Maskiranje podataka u SQL Server bazi podataka

Seminarski rad iz predmeta
Sistemi za upravljanje bazama podataka
na master akadademskim studijama

Mentor:
dr Aleksandar Stanimirović

Student:
Arsenije Arsenijević
br. ind. 793

Maj 2020.

Sadržaj

1. Uvod

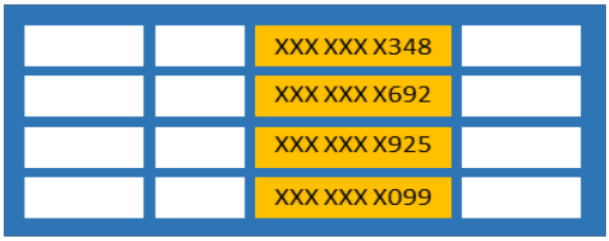
Veoma je često potrebno držati osetljive podatke u bazi podataka na produkciji, ali ne dati svima pristup tim podacima. U velikom broju slučajeva moguće je zabraniti neovlašćeni pristup podacima ograničavanjem pristupa korisnicima (eng. User) na određeni podset celokupne baze. Ali u nekim slučajevima potrebno je dati korisnicima pristup podacima iz podskupa kolona jedne tabele, ili čak podskupu podataka u jednoj koloni u tabeli. Primer takve potrebe je kada je potrebno da radnici korisničkog servisa mogu identifikovati korisnika na osnovu poslednjih brojeva kreditne kartice, ili poslednjih brojeva broja socijalnog osiguranja. Standardni pristupi obezbeđenja baza podataka (eng. Database Security) ne mogu transparentno podržati ovakvo ponašanje. Tu u priču nastupa Dinamičko Maskiranje Podataka (eng. Dynamic Data Masking – *DDM u daljem tekstu*).

Još jedna potreba za uvidom u nepotpune podatke je sa strane tehničkog tima, kada je potrebno developerima dati za rad produkcione podatke. Šta ako su developeri *outsource* timovi kojima kompanija ne želi dati potpuni uvid u produkcionu bazu, ali je ipak potrebno da imaju strukturu baze podataka sa realnim brojem podataka u *DEV, TEST, QA* okruženjima? Za takve potrebe Microsoft je predstavio Statičko Maskiranje Podataka (eng. Static Data Masking – *SDM u daljem tekstu*).

2. Dinamičko maskiranje podataka – DDM

Dinamičko maskiranje podataka (eng. Dynamic Data Masking – *DDM*) MS SQL Server baze podataka je predstavljeno u SQL Server 2016 (13.x) verziji i *Azure* SQL Server, a dostupno u svakoj narednoj verziji.

DDM implemetira ideju obfuskiranja podataka iz postojeće baze podataka, kako bi se određenim klijentima/korisnicima prikazale postojeće funkcionalnosti na postojećem sistemu sa postojećim, realnim podacima. Samo je neophodno izbaciti/maskirati/lažirati sve osetljive podatke dobijene iz baze podataka, kao što su npr. nazivi klijenata, imena korisnika, adrese, brojevi telefona, druge vrste kontakata, kao i svi bitni brojčani podaci.



DDM ograničava pristup osetljivim podacima maskirajući ih za neprivilegovane korisnike. Može se koristiti da umnogo uprosti dizajn i kodiranje obezbeđenja (eng. *security*) u bazi podataka i/ili aplikaciji.

DDM pomaže sprečavanju neautorizovanog pristupa osetljivim podacima tako što omogućava developerima ili administratorima baze podataka (eng. Database Administrator - *u daljem tekstu DBA*) da specificiraju koje delove osetljivih podataka otkriti sa minimalnim udarom na aplikacioni sloj (eng. Application Layer). Dinamičko maskiranje podataka može biti konfigurisano na određenim poljima u bazi podataka da bi sakrilo osetljive podatke u rezultatima upita. DDM ni na koji način ne utiče na same podatke koji se čuvaju, niti na njihov način čuvanja u memoriji. Dinamičko maskiranje podataka se lako koristi u postojećim aplikacijama, kako se pravila maskiranja primenjuju samo na rezultatima upita. Mnoge aplikacije mogu maskirati osetljive podatke bez promene postojećih upita.

Osobine DDM-a:

- Polisa centralnog maskiranja podataka deluje direktno na osetljiva polja u bazi podataka.
- Moguće je odrediti korisnike (eng. Users) ili uloge (eng. Roles) koji imaju pristup osetljivim podacima.
- Postoje potpuno i delimično/parcijalno maskiranje podataka i nasumična (eng. Random) maska za numeričke podatke.
- Jednostavne T-SQL komande definišu i organizuju maske.

Kao na primer, osoba koja radi na uslužnoj pozivnoj podršci korisnicima (eng. *Call Center*) može da identifikuje pozivaoce po samo nekoliko cifara njihovog broja socijalnog osiguranja ili broja kreditne/debitne kartice. Brojevi socijalnog osiguranja i kreditnih kartica ne trebaju biti potpuno prikazani osobi koja pruža podršku. Pravilo maskiranja može biti definisano tako da se maskiraju sve osim poslednjih četiri cifara broja socijalnog osiguranja ili kreditne kartice u rezultatu upita. Drugi primer bi bio da korišćenjem odgovarajuće maske, programer (eng. *Developer*) može pristupiti realnim podacima na produkciji bez otkrivanja/saznavanja osetljivih podataka klijenta, i tako odradi posao koji treba odraditi bez ugrožavanja privatnosti podataka klijenta i kršenja određenih regulativa.

Svrha DDM-a je da se ograniči pristupačnost osetljivih podataka, sprečavajući korisnike koji nemaju pristup podacima da ih čitaju. DDM ne cilja ka tome da spreči korisnike baze podataka da se konektuju direktno na bazu i pokreću iscrpne upite čiji rezultati sadrže osetljive podatke. DDM je komplementaran sa drugim mogućnostima obezbeđivanja SQL Server baza podataka (*auditing,*

encryption, *row level security*...) i preporučljivo je koristiti ovu fukncionalnost zajedno sa njima kako bi se bolje obezbedili osetljivi podaci u bazi podataka.

