

UNIVERZITET U NIŠU  
ELEKTRONSKI FAKULTET

**Seminarski rad**

**Predmet: Sistemi za upravljanje bazama podataka**

**Cloud baze podataka i Database-as-a-service rešenja**

Student:

Dimitrije Mitić 822

Mentor:

Doc. dr Aleksandar Stanimirović

# Sadržaj

1. Uvod .....	3
2. Glavne karakteristike Cloud baza podataka .....	4
2.1 Tipovi <i>Cloud</i> baza podataka.....	4
2.2 Prednosti i nedostaci <i>cloud</i> baza podataka .....	5
3. ElephantSQL (PostgreSQL as a Service) .....	7
3.1 ElephantSQL planovi .....	7
3.2 Načini interakcije sa bazom .....	8
3.3 Dostupne mogućnosti ElephantSQL baze.....	11
4. Zaključak .....	17
5. Literatura .....	18

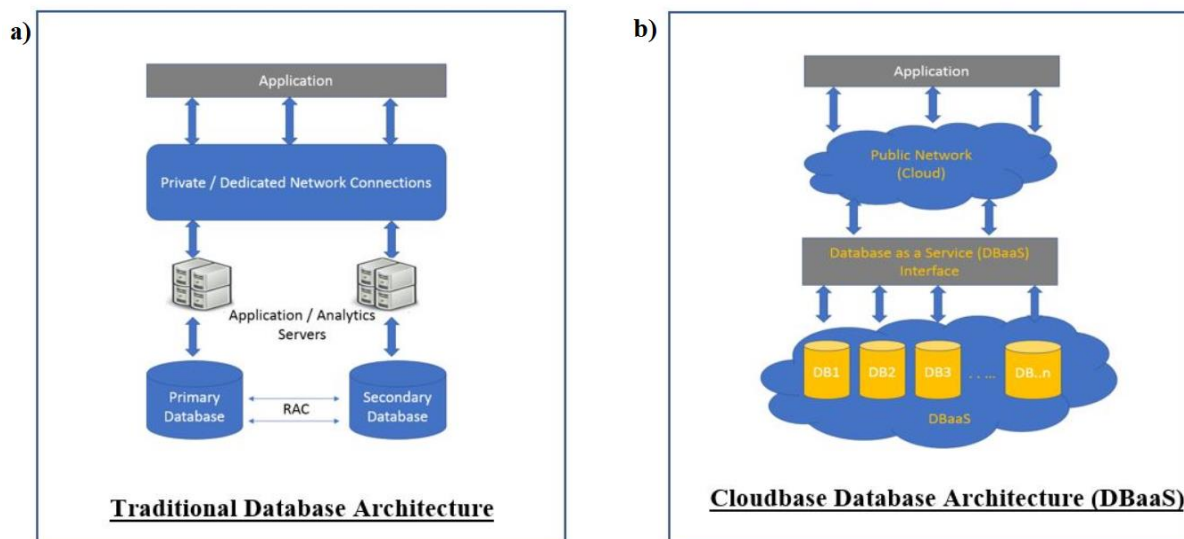
# 1. Uvod

Pojava i razvoj računarstva u oblaku omogućili su veću fleksibilnost i ekonomičnost kod razvoja aplikacija. Kompanije, uz pomoć *cloud*-a, ne moraju da troše sredstva na hardversku infrastrukturu i njeno održavanje, već plaćaju samo za uslugu prema upotrebi (*on demand*). Ovo je takođe omogućilo obezbeđivanje efikasne skalabilnosti – prema potrebi se lako može pribaviti potrebna količina resursa (*bandwidth*, prostor, procesorska moć itd). Još jedan benefit je i povećanje produktivnosti, iz razloga što *cloud* otklanja potrebu za raznim aktivnostima koje su vezane za održavanje sistema (postavka hardvera, *software patching*, itd), tako da se IT osoblje može orijentisati ka važnijim stvarima. Jedna od vrsta *cloud* servisa je ***Software as a service (SaaS)*** koja obezbeđuje korisnicima korišćenje aplikacije preko interneta, najčešće na osnovu pretplate (*subscribe*-a) [1]. Uz pomoć *SaaS*-a, *cloud* provajder hostuje i obezbeđuje softver kao i potrebnu infrastrukturu, dok se korisnik konektuje preko javne mreže na aplikaciju uz pomoć web pretraživača, tableta, PC-a i sl.

Jedna varijanta *SaaS*-a, kada je reč o bazama podataka je ***Database-as-a-Service (DBaaS)***. DBaaS omogućava korisnicima da, putem *cloud*-a, pristupaju i koriste bazu podataka, bez da poseduju svoj hardver (serveri i skладишта) i bez da instaliraju ikakav softver i održavaju samu bazu [2]. Za sve ovo je zadužen sam *cloud* provajder. Prilikom rada sa bazom na *cloud*-u, korisnik (aplikacija) ima iste mogućnosti, kada se radi o manipulaciji nad podacima, kao kada radi sa standardnom bazom koja je u lokalnoj mreži. Takođe *cloud* baze moraju da obezbede konstantnu dostupnost, zaštitu, kao i mehanizme za oporavak u slučaju otkaza hardvera. DBaaS baze su veoma dobro rešenje za manje i srednje kompanije, na taj način što održavanje i administracija baze prelazi u odgovornost provajdera, što bitno utiče na smanjenje troškova. Ovaj rad se bavi temom *cloud* baza podataka, tipovima istih, kao i arhitekturom, analizom prednosti i mana u odnosu na standardne (lokalne) baze, kao i konkretnom ElephantSQL bazom, koja predstavlja *DBaaS* varijantu PostgreSQL objektno-relacione baze.

## 2. Glavne karakteristike Cloud baza podataka

U standardnom okruženju, server baze podataka je deo lokalne računarske infrastrukture i njegovo pokretanje, konfigurisanje i održavanje spada u domen zaduženja IT osoblja (korisnika). Za razliku od ovog modela, kod *Cloud* baza podataka, korisnik koristi bazu kao uslugu (servis), obezbeđen od strane *cloud* provajdera, a ne kao produkt, što znači da instalacija, inicijalno konfigurisanje i održavanje baze ne spada u domen odgovornosti korisnika, već onog ko pruža uslugu. Na slici 1 dat je prikaz arhitekture standardne (1a) i *cloud* baze (1b). Na slici 1a aplikacioni i serveri baze su u vlasništvu kompanije, kao i sama konekcija između aplikacije i baze. Na slici 1b, imamo da aplikacija putem javne mreže (*Cloud-a*), putem odgovarajućeg API-a, komunicira sa bazom. Samo konfigurisanje instanci baze koje su na *cloud-u* se obavlja uz pomoć web konzole. Što se skalabilnosti tiče, različiti provajderi koriste različita rešenja. Neki koriste svojstvo *auto-scale* kojim se skalabilnost automatski prilagođava količini saobraćaja, dok kod drugih *cloud* baza, korisnik putem API-a može da utiče na skalabilnost [3].



Slika 1 Arhitektura a) Standardne b) Cloud Baze podataka [3]

U nastavku ovog poglavlja biće više reči o načinima na koji *Cloud* baza podataka može da bude implementirana, kao i o prednostima i manama korišćenja *Cloud* bazi.

