

UNIVERZITET U NIŠU
ELEKTRONSKI FAKULTET
KATEDRA ZA RAČUNARSTVO

**CLOUD BAZE PODATAKA I DATABASE-AS-A-SERVICE
REŠENJA – NEO4J**

Predmet: Sistemi za upravljanje bazama podataka

Mentor:

doc. dr Aleksandar Stanimirović

Student:

Krstić Katarina, broj indeksa: 1400

SADRŽAJ

1. Uvod.....	3
2. Database-as-a-service.....	4
3. Neo4j Aura	5
3.1. Kreiranje naloga.....	5
4. Neo4j AuraDB	6
4.1. Kreiranje instance	6
4.2. Povezivanje sa instancom baze podataka.....	10
4.3. Izvršavanje upita nad instancom baze podataka.....	14
4.4. Importovanje podataka.....	16
4.4.1. Importovanje podataka iz .csv fajla.....	16
4.4.2. Importovanje podataka iz gotove Neo4j baze podataka.....	17
4.5. Monitoring.....	20
4.6. Backup i restore	21
4.6.1. Snapshot.....	21
4.6.2. Backup	22
4.6.3. Restore.....	22
4.7. Upravljanje instancama.....	23
4.7.1. Rename	24
4.7.2. Resize.....	25
4.7.3. Pauziranje instance	25
4.7.4. Resume instance.....	25
4.7.5. Klonovanje instance	26
4.7.6. Brisanje instance	26
4. Zaključak	28
5. Spisak slika.....	29
6. Literatura	31

1. UVOD

Poslednjih godina, naročito sa porastom veličine sistema koji se implementiraju, sve je veća potreba za maksimalnim pojednostavljenjem u radu sa bazom podataka. Sa tim na umu nastaju i razvijaju se *database-as-a-service* rešenja, koja svojim jednostavnim grafičko-korisničkim interfejsima i podrškom olakšavaju zadatke koje upravljanje bazom podataka nosi sa sobom. Maksimalno rasterećuju *database manager*-a poslova koji se tiču administracije i omogućavaju posvećenost isključivo bazi podataka. Time se efikasnost u radu povećava, a posao pojednostavljuje.

Svojim jednostavnim grafičko-korisničkim interfejsom i svim vidovima aplikacija razvijenih kao podrška ovakvih sistema omogućava se da ih bez imalo muke i korisnici koji nisu profesionalci u ovom polju upotrebljavaju, a, uz to, svaki od njih nudi veliku količinu uputstava i *tutorial*-a kao pomoć u radu. Kao i svako od poznatih *cloud* okruženja, omogućavaju izvršavanje konkretnih upita na konkretnom *query* jeziku, a i za sam način sastavljanja upita postoji puno uputstava i podrške.

Najpoznatije *Neo4j database-as-a-service* rešenje je *Neo4j Aura* i ona nudi mogućnost jednostavnog rada i u polju baza podataka, ali i u polju veštačke inteligencije, a zbog mnoštva pogodnosti koje nudi izdvaja se kao vodeće rešenje ovog tipa kada je reč o *Neo4j*-u.

2. DATABASE-AS-A-SERVICE

Database-as-a-service(DBaaS) je *cloud-computing managed service* rešenje koje omogućava upotrebu baze podataka bez ikakve potrebe za prethodnim podešavanjima koja se tiču fizičkog hardvera, instaliranjem softvera ili konfigurisanjem same baze podataka. Za upravljanje i administraciju je zadužen *service provider*, te se na taj način korisnik maksimalno oslobađa odgovornosti i zadataka takvog tipa. Uzevši to u obzir, zaključuje se da se korisniku dopušta da uživa pogodnosti same baze podataka i koncentriše se na njenu upotrebu pre nego na dodatnu konfiguraciju koja bi uz to morala da ide.

Jedna od kompletnijih definicija *Database-as-a-service* rešenja je: „*A paradigm for data management in which a third-party service provider hosts a database and provides the associated software and hardware support.*”

S obzirom na činjenicu da je *service provider* zadužen za veliki broj stvari u *DBaaS*-u, celokupan zadatak koji se tiče upravljanjem bazom podataka se na ovaj način maksimalno olakšava. Ne samo da se na taj način štedi velika količina vremena koju bi administrativni zadaci zahtevali, već se i sami svakodnevni zadaci koji se tiču upravljanja bazom podataka pojednostavljuju, postaju manji i lakši. Kompletan proces je mnogo fleksibilniji. Upravo su ovo i osnovni razlozi zbog kojih se sve veći broj kompanija odlučuje baš za ovu vrstu rešenja.

Među prednostima *database-as-a-service* rešenja svakako možemo izdvojiti:

- Redukovan broj menadžment zahteva – sam *service provider* je zadužen za najveći broj zahteva ovog tipa
- Fizička infrastruktura – kao i za administrativne zadatke, *service provider* zadužen je da obezbedi i fizičku infrastrukturu neophodnu za rad same baze podataka
- Redukovani troškovi
- Brži postupak implementacije
- Agilnost – činjenica je da zahvaljujući svojim prednostima, *DBaaS* može lako da se nosi sa čestim promenama biznis logike
- Sigurnost i bezbednost – postoji mogućnost kriptovanja podataka koje je neophodno zaštititi, bilo da je reč o skladištenim podacima, podacima koji učestvuju u transakciji ili se nalaze u bilo kom stanju

Ali, i ovakva vrsta rešenja ima nedostatke, među kojima bi najvažniji bili:

- Nedostatak kontrole – ne postoji mogućnost direktnog pristupa skladištu i serverima koji su zaduženi za rad baze podataka
- Internet konekcija – ukoliko dođe do problema sa internet konekcijom sistema koji je zadužen za upravljanje ovom bazom podataka, organizacija koja ga koristi neće moći da pristupa i koristi svoje podatke dokle god se ovaj problem ne reši
- Latencija – potrebno je više vremena pristupiti podacima organizovanim na ovaj način nego što bi to bio slučaj kada bismo koristili standardne sisteme za upravljanje bazama podataka, što automatski utiče i na performanse i brzinu samog sistema koji ga koristi

3. NEO4J AURA

Neo4j Aura je *cloud service* rešenje koje odlikuju pogodnosti poput brzine i skalabilnosti. Aktivan je sve vreme i predstavlja potpuno automatsku graf platformu.

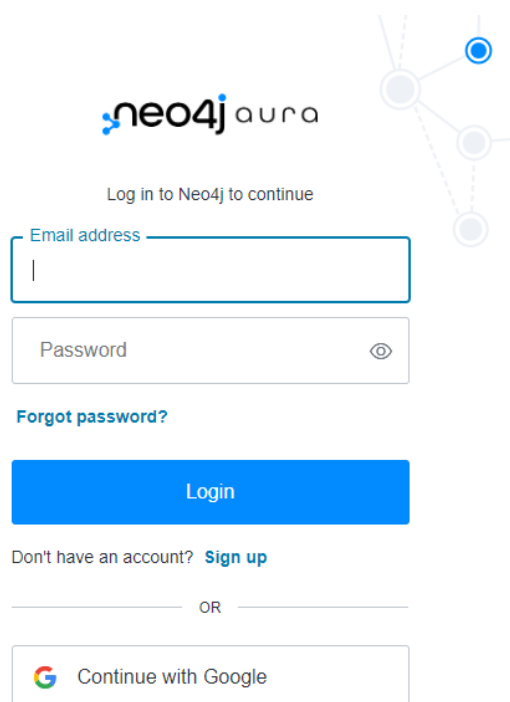
Aura uključuje *AuraDB* i *AuraDS*, pri čemu je *AuraDB* graf *database-as-a-service* rešenje, dok je *AuraDS* graf *data science as a service* rešenje. Možemo reći da *Neo4j AuraDB* omogućava razvoj inteligentnih aplikacija vođenih kontekstom na mnogo brži način. Sa druge strane, *Neo4j AuraDS* je pogodna za mašinsko učenje, jer na jednostavan način nudi mogućnost razvoja modela za predikciju.

3.1. KREIRANJE NALOGA

Da bismo pristupili *Neo4j Aura*-i neophodno je da imamo svoj korisnički nalog. Kreiranje naloga odvija se u par jednostavnih koraka, a upotreba je potpuno besplatna.

Nalog bismo mogli da kreiramo na sledeći način:

1. Prvo je neophodno pristupiti *Neo4j Aura*-i
2. Potom, neophodno je uneti svoju *e-mail* adresu i lozinku ili koristiti *Google* nalog da bismo se registrovali. U svakom slučaju neophodno je još i verifikovati svoju *e-mail*
3. Po prihvatanju uslova korišćenja možemo bez problema koristiti *Neo4j Aura*-u



Slika1: Kreiranje *Neo4j Aura* naloga/prijavljivanje na *Neo4j Aura* nalog

4. NEO4J AURADB

Kao što je već pomenuto, *Neo4j AuraDB* je graf *cloud database service*. I u ovoj vrsti *Neo4j* rešenja, teži se postizanju maksimalnog olakšanja u radu sa relacijama, naglašavajući njihovu važnost i moć. Uz to, vrlo je pouzdano, sigurno, automatizovano, i pruža mogućnost maksimalnog fokusiranja na razvoj same baze podataka. Kao i sva ostala *database-as-a-service* rešenja, maksimalno oslobađa korisnika potrebe za obavljanjem administrativnih zadataka.

Postoje tri *subscription plan*-a *Neo4j AuraDB* koje je moguće koristiti, a to su: *AuraDB Free*, *AuraDB Professional* i *AuraDB Enterprise*, a razlikuju se po pitanju funkcionalnosti i podrške koju imaju. Bez obzira na svoje razlike, svako od ovih rešenja pruža dovoljan broj pogodnosti i olakšica ne bi li predstavljao dobro rešenje.

4.1. KREIRANJE INSTANCE

U zavisnosti od toga za koju se varijantu *AuraDB* rešenja opredelimo, sam postupak kreiranja instance baze podataka će se razlikovati u određenoj meri. Kako je *AuraDB Free* rešenje jedino besplatno, najjednostavnije je koristiti upravo njega ukoliko zadovoljava potrebe koje imamo. Bitno je naglasiti da je za sada moguće kreirati samo jednu instancu baze podataka po korisničkom nalogu.

