Primena serverles SQL tehnologije Azur Sinaps Analitiksa u analitici podataka

Strahinja Rodić, Dušan Vujošević [a]

***Sadržaj* — Razvijanje informacionih sistema i tehnologija omogućilo je stručnjacima da intenzivno koriste udaljene računarske resurse prema konceptu računarstva u oblaku. Između ostalog, došlo je i do razvoja baza podataka u oblaku, kao i prebacivanja skladišta podataka na oblak. Majkrosoftov Azur je jedan od vodećih sistema za računarstvo u oblaku, ima podršku za mnoge baze podataka, a jedna od platformi dostupnih na njemu je i Azur Sinapsa Analitiks.**

U ovom radu dat je kratak kritički pregled Azur Sinapsa Analitiksa, sa akcentom na njegov sistem distribuirane obrade podataka, serverles SQL pul. Ovaj sistem omogućava inženjerima podataka, arhitektama podataka, analitičarima podataka, poslovnim analitičarima i naučnicima da kreiraju virtuelna skladišta podataka, da istražuju velike podatke dostupne na Azurovim jezerima podataka, da izvršavaju kompleksne upite i kreiraju poslovne izveštaje.

Ovim radom čitalac se želi upoznati sa osnovama korišćenja prikazanih tehnologija. S tim ciljem na umu razvijeno je mnoštvo primera, a sagledane su i njihove prednosti, i mane.

***Ključne reči* — SQL, veliki podaci, analitika podataka, serverles, Azur, skladišta podataka, obrada podataka**

I. UVOD

U ovom radu opisana je Azur Sinapsa Analitiks (poznata i kao Azurova Sinapsa za analitiku, ili, kraće, Sinapsa, na eng. *Azure Synapse Analytics*), jedno od rešenja koje različiti softverski inženjeri, inženjeri podataka (eng. *Data Engineers*), naučnici specijalizovani za obradu podataka (eng. *Data Scientists*), poslovni analitičari i drugi mogu da koriste. U prvom delu tada biće dat pregled Sinapse, njenih komponenti, pregled Sinapsinog studija i mogućnosti rada sa praktičnim primerima. U drugom delu rada biće opisani serverles SQL pul, njegovi bitni koncepti i atributi, glavni scenariji i alati koji se služe za korišćenje serverlesa. Takođe, biće prikazano nekoliko praktičnih primera kako bi se ilustrovalo opisano. Na kraju rada će biti

[a] Strahinja Rodić, Računarski fakultet, Srbija (email: strahinja.rodic@hotmail.com).
Dušan Vujošević, Računarski fakultet, Srbija (telefon: 381-11-2627613; faks: 381112623287; email: dvujosevic@raf.rs).

prikazan zaključak autora, njihovo mišljenje, primedbe, pohvale, sugestije i potencijalni predlozi za unapređenje posmatranih tehnologija.

U radu su rezervisane reči sintakse T-SQL jezika, kada su napisane u samom tekstu, a ne u odvojenim ćelijama namenjenim za kod, markirane sivom bojom kako je i praksa u oficijelnoj Majkrosoftovoj dokumentaciji. Primeri ovog pristupa oblikovanju teksta su reči: SELECT, OPENROWSET.

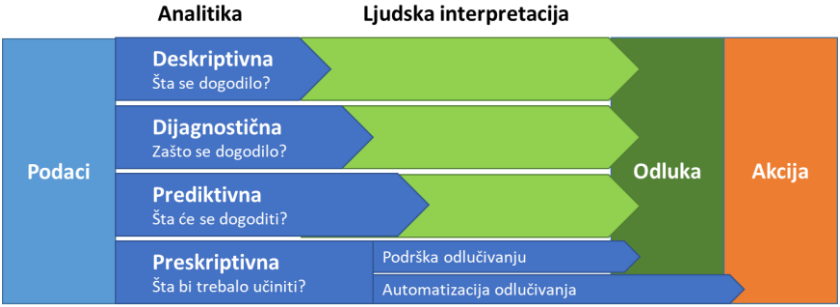
II. Platforma Azur Sinapsa Analitiks

Sinapsa je proizvod koji je razvio Majkrosoft za potrebe analiziranja i obrađivanja velikih podataka. Ova platforma objedinjuje integraciju podataka, skladištenje podataka i obradu velikih podataka. Korisnik može da bira da li želi da koristi namenske resurse (eng. *Dedicated*) ili serverles (eng *Serverless*), što bi značilo bez zakupa namenske mašine, koristi servis po potrebi, to jest na zahtev. Sinapsa omogućava unificirano rešenje za ubacivanje, pretraživanje, pripremanje, obradu, upravljanje i serviranje podataka za primenu u poslovnoj inteligenciji ili mašinskom učenju[1].

Relaciona skladišta podataka se već dugo koriste, kao popularno rešenje u biznisu i preduzećima, za obradu podataka zajedno sa drugim popularnim multidimenzionim rešenjima za poslovnu inteligenciju, kao što su *PowerBI* i *Tableau*[2]. Sinapsa podržava različite tipove analitike, a osnovni tipovi su[2]:

- deskriptivna analitika,
- dijagnostična analitika,
- prediktivna analitika i
- preskriptivna analitika.

U kakvoj povezanosti su podaci, podržane analitike, očekivani intenzitet interpretacije od strane korisnika, odluke i akcije u zavisnosti od tipa analitike može se videti na slici 1 ispod, urađenoj prema[2].



Slika 1 – Odnos automatizovanog i neautomatizovanog odlučivanja u zavisnosti od tipa analitike, prema [2]

A. Pregled komponenti Sinapse

Da bi korisnik mogao da koristi Sinapsu potrebno je da prvo kreira radno okruženje. **Radno okruženje** predstavlja centralnu lokaciju gde se mogu pregledati sve informacije o resursima unutar Sinapse. Prilikom kreiranja Sinapsinog radnog okruženja može se kreirati i Azurovo jezero podataka druge generacije ili odabrati neko već postojeće, kako bi ono moglo da se koristi kao primarno skladište podataka i kontejner gde se čuvaju podaci radnog okruženja. U radnom okruženju postoje dostupni servisi kao što su Sinapsin SQL, Apačeov Spark i Sinapsini pajplajnovi za integraciju podataka.

Sinapsin SQL omogućava implementaciju skladišta podataka i virtuelizaciju podataka kao logičkog skladišta podataka (eng. *Logical Data Warehouse*). Podaci u skladištu podataka se čuvaju u trajnim tabelama koje se mogu popuniti ETL procesima pomoću Sinapsinih pajplajnova ili Azurovom fabrikom podataka. U ovom slučaju korišćenja potrebno je da korisnik bude upoznat sa podacima koje izvlači iz izvora podataka, kako će da ih dovuče u skladište podataka i kako će da ih obradi pre konzumiranja[13].