### 2.1 Tipovi *Cloud* baza podataka

Postoje dva načina za pokretanje baze na *Cloud-u*. Prvi je preko *virtuelne mašine*, na taj način što korisnik zakupljuje virtuelnu mašinu od provajdera, na neko vreme, sa mogućnošću za korisnika da na istoj pokrene server baze. Kod ovakve vrste implementacije, korisnici koriste svoj *machine image* na kome instaliraju bazu, ili gotov *machine image* sa već instaliranom optimalnom bazom. Primer zadnjeg je Oracle koji obezbeđuje gotov *machine image* sa instaliranim Oracle Database 11g Enterprise Edition bazom koja se pokreće na Amazon EC2

*cloud* platformi. Kod ovakvog pristupa, provajder praktično obezbeđuje server (hardver), dok je korisnik zadužen za održavanje same baze [3][4].

Drugi način implementacije *cloud* baze je preko **baze kao servisa (DBaaS)**. Kod ovog pristupa korisnik dobija bazu u obliku servisa, što znači da korisnik ne mora da pokreće virtuelnu mašinu i da na njoj instalira i održava bazu. Sve ovo, za korisnika obezbeđuje provajder. Kod ovakvog pristupa korisnik plaća servis na osnovu toga koliko je koristio bazu. Primeri ovakve implementacije bili bi: Google's BigTable, Amazon Relational Database Service i Azure SQL Database [4].

Od baza koje su dostupne na *cloud*-u, neke su SQL bazirane, dok neke koriste NoSQL model. Relacione baze, koje su bazirane na strukturnom jeziku upita (SQL), su rigidno vezane za koncept tabele, u kojima su smešteni podaci u okviru redova i šemom definisanih kolona. Ovakav tip baza je idealan kada se radi o struktuiranim podacima i biznis transakcijama kao i podacima gde se traži visok nivo konzistentnosti istih (npr bankarske transakcije), razlog za ovo je taj da se relacione baze strogo pridržavaju ACID svojstva kada je reč o transakcijama. Primeri SQL baziranih baza na *cloud*-u bi bili: Amazon RDS, Google Cloud SQL, Oracle Database Cloud Service i dr.

NoSQL (*Not only SQL*) baze podataka nisu zasnovane na tabelarnom pamćenju podataka, što je glavna razlika u odnosu na relacione. Ove baze obezbeđuju fleksibilnu šemu za pamćenje sadržaja. U zavisnosti od baze, podaci mogu biti pamćeni u obliku dokumenta, ključ-vrednost parova, grafova itd. Ovakve baze su pogodne za pamćenje velike količine nestruktuiranih ili polustruktuiranih podataka. Primeri NoSQL *cloud* baza bi bili: Google Cloud Datastore, Amazon SimpleDB, MongoDB preko Amazon EC2 ili kao DBaaS.

## 2.2 Prednosti i nedostaci *cloud* baza podataka

Glavna prednost *cloud* baza podataka u odnosu na tradicionalne jeste velika **ušteta novca**, što je direktna posledica toga da kompanija ne mora da ima svoj *data* centar, na njemu instalirani *database* softver, kao i da održava isti, za šta bi bilo potrebno angažovanje dodatnog IT osoblja. Kod DBaaS rešenja, kompanija plaća samo uslugu na osnovu toga koliko je koristila bazu, što se meri najčešće u megabajtima ili gigabajtima. Takođe, provajder pruža mogućnost lakog praćenja upotreba baze i to sa različitih aspekta (utrošenog prostora, vremena korišćenja, potrošnje resursa itd) [1] [5].

Još jedna bitna prednost *cloud* baza jeste efikasnija **skalabilnost**, tj mogućnost da, kako raste količina podataka koji se pamte, da sistem automatski, ili na zahtev, angažuje nove čvorove za pamćenje istih. Kod DBaaS baza, skalabilnost se ostvaruje brže, jeftinije i efikasnije (uz par klikova) u odnosu na tradicionalne baze, kod kojih bi bila potrebna kupovina i podizanje novih servera. Šta više, DBaaS obezbeđuje svojstvo elastičnosti (*elasticity*) što podrazumeva automatsko obezbeđivanje dodatanih resursa kada imamo veće opterećenje sistema (npr veliki broj upita, ili je unet veliki broj redova), dok se čvorovi i ostali resursi konsoliduju kada je opterećenje na sistem manje [6].

Dodatna prednost *cloud* baza, je i to što je omogućena veća **dostupnost**. Za pristup *cloud* bazi jedino što je potrebno je pristup internetu, što praktično znači da je baza postala bilo kad i bilo gde dostupna. Ovo svojstvo stalne dostupnosti baze olakšava komunikaciju između korisnika na različitim lokacijama, čini bazu lakom za upotrebu, a takođe utiče i na povećanje produktivnosti [5].

S obzirom da se kod *cloud*-a, baza ne nalazi lokalno, kao kod tradicionalnih baza, ovo obezbeđuje eliminisanje rizika od *on premises* pretnji po bezbednost. Provajder je taj koji treba da garantuje zaštitu korisničkih podataka.

Kao posledica toga da se infrastruktura i sami podaci baze nalaze van domašaja korisnika (na *cloud*-u) dovodi do toga da je korisnik prinudjen da se osloni na provajdera i njegove prakse koje se tiču bezbednosti, ovo je jedan od glavnih razloga zašto neke kompanije imaju dileme kada je reč o migriranju baze na *cloud* platformu. Provajderi čuvaju podatke na različitim geografskim lokacijama, na deljenim medijumima, iz razloga performansi i redundantnosti. Takođe *cloud* baze obezbeđuju i automatske *backup*-ove i replike podataka smeštene na druge geo lokacije, za slučaj da dođe do pada sistema. Zbog svega ovoga postavlja se pitanje ko sve može imati pristup podacima korisnika, tj. može doći do narušavanja privatnosti istih. Za neke kompanije, ovo može biti u nesaglasnosti sa propisanim bezbedonosnim polisama [5] [7].

Još jedna od mogućih uskih grla, kada je reč o *cloud* bazama jeste i to da se celokupna komunikacija između korisnika i baze odvija preko interneta. U slučaju da je potrebno preneti velike količine podataka između baze i korisnika, podrazume se da korisnik ima internet sa adekvatnim *download/upload rate*-om, dok sa druge strane, kod tradicionalnih baza, komunikacija nije uslovljena brzinom interneta.

Na slici 2, kroz tabelu je dat pregled i poređenje *cloud* i tradicionalnih baza podataka.

Svojstva	Tradicionalna ( <i>On Premises</i> ) baza	<i>Cloud</i> baza/DBaaS
<b>Pouzdanost</b>	Pouzdana i privatna	Više pouzdana, ali ne nužno i privatna
<b>Skalabilnost</b>	Ograničena skalabilnost	Neograničena skalabilnost
<b>Brzina</b>	Brza, ali je moguća nedostupnost usled hardverskih otkaza	Brza, uvek dostupna
<b>Vreme za implementaciju</b>	Potrebno je više vremena da bi se implementirala	Minimalno vreme za implementiranje
<b>Isplativost</b>	Potrebno je dosta kapitala za pokretanje	Korisnik plaća po upotrebi – isplativiji pristup
<b>Održavanje</b>	Visoki troškovi održavanja (hardver, potreban softver, tehničari)	Nema troškova održavanja, korisnik plaća servis po upotrebi
<b>Sigurnost</b>	Visoka sigurnost i kontrola od strane korisnika. Moguća su <i>on premisses</i> narušavanja sigurnosti	Provajder garantuje sigurnost podataka. Korisnik nema kontrolu nad time ko sve ima pristupa bazi

Slika 2 Poređenje *cloud* i tradicionalnih baza [3]

