TUP seminarske naloge 2024-2025

Oblikujte skupine po dva (2) študenta in prijavite temo na učilnici. Tema je lahko bodisi ena od posebnih tem, ali pa kombinacija oz. par orodja s podatki (npr. nocodb orodje in »filmi« podatki).

Pomembno je da si podatke pravilno prenesete, po potrebi pretvorite iz dokumentne v relacijsko obliko, po potrebi dodate smiselne indekse in podatke analizirate pri čemer želimo najti kakšne nove zanimivosti ali dejstva. Kaj lahko poveste tudi o hitrosti in možnih izboljšavah.

Ko si najdete kolega/e (kolegico/e) za skupno delo na seminarski in določite baze/orodja se najprej posvetujte s profesorjem ali asistentom, če je vaš plan primeren, saj je izbirnost zelo velika.

Posebne teme (po dogovoru)

1. Vektorske podatkovne baze, veliki jezikovni modeli in RAG iskanje

(tema je oddana)

2. Zbiranje, povezovanje, vizualizacija in analiza podatkov o volitvah v ZDA (2024)

(tema je oddana)

3. PRQL - Cevovodni SQL

PRQL (Pipelined Relational Query Language) [1,2,3] je sodoben jezik za transformacijo podatkov, zasnovan kot enostavna, zmogljiva zamenjava za SQL. PRQL je lahko berljiv, ekspliciten in deklarativen, podobno kot SQL, vendar se razlikuje po tem, da tvori logično cevovodno strukturo transformacij. Podpira abstrakcije, kot so spremenljivke in funkcije, ter omogoča boljšo obravnavo oznak NULL in podatkovnih tipov. Ena od glavnih prednosti PRQL je njegova zmožnost cevovodnega sestavljanja in razširjanja poizvedb na bolj intuitiven način, kar omogoča lažje vzdrževanje in razumevanje kompleksnih poizvedb. Vaša naloga je, da preučite PRQL, ga jedrnato opišete, primerjate z SQL in ga ovrednotite na realnih podatkih v vsaj enem od kompatibilnih sistemov (npr. DuckDB [4], PostgreSQL [5],...).

1. PRQL, https://prql-lang.org/
2. PRQL Language Book, https://prql-lang.org/book/
3. https://github.com/PRQL
4. prql – a DuckDB Community Extension, https://community-extensions.duckdb.org/extensions/prql.html
5. PRQL in PostgreSQL, https://github.com/kaspermarstal/plprql

4.Primerjava odprtokodnih vektorskih SUPB

V zadnjih letih se je uporaba vektorskih podatkovnih baz močno povečala, nekdaj nišno orodje v obliki dodatkov prodira tudi v splošnonamenske relacijske SUPB. Preučite vsaj tri (npr. Croma [2], Milvus [3], Qdrant [1]) vektorske SUPB in vsaj en dodatek relacijskim SUPB (npr. PostgreSQL z dodatkom pgvector [4], ali MariaDB Vector [5]) ter definirajte scenarije uporabe in kriterije, po katerih jih lahko primerjate in ovrednotite. Navedene primere z danimi kriteriji opišite in primerjajte, ter naredite tudi praktično primerjavo vsaj enega vektorskega SUPB in vsaj ene vektorske razširitve relacijske SUPB.

1. Qdrant, https://qdrant.tech/documentation/

2. Croma, https://docs.trychroma.com/api
3. MIlvus, https://milvus.io/docs/install_standalone-docker.md
4. pgvector, https://github.com/pgvector/pgvector
5. MariaDB Vector, https://mariadb.org/projects/mariadb-vector/

5. Pregled in primerjava alternativnih orodij za konceptualno načrtovanje

Preglejte in po ključnih funkcionalnostih opisno primerjajte čimveč odprtokodnih in komercialnih orodij (poskusne različice) in jih primerjajte z orodjem PowerDesigner na primernem problemu in z različnimi strategijami (od spodaj navzgor-reverzni inženiring, po delih).

