Sustav za obradu polustrukturiranih tekstnih datoteka poslovnih transakcijskih podataka

Derdić, Ivan

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 458

SUSTAV ZA OBRADU POLUSTRUKTURIRANIH TEKSTNIH DATOTEKA POSLOVNIH TRANSAKCIJSKIH PODATAKA

Ivan Derdić

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 458

SUSTAV ZA OBRADU POLUSTRUKTURIRANIH TEKSTNIH DATOTEKA POSLOVNIH TRANSAKCIJSKIH PODATAKA

Ivan Derdić

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 4. ožujka 2024.

DIPLOMSKI ZADATAK br. 458

Pristupnik:	**Ivan Derdić (0036523165)**
Studij:	Računarstvo
Profil:	Programsko inženjerstvo i informacijski sustavi
Mentor:	izv. prof. dr. sc. Alan Jović

Zadatak: **Sustav za obradu polustrukturiranih tekstnih datoteka poslovnih transakcijskih podataka**

Opis zadatka:

Za poslovne transakcijske podatke pohranjene u tekstnim datotekama u polustrukturiranom obliku ne postoji norma ili specifikacija koja bi odredila najbolji način njihove obrade. Transakcijski podaci u izvornom, neobrađenom obliku pohranjeni su u podatkovnom jezeru (engl. data lake), a potom prolaze proces obrade i spremaju se kroz par koraka u oblicima poznatima pod zajedničkim nazivom jezerskog skladišta (engl. data lakehouse) koje se koristi za daljnju podatkovnu analizu. U ovom radu razmatrat će se učinkovita rješenja za obradu podataka u jezerskom skladištu pri čemu će fokus biti na polustrukturiranim tekstnim (CSV) datotekama (ali ne u obliku JSON i XML) te njihovim transformacijama od neobrađenog oblika do konačnog oblika pogodnog za poslovnu namjenu. U tu svrhu, potrebno je razraditi sustav u obliku podatkovnog cjevovoda koji počinje s neobrađenim oblikom tekstnih datoteka, opcionalnom optimizacijom s međutablicom koja će biti što sličnija neobrađenom obliku neovisno o sadržaju a olakšat će daljnju transformaciju podataka te s pročišćenim podacima u konačnom obliku pohranjenima u jednoj ili više tablica. Potrebno je ostvariti simulaciju transakcijskih poslovnih podataka razvojem generatora izvornih neobrađenih polustrukturiranih tekstnih datoteka. U cjelovitom razvijenom rješenju, svaki korak cjevovoda treba se moći vrednovati u smislu vremenske učinkovitosti obrade. Za transformaciju podataka treba isprobati te kvalitativno i kvantitativno usporediti nekoliko rješenja otvorenog koda (npr. polars, DataFusion).

Rok za predaju rada: 28. lipnja 2024.

Sadržaj

1. Uvod

U današnjem digitalnom dobu, obrada i analiza velikih količina podataka postali su ključni za poslovni uspjeh. Tvrtke prikupljaju i pohranjuju ogromne količine podataka, od kojih su mnogi polustrukturirani i pohranjeni u tekstualnim datotekama. Takvi podaci često dolaze iz poslovnih transakcija i predstavljaju bogat izvor informacija koji se može iskoristiti za donošenje poslovnih odluka, optimizaciju operacija i unapređenje usluga.

Međutim, obrada polustrukturiranih podataka predstavlja izazov zbog nedostatka standarda i specifikacija koje bi odredile najbolji način njihove obrade. U ovom radu fokus je se na razvoju sustava za obradu polustrukturiranih tekstualnih datoteka poslovnih transakcijskih podataka. Cilj je istražiti učinkovita rješenja koja omogućuju transformaciju tih podataka od njihovog izvornog, neobrađenog oblika do oblika pogodnog za poslovnu analizu.

Rad se sastoji od analize i implementacije tri modela obrade podataka: ETL (izvuci [extract], obradi [transform], spremi [load]), ELT (izvuci, spremi, obradi) i ETLL (izvuci, obradi i spremi, spremi). Implementacija koristi tehnologije što su Delta Lake i Apache DataFusion, koje omogućuju pouzdanu i brzu obradu podataka te osiguravaju svojstva ACID.

U radu se daje usporedba performanci ovih modela u kontekstu brzine obrade. Na temelju eksperimentalnih rezultata, identificirat će se prednosti i nedostaci svakog modela, što će pomoći u odabiru optimalnog rješenja za obradu polustrukturiranih tekstualnih podataka u poslovnom okruženju.

2. Tehnologije

U ovom poglavlju opisane su tehnologije koje su korištene u ovom radu. Tehnologije su podijeljene u dvije skupine: tehnologije za spremanje podataka i tehnologije za obradu podataka. Za spremanje podataka korištena je tehnologija *Delta Lake*. *Delta Lake* kao tehnologija omogućuje praćenje informacija o pripadnosti objekata tablicama Delta sa svojstvima ACID u jezerima podataka. To je omogućeno koristeći zapisnik transakcija. Objekti tablica Delta su zapisani u formatu *Parquet*. Jezera podataka (engl. *data lake*) su mjesto za pohranu strukturiranih, polustrukturiranih i nestrukturiranih podataka. Omogućuju pohranu složenih tipova podataka poput videa i slika, ali imaju problema sa strukturiranjem podataka. *Delta Lake* omogućuje verzioniranje podataka, povratak na prethodne verzije podataka, brisanje i ažuriranje podataka, efikasno strujanje podataka, predmemoriranje podataka, optimizaciju raspodjele podataka i evoluciju sheme podataka u jezerima podataka. Za više informacija o *Delta Lakeu* vidjeti [1]. *Delta Lake* je odabrana tehnologija jer omogućuje operacije MERGE nad tablicama bez potrebe za cijelim sustavom baze podataka. Time se smanjuje složenost sustava i uklanja se slanje podatka između procesa za obradu podatak i baze podataka. U ovom radu koristi se implementacija *delta-rs Delta Lakea* u programskom jeziku Rust. Za više informacija o *delta-rs* vidjeti [2].

Za obradu podataka korištena je tehnologija *Apache DataFusion*. To je biblioteka za obradu podataka koja omogućuje izvršavanje SQL upita i korištenje podatkovnih okvira nad različitim izvorima podataka. *Apache DataFusion* je implementiran u programskom jeziku Rust i koristi *Apache Arrow* kao format podataka u memoriji. Za više informacija o *Apache DataFusion* vidjeti [3]. *Apache DataFusion* je odabrana tehnologija jer omogućuje izvršavanje operacija podatkovnih okvira nad CSV datotekama i alternativa je korištenju *Apache Sparka* koji je teži za konfiguraciju i zahtjeva više resursa. Također

Apache DataFusion se prevodi u izvršni kod koji se izvršava na ciljnoj platformi, a ne izvršava se u JVM-u kao *Apache Spark*. Time se smanjuje složenost sustava i ispitivanja.

Za obradu datoteka i korištenje *Delta Lake* i *Apache DataFusion* korišten je programski jezik Rust. To je programski jezik koji je dizajniran za sigurnost i performance. Rust je dizajniran tako da se izbjegnu pogreške u programiranju kao što su prekoračenje memorije, nevalidni pokazivači, utrka za resursima i druge pogreške. Rust je odabran kao programski jezik jer omogućuje sigurno programiranje i visoke performance. Za više informacija o programskom jeziku Rust vidjeti [4]. Rust je odabran jer je *Apache DataFusion* implementiran u Rustu i omogućuje jednostavno integriranje s *Delta Lakeom*. Također Rust je odabran radi potrebe za brzom predobradom datoteka.

3. Modeli podataka

U ovom poglavlju opisani su modeli podataka koji se koriste u ovom radu. Modeli podataka su izvorišna baza podataka, izvozne datoteke i jezersko skladište podataka. Model izvorišne baze podataka je opisan u poglavlju 3.1., model izvozne datoteke je opisan u poglavlju 3.2. i model jezerskog skladišta podataka je opisan u poglavlju 3.3.

3.1. Model izvorišne transakcijske baze podataka

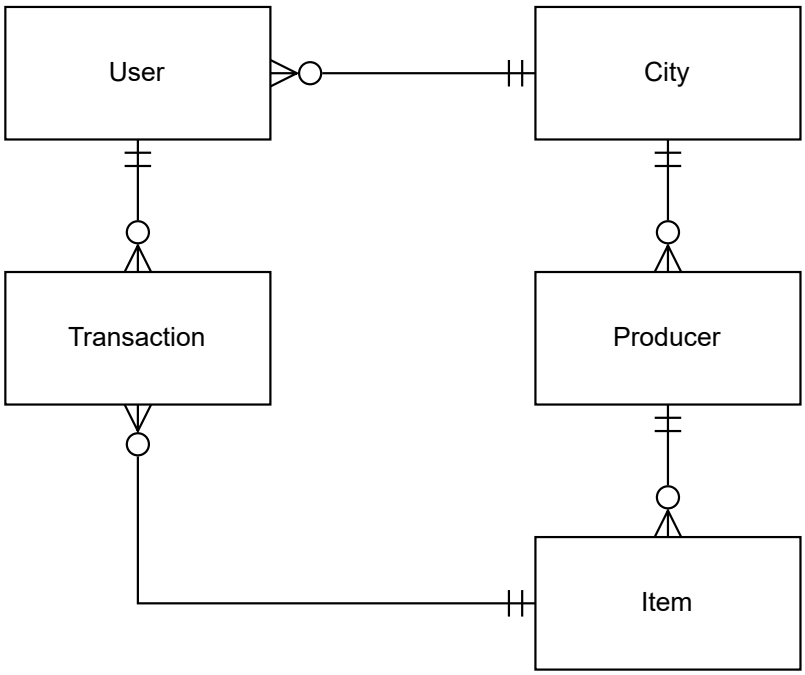
U ovom poglavlju opisan je model izvorišne baze podataka. Baza podataka je fiktivna i služi kao primjer za potrebe ovog rada. Izvorišna baza podataka je jednostavna jer nije fokus ovog rada. Sastoji se od sljedećih entiteta:

- *Transaction* — entitet koji predstavlja transakciju,
- *User* — entitet koji predstavlja korisnika,
- *Item* — entitet koji predstavlja artikl,
- *Producer* — entitet koji predstavlja proizvođača i
- *City* — entitet koji predstavlja grad.