### 3. ElephantSQL (PostgreSQL as a Service)

ElephantSQL predstavlja PostgreSQL *open-source* bazu hostovanu na *cloud*-u. Provajder ovog servisa upravlja administrativnim procesima vezanim za PostgreSQL kao što su: instalacija, *upgrade* do najnovije stabilne verzije, kao i upravljanje *backup*-ima [8]. ElephantSQL je takođe moguće integrisati sa nekoliko *cloud* aplikacionih platformi, što omogućava da korisnička aplikacija i baza mogu da se nađu na istom data centru. ElephantSQL podržava sledeće *cloud* platforme: Amazon Web Services (AWS), Google Cloud Platform, Microsoft Azure i IBM Cloud. Neke od karakteristika ElephantSQL-a su: konstantni monitoring nad serverima (*nodes*), backup na dnevnom nivou, mogućnost da se prilikom hardverskog ili softverskog otkaza, obrada automatski prebaci na drugi server (*Follower*) čija lokacija može biti i na drugom *cloud* provajderu, web browser alat za kreiranje SQL upita, što otklanja potrebu za instaliranjem alata poput pgAdmin-a ili psql-a, kao i veliki

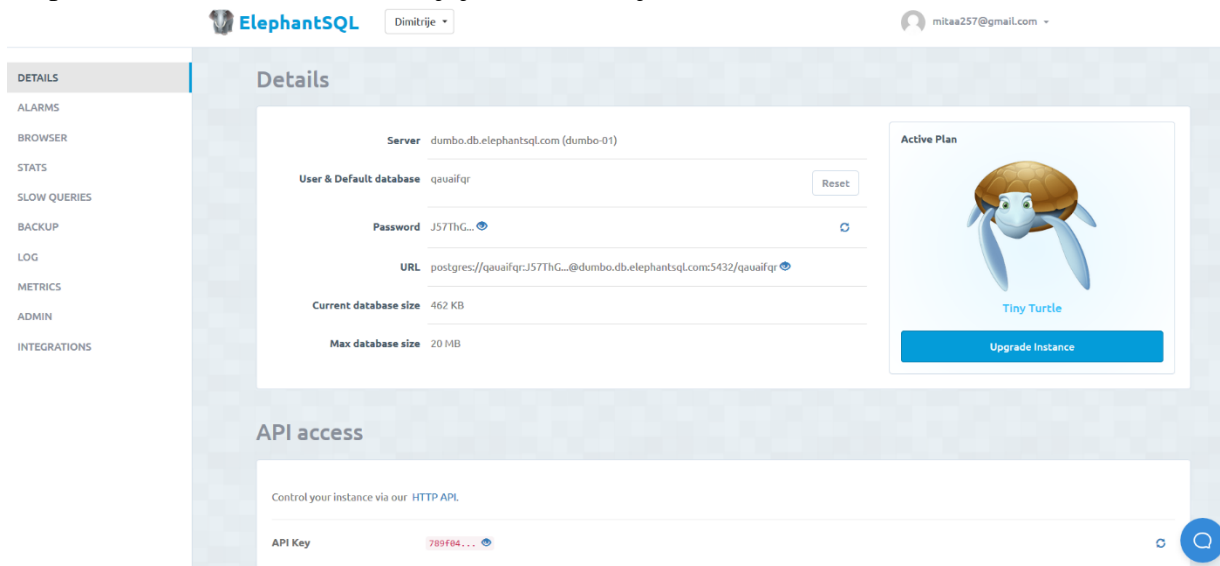
#### 3.1 ElephantSQL planovi

Prilikom zahteva za novom instancom servisa, korisnik specificira na koji plan korišćenja želi da se pretplati. Kao i većina DBaaS-a, korisnik plaća servis po iskorišćenosti, u slučaju ElephantSQL-a radi se o vremenskoj dostupnosti servisa, i plaća se na kraju svakog meseca. Planovi koje ElephantSQL nudi korisnicima su [9]:

- Deljivi planovi (*Shared Plans*) – obuhvata grupu planova kojima je zajedničko to da korisnik dobija jednu automatski prekonfigurisanu bazu, hostovanu na *cloud* provajderu specifikovanom od strane korisnika. Kod ovog plana, na jednom serveru se skladište baze više korisnika. Planovi u okviru ove grupe se razlikuju po tome, koliko je prostora na raspolaganju korisniku, kao i po tome koliko konkurentnih konekcija na instancu, plan podržava Deljivi planovi se preporučuje za manje projekte.
- Namenski planovi (*Dedicated Plans*) – kod ove grupe planova, korisnikova baza se smešta na poseban server koji je dodeljen isključivo tom konkretnom korisniku. Korisnik može da specificira *cloud* provajdera tj. data centar na kome želi da se nalazi njegova baza. Takođe, ovi planovi obezbeđuju znatno više raspoloživog prostora, veći broj konkurentnih konekcija, veći broj mogućnosti kada je reč o monitoringu kao i mogućnost da jedna instanca sadrži veći broj baza.
- Namenski planovi sa *Follower* opcijom (*Dedicated plans with Followers*) – ova grupa planova pruža iste opcije kao i namenski planovi, s tim što obezbeđuje dodatne mogućnosti koje se tiču bržeg oporavka baze kroz održavanje *read only* kopija baze (*follower*-i), koje ne samo da ubrzavaju upite čitanja, već služe i kao *stand-by*, spremne da preuzmu ulogu glavne baze, u koliko dođe do pada iste. *Follower* se može nalaziti i u nekom drugom geografskom regionu, u odnosu na lokaciju glavne instance, ili čak na drugom *cloud* provajderu.

U nastavku ovog rada, u primerima korišćenja ElephantSQL baze, kreirana je instanca sa deljivim planom *Tiny Turtle*, koji je besplatan, s tim da je slobodan prostor za bazu dodeljen korisniku 20 MB i moguće su 5 konkurentne konekcije sa instancom. Prilikom kreiranja

instance, korisnik specifikuje data centar (i region) gde želi da podigne instancu (deljivi planovi imaju manji izbor datacentara), s tim da u koliko se korisnička aplikacija koja komunicira sa bazom, nalazi na nekoj *cloud* platformi, poželjno je zbog latentnosti, da se baza i aplikacija nalaze na istom data centru. Nakon kreiranja instance, u *Details* prozoru, korisnik može da vidi informacije koje se tiču: servera na kome je instanca, plana pretplate, šifre, pristupnog URL-a, iskorišćenosti datog prostora i dr. Na slici 3 dat je prikaz *Details* tab-a, ElephantSQL web konzole, na kojoj se vide detalji kreirane instance.



