

Specificatii Proiect Vinyl_Mag

Specificatii functionale

Proiectul urmărește realizarea unei baze de date a unui magazin care se ocupă cu vânzarea discurilor de vinyl, a cd-urilor si a casetelor audio.

Datele care sunt folosite oferă informații legate de:

- Stocul disponibil
- Albume, artiști, genuri muzicale, format muzical
- Furnizori
- Clienți
- Comenzi

Inserarea datelor s-a realizat manual folosind comanda **Insert**.

Au fost create 7 tabele cu legaturi intre ele pentru a putea genera rapoarte si interogari utile.

Specificatii tehnice

1. Tabele si inserarea datelor

1. Artisti

1. Id_artist – cheie primara, legatura cu tabelul Albume
2. Nume_artist

```
create table Artisti (  
    id_artist int identity primary key not null,  
    nume_artist Varchar(100)  
);
```

```
3. Insert.sql - DE...P-9TC9CO5\me (106)
insert into artisti values ('Jamiroquai');
insert into artisti values ('Archive');
insert into artisti values ('Baba Dochia');
insert into artisti values ('Depeche Mode');
insert into artisti values ('Metallica');
insert into artisti values ('Madonna');
insert into artisti values ('Arctic Monkeys');
insert into artisti values ('Editors');
insert into artisti values ('Partizan');
insert into artisti values ('Varna');
insert into artisti values ('Travka');
insert into artisti values ('Wax Tailor');
insert into artisti values ('Etienne de Crecy');
insert into artisti values ('Sebastien Tellier');
insert into artisti values ('Soulwax');
insert into artisti values ('Genesis');
insert into artisti values ('Taylor Swift');
insert into artisti values ('Abba');
insert into artisti values ('Gorillaz');
insert into artisti values ('Justice');
insert into artisti values ('Daft Punk');
insert into artisti values ('Jean Michel Jare');
```

2. Genuri

1. Id_gen – cheie primara, legatura cu tabelul Albume
2. Nume_gen

```
create table Genuri (
    id_gen int identity primary key not null,
    nume_gen Varchar(50)
);
```

```
insert into genuri values ('electronic');
insert into genuri values ('indie');
insert into genuri values ('alternative');
insert into genuri values ('rock');
insert into genuri values ('pop');
insert into genuri values ('Nu-disco');
insert into genuri values ('trip hop');
insert into genuri values ('hip hop');
```

3. Tip_format

1. Id_tip – cheie primara, legatura cu tabelul Albume
2. Nume_tip

```
create table Tip_Format (  
    id_tip int identity primary key not null,  
    nume_tip Varchar(50)  
);  
  
insert into tip_format values ('vinyl');  
insert into tip_format values ('cd');  
insert into tip_format values ('casetă');
```

4. Furnizor

1. Id_furnizor – cheie primara, legatura cu tabelul Albume
2. Furnizor
3. Strada
4. Nr
5. Localitate
6. Judet
7. Email
8. Telefon

```
create table Furnizor (  
    id_furnizor int identity primary key not null,  
    Furnizor Varchar(250),  
    strada Varchar(150),  
    nr Varchar(150),  
    localitate Varchar(100),  
    judet Varchar(50),  
    email Varchar(150),  
    telefon Varchar(50),  
);
```

```
insert into furnizor values ('Melomusic', 'Depoului', '245', 'Bucuresti', 'IF', 'contact@melomusic.ro', '0732890908');  
insert into furnizor values ('Cirkular', 'Emil Isaac', '3', 'Cluj-Napoca', 'CJ', 'contact@cirkular.ro', '0741051000');  
insert into furnizor values ('Emagic', 'Teilor', '69', 'Bucuresti', 'IF', 'comenzi@emagic.ro', '0722987988');
```