1.1 Definisanje dinamičkog maskiranja podataka

Pravilo maskiranja se može definisati na koloni unutar tabele da bi se obfuskirali podaci u toj koloni. Postoje 4 tipa dostupnih maski.

Funkcija	Opis	Korišćenje
Default	Potpuno maskiranje podataka prema tipu podataka određenog polja. Za *string* tipove podataka koristi se *XXXX* ili manje *X*-eva ako veličina polja ima manje od 4 karaktera (***char, nchar, varchar, nvarchar, text, ntext***). Za numeričke tipove podataka se koriste nulte vrednosti (bigint, bit, decimal, int, money, numeric, smallint, smallmoney, tinyint, float, real). Za datumske tipove podataka se koristi 01.01.1900 00:00:00.0000000 (***date, datetime2, datetime, datetimeoffset, smalldate, time***). Za binarne tipove podataka se koristi jedan bajt (eng. *byte*) ASCII vrednosti 0 (***binary, varbinary, image***).	Definicija: Phone# varchar(12) MASKED WITH (FUNCTION = 'default()') NULL Promena: ALTER COLUMN Gender ADD MASKED WITH (FUNCTION = 'default()')
Email	Ova metoda maskiranja eksponira samo prvo slovo e-mail adrese, ostali deo sakriva sa XXX@XXXX.com uz obavezan „.com“ domen (*aXXX@XXXX.com*)	Definicija: Email varchar(100) MASKED WITH (FUNCTION = 'email()') NULL Promena: ALTER COLUMN Email ADD MASKED WITH (FUNCTION = 'email()')
Random	*Random* (nasumično) maskiranje se koristi na bilo koji numerički tip vrednosti i maskira vrednost tako što vraća nasumične vrednosti unutar datog opsega.	Definicija: Account_Number bigint MASKED WITH (FUNCTION = 'random([start range], [end range])') Promena:

		ALTER COLUMN [Month] ADD MASKED WITH (FUNCTION = 'random(1, 12)')
Custom String	Metoda maskiranja koja eksponira prve i poslednje karaktere i dodaje određeni centralni deo stringa. prefix,[padding],suffix Primetiti: Ako je originalna vrednost previše kratka da bi se kompletirala cela maska, delovi prefiksa i sufiksa se izostavljaju i prikazuje se samo zadati fiksni srednji deo maske.	Definicija: FirstName varchar(100) MASKED WITH (FUNCTION = 'partial(prefix,[padding],suffix)') NULL Promena: ALTER COLUMN [Phone Number] ADD MASKED WITH (FUNCTION = 'partial(1,"XXXXXXX",0)')

1.2 Dozvole (eng. Permissions)

Nisu potrebne nikakve specijalne dozvole da bi se kreirala tabela sa dinamičkom maskom, samo standardne **CREATE TABLE** i **ALTER** dozvole na toj šemi (eng. *Schema*).

Dodavanje, zamena ili uklanjanje maske zahteva **ALTER ANY MASK** dozvolu i **ALTER** dozvolu na konkretnu tabelu. Prikladno je dati **ALTER ANY MASK** dozvolu osobama koje su zadužene za sigurnost baze podataka.

Korisnik sa **SELECT** dozvolom nad nekom tabelom može da vidi podatke u toj tabeli. Podaci kolona koje su definisane kao maskirane će se prikazati maskirani. Moguće je dati **UNMASK** dozvolu korisniku da bi mu se odobrilo da vidi maskirane podatke kao nemaskirane u kolonama za koje su definisane maske.

Dozvola **CONTROL** na bazi podataka uključuje i **ALTER ANY MASK** i **UNMASK** dozvole.

1.3 Dobra praksa i česti slučajevi upotrebe

- Kreiranje maske nad kolonom ne sprečava ažuriranje (eng. *Update*) nad tom kolonom. Tako da iako korisnici dobijaju maskirane podatke u upitima koji sadrže maskirane kolone, isti korisnici mogu da ažuriraju podatke ako imaju dozvolu pisanja (eng. *Write*). Pravilna polisa kontrole pristupa može da se iskoristi da se ograniče dozvole ažuriranja.
- Korišćenje ***SELECT INTO*** ili ***INSERT INTO*** za kopiranje podataka iz tabele sa maskiranim kolonama u drugu tabelu rezultira time da su podaci u ciljanoj tabeli upisani maskirani.
- Dinamičko maskiranje podataka se koristi pri pokretanju *SQL Server Import* ili *Export* funkcionalnosti. Baza podataka koja sadrži maskirane kolone će rezultirati da izvezeni (eng. *Exported*) podaci budu maskirani (pod pretpostavkom da je *Export* rađen od strane korisnika bez **UNMASK** privilegije) i uvezena (eng. *Imported*) baza podataka će sadržati statički maskirane podatke (trajno izmenjene).

1.4 Upiti za izlistavanje maskiranih kolona

Koristiti **sys.masked_columns** pogled (eng. *View*) da bi se dobile informacije o tabelama i njihovim kolonama koje imaju nalepljene funkcije maskiranja na sebi. Ovaj pogled nasleđuje **sys.columns** pogled. Vraća sve kolone u **sys.columns** pogledu, plus **is_masked** i **masking_function** kolone, koje indiciraju da li je kolona maskirana i ako jeste, koja je funkcija maske definisana nad istom. Ovaj pogled samo pokazuje kolone nad kojima jesu definisane funkcije maskiranja.

