SVEUČILIŠTE JURJA DOBRILE U PULI

FAKULTET INFORMATIKE U PULI

Skladišta i rudarenje podataka

**PRODAJA NEKRETNINA U MELBOURNE-U 2016 – 2018 GODINE**

SEMINARSKI RAD

Mateo Kocev, 0303104813, izvanredni student

Informatika

Pula, 22.5.2024.

### 2. Odabir dataseta / skupa podataka

Podaci su učitani iz CSV datoteke pronađene na poveznici: <https://www.kaggle.com/datasets/ronikmalhotra/melbourne-housing-dataset>

U skupu podataka pronalazimo razne informacije o prodaji nekretnina u periodu od veljače 2016. godine do ožujka 2018. godine koje na lokaciji imaju sagrađenu i kuću.

Skup podataka smo podijelili na dva dijela kako bi pokušali realizirati simulaciju realistične situacije gdje nam se podaci neće nalaziti u jednom izvoru, već je na nama da dostupne podatke obradimo i pripremimo za rad.

### 2.1. Opis problema i cilj analize

Tržište nekretnina u Melbourneu suočava se s nizom izazova koji utječu na trgovanje nekretninama. Cijene nekretnina mogu biti vrlo promjenjive te stvoriti nesigurnost u donošenju odluka kod kupaca, prodajnih agenata i investitora što dovodi do financijskog gubitka ili propuštenih prilika. Ovaj problem možemo riješiti analizom povijesnih podataka te identifikacijom trendova na tržištu pružajući uvid ne samo u opčenite trendove već gledajući performansu samih prodajnih agenata.

Problem nam također predstavlja veliku količinu faktora poput lokacije, veličine, dob nekretnine i dostupnih komoditeta koji mogu utjecati na cijene nekretnina. Teško je odrediti koji su najvažniji faktori koje moramo uzeti u obzir no kroz detaljnu analizu podataka ciljamo identificirati najveće utjecaje na kretanje tržišta kako bi mogli dovesti realne i konkurentne poslovne odluke.

Ciljamo optimizirati strategiju prodaje, kupnje i korištenja vremena kako bi minimizirali rizik te maksimizirali dobit financijskih sredstava donoseći pravilne poslovne odluke koristeći primjerene analitičke alate i modele. Također ciljamo održavati veću transparentnost i dostupnost informacija o često privatnim transakcijama kroz prikupljanje, saniranje i obrađivanje podataka kako bi naši prodajni agenti i investitori mogli dovesti preciznu poslovnu odluku.

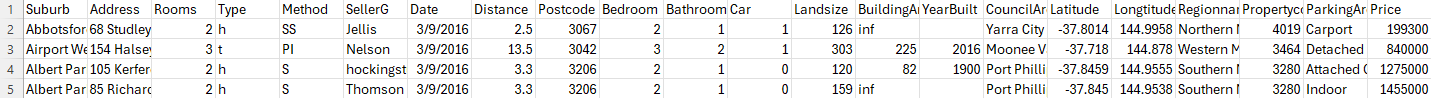
### 2.2. Analiza podataka

Skup podataka nam se sastoji o 22 atributa i ukupno 34,857 redaka gdje svaki individualni redak predstavlja zasebnu prodaju nekretnine. Skup podataka također sadrži prihvatljivu količinu kvalitativnih i kvantitativnih podataka među kojima nalazimo i veoma važnu vremensku dimenziju.

Atributi su sljedeći: Suburb, Address, Rooms, Type, Method, SellerG, Date, Distance, Postcode, Bedroom, Bathroom, Car, Landsize, BuildingArea, YearBuilt, CouncilArea, Latitude, Longitude, Regionname, Propertycount, ParkingArea, Price.