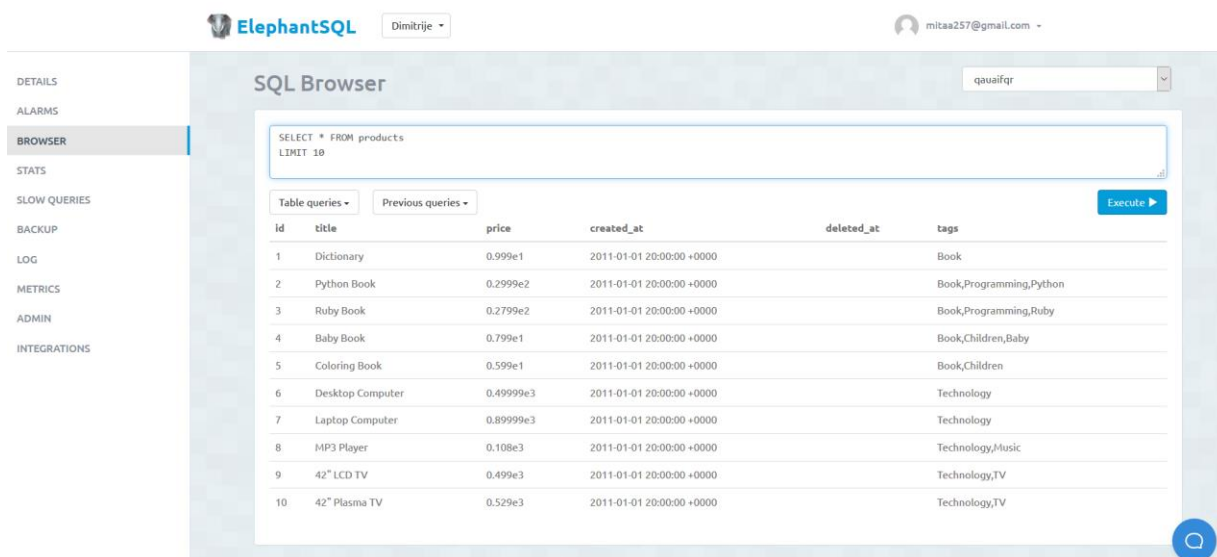
Slika 3 Detalji instance

Važno je napomenuti da je korisnik u mogućnosti da po želji promeni plan, ovo je moguće uz opciju fork (dostupna je na *dedicated* planovima), kojom se kreira kopija konkretne instance, pri čemu kopija može biti i na nekom drugom datacentru. Promena plana je takođe moguća i kreiranjem nove instance (sa novim planom) i uploadovanjem *backup*-a baze, sa stare baze na novu [10].

## 3.2 Načini interakcije sa bazom

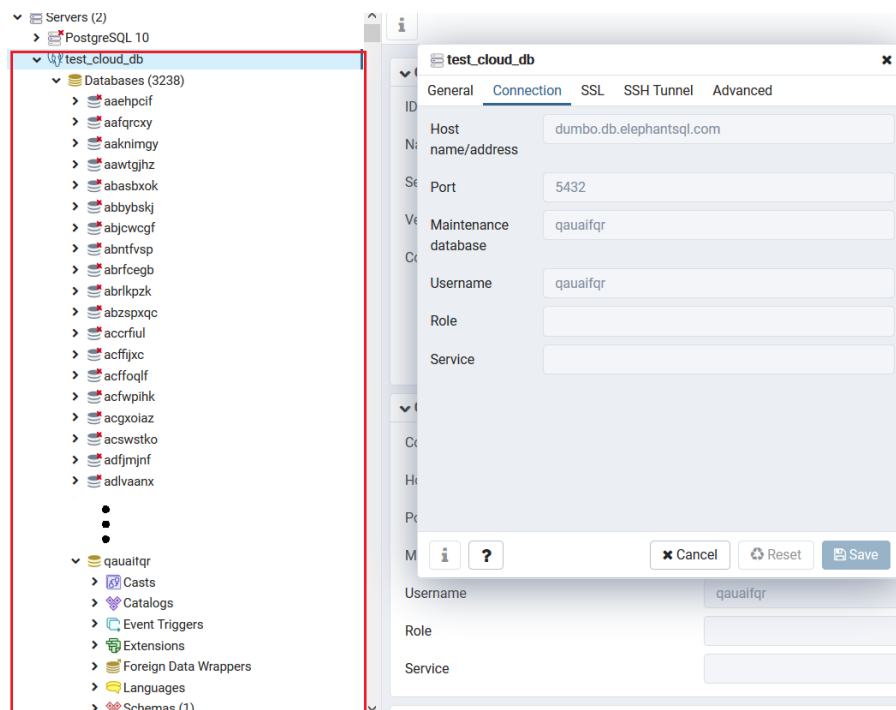
Za interakciju sa bazom, korisniku je na raspoalganju ElephantSQL web konzola, ali je pristup moguć i uz pomoć standardnih alata kao što su pgAdmin ili psql konzola. Web konzola ElephantSQL pruža razne alate za: kreiranje SQL upita, monitoring instanci, takođe pruža mogućnost korisniku za pravljenje *backup*-a i *follower*-a, uvid u razne statistike koje se tiču izvršavanja upita. Na slici 4 dat je primer izdavanja upita za pribavljanjem prvih 10 redova testne tabele *products*, uz pomoć ElephantSQL Browser-a.





Slika 4 Izdavanje upita korišćenjem ElephantSQL Browser-a

U koliko se na instancu konektujemo uz pomoć pgAdmin alata, može se videti da se na deljivom serveru (dumbo.db.elephantsql.com u ovom slučaju) nalaze baze podataka raznih korisnika, pri čemu je potrebno pronaći onu kojoj nam je pristup dozvoljen (quaifqr u našem slučaju). Ovo je posledica toga da *Tiny Turtle* plan, ne obezbeđuje korisniku namenski server, već održava njegovu bazu na zajedničkom serveru.



Slika 5 Konektovanje na bazu uz pomoć PgAdmin alata

Na slici 6 je dat primer interakcije sa bazom preko Python aplikacije. Bazi se, u ovom slučaju, pristupa uz pomoć `psycopg2` modula koji predstavlja jedan od popularnijih db adaptera za rad sa PostgreSQL bazama. Kao što se sa slike 6 vidi, sve što je potrebno da bi se pristupilo bazi, su informacije o serveru, *user*-u, kao i odgovarajuća šifra. Program sa slike, nakon što se konektuje na bazu, kreira indeks nad atributom *title*, za testnu tabelu *products* i upitom vraća sve indekse koji postoje za tu tabelu (rezultat su dva *touple*-a: automatski kreirani *products\_pkey*, i novokreirani *idx\_title*).

```
1  import os
2  import psycopg2
3
4  DB_NAME = 'qauaifqr'
5  DB_USER = 'qauaifqr'
6  DB_PASS = 'J57ThGh9pMptCE-pIIhHOhtZnnmbJpFA'
7  DB_HOST = 'dumbo.db.elephantsql.com'
8  DB_PORT = '5432'
9
10 try:
11     conn = psycopg2.connect(database = DB_NAME, user = DB_USER,
12                             password = DB_PASS, host = DB_HOST, port = DB_PORT)
13     print("Sucessfully connected !")
14     cur = conn.cursor()
15     cur.execute("""
16         CREATE INDEX idx_title ON products(title);
17         SELECT indexname FROM pg_indexes
18         WHERE schemaname = 'public' AND tablename = 'products'
19     """)
20     rows = cur.fetchall()
21     for row in rows:
22         print(row)
23     conn.commit()
24     conn.close()
25 except:
26     print("Couldn't connect to CloudDB")
27
```

