**SEMINARSKI RAD**

## FAKULTET INFORMACIONIH TEHNOLOGIJA

### **Redovne studije**

## Smjer „Programiranje i softversko inžinjerstvo”

### **Predmet**

**BAZE PODATAKA**

**PRODAJA**

**Predmetni nastavnik**

**Student**

**Nikola Popović**

###### Banja Luka, januar 2024.

**Sadržaj**

[**Uvod** 3](file:///C:\Users\Pop\Desktop\BAZE%20-%20SQL\Nikola%20Popović-seminarski%20rad%20.docx#_Toc131705535)

[**1. Dijagram baze podataka** 4](file:///C:\Users\Pop\Desktop\BAZE%20-%20SQL\Nikola%20Popović-seminarski%20rad%20.docx#_Toc131705536)

[**2. DMS i DDS** 6](file:///C:\Users\Pop\Desktop\BAZE%20-%20SQL\Nikola%20Popović-seminarski%20rad%20.docx#_Toc131705537)

[**3. Funkcije** 9](file:///C:\Users\Pop\Desktop\BAZE%20-%20SQL\Nikola%20Popović-seminarski%20rad%20.docx#_Toc131705538)

[**4. UDF funkcije** 11](file:///C:\Users\Pop\Desktop\BAZE%20-%20SQL\Nikola%20Popović-seminarski%20rad%20.docx#_Toc131705539)

[**5. Index** 13](file:///C:\Users\Pop\Desktop\BAZE%20-%20SQL\Nikola%20Popović-seminarski%20rad%20.docx#_Toc131705540)

[**6. JOIN** 14](file:///C:\Users\Pop\Desktop\BAZE%20-%20SQL\Nikola%20Popović-seminarski%20rad%20.docx#_Toc131705541)

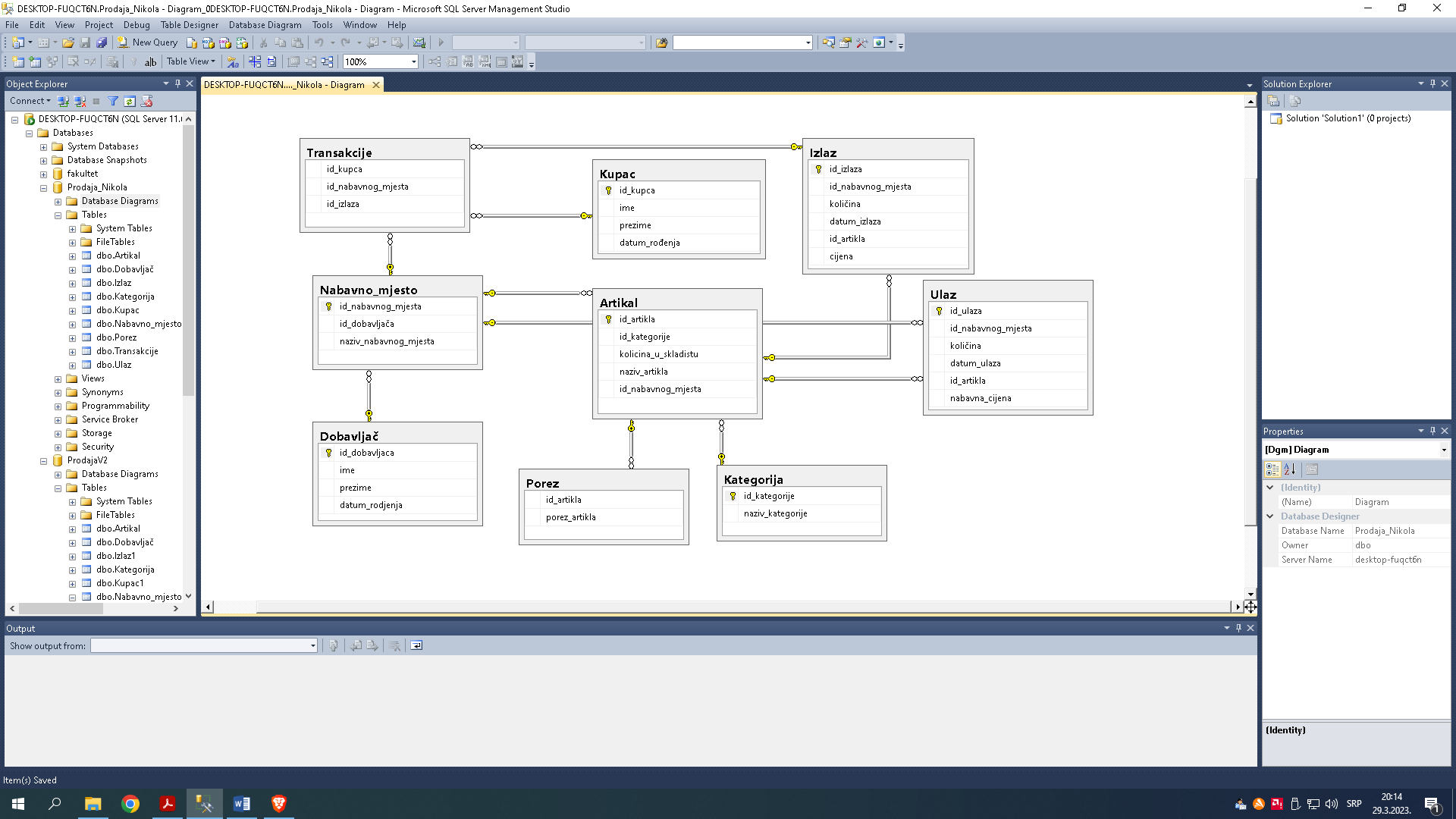
# **Uvod**

Moj zadatak za ovaj seminarski rad je da prikažem projekt baze podataka koja omogućava evidentiranje i korištenje podataka za trgovinu.

Takođe je potrebno da omogućim evidentiranje partnera( kupaca, dobavljača), poreza, artikala, kategorija, ulaza, izlaza(prometa), i drugo iz predmetne oblasti, prema potrebi. Prikazati mogućnosti projektovane baze podataka prikladnim upitima. Da bih riješio ovaj zadatak koristio sam različite funkcije i metode.

# **1. Dijagram baze podataka**

Jedan od uslova je bio da se naprave tabele: Kupac, Dobavljač, Ulaz, Izlaz, Porez, Artikal, Nabavno\_mjesto, Kategorija i Trasakcija. U slici ispod su prikazane sve navedene tabele.



Ilustracija 1. Dijagram baze podataka

Tabela Kupac prikazuje kupca koji je obavio neku kupovinu u prodajnom mjestu.

Tabele Ulaz i Izlaz nam sluze da pratimo kolicinu artikala koju ilaze i izlaze iz prodajnih mjesta.

Tabela Porez nam prikazuje koliki je porez na artikle.

Tabela Artikal nam je mozemo reći najvažnija tabela i ona posjeduje sve informacije o artiklima (naziv, stanje, nabavno mjesto).

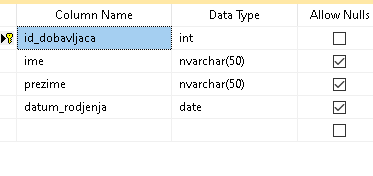
Tabela Nabavno\_mjesto nam pruža informacije o nabavnom mjestu.

Tabela Dobavljač nam pruža informacije o odbavljaču.

Tabela Kategorija je lista kategorija u koju možemo da svrstamo artikal.

Tabela Transakcije nam govori sve o obavljenoj transakciji (koje je nabavno mjesto, ko je kupac i podatak o izlazu artikla).

U nastavku cu prikazati primjer jedne tabele i njenih tipova podataka.



Ilustracija 2. Tipovi podataka

# **2. DMS i DDS**

Data definition language (DDL) je sintaksa za kreiranje i modifikovanje objekata baze podataka kao što su tabele, indeksi i korisnici.

Naredbe za definisanje podataka (engl. Data Definition Statements). Omogućuju definisanje resursa relacione baze podataka.

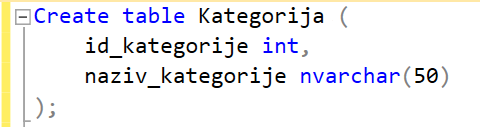
CREATE TABLE - kreiranje tebele baze podataka

CREATE VIEW - kreiranje virtuelne tabele – „pogleda“

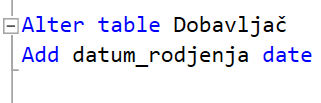
CREATE INDEX - kreiranje indeksa nad kombinacijom kolona tabele

ALTER TABLE - izmjena definicije tabele

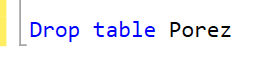
DROP TABLE – izbacivanje (brisanje) tabele iz baze podataka



Ilustracija 3. Kreiranje tabele



Ilustracija 4. Izmjena tabele



Ilustracija 5. Brisanje tabele

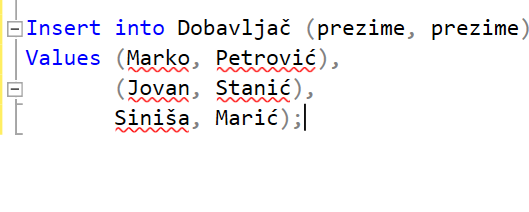
Naredbe za manipulisanje (rukovanje) podacima (engl. Data Manipulation Statements). Omogućuju ažuriranje i izvještavanje iz relacione baze podataka.