Postupak kreiranja *Neo4j AuraDB Free* instance baze podataka:

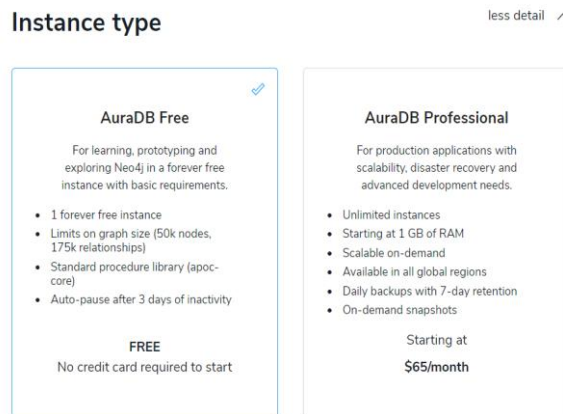
- U *browser*-u pristupiti *Neo4j AuraDB Console*
- Odabrati *New Instance* što će otvoriti *Create new instance* stranu
- Kao tip instance neophodno je odabrati *Neo4j AuraDB Free* za *Instance Type*
- Popuniti informacije koje se odnose na samu instancu, odnosno *Instance Details*:
 - ❖ *Instance name* – naziv koji će instanca koristiti
 - ❖ *Region* – fizička lokacija instance koja će se koristiti. Suštinski nije toliko važno koja se lokacija postavi za ovaj parametar, međutim preporuka je postaviti lokaciju na najbližu moguću iz jednostavnih razloga – što je lokacija bliža to će podaci brže biti pribavljeni, te je latentnost manja
- Odabrati *Create instance* opciju nakon što su svi neophodni podaci unešeni
- Kopirati *Username* i *Password* podatke i sačuvati ih na nekom sigurnom mestu sa kojeg ih lako možemo pribaviti
- Potom selektovati *I have stored these credentials safely to use later checkbox* i odabrati *Continue*

Kada se okonča ovaj postupak imaćemo kreiranu gotovu instancu baze podataka.

Postupak kreiranja instance baze podataka u preostala dva *subscription plan*-a se ne razlikuje preterano od postupka kreiranja instance baze podataka u *Neo4j AuraDB Free subscription plan*-u.

Ono što je svakako jedna od mogućih opcija u postupku kreiranja instance baze podataka putem *Neo4j AuraDB*-a jeste automatska inicijalna popuna instance postojećim

podacima. Naime, moguće je kreirati ili potpuno praznu instancu baze podataka, međutim, moguće je i iskoristiti *dataset Movie dataset*, koji sadrži podatke o filmovima, glumcima, rediteljima i tako dalje. Uglavnom se kreiranje instance sa ovim *dataset*-om preporučuje tokom upoznavanja sa radom kako ovog sistema, tako i *Neo4j*-a generalno, jer se nad već gotovom bazom podataka mogu izvršavati i upiti i može se lakše upoznati sa nekim od mogućnosti samog *cloud* sistema.



Slika2a: Kreiranje instance baze podataka – Odabir tipa instance

Instance details

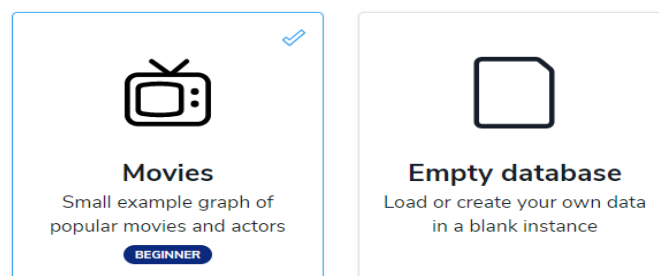
Instance Name

GCP Region

Iowa, USA (us-central1) ▼

Slika2b: Kreiranje instance baze podataka – Popunjavanje osnovnih informacija o instanci

Starting dataset

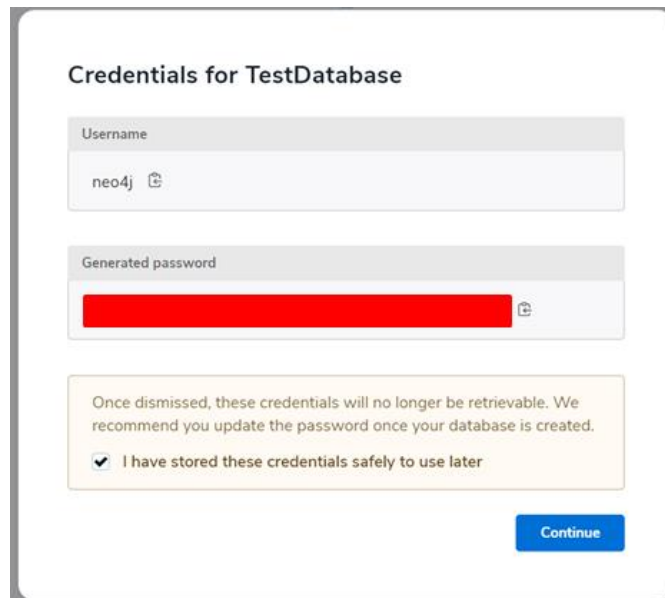


Slika2v: Kreiranje instance baze podataka – Odabir popune podataka u instanci baze podataka



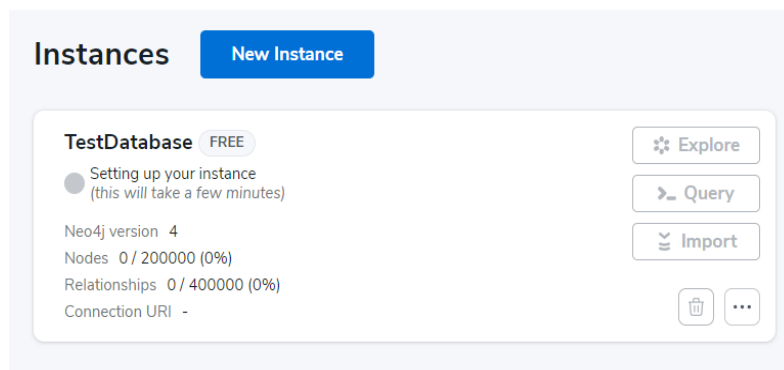
Slika2g: Kreiranje instance baze podataka – Konačno kreiranje instance

Pošto se kreira instanca baze podataka, korisniku se na uvid pružaju informacije o generisanom korisničkom imenu i *password*-u koji je neophodno koristiti za rad sa ovom instancom. Da bi se moglo manipulirati njome, neophodno je poznavati ove parametre, te je iz tog razloga neophodno sačuvati ih na nekom bezbednom i dostupnom mestu.

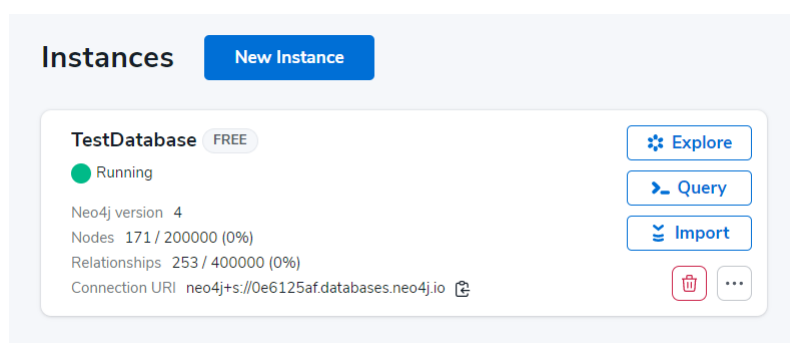


Slika3: Generisani parametri o korisničkom imenu i password-u neophodnim za pristup i rad sa kreiranom instancom bazom podataka

Po kreiranju instance baze podataka neophodno je nekoliko trenutaka da se ona *setup*-je i u tom periodu nije moguće manipulirati njenim podacima i njome. Onda kada njen status postane *Running*, znači da je spremna za korišćenje.



Slika4: Kreirana instanca baze podataka u Setup fazi



Slika5: Kreirana instanca baze podataka u Running fazi

Kako *Neo4j AuraDB* dozvoljava kreiranje samo jedne *Free* instance po korisničkom nalogu, ukoliko bismo hteli da kreiramo novu, odabirom *New instance*, imali bismo mogućnost da kreiramo novu instancu koja bi morala da bude ili *Enterprise* ili *Professional*. U slučaju kreiranja instance jednog od ovih subscription plan-ova nudi nam se mogućnost uticanja na cenu, s obzirom na to da ni *Enterprise* ni *Professional subscription plan* nisu besplatni. Naime, u zavisnosti od veličine instance koja zadovoljava naše potrebe, razlikovaće se i cena.

Instance size ⓘ

	Memory	CPU	Storage	Price
<input type="radio"/>	1GB	1 CPU	2GB	\$0.09/hour (\$64.80/month)
<input type="radio"/>	2GB	1 CPU	4GB	\$0.18/hour (\$129.60/month)
<input type="radio"/>	4GB	1 CPU	8GB	\$0.36/hour (\$259.20/month)
<input checked="" type="radio"/>	8GB	2 CPU	16GB	\$0.72/hour (\$518.40/month)
<input type="radio"/>	16GB	3 CPU	32GB	\$1.44/hour (\$1,036.80/month)
<input type="radio"/>	24GB	5 CPU	48GB	\$2.16/hour (\$1,555.20/month)
<input type="radio"/>	32GB	6 CPU	64GB	\$2.88/hour (\$2,073.60/month)
<input type="radio"/>	48GB	10 CPU	96GB	\$4.32/hour (\$3,110.40/month)
<input type="radio"/>	64GB	12 CPU	128GB	\$5.76/hour (\$4,147.20/month)

Slika6: Kreiranje Neo4j AuraDB Professional instance baze podataka i uticaj veličine instance na cenu

S obzirom na činjenicu da je *Neo4j AuraDB* bazu podataka moguće povezati sa velikim brojem projekata koji koriste različite tehnologije, ovaj sistem nam nudi i uputstva o tome na koji je način to moguće postići u zavisnosti od programskog jezika i tehnologije. Tako, sa leve strane možemo naći sidebar, odabrati tehnologiju i pronaći način na koji ovaj tip rešenja za bazu podataka možemo uključiti u sistem koji implementiramo.