Navedeni entiteti su prikazani ER dijagramom na slici 3.1.

3.2. Izvoz baze podataka

Izvorišna baza podataka izvozi se u polustrukturiranu tekstnu datoteku. Za potrebe ovog rada, izvoz baze podataka je simuliran. Primjer izvoza baze podataka u polustrukturiranu tekstnu datoteku prikazan je u ispisu 3..1 Izvozna datoteka sadrži sve entitete iz izvorišne baze podataka. Svaki redak u datoteci pripada jednom entitetu. Prvi stupac



Slika 3.1. ER dijagram izvorišne baze podataka.

u retku označava tip entiteta, a ostali stupci predstavljaju atribute entiteta. Za obradu izvezene datoteke potrebno je poznavati shemu izvorišne baze podataka jer datoteka ne sadrži informacije o tipovima podataka atributa. Izvoz baze podataka se sprema u direktorij formata *batch_<količina podatala (MiB)>_<broj datoteka>* koji sadrži datoteke formata *file_<veličina datoteke (B)>_<broj datoteka>_<oznaka lokacije>.txt*. Oznaka lokacije je broj koji označava redoslijed izvoza baze podataka.

Ispis 3..1: Primjer izvoza baze podataka u polustrukturiranu tekstnu datoteku.

```
ITEM|569|Cloth|907.1|81
ITEM|570|Metal|483.29|41
ITEM|571|Appliance|588.98|48
...
USER|878|Evan|Thomas|70
USER|879|Ella|Perez|31
USER|880|Lincoln|Lee|78
...
CITY|361|Dallas
CITY|362|Anchorage
CITY|363|Bakersfield
```

```
...
PRODUCER|407|Sigma Solutions|191707801.255857|90
PRODUCER|408|Alpha Innovations|783048039.610927|66
PRODUCER|409|Upsilon Ventures|257760750.720300|14
...
TRANSACTION|642|214|2015-07-17T05:17:13Z|71|7
TRANSACTION|643|816|2005-08-26T20:13:01Z|44|32
TRANSACTION|644|607|2021-11-17T15:41:52Z|52|31
```

3.3. Model jezerskog skladišta podataka

U ovom poglavlju opisan je model jezerskog skladišta podataka. Jezersko skladište podataka je arhitektura skladištenja podataka koja kombinira karakteristike jezera podataka i skladišta podataka (engl. *data warehouse*). To je centralizirano skladište podataka koje omogućava analizu velike količine strukturiranih i nestrukturiranih vrsta podataka u realnom vremenu ili kasnijem trenutku. Jezersko skladište podataka omogućava integraciju podataka iz različitih izvora, olakšava upravljanje podacima, smanjuje troškove i vrijeme potrebno za pripremu podataka za analizu.

U jezerskom skladištu podataka brončani, srebrni i zlatni sloj predstavljaju različite faze obrade i pripreme podataka. Brončani sloj sadrži sirove, neobrađene podatke prikupljene iz različitih izvora. Ovi podaci su najbliži izvornom obliku u kojem su prikupljeni uz minimalne transformacije koje uključuju dodavanje osnovnih metapodataka poput datuma unosa. Glavni cilj brončanog sloja je pohraniti podatke u izvornom obliku kako bi se omogućila kasnija obrada.

Srebrni sloj sadrži podatke koji su prošli kroz određene transformacije i pripreme za daljnju analizu. Ovi podaci su očišćeni, normalizirani i strukturirani te su često organizirani u dimenzijske i činjenične tablice pripremljene za analitičke upite i modele. Cilj srebrnog sloja je osigurati podatke u obliku koji je spreman za detaljniju analizu, uključujući uklanjanje pogrešaka, normalizaciju i spajanje podataka iz različitih izvora.

Zlatni sloj sadrži podatke koji su u potpunosti obrađeni i analizirani te su spremni za korištenje u izvještavanju i donošenju odluka. Podaci u zlatnom sloju su agregirani, sumarizirani i optimizirani za brzi pristup i analizu, što ih čini spremnima za krajnje

korisnike i poslovne aplikacije. Cilj zlatnog sloja je pružiti konačne, pouzdane i brze podatke za analitičke procese, izvještavanje i donošenje odluka.

Kombiniranjem ovih slojeva, jezersko skladište podataka omogućava učinkovito upravljanje cijelim ciklusom obrade podataka od unosa do konačne analize. Ovaj pristup osigurava fleksibilnost i skalabilnost u radu s velikim količinama podataka iz različitih izvora, omogućujući pouzdano i brzo donošenje poslovnih odluka temeljenih na analiziranim podacima. Za više o jezerskom skladištu podataka vidjeti [5].

U ovom radu jezersko skladište podataka se sastoji samo od brončanog i srebrnog sloja. Zlatni sloj jezerskog skladišta podataka nije implementiran jer nije fokus ovog rada.

Brončani sloj jezerskog skladišta podataka se sastoji od entiteta *raw_export* koji predstavlja izvezene podatke iz izvorišne baze podataka. Entitet *raw_export* se sastoji od sljedećih atributa:

- *location_number* — identifikator lokacije,
- *id* — identifikator entiteta,
- *col1* — prvi stupac izvoza baze podataka,
- *col2* — drugi stupac izvoza baze podataka,
- *col3* — treći stupac izvoza baze podataka,
- *col4* — četvrti stupac izvoza baze podataka,
- *col5* — peti stupac izvoza baze podataka i
- *batch_id* — identifikator grupe podataka.

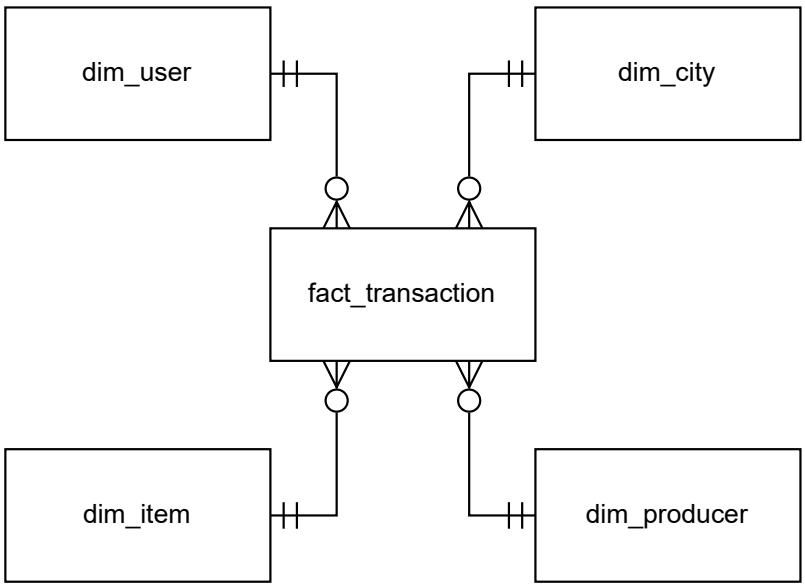
Namjena entiteta *raw_export* je pohrana podatka iz izvoza baze podataka bez velikih transformacija. Spremanjem izvoza u brončani sloj podaci se pretvaraju u strukturirani binarni format. Pretvorbom podataka u binarni format omogućuje se brži pristup podacima i smanjuje se potrošnja memorije. Brončani sloj jezerskog skladišta podataka je prikazan ER dijagramom na slici 3.2.

Slika 3.2. ER dijagram brončanog sloja jezerskog skladišta podataka.

Srebrni sloj je modeliran kao dimenzijski model. Dimenzijski model je model podataka koji se koristi za olakšavanje upita i analize podataka. Dimenzijski model se sastoji činjeničkih i dimenzijskih tablica. Činjenička tablica sadrži mjere i strane ključeve dimenzijskih tablica. Dimenzijske tablice sadrže atribute koji opisuju mjere u činjeničnoj tablici. Za više o dimenzijskim modelima vidjeti [6]. U ovom radu srebrni sloj jezerskog skladišta podataka se sastoji od sljedećih entiteta:

- *fact_transaction* — činjenička tablica koja sadrži mjere transakcija,
- *dim_user* — dimenzijska tablica koja sadrži atribute korisnika,
- *dim_item* — dimenzijska tablica koja sadrži atribute artikla,
- *dim_producer* — dimenzijska tablica koja sadrži atribute proizvođača i
- *dim_city* — dimenzijska tablica koja sadrži atribute grada.

Datumska dimenzija nije implementirana jer nije potrebna za ovaj rad. Srebrni sloj jezerskog skladišta podataka je prikazan ER dijagramom na slici 3.3.



Slika 3.3. ER dijagram srebrnog sloja jezerskog skladišta podataka.

4. Modeli obrade podataka

U ovom poglavlju opisani su modeli obrade podataka koji su korišteni u ovom radu. U ovom radu korišteni su modeli: ETL (izvuci [extract], obradi [transform], spremi [load]), ELT (izvuci, spremi, obradi) i ETLL (izvuci, obradi i spremi, spremi). Modeli ETL i ELT su standardni modeli obrade podataka koji se koriste u praksi. Model ETLL je model koji je predložen u ovom radu i kombinira prednosti modela ETL i ELT.