5. Albume

1. Id_album – cheie primara
2. Id_artist – legatura cu tabelul artisti
3. Id_gen – legatura cu tabelul genuri
4. Id_furnizor – legatura cu tabelul furnizori
5. Id_tip – legatura cu tabelul tip format
6. Titlu
7. Data_aparitiei
8. In_stoc
9. Cantitate_stoc
10. Pret

```
create table Albume(  
    id_album int identity primary key not null,  
    id_artist int REFERENCES artisti(id_artist),  
    id_gen int REFERENCES genuri(id_gen),  
    id_furnizor int REFERENCES furnizor(id_furnizor),  
    id_tip int REFERENCES tip_format(id_tip),  
    titlu Varchar(100),  
    data_aparitie date,  
    in_stoc Varchar(10),  
    cantitate_stoc int,  
    pret decimal(10,2)  
);
```

```
insert into albume values (4 , 1, 2, 1, 'Music for the Masses', '1987-02-19', 'da', 24 , 100 );
insert into albume values (3 , 1, 1, 2, 'Am Spate', '2022-01-09', 'da', 3 , 90 );
insert into albume values (1 , 6, 2, 1, 'A Funk Odyssey', '2001-03-09', 'da', 10 , 149.9 );
insert into albume values (2 , 7, 2, 1, 'Controlling Crowds', '2009-03-30', 'da', 4 , 180 );
insert into albume values (21 , 1, 2, 1, 'Discovery', '2001-03-12', 'da', 5 , 200 );
insert into albume values (20 , 1, 3, 1, 'Woman', '2016-11-18', 'da', 8 , 249.9 );
insert into albume values (19 , 3, 3, 1, 'Demon Days', '2005-11-05', 'da', 7 , 110 );
insert into albume values (18 , 5, 1, 3, 'Waterloo', '1974-04-03', 'da', 5 , 70 );
insert into albume values (17 , 5, 2, 1, 'Midnights', '2022-10-21', 'da', 33 , 169.9 );
insert into albume values (16 , 4, 1, 1, 'We can dance', '1991-11-11', 'da', 18 , 120 );
```

6. Clienți

1. Id_client – cheie primară, legatură cu tabelul Detalii_comenzi
2. Nume
3. Prenume
4. Strada
5. Nr.
6. Ap.
7. Localitate
8. Județ
9. Email
10. Telefon

```
create table Clienți (
    id_client int identity primary KEY ,
    nume      Varchar(50),
    prenume   Varchar(50),
    strada    Varchar(150),
    nr         Varchar(150),
    ap         Varchar(10),
    localitate Varchar(100),
    judet      Varchar(50),
    email      Varchar(150),
    telefon    Varchar(50)
);
```

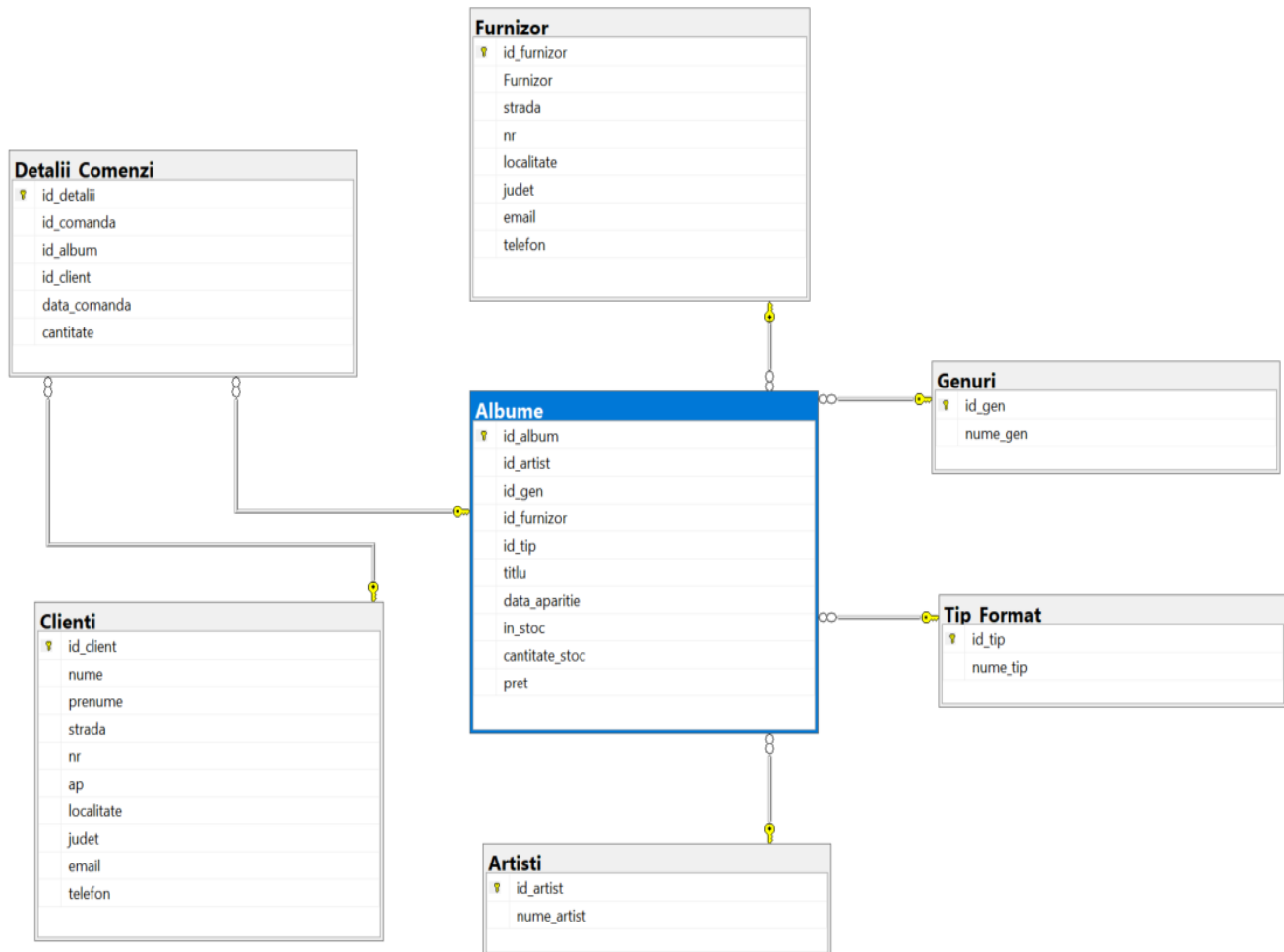
```
insert into clienti values ('Costea', 'Cristian', 'Victoriei ', '47', '77', 'Baia Mare', 'Maramures', 'costeafiorin123@yahoo.com', '0722178987');
insert into clienti values ('Hanganu', 'Florina', 'Magheru ', '23', '60', 'Bucuresti', 'Ilfov', 'florinahanganu@gmail.com', '0722134512');
insert into clienti values ('Candea', 'Larisa', 'Panselutelor', '21', '69', 'Turda', 'Cluj', 'lary.cand@yahoo.com', '0723450987');
insert into clienti values ('Balaj', 'Cristian', 'Buna Ziua', '278', '13', 'Cluj-Napoca', 'Cluj', 'cristibalaj@yahoo.com', '0741051001');
insert into clienti values ('Bergel', 'Erich', 'Antim Ivireanul', '17', '5', 'Cluj-Napoca', 'Cluj', 'erich.bergel@gmail.com', '0749189032');
```