Slika7: Sidebar

4.2. POVEZIVANJE SA INSTANCOM BAZE PODATAKA

Postoji 4 načina na koje je moguće povezati se na instancu baze podataka:

- *Neo4j Browser*
- *Neo4j Bloom*
- *Neo4j Desktop*
- *Neo4j Cypher Shell*

Neo4j Browser je *browser-based* interfejs koji je moguće koristiti za izvršavanje upita i manipulaciju nad podacima unutar instance baze podataka. To je alat u okviru kojeg je moguće izvršavati *Cypher* upite i koji omogućava vizuelizaciju rezultata tih upita. On je podrazumevani *developer* interfejs i u *Professional* i u *Enterprise subscription plan*-u *Neo4j*-a. *Neo4j Browser* sadrži *Cypher Editor* u okviru kojeg se detektuju sintaksne greške, nude predlozi naredbi i pružaju upozorenja koja olakšavaju pisanje *Cypher* upita.

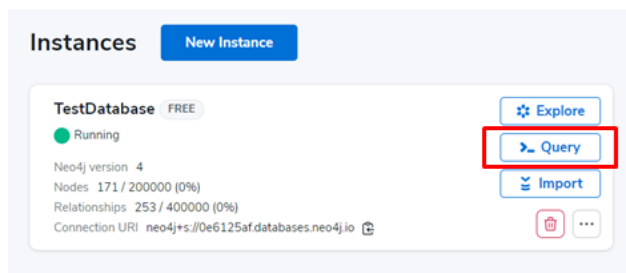
Neo4j Bloom je aplikacija za vizuelnu interakciju sa graf bazom podataka. Osim prikazivanja paterna koji su nam kao takvi poznati, da postoje u grafu koji koristimo, *Neo4j Bloom* pruža nam uvid u paterne i veze među podacima koje možda i nismo ustanovili sami, te nam na taj način pruža i novi način razmišljanja i nudi nova rešenja za probleme sa kojima bismo se mogli susreti. Smatra se vrlo moćnim *engine*-om za izvršavanje upita nad velikom količinom duboko povezanih podataka, a svoju snagu oblikuje u vidu aplikacije za vizuelizaciju grafa.

Moguće je povezati se sa *AuraDB Neo4j* bazom podataka i putem *Neo4j Desktop* aplikacije, pri čemu je moguće koristiti istu aplikaciju za pristup svim bazama podataka, bilo da su one skladištene lokalno ili na *cloud*-u. *Neo4j Desktop* nudi mogućnost povezivanja sa samo jednom instancom baze podataka odjednom, što će reći da je neophodno zatvoriti otvorenu konekciju sa instancom baze podataka ne bi li smo mogli da se povežemo sa novom instancom.

Cypher Shell je *command line interface* koji se koristi za izvršavanje upita i određenih administrativnih zadataka nad instancom *Neo4j* baze podataka. Podrazumevano, sam *shell* je interaktivan, međutim, pruža i mogućnost izvršavanja skripte ili tako što bi se *cypher* upiti direktno prosledili na ulaz ili tako što bi se fajl koji sadrži *cypher* upite prosledio na standardni ulaz.

Povezivanje sa instancom Neo4j baze podataka korišćenjem Neo4j Browser-a:

- Pristupiti *Neo4j Aura Console*-i u *browser*-u
- Kliknuti na *Query* dugme pored one instance baze podataka sa kojom želimo da radimo
- Uneti korisničko ime i lozinku koji su generisani u postupku kreiranja instance
- Odabrati *Connect*



Slika8a: Povezivanje sa instancom baze podataka korišćenjem Neo4j Browser-a – Pristup Neo4j Browser-u

Connect to Neo4j

Database access might require an authenticated connection

Connect URL
 neo4j+s:// 0e6125af.databases.neo4j.io

Authentication type
 Username / Password

Username
 neo4j

Password

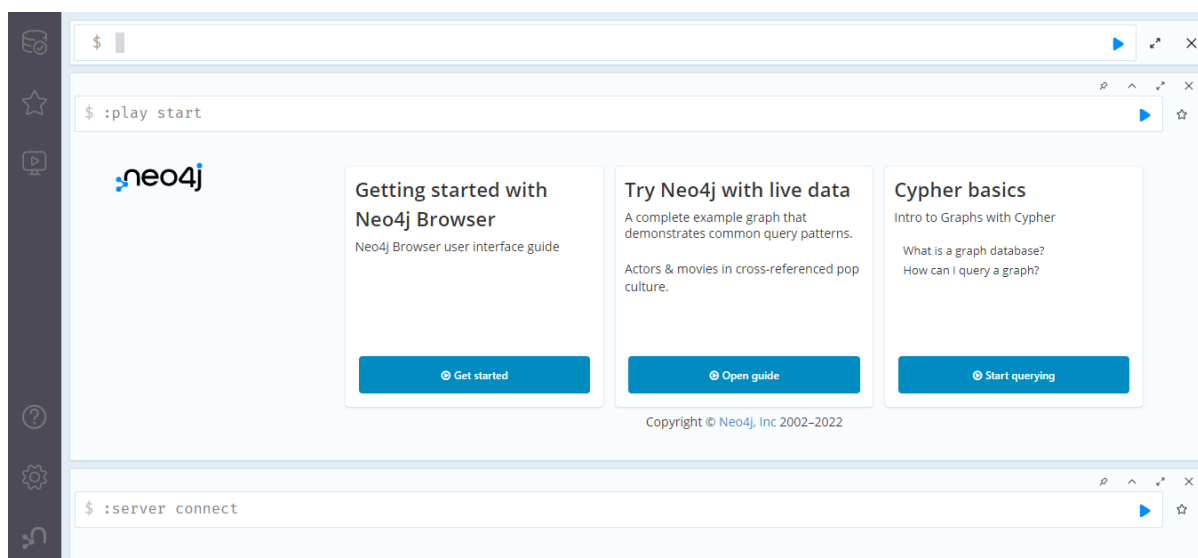
Connect

Slika8b: Povezivanje sa instancom baze podataka korišćenjem Neo4j Browser-a – Popuna informacija za konekciju

Connected to Neo4j
Nice to meet you.

You are connected as user **neo4j** to **neo4j+s://0e6125af.databases.neo4j.io:7687**
 Connection credentials are not stored in your web browser.

Slika8v: Povezivanje sa instancom baze podataka korišćenjem Neo4j Browser-a – Ostvarena konekcija



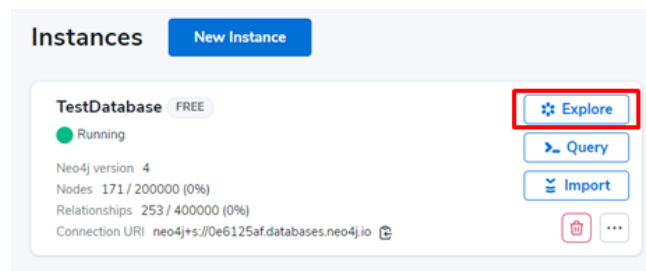
Slika8g: Povezivanje sa instancom baze podataka korišćenjem Neo4j Browser-a – Neo4j Browser

Pošto se konektujemo sa instancom baze podataka putem *Neo4j Browser*-a otvoriće nam se strana kao na slici *Slika8g*. Naime, već na početnoj strani imamo mogućnost da odaberemo neki od osnovnih *tutorial*-a i uputstava kako za upotrebu samog *Neo4j Browser*-a, tako i za osnove *Cypher query* jezika, prvenstveno zbog toga što je *Neo4j Browser* upravo namenjen za manipulaciju nad podacima upitima na ovom jeziku.

U komandnoj liniji *:play start* moguće je direktno kucati upite na *Cypher* jeziku i izvršavati ih, dok bi se rezultati tih upita direktno prikazivali u polju ispod.

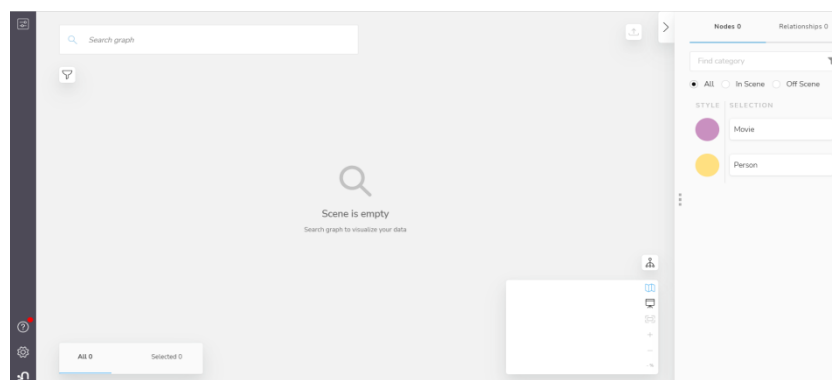
Povezivanje sa instancom Neo4j baze podataka korišćenjem Neo4j Bloom-a:

- Pristupiti *Neo4j Aura Console*-i u *browser*-u
- Kliknuti na *Explore* dugme pored one instance baze podataka sa kojom želimo da radimo
- Selektovati *Neo4j Bloom* iz opadajućeg menija
- Uneti korisničko ime i lozinku koji su generisani u postupku kreiranja instance
- Odabrati *Connect*



Slika9a: Povezivanje sa instancom baze podataka pomoću Neo4j Bloom-a – Pristup Neo4j Bloom-u

Slika9b: Povezivanje sa instancom baze podataka pomoću Neo4j Bloom-a – Popuna informacija za konekciju



Slika9v: Povezivanje sa instancom baze podataka pomoću Neo4j Bloom-a – Neo4j Bloom

Povezivanje sa instancom Neo4j baze podataka korišćenjem Neo4j Desktop-a:

- Pristupiti *Neo4j Aura Console*-i putem *browser*-a
- Kopirati *Connection URI* instance baze podataka sa kojom želimo da se povežemo. *Connection URI* se nalazi ispod indikatora statusa instance baze podataka
- U *Neo4j Desktop*-u selektovati tab *Projects* i odabrati ili *existing project* ili kreirati novi
- Odabrati *Add* a zatim *Remote connection*
- Uneti zatim naziv baze podataka i kopirani *URI* i kliknuti na *Next*
- Uneti korisničko ime i lozinku koji su generisani u postupku kreiranja instance
- Kada postane dostupno, kliknuti na *Connect* dugme i ostvariti konekciju sa instancom baze podataka

Povezivanje sa instancom Neo4j baze podataka korišćenjem Neo4j CLI-a:

- Pristupiti *Neo4j Aura Console*-i putem *browser*-a
- Kopirati *Connection URI* instance baze podataka sa kojom želimo da ostvarimo konekciju
- Otvoriti terminal a zatim se pozicionirati na onu lokaciju gde je *Cypher Shell* instaliran

