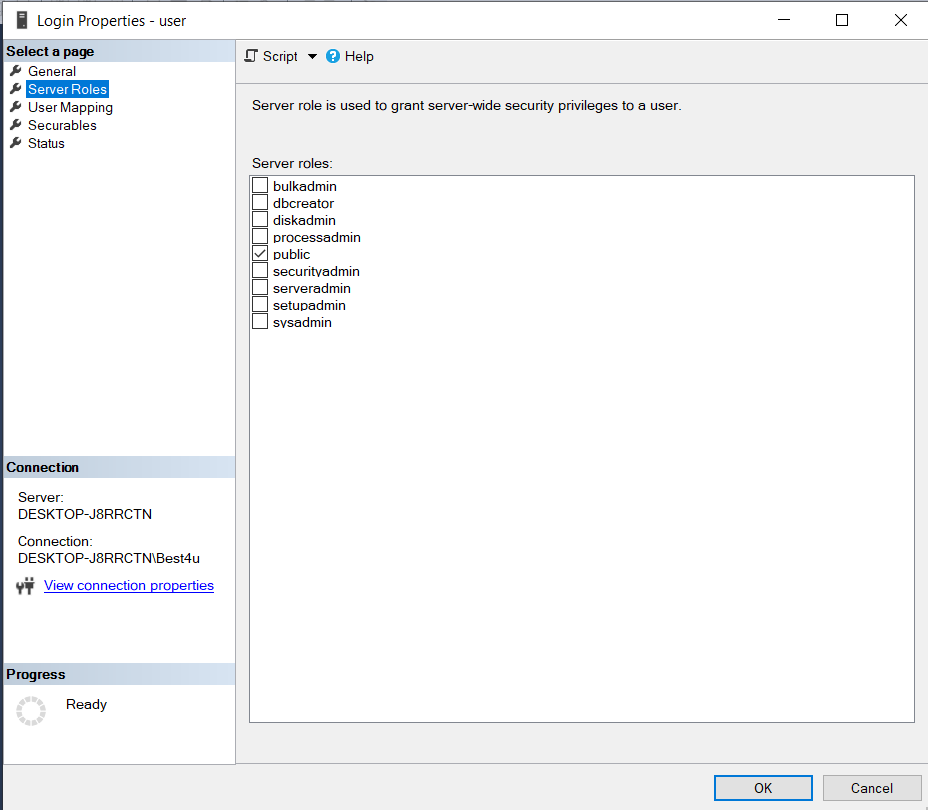
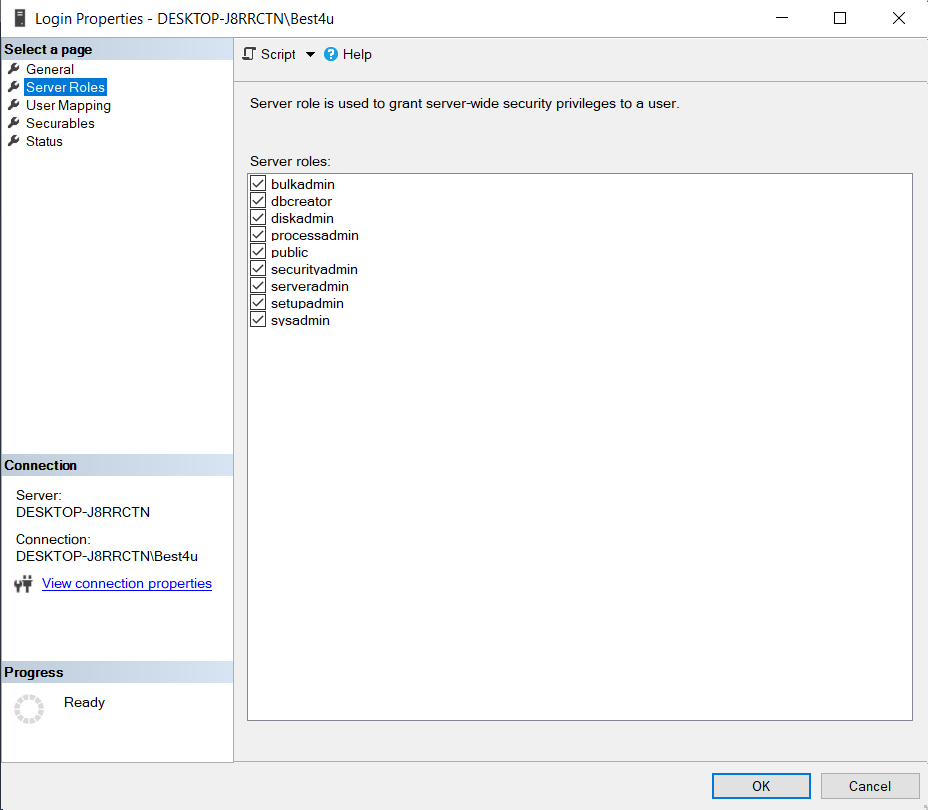
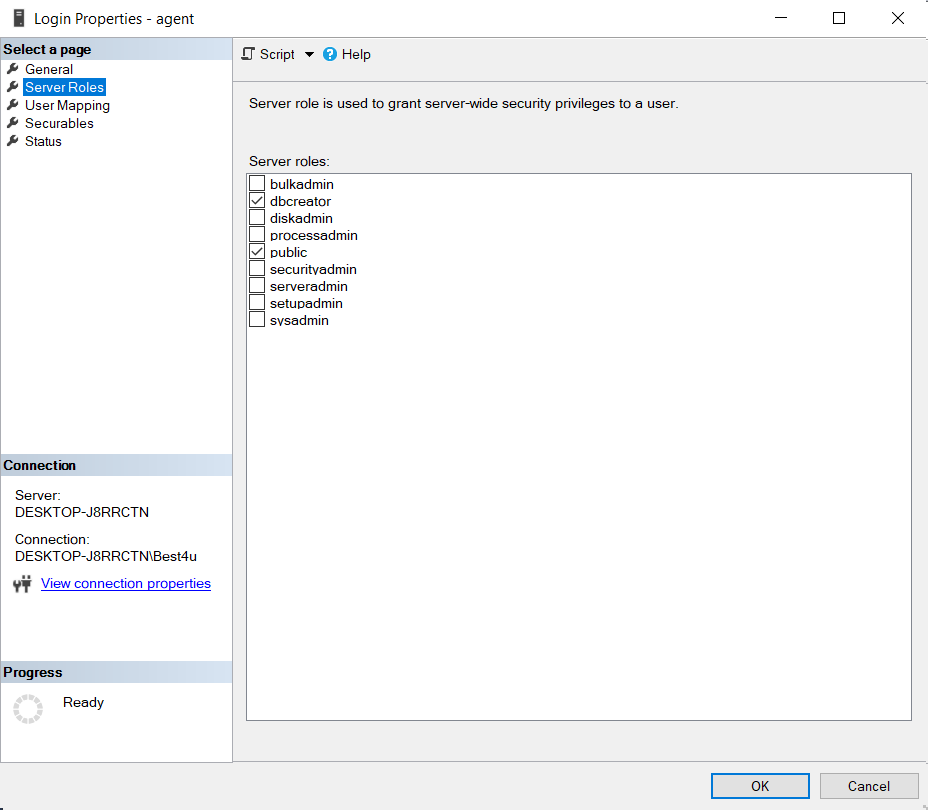
Au fost create doua planuri de întreținere al bazei de date care realizeaza :

* Reconstruirea indecșilor numai asupra tabelelor de baza din schemele care exista în baza de date,
* Spațiul liber pe pagina trebuie sa fie 10%. După reconstruire, trebuie sa urmeze colectarea statisticilor complete despre indecșii reconstruiți.
* Crearea backup complet și diferențiat al bazei de date
* Ștergerea istoriei despre operațiile de **Backup-Restore** ce au avut loc pe SQL Server.
* Trebuie șters istoricul care este mai vechi de 6 săptămâni.
* Acest plan trebuie sa fie executat în fiecare prima duminica a lunii. Raportul de executare a planului trebuie să fie adăugat în dosarul de pe discul D unde sunt salvate și fișierele BD. Inițializați executarea planului.



PrtSC – imagine cu planul de intretinere

La schema bazei de date vor fi definiti 2-3 utilizatori , si anume agent, best4u și user. Â



PrtSC – imagine cu utilizatorii si drepturile de acces

# 4. Crearea și modificarea tabelelor

1.De creat structura baza de date tabelele, constrângerile de integritate, utilizând limbajul -SQL.

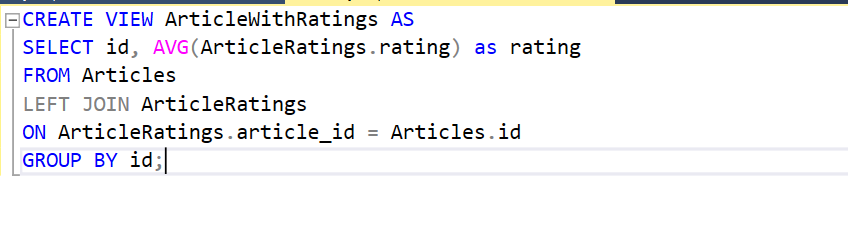
create table Materials  
(  
 id int IDENTITY  
 constraint "PK\_0343d0d577f3effc2054cbaca7f"  
 primary key,  
 created\_at datetime default GETDATE() not null,  
 updated\_at datetime default GETDATE() not null  
)  
  
create table MaterialTranslations  
(  
 created\_at datetime default GETDATE() not null,  
 updated\_at datetime default GETDATE() not null,  
 language varchar(4000) CHECK(language in ('ro-RO', 'ru-RU')) not null,  
 name varchar(4000) not null  
 constraint "UQ\_49eb82b7eb1196daab527cacbef"  
 unique,  
 description varchar(4000),  
 material\_id int not null  
 constraint "FK\_1b92e071b0a73cfd6c7effd307a"  
 references Materials  
 on delete cascade,  
 constraint "PK\_0a5040c5d70721235f37476c34f"  
 primary key (language, material\_id)  
);  
  
create table Products  
(  
 id int IDENTITY not null  
 constraint "PK\_bebc9158e480b949565b4dc7a82"  
 primary key,  
 created\_at datetime default GETDATE() not null,  
 updated\_at datetime default GETDATE() not null,  
 length int,  
 width int,  
 height int,  
 weight int,  
 slug varchar(4000) not null  
 constraint "UQ\_8cfaf4a1e80806d58e3dbe69224"  
 unique  
);  
  
create table ProductTranslations  
(  
 created\_at datetime default GETDATE() not null,  
 updated\_at datetime default GETDATE() not null,  
 language varchar(4000) CHECK(language in ('ro-RO', 'ru-RU')) not null,  
 name varchar(4000) not null  
 constraint "UQ\_8f066b3fc13ecebe2ef6c7e423d"  
 unique,  
 description varchar(4000) not null,  
 product\_id int not null  
 constraint "FK\_045befe4da0d3c207a981f4e88b"  
 references Products  
 on delete cascade,  
 constraint "PK\_c16e540c11dbfdbccd8825bf688"  
 primary key (language, product\_id)  
);  
  