Nekaj izhodiščnih produktov:

ER/Studio, https://erstudio.com/free-trial/

Erwin data modeler, https://www.erwin.com/software-demos-and-trials/

EdrawMax, https://www.edrawsoft.com/solutions/edrawmax-for-education.html

Archi, https://www.archimatetool.com/

pgModeller, https://pgmodeler.io/

dbDesigner, https://www.dbdesigner.net

Gliffy, https://www.gliffy.com/free-trial

Creately, https://creately.com/

LucidChart, https://www.lucidchart.com/pages

6. Normalizacija in uporaba podatkovnih tabel »Agromet«.

V nalogi imamo opravka s surovimi podatki opazovanja škodljivih organizmov, fenoloških faz rastlin in meteoroloških podatkov lokacij po Sloveniji. Podatki so predstavljeni v surovi obliki (podobno, kot da bi jih pridobivali na javnem servisu oz. v našem primeru, kot izvoz iz izključno podatkovnih tabel - to so tabele brez šifrantov, ki vsebujejo dejanske podatke o opazovanju).

Tukaj so lahko trije študenti v skupini.

Vaša naloga je, da podatke iz datotek normalizirate, pripravite podatkovne strukture na izbranem SUPB in te podatke uvozite. **Pogledi se nahajajo na *pb.fri.uni-lj.si* pod shemo *agromet/Views***.

Pogled **Meteorologija** predstavlja zapise na vsake pol ure z meteorološkimi parametri po lokacijah, ta tabela je že normalizirana.

Pogled **Fenologija** predstavlja fenološka opažanja za lokacije.

Pogled **Skodljivci** predstavlja opažanja škodljivega organizma za lokacije.

Funkcionalne odvisnosti med stolpci so sledeče:

Fenologija: FazaBBCH -> FazaBBCHSIfra, FazaBBCH3Sifra, FazaBBCHTaxGrp, FazaBBCHTaxGrpName, FazaBBCHTaxGrpDesc;

FazaBBCHTaxGrp -> FazaBBCHTaxGrpName, FazaBBCHTaxGrpDesc;

FenoPostajaID -> Vse atribute, ki se začnejo z FenoPostaja;

FenoPostajaSpremljanOrganizemId -> FenoPostajaSpremljanOrganizemNaziv, FenoPostajaSpremljanOrganizemNazivLat;

FenoPostajaTipKultID -> FenoPostajaTipKultName

FenoLokacijaID -> Vse atribute ki se začnejo z Feno (stolpci gledano od Stolpca FenoPostajaMeteoId naprej).

Pomemben je tudi stolpec FenoPostajaMeteoId, ki določa Meteorološko postajo kateri ta postaja Feno pripada. V podatkih atributov, ki so določeni s strani tega ključa ni, saj bi se v večini primerov podvajali, se pa nanaša na FenoLokacijaID (kažeta na isto strukturo, le na različne zapise v tej strukturi.

Škodljivi Organizem (ŠO): StadijID -> StadijTXT; MestoSpr -> MestoSpremljanjaName, MestoSpremljanjaOpis; ObserverID -> ObserverTxt, PostajaSOSpremljanjeSO -> PostajaSOOperational, PostajaSOSpremljanOrganizemOrganizemId, PostajaSOSpremljanOrganizemOrganizemNaziv, PostajaSOSpremljanjePovrsina, PostajaSOAktivnaOd, PostajaSOAktivnaDo, PostajaSOOpis, PostajaSOInsektarijID, PostajaSOInsektarijTXT, PostajaSOMeteoLokacijaID;

PostajaSOSpremljanOrganizemOrganizemId -> PostajaSOSpremljanOrganizemOrganizemNaziv,

SOLokacijaID -> Vse atribute, ki se začnejo s SO

SOMicroLocID -> SOMicroLocName; SOMacroLocID -> SOMacroLocName; SOUrbanID -> SOUrbanName

Pomemben je tudi stolpec PostajaSOMeteoLokacijaID, ki določa Meteorološko postajo kateri ta postaja ŠO pripada. V podatkih atributov, ki so določeni s strani tega ključa ni, saj bi se v večini primerov podvajali, se pa nanaša na SOLokacijaID (kažeta na isto strukturo, le na različne zapise v tej strukturi.

Pripravite nekaj smiselnih netrivialnih poizvedb nad dobljeno strukturo podatkov, kjer v vsaj treh poizvedbah vključite Meteo podatke. Uporabite tudi grupiranje in gnezdenje. Opišite ali je za vašo poizvedbo potrebno kreirati indeks?

Za vsako poizvedbo opišite, katere indekse ste morali kreirati (ali ste ta indeks kreirali ob kreiranju sheme, ali ste ga definirali kasneje). Koliko časa traja izvajanje te procedure? Ali ste za potrebe poizvedbe kreirali začasne tabele oz. denormalizirane tabele, ki vsebujejo vnaprej izračunane podatke?