Upit koji vraća kolone po tabelama koje su maskirane - SQL

```
SELECT c.name, tbl.name as table_name, c.is_masked, c.masking_function
FROM sys.masked_columns AS c
JOIN sys.tables AS tbl
   ON c.[object_id] = tbl.[object_id]
WHERE is_masked = 1;
```

1.5 Ograničenja i zabrane (eng. Limitations and Restrictions)

Pravilo maskiranja ne može biti definisano za sledeće tipove kolona:

- Enkriptovane kolone (eng. Encripted columns, **Always Encrypted**)
- FILESTREAM
- **COLUMN_SET** ili delimična kolona koja je deo seta kolona
- Maska ne može biti konfigurisana nad obračunatim kolonama (eng. Computed columns), ali ako obračunata kolona zavisi od kolone koja ima definisanu masku, onda i preračunata kolona vraća maskirane podatke.
- Kolona sa maskiranim podacima ne može biti ključ za (eng.) **FULLTEXT** indeks (eng. Index).

Za korisnike koji nemaju **UNMASK** dozvolu, prevaziđene (eng. Deprecated) **READTEXT, UPDATETEXT** i **WRITETEXT** izjave ne funkcionišu pravilno nad kolonom kojoj je konfigurisana dinamička maska.

Dodavanje dinamičke maske je implementirano kao promena šeme (eng. Schema) nad tabelom ispod, i zato ne može biti izvedeno na kolonu sa zavisnostima (eng. Dependency). Da bi se zaobišla ova restrikcija, može se prvo skloniti zavisnost, zatim dodati dinamička maska i onda rekreirati zavisnost. Na primer, ako postoji zavisnost zbog indeksa nad tom kolonom, može se **DROP**-ovati indeks, zatim dodati maska i na kraju rekreirati pomenuti indeks.

1.6 Sigurnosno zapažanje: Zaobilaženje maskiranja

DDM je dizajniran da uprosti razvoj aplikacija tako što ograničava izloženost podataka u setu predefinisanih upita (eng. Query) koje aplikacija koristi. Dok DDM takođe može biti koristan da preventivno zaustavi slučajnu izloženost osetljivih podataka kada se pristupa bazi podataka na produkciji, važno je upamtiti da neprivilegovani korisnici sa ad-hoc dozvolama upita mogu da

iskoriste neke tehnike koje bi im dale pristup pravim podacima. Ako postoji potreba da se dozvoli takav ad-hoc pristup, auditiranje (eng. *Auditing*) se treba koristiti da bi se pratile aktivnosti nad bazom podataka i presreo ovakav scenario.

Kao na primer, razmatrajmo korisnika baze podataka koji ima dovoljno privilegija da pokreće ad-hoc upite nad bazom podataka, i pokušava da ,,pogodi'' podatke koji su maskirani i na kraju inferencijom dolazi do pravih vrednosti. Pretpostavimo da imamo masku nad *[Employee].[Salary]* kolonom, i da se korisnik konektuje direktno na bazu podataka i počinje da pogađa vrednosti i vremenom dolazi do prave vrednosti *[Salary]* nad setom podataka *[Employee]:*

SQL
SELECT ID, Name, Salary FROM Employees WHERE Salary > 99999 and Salary < 100001;

ID	Name	Salary
64543	Jane Doe	0
91245	John Smith	0

Ovo pokazuje da se DDM ne treba koristiti kao jedina mera zaštite da bi se osigurali osetljivi podaci od korisnika koji mogu pokretati ad-hoc upite nad bazom podataka. Pogodno je kao zaštita od slučajnih izliva osetljivih podataka, ali neće zaštititi podatke od zlonamernog pokušaja naslućivanja podataka.

Važno je pravilno rukovati dozvolama u bazi podataka i praćenje pravila dodele minimalnih dozvola potrebnih korisniku. Takođe, korisno je upaliti auditiranje kako bi se pratile sve aktivnosti koje se dešavaju nad bazom podataka.

1.7 Primeri

1.7.1 Kreiranje dinamičke maske podataka

Sledeći primer pokazuje kreiranje tabele sa definisanim tri različitih vrsti dinamičkih maski. Primer ispunjava tabelu i upitom selektuje podatke iz rezultata.

SQL

```
CREATE TABLE Membership
  (MemberID int IDENTITY PRIMARY KEY,
   FirstName varchar(100) MASKED WITH (FUNCTION = 'partial(1,"XXXXXXX",0)') NULL,
   LastName varchar(100) NOT NULL,
   Phone varchar(12) MASKED WITH (FUNCTION = 'default()') NULL,
   Email varchar(100) MASKED WITH (FUNCTION = 'email()') NULL);
INSERT Membership (FirstName, LastName, Phone, Email) VALUES
('Roberto', 'Tamburello', '555.123.4567', 'RTamburello@contoso.com'),
('Janice', 'Galvin', '555.123.4568', 'JGalvin@contoso.com.co'),
('Zheng', 'Mu', '555.123.4569', 'ZMu@contoso.net');
SELECT * FROM Membership;
```

Novi korisnik se kreira i dobija **SELECT** dozvolu nad tabelom. Upiti izvršeni kao *TestUser* vraćaju maskirane podatke.

SQL

```
CREATE USER TestUser WITHOUT LOGIN;
GRANT SELECT ON Membership TO TestUser;


EXECUTE AS USER = 'TestUser';
SELECT * FROM Membership;
REVERT;
```

Primer demonstrira maske tako što su podaci izmenjeni iz:

MemberID	FirstName	LastName	Phone	Email
1	Roberto	Tamburello	555.123.4567	RTamburello@contoso.com

u

MemberID	FirstName	LastName	Phone	Email
1	RXXXXXXX	Tamburello	xxxx	RXXX@XXXX.com

1.7.2 Dodavanje ili izmena maske na postojećoj koloni

ALTER TABLE izjava (eng. *Statement*) se koristi da se doda maska na postojeću kolonu u tabeli, ili da se izmeni ta maska na toj koloni.