create table ProductMainImages  
(  
 id int IDENTITY not null  
 constraint "PK\_4fc42eba35343ea2e2068318698"  
 primary key,  
 created\_at datetime default GETDATE() not null,  
 updated\_at datetime default GETDATE() not null,  
 original\_name varchar(4000) not null,  
 path varchar(4000) not null,  
 mimetype varchar(4000) not null,  
 description varchar(4000),  
 product\_id int  
 constraint "REL\_d81a9d6a5de6aa969014ee18a9"  
 unique  
 constraint "FK\_d81a9d6a5de6aa969014ee18a96"  
 references Products  
);  
  
create table [Files]  
(  
 id int IDENTITY not null  
 constraint "PK\_36b46d232307066b3a2c9ea3a1d"  
 primary key,  
 created\_at datetime default GETDATE() not null,  
 updated\_at datetime default GETDATE() not null,  
 original\_name varchar(4000) not null,  
 path varchar(4000) not null,  
 mimetype varchar(4000) not null,  
 description varchar(4000),  
 product\_id int  
 constraint "FK\_e4c65a52e0203d2daee81936bcc"  
 references Products  
);  
  
create table [Users]  
(  
 id int IDENTITY not null  
 constraint "PK\_cace4a159ff9f2512dd42373760"  
 primary key,  
 created\_at datetime default GETDATE() not null,  
 updated\_at datetime default GETDATE() not null,  
 first\_name varchar(4000) not null,  
 last\_name varchar(4000) not null,  
 email varchar(4000) not null  
 constraint "UQ\_e12875dfb3b1d92d7d7c5377e22"  
 unique,  
 reset\_key varchar(4000),  
 reset\_code varchar(4000),  
 reset\_date datetime,  
 phone varchar(4000) not null,  
 role varchar(4000) CHECK(role in ('admin', 'user')) not null,  
 password varchar(4000) not null  
);  
  
create table Tags  
(  
 id int IDENTITY not null  
 constraint "PK\_8e4052373c579afc1471f526760"  
 primary key,  
 created\_at datetime default GETDATE() not null,  
 updated\_at datetime default GETDATE() not null,  
 name varchar(4000) not null  
 constraint "UQ\_6a9775008add570dc3e5a0bab7b"  
 unique  
);  
  
create table Articles  
(  
 id int IDENTITY not null  
 constraint "PK\_40808690eb7b915046558c0f81b"  
 primary key,  
 created\_at datetime default GETDATE() not null,  
 updated\_at datetime default GETDATE() not null,  
 image\_id int  
 constraint "REL\_9907c8dd69b933adbc0e5e9f37"  
 unique  
 constraint "FK\_9907c8dd69b933adbc0e5e9f37e"  
 references Files  
 on delete cascade,  
 user\_id int not null  
 constraint "FK\_fae0bad5f06a58f3d2b68e37f11"  
 references Users,  
 slug varchar(4000) not null  
 constraint "UQ\_0ab85f4be07b22d79906671d72f"  
 unique  
);  
  
create table ArticleTranslations  
(  
 created\_at datetime default GETDATE() not null,  
 updated\_at datetime default GETDATE() not null,  
 language varchar(4000) CHECK(language in ('ro-RO', 'ru-RU')) not null,  
 title varchar(4000) not null  
 constraint "UQ\_f3f5a6ff21e37712754cbc9ec40"  
 unique,  
 content varchar(4000) not null,  
 article\_id int not null  
 constraint "FK\_cda6a53008fc816fdbe004a5c3f"  
 references Articles  
 on delete cascade,  
 constraint "PK\_bfce0605ca87e0b0c597f2174fb"  
 primary key (language, article\_id)  
);  
  
  
create table ArticleRatings  
(  
 created\_at datetime default GETDATE() not null,  
 rating smallint not null,  
 comment varchar(4000),  
 "lastName" varchar(4000),  
 "firstName" varchar(4000),  
 email varchar(4000) not null,  
 article\_id int not null  
 constraint "FK\_931dcc2609ae0ed67947a6f0fcb"  
 references Articles  
 on delete cascade,  
 constraint "PK\_aa6dda6ef564039963f42bf8e49"  
 primary key (email, article\_id)  
);  
  
create table Constants  
(  
 id int IDENTITY not null  
 constraint "PK\_1113c850855b6cf6cc507b64f6e"  
 primary key,  
 created\_at datetime default GETDATE() not null,  
 updated\_at datetime default GETDATE() not null,  
 name varchar(4000) CHECK(name in ('HAPPY\_CUSTOMERS', 'PRODUCT\_TYPES', 'YEARS\_EXPERIENCE', 'REGIONS')) not null  
 constraint "UQ\_9e0693169d436cc70ad6610922f"  
 unique,  
 value int not null  
);  
  
create table ProductMaterials  
(  
 product\_id int not null  
 constraint "FK\_0945c9cf0547c6ce88ff4b73301"  
 references Products  
 on delete cascade,  
 material\_id int not null  
 constraint "FK\_74e3161a455fc1fb9b612a0182c"  
 references Materials  
 on delete cascade,  
 constraint "PK\_0acb2243bba1f0a7b9de1469ac5"  
 primary key (product\_id, material\_id)  
);  
  
create index "IDX\_0945c9cf0547c6ce88ff4b7330"  
 on ProductMaterials (product\_id);  
  
create index "IDX\_74e3161a455fc1fb9b612a0182"  
 on ProductMaterials (material\_id);  
  
create table ArticleTags  
(  
 article\_id int not null  
 constraint "FK\_26455b396109a0b535ddb614832"  
 references Articles  
 on delete cascade,  
 tag\_id int not null  
 constraint "FK\_cdc3f155737b763c298ab080f84"  
 references Tags  
 on delete cascade,  
 constraint "PK\_82061ea91c181fc708715430361"  
 primary key (article\_id, tag\_id)  
);  
  
create index "IDX\_26455b396109a0b535ddb61483"  
 on ArticleTags (article\_id);  
  
create index "IDX\_cdc3f155737b763c298ab080f8"  
 on ArticleTags (tag\_id);  
  
create table Questions  
(  
 id int IDENTITY not null  
 constraint "PK\_21e5786aa0ea704ae185a79b2d5"  
 primary key,  
 created\_at datetime default GETDATE() not null,  
 updated\_at datetime default GETDATE() not null,  
 email varchar(4000),  
 phone varchar(4000) not null,  
 first\_name varchar(4000) not null,  
 last\_name varchar(4000) not null,  
 content varchar(4000) not null,  
 [read] bit default 0 not null,  
 handled bit default 0 not null  
);

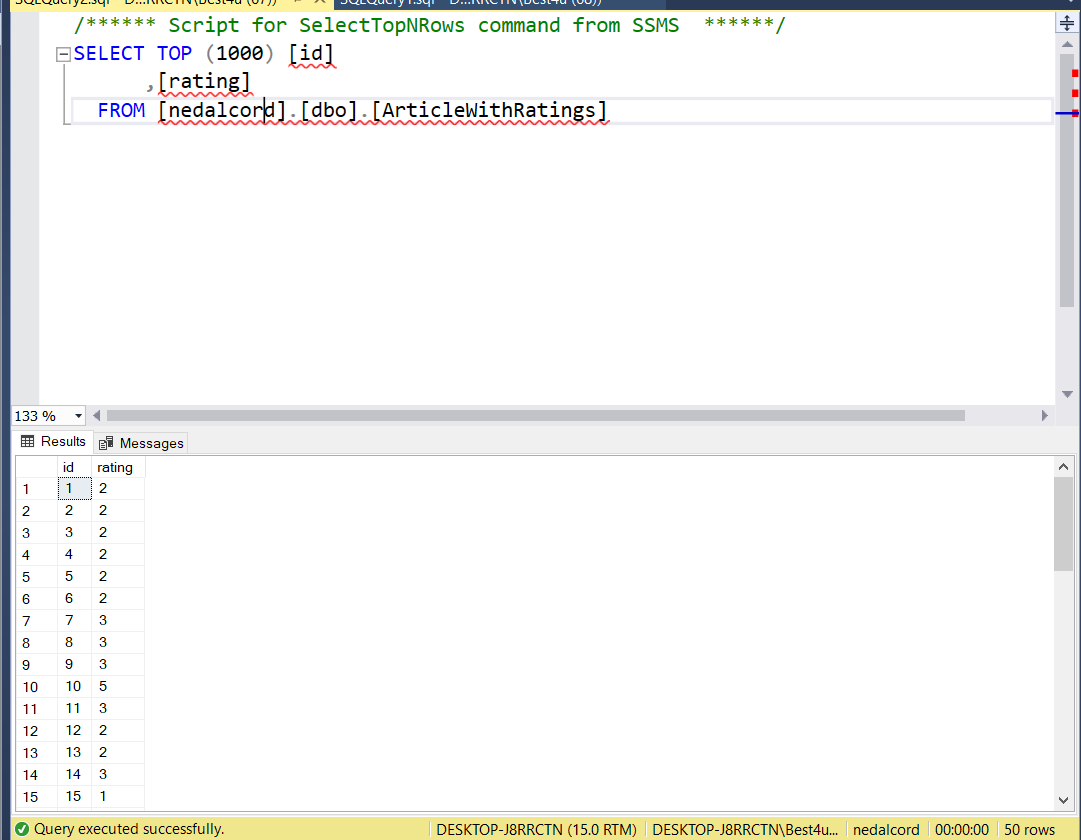
2.De reprezentat grafic structura bazei de date.



3.De definit pentru obiectele create nume alternative (SINONIME) utilizând limbajul -SQL.