Virtuelizacija podataka omogućava interakciju sa podacima pri kojoj nije potrebno razumeti kako su podaci struktuirani ili formatirani. To omogućava istraživanje podataka što može biti veoma korisno prilikom dijagnostičke analitike, kada je bitno da se podacima što pre pristupi, da se sagleda šta se u njima nalazi. To, takođe, omogućava spontano (eng. *Ad Hoc*) pristupanje podacima i pripremanje istih, gde analitičari prvobitno žele da učitaju sirove (eng. *Raw*) podatke pre nego što formalno podese skladište podataka. Podaci se mogu čitati u sirovom formatu i učitati u ADLS odakle se podaci mogu čitati i istraživati dosta brzo, pre nego što se korisnici odluče za dalje obrađivanje i pripremu podataka [2].

Kako bi se ova dva slučaja korišćenja iskoristila na najbolji mogući način Sinapsin SQL omogućava dva različita modela:

- namenski model (eng. *Dedicated*) i
- serverles model (eng. *Serverless*).

Namenski model se takođe naziva i namenskim SQL pulom. Referencira funkcionalnosti skladišta podataka i predstavlja kolekciju analitičkih resursa koji su dostupni prilikom korišćenja Sinapsinog SQL-a. Kada je korisniku potrebno da ima predvidljive performanse i trošak za procesuiranje podataka trajno skladištenih u SQL tabelama u skladištu podataka kreiranje namenskih SQL pulova je najbolja opcija[2].

Serverles model/serverles SQL pul je, nasuprot toga, idealan za neplanirane, improvizovane zadatke, koji služe za neku dijagnostičku analizu podataka. Na primer, serverles model je bolji izbor za istraživanje podataka ili pripremanje podataka za virtuelizaciju[2].

Apače Spark je softver otvorenog koda (eng. *Open-Source software*) koji predstavlja distribuirani sistem za paralelno procesuiranje velikih podataka. Najpre služi za zadatke koji su preveliki ili previše kompleksni da bi ih tradicionalne baze podataka mogle obraditi. Apače Spark procesuira velike količine podataka u memoriji, što povećava performanse analiziranja i čini ga efikasnijim. Ovo procesuiranje dostupno je unutar Sinapse pod nazivom Apače Spark pul (u daljem tekstu Spark pul)[2].

Prilikom kreiranja Spark pula u Sinapsinom studiju, dobija se na korišćenje i klaster. Klaster je grupa kompjutera koji se tretiraju kao jedinstven računar i izvršavaju različite komande koje se pokreću pomoću radnih svezaka (eng. *Notebooks*) dostupnih u okviru Sparka. Klasteri omogućavaju paralelizovano procesuiranje podataka na više različitih računara kako bi poboljšali performanse. Sastoje se iz Spark drajvera (eng. *Driver*) i radnih čvorova (eng. *Worker nodes*). Drajver pošalje posao radnim čvorovima i instrukcije kako da izvuku i obrade podatke.

Spark pulovi u Sinapsi omogućavaju potpuno upravljane Spark servise. Neke od koristi od kreiranja Spark pula u Sinapsi su[2]:

- **Odabir performansi** – korisnik može da bira koliko želi da ima čvorova (eng. *Nodes*) Spark pul i srazmerno tome dobija i količinu memorije dostupnu za učitavanje i obrađivanje podataka.
- **Brzina i efikasnost** – ukoliko se pokreće instanca sa manje od 60 čvorova, biće dostupna u roku od 2 minuta otprilike, a ukoliko se kreira instanca sa više od 60 čvorova onda u roku od 5 minuta. Instance se pauziraju nakon isteka *x* minuta nekorišćenja, gde je *x* broj minuta koje korisnik može da definiše.
- **Jednostavno kreiranje** – Spark pul može da se kreira u okviru Sinapse direktno u Sinapsinom studiju, pomoću Azurove komandne linije (eng. *Azure PowerShell*) ili pomoću SDK-a Sinapse.
- **Jednostavno korišćenje –** Sinapsa uključuje radne sveske koje mogu da se koriste za interaktivno procesuiranje i vizuelizaciju podataka.
- **Skalabilnost** – Spark pulovi mogu da se automatski skaliraju dodavanjem ili gašenjem čvorova u pulu po potrebi sistema. Takođe, Spark pulovi se mogu i u potpunosti ugasiti ili obrisati, jer se podaci, koji se obrađuju, čuvaju na Azurovom skladištu ili na ADLS gen2.

Korisnici Sinapse zahvaljujući Apače Spark pulu mogu da implementiraju različita rešenja za inženjering velikih podataka i mašinsko učenje u okviru Sinapse. Ovo se može iskoristiti kao prednost za računanje kompleksnih transformacija. Za mašinsko učenje u okviru Sinapse mogu se koristiti SparkML i AzureML[2]. Apače Spark pul u potpunosti podržava implementacije rešenja u sledećim programskim jezicima[2]:

- *Scala,*
- *Python,*
- *SparkSQL i*
- *C#.*

U okviru Sinapse postoji Sinapsini Pajplajnovi koji omogućavaju funkcionalnosti Azurove fabrike podataka (eng. *Azure Data Factory*) u okviru Sinapse. To je zapravo klaud servis koji omogućava ETL procese, pomeranje i transformaciju podataka[2].

Korišćenjem Pajplajnova korisnik može da kreira i zakazuje tokove posla koji importuju podatke iz različitih izvora. Mogu se kreirati kompleksni ETL ili ELT procesi koji omogućavaju transformaciju podataka vizuelno ili pomoću servisa kao što su Azure Databricks, Azure HDInsight ili Sinapsa. Pajplajnovi u okviru Sinapse omogućavaju integraciju između (namenskih) SQL pulova, serverles SQL pula i Spark pulova[2].