Dober test pravilnosti uvoza je poizvedba: Izpiši vsa fenološka opažanja in vse ŠO v zadnjem letu pri lokaciji »%Bionomija%«. Pri tej poizvedbi morate dobiti tako ŠO kot tudi Fenološke zapise.

Razmislite na katerih stolpcih in tabelah boste potrebovali indekse. V praksi se veliko poizvedb implementira v procedurah, saj so le te preveč kompleksne (semantično ali pa morebiti le težko berljive), da bi bile implementirane kot klic enega SQL SELECT stavka. Ko uvedemo še parametre pri klicu, ali to vpliva na hitrost poizvedb? Ali indeksi igrajo kakšno vlogo?

Podatki

1. Podatkovna zbirka MongoDB sample_mflix

Zbirka sample_mflix vsebuje podatke o filmih in kinodvoranah, ter določene metapodatke, vključno z uporabniki in komentarji filmov. Do podatkov najlažje dostopate tako, da jih naložite v MongoDB in potem prepišete v relacijsko bazo (če je to potrebno oz. če jo uporabljate).

Ker ima baza mflix/movies že lepo urejeno strukturo (dokumenti imajo enake oznake) lahko napišete Python program, ki prepiše bazo movies v relacijsko bazo, pri čemer vse oznake tipa seznam (Array) pretvori v ustrezen šifrant tako, da se baza že delno normalizira. Razmislite še, če je smiselno enako normalizirati tudi oznake tipa Object.

- Opis: https://www.mongodb.com/docs/atlas/sample-data/sample-mflix/
- Podatki: https://github.com/neelabalan/mongodb-sample-dataset

Število teh filmov ni dovolj veliko, zato vključimo še druge baze filmov:

1. MovieLens

- **Kratek opis**: MovieLens je ena izmed najbolj priljubljenih zbirk filmskih podatkov, ki vsebuje informacije o filmih, uporabniških ocenah, oznakah in kategorijah. Gre za zbirko, ki jo pogosto uporabljajo raziskovalci za analizo priporočilnih sistemov.
- **Kje dobiti podatke**: Podatke za MovieLens lahko prenesete z njihove uradne spletne strani: MovieLens Dataset.
- **Oblika podatkov**: Podatki so običajno v **CSV** (Comma-Separated Values) obliki.
- **Vrsta baze**: Relacijska. Podatki so shranjeni v več tabelah (npr. filmi, uporabniške ocene, oznake).

2. IMDb (Internet Movie Database)

- **Kratek opis**: IMDb je ena največjih spletnih zbirk podatkov o filmih, TV serijah, igralcih, režiserjih, ocenah, kritikah itd. IMDb ponuja obsežen nabor informacij o filmih.
- **Kje dobiti podatke**: Podatke iz IMDb-ja lahko pridobite preko njihovega IMDb API. Za večje in obsežnejše podatke lahko uporabite tudi IMDb Datasets.
- **Oblika podatkov**: Podatki so na voljo v **CSV** ali **TSV** (Tab-Separated Values) obliki.
- **Vrsta baze**: Relacijska. IMDb uporablja različne tabele za filme, igralce, ocene, žanre itd.

3. OMDb (Open Movie Database)

- **Kratek opis**: OMDb je brezplačna API storitev, ki ponuja podatke o filmih, TV serijah, igralcih, režiserjih, ocenah in mnogih drugih podrobnostih.
- **Kje dobiti podatke**: Podatke iz OMDb lahko pridobite preko njihovega OMDb API.
- **Oblika podatkov**: Podatki so na voljo v **JSON** formatu.
- **Vrsta baze**: Dokumentna. Podatki so v JSON formatu, primeren za uporabo v aplikacijah in sistemih.

4. Kaggle Movie Datasets

- **Kratek opis**: Kaggle ponuja različne zbirke podatkov, vključno z obsežnimi filmskimi podatki, kot so ocene, žanri, proračuni, zaslužki in drugo.
- **Kje dobiti podatke**: Zbirke podatkov so na voljo na Kaggle Movies Dataset.
- **Oblika podatkov**: Podatki so običajno v **CSV** formatu.
- **Vrsta baze**: Relacijska. Zbirke podatkov so običajno v obliki tabel, ki vsebujejo različne vrste podatkov (npr. filmi, ocene, zaslužki).