Po listi atributa primjećujemo nekoliko kategorija / vrsta atributa koje možemo podijeliti na sljedeći način:

* **Geografski atributi:** Suburb, Postcode, CouncilArea, Regionname, Latitude, Longitude predstavljaju neku lokaciju te opisuju razne aspekte lokacije u kojoj se nekretnina nalazi među kojima nalazimo regiju, kvart, poštanski broj i najvažnije koordinate koje nam daju preciznu informaciju o lokaciji.
* **Podaci o nekretnini:** Address, Landsize, buildingArea, YearBuilt, Propertycount, ParkingArea nam opisuju razne aspekte nekretnine koje bi potencijalni kupci uzimali u obzir za moguće modifikacije i buduće planove nadogradnje u reguliranom urbanom okolišu.
* **Podaci o kući:** Rooms, Type, Bedroom, Bathroom, Car nam opisuju detalje koji potencijalnim kupcima početno padaju na um te nam govore o dostupnim komoditetima kuće.
* **Podaci o prodavačima:** Skup podataka sadrži samo jedan atribut koji pripada prodavaču SellerG te predstavlja ime prodavača.
* **Podaci o prodaji:** Date, Method i Price opisuju informacije o prodaji nekretnine.



Slika 1: Prikaz atributa u CSV datoteci

Nakon analize svakog atributa možemo identificirati potrebne dimenzije za postizanje našeg cilja. Atributi koje ćemo koristiti primarno za analizu tržišta nekretnina glase:

* Latitude, Longitude, RegionName će nam dati prosječnu i preciznu informaciju o lokaciji te će nam pokazati moguće trendove vezane za lokaciju nekretnine.
* Karakteristike nekretnine i kuće poput Bedroom, Bathroom, Car i Landsize nam mogu ukazati trendove oko ponuđenih komoditeta koje potencijalni kupci traže te YearBuilt i Type nam ukazuje preference na vrstu kuće koja se cilja u prodajama
* Method i SellerG nam ukazuje trendove u načinu prodaje te nam pokazuje preferencije na tržištu.
* Date nam dodaje vremenski faktor gdje možemo ugledati promjenu trendova i performansi kroz vrijeme
* Price je najvažniji atribut u skupu s time da nam povezuje atribute i omogućuje spajanje atributa u svrhu usporedbe trendova

Iako neke atribute nećemo koristiti zbog nedostatka ili nezadovoljavajuće kvalitete podataka, skup nam pruža više nego dovoljno opširan skup podataka za izvođenje kvalitetne analize. Osim potencijalno malene količine iskoristivih redaka zbog nedostajećih podataka, naš skup zadovoljava ostale aspekte potrebne za izvršavanje projekta.

### 3. Izrada relacijskog modela i baze podataka

Kako bi stvorili bazu podataka i relacijski model za pripremu analize tržišta nekretnina, prvo ćemo primjeniti alate dostupne u programskom jeziku Python kako bi analizirali stanje integriteta podataka u našoj CSV datoteci. Sa Python programskim jezikom imamo dostupnu biblioteku *pandas*, koja nam omogućava rad na našim podacima putem manipulacije pandas specifičnog objekta *DataFrame* u koji učitavamo naše podatke te putem kojeg možemo jednostavno izvesti analizu, čišćenje i modifikaciju podataka u našoj CSV datoteci.

Nakon analize skupa koristeći Python alate, prelazimo na modeliranje i implementaciju baze podataka koristeći *sqlalchemy* i *MySQL Workbench*. sqlalchemy je biblioteka dostupna u Pythonu koja omogućava deklarativno mapiranje modela, tj. definiranje modela baze podataka koristeći Python klase, omogućava upravljanje sesijama za interakciju s bazom podataka gdje možemo implementirati promjene i upravljati relacijama između modela. U ovoj situaciji ćemo primjeniti sqlalchemy kako bi implementirali naš relacijski model u MySQL bazu podataka učitavajući podatke iz CSV-a direktno u MySQL bazu.

### 3.1. Analiza i čišćenje skupa putem implementacije u Pythonu

Implementacijom sljedeće Python skripte smo analizirali, očistili te raspodijelili skup podataka na dva dijela kako bi simulirali realnu situaciju gdje nam podaci dolaze sa više izvora.

Korišteni alati:

* Poveznica na službenu stranicu Python dokumentacije: [docs.python.org/3/](https://docs.python.org/3/)
* Poveznica na službenu stranicu pandas dokumentacije: [pandas.pydata.org/docs/](https://pandas.pydata.org/docs/)

1. import pandas as pd

2.