7. Detalii_comenzi

1. Id_detalii – cheie primara
2. Id_comanda
3. Id_album – legatura cu tabelul Albume
4. Id_client – legatura cu tabelul Clienti
5. Data_comanda
6. Cantitate

```
create table Detalii_Comenzi (  
    id_detalii int identity primary key not null,  
    id_comanda int,  
    id_album int references Albume(id_album),  
    id_client int references clienti(id_client),  
    data_comanda date,  
    cantitate int  
);  
  
insert into Detalii_Comenzi values (1, 2, 1, '2022-01-10', 1);  
insert into detalii_comenzi values (1, 13, 1, '2022-01-10', 1);  
insert into detalii_comenzi values (2, 11, 2, '2022-08-03', 1);  
insert into detalii_comenzi values (2, 23, 2, '2022-08-03', 1);  
insert into detalii_comenzi values (2, 20, 2, '2022-08-03', 1);  
insert into detalii_comenzi values (3, 16, 3, '2022-09-03', 2);  
insert into detalii_comenzi values (4, 2, 4, '2022-07-18', 1);  
insert into detalii_comenzi values (4, 3, 4, '2022-07-18', 1);  
insert into detalii_comenzi values (4, 1, 4, '2022-07-18', 1);  
insert into detalii_comenzi values (4, 11, 4, '2022-07-18', 1);  
insert into detalii_comenzi values (5, 21, 5, '2022-08-08', 1);  
insert into detalii_comenzi values (5, 8, 5, '2022-08-08', 1);  
insert into detalii_comenzi values (5, 18, 5, '2022-08-08', 1);  
insert into detalii_comenzi values (6, 23, 2, '2022-08-08', 1);
```

2. Relatii intre tabele



3. Interogari

Pentru interogarea bazei de date s-a incercat folosirea celor mai uzuale functii:
CONCAT, COUNT, SUM, MONTH.