4.1. Model ETL

Model ETL sastoji od tri koraka: izvuci, obradi i spremi. U modeli ETL podaci se izvlače iz izvora podataka, obrađuju i spremaju u ciljano odredište podataka. Korak obrade podataka uključuje čišćenje podataka, provjeru podataka, transformaciju podataka i obogaćivanje podataka. Model ETL je prikazan na slici 4.1. Model ETL se koristi za obradu podataka koji se nalaze u različitim izvorima podataka i potrebno ih je obraditi i spremiti u ciljano odredište podataka. Za više informacija o modelu ETL vidjeti [7].

Jedna od prednosti modela ETL je brzina obrade podataka jer se podaci obrađuju u memoriji i zatim se spremaju u ciljano odredište podataka. Ovime se smanjuje vrijeme obrade podataka jer se podaci ne spremaju na disk između koraka obrade podataka. Nedostatak modela ETL je da vrijeme obrade podataka traje jednako za inicijalnu obradu podataka i ponovnu obradu podataka. Kao primjer može se uzeti obrada podataka iz datoteke JSON u bazu podataka za jedan poslovni dan. Obrada podataka traje n sekundi za jedan poslovni dan. Ako je potrebna ponovna obrada podataka za isti poslovni dan, tada će obrada podataka opet trajati n sekundi. Do ovog problema dolazi jer se izvorišni podaci ne spremaju u formatu optimalnom za strojnu obradu podataka kao međukorak.

Slika 4.1. Model obrade podataka ETL.

Slika 4.2. Model obrade podataka ELT.

4.2. Model ELT

Model ELT sastoji od tri koraka: izvuci, spremi i obradi. U modelu ELT podaci se izvlače iz izvora podataka, spremaju u ciljano odredište podataka i zatim se obrađuju. Model ELT je prikazan na slici 4.2. Model ELT se koristi za obradu podataka koji se nalaze u različitim izvorima podataka i potrebno ih je obraditi i spremiti u ciljano odredište podataka. Za više informacija o modelu ELT vidjeti [8].

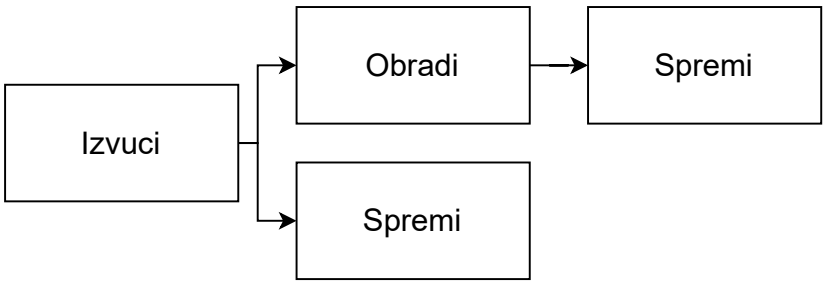
Jedna od prednosti modela ELT je da ponovna obrada podataka može trajati kraće nego inicijalna obrada podataka. Ponovna obrada podataka može trajati kraće ako se u koraku spremi podaci spremaju u formatu optimalnom za strojnu obradu podataka. Nedostatak modela ELT je da inicijalno vrijeme obrade podataka traje dulje nego kod modela ETL jer se podaci spremaju u ciljano odredište podataka prije obrade podataka. Nakon spremanja podaci se moraju ponovno pročitati prije obrade podataka i onda se nakon obrade ponovno zapisuju u konačno odredište.

4.3. Model ETLL

Model ETLL se sastoji od četiri koraka: izvuci, obradi pa spremi i spremi. U model ETLL podaci se izvlače iz izvora podataka zatim se paralelno odvijaju dvije radnje:

1. podaci se obrađuju i spremaju u konačno odredište podataka i
2. podaci se spremaju u međuodredište podataka.

U prvoj radni nad izvučenim podacima se vrše koraci obrade podataka kao što su čišćenje podataka, provjera podataka, transformacija podataka i obogaćivanje podataka. Nakon



Slika 4.3. Model obrade podataka ETLL.

obrade podaci se spremaju u konačno odredište podataka. U drugoj radnji izvučeni podacima se direktno spremaju u međuodredište podataka. model ETLL je prikazan na slici 4.3. Model ETLL se koristi za obradu podataka koji se nalaze u različitim izvorima podataka i potrebno ih je obraditi i spremiti u ciljano odredište podataka.

Jedna od prednosti modela ETLL je da ponovna obrada podataka može trajati kraće nego inicijalna obrada podataka. Ponovna obrada podataka može trajati kraće jer se podaci spremaju u međuodredište podataka u formatu optimalnom za strojnu obradu podataka. Nedostatak modela ETLL je da inicijalno vrijeme obrade podataka traje dulje nego kod modela ETL, ali ne traje dulje nego kod modela ELT jer se prva i druga radnja odvijaju paralelno. Vrijeme obrade je dulje u odnosu na model ETL zbog troškova paralelizma.

5. Implementacija modela obrade podataka

U ovom poglavlju opisana je implementacija modela obrade podataka. Implementacija je izvršena u programskom jeziku Rust, a korištene su biblioteke datafusion i deltalake. Model ETL ima jednu implementaciju, modela ELT dvije implementacije i model ETLL ima jednu implementaciju. Za potrebe ovog rada razvijen je meta algoritam obrade podataka koji je prikazan u ispisu 5..1 Meta algoritam obrade podataka je jednostavan i sastoji se od dvije funkcije: *predobrada_datoteke* i *obrada_datoteke*. Funkcija *predobrada_datoteke* vrši potrebne predobrade nad datotekom, a funkcija *obrada_datoteke* vrši obradu datoteke. Svaka implementacija modela obrade podataka koristi meta algoritam kao kostur i nadodaje potrebne funkcionalnosti za obradu podataka. Meta algoritam je iterativan i obradu vrši nad svakom datotekom iz liste datoteka. Meta algoritam je razvijen kako bi implementacije modela obrade podataka bile konzistentne radi potreba vrednovanja rješenja.

Ispis 5..1: Meta algoritam obrade podataka.

```
datoteke = popis putanja datoteka
za datoteka u datoteke:
    datoteka = predobrada_datoteke(datoteka)
    obrada_datoteke(datoteka)
```

5.1. Zajedničke funkcionalnosti

Zajedničke funkcionalnosti implementacija modela obrade podataka su funkcionalnosti koje se koriste u svim implementacijama. Zajedničke funkcionalnosti su:

- predobrada datoteke — funkcionalnost koja vrši predobradu datoteke,

- brončana obrada — funkcionalnost koja vrši obradu podataka sa spremanjem u brončani sloj i
- srebrna obrada — funkcionalnost koja vrši obradu podataka sa spremanjem u srebrni sloj.

Navedene funkcionalnosti su implementirane u modulu *common_functionality* i koriste se u svim implementacijama modela obrade podataka. Navedene funkcionalnosti nisu sve zajedničke funkcionalnosti, ali su najvažnije za potrebe ovog rada.

5.1.1. Predobrada datoteke

U ovom poglavlju opisan je algoritam za predobradu datoteke. Algoritam obrađuje datoteku tako da čita datoteku strujanjem bajtova. Bajtovi se spremaju u međuspremnik retka dok se ne pročita znak za novi redak. Redak je niz znakova koji sadrži polja odvojena znakom za odvajanje polja. Polje je niz znakova koji predstavljaju jedan podatak. Kada se pročita znak za novi redak, redak se dopunjava znakovima za odvajanje polja tako da svaki redak ima šest polja. Broj polja u retku je jednak broju znakova za odvajanje polja plus jedan. Redak se dopunjava do šest polja jer je to najveći broj polja koji se pojavljuje u izvoznoj datoteci iz baze podataka (3.2.). Dopunjavanje se vrši funkcijom *dodaj_polja_desno*. Dopunjavanje je potrebno jer biblioteka datafusion očekuje da svaki redak ima isti broj polja, a po modelu izvozne datoteke iz poglavlja 3.2. redci nemaju isti broj polja. Nakon nadopunjavanja obrade redak se zapisuje u privremenu datoteku. Nakon obrade svih redaka privremena datoteka se vraća kao rezultat funkcije. Algoritam predobrade datoteke prikazan je u ispisu 5..2 Za implementaciju algoritma predobrade datoteke vidjeti funkciju *pre_process_file* u datoteci [9].

Ispis 5..2: Algoritam za predobradu datoteke.

```
meduspremnik = []
za bajt u datoteka:
    ako bajt nije znak za novi redak:
        meduspremnik.dodaj(bajt)
    inace:
        redak = dodaj_polja_desno(meduspremnik, znakovi_za_odvajanje)
        privremena_datoteka.dodaj(redak)
        meduspremnik.ocisti()
```

5.1.2. Brončana obrada

U ovom poglavlju opisan je algoritam za obradu podataka u brončani sloj. Algoritam prima podatke za obradu i zapisuje ih u brončani sloj. Algoritam obrade podataka u brončani sloj prikazan je u ispisu 5..3 Za implementaciju algoritma obrade podataka u brončani sloj vidjeti funkciju *process_bronze* u datoteci [9].

Ispis 5..3: Algoritam za obradu podataka u brončani sloj.

```
podaci_za_obradu
spremi_podatke(broncana_tablica, podaci_za_obradu)
```

5.1.3. Srebrna obrada

U ovom poglavlju opisan je algoritam za obradu podataka u srebrni sloj. Najprije se primaju podaci za obraditi. Podaci se dijele na dimenzije i činjenice. Dimenzije se spremaju u dimenzijske tablice, a činjenice u činjeničnu tablicu. Dimenzijske tablice se spremaju u tablice srebrnog sloja koristeći operaciju MERGE. Operacija MERGE omogućuje ažuriranje i umetanje podataka u tablicu. Korištenjem operacije MERGE osigurava se da dimenzijske tablice imaju točne podatke. Činjenična tablica se sprema u tablicu srebrnog sloja koristeći operaciju INSERT. Operacija INSERT omogućuje umetanje podataka u tablicu. Korištenjem operacije INSERT omogućuje se brzi unos podataka. Algoritam obrade podataka u srebrni sloj prikazan je u ispisu 5..4 Za implementaciju algoritma obrade podataka u srebrni sloj vidjeti funkciju *process_silver* u datoteci [9].