Izvršiti sledeću komandu, pri čemu treba zameniti parametre *connection_uri*, *username* i *password* konkretnim vrednostima, tako da umesto *connection_uri*-a stoji kopirani *Connection URI* instance baze podataka sa kojom se ostvaruje konekcija, a umesto *username* i *password*-a treba da stoje konkretno korisničko ime i lozinka koji su generisani prilikom kreiranja instance baze podataka

```
./cypher-shell -a <connection_uri> -u <username> -p <password>
```

Slika10a: Povezivanje sa instancom baze podataka pomoću Neo4j CLI-a - Komanda

Pošto se ostvari konekcija, izvršavanjem *:help* moguće je izlistati sve dostupne komande koje je moguće izvršiti u *CLI*-u:

```
Available commands:
:begin      Open a transaction
:commit     Commit the currently open transaction
:exit       Exit the logger
:help       Show this help message
:history     Print a list of the last commands executed
:param      Set the value of a query parameter
:params     Print all currently set query parameters and their values
:rollback   Rollback the currently open transaction
:source     Interactively executes cypher statements from a file
:use        Set the active instance

For help on a specific command type:
:help command
```

Slika10b: Povezivanje sa instancom baze podataka pomoću Neo4j CLI-a – Komande koje je moguće izvršiti nad instancom baze podataka iz CLI-a

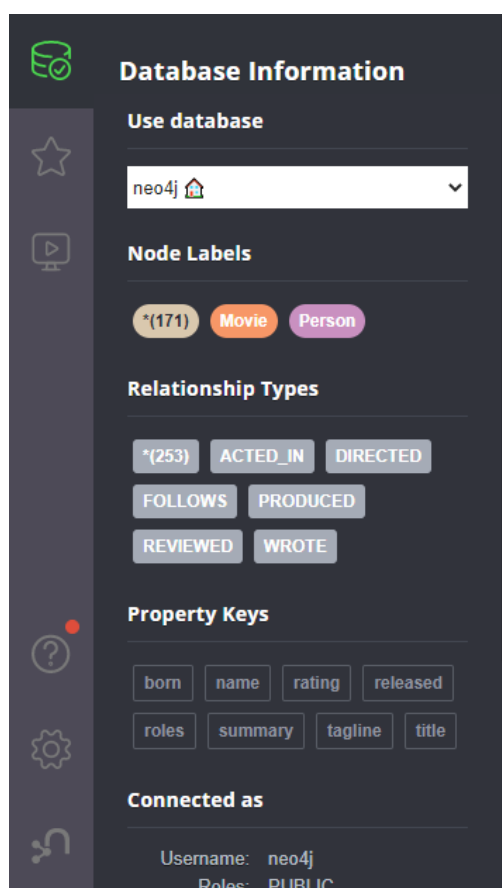
4.3. IZVRŠAVANJE UPITA NAD INSTANCOM BAZE PODATAKA

Izvršavanje upita nad instancom baze podataka podrazumeva korišćenje *Cypher* query jezika. Naime, *Cypher* je deklarativni *query* jezik kreiran za potrebe *Neo4j*-a, a može se koristiti kako za obavljanje osnovnih *CRUD* operacija nad instancom baze podataka, tako i za obavljanje nekih osnovnih administrativnih zadataka nad njom.

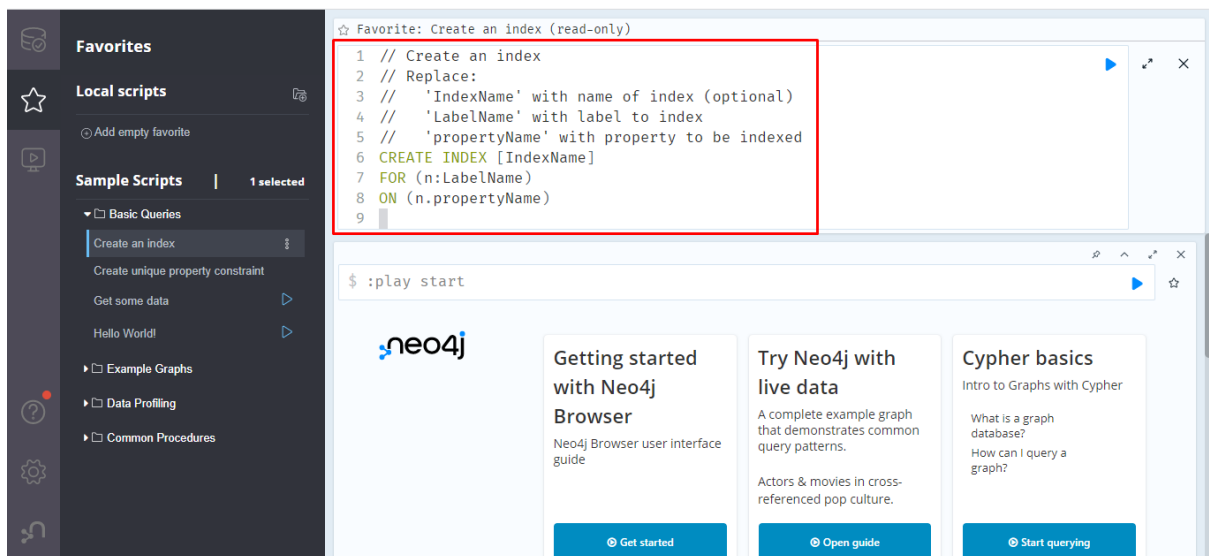
Cypher upiti mogu se izvršavati korišćenjem *Neo4j Browser*-a ili *Neo4j Cypher Shell*-a, a važno je pomenuti da *AuraDB* ne dozvoljava izvršavanje *Cypher* upita koji bi zahtevali pristup lokalnim fajlovima na sistemu.

Izvršavanje upita nad instancom baze podataka korišćenjem Neo4j Browser-a:

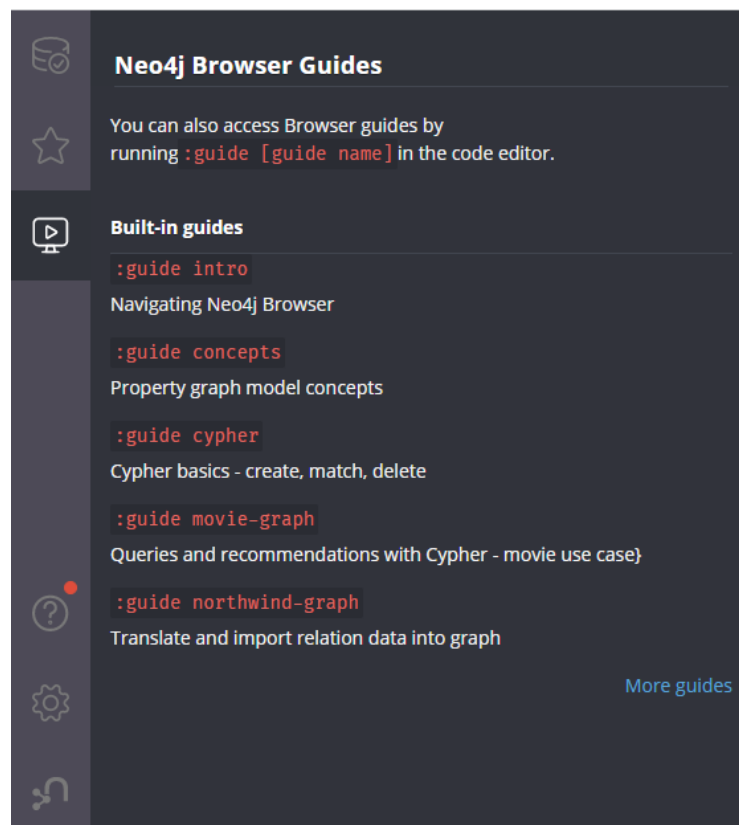
Jedna od, svakako, važnih pogodnosti *Neo4j Browser*-a je da unutar *Sidebar*-a imamo mogućnost pregleda svih relacija, atributa, labela čvorova, u zavisnosti od baze podataka koju odaberemo. Takođe, nudi mogućnost pregleda načina na koji bi trebalo sastaviti *query* u *Cypher*-u za neke osnovne elemente baze podataka, poput indeksa ili *constraint*-a, a pruža i uputstva za mnoge od tih stvari.



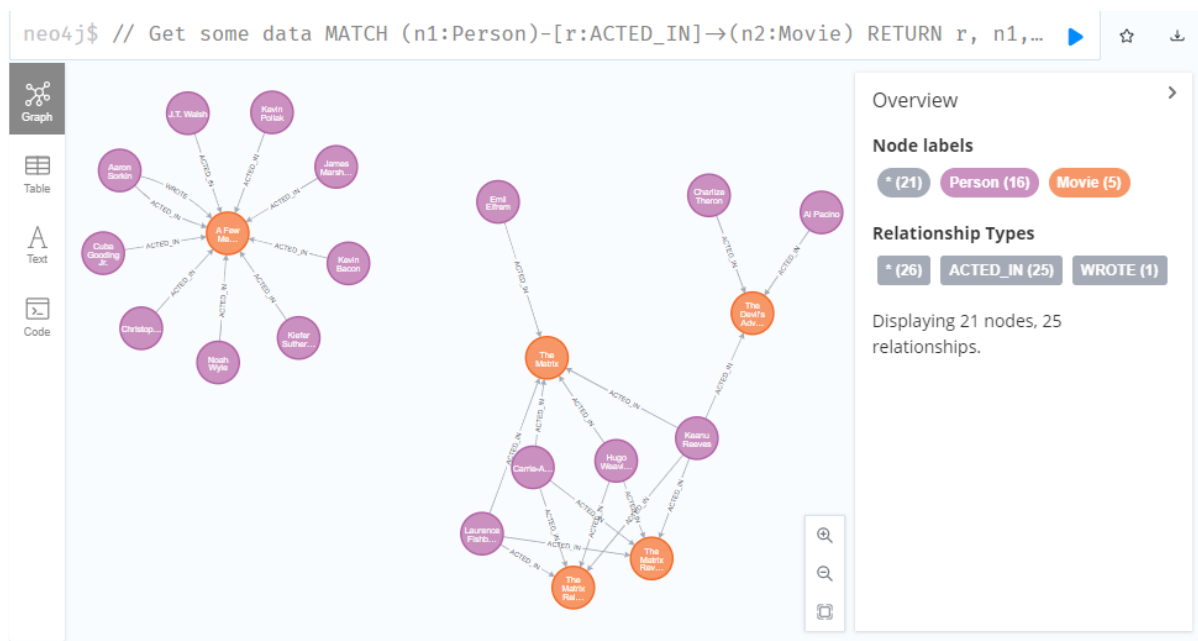
Slika 11a: Neo4j Browser – Database Information



Slika11b: Neo4j Browser – Favorites – Primer uvida u način na koji bi trebalo kreirati indeks Cypher upitom



Slika11v: Neo4j Browser – Uputstva



Slika 11g: Neo4j Browser – Primer izvršenja upita za pribavljanje podataka i prikaz rezultata

4.4. IMPORTOVANJE PODATAKA

Pre izvršavanja postupka za importovanje podataka neophodno je kreirati *AuraDB* instancu. Potom, moguće je izvršiti i sam postupak importovanja podataka iz *.csv* fajla u *AuraDB* instancu i to je moguće učiniti na dva načina:

- *Load CSV*
- *Neo4j Data Importer*

4.4.1. IMPORTOVANJE PODATAKA IZ *.CSV* FAJLA

Load CSV je komanda koju je moguće izvršiti u *Neo4j Browser*-u ili u *Neo4j CLI*-u. Ono što je možda i glavni nedostatak izvršavanja ove komande u svrhe importovanja podataka iz *.csv* fajla u *Neo4j* instancu baze podataka jeste činjenica da je neophodno hostovati ovaj fajl na serveru koji bi javno bio dostupan. Odnosno, potrebno je postaviti ovaj fajl na serveru kao što je *GitHub*, *DropBox* ili *Google Drive*, i na taj način *expose*-ovati sam fajl javno.