De asemenea a fost ilustrata utilizarea subquery-urilor, clauza WITH, clauza SELECT TOP.

queries.sql - DESK...P-9TC9CO5\me (54)

```
4
5 ---suma pt fiecare comanda
6 select dc.id_comanda, sum(a.pret*dc.cantitate) as suma_comanda
7 from detalii_comenzi dc
8 join albume a on dc.id_album = a.id_album
9 group by dc.id_comanda
10
11 ---suma comenzilor pt luna 08
12 select month(data_comanda) as data_comanda, sum(a.pret*dc.cantitate) as suma_comanda
13 from detalii_comenzi dc
14 join albume a on dc.id_album = a.id_album
15 where month(data_comanda) = '08'
16 group by month(data_comanda)
17
```

100 %

Results Messages

	id_comanda	suma_comanda
1	1	240.00
2	2	309.90
3	3	120.00
4	4	449.90
5	5	309.90
6	6	80.00

	data_comanda	suma_comanda
1	8	699.80

queries.sql - DESK...P-9TC9CO5\me (54)

```
19 ---clientul cu cele mai multe comenzi
20 select top 1 concat(c.nume, ' ', c.prenume) as nume_client, count(distinct id_comanda) as nr_comenzi
21 from detalii_comenzi dc
22 join clienti c on dc.id_client = c.id_client
23 group by concat(c.nume, ' ', c.prenume)
24 order by nr_comenzi desc
25
26 ---comenzi provincie vs bucuresti
27
28 select case when localitate = 'Bucuresti' then localitate else 'Provincie' end as regiune,
29 count(distinct id_comanda) as numar_comenzi from detalii_comenzi dc
30 join clienti c on dc.id_client = c.id_client
31 group by case when localitate = 'Bucuresti' then localitate else 'Provincie' end;
32
```

100 %

Results Messages

	nume_client	nr_comenzi
1	Hanganu Florina	2

	regiune	numar_comenzi
1	Bucuresti	2
2	Provincie	4


```
queries.sql - DESK...P-9TC9CO5\me (54)) X
32
33  --cel mai vechi album
34  select top 1 nume_artist, nume_gen, titlu, data_aparitie from albume a
35  join genuri g on a.id_gen = g.id_gen
36  join artisti ar on a.id_artist = ar.id_artist
37  order by data_aparitie asc
38  |
39
40  --cel mai nou album
41  select top 1 nume_artist, nume_gen, titlu, data_aparitie from albume a
42  join genuri g on a.id_gen = g.id_gen
43  join artisti ar on a.id_artist = ar.id_artist
44  order by data_aparitie desc
45
```

100 %

Results Messages

	nume_artist	nume_gen	titlu	data_aparitie
1	Abba	pop	Waterloo	1974-04-03

	nume_artist	nume_gen	titlu	data_aparitie
1	Taylor Swift	pop	Midnights	2022-10-21

4. Procedura stocata

Scopul acestei proceduri este de a calcula pretul de transport pentru fiecare comanda. Comenzile peste 200 de lei beneficiaza de transport gratuit iar pentru comenzile sub 200 de lei se percepe o taxa de 16 lei.

Procedura are ca parametru de intrare @Id_comanda. Se foloseste clauza WITH si CASE WHEN pentru a calcula cit mai usor.

```
proceduri.sql - DE...P-9TC9CO5\me (54)) X
3  CREATE PROCEDURE PretTransport
4  @id_comanda int
5  AS
6  with pret_per_comanda as (
7  select dc.id_comanda, sum(a.pret*dc.cantitate) as suma_comanda
8  from detalii_comenzi dc
9  join albume a on dc.id_album = a.id_album
10 group by dc.id_comanda
11 )
12 select id_comanda, suma_comanda,
13 case when suma_comanda >= 200 then 0
14 else 16 end pret_transport
15 from pret_per_comanda
16 where id_comanda = @id_comanda;
17
18 exec PretTransport @id_comanda = 3
```