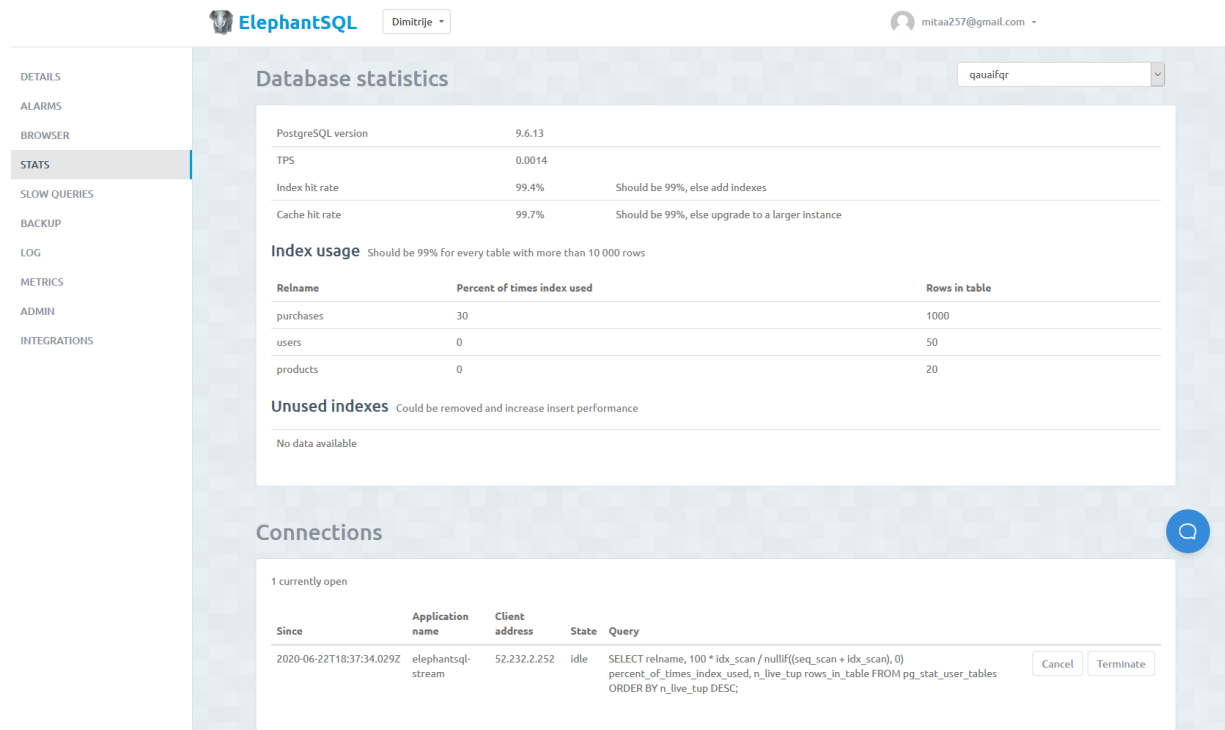
Slika 6 Primer interakcije Python aplikacije sa ElephantSQL bazom

Iz primera sa slika 5 i 6 se vidi da se korisnik na isti način konektuje i interaguje sa *Cloud* bazom kao da je u pitanju tradicionalna (lokalna), što u stvari i jeste jedna glavnih prednosti *cloud* baza, da pruža korisniku iste (i veće) funkcionalnosti, s tim da ga oslobađa od toga da isti vodi računa o hardverskom i softverskom podizanju i održavanju baze.

### 3.3 Dostupne mogućnosti ElephantSQL baze

Preko ElephantSQL konzole korisnik ima jednostavan pristup mnogim opcijama koje nisu dostupne kod klasičnih alata za pristup bazi. U ove opcije spadaju: praćenje raznih statistika vezanih za performanse izvršavanja upita, jednostavan način na uticanje na skalabilnost sistema, laka manipulacija *backup*-ima, praćenje performansi servera uz pomoć dijagrama i dr. U nastavku ovog poglavlja će biti detaljno opisane ove funkcionalnosti ElephantSQL-a.

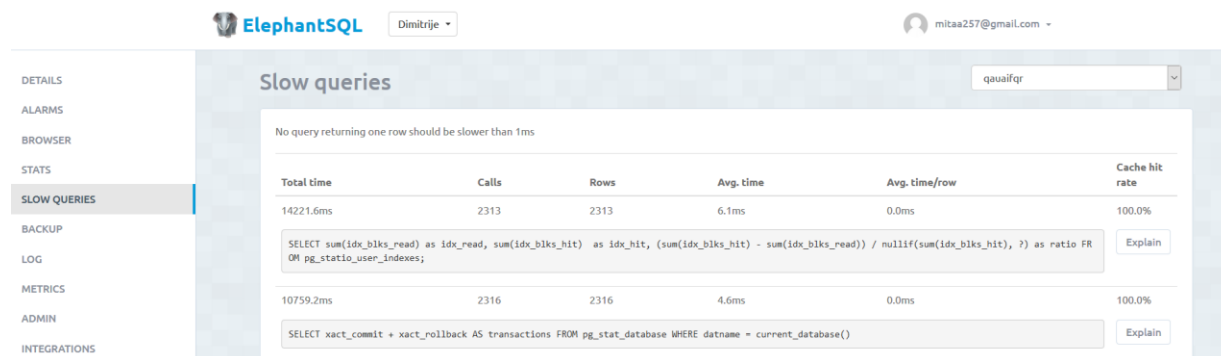
Preko *Stats* tab-a u okviru ElephantSQL web konzole, korisnik ima uvid u statistike koje se tiču izvršavanja upita, kao i informacije o trenutno aktivnim konekcijama sa instancom. Na slici 7 dat je prikaz statistike. Pored informacije o verziji PostgreSQL-a koja se koristi, ovaj tab nam daje statistike vezane za iskorišćenost indeksa kod pretraživanja podataka. Statistika *Index hit rate* pokazuje koliki procenat indeksa koji se koriste, se nalaze u baferu glavne memorije (ne mora im se pristupati na disku), dok *Cache hit rate* daje informacije o gneralnom procentu pogodaka kod pribavljanja podataka. *Index Usage* pokazuje, za svaku tabelu, kolika je u procentima iskorišćenost indeksa prilikom pretraga tih tabela. U koliko je ovaj broj mali, a tabela sadrži veliki broj redova, poželjno je kreirati indekse nad poljima nad kojima se često vrši pretraga. Takođe statistika pokazuje i kreirane, ali neiskorišćene indekse, čije bi se uklanjanje moglo razmotriti, da bi se ubrzale operacije *update*-a (upisa, izmene i brisanja redova) nad tom tabelom. U ovom tabu se takođe mogu videti i trenutno otvorene konkecije sa instancom. Ove konekcije se mogu po potrebi i zatvoriti.



Slika 7 Stats tab ElephantSQL Web Konzole

Korisnik može preko taba *Slow Queries* da ima uvid u vreme koje je bilo potrebno da bi se izvršili konkretni upiti. Ovaj tab omogućava identifikaciju upita koji se sporo izvršavaju.

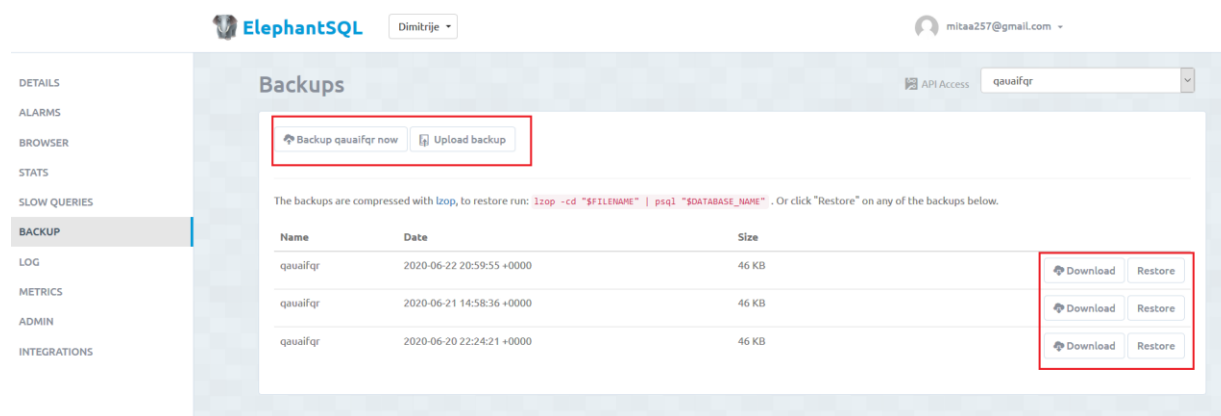
Ovakvi neefiksni upiti mogu bitno da utiču na performanse aplikacije koja koristi bazu. Na slici 8 dat je prikaz *Slow Queries* taba. Upiti na slici, čije je vreme izvršenja najduže, su implicitno izvršeni prilikom pregleda *Stats* taba, na osnovu njihovog rezultata je računata statistika vezana za indekse.