Sledeći primer dodaje funkciju maskiranja na *LastName* koloni:

```
SQL
ALTER TABLE Membership
ALTER COLUMN LastName ADD MASKED WITH (FUNCTION = 'partial(2,"XXX",0)');
```

Sledeći primer menja funkciju maskiranja na *LastName* koloni:

```
SQL
ALTER TABLE Membership
ALTER COLUMN LastName varchar(100) MASKED WITH (FUNCTION = 'default()');
```

1.7.3 Davanje dozvole za nemaskirani pogled na podatke

Davanje **UNMASK** dozvole dozvoljava korisniku *TestUser* da vidi nemaskirane podatke.

```
SQL
GRANT UNMASK TO TestUser;
EXECUTE AS USER = 'TestUser';
SELECT * FROM Membership;
REVERT;

-- Uklanjanje UNMASK dozvole
REVOKE UNMASK TO TestUser;
```

1.7.4 Skidanje maske sa kolone

Sledeća SQL izjava skida/briše masku sa *LastName* kolone kreirane u prethodnom primeru:

```
SQL
ALTER TABLE Membership
ALTER COLUMN LastName DROP MASKED;
```

[1]

3. Statičko maskiranje podataka - SDM

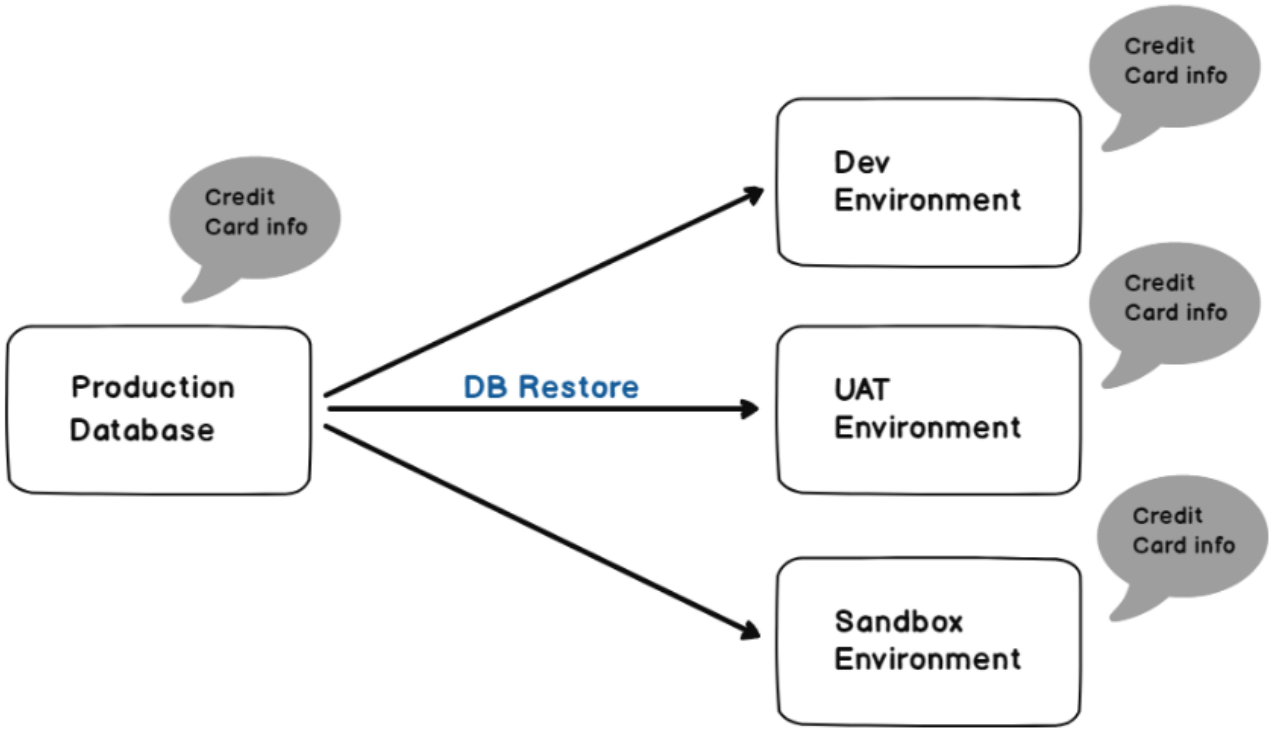
Dinamičko maskiranje podataka je pogodno kada je potrebno samo nekim korisnicima zabraniti pristup određenim podacima iz baze podataka. Ali kada je potrebno izmeniti podatke na smislen način i dati takve podatke određenim korisnicima/organizacijama/službama kao takve (odnosno kopira se cela baza na drugu lokaciju za te korisnike), više je smisleno izmeniti podatke samo jednom, a tu nastupa Statičko maskiranje podataka (eng. Static Data Masking – *u daljem tekstu SDM*).

Microsoft je predstavio ovu funkcionalnost u okviru SQL Server Management Studio (*u daljem tekstu SSMS*) 18.0 preview 5 verziji i moguće je koristiti nad bazama podataka SQL Server 2012 (11.x) i novijim. Iz ovoga vidimo da je SDM zapravo funkcionalnost okruženja koje se koristi, a ne konkretno SQL Server baze. Funkcionalnost je takodje dostupna u *Azure* SQL Server bazama podataka.

Napomena: Nažalost, funkcionalnost je izbačena iz konačne verzije 18 SSMS-a (ne preview). Nije dato nikakvo objašnjenje o tome zašto je izbačena iako je pretežno dobro istestirana od strane korisnika u preview verzijama. Pretpostavke su mnoge, od toga da Microsoft želi da korisnici još malo sačekaju na funkcionalnost iz marketinških razloga do toga da funkcionalnost još uvek nije bila dovoljno stabilna da bi Microsoft mogao da garantuje njenu upotrebnost. Mi ćemo ovde razmatrati funkcionalnost kakva se mogla videti i koristiti u okviru SSMS 18.0 preview 5 verziji i nadati funkcionalnosti kao takvoj ili boljoj u narednim verzijama.

Statičko maskiranje podataka je funkcionalnost preko potrebna DBA korisnicima kako bi dostavili podatke na niže prioritetna okruženja (DEV, TEST, QA) bez kršenja pravila i ugovora privatnosti podataka.