4.De creat un tabel virtual care ar permite adăugarea , modificarea informației in baza de date;

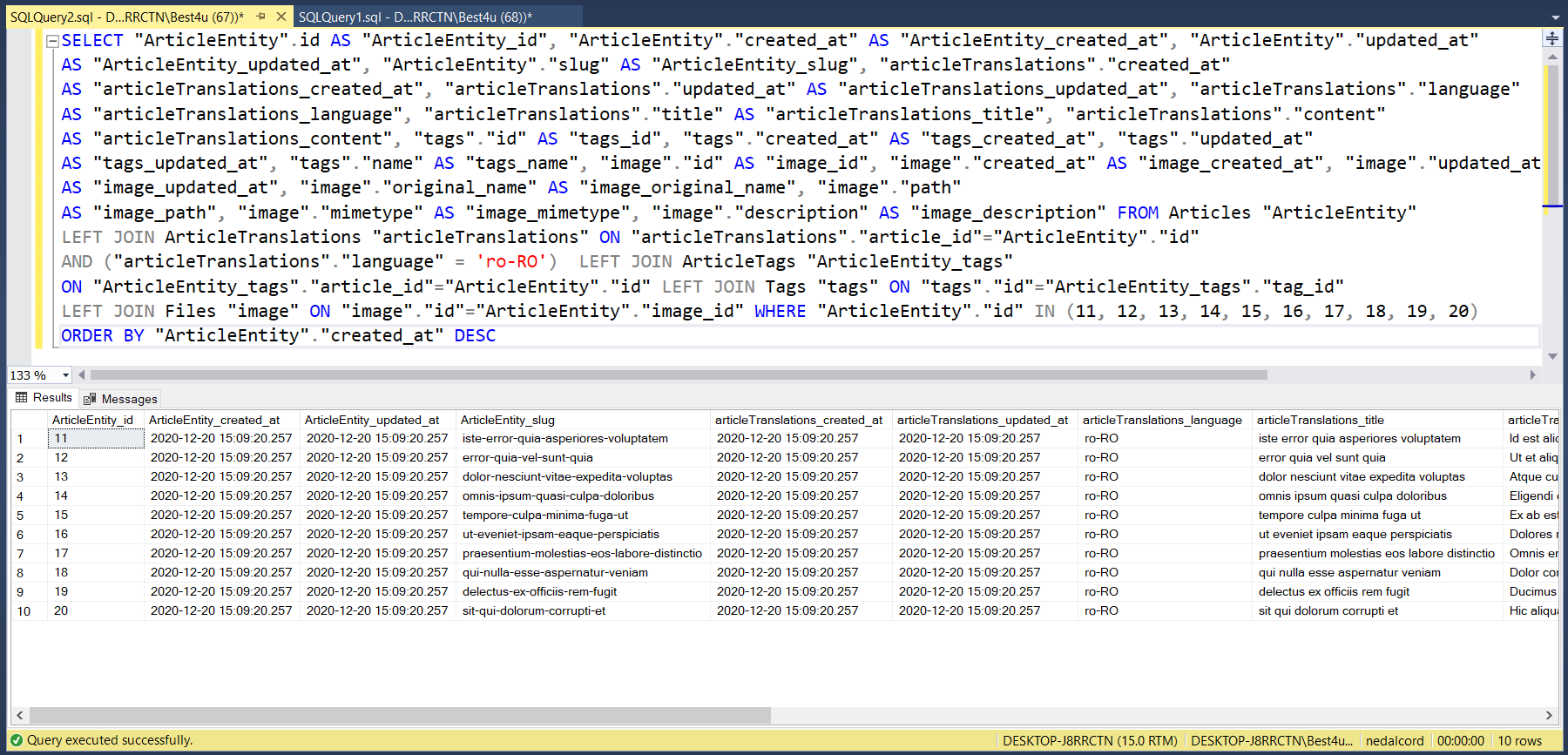




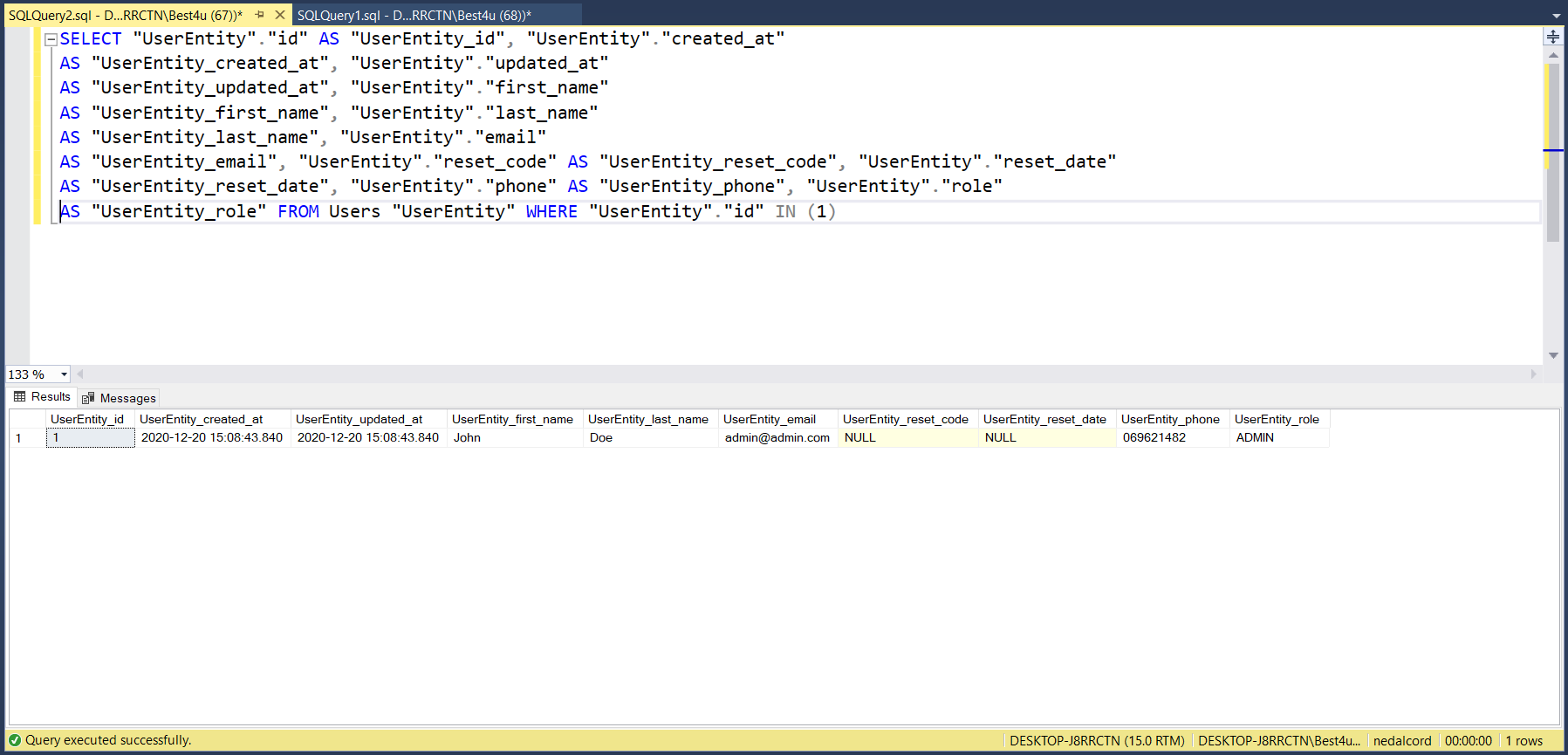
Tabel virtual cu ratingul mediu per articol.