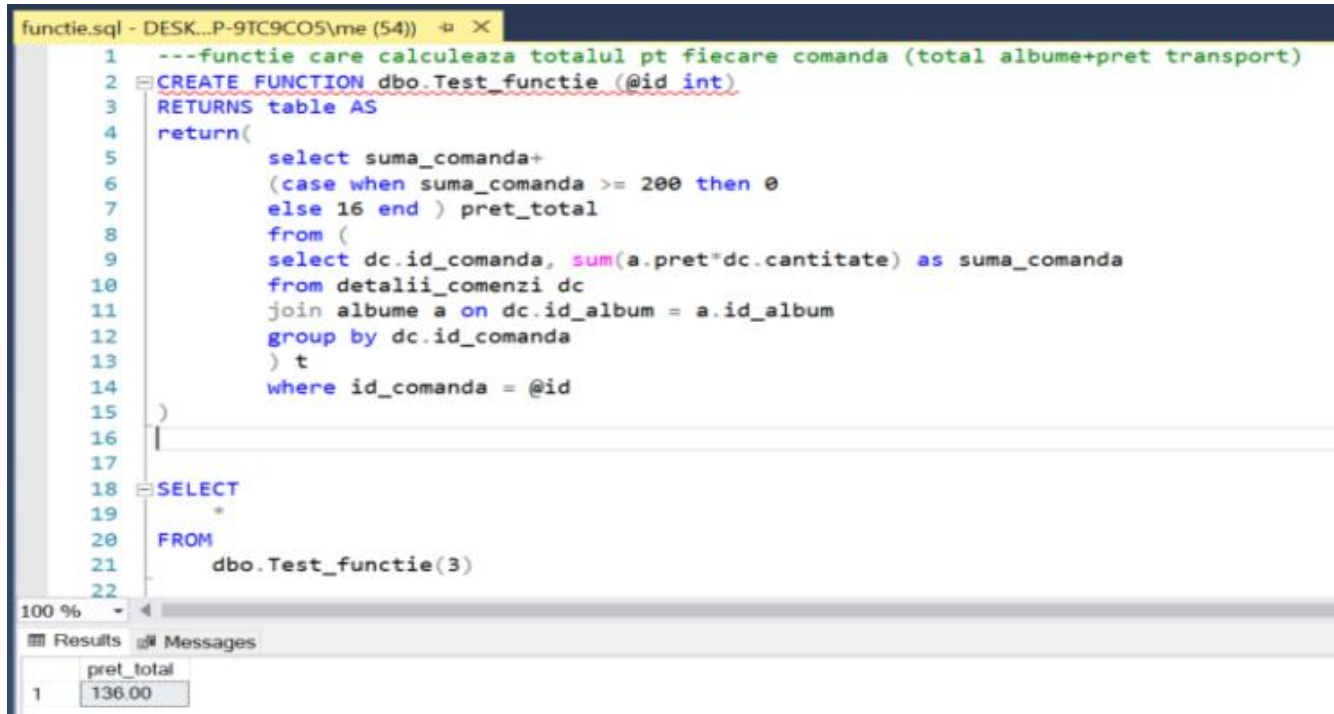
100 %

Results Messages

	id_comanda	suma_comanda	pret_transport
1	3	120.00	16

5. Functia

Functia creata calculeaza totalul pentru fiecare comanda, totalul fiind compus din pretul total al albumelor comandate la care se adauga contravaloarea transportului. Functia are ca parametru de intrare @id si returneaza valorile intr-un tabel.



```
1  ---functie care calculeaza totalul pt fiecare comanda (total albume+pret transport)
2  CREATE FUNCTION dbo.Test_functie (@id int)
3  RETURNS table AS
4  return(
5      select suma_comanda+
6          (case when suma_comanda >= 200 then 0
7            else 16 end ) pret_total
8      from (
9          select dc.id_comanda, sum(a.pret*dc.cantitate) as suma_comanda
10         from detalii_comenzi dc
11         join albume a on dc.id_album = a.id_album
12         group by dc.id_comanda
13       ) t
14     where id_comanda = @id
15 )
16
17
18 SELECT
19 *
20 FROM
21     dbo.Test_functie(3)
22
```

100 %

Results Messages

	pret_total
1	136.00

6. View

View-ul creat contine informatii din tabelele detalii_clienti, albume si clienti si afiseaza toate datele financiare necesare generarii unei facturi: data, nume client, pret albume, cost transport si cost total.

S-au folosit functii precum CONCAT, SUM si clauza CASE WHEN.

```
view.sql - DESKTO...P-9TC9CO5\me (54)) *  X
1  ---view care contine toate informatiile financiare necesare generarii unei facturi
2  -- id_comanda
3  -- data comanda
4  -- nume, prenume client
5  -- cost transport
6  -- cost produse
7  -- cost total
8
9  create or alter view Detalii_financiare as
10 select dc.id_comanda,
11        dc.data_comanda,
12        concat(c.nume, ' ', c.prenume) as nume_prenume,
13        sum(a.pret*dc.cantitate) as pret_albume,
14        case when sum(a.pret*dc.cantitate) >= 200 then 0
15              else 16 end as cost_transport,
16        sum(a.pret*dc.cantitate) + (case when sum(a.pret*dc.cantitate) >= 200 then 0
17                                   else 16 end) as cost_total
18 from detalii_comenzi dc
19 join albume a on dc.id_album = a.id_album
20 join clienti c on dc.id_client = c.id_client
21 group by id_comanda, data_comanda, nume, prenume;
22
23 select * from Detalii_financiare
```