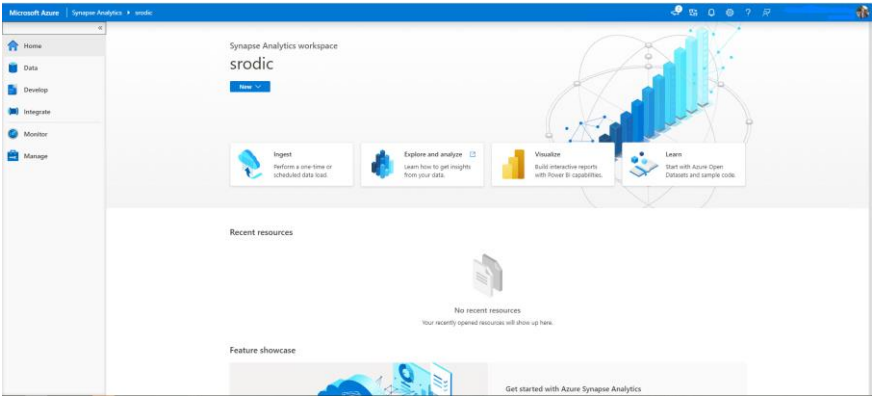
Podrška za različite izvore podataka omogućena je pomoću funkcionalnosti povezani servisi (eng. *Linked Services*), koja omogućava uvoz podataka radi pripreme za transformaciju ili analizu. Podaci importovani iz povezanih servisa mogu da se predstave kao objekti skupova podataka (eng. *Datasets objects*), a koji predstavljaju strukturu podataka na koju referencira povezani servis. Mogu ih koristiti ADF objekti tipa aktivnosti (eng. *Activity*)[2].

Aktivnosti uglavnom sadrže transformacionu logiku posla. Ona uključuje kopiranja pomoću kojih se importuju podaci iz različitih izvora, takođe može uključiti i mapiranje tokova podataka pomoću kojih se izvršavaju vizuelne transformacije bez kodiranja. Aktivnosti takođe mogu da uključuju izvršavanje sačuvanih procedura (eng. *Stored procedures*), i slično[2].

Može se organizovati više aktivnosti koje zajedno sarađuju, na primer ubacivanje podataka, potom izvršavanje sačuvanih procedura i na kraju pokretanje modela mašinskog učenja za analizu. Ovako organizovane aktivnosti nazivaju se pajplajnovima. Pajplajnovi mogu biti zakazani ili okinuti (eng. Trigger) u zavisnosti od toga kad je izvršavanje istih potrebno[2].

B. Pregled Sinapsinog studija

Sinapsin studio je glavni alat za interakciju sa mnogim komponentama u okviru Sinapse. Podeljen je u više različitih odeljaka koji se mogu videti sa leve strane korisničkog interfejsa. Primer kako ovaj interfejs izgleda može se videti na slici 2, dobijenoj na osnovu izvora [3].



Slika 2 - Pregled Sinapsinog studija

Na početnoj strani može se videti pregled poslednje korišćenih resursa, kao i neke prečice do određenih funkcionalnosti koje pruža Sinapsa. Sa leve strane u meniju mogu se videti različite sekcije u okviru Sinapse, a to su:

- sekcija za podatke (eng. *Data Hub*),
- sekcija za razvoj (eng. *Develop*),
- sekcija za integraciju (eng. *Integrate*),
- sekcija za monitoring (eng. *Monitor*) i
- sekcija za upravljanje (eng. *Manage*).

U **sekciji za podatke** može se pristupiti bazama podataka namenskog SQL pula, serverles SQL pula i Sparka, kao i eksternim izvorima kao što su nalozi skladišta podataka, te drugim servisima povezanim sa radnim okruženjem.

U **sekciji za razvoj** mogu se kreirati nove skripte i funkcionalnosti pomoću kojih se mogu izvršavati različite analitike i obrade podataka u okviru Azurove Sinapse. Odabirom dugmeta + (eng. *Add*) mogu se dodati Spark radne sveske (eng. *Notebooks*), SQL skripte, tokovi podataka (eng. *Data flow*) i Spark definicije posla (eng. *Apache Spark job definition*). U ovom poglavlju ćemo se fokusirati samo na radne sveske i SQL skripte.

Postoji opcija da se kreiraju radne sveske (eng. *Notebooks*), što je poznato i standardno radno okruženje većini naučnika specijalizovanih za obradu podataka, inženjerima mašinskog učenja i analitičarima velikih podataka. Ove radne sveske omogućavaju pisanje upita i manipulacijom nad podacima u okviru Sparka, pošto su one takozvane Spark radne sveske. U Spark radnim sveskama može se birati između različitih jezika koji žele da se koriste, kao što su *PySpark* (Pajton), .NET Spark (C#), Spark Scala i Spark SQL. Spark radne sveske se povezuju direktno sa Spark pulovima i pomoću tih pulova i

njihovih performansi može se izvršiti analitika nad podacima, kreiranje eksternih tabela i slično.

Pored Sparkovih radnih svezaka, moguće je pisati i T-SQL skripte u okviru razvojnog dela Sinapsinog studija. SQL skripta može da se poveže na različite pulove i izbor pula može da se odredi iz padajućeg menija pored labele *Connect to*, a može se menjati i baza nad kojom želimo da izvršavamo upit odabirom iz padajućeg menija pored labele *Use database*.

Sekcija za integraciju služi korisniku da upravlja Pajplajnovima. Pajplajnovi i aktivnosti su zasnovani na Azurovoj fabrici podataka (eng. *Azure Data Factory - ADF*). Veliki podaci zahtevaju servis koji omogućava upravljanje i orkestraciju procesima koji mogu da transformišu velike podatke u korisne biznis informacije. ADF je klaud servis koji je napravljen za orkestraciju ETL i ELT procesa[2].

Sekcija za monitoring služi da se pregledaju različita izvršavanja pajplajnova, SQL skripti, Apače Spark poslovi i druge aktivnosti. Sekcija za monitoring je polazna stanica kada treba da se debaguju problemi ili kada je potrebno imati uvid u potrošnju resursa. Ovde se može videti lista svih aktivnosti u radnom okruženju, kao i koja su trenutno aktivna[2].

Sekcija za upravljanje, kao što joj i ime kaže, služi za upravljanje različitim funkcionalnostima u okviru Sinapse. Ovde se mogu kreirati, brisati ili pauzirati/ponovo pokretati SQL i Spark pulovi. Pored toga, moguće je uvezati druge servise sa Sinapsom, kreirati ili brisati trigere i integracije, upravljati pravima pristupa i kredencijalima, kao i povezati različite biblioteke i konfigurisati Git[2].

III. SISTEM DISTRIBUIRANE OBRADE PODATAKA SERVERLES SQL PUL

Nakon što su opisane najvažnije komponente Sinapse i Sinapsin studio, u ovom delu rada biće opisan serverles SQL pul: kako funkcioniše, čemu služi i kako se može koristiti. Biće dati i gotovi primeri za obrađivanje podataka.

Svako radno okruženje Sinapse dolazi sa već predefinisanim i konfigurisanim serverles SQL pulom. On ima svoju pristupnu tačku na koju se korisnici povezuju kako bi mogli da izvršavaju SQL upite nad svojim podacima pomoću ovog pula. Pomoću ovog pula korisnici mogu da analiziraju podatke koji se nalaze na Azurovim jezerima podataka, na *Cosmos DB* bazi ili *Dataverse*-u[4]. Serverles SQL pul je servis za izvršavanje upita nad podacima pomoću različitih funkcionalnosti:

- Može se koristiti standardna T-SQL sintaksa za izvršavanje upita nad podacima, bez potrebe da se podaci kopiraju ili unesu u neku specijalizovanu bazu.