INSERT – dodavanje redova postojećoj tabeli

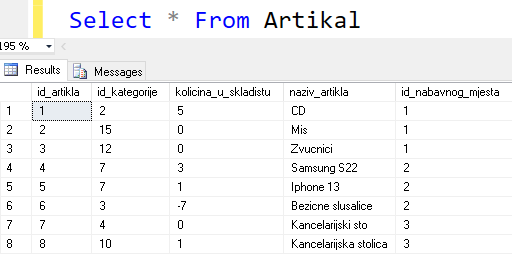
SELECT – prikaz sadržaja relacione baze podataka

UPDATE – izmjena vrijednosti kolona tabele

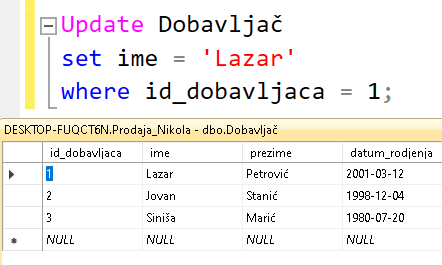
DELETE – izbacivanje redova tabele



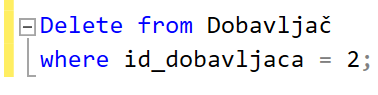
Ilustracija 6. Dodavanje redova



Ilustracija 7. Prikaz sadržaja



Ilustracija 8. Ažuriranje sadržaja

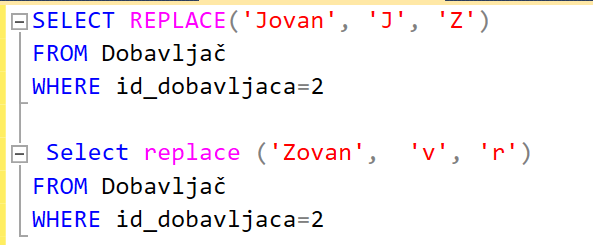


Ilustracija 9. Brisanje reda

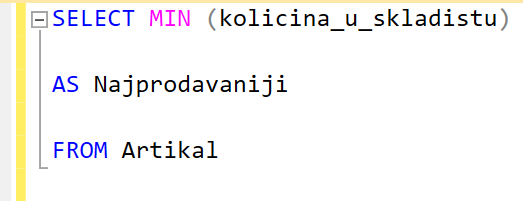
# **3. Funkcije**

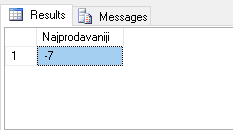
SQL ima mnogo različitih funkcija, u nastavku ćemo ih prikazati.

1. Funkcija za nizove

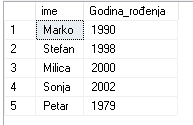
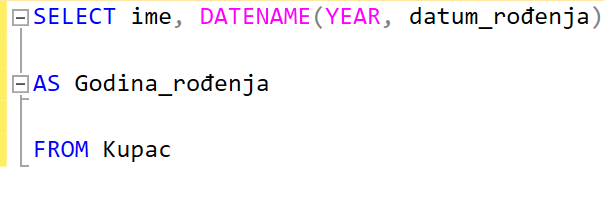


2. Funkcija za brojeve

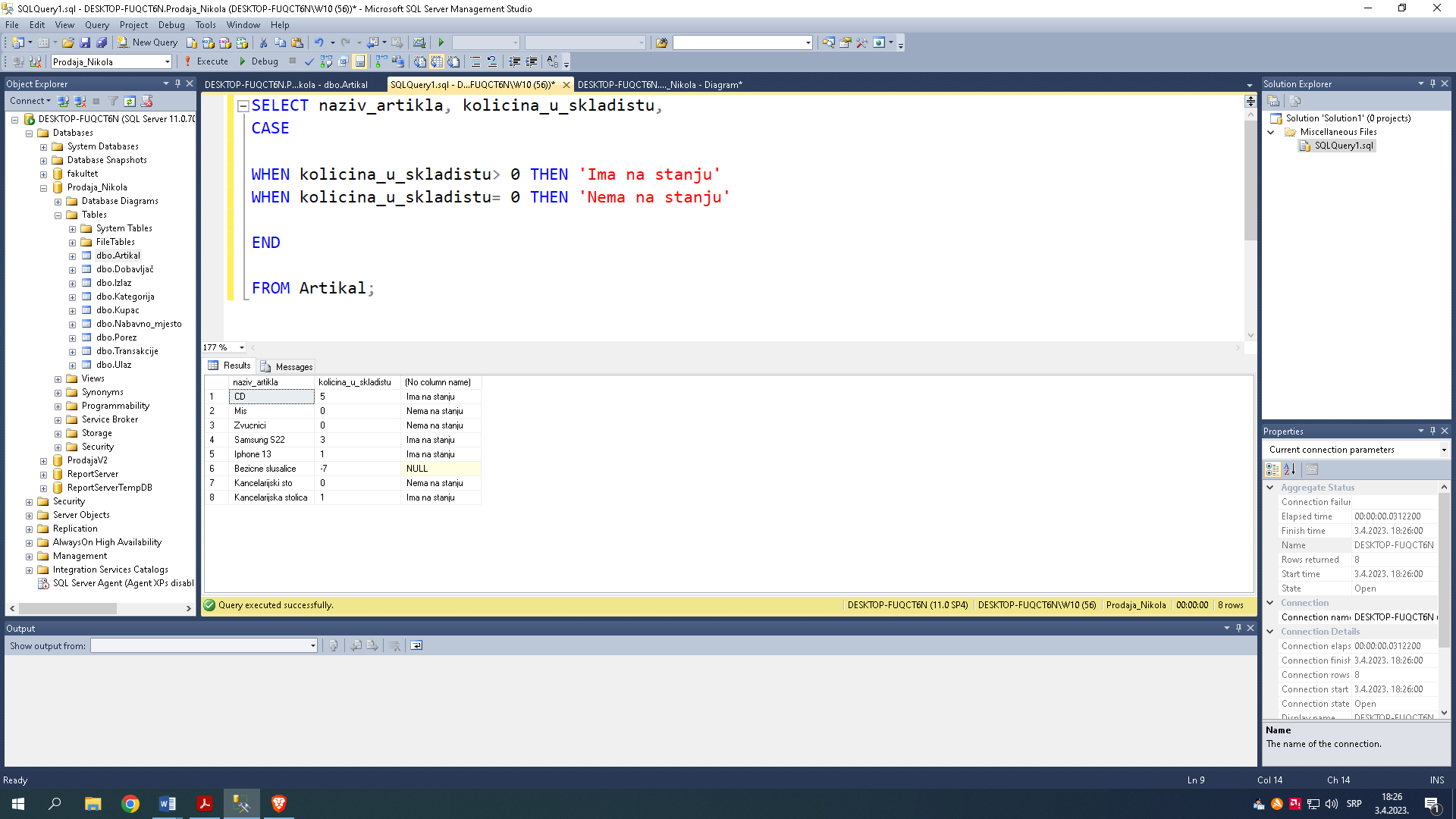




3. Funkcija za datume



4. Funkcija opšte namjene



**STRING\_AGG** funkcija nažalost nije dostupna u SQL Server 2016 verziji tako da ne mogu prikazati primjer na svojoj bazi podataka.

# **4. VIEW (Pogled)**

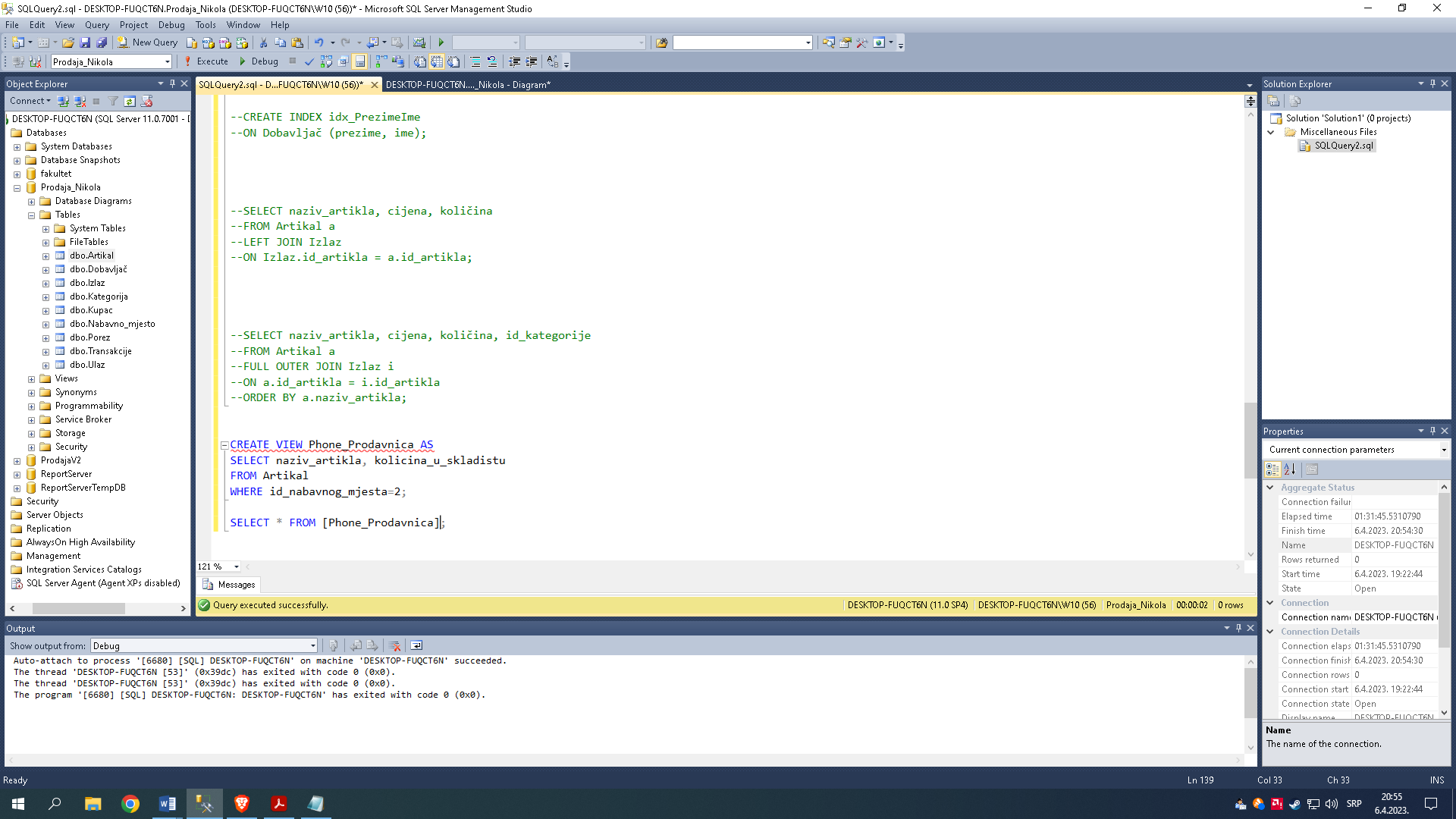
U SQL-u, pogled je virtuelna tabela zasnovana na skupu rezultata SQL naredbi.