Ispis 5..4: Algoritam za obradu podataka u srebrni sloj.

```
podaci_za_obradu
dimenzije = dimenzije_iz_podataka(podaci_za_obradu)
cinjenica = cinjenica_iz_podataka(podaci_za_obradu, dimenzije)
spremi_dimenzije_srebrni_sloj(dimenzije, MERGE)
spremi_cinjenicu_srebrni_sloj(cinjenica, INSERT)
```

5.2. Implementacija modela ETL

Implementacija modela obrade podataka ETL koristi zajedničke funkcionalnosti opisane u poglavlju 5.1. Implementacija modela obrade podataka ETL slijedi meta algoritam obrade podataka prikazan u primjeru koda 5..1 Implementacija modela obrade podataka

ETL se sastoji od tri koraka: predobrada datoteke, čitanje datoteke i obrada podataka u srebrni sloj. Navedeni koraci se ponavljaju za svaku datoteku iz liste datoteka. Algoritam implementacije modela ETL prikazan u ispisu 5..5 Za implementaciju algoritma modela ETL vidjeti datoteku [10].

Ispis 5..5: Algoritam implementacije modela obrade podataka ETL.

```
datoteke = popis putanja datoteka
za datoteka u datoteke:
    privremena_datoteka = pre_process_file(datoteka)
    podatkovni_okvir = procitaj_datoteku(privremena_datoteka)
    process_silver(podatkovni_okvir)
```

5.3. Implementacija modela ELT

Za implementaciju modela obrade podataka ELT razvijene su dvije implementacije. Obje implementacije modela obrade podataka ELT koriste zajedničke funkcionalnosti opisane u poglavlju 5.1.

Prva implementacija modela obrade podataka ELT slijedi meta algoritam obrade podataka prikazan u ispisu 5..1 Prva implementacija modela ELT sastoji se od pet koraka: predobrada datoteke, čitanje datoteke, obrada podataka u brončani sloj, čitanje brončane tablice i obrada podataka u srebrni sloj. Navedeni koraci se ponavljaju za svaku datoteku. Algoritam prve implementacije modela ELT prikazan u ispisu 5..6 Za implementaciju prvog algoritma modela ELT vidjeti datoteku [11].

Druga implementacija modela ELT (model ETL2) obrade podataka slijedi modificirani meta algoritam obrade podataka prikazan u ispisu 5..1 Model ETL2 sastoji se od tri koraka koji se ponavljaju za svaku datoteku: predobrada datoteke, čitanje datoteke i obrada podataka u brončani sloj. Nakon obrade svih datoteka čita se brončana tablica i vrši se obrada podataka u srebrni sloj. Algoritam modela ETL2 prikazan u ispisu 5..7 Drugi algoritam je razvijen jer je pretpostavka da je čitanje svih podatka iz brončanog sloja i obrada podataka u srebrni sloj brža od obrade za svaku datoteku. Za implementaciju algoritma modela ELT2 vidjeti datoteku [12].

Ispis 5..6: Prvi algoritam implementacije modela obrade podataka ELT.

```
datoteke = popis putanja datoteka
za datoteka u datoteke:
    privremena_datoteka = pre_process_file(datoteka)
    podatkovni_okvir = procitaj_datoteku(privremena_datoteka)
    process_bronze(podatkovni_okvir)
    podatkovni_okvir = procitaj_broncanu_tablicu()
    process_silver(podatkovni_okvir)
```

Ispis 5..7: Drugi algoritam implementacije modela obrade podataka ELT.

```
datoteke = popis putanja datoteka
za datoteka u datoteke:
    privremena_datoteka = pre_process_file(datoteka)
    podatkovni_okvir = procitaj_datoteku(privremena_datoteka)
    process_bronze(podatkovni_okvir)
podatkovni_okvir = procitaj_broncanu_tablicu()
process_silver(podatkovni_okvir)
```

5.4. Implementacija modela ETLL

Implementacija modela obrade podataka ETLL koristi zajedničke funkcionalnosti opisane u poglavlju 5.1. Implementacija modela obrade podataka ETLL slijedi modificirani meta algoritam obrade podataka prikazan u ispisu 5..1 Implementacija modela obrade podataka ETLL se sastoji od tri koraka: predobrada datoteke, čitanje datoteke i obrada podataka u brončani i srebrni sloj. Obrada podataka u brončani i srebrni sloj izvršava se paralelno. U ovoj implementaciji modela ETLL prva grana obrađuje i sprema podatke u srebrni sloj, a druga grana sprema podatke u brončani sloj 4.3. Navedeni koraci se ponavljaju za svaku datoteku. Algoritam implementacije modela ETLL prikazan u ispisu 5..8 Za implementaciju algoritma modela ETLL vidjeti datoteku [13].

Ispis 5..8: Algoritam implementacije modela obrade podataka ETLL.

```
datoteke = popis putanja datoteka
za datoteka u datoteke:
    privremena_datoteka = pre_process_file(datoteka)
    podatkovni_okvir = procitaj_datoteku(privremena_datoteka)
    paralelno:
        process_bronze(podatkovni_okvir)
        process_silver(podatkovni_okvir)
```

6. Ispitivanje performanci modela obrade podataka

U ovom poglavlju opisano je ispitivanje performanci modela obrade podataka. Ispitivanje je provedeno na skupu podataka koji je generiran za potrebe ovog rada. Skupovi podataka su generirani tako da za svaku ukupnu količinu podataka, 64 MiB, 128 MiB i 256 MiB, se generira 1, 64 i 128 datoteka. Svaka datoteka sadrži sve tablice opisane u poglavlju 3. Za svaku tablicu je određeno koji postotak podataka zauzima. Tablice *User*, *City*, *Producer* i *Item* svaka zauzimaju 10 posto podataka, dok tablica *Transaction* zauzima 60 posto podataka. Brojčani podaci su generirani kao slučajni brojevi u rasponu po normalnoj distribuciji. Na primjer za stupac *Item.price* 3.1. cjelobrojni dio je generiran kao cjelobrojni slučajni broj u rasponu od 0 do 1000, a decimalni dio je generiran kao cjelobrojni slučajni broj u rasponu od 0 do 99. Tekstualni podaci su generirani kao slučajno izabrani tekst iz liste riječi. Na primjer za stupac *Item.item_name* 3.1. je generiran kao slučajno izabrani tekst iz liste riječi koja sadrži 100 riječi. Ostali stupci tablica su generirani na sličan način. Za cjeloviti uvid u generiranje podataka vidjeti [14].

Kod ispitivanja za svaku implementaciju modela obrade podataka mjere se sljedeće metrike: ukupno vrijeme izvođenja, vrijeme predobrade datoteka, vrijeme brončane obrade (ne mjeri se kod implementacije modela ETL) i vrijeme srebrne obrade. Sve metrike se mjere u nanosekundama radi preciznosti, ali rezultati su prikazani na razini sekunde radi jednostavnijeg razumijevanja. Mjerenjem navedenih metrika dobivamo uvid u vremenske performance implementacija modela obrade podataka.

Svaka implementacija modela obrade podataka se pokreće za dva slučaja: inicijalna obrada i ponovna obrada. Inicijalna obrada je obrada kod koje u brončanim i srebrnim tablicama ne postoje podaci. Ponovna obrada je obrada kod koje u brončanim i srebrnim tablicama već postoje podaci. Navedeni slučajevi se pokreću jedan za drugim. Navedena

mjerenja se pokreću 20 puta za svaku implementaciju modela obrade podataka i za svaki skup podataka. Skripta za ispitivanje performanci modela obrade podataka je prikazana u ispisu 6..1

Ispis 6..1: Skripta za ispitivanje performanci modela obrade podataka

```
$size = @(64, 128, 256)
$file_count = @(1, 64, 128)
$runtimes = $( "etl", "elt", "elt2", "etll" )

foreach ($s in $size) { foreach ($f in $file_count) {
    $batch = "batch_$( $s )_$( $f )"
    foreach ($r in $runtimes) { for ($i = 1; $i -le 20; $i++){
        if (Test-Path "lakehouse")
        {
            Remove-Item -LiteralPath "lakehouse" -Recurse
        }
        for ($j = 1; $j -le 2; $j++) {
        Write-Output "Running $( $r ) $( $batch ) $( $i ) $( $j )"
        & "target\release\$( $r ).exe" --batch-id $j data\$batch \
            >> logs\$( $r )\$($batch).jsonl
        Write-Output "Finished $( $r ) $( $batch ) $( $i ) $( $j )"
        }
    }}
}}
```

Ispitivanje se provodi na računalu s operacijskim sustavom Windows 11 Pro 64-bit i sa specifikacijama:

- procesor — AMD Ryzen 9 3900X,
- radna memorija — 32 GB Dual-Channel DDR4 @ 1589MHz (16-18-18-36)
- matična ploča — ASUSTeK COMPUTER INC. TUF GAMING X570-PLUS (WI-FI) (AM4) i
- pohrana — WD_BLACK SN850X HS 1000 GB

7. Rezultati ispitivanja

U ovom poglavlju opisani su očekivani rezultati ispitivanja implementacija modela obrade podataka te su prikazani i opisani dobiveni rezultati ispitivanja. Rezultati ispitivanja su uspoređeni s očekivanim rezultatima ispitivanja. Očekivani rezultati ispitivanja su opisani u poglavlju 7.1. Dobiveni rezultati ispitivanja su prikazani u poglavlju 7.2.