- Integrisana je povezivost sa drugim servisima koji služe za različite potrebe poslovne inteligencije i spontano (ad hok) izvršavanje upita nad podacima pomoću najpopularnijih drajvera za SQL server.

T-SQL (eng. *Transact SQL*) je jezik koji se koristi u Sinapsi. To je proširen jezik koji je razvio Majkrosoft za potrebe Majkrosoft SQL servera, baza podataka i softvera[5].

Serverles SQL pul je sistem distribuirane obrade podataka, napravljen za obradu velike količine podataka. Omogućava obradu velikih podataka u vremenskom rasponu od nekoliko sekundi do 30 minuta, u zavisnosti od količine podataka koje je potrebno obraditi, kao i od kompleksnosti upita[4].

Serverles predstavlja model korišćenja resursa za koje ne postoji infrastruktura, koja treba da se podešava, niti serveri koji treba da se održavaju i kojima treba da se upravlja. Ovo omogućava korisniku da jednostavno koristi dati servis po potrebi[4].

A. Mogućnosti korišćenja serverles SQL pula

Serverles SQL pul se najčešće koristi kada korisnik želi da brzo i lako istražuje podatke dostupne u jezeru podataka, kako bi dobio detaljnije informacije o njima. Pogodan je za sledeće scenarije[4]:

- **Spontani upiti** iz različitih formata (*Parquet*, CSV, JSON, Delta), kako bi korisnik mogao napraviti plan ekstrakcije i analize podataka.
- **Logičko skladište podataka** – serverles se može iskoristiti za kreiranje, bez relociranja i transformacije podataka, apstraktne relacije nad neobrađenim podacima skladištenim u više izvora.
- **Transformacija podataka** – serverles se može koristiti kao jednostavan i skalabilan alat za transformaciju podataka i kasnije skladištenje na ADLS-u.

Različite role kojima je namenjen serverles[4]:

- **Inženjeri podataka** (eng. *Data Engineers*) mogu da koriste serverles za istraživanje podataka, transformaciju i pripremu podataka.
- **Naučnici za podatke** (eng. *Data Scientists*) mogu da koriste serverles da lako i brzo razumeju sadržaj i strukturu podataka zahvaljujući funkcionalnostima kao što su OPENROWSET i automatsko zaključivanje šeme (eng. *Automatic schema inference*).
- **Analitičari podataka** (eng. *Data Analysts*) mogu da istražuju podatke i sinhronizovane Spark tabele (biće više reči o ovome u sekciji koja se tiče sinhronizacije metapodataka) pomoću T-SQL jezika ili da kreiraju Pauer Bi-Aj izveštaje nad željenim podacima pomoću serverlesa.

Serverles se može koristiti pomoću bilo koje aplikacije/klijenta koji ima mogućnost izvršavanja SQL upita i mogućnost da ostvari konekciju sa SQL serverom. Najčešće se koriste SSMS (*SQL Server Management Studio*) i ADS (*Azure Data Studio*) alati.

B. Podržane funkcionalnosti u serverles SQL pulu

Serverles SQL pul ima svoje prednosti i podržava različite funkcionalnosti koje su dostupne i u standardnim SQL serverima, ali i neke nedostatke u odnosu na klasične baze i skladišta. U ovoj sekciji će biti dat prikaz koje funkcionalnosti serverles podržava, a koje ne, uz kratke komentare[6].

Baze serverles SQL pula ne čuvaju konkretne podatke, već se podaci čuvaju na ADLS-u, dok se u bazi čuvaju metapodaci. U tabeli 1 navedeni su neki objekti dostupni u bazama podataka uz informaciju da li serverles to podržava i kratak komentar[6].

TABELA 1: DOSTUPNOST OBJEKATA U BAZAMA SERVERLES SQL PULA, PREMA [6]

Funkcionalnost	**Podrška**	**Komentar**
Tabele	Ne	Serverles služi samo za izvršavanje upita nad eksterno skladištenim podacima.
Pogledi (eng. *Views*)		
Šeme	Da	
Privremene tabele	Ne	Ne postoji podrška iz istog razloga kao i za obične tabele.
Procedure	Da	
Funkcije	Da	Isključivo *inline table-valued* funkcije
Eksterne tabele	Da	
Keširanje	Ne	
Statistike	Da	
Kontrola potrošnje	Da	Kontrola potrošnje je moguća na dnevnom, nedeljnom ili mesečnom nivou.

U tabeli 2 su navedene neke od bezbednosnih funkcionalnosti koje serverles SQL pul (ne) podržava, uz opcioni kratki komentar[6].

TABELA 2: BEZBEDNOSNE FUNKCIONALNOSTI U SERVERLES SQL PULU, PREMA [6]

Funkcionalnost	**Podrška**	**Komentar**
Login	Da	
Korisnici	Da	
SQL user/pass	Da	
Azurov AD (AAD)	Da	I korisnike i loginove
Autentifikacija pomoću AAD-a	Da	
Autentifikacija SAS tokenom	Da	Korišćenjem *DATABASE SCOPED CREDENTIAL* u eksternim izvorima podataka ili kredencijalima na nivou instance.
Storage access key	Ne	

Autentifikacija upravljanim identitetom (eng. *Managed Service Identity*, MSI)	Da	Korišćenjem *Managed Identity* kredencijala
Serverske role	Da	
Serverski ograničeni kredencijali	Da	
Permisije na nivou servera	Da	
Kredencijali ograničeni nad bazom	Da	
Permisije na nivou baze	Da	
Permisije na nivou šeme	Da	
Permisije na nivou objekta	Da	
Bezbednost na nivou kolona	Da	
Bezbednost na nivou redova	Ne	

U tabeli 3 navedeni su primeri skladišta podataka. Za svaki se navodi da li serverles ima mogućnost da povezivanja sa njim[6].

TABELA 3: DOSTUPNOST PODRŠKE SKLADIŠTA U SERVERLES SQL PULU, PREMA [6]

Skladište	**Podrška**
Interno skladište	Ne
ADLS v2 (gen 2)	Da
Azure Blob Storage	Da
Azurova SQL baza	Ne
CosmosDB transakciono skladište	Ne
CosmosDB analitičko skladište	Da, pomoću Sinapsine veze
Apač Spark tabele u radnom okruženju	Da, pomoću sinhronizacije metapodataka
Apač Spark tabele van radnog okruženja	Ne
Dejtabriks tabele	Ne

Serverles SQL pul podržava izvršavanje upita nad sledećim formatima podataka[6]:

- delimitirani (CSV, TSV),
- parke (eng. *Parquet*),
- džejson (eng. *JSON*) i
- delta.