Total time	Calls	Rows	Avg. time	Avg. time/row	Cache hit rate
14221.6ms	2313	2313	6.1ms	0.0ms	100.0%
<pre>SELECT sum(idx_bkts_read) as idx_read, sum(idx_bkts_hit) as idx_hit, (sum(idx_bkts_hit) - sum(idx_bkts_read)) / nullif(sum(idx_bkts_hit), ?) as ratio FR OM pg_statio_user_indexes;</pre>					
10759.2ms	2316	2316	4.6ms	0.0ms	100.0%
<pre>SELECT xact_commit + xact_rollback AS transactions FROM pg_stat_database WHERE datname = current_database();</pre>					

Slika 8 *Slow Queries* tab ElephantSQL web konzole

Kada je reč o *backup*-ima, ElephantSQL automatski kreira *backup*-e baza na dnevnom nivou, za sve plaćene planove. Ovi *backup*-i se skladište na skladištu u oblaku (*cloud storage*) i uvek su dostupni korisnicima (nalaze se na istom *cloud*-u kao i korisnikova instanca). Za namenske servere se *backup* celokupnog servera radi svake nedelje, s tim da se *backup Write ahead Log*-a (*WAL*) uploaduje na Amazon S3 Object storage svakog minuta. Na slici 9 je dat prikaz *Backup* taba, na kome se mogu ručno kreirati *backup*-i baza, kao i učitavati isti iz fajlova sa sledećim ekstenzijama: .sql, .zip, .gz, .tar. Takođe na ovom tabu se nalaze informacije koje se tiču svih kreiranih *backup*-a koji su uploadovani na *cloud*. Svaki *backup* baze, korisnik može preuzeti u obliku *lzop* kompresovanog fajla, a takođe može uz pomoć *Restore* opcije odmah učitati isti.



Name	Date	Size
qauaifqr	2020-06-22 20:59:55 +0000	46 KB
qauaifqr	2020-06-21 14:58:36 +0000	46 KB
qauaifqr	2020-06-20 22:24:21 +0000	46 KB

Slika 9 *Backup* tab ElephantSQL konzole

Interakcija sa *backup*-ima baza je moguća i korišćenjem *Backup API*-a. Preko ovog API-a, korisnik može programski da kreira, lista i učitava *backup*-ove baza. Na slici 10 dat je primer korišćenja API-a za pribavljanje informacija o svim postojećim *backup*-ima baze „qauaifqr”.

Ovo je ostvareno preko GET zahteva API-u, koji u zaglavlju nosi API-Key vrednost (može se naći u *Details* tabu, slika 3) kao *Username* za autentifikaciju.

```
import requests
import json

API_KEY = '789f04fb-9d5f-4c50-9b32-7076ade71d4e'
BACKUP_URL = 'https://api.elephantsql.com/api/backup'

param = {'db':'qauaifqr'}
response = requests.get(BACKUP_URL, auth = (API_KEY, '') , params = param, verify=True)
print(response.json())
```

Slika 10 API zahtev za pribavljanjem informacija o svim backup-ima konkretne baze

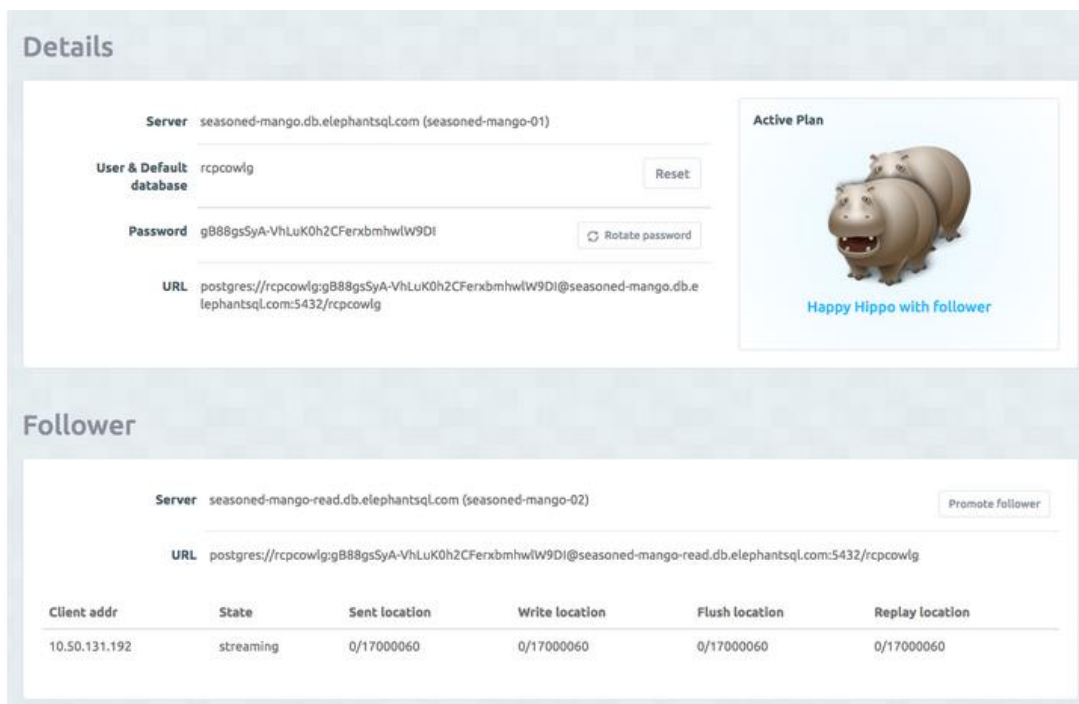
Kao odgovor na ovaj zahtev, dobijaju se JSON objekti sa informacijama za svaki od postojećih *backup*-a sa slike 9. Informacije se odnose na to: kada je *backup* kreiran, njegova lokacija, kao i link za preuzimanje *.lzo* fajla. Prikaz odgovora koji API vraća dat je na slici 11.

```
▼ 0:
  id: 45523891
  database_id: "851f8c67-45a1-49f9-8868-a8783e137606"
  backup_date: "2020-06-22 20:59:55 +0000"
  size: 47236
  bucket: null
  file_name: "dumbo/qauaifqr.2020-06-22T20:59:50+00:00.sql.lzo"
  region: "azure-arm:westeurope"
  database_name: "qauaifqr"
  url: "https://esqlwesteuropegr...Kaa5rGBHWJREmOV6e%2Fo%3D"
▼ 1:
  id: 45522491
  database_id: "851f8c67-45a1-49f9-8868-a8783e137606"
  backup_date: "2020-06-21 14:58:36 +0000"
  size: 47183
  bucket: null
  file_name: "dumbo/qauaifqr.2020-06-21T14:58:32+00:00.sql.lzo"
  region: "azure-arm:westeurope"
  database_name: "qauaifqr"
  url: "https://esqlwesteuropegr...ZwLxcoDwtJAU2HiLFT8uo%3D"
▼ 2:
  id: 45521129
  database_id: "851f8c67-45a1-49f9-8868-a8783e137606"
  backup_date: "2020-06-20 22:24:21 +0000"
  size: 47183
  bucket: null
  file_name: "dumbo/qauaifqr.2020-06-20T22:24:17+00:00.sql.lzo"
  region: "azure-arm:westeurope"
  database_name: "qauaifqr"
  url: "https://esqlwesteuropegr...gKJ4orDipZ3BzEBia%2BQ%3D"
```

Slika 11 API odgovor, lista kreiranih backup-a

Namenski (*dedicated*) planovi imaju i mogućnost *point-in-time* oporavka. Kod ove vrste oporavka, sve sheme i baze se brišu sa servera, nakon čega se oporavak baza vrši na osnovu zadnjeg *backup*-a pre odabranog datuma. Posle čega sledi prolazak i izvršenje konkretnih komandi iz WAL-a. *Backup*-i servera se takođe mogu koristiti prilikom migracije sa postojećeg na novu instancu. Ovo se obavlja tako što se kreira nova instanca sa željenim *dedicated* planom, nakon čega se importuje *backup* neke druge instance, za inicijalizaciju ove nove.