3. CSV\_FILE\_PATH = "Melbourne\_housing.csv"

4. df = pd.read\_csv(CSV\_FILE\_PATH, delimiter=',')

5.

6. print("Veličina skupa (broj redaka, broj stupaca):", df.shape)

7.

8. print("Nazivi stupaca:", df.columns.tolist())

9.

10. for column in df.columns:

11.     print(f"Jedinstvene vrijednosti u stupcu {column}: {df[column].nunique()}")

12.

13. print("Tipovi podataka po stupcu:\n", df.dtypes)

14.

15. print("Broj nedostajućih vrijednosti po stupcu:\n", df.isna().sum())

16.

17. print("CSV size before: ", df.shape)

18.

19. df = df.dropna()

20. print("CSV size after: ", df.shape)

21. print(df.head())

22.

23. df20 = df.sample(frac=0.2, random\_state=1)

24. df = df.drop(df20.index)

25. print("CSV size 80: ", df.shape)

26. print("CSV size 20: ", df20.shape)

27.

28. df.to\_csv("Melbourne\_housing\_PROCESSED.csv", index=False)

29. df20.to\_csv("Melbourne\_housing\_PROCESSED\_20.csv", index=False)

U skripti pozivamo biblioteku pandas te učitavamo CSV datoteku sa lokalnog foldera (skripta se mora nalaziti u istom folderu da bi korištena ruta radila) u pandas DataFrame objekt.

Putem *.shape* metode provjeravamo veličinu učitanog DataFrame-a



Slika 2: Ispis veličine DF-a u konzoli

Koristeći *.tolist()* metodu na kolonama DF-a smo ispisali sve učitane atribute



Slika 3: Ispis atributa u konzoli

Iterirajući kroz kolone DF-a ispisujemo količinu jedinstvenih vrijednosti u skupu podataka. Koristeći *.nunique()* dobijamo količinu jedinstvenih vrijednosti na osi.

A screenshot of a computer

Description automatically generated

Slika 4: Ispis količine jedinstvenih vrijednosti po atributu

Koristeći *.dtypes* metodu provjeravamo vrstu podataka koje nalazimo među atributima

A screenshot of a computer

Description automatically generated

Slika 5: Prikaz vrsta podataka u DF-u

Date je zapisan kao object umjesto podržane jedinice vremena u pandas DF-u no to se lagano ispravi prije ubacivanja u bazu podataka.

U zadnjem koraku analize provjeravamo količinu nedostajućih vrijednosti po stupcu kostiteći *.isna()* metodu te sumirajući vrijednosti po atributima.

A screenshot of a computer

Description automatically generated

Slika 6: Prikaz količine nedostajećih podataka po atributu

U sljedećem koraku provjeravamo početnu veličinu DF-a te brišemo sve retke sa null vrijednostima. Tijekom provjere preostalih podataka uočavamo da nam je ostalo oko 9000 preciznih podataka o prodajama. Iako smo izgubili veliki dio podataka, čisti i ispunjeni skup podataka će nam pružati preciznije rezultate tijekom analize.

Za kraj nasumično dijelimo skup podataka na 80% I 20% te ih spremamo u zasebne CSV datoteke koje se nalaze na lokalnoj ruti.

### 3.2. Izrada konceptualnog modela

A diagram of a server

Description automatically generated

Slika 7: Konceptualni model

Entiteti konceptualnog modela:

* **SALE:** prodaja je glavni entitet te sadrži atribute id, DateOfSale(datum prodaje), Method(metoda prodaje) i Price(cijena). Opisuje vrijeme, cijenu i metodu prodaje nekretnine. Metoda prodaje može biti S(prodano na dražbi), SP(prodano prije dražbe), PI(prodano van dražbe), VB(prodano agentu u pokušaju da digne cijenu), SA(prodano poslije dražbe)
* **SELLER:** prodavač je drugi entitet. Sadrži atribute id, SellerName(ime prodajnog agenta) te se povezuje na prodavača pošto opisuje koji prodavač je povezan sa kojom prodajom. Jedan prodavač može biti povezan na više prodaja.
* **PROPERTY:** entitet koji opsuje osnovne informacije o nekretnini te se povezuje na informacije o prodaji. Sastoji se od id, Address(adresa), LandSize(veličina zemljišta), BuildingArea(dopuštena veličina gradnje), YearBuilt(godina gradnje), PropertyCount(broj nekretnina uključenih u zoni) i ParkingArea(vrsta dostupnog parkinga). Parking area ima 8 mogućih stanja te može biti Attached Garage(garaža spojena na kuću), Detached Garage(garaža odvojena od kuće), Carport(otvoreni parking sa zaštitom), Indoor(zatvoreni javni parking), Parkade(javni parking na više katova), Underground(podzemni parking), Outdoor Stall(javni otvoreni parking), Parking Pad(privatno mjesto pred kućom).
* **HOUSE DETAILS:** entitet koji se veže na nekretninu, opisuje vrstu i komoditete dostupne kući. Sastoji se od atributa id, Rooms(broj prostorija), Type(vrsta kuće), Bedroom(broj spavaćih soba), Bathroom(broj kupaona), Car(kapacitet vozila). Type ima 3 moguća stanja te može biti h(Individualna kuća), t(gradska kuća), u (stan)
* **LOCATION:** entitet koji opisuje informacije o preciznoj lokaciji nekretnine, nadovezuje se na nekretninu. Sastoji se od Suburb(naselje), Postcode(poštanski broj), CouncilArea(opčinsko područje), RegionName(regija), Latitude(zemljopisna širina), Longitude(zemljopisna dužina). Koordinate su provjerene te se mogu koristiti u prikazu heatmape koja ukazuje na trendove.

### 3.3. EER Diagram

A screenshot of a computer

Description automatically generated

Slika 8: Extended Entity-Relationship Diagram

### 3.4. Popunjavanje relacijskog modela

Popunjavanje relacijskog modela je obavljeno koristeći skriptu napravljenu u programskom jeziku Python implementirajući biblioteku *sqlalchemy* kako bi uspostavili vezu sa MySQL bazom podataka.

1. import pandas as pd

2. from sqlalchemy import create\_engine, Column, Integer, String, Float, ForeignKey, Date, UniqueConstraint

3. from sqlalchemy.orm import sessionmaker, relationship, declarative\_base

4. from datetime import datetime

5.

6. CSV\_FILE\_PATH = "Melbourne\_housing\_PROCESSED.csv"

7. df = pd.read\_csv(CSV\_FILE\_PATH, delimiter=',')

8.

9. Base = declarative\_base()

10.

11. class Location(Base):

12.     \_\_tablename\_\_ = 'location'

13.     id = Column(Integer, primary\_key=True)

14.     suburb = Column(String(100), nullable=False)

15.     postcode = Column(Integer, nullable=False)

16.     CouncilArea = Column(String(100), nullable=False)

17.     regionname = Column(String(100), nullable=False)

18.     latitude = Column(Float, nullable=False)

19.     longitude = Column(Float, nullable=False)

20.

21. class Property(Base):

22.     \_\_tablename\_\_ = 'property'

23.     id = Column(Integer, primary\_key=True)

24.     location\_id = Column(Integer, ForeignKey('location.id'))

25.     address = Column(String(90), nullable=False)

26.     landsize = Column(Integer)

27.     buildingArea = Column(String(50))

28.     yearBuilt = Column(String(50))

29.     propertycount = Column(Integer)

30.     parking\_area = Column(String(50))

31.

32. class HouseDetails(Base):

33.     \_\_tablename\_\_ = 'house\_details'

34.     id = Column(Integer, primary\_key=True)

35.     property\_id = Column(Integer, ForeignKey('property.id'))

36.     rooms = Column(Integer)

37.     type = Column(String(1))

38.     bedroom = Column(Float)

39.     bathroom = Column(Float)

40.     car = Column(Float)

41.

42. class Seller(Base):

43.     \_\_tablename\_\_ = 'seller'

44.     id = Column(Integer, primary\_key=True)

45.     seller\_name = Column(String(100))

46.

47. class Sale(Base):

48.     \_\_tablename\_\_ = 'sale'

49.     id = Column(Integer, primary\_key=True)

50.     property\_id = Column(Integer, ForeignKey('property.id'))

51.     seller\_id = Column(Integer, ForeignKey('seller.id'))

52.     dateOfSale = Column(Date, nullable=False)

53.     method = Column(String(2))

54.     price = Column(Integer)

55.