Pošto serverles SQL pul radi sa eksternim podacima, za izvršavanje upita nad tim podacima koristi se OPENROWSET funkcija, koja omogućava pristup podacima na skladištu, čita izvorne podatke i za rezultat vraća tabelu redova sadržaja. Kada se u serverlesu izvršava OPENROWSET funkcija potrebno je definisati BULK operator da bi se specificirala lokacija podataka, kao i FORMAT da bi se definisao format koji se čita[7].

Primer kako izgleda korišćenje OPENROWSET funkcije da bi se dohvatili podaci sa ADLS-a pomoću serverlesa izgleda ovako:

```
SELECT * FROM OPENROWSET(
    BULK 'http://<ADLS nalog>.dfs.core.windows.net/container/folder/*.format_fajla,
    FORMAT = IME_FORMATA
) AS file
```

Takođe, postoji mogućnost definisanja parametra **izvor podataka** (eng. *Data Source*) u OPENROWSET funkciji. Ovaj parameter omogućava da se definiše putanja do nekog dela skladišta podataka odakle se izvorno traže podaci, a potom pomoću BULK operatora da sw definiše tačna putanja do samih foldera/fajlova.

Ova opcija, pored preglednosti, omogućava i bolju kontrolu prava pristupa podacima, pošto se prilikom kreiranja objekta za izvor podataka, osim lokacije, mogu definisati i kredencijali i način pristupa lokaciji.

Primer OPENROWSET upita uz ovaj parameter može se videti ispod.

```
SELECT *
FROM OPENROWSET(
      BULK '/folder/*.format_fajla,
      DATA_SOURCE='IME_IZVORA_PODATAKA', --> Putanja do skladišta definisana u
parametru LOCATION unutar kreiranog DATA SOURCE objekta
      FORMAT = 'IME_FORMATA'
) AS file
```

Automatsko zaključivanje šeme (eng. *Automatic Schema Inference*) predstavlja funkcionalnost OPENROWSET funkcije da, na osnovu pristupa podacima i specificiranog formata, razume koje kolone postoje u nekom fajlu bez dodatnog definisanja kolona u samom upitu.

Primer upita sa definisanjem kolona:

```
SELECT TOP 10 *
FROM OPENROWSET(
      BULK 'putanja/covid_slučajevi.format_fajla',
      FORMAT = 'ime_formata'
   ) with (
      date_rep date 1, -- Kolona koja predstavlja datum kada su uneti podaci
      cases int 5, -- Kolona koja predstavlja broj prijavljenih slučajeva za taj datum
      geo_id varchar(6) 8 -- Kolona koja predstavlja identifikator geografske lokacije gde su
slučajevi prijavljeni.
   ) as rows
```

Primer upita sa automatskim zaključivanjem šeme:

```
SELECT TOP 10 *
FROM OPENROWSET(
      BULK 'putanja/covid_slučajevi.format_fajla',
      FORMAT = 'ime_formata'
) as rows
```

Ova funkcionalnost dosta uprošćava spontano izvršavanje upita zarad istraživanja sadržaja iz fajlova.

C. Spontani upiti

Serverles SQL pul je servis za upite koji omogućava korisnicima da pokreću T-SQL upite nad podacima koji se nalaze na Azurovim servisima za skladištenje podataka. Najčešće se koristi kako bi se izvršavali spontani upiti, po potrebi korisnika, te kako bi se istražili podaci koje korisnik želi da obrađuje. Ovo je omogućeno zahvaljujući tome što je serverles SQL pul

dostupan odmah nakon kreiranja Sinapsinog radnog okruženja. Bez potrebe za dodatnim podešavanjima korisnik može odmah da izvršava. Ovaj slučaj upotrebe je zajednički gotovo za sve korisnike, bilo za one koji žele da vrše neku analitiku nad podacima i kreiraju vizuelne izveštaje u Pauer Bi-Aj-u (ili nekom drugom alatu) ili za one koji žele da istraže podatke kako bi znali kako da dizajniraju svoje skladište podataka koje žele da naprave nad njima. U nastavku će biti dati primeri izvršavanja spontanih upita nad Parke i Delta lejk formatima podataka.

Parke (eng. *Parquet*) je fajl format otvorenog koda. Efikasno je dizajniran sa ciljem da performanse budu na visokom nivou. Za razliku od ostalih fajl formata koji su bazirani na redovima, Parke je fajl format baziran na kolonama; to znači da su vrednosti kolona svake tabele poređane jedne pored druge, a ne da su poređane vrednosti celog jednog zapisa/reda[8].

Najlakši način za izvršavanje upita nad Parke podacima jeste da se iskoristi OPENROWSET funkcija i navede putanja do koje korisnik ima pristup i može da pročita podatke sa tražene putanje. Primer Parke upita sa automatskim zaključivanjem šeme izgleda ovako[8]:

```
select top 10 *
from openrowset (
   bulk 'https://pandemicdatalake.blob.core.windows.net/public/curated/covid-
19/ecdc_cases/latest/ecdc_cases.parquet,
   format = parquet
) as rows
```

Delta lejk (eng. *Delta Lake*) je sloj fajl formata otvorenog koda koji omogućava atomičnost, konzistentnost, izolaciju i trajnost (skraćeno ACID, eng. *Atomicity*, *Consistency*, *Isolation and Durability*) transakcija izvršenih nad podacima skladištenih u jezeru podataka. Ovo donosi veću pouzdanost, bezbednost i performanse[9].

Serverles SQL pul omogućava čitanje podataka skladištenih u Delta lejk formatu i serviranje podataka različitim alatima. Delta lejk fajlovi su uglavnom kreirani koristeći Apač Spark ili Azurov Dejtabriks[9].

Upiti nad Delta lejk podacima se mogu izvršavati pomoću OPENROWSET funkcije, a primer kako izgleda jedan upit može se videti ispod:

```
SELECT TOP 10 *
FROM OPENROWSET(
   BULK 'https://sqlondemandstorage.blob.core.windows.net/delta-lake/covid/',
   FORMAT = 'delta') as rows;
```

Spontani upiti se mogu izvršavati i nad ostalim podržanim formatima podataka navedenim u sekciji B.

D. Naredba CETAS i transformacija podataka

CETAS (eng. *CREATE EXTERNAL TABLE AS SELECT*) jeste DDL (skraćeno od eng. *Data Definition Language*) naredba koja omogućava izvršavanje sledećih poslova[10]:

- kreiranje eksterne tabele.
- paralelno eksportovanje podataka koji su rezultat SELECT upita nad definisanom lokacijom.