Takođe, *Load CSV* namenjena je za manje *dataset*-ove, odnosno skupove podataka koji bi sadržali najviše 10 miliona čvorova i relacija, pa bi ovaj način trebalo izbegavati u slučaju da skup podataka koji se koristi prevazilazi ove granice.

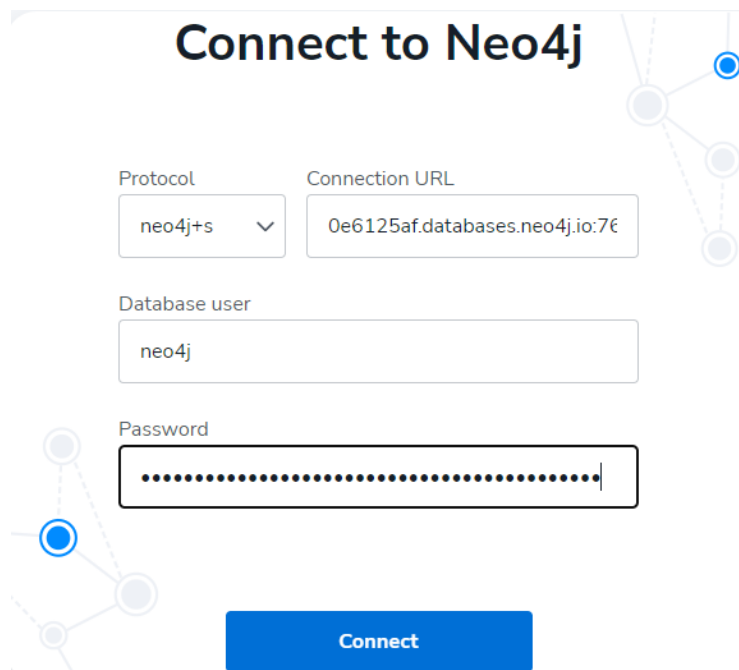
Neo4j Data Importer je aplikacija koju je moguće startovati iz *Console*-e i predstavlja *no-code* alat koji je moguće koristiti za učitavanje podataka iz fajlova u *.csv* i *.tsv* formatima, definisanje graf modela i mapiranje podataka na njega, kao i za jednostavno učitavanje podataka u *AuraDB* instancu.

Postupak za učitavanje podataka pomoću *Neo4j Data Importer*-a je vrlo jednostavan i zahteva da se pristupi *Neo4j Aura Console*-i i klikne na *Import* instance koju je potrebno

otvoriti. Takođe, pre nego što se podaci učitaju pomoću *Neo4j Data Importer*-a zahteva se unos lozinke same instance generisane prilikom njenog kreiranja.



Slika12a: Importovanje podataka – pristup Neo4j Data Importer-u



Slika12b: Importovanje podataka – konekcija sa instancom baze podataka

4.4.2. IMPORTOVANJE PODATAKA IZ GOTOVE NEO4J BAZE PODATAKA

Osim učitavanja podataka iz *.csv* fajlova u *Neo4j* instancu, moguće je izvršiti i učitavanje podataka iz gotove *Neo4j* baze podataka.

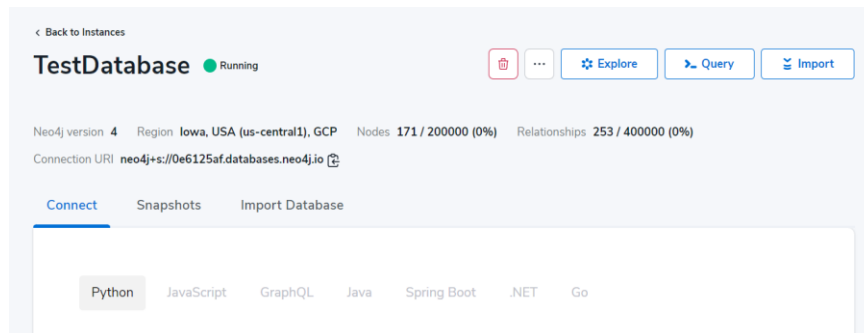
Importovanje podataka iz gotove Neo4j baze podataka korišćenjem .dump fajla

Najjednostavniji način na koji je ovo moguće obezbediti jeste putem *.dump* fajla. Naime, neophodno bi bilo eksportovati gotovu *Neo4j* bazu podataka u *.dump* fajl, a zatim ga učitati. Međutim, na ovaj način moguće je učitati samo *.dump* fajlove čija veličina ne prelazi *4GB*.

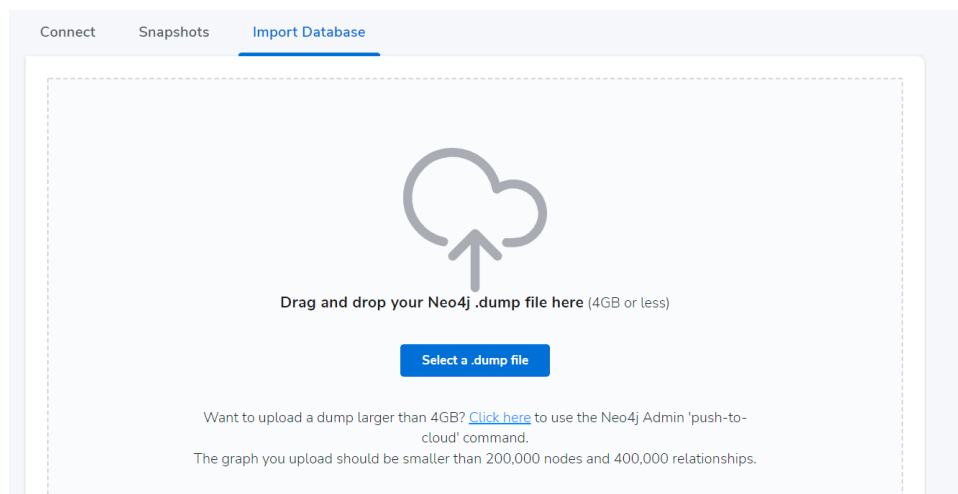
Postupak učitavanja podataka u AuraDB instancu baze podataka iz *.dump* fajla Neo4j baze podataka:

- Pristupiti Neo4j Aura Console-i putem browser-a
- Selektovati instancu u koju je neophodno učitati podatke
- Selektovati zatim Import Database tab

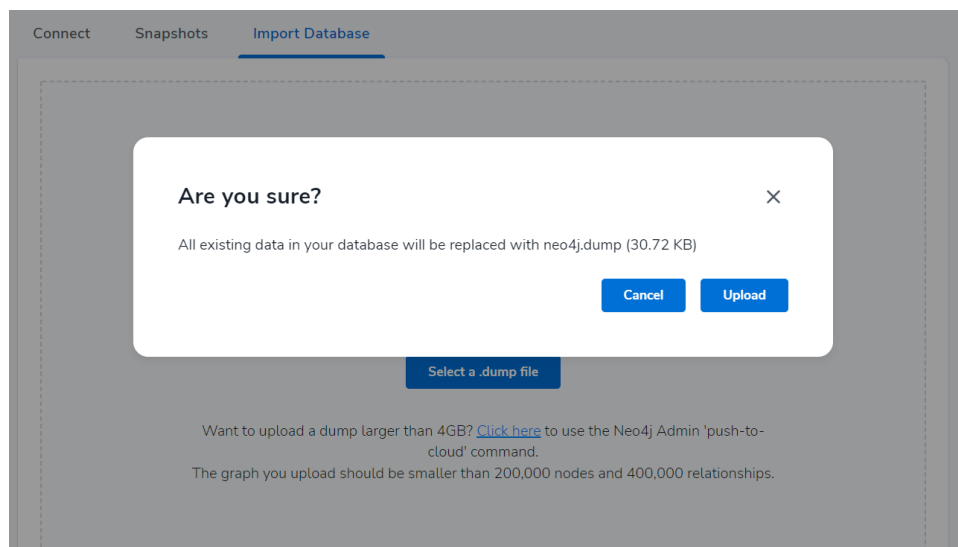
- Prevući željeni .dump fajl ili selektovati Select a .dump file i odabrati željeni .dump fajl
- Kliknuti na Upload



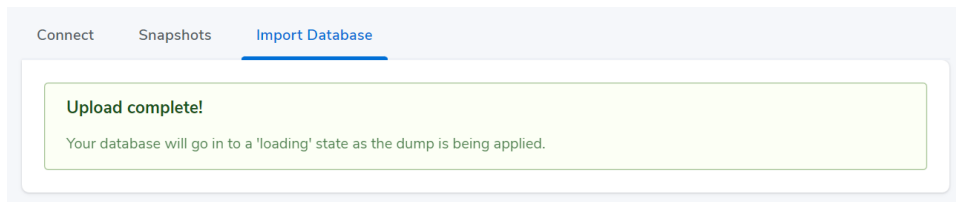
Slika13a: Importovanje podataka iz gotove Neo4j baze podataka – pristup Import Database



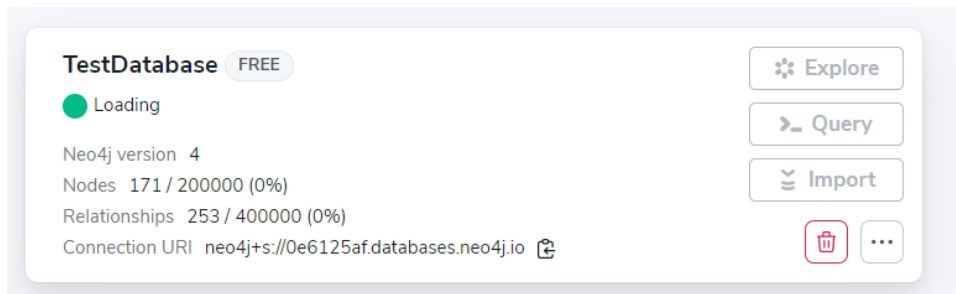
Slika13b: Importovanje podataka iz gotove Neo4j baze podataka – Import Database



Slika13v: Importovanje podataka iz gotove Neo4j baze podataka – Confirmation dialog



Slika13g: Importovanje podataka iz gotove Neo4j baze podataka – Upload complete



Slika13d: Importovanje podataka iz gotove Neo4j baze podataka – Loading state

Po završetku *upload*-a, instanca prelazi u stanje *Loading* dokle god se željeni podaci ne učitaju iz *.dump* fajla koji je učitao, a kada se to okonča vraća se u *Running* stanje i podaci su tada dostupni za korišćenje.