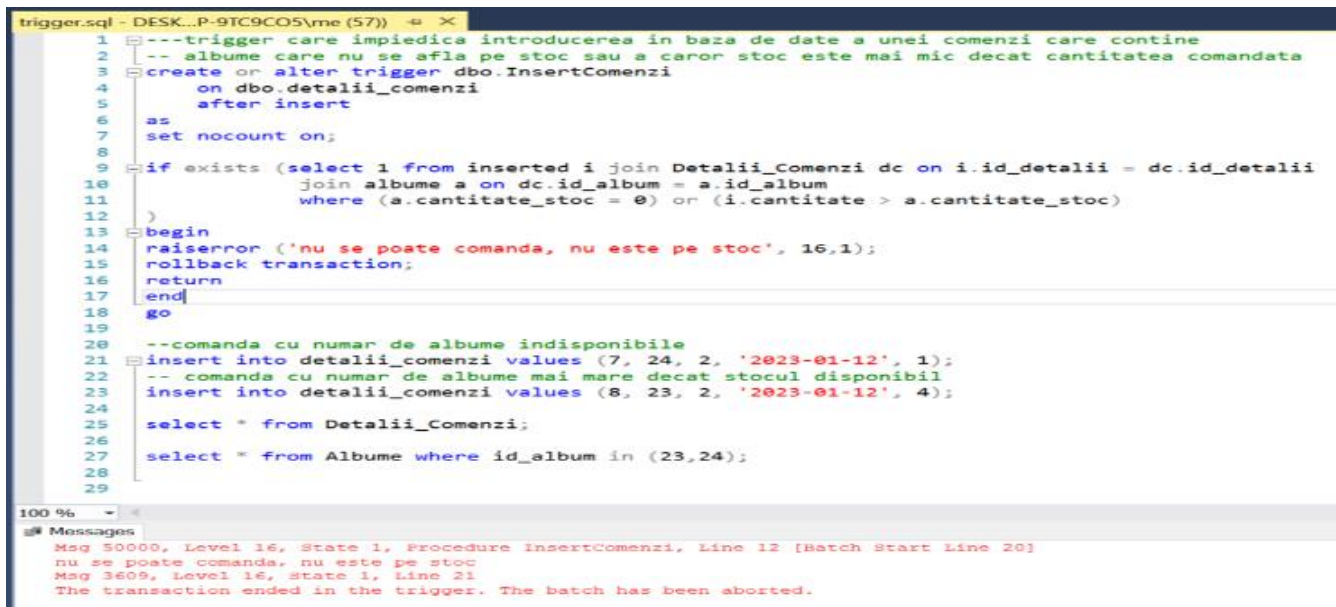
100 %

Results Messages

	id_comanda	data_comanda	nume_prenume	pret_albume	cost_transport	cost_total
1	1	2022-01-10	Costea Cristian	240.00	0	240.00
2	2	2022-08-03	Hanganu Florina	309.90	0	309.90
3	3	2022-09-03	Candea Larisa	120.00	16	136.00
4	4	2022-07-18	Balaj Cristian	449.90	0	449.90
5	5	2022-08-08	Bergei Erich	309.90	0	309.90
6	6	2022-08-08	Hanganu Florina	80.00	16	96.00

7. Trigger

Trigger-ul a fost creat asupra tabelii detalii_comenzi si nu permite inserarea unor comenzi ce contin albume care nu sunt pe stoc sau a caror stoc este mai mic decit comanda. Pentru aceasta se interogheaza tabela albume. In caz ca inserarea datelor nu poate fi realizata din motivele enumerate, trigger-ul returneaza eroarea.



```
trigger.sql - DESK...P-9TC9C05\me (57)
1  --trigger care impiedica introducerea in baza de date a unei comenzi care contine
2  -- albume care nu se afla pe stoc sau a caror stoc este mai mic decat cantitatea comandata
3  create or alter trigger dbo.InsertComenzi
4  on dbo.detalii_comenzi
5  after insert
6  as
7  set nocount on;
8
9  if exists (select 1 from inserted i join Detalii_Comenzi dc on i.id_detalii = dc.id_detalii
10             join albume a on dc.id_album = a.id_album
11             where (a.cantitate_stoc = 0) or (i.cantitate > a.cantitate_stoc)
12         )
13  begin
14      raiserror ('nu se poate comanda, nu este pe stoc', 16,1);
15      rollback transaction;
16      return
17  end
18  go
19
20  --comanda cu numar de albume indisponibile
21  insert into detalii_comenzi values (7, 24, 2, '2023-01-12', 1);
22  -- comanda cu numar de albume mai mare decat stocul disponibil
23  insert into detalii_comenzi values (8, 23, 2, '2023-01-12', 4);
24
25  select * from Detalii_Comenzi;
26
27  select * from Albume where id_album in (23,24);
28
29
```