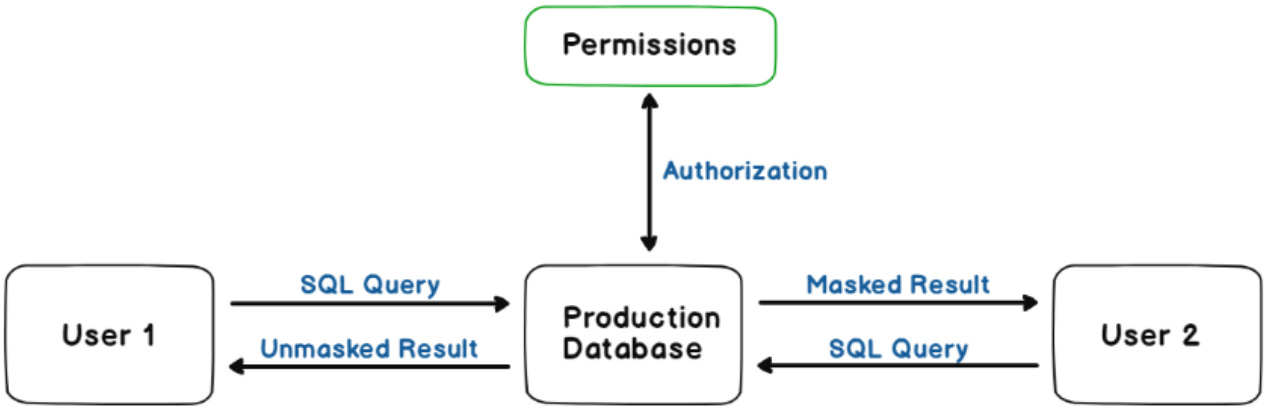
Ako bismo jednostavno kopirali bazu podataka na niža okruženja, dobili bismo situaciju koja se može videti na dijagramu (broj 1) ispod.



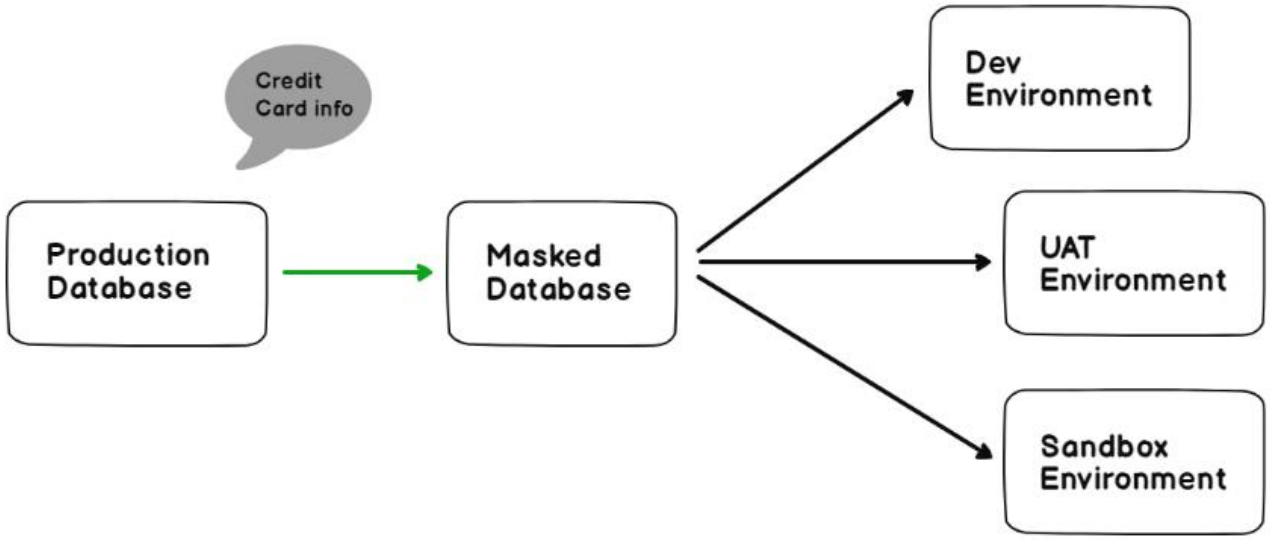
Statičko maskiranje podataka se okreće oko jednostavno razumljive paradigme. Naime, nalik dinamičkom maskiranju podataka, postoji nekoliko mogućih izbora kada je u pitanju konkretna maska nad kolonom, ali razlika je u tome što ovo (kao što ime i kaže), kod DDM-a se informacije maskiraju dinamički, odnosno, sama baza i dalje sadrži nemaskirane podatke. Manipulacijom

obezbeđenja SQL Server baze podataka (nad čime developeri pretežno imaju potpunu kontrolu u nižim okruženjima - *DEV, TEST, QA, UAT, Sandbox*) i dalje je moguće doći do pravih podataka.

Način rada dinamičkog maskiranja podataka je prikazan na dijagramu (broj 2) ispod:



SDM dozvoljava kopiranje baze podataka i izvršavanje operacija transformacija podataka nad tom kopijom. Nakon toga moguće je odraditi kreiranje rezervne kopije (eng. Backup) izmenjene baze i obnoviti je (eng. Restore) na nižim okruženjima. Dobijamo situaciju kao na dijagranu (broj 3) ispod:



Kao što na poslednjem dijagramu vidimo, nema osetljivih informacija (*Credit Card info*) na nižim okruženjima.