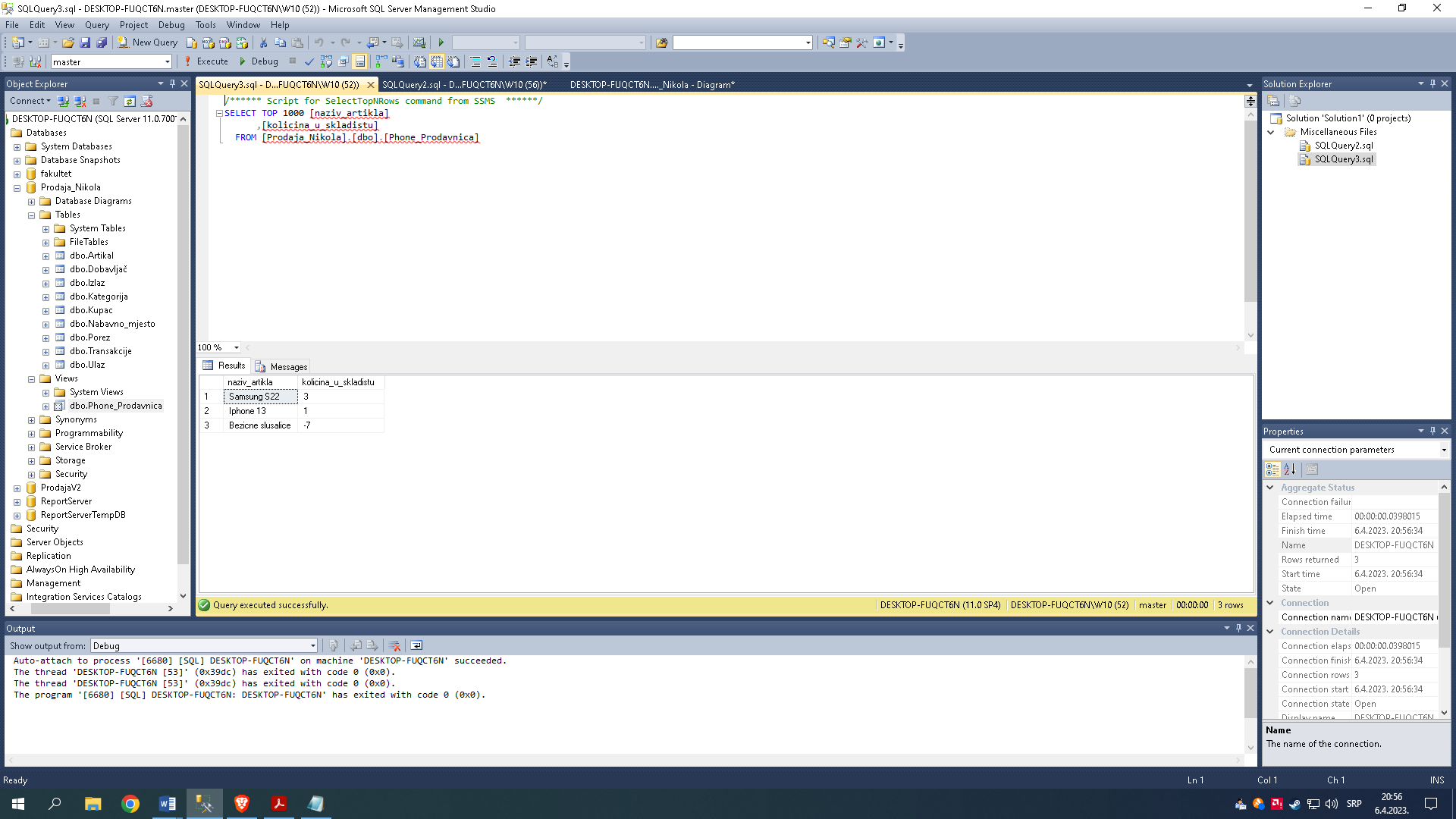
Prikaz sadrži redove i kolone, baš kao prava tabela. Polja u prikazu su polja iz jedne ili više stvarnih tabela u bazi podataka.

Možete dodati SQL izraze i funkcije u prikaz i predstaviti podatke kao da podaci dolaze iz jedne tabele.

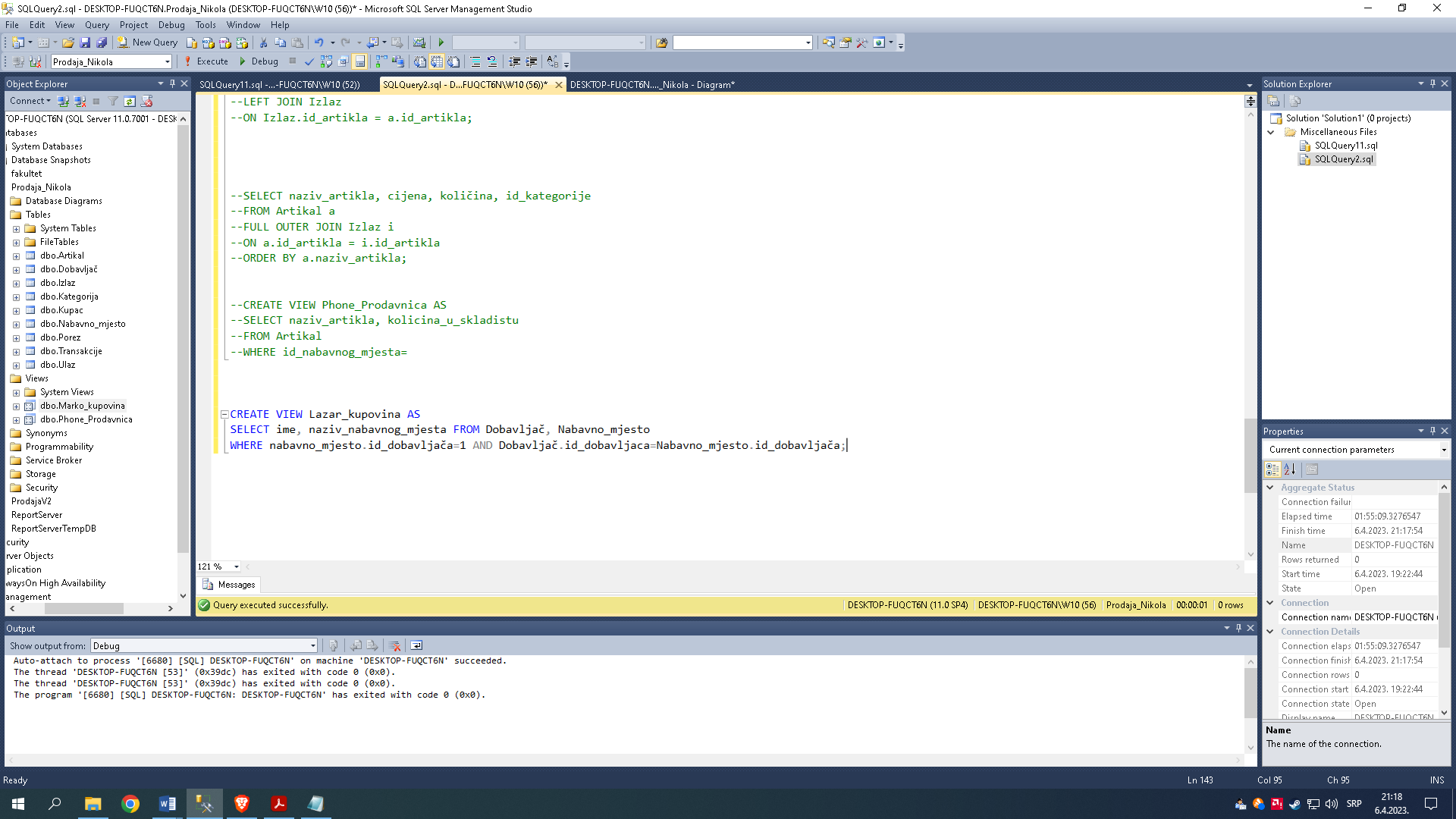
Prikaz se kreira pomoću naredbe CREATE VIEW.