7.1. Očekivani rezultati ispitivanja

Očekivani rezultati ispitivanja su da će implementacija modela obrade podataka ETL imati najkraće vrijeme izvođenja, dok će implementacija modela obrade podataka ELT imati najduže vrijeme izvođenja. Implementacija modela obrade podataka ELT2 će imati kraće vrijeme izvođenja od implementacije modela obrade podataka ELT. Implementacija modela obrade podataka ETLL će imati vrijeme izvođenja između modela ETL i ELT. Model obrade podataka ETL će imati najkraće vrijeme izvođenja jer se podaci samo jednom učitavaju u tablice. Model obrade podataka ELT će imati najduže vrijeme izvođenja jer se podaci prvo učitavaju u brončane tablice, zatim se čitaju iz brončane tablice i na kraju se učitavaju u srebrne tablice. Model obrade podataka ELT2 će imati kraće vrijeme izvođenja od modla ELT jer se podaci samo jednom čitaju iz brončane tablice i zapisuju u srebrne tablice. Model obrade podataka ETLL će imati vrijeme izvođenja između modela ETL i ELT jer se podaci paralelno učitavaju u brončane tablice i srebrne tablice. Paralelna obrada u model obrade podataka ETLL dodaje vremensku zadršku kod stvaranja dretvi te zbog toga će vrijeme izvođenja biti veće od modela ETL obrade podataka. Vrijeme izvođenja modela obrade podataka ETLL će biti manje od modela obrade podataka ELT jer učitavanje podataka u srebrne tablice ne čeka na unos podataka u brončanu tablicu.

Vrijeme predobrade datoteka će trajati jednako za sve implementacije modela obrade podataka jer se datoteke predobrađuje na isti način. Vrijeme obrade brončanog sloja će

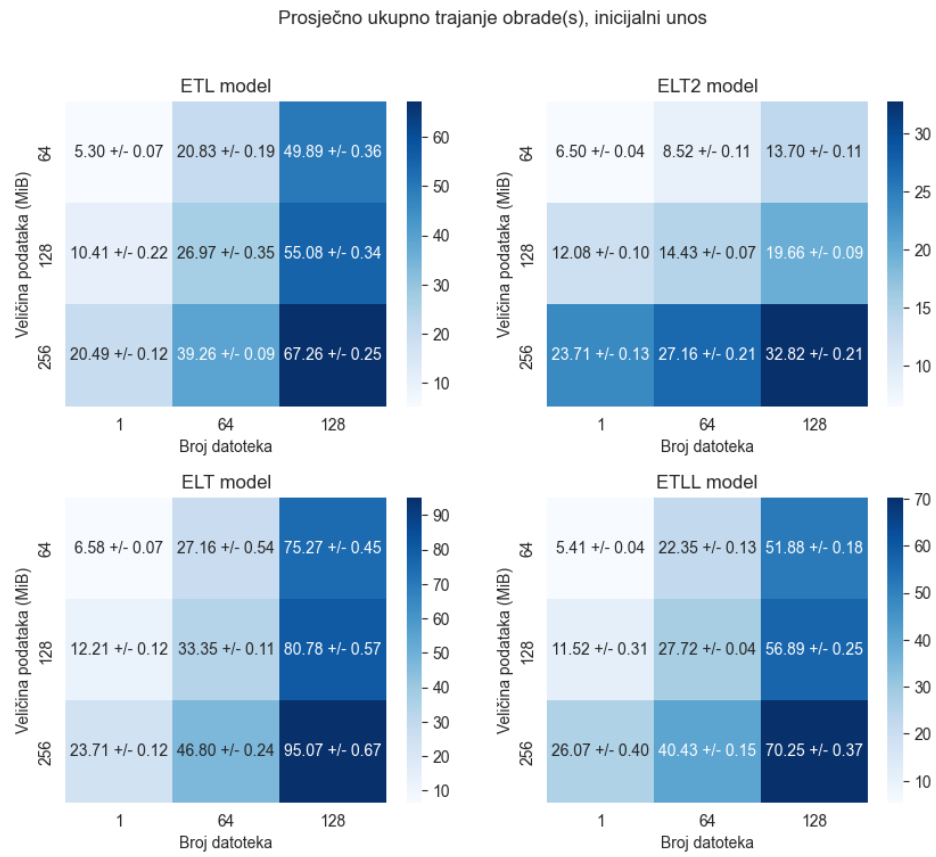
trajati jednako za sve implementacije modela obrade podataka jer se podaci učitavaju u brončane tablice na isti način. Vrijeme obrade će trajati jednako za modele obrade podataka ETL i ETLL jer se podaci učitavaju u srebrne tablice na isti način. Vrijeme obrade srebrnog sloja će trajati duže za model obrade podataka ELT jer se podaci prvo čitaju iz brončane tablice, a zatim se učitavaju u srebrne tablice. Vrijeme obrade srebrnog sloja za model obrade podataka ELT2 će trajati kraće od modela obrade podataka ELT jer se podaci samo jednom čitaju iz brončane tablice i zapisuju u srebrne tablice. Vrijeme obrade za sve metrike će trajati duže kod ponovne obrade jer se podaci već nalaze u tablicama i potrebno je provesti ažuriranje podataka. Za sve metrike će vrijeme obrade ovisiti o veličini podataka i broju datoteka.

7.2. Dobiveni rezultati ispitivanja

Toplinske mape prosječnog ukupnog trajanja obrade za inicijalni unos i ponovni unos su prikazane na slikama 7.1. i 7.2. Vrijeme trajanja obrade je prikazano u sekundama. Iz slika 7.1. i 7.2. se vidi da implementacija modela obrade podataka ELT2 ima najkraće vrijeme obrade podataka što odudara od očekivanih rezultata. Do odudaranja dolazi zbog značajno brže obrade podataka u srebrne tablice. Ostali modeli obrade podataka imaju vrijeme obrade koje odgovara očekivanim rezultatima. Vrijeme obrade za sve modele obrade podataka je veće kod ponovnog unosa što odgovara očekivanim rezultatima.

Toplinske mape prosječnog ukupnog trajanja predobrade za inicijalni unos i ponovni unos su prikazane na slikama 7.3. i 7.4. Vrijeme trajanja predobrade je prikazano u sekundama. Iz slika 7.3. i 7.4. se vidi da vrijeme predobrade datoteka je približno jednako za sve modele obrade podataka što odgovara očekivanim rezultatima. Vrijeme predobrade datoteka ovisi samo o veličini podataka što odudara od očekivanog rezultata. Moguće objašnjenje odstupanja je da otvaranje datoteke je vremenski zanemarivo te zbog toga vrijeme predobrade podataka ne ovisi o broju datoteka. Graf na slici 7.5. prikazuje trend trajanja predobrade podataka u ovisnosti o količini podataka. Iz grafa se vidi da vrijeme predobrade podataka raste linearno s količinom podataka, dok vrijeme predobrade podataka ne ovisi o broju datoteka.

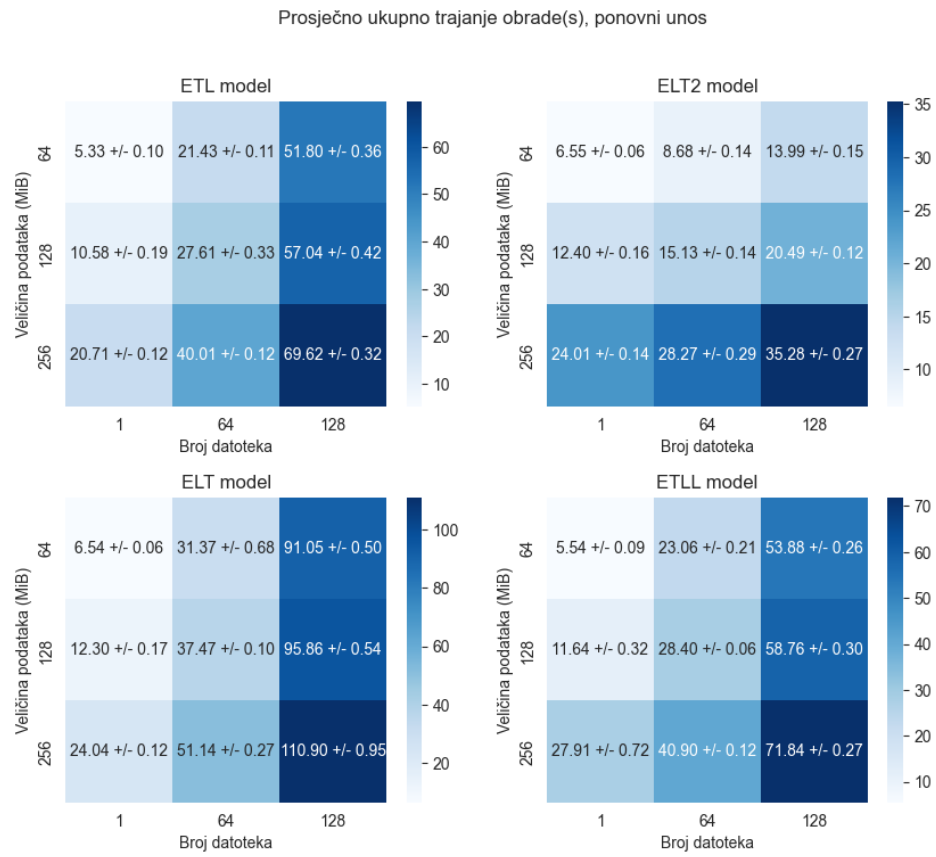
Toplinske mape prosječnog ukupnog trajanja obrade brončanog sloja za inicijalni unos i ponovni unos su prikazane na slikama 7.6. i 7.7. Vrijeme trajanja obrade bronča-



Slika 7.1. Toplinska mapa prosječnog ukupnog trajanja obrade u sekundama za inicijalni unos.