# 5. Interogări SQL:

* 1. ***De creat Interogări simple (2):***

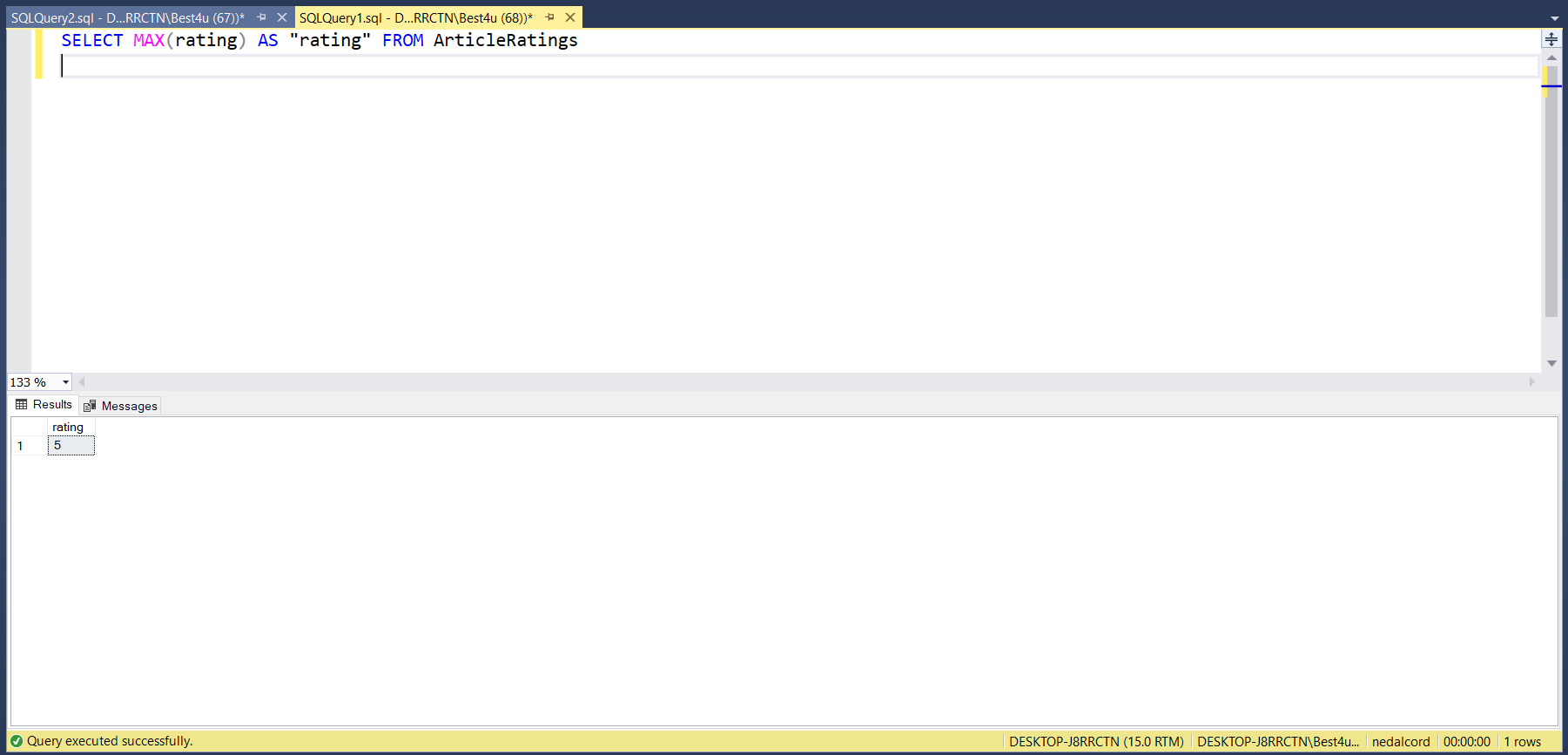
******

**Interogarea de extragere a articolelor cu id-urile 11, 12, 13, 14, 15, 16, 17, 18, 19, 20.**

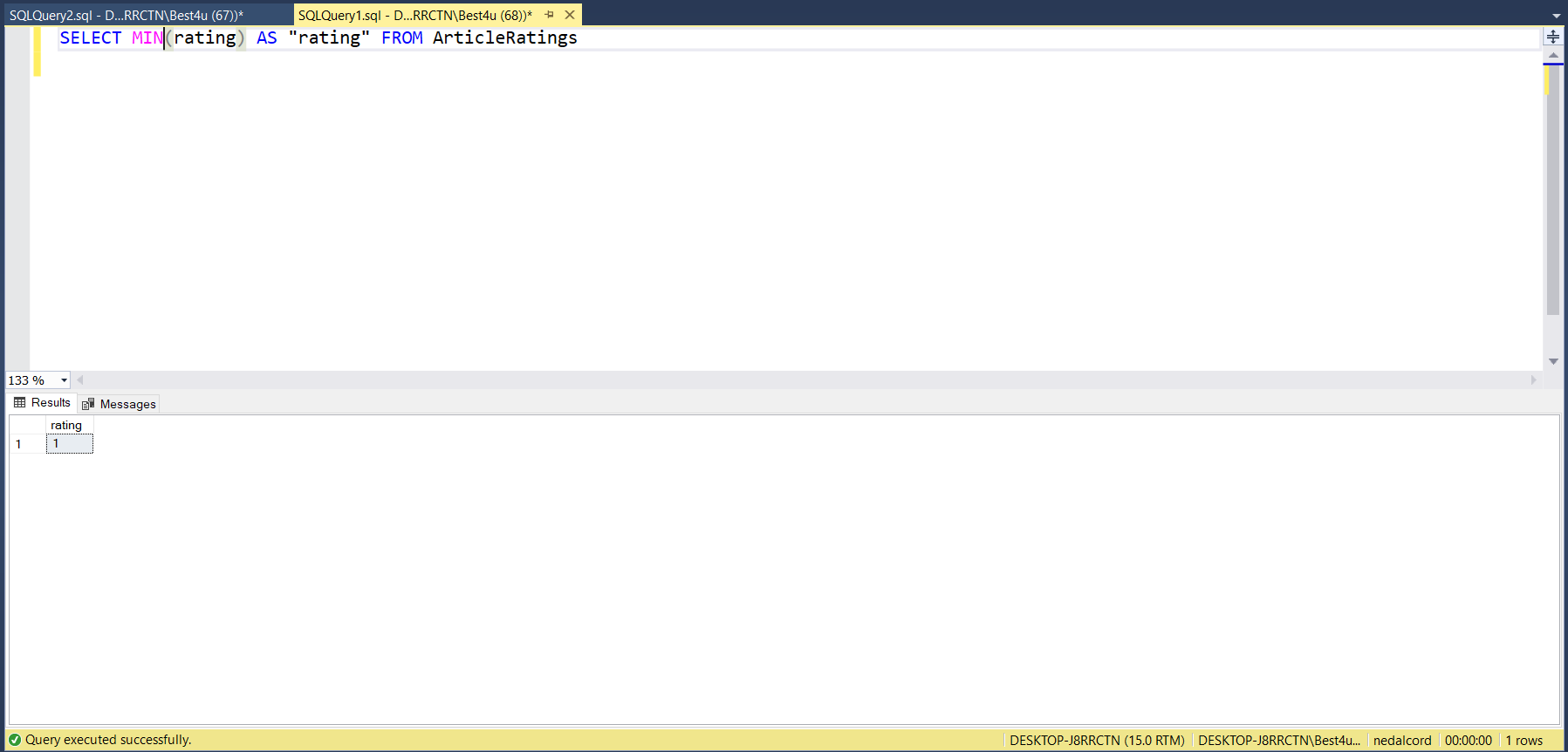
******

**Interogarea de extragere a utilizatorului cu id 1.**

* 1. ***Interogări cu funcții (2):***

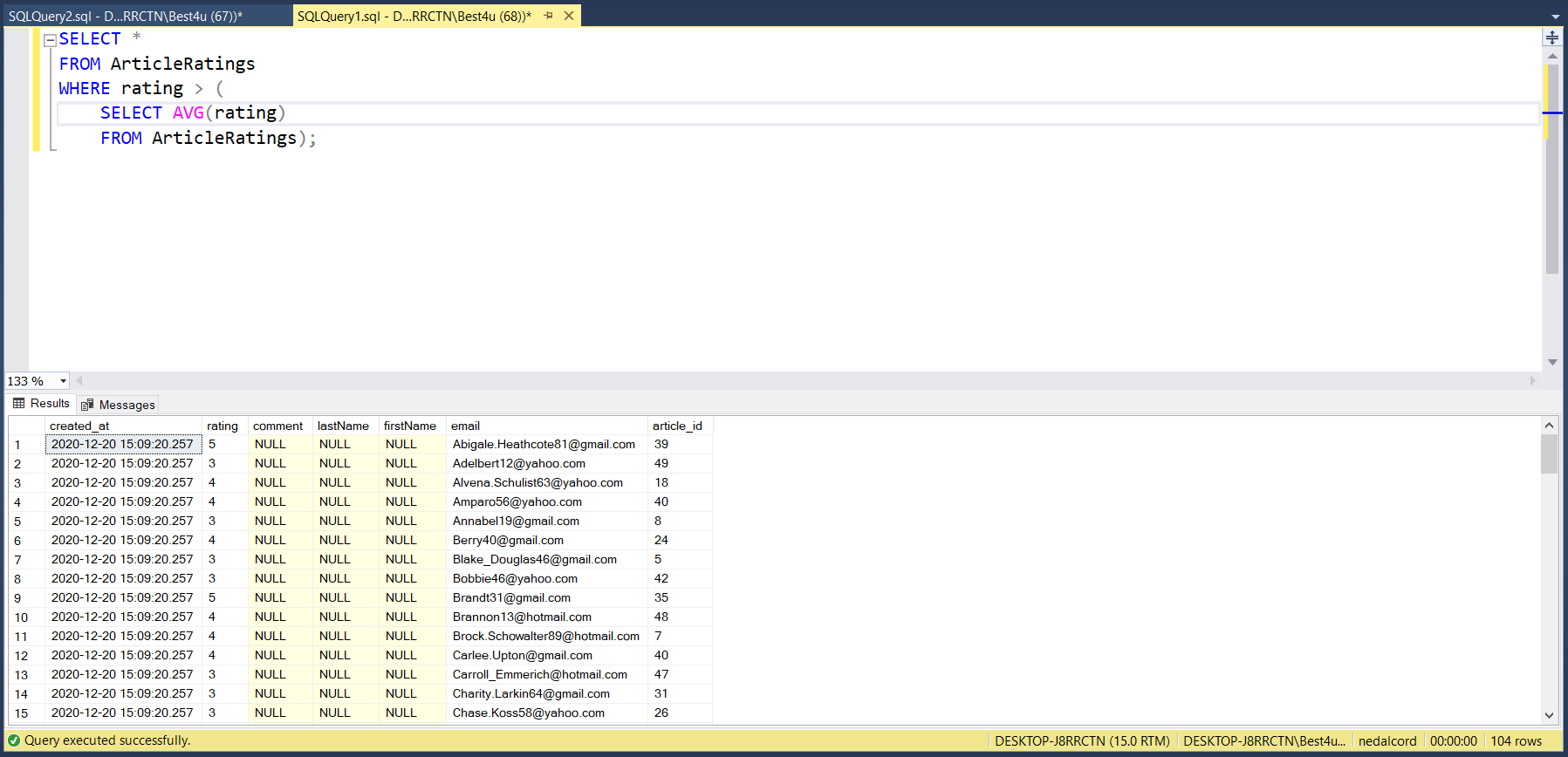
******

***Afișarea celui mai mare rating.***

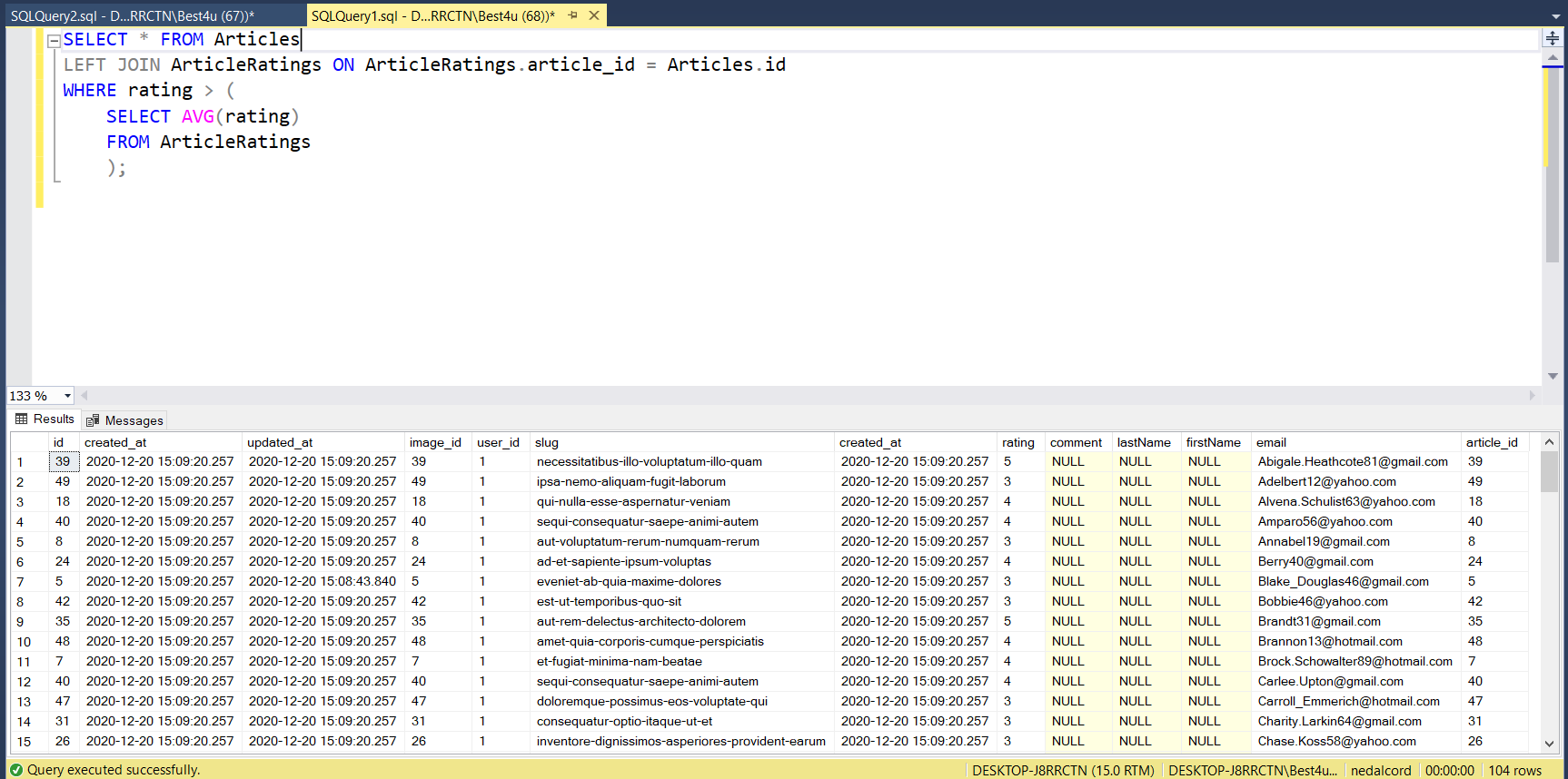
******

***Afișarea celui mai mic rating.***

* 1. ***Interogări imbricate (2):***