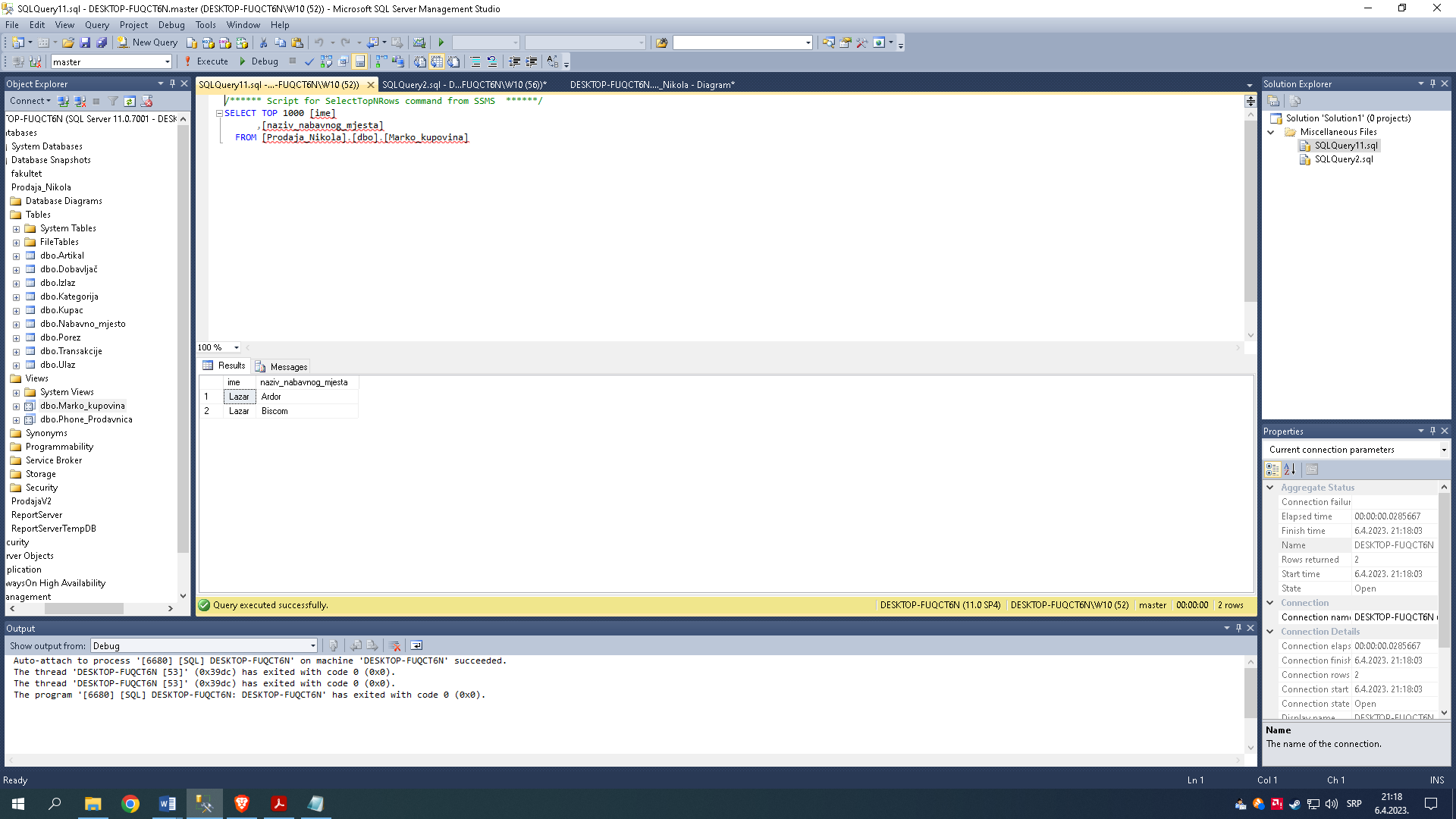
Prvi primjer:





Drugi primjer:





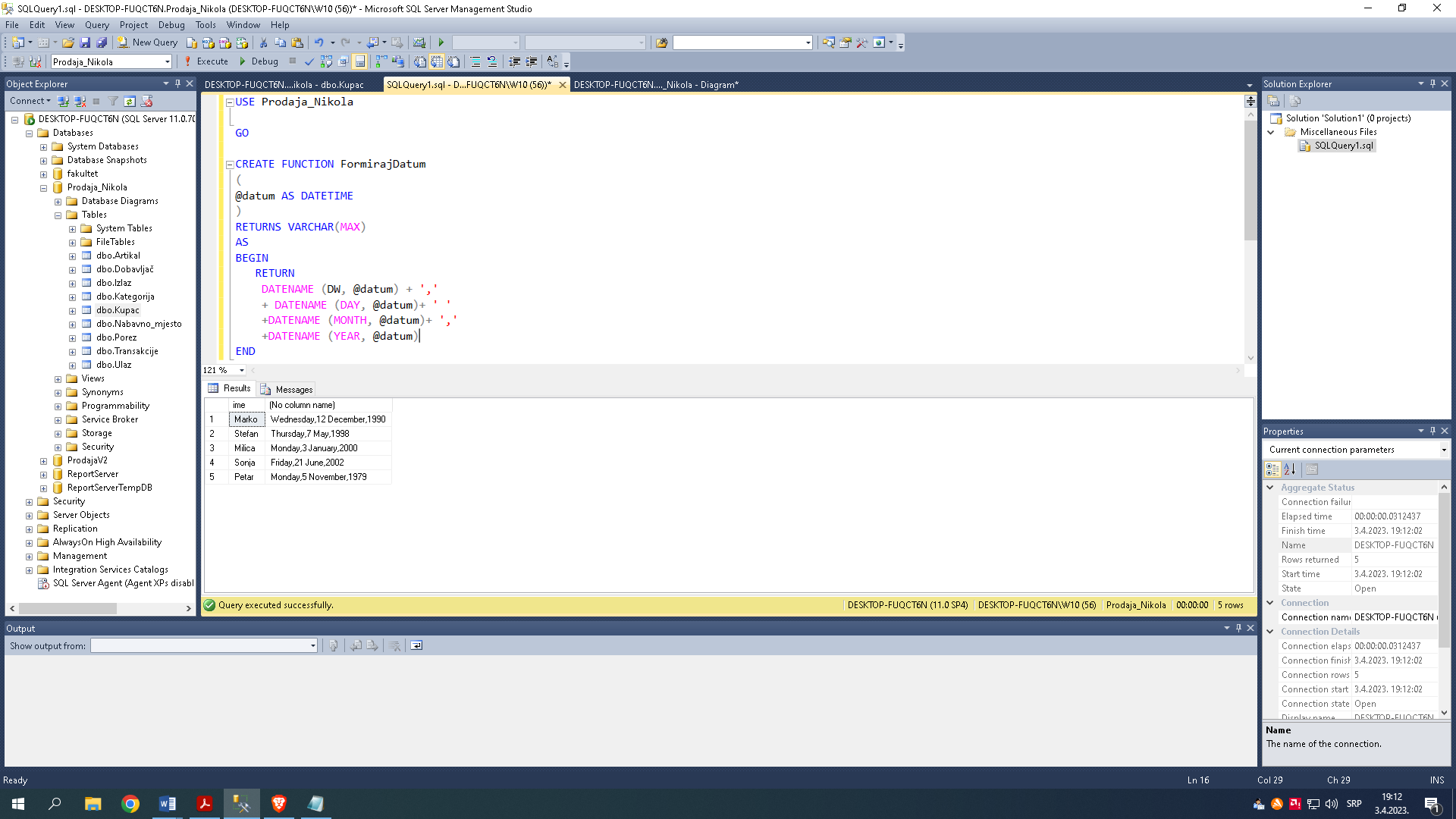
# **5. UDF funkcije**

Ugrađene funkcije ne daju uvijek željene rezultate. Uzmimo funkciju DATENAME u prethodnom poglavlju kao primjer, iako može kreirati datume u različitim formatima, šta ako želimo dohvatiti datume u različitim formatima, a funkcija DATENAME ne podržava ovaj format.

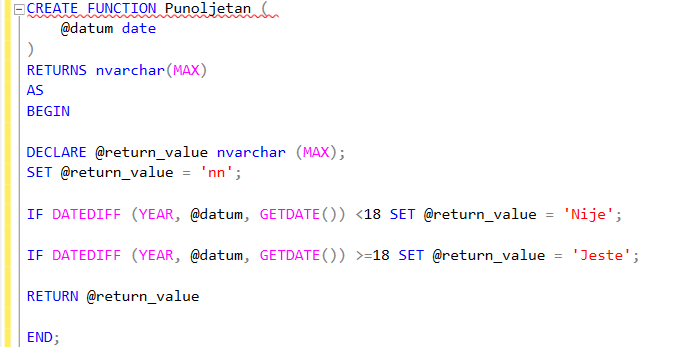
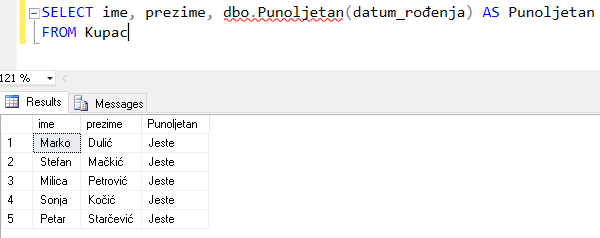
Na primjer, šta ako želimo da preuzmemo datum rođenja korisnika u formatu "srijeda, 23. maj 2003."? Ne postoji ugrađena funkcija za preuzimanje datuma rođenja u ovom formatu.