Najčešće se koristi kako bi se transformisali neki podaci, na primer prebacili iz *CSV* formata u *Parquet* format. Korisnik koji izvršava CETAS mora da ima prava pristupa podacima koje čita kao i prava pristupa skladištu podataka gde želi da eksportuje podatke[10].

Primer sintakse[10]:

```
CREATE EXTERNAL TABLE [ [naziv_baze. [ naziv_šeme ] . ] | naziv_šeme. ] naziv_tabele
   WITH (
      LOCATION = 'putanja_do_foldera',
      DATA_SOURCE = naziv_izvora_podataka,
      FILE_FORMAT = naziv_fajl_formata
)
   AS <select_upit>  [;]
```

Bitno je napomenuti da opcije koje se definišu prilikom kreiranja eksterne tabele odnose isključivo na eksternu tabelu, odnosno lokacija će ukazivati na tačnu putanju gde će podaci biti eksportovani, fajl format u kom formatu će podaci biti skladišteni, a izvor podataka definiše na kom prostoru će biti primenjena putanja i gde će podaci biti skladišteni.

Transformacija podataka je drugi često korišćeni način korišćenja serverles SQL pula. Korisnik prvobitno pomoću serverles SQL pula može da istraži podatke, tzv spontanim (ad hok) upitima o kojima je bilo prethodno reči i kada razume koji se podaci nalaze u setu podataka onda može da odluči, shodno svojim potrebama, kako želi da ih transformiše i eksportuje. Korisnik pomoću SELECT upita u okviru CETAS-a može da definiše koje kolone želi da sačuva u novokreiranoj eksternoj tabeli, da ih transformiše kako mu odgovara i da primeni neki filter.

U praktičnom primeru ispod, dato je kreiranje eksterne tabele nad podacima koji su javno dostupni na Azurovom blob skladištu i mogu se naći u galeriji u okviru prethodno pomenutog centra znanja Sinapsinog studija. Konkretno je odabran Bingov set podataka o slučajevima bolesti *Covid19* i transformisali su se podaci u Parquet fajl format sa tipom konverzije *Snappy* što omogućava brže i efikasnije upite[11].

Praktični primer kreiranja eksterne tabele pomoću CETAS-a nad Bingovim *Covid19* podacima:

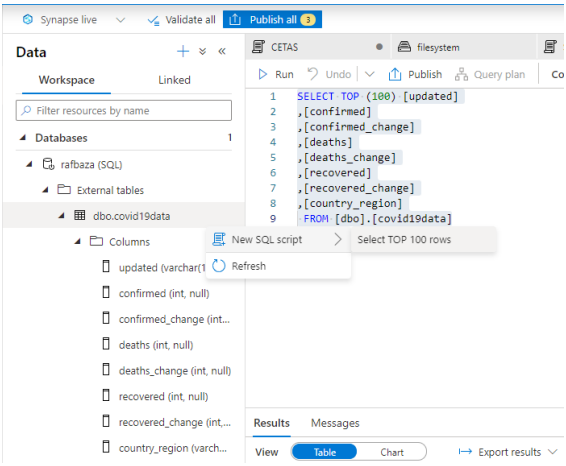
```
-- Upit za kreiranje eksternog fajl formata
CREATE EXTERNAL FILE FORMAT [SnappyParquet]
```

```
    WITH (
        FORMAT_TYPE = PARQUET,  -- Definisan je Parquet kao tip formata
        DATA_COMPRESSION = 'org.apache.hadoop.io.compress.SnappyCodec'
-- Definiše se kompresija podataka, želimo da podatke čuvamo u Snappy kompresiji. )
GO

-- Upit za kreiranje eksternog izvora podataka
CREATE EXTERNAL DATA SOURCE [MyFileSystem]
    WITH (
        LOCATION   = 'https://rafskladiste.dfs.core.windows.net/filesystem/'
-- Definiše se putanja do izvora podataka, ADLS Gen2 nalog sa nazivom ‘rafskladiste’. )
GO

-- Upit za kreiranje eksterne tabele gde se definiše naziv tabele, putanja do lokacije gde će da se skladište podaci na eksternom prostoru kao I tip formata u kom će podaci da se skladište.
CREATE EXTERNAL TABLE covid19data
    WITH (
    LOCATION = 'Bing-Covid19-Data/ExternalTables',
    DATA_SOURCE = [MyFileSystem],
    FILE_FORMAT = [SnappyParquet]
    )
AS
SELECT -- Biraju se kolone koje korisnik želi da ima u eksternoj tabeli koju kreira
    CONVERT(varchar(10), updated, 103) as updated,  -- Konvertuje se datum ažuriranja reda u DD/MM/YYYY format varchar tipa sa definisanom veličinom od 10 karaktera.
    SUM(confirmed) as confirmed,  -- Sumiraju se svi potvrđeni slučajevi.
    SUM(confirmed_change) as confirmed_change, -- Sumiraju promene nad potvrđenim slučajevima.
    SUM(deaths) as deaths, -- Sumiraju se brojevi smrtnih slučajeva.
    SUM(deaths_change) as deaths_change, -- Sumiraju se promene broja smrtnih slučajeva.
    SUM(recovered) as recovered, -- Sumiraju se brojevi oporavljenih.
    SUM(recovered_change) as recovered_change, -- Sumiraju se promene broja oporavljenih.
    country_region as country_region -- Bira se kolona koja predstavlja zemlju na koju se odnosi red.
FROM
    OPENROWSET(
        BULK  'https://pandemicdatalake.blob.core.windows.net/public/curated/covid-19/bing_covid-19_data/latest/bing_covid-19_data.parquet',
        FORMAT = 'parquet'
    ) AS [result]
GROUP BY updated, country_region -- Grupišemo podatke po neagregiranim kolonama za datum ažuriranja reda i državu na koju se red odnosi.
GO
```

Nakon izvršavanja ove skripte u okviru serverles SQL pul baze *rafbaza*, u podrazumevanoj šemi *dbo* vidljiva je kreirana eksterna tabela *covid19data* sa transformisanim kolonama. Takođe, pomoću grafičkog korisničkog interfejsa Sinapsinog studija, može se generisati upit za odabir TOP 100 redova iz ove eksterne tabele. To se može videti na slici 3, ispod[12].