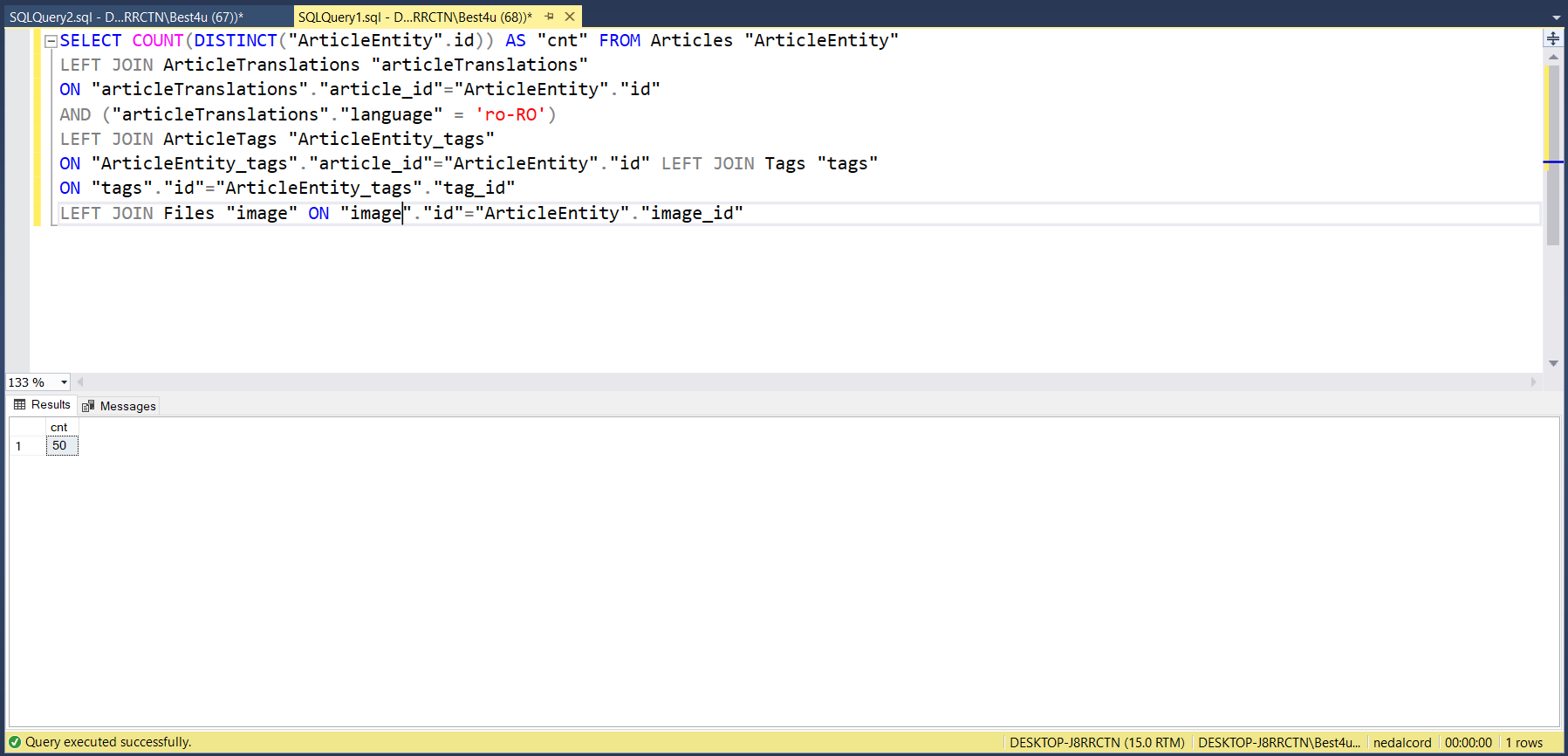
******

**Afișarea ratingului mai mare ca valoarea medie per tabel.**

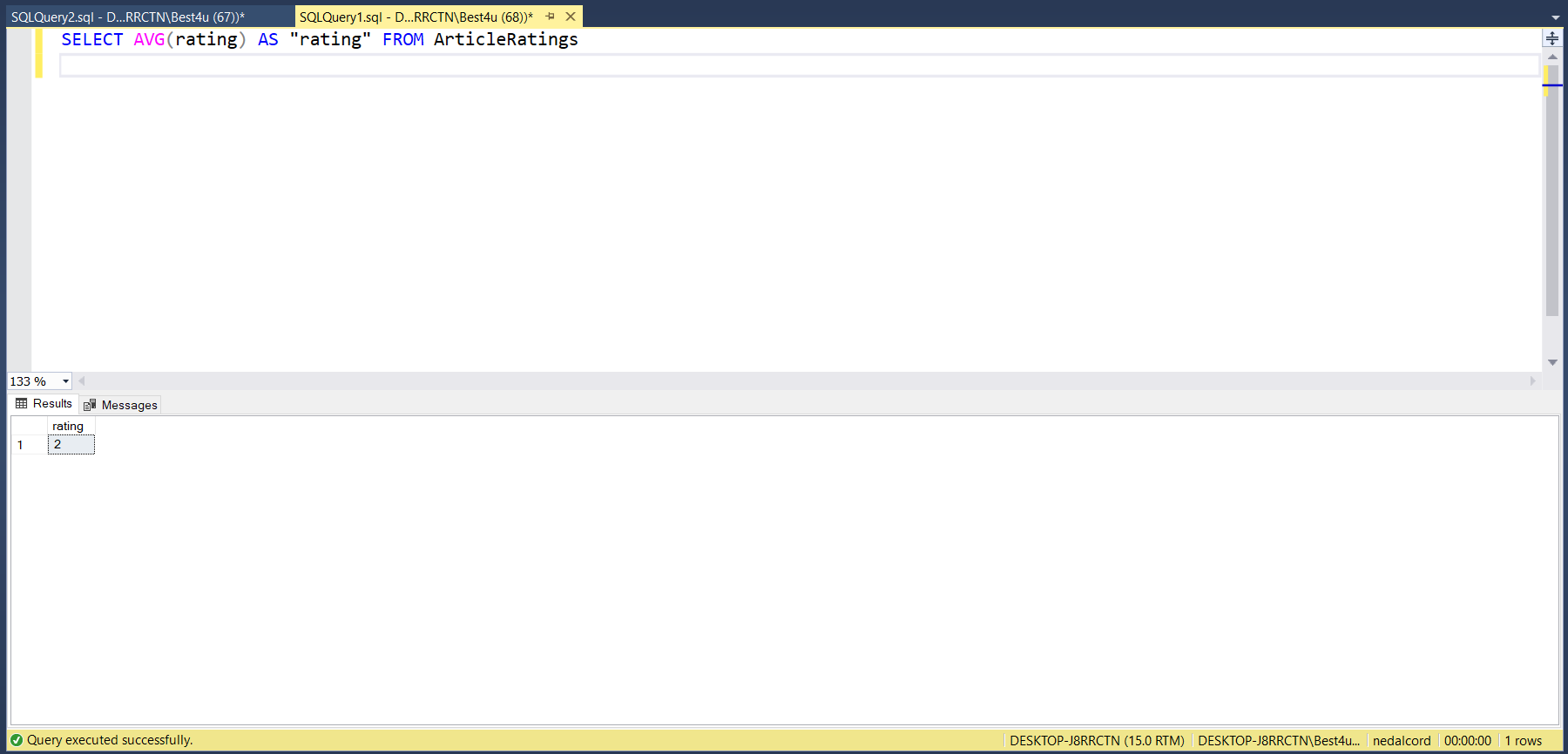
******

**Afișarea articolelor care au ratingul mai mare ca valoarea medie pe tabel.**

* 1. ***Interogări cu funcții agregate si cu grupare (2):***

******

**Interogarea care afisează numărul articolelor.**

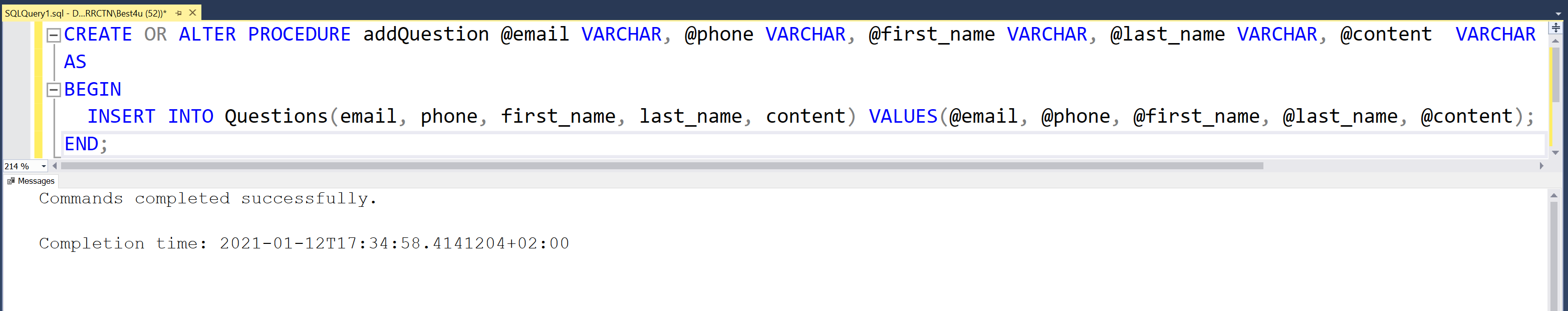
****

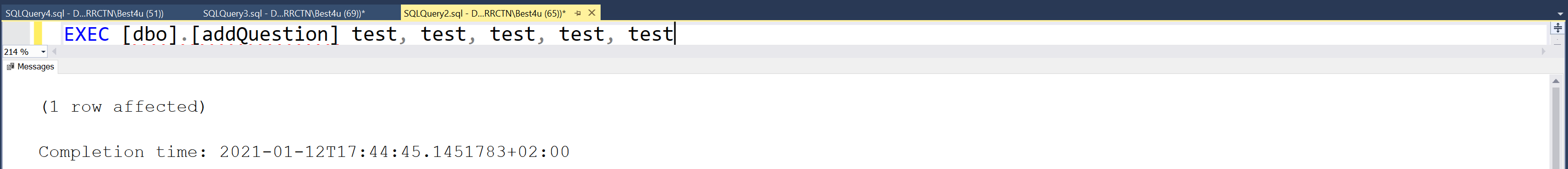
**Afisarea rating-ului mediu al tuturor articolelor.**