Da bismo to učinili, moramo više puta pozvati funkciju DATENAME i osloniti se na povezivanja nizova da bismo dohvatili datum u željenom formatu. Pogledajte skriptu ispod, ona preuzima datum u formatu o kojem smo upravo razgovarali, to je UDF (korisnički definirana funkcija).

Prvi primjer:

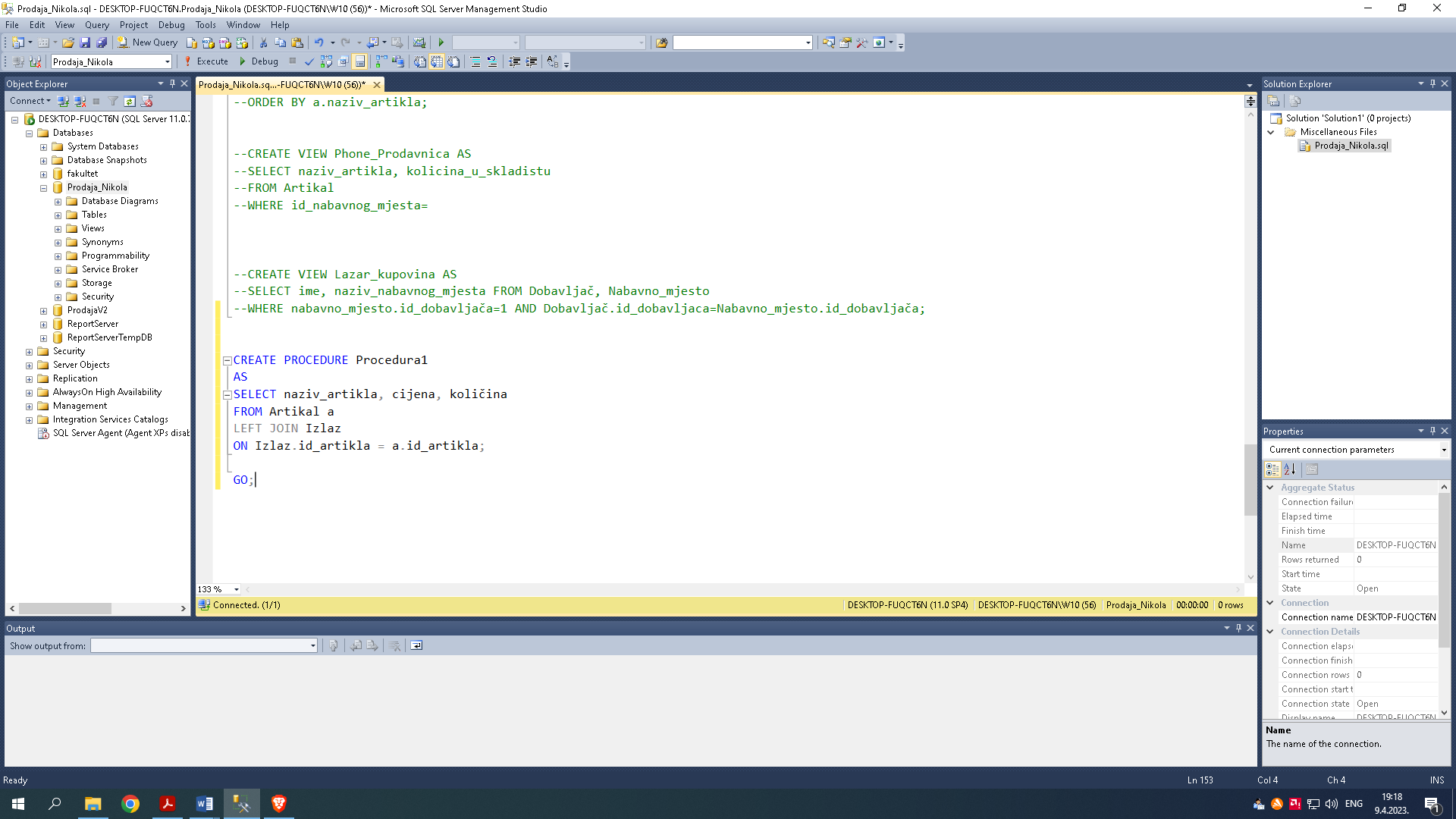


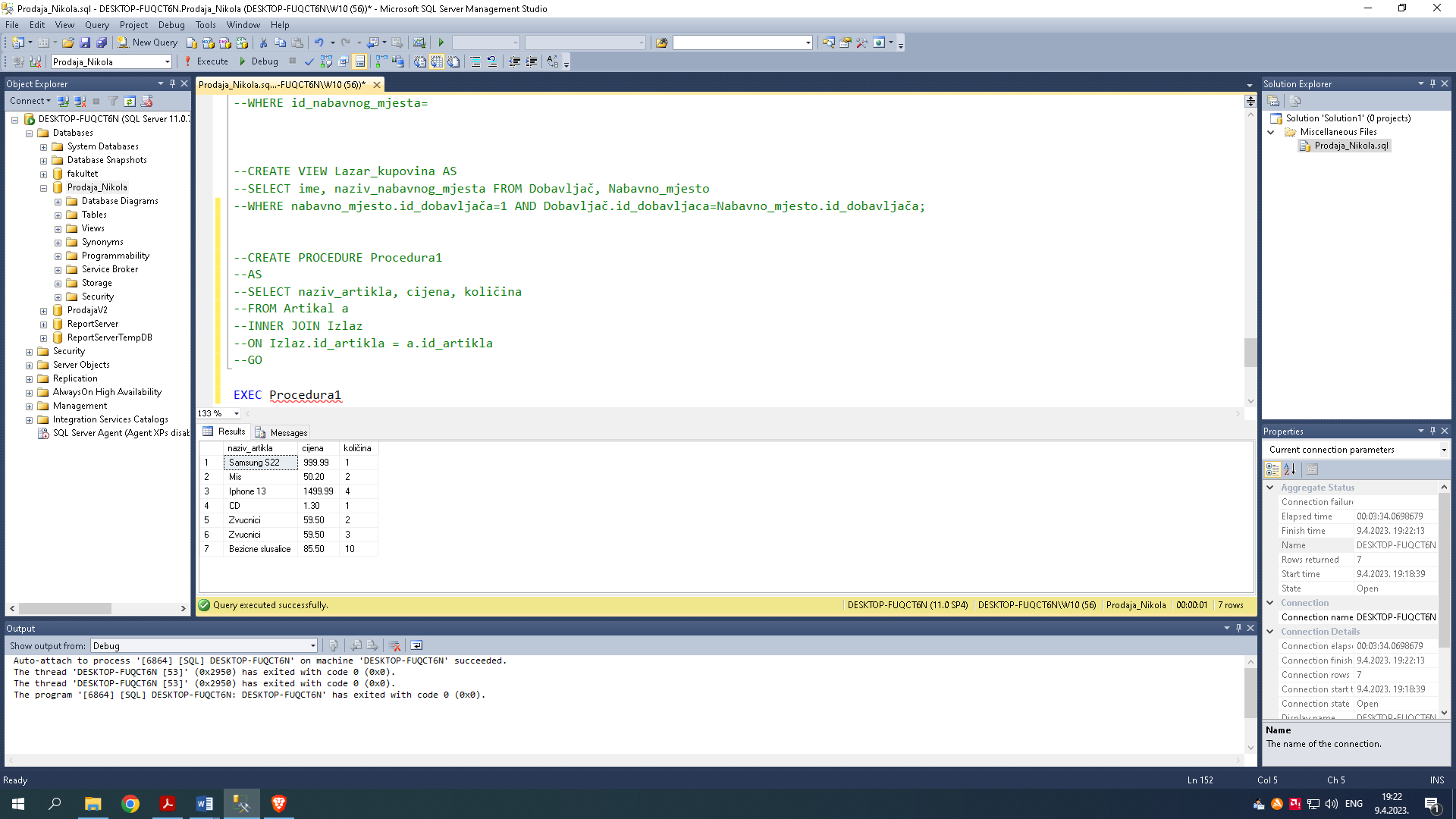
Drugi primjer je funkcija koja provjerava da li je kupac punoljetan.