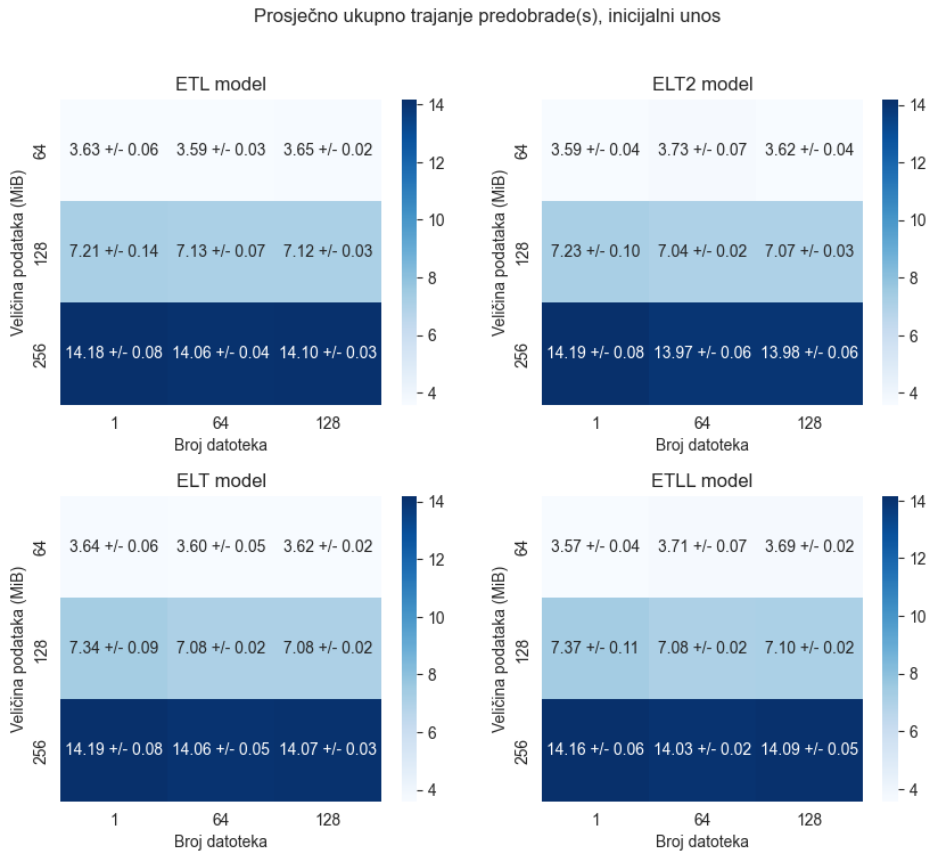
nog sloja je u sekundama. Iz slika 7.6. i 7.7. se vidi da vrijeme obrade brončanog sloja je približno jednako za sve modele obrade podataka što odgovara očekivanim rezultatima. Postoji anomalija kod modela ETLL za sve skupove podataka koji imaju jednu datoteku. U tom slučaju trajanje obrade brončanog sloja je drastično veće od ostalih modela obrade podataka. Uzrok anomalije nije poznat, ali pretpostavka je da je uzrok povezan s paralelnom obradom podataka. Iz grafova se vidi da vrijeme obrade brončanog sloja traje dulje kod ponovnog unosa što odgovara očekivanim rezultatima. Također je vidljivo da vrijeme obrade brončanog sloja ovisi i o veličini podataka i o broju datoteka.

Toplinske mape prosječnog ukupnog trajanja obrade srebrnog sloja za inicijalni unos i ponovni unos su prikazane na slikama 7.8. i 7.9. Vrijeme trajanja obrade srebrnog sloja je u sekundama. Iz slika 7.8. i 7.9. se vidi da vrijeme obrade srebrnog sloja je najkraće za model obrade podataka ELT2 što djelomično odgovara očekivanim rezultatima. Očekivanje je da će model obrade podataka ELT2 trajati kraće od modela obrade podataka ELT. Vrijeme obrade srebrnog sloja za model obrade podataka ELT2 može se objasniti time što se podaci čitaju iz formata pogodnog za strojnu obradu, podaci se čitaju samo jednom i

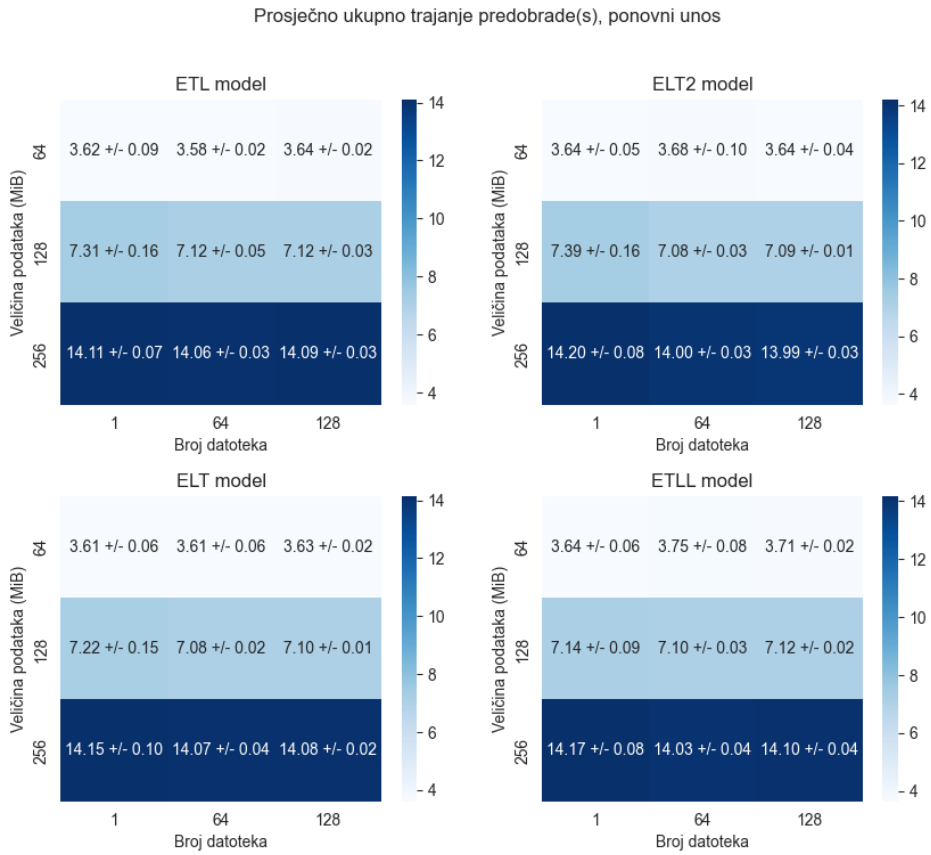


Slika 7.2. Toplinska mapa prosječnog ukupnog trajanja obrade u sekundama za ponovni unos.

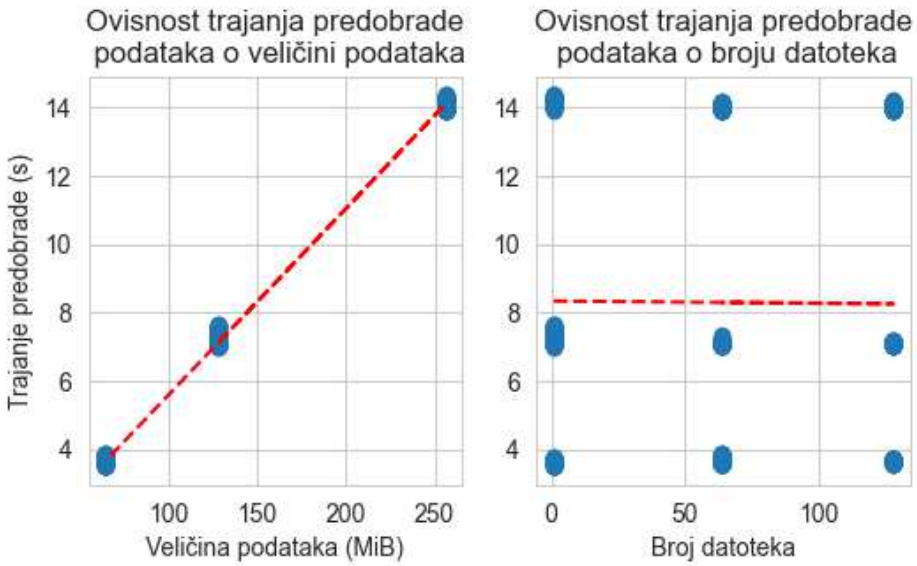
zapisuju se samo jednom. Navedenim se značajno smanjuje broj ulazno-izlaznih operacija na disku što smanjuje vrijeme obrade podataka. Vrijeme obrade srebrnog sloja je najduže za modela obrade podataka ELT što odgovara očekivanim rezultatima. Vrijeme obrade srebrnog sloja za model obrade podataka ETLL je između modela obrade podataka ETL i ELT što odgovara očekivanim rezultatima. Vrijeme obrade srebrnog sloja za sve modele obrade podataka ovisi o veličini podataka i broju datoteka. Vrijeme obrade srebrnog sloja je veće kod ponovnog unosa što odgovara očekivanim rezultatima.