# 6. Proceduri si funcții:

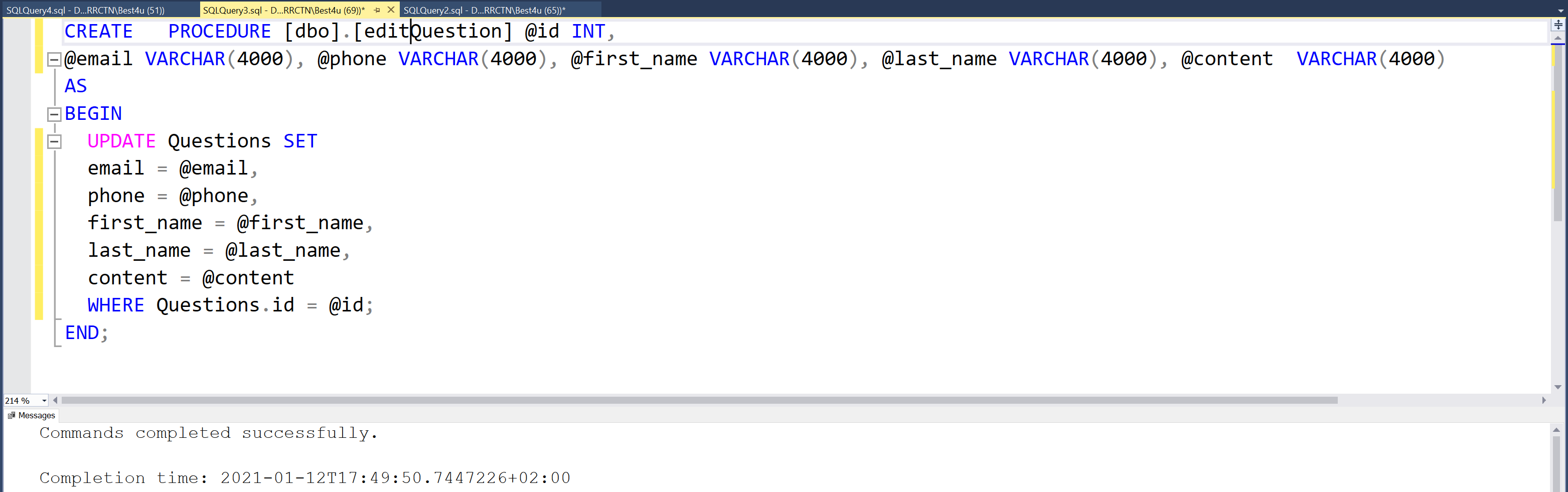
* 1. ***De creat 2 proceduri stocate cu parametri de intrare si ieșire***

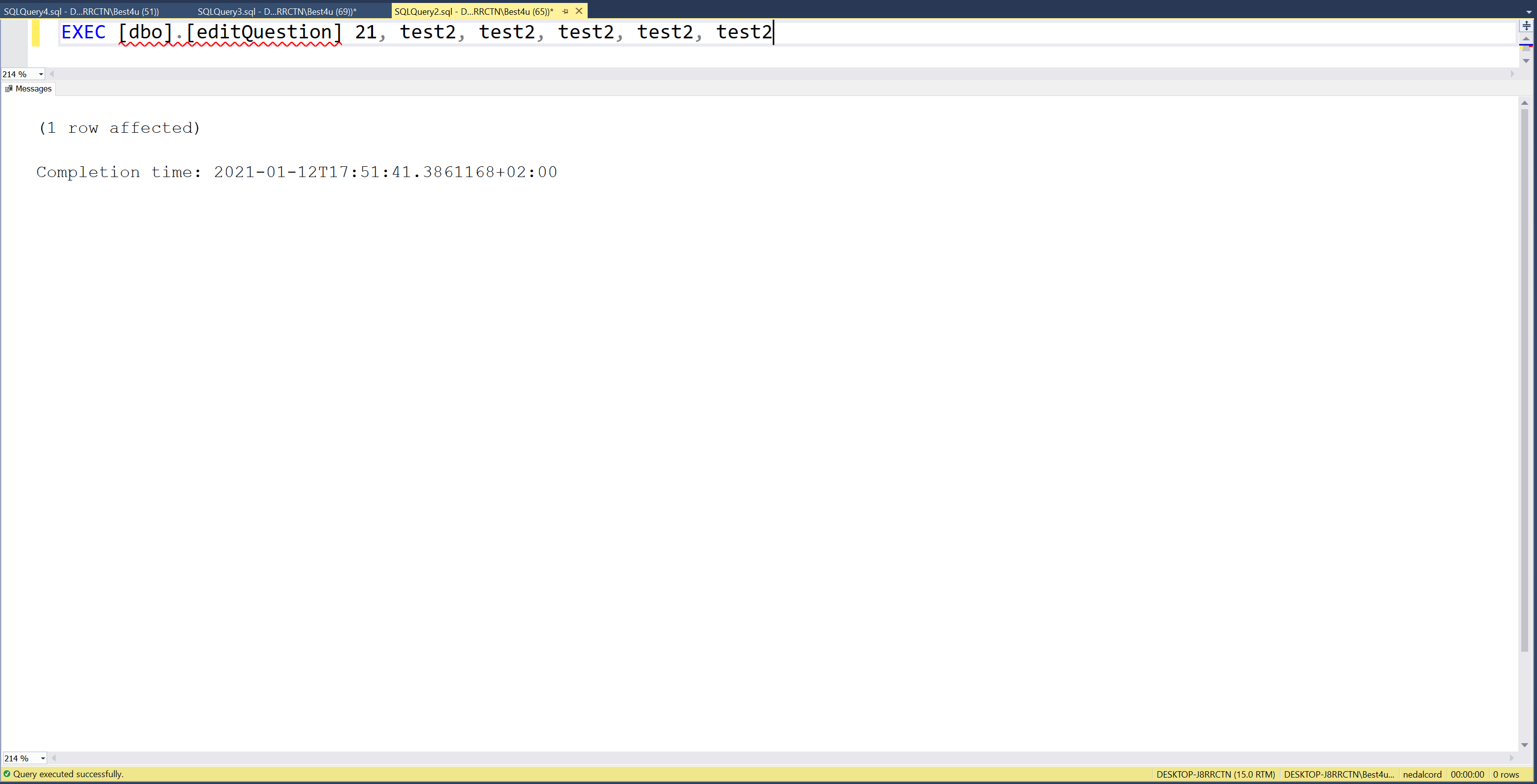
***Procedura de adăugare a unei întrebări.***

******

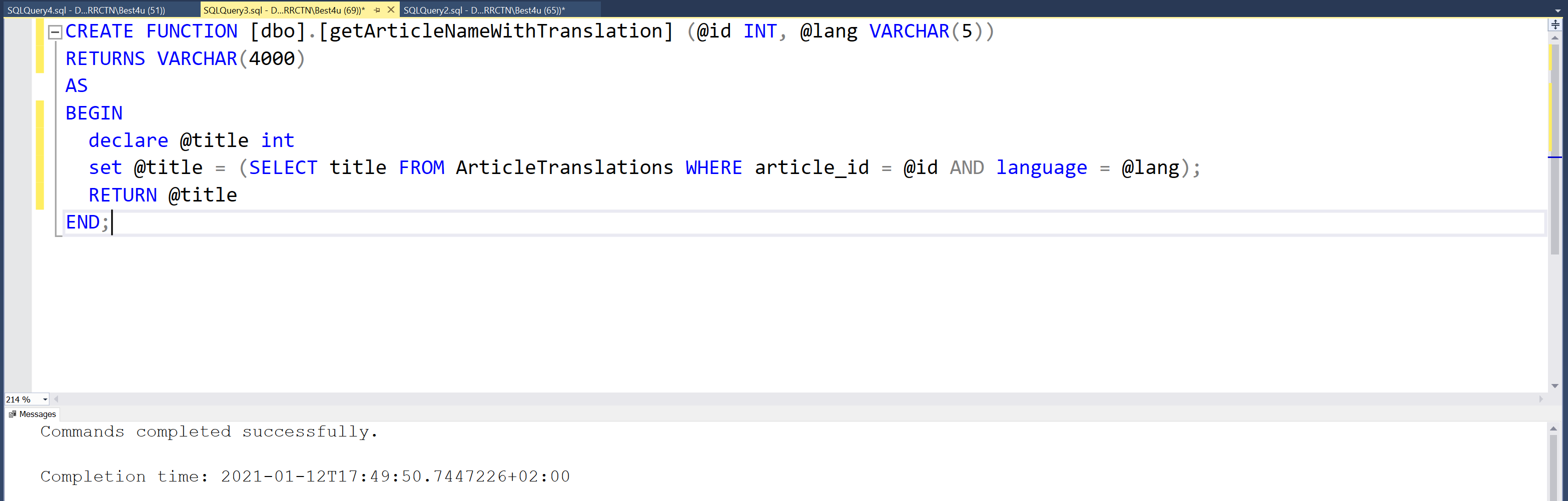
******

***Procedura de editare unei întrebări.***

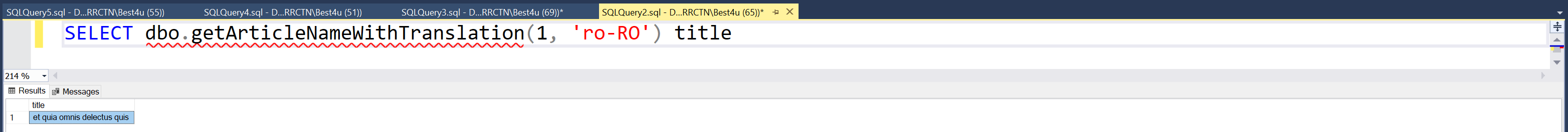
******

******

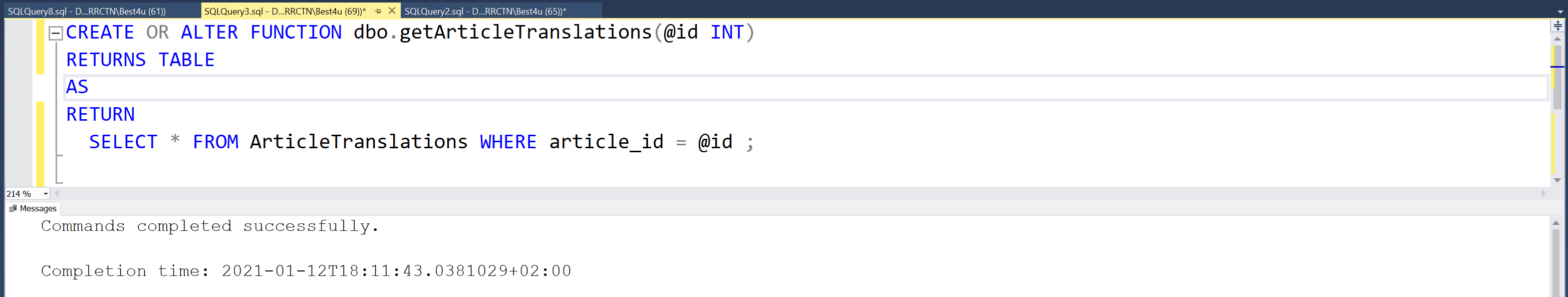
* 1. ***De creat o funcție de tip scalar+ 1 funcție de tip tabel simplu***