SDM se može koristiti u sledeće (i druge) svrhe:

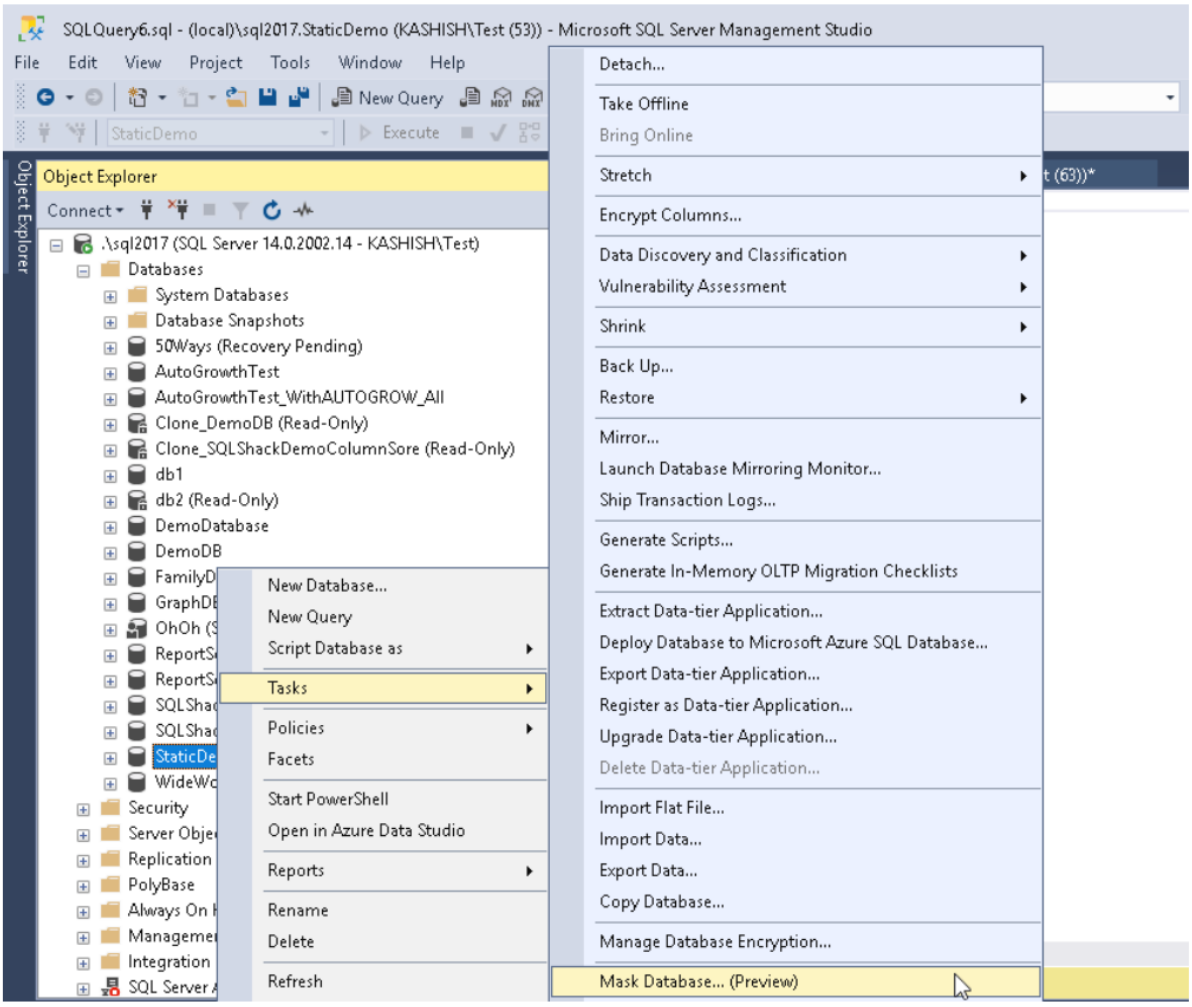
- Pripremanje nižih okruženja
- Razvijanje baze podataka
- Traženje uzroka problema u bazi podataka
- Deljenje podataka sa trećim licima
- ...

3.1. Korišćenje SDM-a

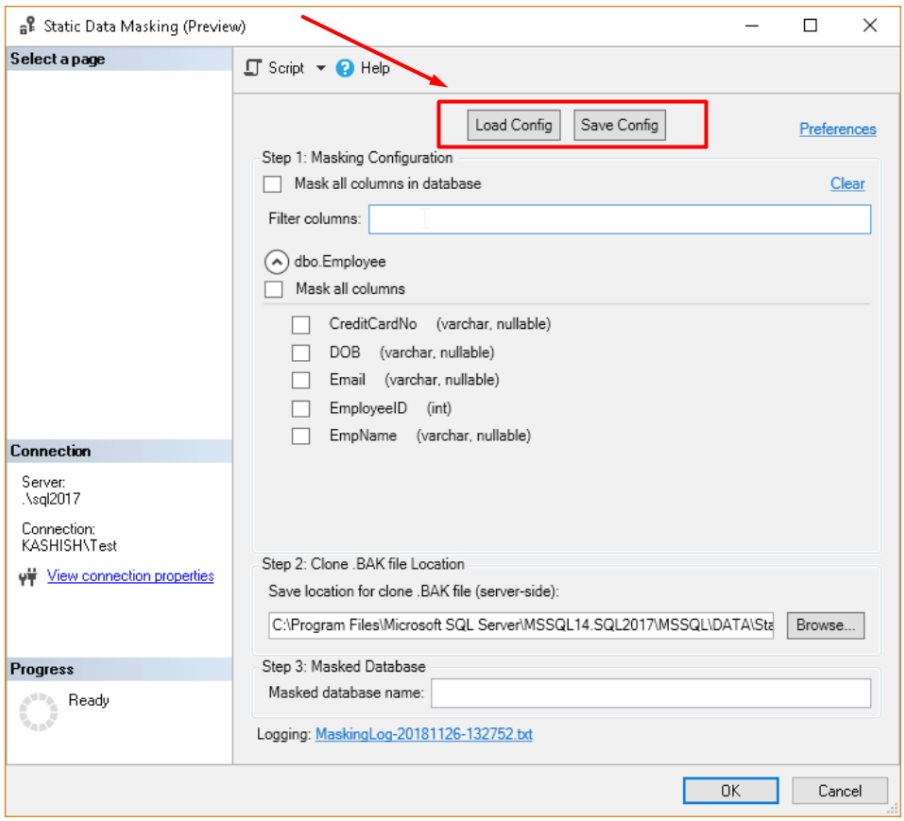
Trenutno je statičko maskiranje podataka moguće koristiti samo kroz SSMS, što znači da je statičko maskiranje podataka ustvari samo funkcionalnost radnog okruženja SSMS. Kada se malo razmisli, ralog je apsolutno logičan, upravo zato što se kreira apsolutno odvojena kopija baze sa kolonama tabela maskiranim iz originalne tabele. Zbog toga nije potrebno pisati nikakve upite, razmišljati o dodatnim osiguravanjima obezbeđenja, jer pravimo odvojenu kopiju sa direknto izmenjenim/maskiranim podacima.