Što se skalabilnosti baza tiče, ElephantSQL sa uvođenjem koncepta ***Follower-a*** omogućava korisnicima da skladište svoje podatke na različitim datacentrima kao i na različitim *cloud* provajderima. *Follower* je *read-only* kopija glavne baze, s tim da u svako doba može da bude unapređena u *master* čime će moći da opslužuje zahteve za upisivanjem. Prilikom modifikovanja i komitovanja podataka na *master* bazi, promene se automatski strimuju do *follower-a*. *Follower*-i omogućuju ne samo bolje performanse preko balansiranja opterećenja kod *read* upita, već i *hot-standby* bazu, spremnu da preuzme ulogu *master* baze u slučaju otkaza sistema. *Follower* automatski biva unapređen u *master* bazu, u koliko *master* baza nije dostupna, pri čemu je vreme nakon koga se utvrđuje da je došlo do otkaza baze 30s. Takođe, korisnik sam može da unapredi *follower-a* za glavnu bazu. Na slici 12 dat je prikaz instance, koja koristi namenski plan, podignute na AWS *cloud* provajderu sa krieanim *follower*-om. Kod AWS-a, *follower*-i se fizički smeštaju u različite zone dostupnosti (*Availability Zones AZ*) da bi baza bila dostupna ako se desi otkaz na nivou celog *AZ*-a.



The screenshot displays the ElephantSQL console interface. The top section, titled 'Details', shows configuration for a server named 'seasoned-mango.db.elephantsql.com (seasoned-mango-01)'. It includes fields for 'User & Default database' (rcpcowlg), 'Password' (gB88gsSyA-VhLuK0h2CFerxbmhwIW9DI), and 'URL' (postgres://rcpcowlg:gB88gsSyA-VhLuK0h2CFerxbmhwIW9DI@seasoned-mango.db.elephantsql.com:5432/rcpcowlg). A 'Reset' button is next to the user field, and a 'Rotate password' button is next to the password field. To the right, under 'Active Plan', there is a cartoon hippo illustration and the text 'Happy Hippo with follower'. The bottom section, titled 'Follower', shows details for a follower server named 'seasoned-mango-read.db.elephantsql.com (seasoned-mango-02)'. It includes a 'Promote follower' button and a 'URL' field (postgres://rcpcowlg:gB88gsSyA-VhLuK0h2CFerxbmhwIW9DI@seasoned-mango-read.db.elephantsql.com:5432/rcpcowlg). Below this, a table displays follower status information.

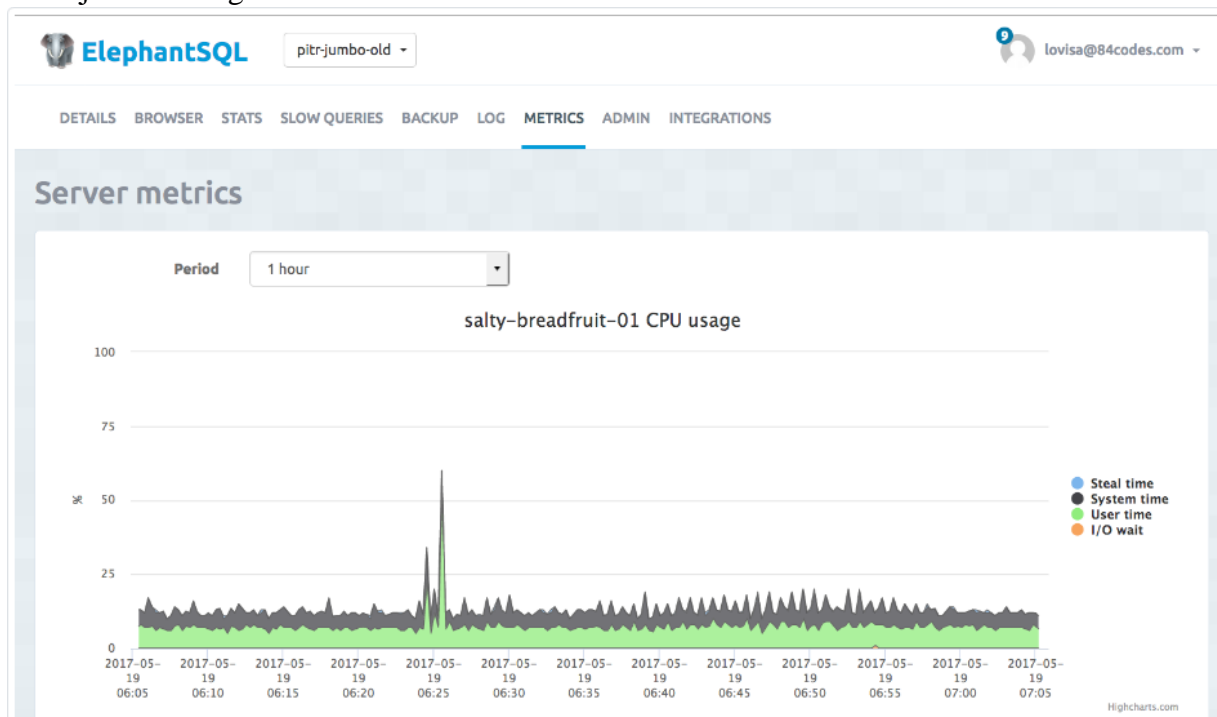
Client addr	State	Sent location	Write location	Flush location	Replay location
10.50.131.192	streaming	0/17000060	0/17000060	0/17000060	0/17000060

Slika 12 Details tab ElephantSQL konzole za instancu sa *dedicated* planom i detaljima o *follower*-u [11]

ElephantSQL obezbeđuje alate i za aktivan montiroing performansi servera. U ove alate spadaju alarmi, kao i razni dijagrami koji pokazuju iskorišćenost CPU-a i memorije servera na kome je baza. **Alarmi**, čije je korišćenje moguće samo za namenske planove, služe tome da korisnik dobija upozorenja prilikom pojava izvesnih anomalija vezanih za performanse servera

(CPU, RAM i *memory* alarmi). Alarmi mogu da šalju upozorenja na *email* adresu, preko *webhook*-a, na *PagerDuty* platformu, itd.