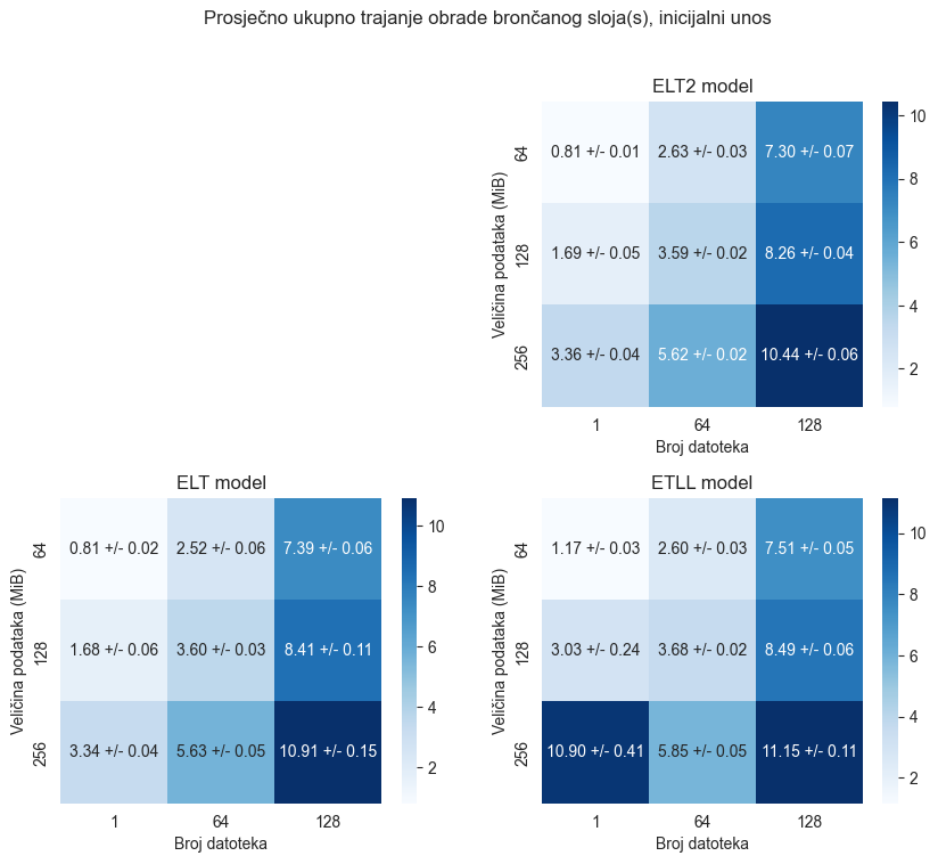
Slika 7.3. Toplinska mapa prosječnog ukupnog trajanja predobrade u sekundama za inicijalni unos.



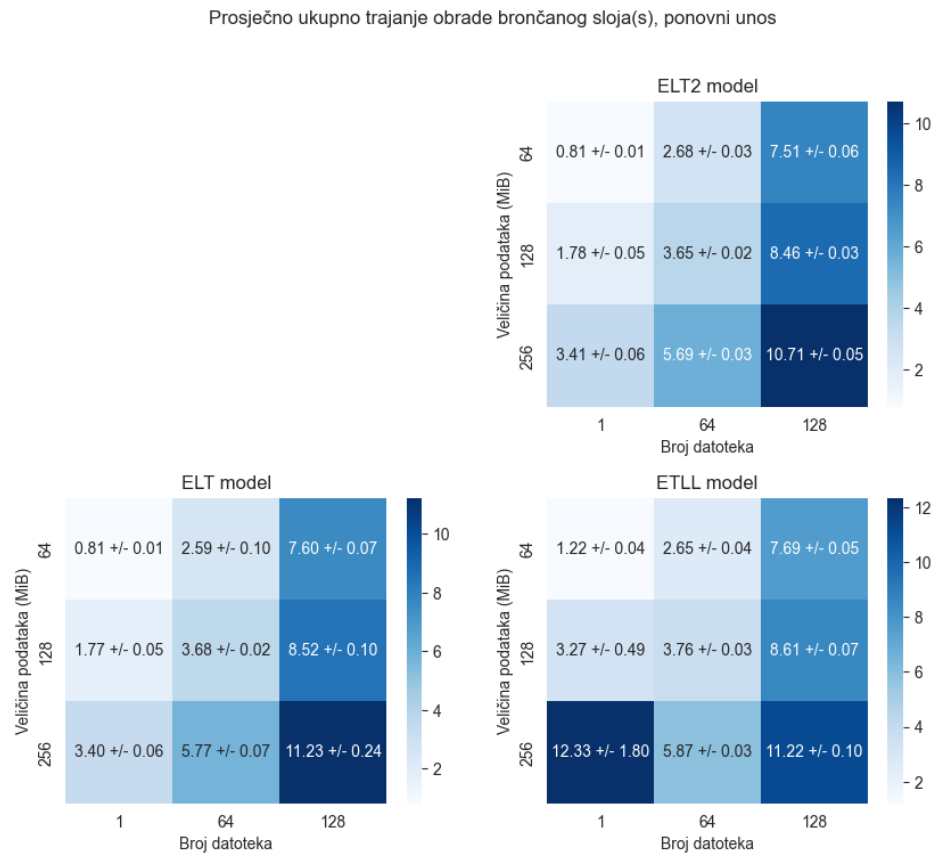
Slika 7.4. Toplinska mapa prosječnog ukupnog trajanja predobrade u sekundama za ponovni unos.



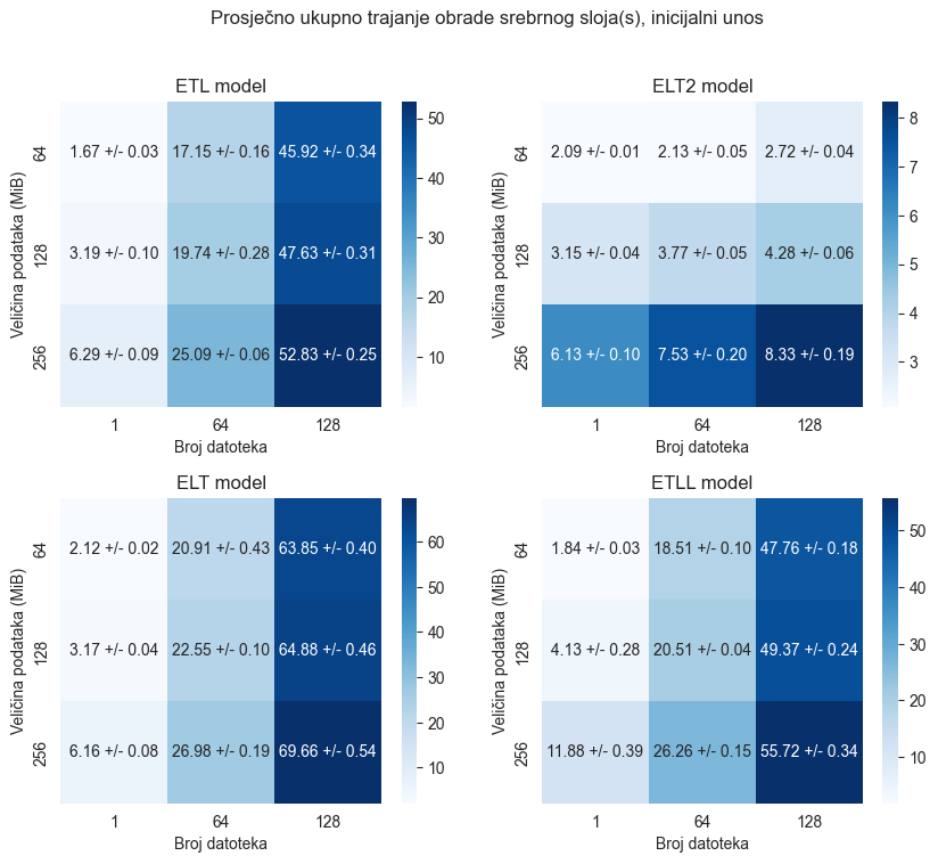
Slika 7.5. Graf trenda trajanja predobrade podataka u ovisnosti o količini podataka i broju datoteka.



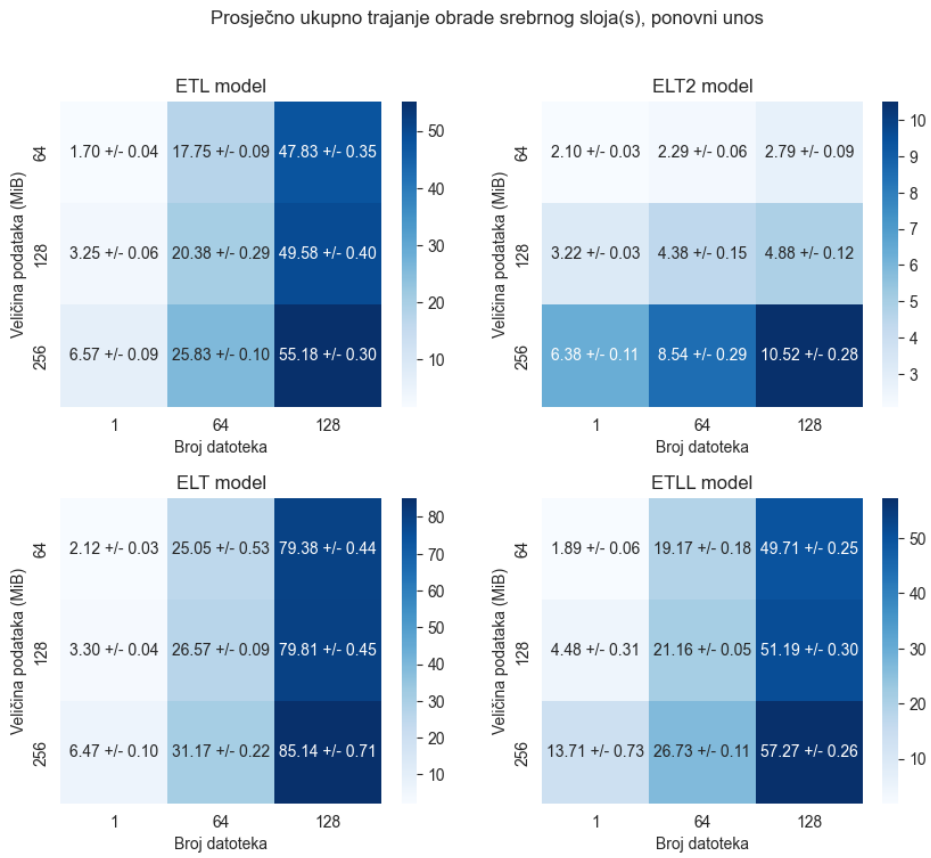
Slika 7.6. Toplinska mapa prosječnog ukupnog trajanja obrade brončanog sloja u sekundama za inicijalni unos.