Slika 3 - Kreirana eksterna tabela i generisan upit, na osnovu[12]

Bitno je napomenuti da se, kada se izvršava upit nad ovom eksternom tabelom, podaci čitaju sa lokacije koja je definisana u CETAS-u. U primeru je definisana lokacija na ADLS-u pod nazivom *rafskladiste* unutar kontejnera *filesystem* sa putanjom do *Bing-Covid19-Data/ExternalTables.* Prilikom navigacije na ovu lokaciju, mogu se videti generisani kompresovani Parquet fajlovi koji sadrže podatke koji se dobijaju prilikom izvršavanja upita nad eksternom tabelom *covid19data.*

E. Logičko skladište podataka

Logičko skladište podataka (eng. *Logical data warehouse, LDW*) jeste relacioni sloj koji se može izgraditi nad različitim izvorima podataka kao što su Azurovo jezero podataka (ADLS), Azurovo analitičko skladište CosmosDB ili Azurovo blob skladište[13].

Ideja logičkog skladišta podataka jeste kreirati eksterne tabele nad više različitih izvora podataka i tako dobiti skladište podataka nad podacima koji nisu fizički prisutni na jednom skladištu, već su dostupni sa više različitih lokacija pomoću objedinjene baze i kreiranih eksternih tabela ili pogleda nad tim podacima. U ovoj sekciji biće prikazan primer logičko skladišta podataka kreiranog nad javno dostupnom Azurovom setu podataka ECDC COVID-19[14].

Prilikom kreiranja baze gde će se skladištiti eksterne tabele i pogledi koji će referencirati eksterne izvore podataka bitno je podesiti kolaciju koja će pružiti optimalne performanse nad Parke i CosmosDB podacima. Primer skripte koja kreira bazu[13]:

```
CREATE DATABASE Ldw COLLATE Latin1_General_100_BIN2_UTF8;
```

Jedna od mana serverles SQL pula je to što ne pruža performanse od starta prilikom kreiranja baze, već moraju da se vrše dodatna podešavanja, poput postavljanja specifične kolacije i slično[15].

Nakon kreiranja baze, potrebno je definisati izvore podataka nad kojima će se definisati eksterne tabele i pogledi[13]:

```
CREATE EXTERNAL DATA SOURCE ecdc_cases WITH (
   LOCATION = 'https://pandemicdatalake.blob.core.windows.net/public/curated/covid-19/ecdc_cases/');
```

Pošto su ovi podaci otvorenog tipa, bilo koji Azurov AD identitet može da pristupi istim. Međutim, to često nije slučaj, već se moraju eksplicitno podesiti prava pristupa Azurovim AD identitetima kako bi mogli da pristupe podacima nad eksternim izvorima podataka.

Može se koristiti više različitih tipova kredencijala za pristup eksternim izvorima podataka, a neki su već i spomenuti u sekciji o pravima pristupa nad podacima[13]:

- *Shared access signature (SAS)* – deljeni token pristupa
- *Managed identity (MSI)* – upravljani identitet
- CosmosDB ključ za pristup CosmosDB analitičkom skladištu

Prvobitno se mora kreirati master ključ nad bazom. To je potrebno uraditi samo jednom nad bazom:

```
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'Nek! K0mpl3k$4n KLJYCH';
```

U primerima koji slede koristiće se sledeći CosmosDB i MSI kredencijali[13]:

```
CREATE DATABASE SCOPED CREDENTIAL WorkspaceIdentity
WITH IDENTITY = 'Managed Identity';
GO
-- Kreira se eksterni izvor podataka kojim se pristupa pomoću kreiranog MSI-a
CREATE EXTERNAL DATA SOURCE ecdc_cases WITH (
   LOCATION = 'https://pandemicdatalake.blob.core.windows.net/public/curated/covid-19/ecdc_cases/',
   CREDENTIAL = WorkspaceIdentity
);
-- Definiše se kredencijal koji predstavlja ključ kojim se pristupa Cosmos DB nalogu
CREATE DATABASE SCOPED CREDENTIAL MyCosmosDbAccountCredential
WITH IDENTITY = 'SHARED ACCESS SIGNATURE',
    SECRET = 's5zarR2pT0JWH9k8roipnWxUYBegOuFGjJpSjGlR36y86c W0GQ6RaaG8kGjsRAQoW Mw1QKTkkX8HQtFpJjC8Hg==' -- Obrisati razmak u tajnom ključu ukoliko se kopira primer
```

Ove kredencijale mogu da koriste samo Sinapsini Administratori ili Sinapsini SQL administratori. Ukoliko postoje neki drugi korisnici potrebno im je eksplicitno dati pristup ovim kredencijalima[13]:

```
GRANT REFERENCES ON DATABASE SCOPED CREDENTIAL::WorkspaceIdentity TO <korisnik>
GO
GRANT REFERENCES ON DATABASE SCOPED
```

```
CREDENTIAL::MyCosmosDbAccountCredential TO <korisnik>
GO
```

Nakon kreiranih kredencijala, potrebno je definisati eksterni fajl format, šemu, eksternu tabelu nad podacima u Azurovom skladištu i pogled nad podacima u CosmosDB-u[13]:

```
CREATE EXTERNAL FILE FORMAT ParquetFormat WITH (  FORMAT_TYPE =
PARQUET );
CREATE EXTERNAL FILE FORMAT CsvFormat WITH (  FORMAT_TYPE =
DELIMITEDTEXT );
-- Kreira se šema i eksterna tabela nad podacima u ADLS-u
create schema ecdc_adls;
GO
create external table ecdc_adls.cases (
   date_rep                 date,
   day                      smallint,
   month                    smallint,
   year                     smallint,
   cases                    smallint,
   deaths                   smallint,
   countries_and_territories  varchar(256),
   geo_id                   varchar(60),
   country_territory_code     varchar(16),
   pop_data_2018              int,
   continent_exp              varchar(32),
   load_date                  datetime2(7),
   iso_country                varchar(16)
) with (
   data_source= ecdc_cases,
   location = 'latest/ecdc_cases.parquet',
   file_format = ParquetFormat
);
GO
-- Kreira se šema i pogled nad podacima u CosmosDB-u
create schema ecdc_cosmosdb;
GO
CREATE OR ALTER VIEW ecdc_cosmosdb.Ecdc
AS SELECT *
FROM OPENROWSET(
     PROVIDER = 'CosmosDB',
     CONNECTION = 'Account=synapselink-cosmosdb-sqlsample;Database=covid',
     OBJECT = 'Ecdc',
     CREDENTIAL = 'MyCosmosDbAccountCredential'
   ) WITH
    ( date_rep varchar(20),
      cases bigint,
      geo_id varchar(6)
    ) as rows
GO
```

Može se primetiti da su iskorišćene najmanje moguće veličine za kolone kako bi se optimizovale performanse.