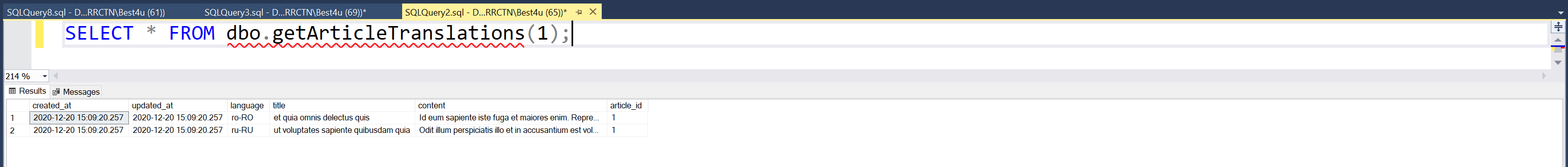
******

***Funcție care returnează titlul unui articol intr-o anumită limbă.***

******

***Funcție care returnează traducerile unui articol***

******

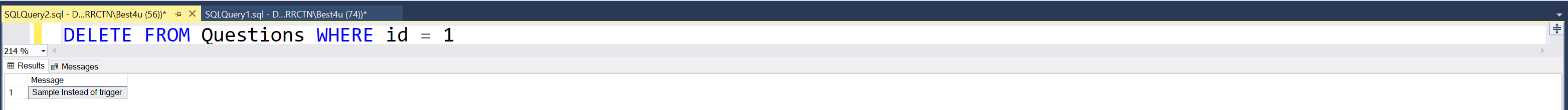
******

# 7. Declanșatoare:

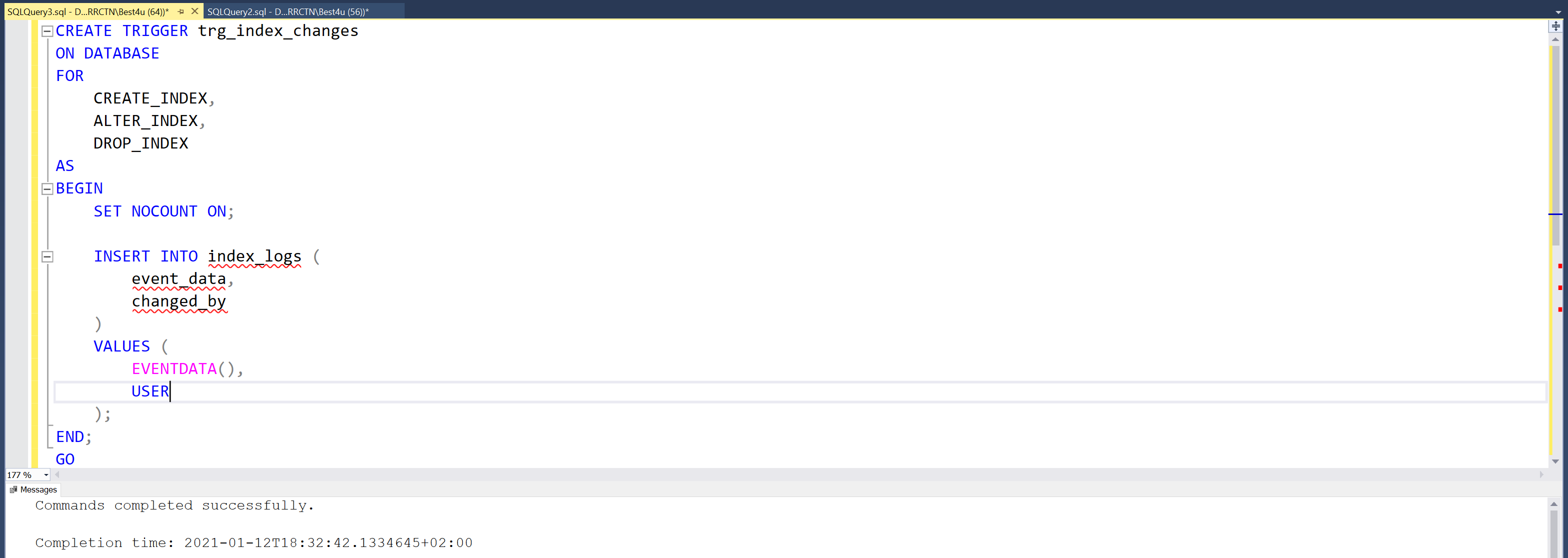
* 1. ***De creat 1 declanșatoare DML***

***Declanșator care afișează un mesaj în locul ștergerii unei întrebări.***

******

******

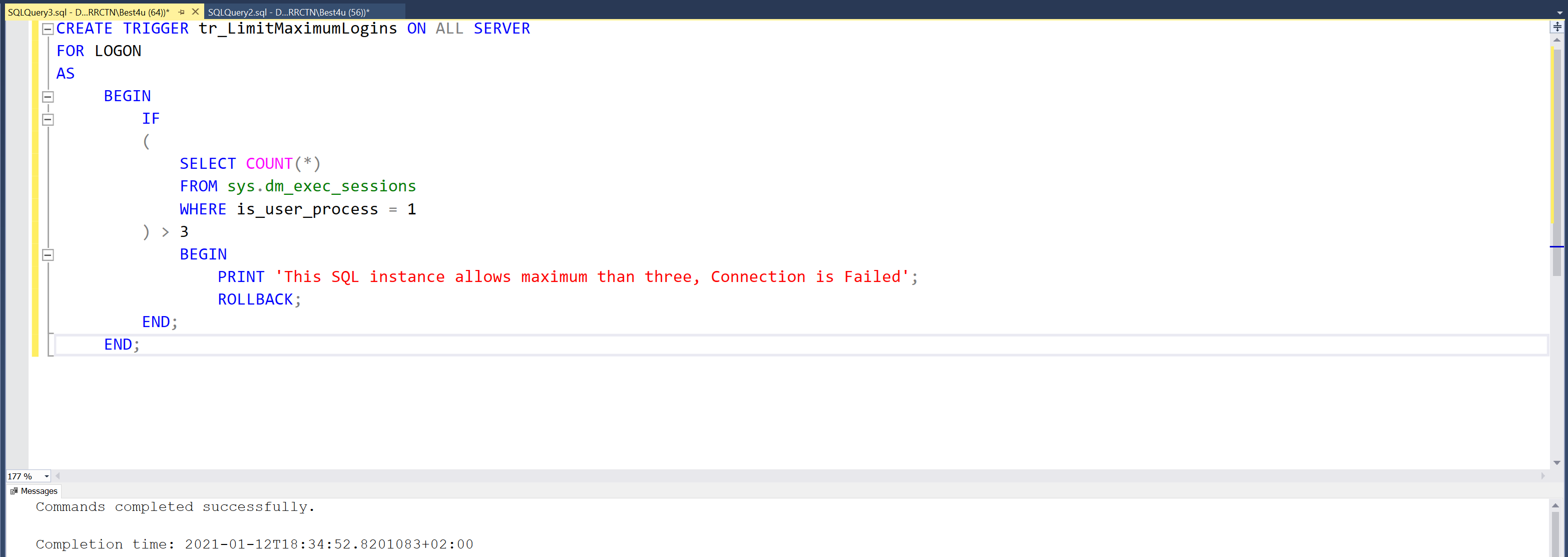
* 1. ***De creat 1 declanșatoare DDL***

******

***Declanșator care stochează operațiile cu indexuri intr-un tabel aparte.***

* 1. ***De creat si 1 declanșator LOGON***

***Declanșator care nu permite mai mulți de 3 utilizatori autentificați.***

******

# 8. Exportarea importarea datelor

1.Sa se efectuaieze 1 exercitii de exportare utilizând SSIS rezultatele interogării „***\_\_\_\_\_\_***

SELECT TOP (1000) [id]

,[created\_at]

,[updated\_at]

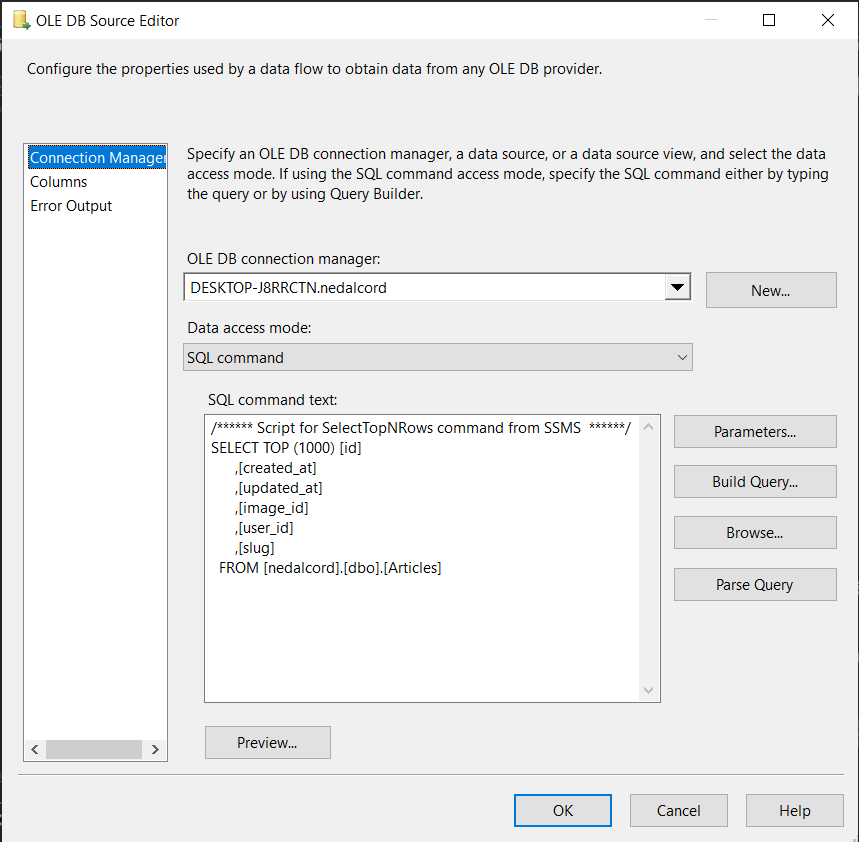
,[image\_id]