Korišćenje je izuzetno jednostavno i intuitivno:

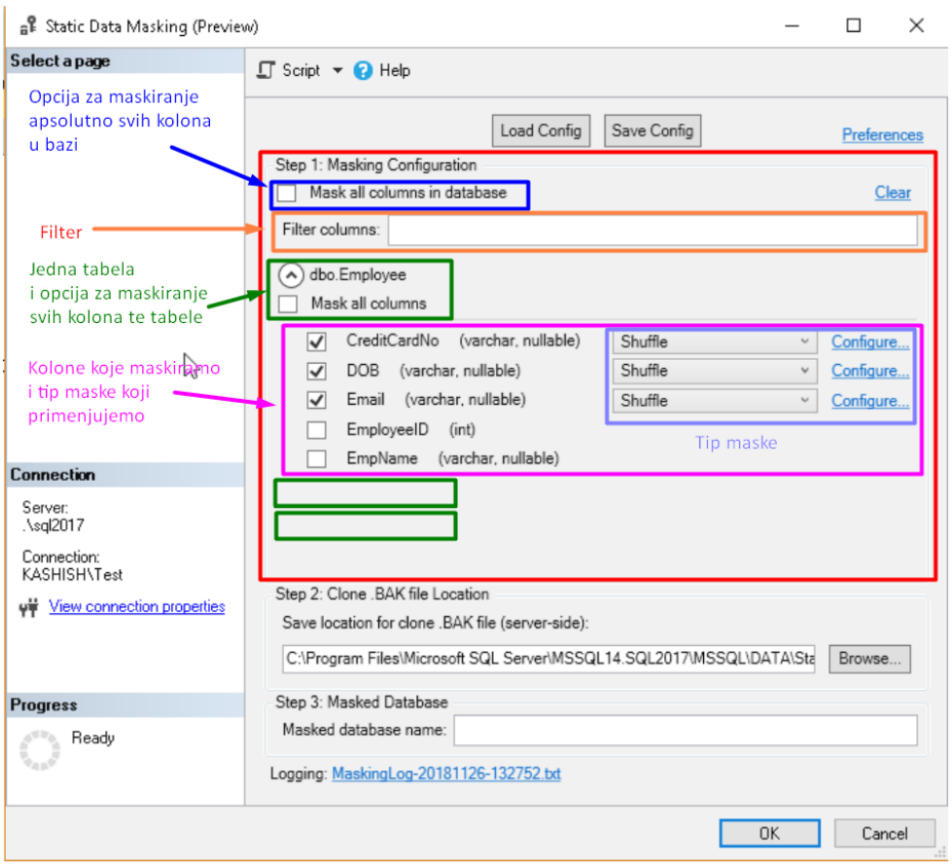
1. Odabrati bazu koju želimo statički maskirati.
2. Desni klik na bazu i odabrati opciju *Mask Database* u podmeniju *Tasks*



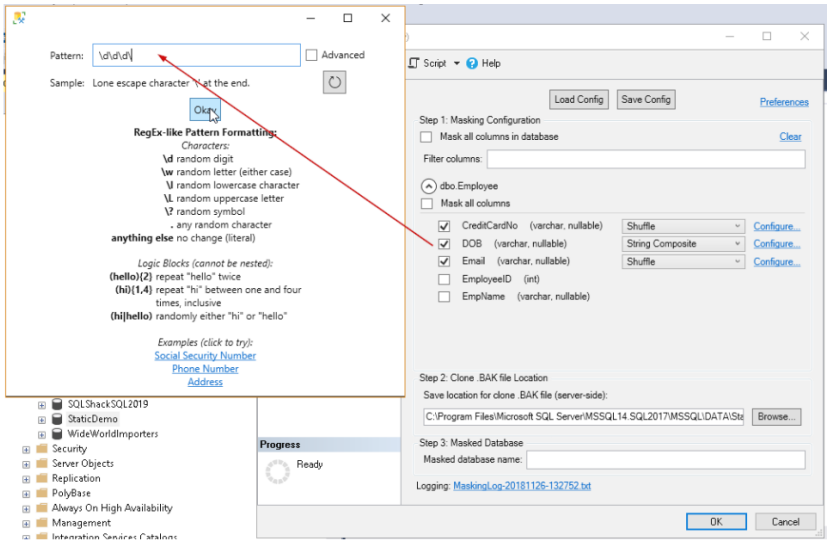
3. U meniju koji nam je prikazan imamo nekoliko stvari koje možemo uraditi. Prva je da sačuvamo konfiguraciju – *Save Config* – koju ćemo sada da postavimo (ovo je ako smo već postavili pa za buduću upotrebu). Pored te opcije je opcija da učitamo postojeću konfiguraciju – *Load Config* – koju smo sačuvali ranije preko prve pomenute opcije.