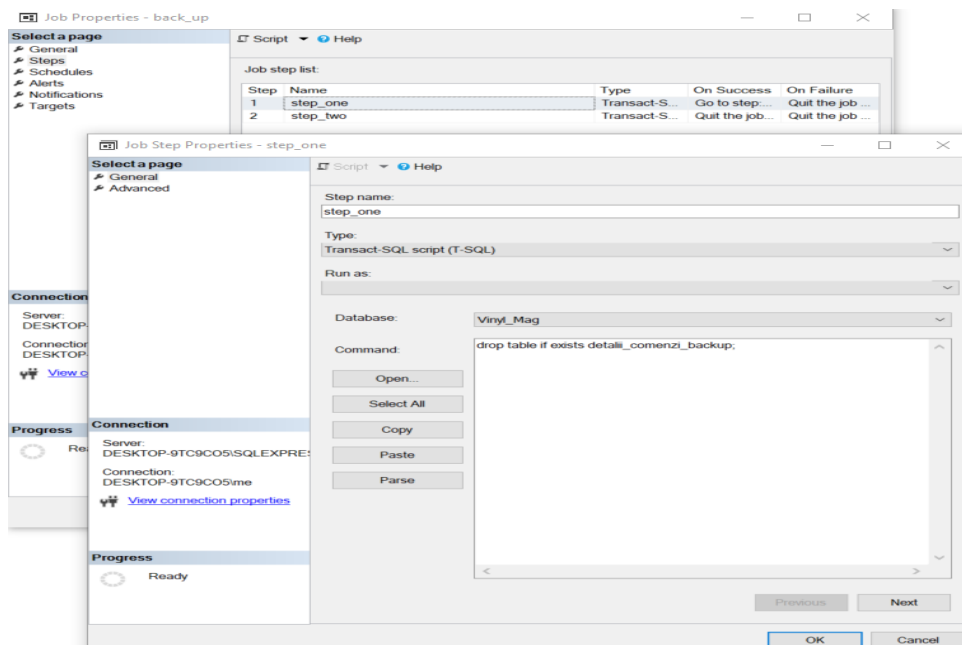
100 %
Messages
Msg 50000, Level 16, State 1, Procedure InsertComenzi, Line 12 [Batch Start Line 20]
nu se poate comanda, nu este pe stoc
Msg 3609, Level 16, State 1, Line 21
The transaction ended in the trigger. The batch has been aborted.