Slika 7.7. Toplinska mapa prosječnog ukupnog trajanja obrade brončanog sloja u sekundama za ponovni unos.



Slika 7.8. Toplinska mapa prosječnog ukupnog trajanja obrade srebrnog sloja u sekundama za inicijalni unos.



Slika 7.9. Toplinska mapa prosječnog ukupnog trajanja obrade srebrnog sloja u sekundama za ponovni unos.

8. Rasprava i zaključak

U ovom diplomskom radu istraženi su modeli ETL, ELT i ETLL za obrade polustrukturiranih tekstualnih datoteka poslovnih transakcijskih podataka. Kroz analizu i eksperimentalnu evaluaciju, došli smo do nekoliko važnih zaključaka koji mogu biti korisni za daljnju primjenu u poslovnom okruženju.

Jedan od ključnih uvida jest da smanjenje broja ulazno-izlaznih (I/O) operacija značajno poboljšava performance obrade podataka. Manji broj I/O operacija smanjuje vrijeme potrebno za prijenos podataka između sustava za pohranu i obrade, što dovodi do bržeg izvršavanja procesa. Stoga, pri dizajnu sustava za obradu podataka, preporučuje se optimizacija postupaka kako bi se minimizirale nepotrebne I/O operacije.

Paralelna obrada podataka kod modela ETLL pokazala se kao učinkovit način za smanjenje vremena obrade podataka u odnosu na model ELT. Izvršavanjem obrade i učitavanja podataka u konačne tablice u jednoj dretvi i učitavanjem podataka u međutablice u drugoj dretvi, postiže se brzina obrade koja je usporediva s modelom ETL. Također omogućuje se efikasna ponovna obrada podataka kao kod modela ELT.

Na temelju provedenih eksperimenata, može se zaključiti da svaki od proučavanih modela obrade podataka ima svoje specifične prednosti i nedostatke, ovisno o kontekstu primjene. Model ETL je pogodan za situacije gdje je potrebna temeljita transformacija podataka prije pohrane, dok je model ELT učinkovitiji kada se transformacije mogu izvoditi unutar sustava za pohranu. Model ETLL, kombinirajući prednosti oba pristupa, nudi fleksibilnost i robusnost u specifičnim scenarijima.

Modeli obrade podataka istraženi u ovom radu mogu se primijeniti u različitim poslovnim okruženjima, od malih tvrtki do velikih korporacija. Modeli su agnostični prema izvorima podataka te tehnologijama za obradu i pohranu podataka, što ih čini prilagod-

ljivima i skalabilnima za različite primjene. Modeli se mogu koristiti za obradu podataka s API-ja pomoću *Apache Sparka* i pohraniti u *SQL Server*, kao i, iako manje vjerojatno, za obradu podataka iz izvoza baze podataka kao SQL skripte koristeći programski jezik Perl i pohranu u tekstualne datoteke. Navedeni primjeri su ilustrativnog karaktera, ali ukazuju na širok spektar primjena modela obrade podataka. Kao realni primjer može se uzeti sljedeći scenarij: tvrtka A koja posluje u maloprodaji dostavlja podatke o prodaji iz svojih trgovina na poslužitelj SFTP u formatu *Parquet* tvrtki B. Tvrtka B se bavi skladištenjem i analizom podatka. Ona koristi programski jezik *Python* za kopiranje podataka s poslužitelja SFTP na lokalno računalo, a zatim koristi *Apache Spark* za obradu podataka i pohranu u bazu podataka *SQL Server*. Tvrtka B je odabrala model ETL za obradu podataka jer tvrtka A često ponovno šalje podatke, a model ETL omogućuje jednostavnu ponovnu obradu sirovih podataka. Ako su sirovi podaci stabilni, tvrtka B bi mogla koristiti ili model ELT ili model ETLL.

U konačnici, izbor optimalnog modela obrade podataka treba biti vođen specifičnim zahtjevima poslovnog okruženja, uključujući potrebu za brzinom obrade, resursnom učinkovitošću i složenošću implementacije. Ovaj rad pruža smjernice koje mogu pomoći u donošenju informiranih odluka pri odabiru i implementaciji sustava za obradu polustrukturiranih tekstualnih podataka.

Budući rad na ovu temu može se fokusirati na smanjenje ulazno-izlaznih operacija u pojedinačnim modelima, kao i na razvoj novih modela obrade podataka koji će kombinirati prednosti modela ETL, ELT i ETLL. Također, moguće je istražiti primjenu različitih tehnologija i alata za obradu podataka, kako bi se dodatno poboljšale performance i resursna učinkovitost sustava.

Bibliografija

[1] Michael Armbrust i dr. *Delta Lake: High-Performance ACID Table Storage over Cloud Object Stores*. Teh. izv. Databricks, 2020.

[2] Delta-IO. *Delta Lake Documentation*. https://delta-io.github.io/delta-rs/. 2024.

[3] Apache Software Foundation. *Apache DataFusion*. https://datafusion.apache.org/. 2024.

[4] Rust Team. *Rust*. https://www.rust-lang.org/. 2024.

[5] Ivan Derdić. *Po četak rada s jezerskim skladištem podataka: teorija i tehnologija*. https://github.com/IvanDerdicDer/seminar2/blob/89814f414ec64c848fe98e491123d04seminar.pdf. 2023.

[6] Chen Cuello. *What is Dimensional Data Modeling?* https://rivery.io/data-learning-center/dimensional-data-modeling-guide/. 2023.

[7] Kihara Kimachia. *What is ETL? (Extract, Transform, Load): The Ultimate Guide*. https://www.techrepublic.com/article/what-is-etl/. 2024.

[8] DBT. *What is ELT (Extract, Load, Transform)?* https://docs.getdbt.com/terms/elt. 2024.

[9] Ivan Derdić. *Common Functionality*. https://github.com/IvanDerdicDer/diplomski/blob/842cd5dfdcec78501d1343a6fb6a8b36b8b98069/common_functionality/src/lib.rs. 2024.

[10] Ivan Derdić. *ETL Data Processing*. https://github.com/IvanDerdicDer/diplomski/blob/a502fbaa861c654c299304d20dbcf1f957f586c6/src/etl/main.rs. 2024.

[11] Ivan Derdić. *ELT Data Processing*. https://github.com/IvanDerdicDer/diplomski/blob/a502fbaa861c654c299304d20dbcf1f957f586c6/src/elt/main.rs. 2024.

[12] Ivan Derdić. *ELT2 Data Processing*. https://github.com/IvanDerdicDer/diplomski/blob/a502fbaa861c654c299304d20dbcf1f957f586c6/src/elt2/main.rs. 2024.

[13] Ivan Derdić. *ETLL Data Processing*. https://github.com/IvanDerdicDer/diplomski/blob/a502fbaa861c654c299304d20dbcf1f957f586c6/src/etll/main.rs. 2024.

[14] Ivan Derdić. *ETLL Data Processing*. https://github.com/IvanDerdicDer/diplomski/blob/a502fbaa861c654c299304d20dbcf1f957f586c6/src/data_generator/main.rs. 2024.

Sažetak

Sustav za obradu polustrukturiranih tekstnih datoteka poslovnih transakcijskih podataka

Ivan Derdić

Ovaj diplomski rad bavi se razvojem sustava za obradu polustrukturiranih tekstualnih datoteka poslovnih transakcijskih podataka. U radu se analiziraju i implementiraju tri modela obrade podataka: ETL (izvuci, obradi, spremi), ELT (izvuci, spremi, obradi) i ETLL (izvuci, obradi i spremi, spremi). Implementacija koristi tehnologije Delta Lake i Apache DataFusion, omogućujući učinkovitu obradu velikih količina podataka i osiguravajući svojstva ACID. Cilj rada je usporediti performance navedenih modela u kontekstu brzine obrade. Eksperimentalni rezultati pokazuju prednosti i nedostatke svakog modela, ovisno o specifičnim uvjetima i zahtjevima obrade podataka.

Model ELT2 ima najbolje performance u odnosu na modele ETL, ELT i ETLL. Model ELT2 prvo sekvencijalno obradi datoteke u brončanu tablicu i zatim sve podatke iz brončane tablice obradi u srebrne tablice. Model ELT2 je najbrži jer se podaci iz brončane tablice učitavaju u memoriju samo jednom.

Ključne riječi: Obrada podataka; model ETL; model ELT; model ETLL; Delta Lake; Apache DataFusion; Poslovni transakcijski podaci

Abstract

System for processing semi-structured text files of transactional business data

Ivan Derdić

This thesis addresses the development of a system for processing semi-structured text files of business transaction data. The study analyzes and implements three data processing models: including ETL (Extract, Transform, Load), ELT (Extract, Load, Transform), and ETLL (Extract, Transform, Load, Load) models. The implementation utilizes Delta Lake and Apache DataFusion technologies, enabling efficient processing of large data volumes while ensuring ACID properties. The aim is to compare the performance of these models in terms of processing speed. Experimental results highlight the advantages and disadvantages of each model, depending on specific conditions and data processing requirements.

The ELT2 model has the best performance compared to the ETL, ELT, and ETLL models. The ELT2 model first sequentially processes files into the bronze table and then processes all the data from the bronze table into the silver tables. The ELT2 model is the fastest because the data from the bronze table is loaded into memory only once.

Keywords: Data processing; ETL model; ELT model; ETLL model; Delta Lake; Apache DataFusion; Business transaction data