56. engine = create\_engine('mysql+pymysql://root:root@localhost:3306/melhouse', echo=False)

57.

58. Base.metadata.create\_all(engine)

59.

60. Session = sessionmaker(bind=engine)

61. session = Session()

62.

63. def convert\_date(date\_str):

64.     return datetime.strptime(date\_str, '%d/%m/%Y')

65.

66. for index, row in df.iterrows():

67.

68.     location = Location(suburb=row['Suburb'],

69.                         postcode=row['Postcode'],

70.                         CouncilArea=row['CouncilArea'],

71.                         regionname=row['Regionname'],

72.                         latitude=row['Latitude'],

73.                         longitude=row['Longitude'])

74.     session.add(location)

75.     session.flush()

76.

77.     property = Property(address=row['Address'],

78.                         landsize=row['Landsize'],

79.                         buildingArea=row['BuildingArea'],

80.                         yearBuilt=row['YearBuilt'],

81.                         propertycount=row['Propertycount'],

82.                         parking\_area=row['ParkingArea'])

83.     property.location\_id = location.id

84.     session.add(property)

85.     session.flush()

86.

87.     house\_details = HouseDetails(rooms=row['Rooms'],

88.                                  type=row['Type'],

89.                                  bedroom=row['Bedroom'],

90.                                  bathroom=row['Bathroom'],

91.                                  car=row['Car'])

92.     house\_details.property\_id = property.id

93.     session.add(house\_details)

94.     session.flush()

95.

96.     seller = Seller(seller\_name=row['SellerG'])

97.     session.add(seller)

98.     session.flush()

99.

100.     sale = Sale(dateOfSale=convert\_date(row['Date']),

101.                 method=row['Method'],

102.                 price=row['Price']

103.     )

104.     sale.property\_id = property.id

105.     sale.seller\_id = seller.id

106.     session.add(sale)

107.     session.flush()

108.

109. session.commit()

110. session.close()

111. print("Tablice su popunjene.")

Program učitacva podatke iz CSV datoteke pozvanoj na definiranoj ruti i pohranjuje ih u bazu podataka. Koristeći sqlalchemy definirane su tablice location, property, house\_details, seller i sale. Podaci se iteriraju redak po redak, gdje se za svaki redak popunjavaju objekti tablica i spremaju u bazu podataka. Na kraju se svi podaci trajno pohranjuju i veza sa bazom se zatvara.

### 4. Izrada dimenzijskog modela

Izrada dimenzijskog modela služi za organizaciju podataka radi poboljšanja performance I jednostavnosti analize. Svrha dimenzijskog modela je olakštati i omogućiti učinkovitu pohranu, pristup, modifikaciju i analizu velikih količina podataka, koji često dolaze iz različitih izvora. Dimenzijski model se sastoji od jedne činjenice i pet različitih dimenzija koje ćemo primjeniti kako bi mjerljivom podatku predstavili kontekst, tj.dimenziju.

Dimenzijski model je također optimiziran za brzu obradu podataka zbog implementirane strukture zvijezde koja omogućuje brzo pridruživanje tablica i smanjuje potrebno vrijeme za obrađivanje zahtjeva, što nam dalje olakšava složene analitičke upite, među kojima nalazimo agregaciju, filtriranje i grupiranje podataka.

Među alternativama nalazimo i strukturu pahuljice, no za veličinu našeg skupa podataka, nije potrebna te granana sturktura modela bi patila na performansi operacija na podacima.

Sam model implementiran u projektu se sastoji od tablice činjenica *sale* i pet dimenzijskih tablica koje glase: *sale\_info*, *seller*, *property*, *location* i *house\_details*.

Tablica činjenica sastoji se od jedne mjere – cijena prodaje nekretnine. Cijena prodaje služi kao osnovni indikator informacije o prodaji nekretnine te se na nju nadovezuju ostale ključne dimenzije kroz strane ključeve za prodavača, prodaju, nekretninu, kuću i lokaciju.