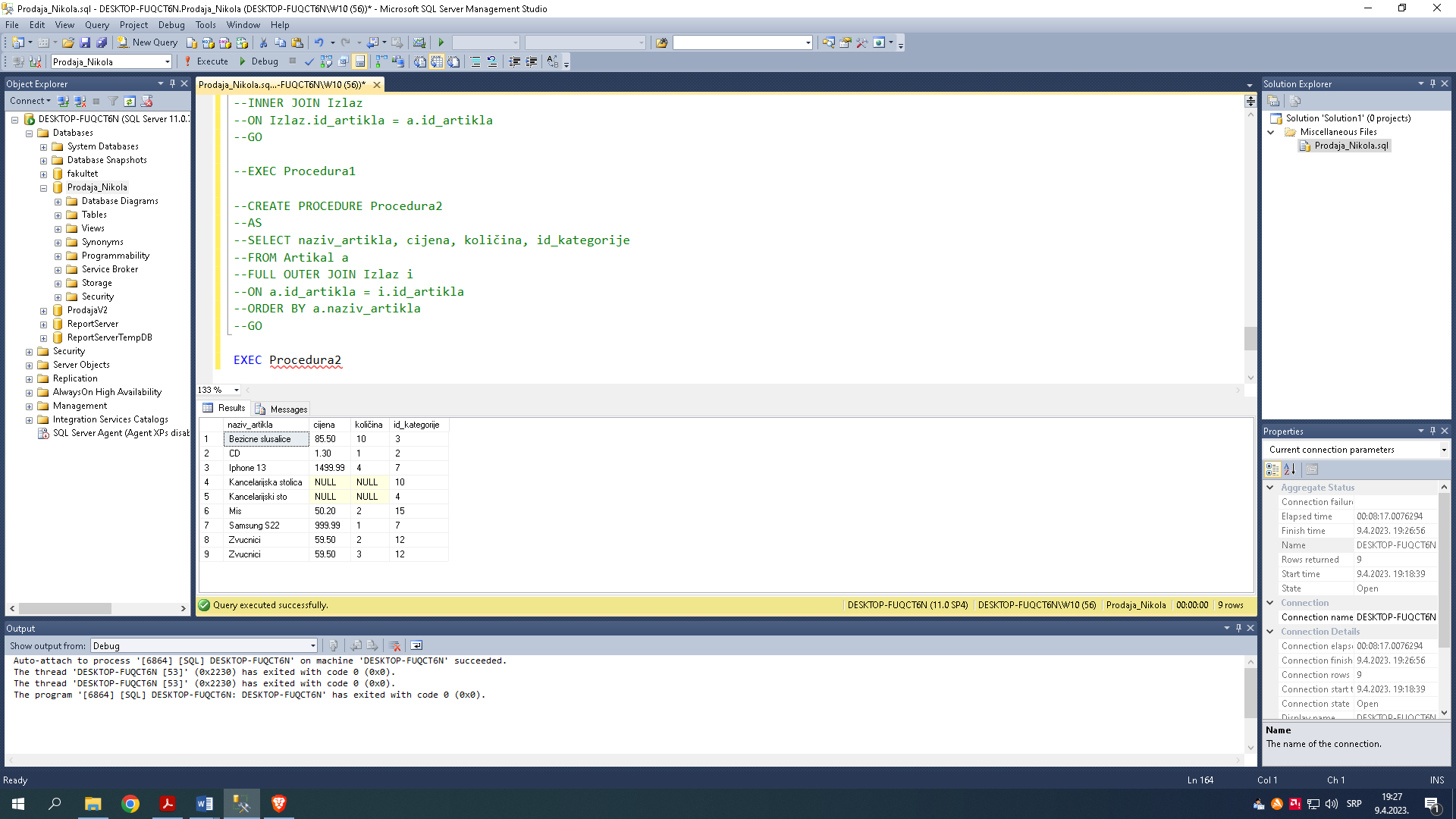


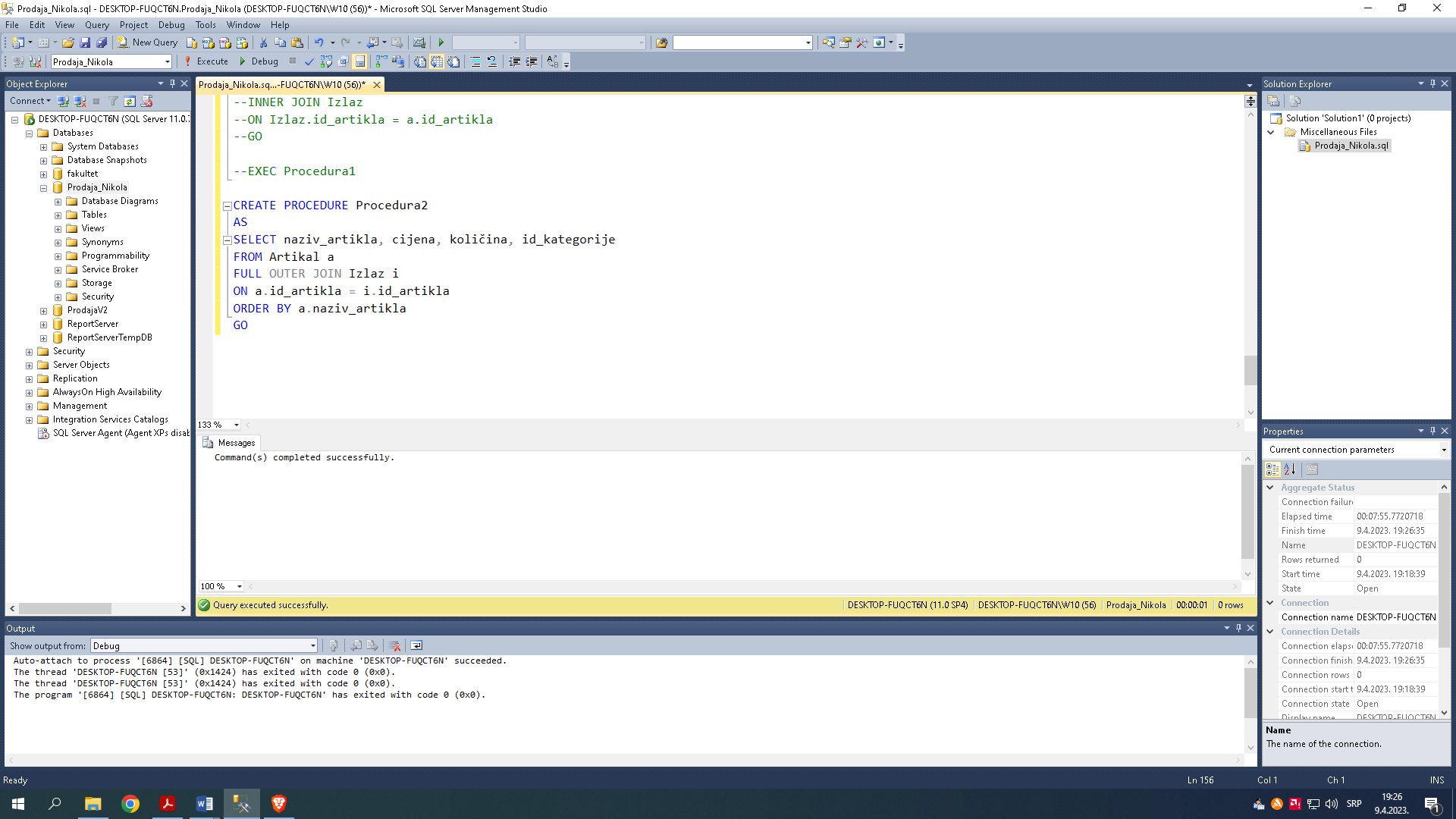
# **6. Procedure**

Uskladištena procedura je pripremljen SQL kod koji se može sačuvati, tako da se kod može ponovo i iznova koristiti. Dakle, ako imate SQL upit koji pišete iznova i iznova, sačuvajte ga kao uskladištenu proceduru, a zatim ga samo pozovite da biste ga izvršili. Takođe možete proslediti parametre uskladištenoj proceduri, tako da uskladištena procedura može da deluje na osnovu vrjednosti parametra koje se prosljeđuju.

Prvi primjer:



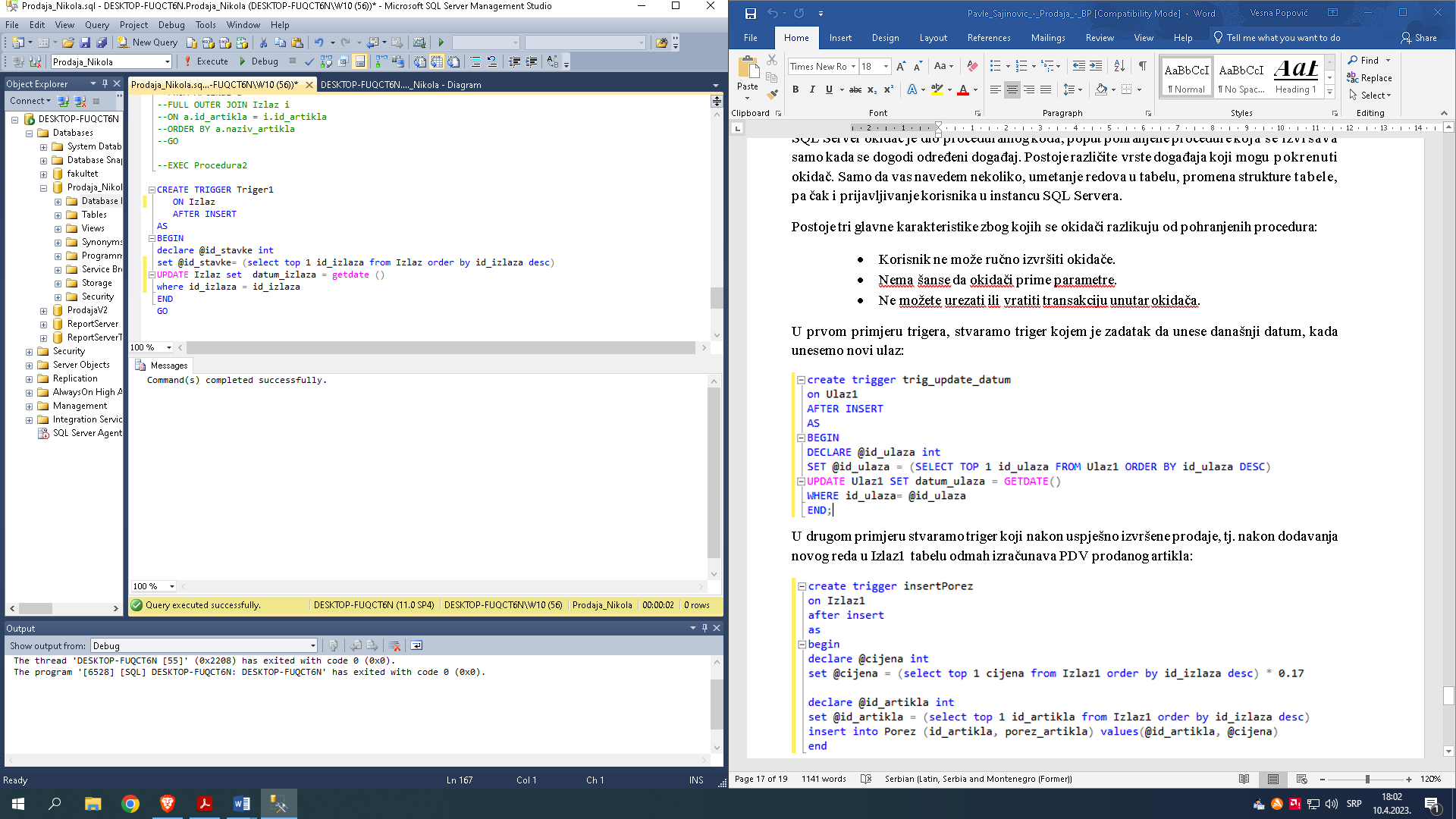
Drugi primjer:



# **7. Trigeri**

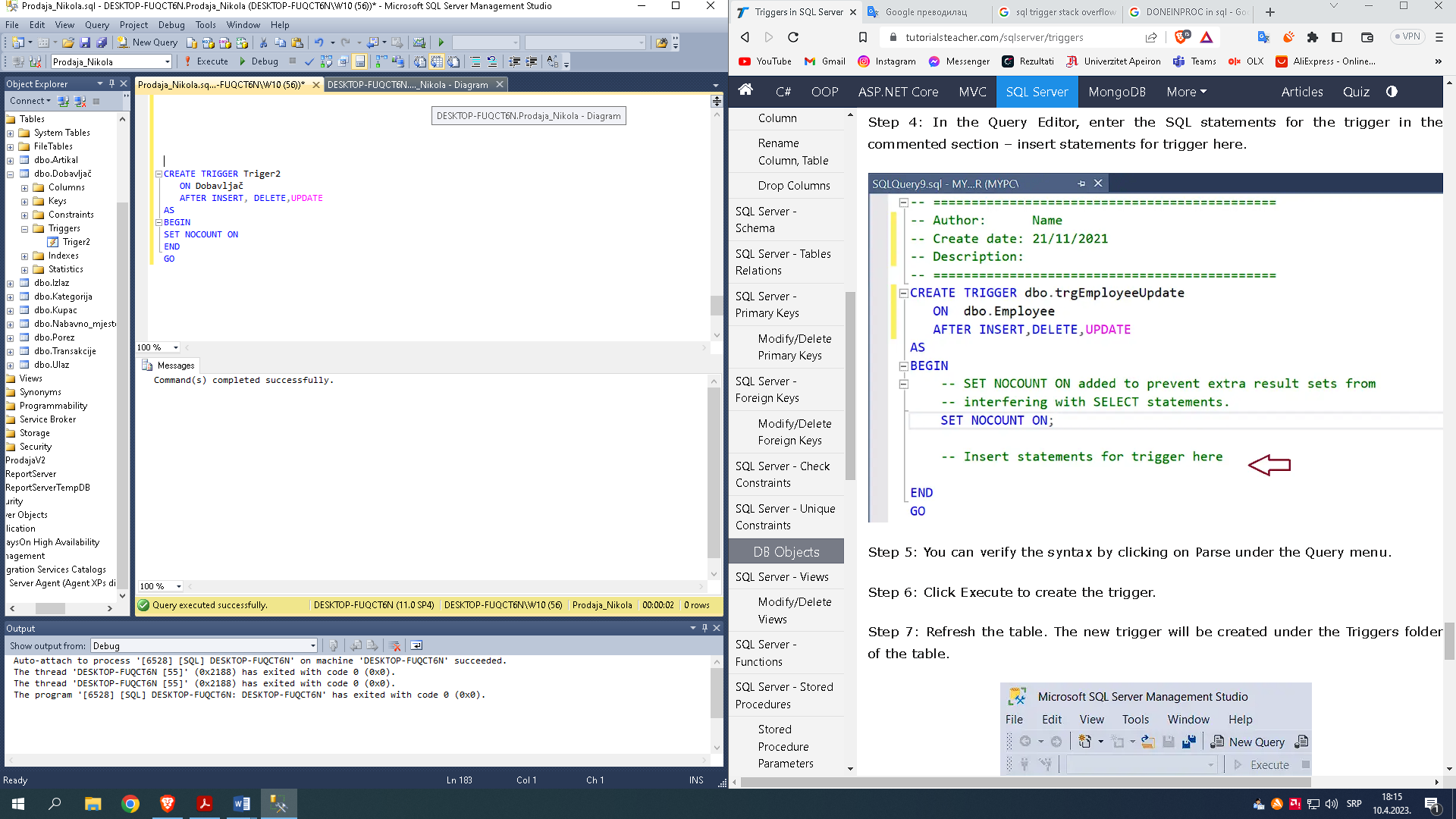
Naredba CREATE TRIGGER omogućava kreiranje novog okidača koji se automatski pokreće svaki put kada se desi događaj. Funkciju trigger mogu pokrenuti različiti događaji kao što su naredbe: INSERT, DELETE, UPDATE itd.

U prvom primjeru pravimo triger koji unosi trenutni datum nakon novog unosa.





Drugi primjer će da sprečava slanje svake izvršene SQL naredbe (DONEINPROC), tj. poruke klijentu za svaku naredbu u uskladištenoj proceduri

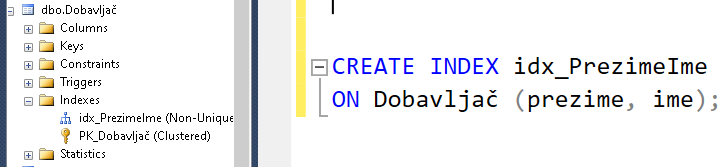


# **8. Index**

Naredba CREATE INDEX se koristi za kreiranje indeksa u tabelama.

Indeksi se koriste za preuzimanje podataka iz baze podataka brže nego inače. Korisnici ne mogu da vide indekse, oni se samo koriste za ubrzavanje pretraga/upita.

Ažuriranje tabele sa indeksima zahtjeva više vremena nego ažuriranje tabele bez (jer je i indeksima potrebno ažuriranje). Indeksi se kreiraju samo za kolone po kojima će se često pretraživati.

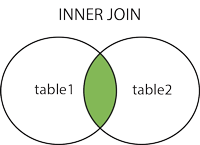
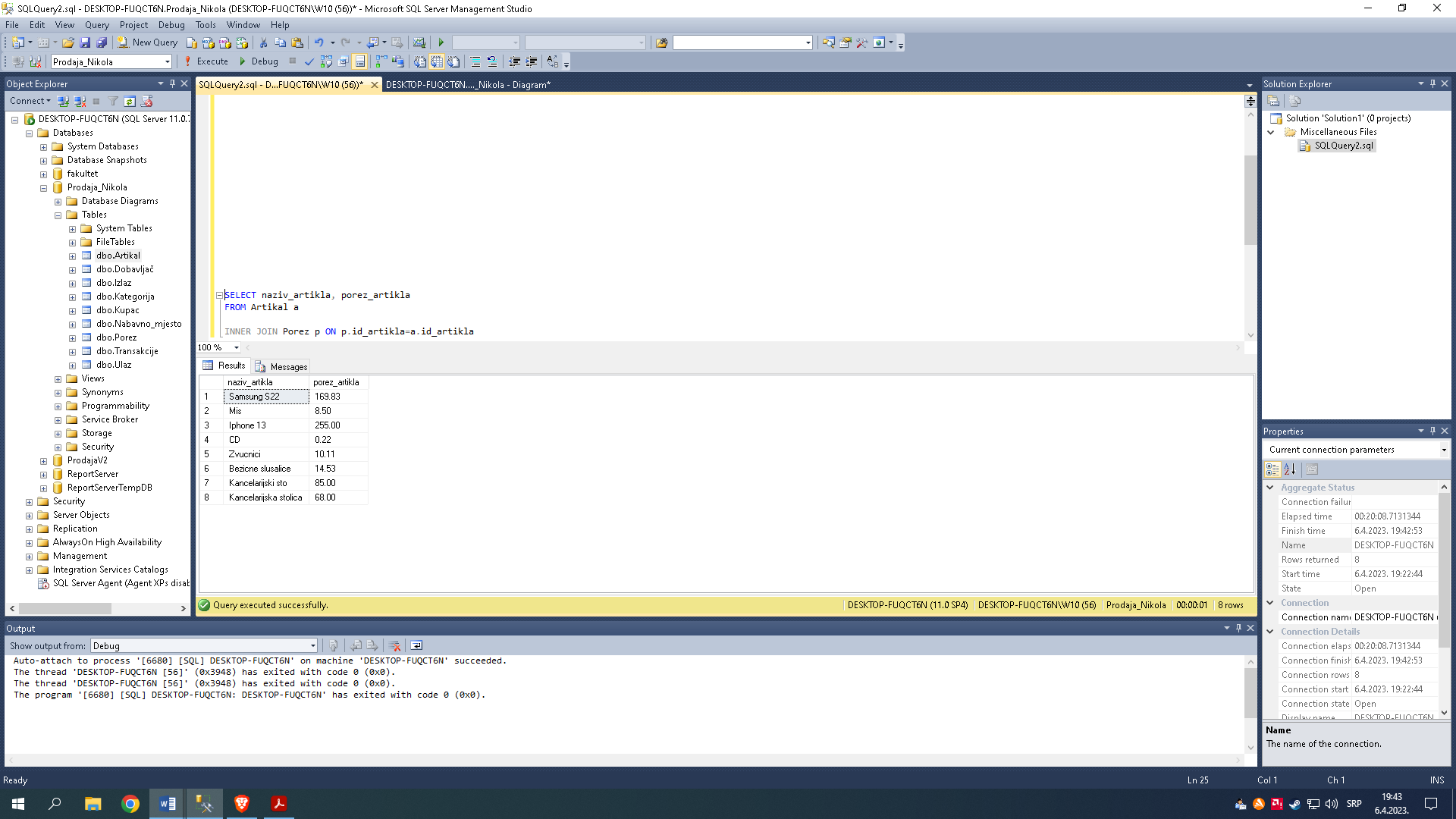


# **9. JOIN**

JOIN naredba se koristi za kombinovanje redova iz dve ili više tabela, na osnovu povezane kolone između njih.