Poslednji korak jeste kreiranje i dodeljivanje prava pristupa nekom SQL ili AAD korisniku nad bazom i šemama ili objektima[13]:

```
-- Kreiranje logina i SQL korisnika
CREATE LOGIN [strahinja] WITH PASSWORD = 'Veoma Jak P4$$W0RD';
CREATE USER [strahinja] FROM LOGIN [strahinja];
GO
-- Onemogućavanje administratorskih privilegija nad bazom ovom korisniku I dodavanje
permisija za pristup kompletnoj ecdc_adls šemi i cases pogledu u ecdc_cosmosDB šemi.
DENY ADMINISTER DATABASE BULK OPERATIONS TO [strahinja]
GRANT SELECT ON SCHEMA::ecdc_adls TO [strahinja]
GRANT SELECT ON OBJECT::ecdc_cosmosdb.ecdc TO [strahinja]
GO
-- Davanje permisija za korišćenje kredencijala
GRANT REFERENCES ON DATABASE SCOPED CREDENTIAL::WorkspaceIdentity TO
[strahinja]
GRANT REFERENCES ON DATABASE SCOPED
CREDENTIAL::MyCosmosDbAccountCredential TO [strahinja]
GO
```

Nakon svega izvršenog u ovoj sekciji, postoji kreirano logičko skladište podataka u bazi LDW. Ova baza sadrži metapodatke pomoću kojih se pristupa eksternim podacima u ADLS-u i CosmosDB-u, i pomoću kreirane eksterne tabele, odnosno pogleda nad ovim podacima, kreirano je logičko skladište podataka. Prilikom izvršavanja upita se mogu kombinovati podaci iz oba izvora podataka.

Primer upita gde se kombinuju podaci iz pogleda nad podacima u CosmosDB-u i eksterne tabele kreirane nad podacima u ADLS-u:

```
SELECT cosmosdb.date_rep, cosmosdb.cases FROM ecdc_cosmosdb.Ecdc AS cosmosdb
  JOIN ecdc_adls.cases AS adls ON cosmosdb.geo_id = adls.geo_id
WHERE cosmosdb.geo_id = 'RS';
```

Ovo predstavlja jako fleksibilno rešenje i ima ogroman potencijal ukoliko bi se omogućilo kreiranje logičkih skladišta podataka i nad podacima iz drugih izvora podataka kao što su AWS S3, SQL DB, neki drugi SQL server i slično. Do tada, ostaje ograničenje samo nad podacima skladištenim u ADLS-u i CosmosDB-u.

IV. Zaključak

Kao što se može zaključiti iz razvijenih primera, Sinapsa je, generalno, dosta dobro trenutno dostupno rešenje za analitiku velikih podataka u klaudu. Sinapsin studio je razvojno okruženje koje je slično nekim drugim servisima (kao što je Azurova fabrika podataka) i to daje srodno korisničko iskustvo, tako da se već postojeći korisnici Azura mogu lako privići na novi proizvod. Serverles SQL pul je moderan servis koji omogućava brzo izvršavanje ad hok upita, te daje brz pregled podataka dostupnih na ADLS-u. Mogućnost

kreiranja logičkih skladišta podataka koje povezuje podatke sa *CosmosDB*-a, Dejtaversa i ADLS-a omogućava zanimljive scenarije.

Nedostatak serverles SQL pula je upitna mogućnost povezivanja sa drugim rešenjima za računarstvo u oblaku kao što su Amazonov *AWS* ili Guglov klaud. Ova mogućnost potrebna je za direktno izvršavanje upita i nad skladištima u eksternim ekosistemima. Takođe, nedostatak je i to što je potrebno dosta podešavanja serverles SQL pula da bi se dobio optimalan način za izvršavanje upita, tako da je potrebno posvetiti dosta vremena i praktično postati ekspert. Bilo bi bolje da je optimalno izvršavanje upita nad podacima omogućeno od samog starta bez dodatnih podešavanja.

U okviru rada nije bilo prostora za adekvatno istraživanje i detaljnije prikazivanje korišćenja servisa kao što su Apače Spark pul i Sinapsini Pajplajnovi, koji omogućavaju drugačije obrade u odnosu na serverles SQL pul. Izbor Sinapse u konkretnom slučaju primene podrazumevao bi i prethodno komparativno izučavanje konkurentskih rešenja kao što su Guglov *BigQuery*, Amazon *Athena* i *Snowflake*. Adekvatno poređenje verovatno bi dovelo do zaključka da je svako od ovih rešenja najbolje za neki od scenarija.

V. LITERATURA

[1] https://azure.microsoft.com/en-us/services/synapse-analytics/#overview, 07.01.2021.
[2] https://docs.microsoft.com/en-us/learn/paths/realize-integrated-analytical-solutions-with-azure-synapse-analytics/ , 16.01.2021.
[3] https://portal.azure.com/, 12.12.2021.
[4] https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/on-demand-workspace-overview, 04.10.2021.
[5] https://www.dataquest.io/blog/sql-vs-t-sql/, 05.10.2021.
[6] https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/overview-features, 05.10.2021.
[7] https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/develop-openrowset, 06.10.2021.
[8] https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/query-parquet-files, 20.12.2021.
[9] https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/query-delta-lake-format, 16.10.2021.
[10] https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/develop-tables-cetas, 06.10.2021.
[11] https://en.wikipedia.org/wiki/Snappy_(compression), 10.10.2021.
[12] https://ms.portal.azure.com/, 18.01.2021.
[13] https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/tutorial-logical-data-warehouse, 16.10.2021.
[14] https://docs.microsoft.com/en-us/azure/open-datasets/dataset-ecdc-covid-cases?tabs=azure-storage, 16.10.2021.
[15] https://docs.microsoft.com/en-us/azure/synapse-analytics/sql/best-practices-serverless-sql-pool, 20.12.2021.

ABSTRACT

The development of information systems and technologies has enabled experts to intensively use remote computing resources according to the concept of cloud computing. Among other things, there is a development of cloud databases, as well as a transfer of data warehouses to the cloud. Microsoft's Azure is one of the leading cloud computing systems; it has support for many databases, and one of the platforms available on it is Azure Synapse Analytics.

This paper provides a brief critical overview of Azure Synapse Analytics, with an emphasis on its distributed data processing system, serverless SQL pool. This system allows data engineers, data architects, data analysts, business analysts and scientists to create virtual data warehouses, explore the big data available in the Azure Data Lakes, perform complex queries and create business reports.

Through this paper, we want to acquaint the reader with the basics of using the presented technologies. With this goal in mind, many examples have been developed, and the advantages and disadvantages of the presented technologies have been discussed.

THE USE OF AZURE SYNAPSE ANALYTICS' SERVERLESS SQL POOL TECHNOLOGY IN DATA ANALITICS

Strahinja Rodić, Dušan Vujošević