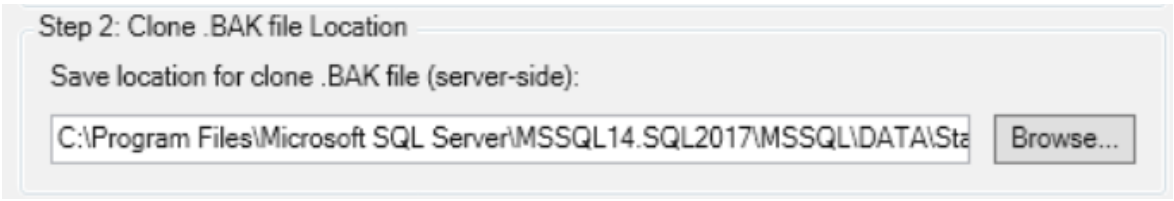
4. U delu ispod, nazvanom *Step 1: Masking Configuration* zapravo definišemo konfiguraciju maskiranja koju smo pomenuli ranije. Ovde biramo koje kolone u kojoj tabeli i na koji način želimo da maskiramo. Postoji i opcija filtriranja kolona radi lakšeg snalaženja (u slučaju prevelikog broja tabela i njihovih kolona).



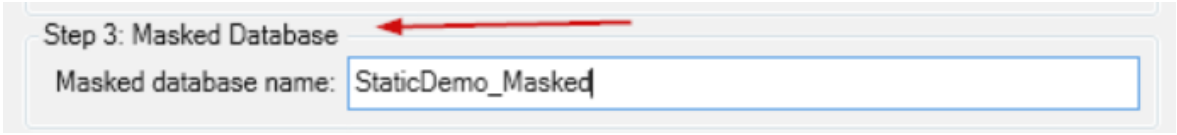
a. U slučaju da je moguće neko dodatno podešavanje kao u slučaju *String Composite* opcije, možemo pozvati pod-meni za ta podešavanja klikom na *Configure* labelu smeštenu pored opcije maske.



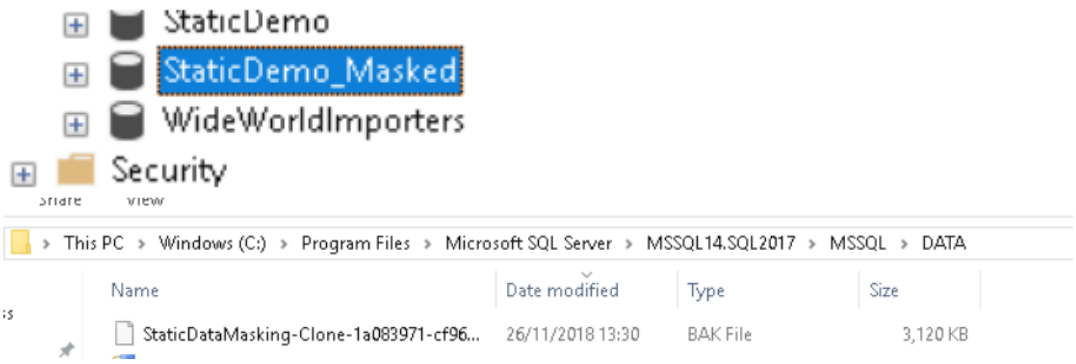
5. Sledeći deo je *Step 2: Clone .BAK file Location* koji je upravo to, lokacija na koju će se sačuvati .bak fajl za novokreirani klon baze.



6. Sledeći deo je biranje naziva nove baze, nazvan *Step 3: Masked Database*.



7. Klikom na *OK* se pokreće proces kloniranja baze po pravilima koje smo definisali u prethodnim koracima. Ako sve prođe kako treba dobićemo novu bazu kakvu smo konfigurisali i .bak fajl:



a. Ako pak nešto nije konfigurisano kako treba za određene kolone ili je nešto drugo pošlo po zlu, izaći će prozor upozorenja da postoji greška i neće biti ništa kreirano.

3.2. Tipovi maski

Kod SDM-a postoji 5 mogućih maski koju je moguće postaviti nad kolonom:

1. *Null* – menja sve podatke u koloni NULL vrednošću
2. *Single Value masking* – slično *Null* maski, menja sve vrednosti kolone onom vrednošću koju postavimo kao *Single Value*
3. *Shuffle* – vrednosti kolone se mešaju između redova
4. *Group Shuffle* – više kolona se zajedno mešaju (ako je neophodno zadržati zavisnosti izmedju kolona)
5. *String Composite* – definišemo string šablon kojim će se ispuniti svaka kolona pojedinačno prateći taj šablon (radi na principu kvazi-regex –a, ništa neviđeno dosad).

Postoji određeni tip grešaka koje će SSMS da prepozna čim je napravimo, tipa stavimo *String Composite* nad INT kolonom, ili *Null* nad kolonom koja ne dozvoljava NULL vrednosti i tako to.

Neke druge greške će se ispoljiti tek nakon što se pokrene proces kloniranja, kao npr. pogrešno definisan šablon u *String Composite* načinu maskiranja.

[2]

4. Zaključak

Kao što smo videli postoje ograničenja korišćenja i DDM-a i SDM-a. Na DBA korisnicima je da odluče koje su pozitivne a koje negativne strane oba i koliko utiču na konkretan sistem koji se koristi. Ali pored tih ograničenja, prednosti slučajeva korišćenja su izuzetno korisne za dobar deo korisnika SQL Server baza podataka koji žele da prikažu podatke tako da sakriju osetljive podatke, ali zadrže verodostojnost podataka.

DDM će se dobro pokazati u okolini sa lepo definisanim pravilima obezbeđenja jer dinamički skriva podatke na osnovu korisnika koji traži podatke, pa tako može više aplikacija istovremeno koristiti jednu bazu, ali neke će prikazivati sirove podatke, a neke maskirane podatke, po potrebi.

SDM je izuzetno koristan u svrsi olakšavanja posla DBA-u koji je zadužen za distribuiranje produkcionih podataka nižim okruženjima za razvoj ili testiranje, ili pak slanje celokupnih podataka sa par skrivenih informacija koje nisu od važnosti za sadržane podatke.

Sve u svemu, izuzetno korisne funkcionalnosti u zavisnosti kako se koriste (s tim što moramo još malo sačekati na SDM funkcionalnosti u GA verzijama SSMS-a).

5. Literatura

[1] https://docs.microsoft.com/en-us/sql/relational-databases/security/dynamic-data-masking?view=sql-server-2017

[2] https://www.sqlshack.com/static-data-masking-in-ssms-18/