Importovanje podataka iz gotove Neo4j baze podataka korišćenjem push-to-cloud komande:

U slučaju da *.dump* fajl baze podataka prelazi veličinu od *4GB* neophodno je koristiti *push-to-cloud* komandu da bi se obavio *import* podataka.

Push-to-cloud je *Admin* komanda koju je moguće iskoristiti za učitavanje podataka direktno iz *Neo4j* baze podataka u *AuraDB* instancu baze podataka. Pritom, u ovom slučaju ne postoji nikakvo ograničenje po pitanju veličine i količine podataka unutar baze podataka iz koje se podaci učitavaju.

```
neo4j-admin push-to-cloud [--overwrite] [--verbose] --bolt-uri=<boltURI>
                        [--database=<database>] [--dump=<dump>]
                        [--dump-to=<tmpDumpFile>] [--password=<password>]
                        [--username=<username>]
```

Slika14a: Importovanje podataka iz gotove Neo4j baze podataka korišćenjem neo4j-admin push-to-cloud komande - Sintaksa neo4j-admin push-to-cloud komande

Kao i većina *Neo4j Admin* komandi, i *push-to-cloud* je komanda koja ima mnoštvo opcija koje je moguće definisati prilikom izvršavanja iste.

Već na osnovu spiska opcija ove komande možemo zaključiti da je moguće direktno navesti i naziv baze podataka iz koje bi trebalo učitati podatke, a moguće je navesti i putanju do *.dump* fajla koji bi trebalo učitati u ove svrhe. Međutim, ove dve opcije nije dozvoljeno koristiti istovremeno.

Takođe, postoje određeni preduslovi koji bi morali biti ispunjeni da bi se ova komanda ispravno izvršila. Naime, *AuraDB* instanca baze podataka mora biti, pre svega

kreirana, a pored toga, neophodno je da je i stanju *Running*. Sa druge strane, baza podataka koja se definiše kao vrednost *database* opcije ove komande mora biti stopirana pre poziva ove naredbe, jer se ne dozvoljava da se komanda *push-to-cloud* izvrši nad bazom podataka koja je u *Running* stanju. U slučaju da ovu komandu pozovemo šre stopiranja baze podataka iz koje se učitavaju podaci, komanda će generisati grešku.

Siguran znak da je komanda *push-to-cloud* izvršena uspešno je rezultat:

```
"Your data was successfully pushed to cloud and is now running".
```

Slika14b: Importovanje podataka iz gotove Neo4j baze podataka korišćenjem neo4j-admin push-to-cloud koamnde – Uspešno izvršena komanda

dok, u slučaju da prilikom izvršenja komande dođe do greške, biće pruženo uputstvo o načinu na koji bi trebalo ponoviti postupak ne bi li se ona okončala uspešno ili način na koji bi bilo moguće kontaktirati *Aura* podršku.

4.5. MONITORING

AuraDB nam nudi mogućnost praćenja stanja i upotrebe mnogih sistemskih resursa, a da bismo to uradili, postupak je vrlo jednostavan:

- Pristupiti *Neo4j Aura Console*-i
- Selektovati instancu čije je sistemske resurse potrebno pratiti
- Odabrati *Metrics* tab

Pritom, parametri koje je moguće pratiti na ovaj način su:

- *CPU Usage(%)* - količina procesora koja se koristi izražena u procentima
- *Storage Used(%)* - veličina skladišnog prostora koji se koristi izražena u procentima
- *Heap Usage(%)* - količina *Java Virtual Machine(JVM)* memorije koju instanca koristi, izražena u procentima. Ovaj parametar moguće je pratiti isključivo u *Enterprise subscription plan*-u.
- *Out of Memory Errors* – broj *Out of Memory Error*-a koje je instanca prouzrokovala
- *Garbage Collection Time(%)* - vreme koje instanca provede u procesu *reclaim*-ovanja *heap space*-a, izražena u procentima
- *Page Cache Evictions* – ovaj parametar označava broj koliko je puta instanca zamenila podatke u memoriji

Takođe, pogodnost koju *AuraDB* nudi kada je u pitanju praćenje sistemskih resursa jeste vremenski period za koji se prate ovi parametri, pa je tako moguće prikazati metriku ovih resursa u poslednjih 6 sati, 24 sata, 3 dana, 7 dana i 30 dana.

Ipak, analiza upotrebe ovih sistemskih resursa dostupna je u *Professional* i *Enterprise*, ali ne i u *Free subscription plan*-u *Neo4j AuraDB*-a.

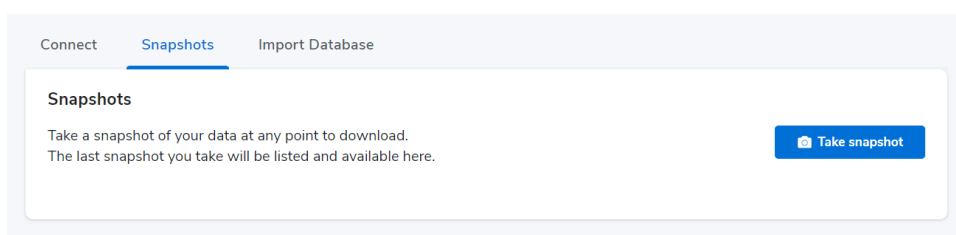
4.6. BACKUP I RESTORE

Svi podaci unutar *AuraDB* instance mogu biti *backup*-ovani, *export*-ovani i *restore*-ovani korišćenjem *snapshot*-ova. *Snapshot* predstavlja kopiju stanja podataka unutar baze podataka, odnosno kopiju same baze podataka u određenom trenutku.

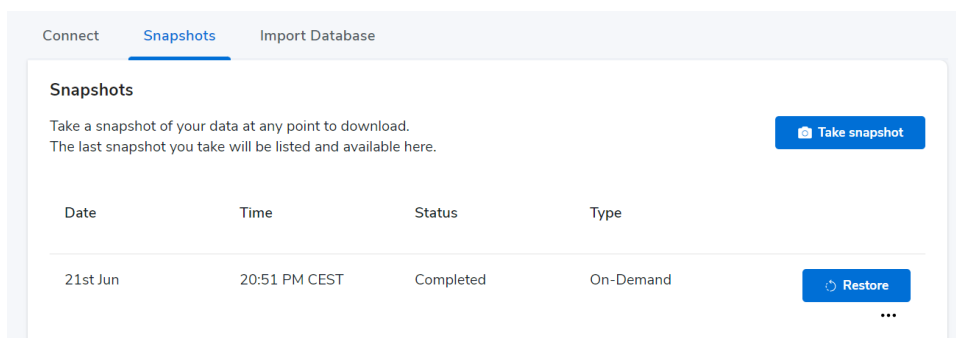
4.6.1. SNAPSHOT

Moguće je pristupiti i pregledati dostupne i postojeće *snapshot*-ove instance baze podataka navigiranjem do odgovarajućeg *Snapshot tab*-a:

- Pristupiti *Neo4j Aura Console*-i putem *browser*-a
- Selektovati instancu čije bi *snapshot*-ove trebalo pregledati
- Selektovati *Snapshot tab*



Slika15a: Snapshot tab – Bez ijednog postojećeg snapshot-a



Slika15b: Snapshot tab – Sa postojećim snapshot-om

Postoji određeni broj *snapshot* tipova, pri čemu su neki dozvoljeni u svim *subscription plan*-ovima *AuraDB*-a, a neki nisu svuda podržani. Tako, imamo:

Scheduled – prvi put se izvršava kada se instancira sama baza podataka, a zatim i svaki naredni put kada dođe do određene promene unutar osnovnog sistema ili automatski, u unapred definisanom vremenskom trenutku. Ovaj tip *snapshot*-ova podržavaju *Enterprise* i *Professional*, ali ne i *Free subscription plan*.

Load - izvršava se prilikom učitavanja *.dump* fajla putem *Import Database* taba instance i prilikom izvršavanja *push-to-cloud* komande. Kao i *Scheduled*, i *Load* tip *snapshot*-ova podržavaju samo *Enterprise* i *Professional*, ali ne i *Free subscription plan*.

On-demand - izvršava se onda kada se ručno to uradi i zahteva odabirom *Take snapshot* u *Snapshot tab*-u instance.

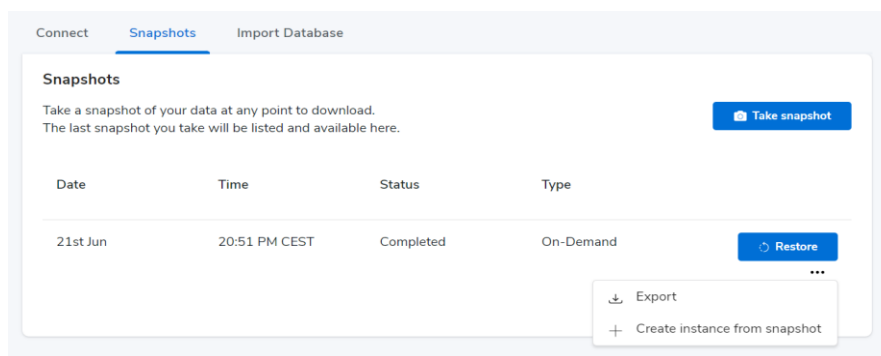
Kreirani *snapshot*-ovi se u *Free* i *Professional subscription plan*-ovima čuvaju 7 dana, dok se u *Enterprise subscription plan*-u oni čuvaju 90 dana. Takođe, *Scheduled snapshot*-ovi se u *Professional subscription plan*-u izvršavaju jednom dnevno, dok se u *Enterprise subscription plan*-u izvršavaju na svakih sat vremena.