Dimenzijska tablica *location* sadrži hijerarhiju podataka o lokaciji, od samog naselja (‘suburb’), poštanskog broja (‘postcode’), općinskog područja (‘CouncilArea’), do regije (‘regionname’) u kojoj se nalazi nekretnina. Također nalazimo I geografske coordinate (‘latitude’ i ‘longitude’) koje omogućuju prostornu analizu podataka.Tablica sadrži surogat ključ (‘id’) koji služi kao primarni ključ tablice.

Dimenzijska tablica *sale\_info* sadrži podatke o datumu prodaje (‘dateOfSale’) I metodi prodaje (‘method’) koji omogućavaju analizu podataka po danu, mjesecu i godini. Vremenska dimenzija nije sporo mjenjajuća te se ne mijenja nakon unosa. Također sadrži surogat kljuć (‘id’) koji služi kao primarni ključ tablice.

Dimenzijska tablica *property* sadrži podatke o godini izgradnje (‘yearBuilt’), veličine nekretnine (‘landsize’) i površine dopuštene gradnje (‘buildingArea’). Dimenzija omogućava grupiranje prema godini izgradnje i analizu utjecaja veličine parcele na činjenicu. Također sadrži surogat kljuć (‘id’) koji služi kao primarni ključ tablice.

Dimenzijska tablica *house\_details* sadrži detalje o kući te u njoj nalazimo broj prostorija (‘rooms’), vrsta kuće (‘type’), broj spavaćih soba (‘bedroom’), broj kupaonica (‘bathroom’) I kapacitet parkirnog mjesta (‘car’). Omogućava filtriranje i analizu po komoditetima dostupnim sa nekretninom. Također sadrži surogat kljuć (‘id’) koji služi kao primarni ključ tablice.

Dimenzijska tablica *seller* sadrži jedan podatak – ime prodajnog agenta. Omogućava nam da pratimo performansu i trendove među uspješnim agentima. Također sadrži surogat kljuć (‘id’) koji služi kao primarni ključ tablice.

Pružajući ukupni pogled na dimenzijski model može se uočiti da su sve dimenzije osim vremenske sporo mijenjajuće te se njihove promjene mogu pratiti kroz atribute (‘version’, ‘date\_from\_date’ i ‘date\_to\_date’). Dostupna je odlična osnova za analizu podataka o prodaji nekretnina prateći utjecaje dimenzija na cijenu prodaje što će dovesti do bolje informiranih poslovnih odluka i otvoriti put za bolje strateško planiranje.

### 4.1. Diagram dimenzijskog modela

A diagram of a computer

Description automatically generated

Slika 9: Dimenzijski model

### 5.1. ETL Proces

ETL (*extract, transform, load*) je process koji se koristi za prikupljanje podataka iz potencijalno različitih izvora, transformaciju i pripremu te spremanje u ciljano skladište podataka. Sastoji se od tri glavna koraka:

1. **Extract** (Izvlačenje): najčešće se podaci prikupljaju sa više izvora poput relacijskih baza, CSV datoteka, API-eva te ostale metode skladištenja podataka. Cilj je prikupiti što više relevantnih podataka za obradu.
2. **Transform** (Transformacija): sljedeći korak ukljućuje pripremu podataka za skladištenje podataka je priprema, tj. transformacija istih u format koji će omogučiti učitavanje istih bez problema u ciljni sustav. Transformacija može uključivati čišćenje, agregaciju, normalizaciju, filtriranje i sortiranje podataka.
3. **Load** (Učitavanje): posljednji korak ETL-a je učitavanje podataka u namjenjeno skladište podataka. Cilj je osigurati da su svi relevantni podaci dostupni za rad.

Za trenutačni korak korišten je *Pentaho Data Integration tool* koji pomoću jednostavnog sučelja omogućava izradu ETL procesa za svaku dimenzijsku tablicu.

Svaka tablica će se sastojati o procesa spajanja podataka iz dva različita izvora gdje 80% posto podataka će doći od relacijskog modela spremljenog u relacijskoj bazi podatakaa a preostalih 20% dolaze iz CSV datoteke. Nakon učitavanja preoblikujemo strukturu podataka za stapajanje i spremanje u dimenzijski model koji se također skladišti u MySQL bazu podataka.