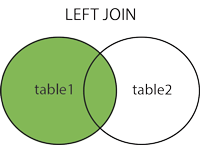
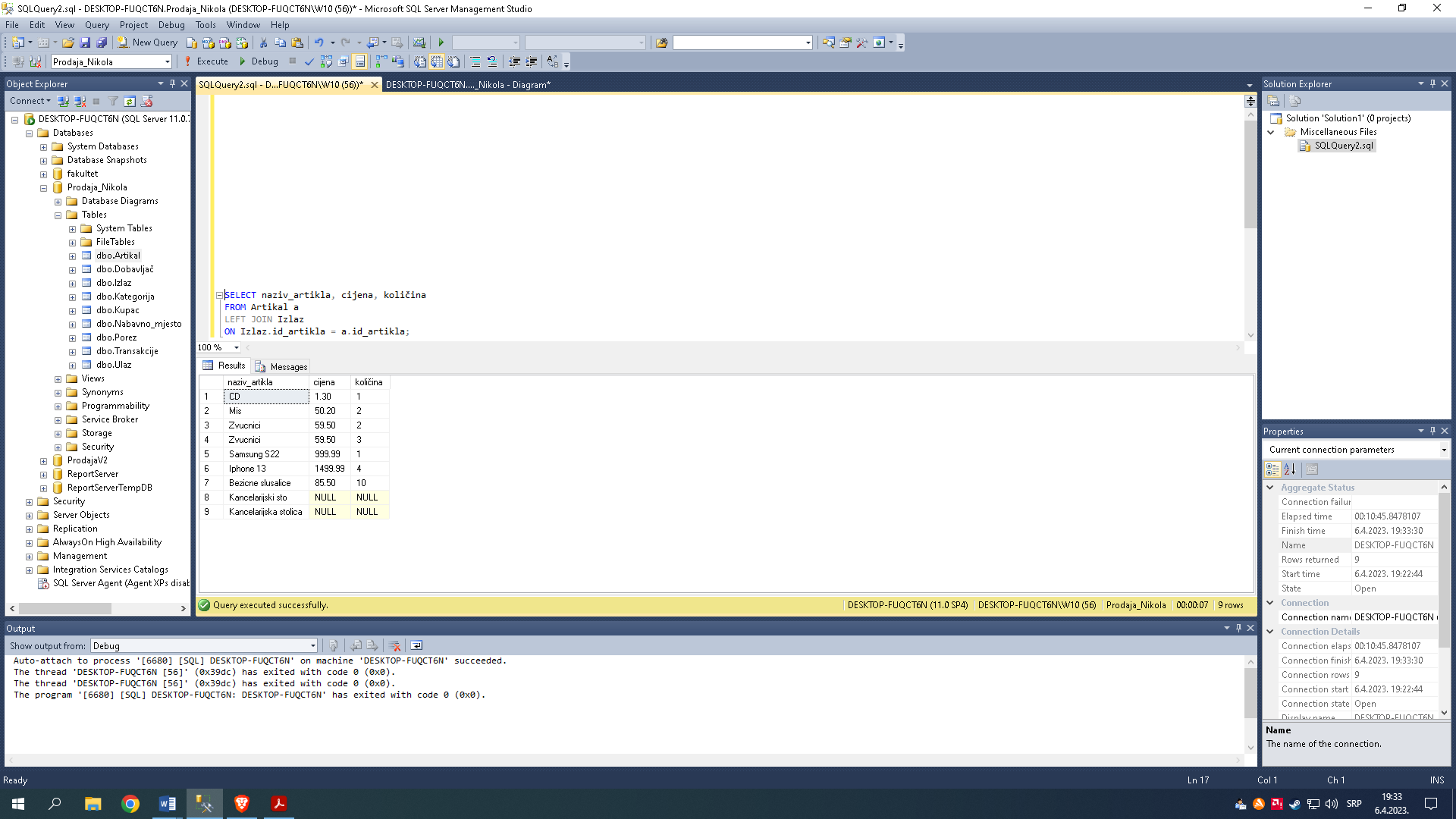
**INNER JOIN**

Ključna reč INNER JOIN bira zapise koji imaju odgovarajuće vrednosti u obe tabele.



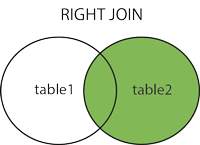
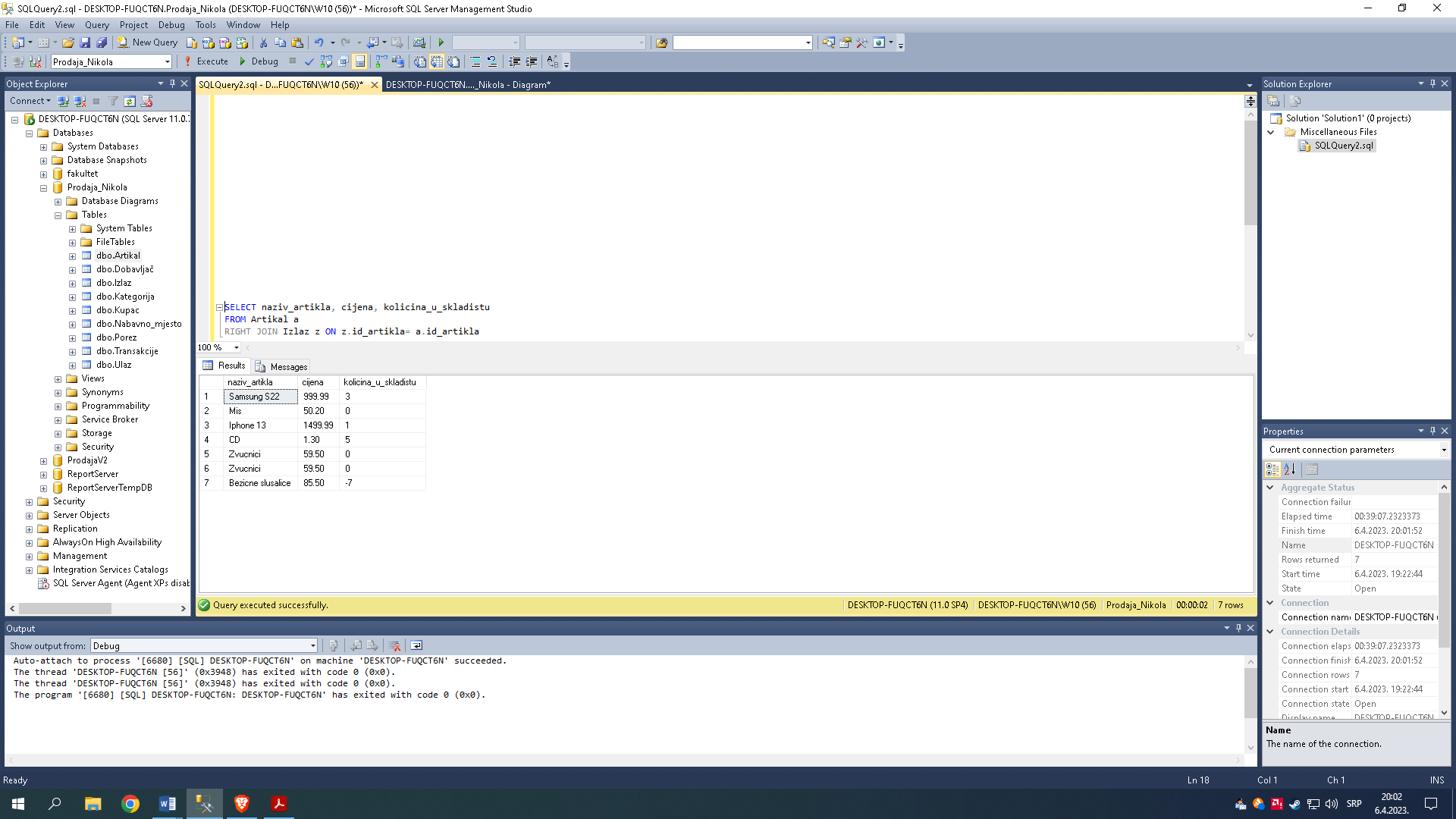
**LEFT JOIN**

Ključna reč LEFT JOIN vraća sve zapise iz leve tabele (tabela1) i odgovarajuće zapise iz desne tabele (tabela2). Rezultat je 0 zapisa sa desne strane, ako nema podudaranja.



**RIGHT JOIN**

Ključna reč RIGHT JOIN vraća sve zapise iz desne tabele (tabela 2) i odgovarajuće zapise iz leve tabele (tabela 1). Rezultat je 0 zapisa sa leve strane, ako nema podudaranja.



**FULL JOIN**

Ključna reč FULL OUTER JOIN vraća sve zapise kada postoji podudaranje u levoj (tabela1) ili desnoj (tabela2) zapisima tabele.