4.6.2. BACKUP

Bilo koji postojeći *snapshot* može biti iskorišćen u svrhe kreiranja *backup* kopije instance baze podataka. Akcije koje je moguće izvršiti nad samim *snapshot*-om moguće je videti odabirom ... ikonice koja stoji pored nje:

- *Export*
- *Create instance from snapshot*

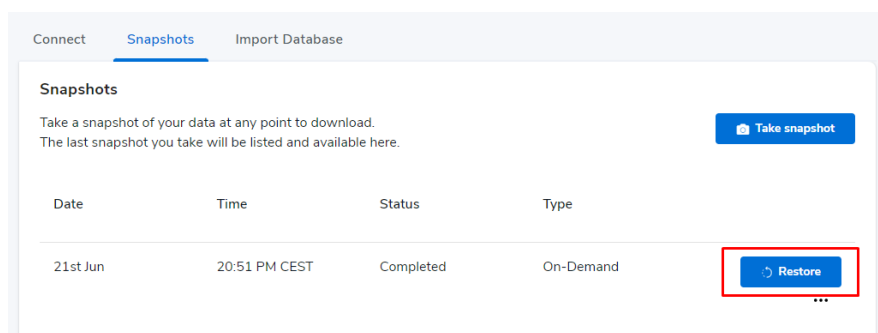
Export opcija nam omogućava da *download*-ujemo *.dump* fajl, čime se formira lokalna kopija instance baze podataka i pruža se mogućnost manipulacije nad podacima offline. Sa druge strane, opcija *Create instance from snapshot* nudi nam mogućnost da kreiramo novu *AuraDB* instancu na osnovu podataka u *snapshot*-u.



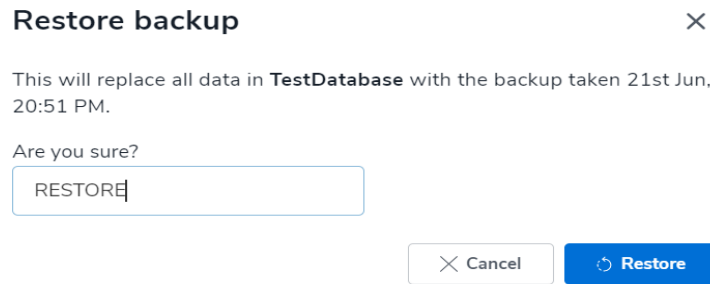
Slika16: Backup – Opcije

4.6.3. RESTORE

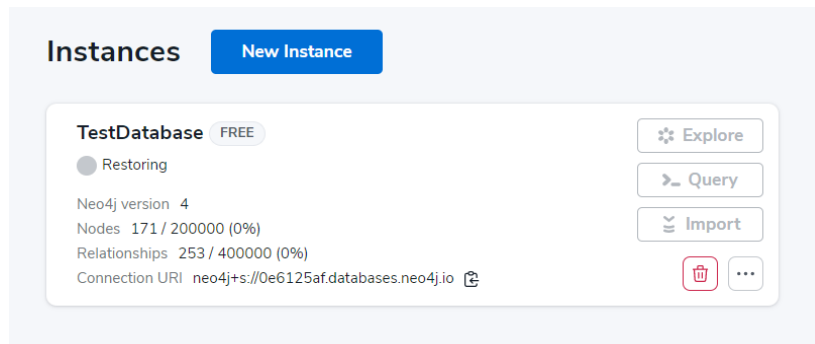
Sa druge strane, restore operacija nudi mogućnost *restore*-ovanja podataka unutar instance na neku od prethodnih verzija baze podataka, čije je stanje zapamćeno u *snapshot*-u. Da bi se izvršio *restore*, sve što je potrebno jeste odabrati *Restore* opciju pored postojećeg *snapshot*-a. Ukoliko je neophodno sačuvati trenutno stanje baze podataka pre obavljanja *restore* operacije potrebno je izvršiti kreiranje novog *snapshot*-a same baze podataka, jer bi u suprotnom to trenutno stanje bilo nepovratno izgubljeno. Naime, *restore* operacijom se *override*-uju podaci unutar instance baze podataka.



Slika17a: Restore – Odabir Restore opcije konkretnog postojećeg snapshot-a baze podataka



Slika17b: Restore – Confirmation dialog Restore operacije



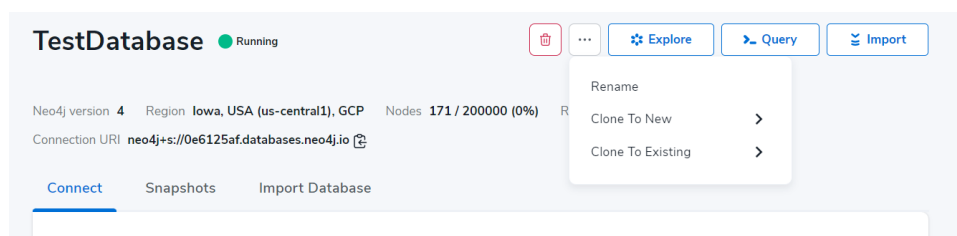
Slika17v: Restore – Restoring state

Po odabiru *Restore* opcije pored *snapshot*-a koji želimo da iskoristimo, otvara nam se *confirmation dialog* u okviru kojeg se od nas zahteva da unesemo *RESTORE* u odgovarajuće polje i tek nakon toga nam se *Restore* opcija uopšte dozvoljava. Po odabiru *Restore* opcije u *confirmation dialog*-u, instanca baze podataka prelazi u stanje *Restoring* dok se *Restore* operacija ne završi nakon čega se vraća u svoje regularno *Running* stanje.

4.7. UPRAVLJANJE INSTANCAMA

Moguće je izvršiti i određene osnovne operacije nad instancom baze podataka direktno iz *Neo4j Aura Console*-e.

Te osnovne operacije moguće je pronaći odabirom ... ikonice pored naziva same baze podataka.

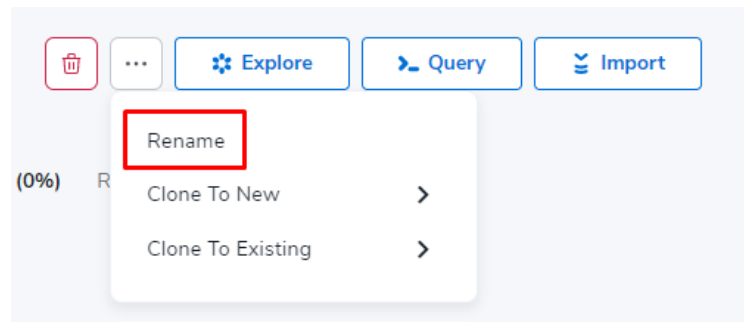


Slika18: Pristup osnovnim operacijama za upravljanje instancom baze podataka

4.7.1. RENAME

Moguće je izvršiti promenu naziva instance na sledeći način:

- Odabrati ... pored instance čiji je naziv potrebno promeniti
- Odabrati *Rename* opciju iz padajućeg menija
- Uneti novi naziv instance baze podataka
- Odabrati *Rename*



Slika19a: Rename – Odabir Rename opcije

Rename TestDatabase ×

Instance Name

TestDatabase

✕ Cancel Rename

Slika19b: Rename – Rezultat odabira Rename opcije

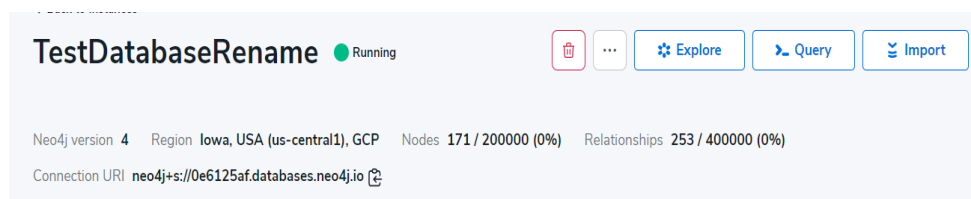
Rename TestDatabase ×

Instance Name

TestDatabaseRename

✕ Cancel Rename

Slika19v: Rename - Promena naziva instance baze podataka



Slika19g: Rename – Rezultat promene naziva instance baze podataka

4.7.2. RESIZE

Resize-ovanje instance dostupno je u *Professional* i *Enterprise subscription plan*-ovima, a sve što je potrebno uraditi da bi se izvršio *resize* instance je:

- Odabrati ... pored instance čiji je *resize* potrebno izvršiti
- Odabrati *Resize* opciju iz padajućeg menija
- Uneti novu vrednost za veličinu instance
- Selektovati *I understand checkbox*
- Kliknuti na *Submit*

Tokom operacije *resize*-ovanja instance, ona ostaje dostupna za korišćenje, bez obzira na to što se njena veličina menja u tom postupku.

4.7.3. PAUZIRANJE INSTANCE

U *Professional* i *Enterprise subscription plan*-u moguće je ručno pauzirati instancu baze podataka i to na sledeći način:

- Kliknuti na dugme za pauziranje instance koju je potrebno pauzirati
- Selektovati *I understand checkbox*
- Kliknuti *Pause*

Nakon toga, instanca se pauzira, a *pause* dugme menja *play* dugme.

Pre svega, pauzirana instanca ima manju stopu potrošnje od one koja je u radnom stanju, pa je zato najbolja praksa uvek je pauzirati kada je nije potrebno koristiti.

Kada je reč o *Free subscription plan*-u, on ne podržava mogućnost ručnog pauziranja instanci, međutim, ukoliko se instanca ne koristi 72 sata, ona se u tom slučaju automatski pauzira.

4.7.4. RESUME INSTANCE

Za ponovno pokretanje pauzirane instance potrebno je:

- Selektovati *play* dugme pored pauzirane instance koju je potrebno ponovo pokrenuti
- Selektovati *I understand checkbox*
- Kliknuti *Resume*

Nakon toga, instanca se ponovo pokreće.