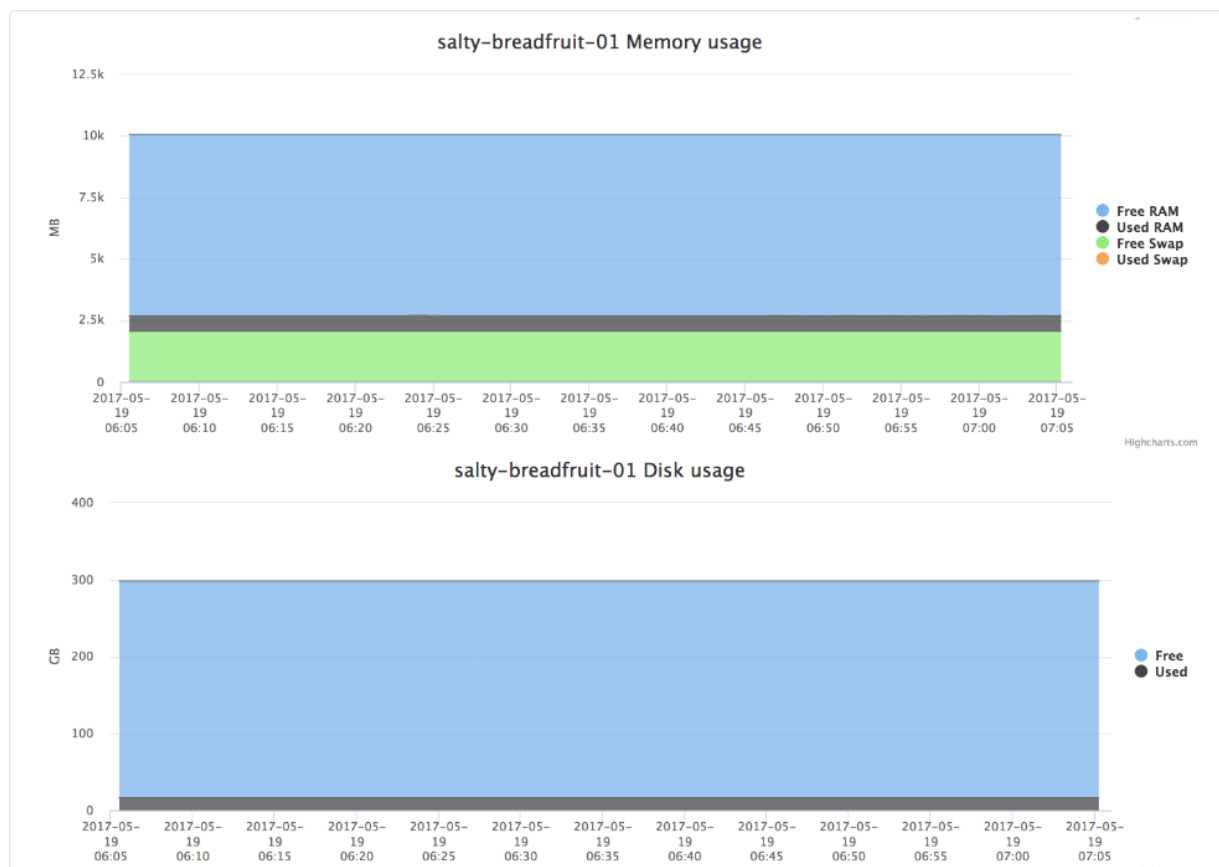
Uz pomoć **dijagrama performansi**, korisnik ima uvida u metriku performansi servera. Na slici 13 dat je prikaz metrike performansi CPU-a. Na ovom dijagramu je prikazana iskorišćenost procesorske moći kroz vreme i to sa osvrtom na sledeće aspekte: *user time* – koliko vremena CPU provede izvršavajući sam PostgreSQL task, *system time* - koliko vremena CPU provede izvršavajući taskove OS-a, *I/O* čekanje – vreme koje CPU provede u čekanju prilikom upisa i čitanja sa diska, *steal time* – vreme koje ode na virtuelizaciju tj. na čekanje virtuelnog CPU-a na stvarni.



Slika 13 Dijagram iskorišćenosti CPU-a [11]

Na slici 14 su prikazani dijagrami iskorišćenosti RAM memorije i slobodnog prostora na disku. Praćenjem dijagrama iskorišćenosti CPU-a i memorije, korisnik može da prati performanse aplikacije, kao i na koji način aplikacija iskorišćava dostupne resorse. Analizom korisnik može

da preduzme određene korake, kao što su podizanje novih *follower*-a ili migracija na plan sa više mogućnosti, u koliko dijagrami pokazuju veliko opterećenje ovih resursa.



Slika 14 Dijagram iskorišćenosti RAM-a i prostora na disku, respektivno [11]



## 4. Zaključak

*Cloud* baza podataka je kolekcija struktuiranih ili nestruktuiranih podataka, koji se nalaze na privatnoj ili javnoj *cloud* infrastrukturi. Bazu na *cloud*-u je moguće implementirati preko zakupa virtuelne mašine, na kojoj korisnik sam podiže odgovarajuću bazu, ili preko baze kao servisa (*DBaaS*), gde korisnici plaćaju korišćenje servisa baze koju obezbeđuje provajder tj. gubi se potreba za posedovanjem adekvatnog hardvera, kao i instaliranja i održavanja same baze. Prednosti *DBaaS*-a su: ušteda u troškovima – korisnik ne mora da ima infrastrukturu, da plaća softver i osoblje koje bi održavalo ceo sistem, efikasna *at a run-time* skalabilnost, kako vertikalana tako i horizontalna, veća dostupnost nego kod tradicionalna baza (potrebna je samo internet konekcija), ubrzan razvoj aplikacija, jednostavno upravljanje bazama (preko web browser konzole) i dr.

U ovom radu je kao primer *DBaaS*-a obrađen ElephantSQL hosting servis za PostgreSQL bazu. Ovaj servis automatizuje ceo proces postavljanja i izvršavanja PostgreSQL kalstera i dostupan je na većini većih *cloud* provajdera širom sveta. ElephantSQL ima široku ponudu kada je reč o servisima, počevši od baza na zajedničkom serveru (pogodne za manje projekte), sve do velikih baza, koje se nalaze na više namenskih servera (pogodne za veće kompanije koje insistiraju na boljim performansama). U slučaju hardverskog i softverskog otkaza glavnog čvora, obezbeđeno je automatsko prebacivanje opterećenja na *read only* redundantne čvorove – *follower*-e, s tim da ovi čvorovi mogu da se nalaze i na drugim geografskim regijama, kao i na drugim provajderima. ElephantSQL obezbeđuje i efikasan sistem automatskog, dnevnog, *backup*-ovanja baza sa servera na *cloud* skladište koje je korisniku stalno dostupno. Takođe je korisniku na raspolaganju i *point-in-time* oporavak, u koliko je potrebno vratiti bazu u neko stanje iz prošlog vremena. Još jedna bitna funkcionalnost obezbeđena od ElephantSQL jesu alati za monitoring baze, gde korisnik ima uvid u performanse servera (CPU i memorijska iskorišćenost), koji upiti zahtevaju najviše vremena, uvid u to da li je potrebno kreiranje ili uklanjanje indeksa radi ubrzavanja pretraga, kao i informacije o otvorenim konekcijama sa bazom. Interakcija sa bazom je moguća uz pomoć *user-friendly* browser konzole, ali je isto moguće i preko standardnih alata.

## 5. Literatura

- [1] <https://azure.microsoft.com/en-us/overview/what-is-cloud-computing/#benefits>
- [2] <https://www.ibm.com/cloud/learn/dbaas>
- [3] <https://labs.sogeti.com/cloud-database-dbaas-database-as-a-service/>
- [4] [https://en.wikipedia.org/wiki/Cloud\\_database](https://en.wikipedia.org/wiki/Cloud_database)
- [5] <https://www.red-gate.com/simple-talk/cloud/cloud-data/designing-highly-scalable-database-architectures/>
- [6] Al Shehri, Waleed. (2013). Cloud Database Database as a Service. International Journal of Database Management Systems. 5. 1-12. 10.5121/ijdms.2013.5201.
- [7] <https://www.stratosphenetworks.com/advantages-and-disadvantages-of-cloud.html>
- [8] <https://www.elephantsql.com/docs/index.html>
- [9] <https://www.elephantsql.com/plans.html>
- [10] <https://www.elephantsql.com/docs/faq.html>
- [11] <https://www.elephantsql.com/product-tour.html>