Povzetek:

- **MovieLens**: Relacijska baza v CSV formatu, ki vsebuje podatke o filmih, ocenah in uporabnikih.
- **IMDb**: Relacijska baza v CSV/TSV formatu, vsebuje podatke o filmih, igralcih, režiserjih in ocenah.
- **OMDb**: Dokumentna baza v JSON formatu, vsebuje podatke o filmih in TV serijah.
- **Kaggle Movie Datasets**: Relacijska baza v CSV formatu z obsežnimi filmskimi podatki.

Filme iz vsaj dveh različnih baz poskusite prenesti v relacijsko bazo in jih uparite po naslovu. Poskusite če se ujemajo tudi po kakšnem drugem ključu (leto, igralci ...).

2. Normalizacija in uporaba podatkovne tabele »ProdajaVin«
(lahko več skupin uporabi te podatke, vendar z različnimi izbranimi orodji)

Gre za podatke o dobavnicah podjetja za veleprodajo vina za leto 2022. Tabela ni normalizirana. Najprej prenesete podatke na svoj računalnik v MySQL bazo. Tabela »ProdajaVin« se nahaja na pb.fri.uni-lj.si v shemi Seminar/Tables/ProdajaVin. Prenos lahko izvedete na več načinov (izvoz v datoteko, preko ODBC-ja ali PyODBC-ja). Izberite svojega in ga v nalogi opišite.

Podatki za normalizacijo:
ProdajaVin(stevilka, partnerL, DavcniZavezanec, SkZnesek, SkDavek, DatumDokumenta, DatumValuta, RabatProcentGlava, counter, nivoL, ArtikelStoritev, tip, OpisDavOsn, SifraArtikla, OpisArtikla, BlagovnaSkupinaSubL, Naziv, DavcnaSkupinaL, OpisDavcneOsnove, Procent, MerskaEnotaL, Enota, OpisMerskeEnote, Kolicina, ProdajnaCena, RabatTelo, Kupec, PostnaStevilka, Posta, Oznaka, Drzava)

Funkcionalne odvisnosti:
stevilka->PartnerL, DavcniZavezanec, SkZnesek, SkDavek, DatumDokumenta, DatumValuta, RabatProcentGlava, NivoL
NivoL->stevilka
counter->NivoL, ArtikelStoritev, BlagovnaSkupinaSubL, DavcnaSkupinaL, MerskaEnotaL, Kolicina, ProdajnaCena, RabatTelo, SifraArtikla, tip
SifraArtikla->OpisArtikla, ArtikelStoritev
DavcnaSkupinaL->OpisDavcneStopnje, Procent
tip->OpisDavOsn
MerskaEnotaL->Enota, OpisMerskeEnote
PartnerL->Kupec, DavcniZavezanec, PostnaStevilka, Oznaka
Oznaka->Drzava

PostnaStevilka->Posta
BlagovnaSkupinaSubL->Naziv

še način izračuna vrednosti:
SkZnesek = vsota vrednosti postavk po formuli: Kolicina * ProdajnaCena * (1 – RabatProcentGlava / 100) * (1 – RabatTelo / 100)
SkDavek = vsota vrednosti postavk po formuli: Kolicina * ProdajnaCena * (1 – RabatProcentGlava / 100) * (1 – RabatTelo / 100) * if(tip="N", Procent / 100, 0)
Preverite, če izračuni veljajo za podatke v tabeli.

Kateri atributi pri funkcionalnih odvisnostih so na prvi pogled nepotrebni?
Napišite kateri in zakaj oziroma razložite zakaj morda niso odveč.

Podatke normalizirajte v 3 normalno obliko in dobljenim entitetam podajte smiselno ime glede na njihovo vsebino.

Tako normalizirane podatke uporabite za testiranje različnih orodij, navedenih v tem dokumentu, in realizirajte vsaj 10 netrivialnih poizvedb nad povezanimi tabelami. Naredite tudi OLAP kocko, kjer lahko izbirate med kupci, znamko vin, letnimi časi, vsoto prodaje za izbrane parametre ipd. Predlagajte tudi kako bi smiselno načrtovali nabavo posameznih vin za prihodnje leto. Upoštevajte obliko nihanja prodaje posameznega vina.