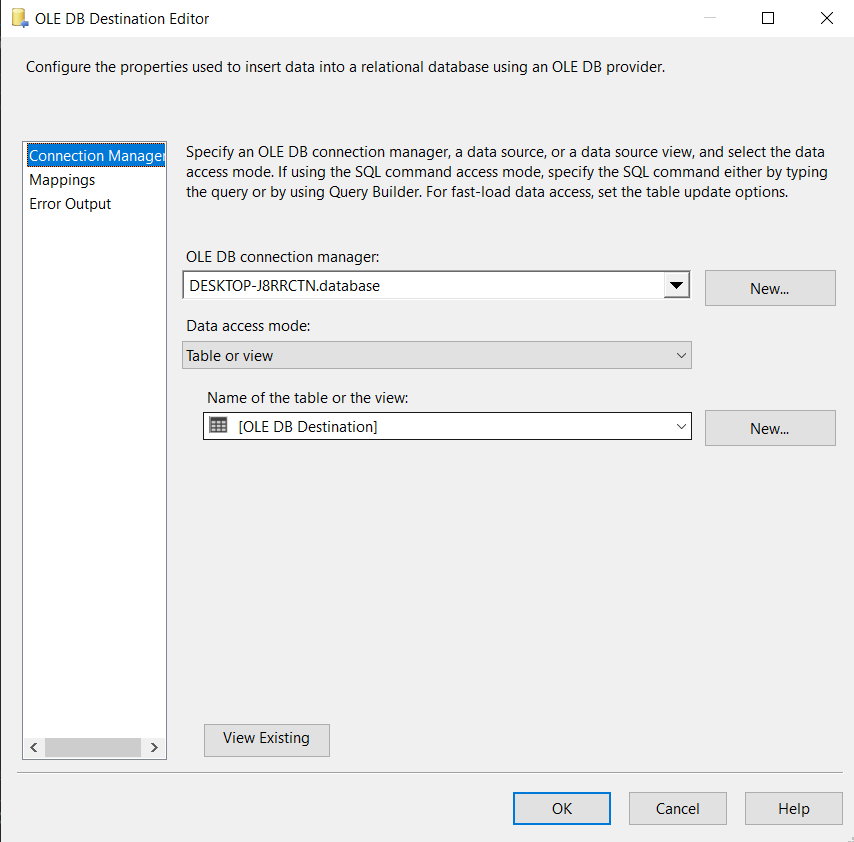
,[user\_id]

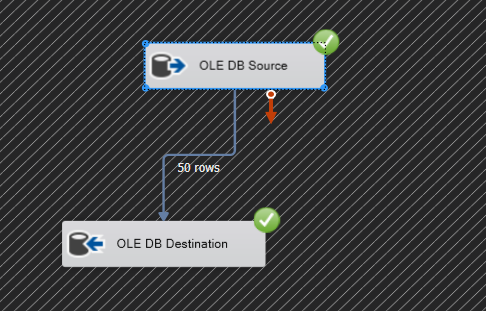
,[slug]

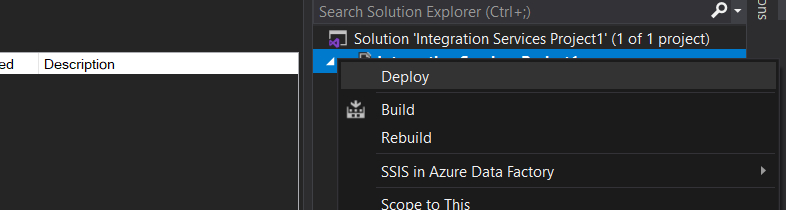
FROM [nedalcord].[dbo].[Articles]

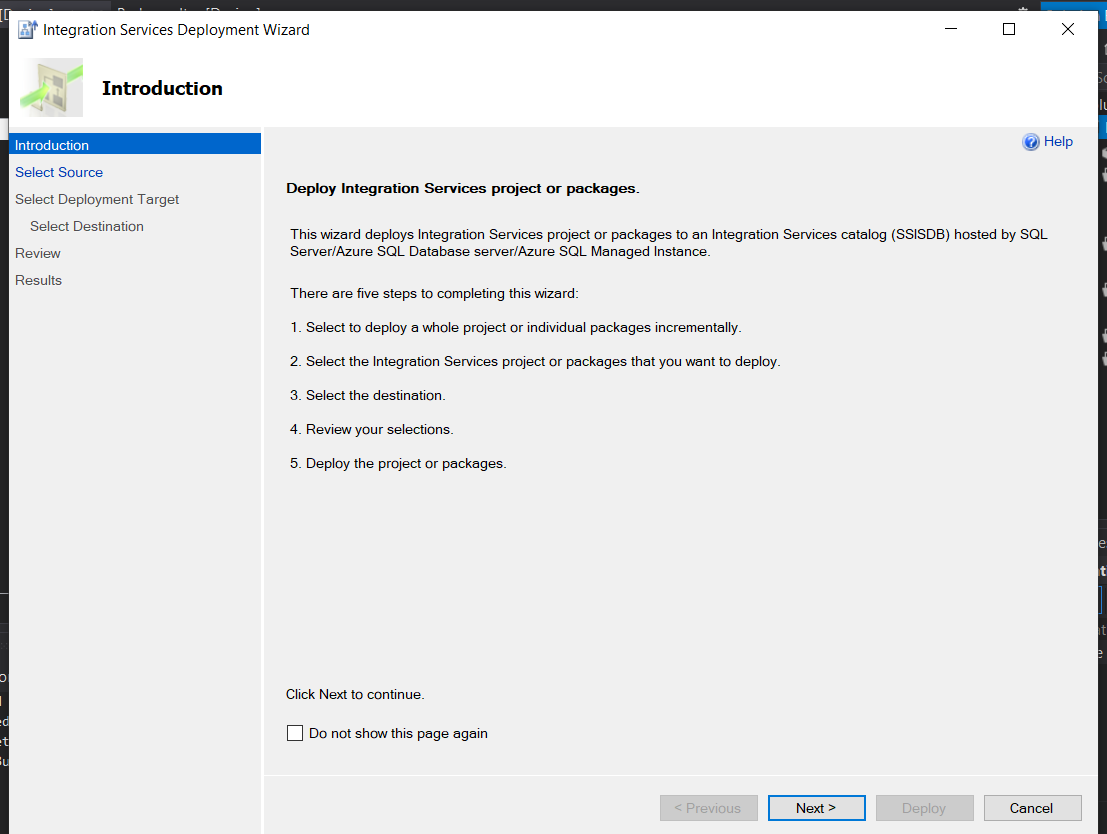
***”*** într-o baza de date ***database.dbo***

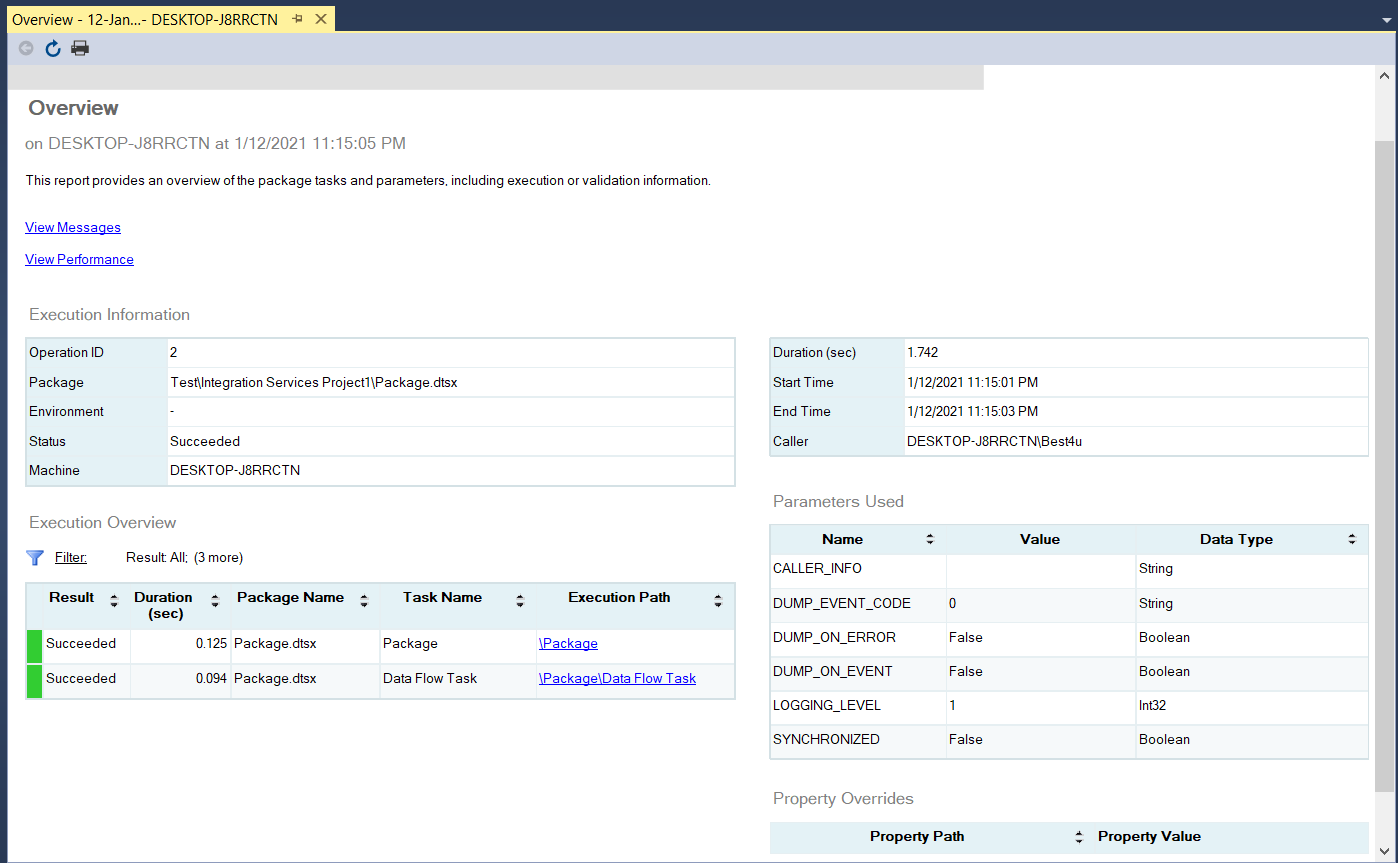
******





2.Proiectul SSIS, creat în Package Designer să fie desfășurat în Intergration Service Catalog și testat





# 9. Servicii de creare a rapoartelor

In baza interogarilor create, sa se creeze 2 rapoarte.

1. Primul - cu ajutorul asistentului, al doilea - in regim de designer. Datele pentru primul raport sa se obtina in Query Builder si sa fie vizualizat sub forma tabelara in formatul **xlsx**.

2. Datele pentru al doilea raport sa reprezinte rezultatul executarii interogarii scrise direct in Query string si sa fie vizualizat sub forma matriceala intr-un fisier **pdf.**

# ANEXE:

**Anexa 1**

Documente care au fost analizate

**Anexa 2**

**Anexa 3**

**Anexa 4**