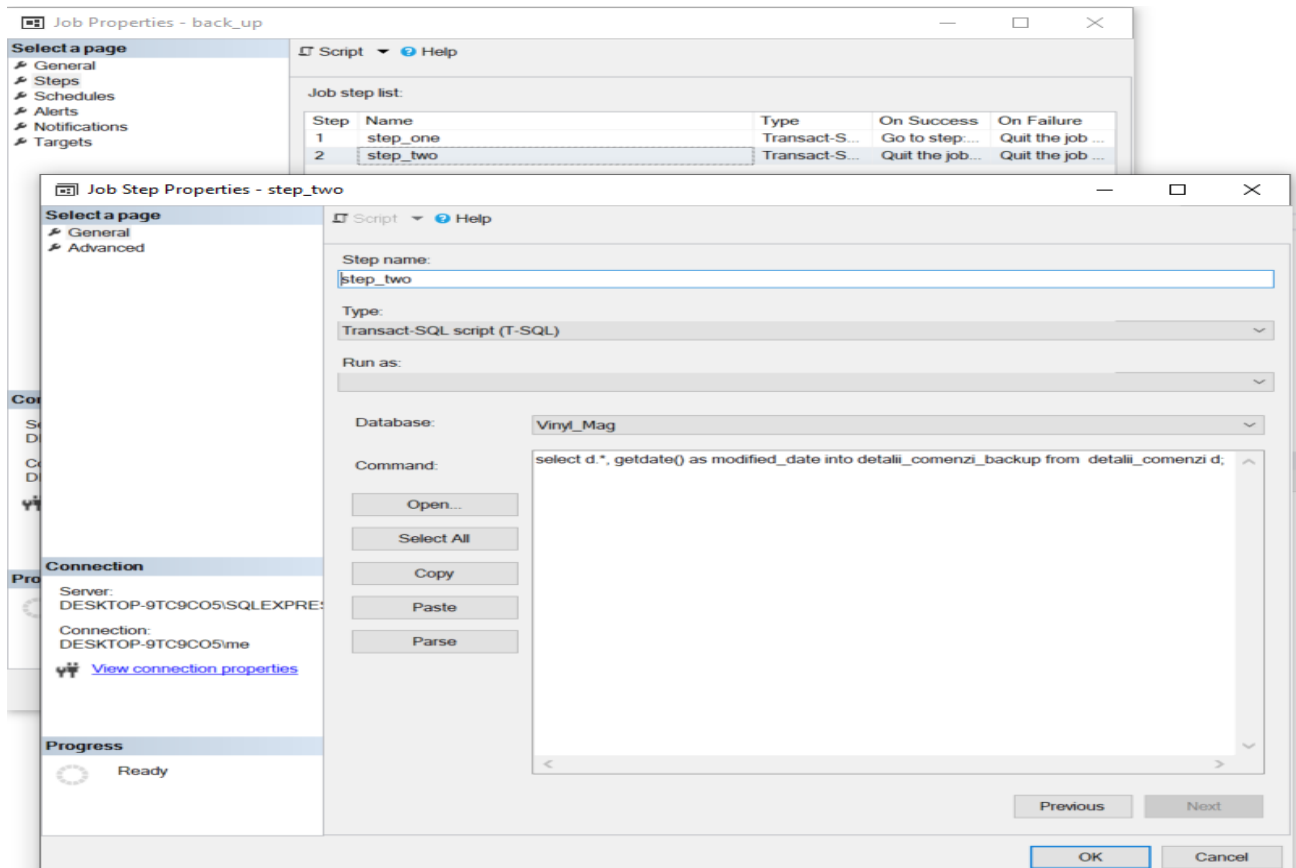
8. Job

Jobul creat contine doi pasi si are ca scop crearea unui back-up zilnic a tabelii detalii_comenzi.

Primul pas sterge din baza de date tabela de back_up daca exista, iar pasul doi recreaza tabela. In plus pe langa datele existente in detalii_comenzi, detalii_comenzi_backup contine si coloana modified_date care este populata cu getdate() de la momentul rularii jobului.

Jobul este programat sa ruleze zilnic.





Concluzii

Baza de date creată oferă informații relevante despre vânzări, cumpărători, stocuri. Prin diverse interogări se pot afla toate informațiile generării unor facturi sau rapoarte de activitate. Am identificat următoarele îmbunătățiri posibile:

- Adăugarea coloanelor `modified_date` și `modified_by_user` pentru a identifica cine și când a efectuat ultima dată modificări;
- Job-ul creat face o copie zilnică a celui mai important tabel, ceea ce pentru tabele de dimensiuni mari ar putea afecta performanța. Am putea modifica acest job astfel încât doar ultimele modificări (incremental) să fie inserate în tabelul de back-up;
- Datele personale care identifică clienții ar putea fi protejate/criptate.

Bibliografie

- <https://www.mssqltips.com/sqlservertip/6269/sql-server-database-diagram-tool-in-management-studio/>
- https://www.w3schools.com/sql/sql_create_table.asp
- https://www.w3schools.com/sql/sql_insert.asp
- <https://learn.microsoft.com/en-us/sql/t-sql/functions/concat-transact-sql?view=sql-server-ver16>
- <https://learn.microsoft.com/en-us/sql/t-sql/functions/sum-transact-sql?view=sql-server-ver16>
- https://www.w3schools.com/sql/sql_case.asp
- <https://learn.microsoft.com/en-us/sql/relational-databases/stored-procedures/execute-a-stored-procedure?view=sql-server-ver16>
- <https://learn.microsoft.com/en-us/sql/relational-databases/user-defined-functions/execute-user-defined-functions?view=sql-server-ver16>
- <https://learn.microsoft.com/en-us/sql/t-sql/statements/create-trigger-transact-sql?view=sql-server-ver16>
- <https://learn.microsoft.com/en-us/sql/ssms/agent/create-a-job?view=sql-server-ver16>