3. Podatki iz repozitorija "Relational dataset repository"

V tem repozitoriju se nahaja več kot 150 podatkovnih zbirk, do katerih lahko dostopate iz MariaDB/MySQL klienta (npr MySQL Workbench). Podatke **obvezno prenesite** na svoj ali FRI podatkovni strežnik (to lahko naredite iz Pythona s povezavama na dve bazi). Vsaka zbirka vsebuje tudi kratek opis podatkov, nekatere pa tudi povezavo na tekst ali članek z opisom in načini uporabe.

Primerne zbirke so tiste z večjim številom tabel (vsaj 10) in večjo količino podatkov (skupnim številom vrstic vsaj nekaj 100.000). Nekatere zbirke, npr. Accidents (prometne nesreče v Ljubljani 1995-2005) so pogojno primerne (če jih dopolnite z dodatnimi/novejšimi podatki).

Iskalnik podatkov:

- Statistični opisi zbirk: https://relational-data.org/statistics
- Iskalnik zbirk: https://relational-data.org/search
- Dostop do podatkov: https://arxiv.org/abs/1511.03086

3. Splošni odprti podatki

Poiščite primerno velike podatke, po možnosti kaj, kar vas zanima. Lahko si pomagate s smiselnim povezovanjem več virov, vendar pazite, da bo podatkov dovolj (skupno vsaj nekaj 100.000 vrstic, še bolje pa nekaj milijonov) in da bodo organizirani v več tabelah (vsaj 5)!

Nekaj primerov:

- World Bank Data
  https://data.worldbank.org/

- New York yellow taxi trip record data: https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page
- New York parking tickets (primer za leto 2023): https://data.cityofnewyork.us/City-Government/Parking-Violations-Issued-Fiscal-Year-2023/869v-vr48
- European Air Quality Index (EAQI) https://www.eea.europa.eu/data-and-maps/data/aqereporting-8
- Urban Atlas https://www.eea.europa.eu/data-and-maps/data/urban-atlas
- Seznami virov odprtih podatkov
    - https://github.com/awesomedata/awesome-public-datasets
    - https://www.bernardmarr.com/default.asp?contentID=960
    - https://registry.opendata.aws
    - ...
- OPSI – Odprti podatki Slovenije: https://podatki.gov.si/
- Yelp Open Dataset: Velika zbirka zapisov o podjetjih, uporabnikih, ocenah in pregledih. Relacijski podatki vključujejo več tabel, ki omogočajo analizo omrežij in tržnih trendov.
  Povezava: https://www.yelp.com/dataset
  Primer uporabe: Analiza trga, priporočilni sistemi.

Razmislite tudi o časovnih in prostorskih podatkih, ali (delno) strukturiranih podatkih (XML, JSON) in njihovi integraciji v relacijsko bazo.

Orodja

A. PostgreSQL in PostgresML: analitika odprtih podatkov znotraj SUPB (lahko več skupin, vendar z različnimi podatki in načini uporabe)

PostgresML [1] je orodje, ki naj bi nam omogočalo uporabo priljubljenih orodij za podatkovno rudarjenje (Scikit-Learn, TensorFlow, XGBoost, …) znotraj podatkovne baze PostgreSQL. Raziščite zmožnosti orodja, podporo korakom CRISP-DM, integracijo z izbranim SUPB in različne načine uporabe. Vse skupaj ilustrirajte z uporabo na eni od večjih podatkovnih zbirk, rezultate kritično ovrednotite in predstavite v poročilu (vsaj 10 strani).

Vaša analiza naj vsebuje naslednje točke:

- Vnos podatkov v bazo
- Potrebna predpriprava podatkov za PostgresML (znotraj baze)
- Procesiranje podatkov (vsaj en pristop)
    - Nenadzorovano učenje (gručenje, asociacije, vizualizacija, …)
    - Nadzorovano učenje: izgradnja in uporaba napovednih modelov
    - Priporočilni sistemi (gradnja in uporaba)
- Ovrednotite tudi pomnilniško zahtevnost in možnost uporabe GPU (opcijsko)

Litertura:

1. PostgresML: https://postgresml.org/
2. PostgreSQL: https://www.postgresql.org/
3. CRISP-DM:https://en.wikipedia.org/wiki/Cross-industry_standard_process_for_data_mining