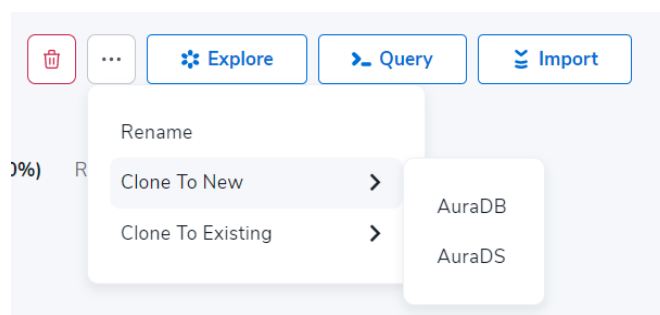
AuraDB Professional i *Enterprise subscription plan*-ovi dozvoljavaju da instanca u kontinuitetu bude pauzirana najviše 30 dana, nakon čega je automatski ponovo pokreću. To nije slučaj i u *Free subscription plan*-u. Naime, ukoliko se instanca baze podataka u *Free subscription plan*-u ne koristi više od 90 dana, ona se automatski briše i podaci se nepovratno gube.

4.7.5. KLONOVANJE INSTANCE

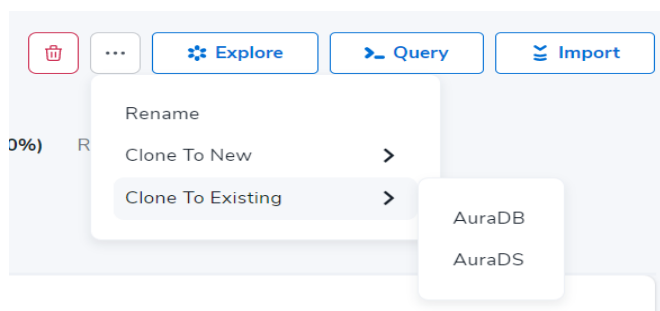
Postoji 4 načina na koje je moguće klonovati instancu baze podataka:

- Klonovanje u novu *AuraDB* instancu
- Klonovanje u novu *AuraDS* instancu
- Klonovanje u postojeću *AuraDB* instancu
- Klonovanje u postojeću *AuraDS* instancu

Jedina razlika između klonovanja u postojeću u odnosu na klonovanje u novu *Aura* instancu jeste u tome što bi u tom slučaju trebalo odabrati postojeću instancu u koju je potrebno klonovati. Takođe, u slučaju da neka od postojećih instanci ne zadovoljava uslove koji se tiču veličine same instance, one neće biti moguće za odabir. Takve instance imaju veličinu koja nije jednaka ili veća od veličine instance koju klonujemo.



Slika20a: Klonovanje instance – Odabir klonovanja u novu instancu

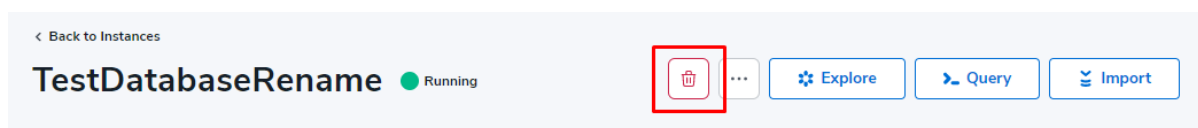


Slika20b: Klonovanje instance – Odabir klonovanja u postojeću instancu

4.7.6. BRISANJE INSTANCE

Moguće je izvršiti brisanje bilo koje instance u *Aura*-i, a sve što je potrebno je:

- Odabrati *trash* ikonicu pored instance koju je potrebno obrisati
- Uneti kompletan naziv instance koju je potrebno obrisati
- Odabrati *Destroy*



Slika21a: Brisanje instance – Odabir brisanja instance

Are you sure you want to destroy TestDatabaseRename? ×

This is irreversible. We will destroy your instance and all associated snapshots.

Are you sure?

Type "TestDatabaseRename" to c

✕ Cancel

🗑 Destroy

Slika21b: Brisanje instance – Rezultat odabira brisanja instance

Are you sure you want to destroy TestDatabaseRename? ×

This is irreversible. We will destroy your instance and all associated snapshots.

Are you sure?

TestDatabaseRename

✕ Cancel

🗑 Destroy

Slika21v: Brisanje instance – Kompletiranje confirmation dialog-a za brisanje instance

Jednom obrisanu *AuraDB* instancu nemoguće je povratiti ponovo.

4. ZAKLJUČAK

Neo4j AuraDB je *database-as-a-service cloud* rešenje koje se može bez ikakvih poteškoća koristiti za upravljanje *Neo4j* bazom podataka. Sama činjenica da uopšte ne postoji potreba za rešavanjem bilo kojih zadataka administrativne prirode umnogome umanjuje i olakšava posao koji se vezuje za menadžment baze podataka. Osim toga, *AuraDB* nudi jednostavno i vrlo intuitivno grafičko-korisničko okruženje koje od nas uglavnom ne zahteva ni duboko poznavanje *Cypher query* jezika da bismo mogli da ga koristimo.

Kako postoje tri *subscription plan*-a koje je moguće koristiti, tako u zavisnosti od funkcionalnosti koje je neophodno da zadovoljimo imamo šansu da odaberemo ono što bi nam najviše odgovaralo. Čak i osnovna, besplatna verzija ima ogromnu podršku i veliku količinu mogućnosti koje pruža, te nije pravilo da je neophodno platiti cenu za kvalitetan sistem. Takođe, i preostala dva *subscription plan*-a nude mogućnost da cena zavisi od ograničenja koja bi odgovarala sistemu koji se implementira.

Gotovo sve funkcionalnosti koje nudi klasičan sistem za upravljanje bazama podataka nudi i *cloud* sistem *Neo4j AuraDB*, a uz to nas oslobađa mnoštva odgovornosti i posla. Sve ovo savršeno ide u prilog generalno *database-as-a-service* sistemima i potpuno opravdava sve veću potrebu za rešenjima ovakvog tipa.

5. SPISAK SLIKA

Slika1: Kreiranje Neo4j Aura naloga/prijavljivanje na Neo4j Aura nalog

Slika2a: Kreiranje instance baze podataka – Odabir tipa instance

Slika2b: Kreiranje instance baze podataka – Popunjavanje osnovnih informacija o instanci

Slika2v: Kreiranje instance baze podataka – Odabir popune podataka u instanci baze podataka

Slika2g: Kreiranje instance baze podataka – Konačno kreiranje instance

Slika3: Generisani parametri o korisničkom imenu i password-u neophodnim za pristup i rad sa kreiranom instancom bazom podataka

Slika4: Kreirana instanca baze podataka u Setup fazi

Slika5: Kreirana instanca baze podataka u Running fazi

Slika6: Kreiranje Neo4j AuraDB Professional instance baze podataka i uticaj veličine instance na cenu

Slika7: Sidebar

Slika8a: Povezivanje sa instancom baze podataka korišćenjem Neo4j Browser-a – Pristup Neo4j Browser-u

Slika8b: Povezivanje sa instancom baze podataka korišćenjem Neo4j Browser-a – Popuna informacija za konekciju

Slika8v: Povezivanje sa instancom baze podataka korišćenjem Neo4j Browser-a – Ostvarena konekcija

Slika8g: Povezivanje sa instancom baze podataka korišćenjem Neo4j Browser-a – Neo4j Browser

Slika9a: Povezivanje sa instancom baze podataka pomoću Neo4j Bloom-a – Pristup Neo4j Bloom-u

Slika9b: Povezivanje sa instancom baze podataka pomoću Neo4j Bloom-a – Popuna informacija za konekciju

Slika9v: Povezivanje sa instancom baze podataka pomoću Neo4j Bloom-a – Neo4j Bloom

Slika10a: Povezivanje sa instancom baze podataka pomoću Neo4j CLI-a – Komanda

Slika10b: Povezivanje sa instancom baze podataka pomoću Neo4j CLI-a – Komande koje je moguće izvršiti nad instancom baze podataka iz CLI-a

Slika11a: Neo4j Browser – Database Information

Slika11b: Neo4j Browser – Favorites – Primer uvida u način na koji bi trebalo kreirati indeks Cypher upitom

Slika11v: Neo4j Browser – Uputstva

Slika11g: Neo4j Browser – Primer izvršenja upita za pribavljanje podataka i prikaz rezultata

Slika12a: Importovanje podataka – pristup Neo4j Data Importer-u

Slika12b: Importovanje podataka – konekcija sa instancom baze podataka

Slika13a: Importovanje podataka iz gotove Neo4j baze podataka – pristup Import Database

Slika13b: Importovanje podataka iz gotove Neo4j baze podataka – Import Database

Slika13v: Importovanje podataka iz gotove Neo4j baze podataka – Confirmation dialog

Slika13g: Importovanje podataka iz gotove Neo4j baze podataka – Upload complete

Slika13d: Importovanje podataka iz gotove Neo4j baze podataka – Loading state

Slika14a: Importovanje podataka iz gotove Neo4j baze podataka korišćenjem neo4j-admin push-to-cloud komande - Sintaksa neo4j-admin push-to-cloud komande

Slika14b: Importovanje podataka iz gotove Neo4j baze podataka korišćenjem neo4j-admin push-to-cloud komande – Uspešno izvršena komanda

Slika15a: Snapshot tab – Bez ijednog postojećeg snapshot-a

Slika15b: Snapshot tab – Sa postojećim snapshot-om

Slika16: Backup – Opcije

Slika17a: Restore – Odabir Restore opcije konkretnog postojećeg snapshot-a baze podataka

Slika17b: Restore – Confirmation dialog Restore operacije

Slika17v: Restore – Restoring state

Slika18: Pristup osnovnim operacijama za upravljanje instancom baze podataka

Slika19a: Rename – Odabir Rename opcije

Slika19b: Rename – Rezultat odabira Rename opcije

Slika19v: Rename - Promena naziva instance baze podataka

Slika19g: Rename – Rezultat promene naziva instance baze podataka

Slika20a: Klonovanje instance – Odabir klonovanja u novu instancu

Slika20b: Klonovanje instance – Odabir klonovanja u postojeću instancu

Slika21a: Brisanje instance – Odabir brisanja instance

Slika21b: Brisanje instance – Rezultat odabira brisanja instance

Slika21v: Brisanje instance – Kompletiranje confirmation dialog-a za brisanje instance

6. LITERATURA

- [1] - <https://neo4j.com/>
- [2] - <https://neo4j.com/docs/aura/>
- [3] - <https://www.techtarget.com/searchdatamanagement/definition/database-as-a-service-DBaaS>