B. Vgrajena podatkovna analitika odprtih podatkov z orodjem DuckDB (lahko več skupin, vendar z različnimi podatki)

DuckDB [1] je vgrajen (embedded) SUPB, soroden SQLite [2], ki nam omogoča uporabo OLAP analitike (večdimenzionalne vrtilne tabele, hierarhična agregacija, …) znotraj podatkovne baze. Raziščite zmožnosti orodja, podporo korakom CRISP-DM, ilustrirajte analitične ukaze in različne načine uporabe. Še posebej se osredotočite na možnost interaktivne analize v uporabniku prijaznem okolju, npr. Jupyter/Python/duckdb in Dbeaver/duckdb. Vse skupaj na koncu preizkusite na eni od večjih podatkovnih zbirk, rezultate kritično ovrednotite in predstavite v poročilu (vsaj 10 strani).

Vaša analiza naj vsebuje naslednje točke:

- Vnos podatkov v bazo
- Potrebna predpriprava podatkov za analitiko (znotraj baze)
- Analitično procesiranje podatkov (opis in uporaba razširitev jezika SQL)
- Ovrednotite tudi skalabilnost, hitrost in pomnilniško zahtevnost sistema

Litertura:

1. DuckDB: https://duckdb.org
2. SQLite: https://www.sqlite.org
3. Python in DuckDB: https://duckdb.org/docs/api/python/overview

4. Dbeaver in DuckDB: https://duckdb.org/docs/guides/sql_editors/dbeaver
5. OLAP Hierarchical Aggregation with SQL: https://medium.com/learning-sql/olap-hierarchical-aggregation-with-sql-6c45ebc206d7
6. CRISP-DM:https://en.wikipedia.org/wiki/Cross-industry_standard_process_for_data_mining

C. Odprti podatki z NocoDB: relacijska baza kot pametna preglednica (lahko več skupin, vendar z različnimi podatki)

NocoDB [1,2] je platforma, ki omogoča gradnjo aplikacij (skoraj) brez programiranja in razširja uporabo popularnih SUPB (PostgreSQL, Microsoft SQL Server, SQLite, MySQL/MariaDB) s spletnim uporabniškim vmesnikom na temelju preglednic (kot Excel). Omogoča preprosto konstrukcijo [3] vseh potrebnih gradnikov aplikacij in uporabniškega vmesnika (datotečne operacije, manipulacije s stolpci, filtriranje vrednosti, nadzor dostopa, …). Obenem omogoča tudi uporabo aplikacijskega programskega vmesnika (API) z različnimi programskimi jeziki, npr. s Pythonom [4].

Lokalno si namestite orodje (npr. Docker), raziščite njegove zmožnosti, podporo korakom CRISP-DM, povezljivost s programskimi jeziki (Python) ter ilustrirajte različne scenarije in načine uporabe. Vse skupaj na koncu preizkusite v obliki aplikacije na eni od večjih podatkovnih zbirk, rezultate kritično ovrednotite in predstavite v poročilu (vsaj 10 strani).

Vaša analiza naj vsebuje naslednje točke:

- Vnos podatkov in potrebna predpriprava podatkov
- Uporabniške in analitične funkcionalnosti preglednice
- Gradnja aplikacije
- Ocena uporabniške prijaznosti, enostavnosti razvoja, zmogljivosti in pomnilniške zahtevnosti

Litertura:

1. NocoDB: https://www.nocodb.com
2. NocoDB GitHub: https://github.com/nocodb/nocodb
3. How to get started with NocoDB:
    - https://www.madeleinesmith.uk/blog/get-started-nocodb
    - https://docs.nocodb.com/getting-started/self-hosted/installation
4. NocoDB Python client: https://pypi.org/project/nocodb

D. Odprti podatki s splošnimi ML/DM/AI ali DB/DW/DL analitičnimi orodji (lahko več skupin, vendar z različnimi podatki in orodji)

Izberite si kombinacijo SUPB, analitičnega orodja in podatkov. Raziščite zmožnosti izbranega orodja, podporo korakom CRISP-DM, integracijo z izbranim SUPB in različne načine uporabe. Vse skupaj ilustrirajte z uporabo na večji podatkovni zbirki (vsaj 2 milijona zapisov).

- Orodja:
    - ML/DM/AI:
        - MindsDB: https://mindsdb.com
        - Rapid Miner: https://rapidminer.com/
        - Knime: https://www.knime.com/
        - Orange: https://orangedatamining.com/

        - Weka: https://www.cs.waikato.ac.nz/ml/weka/
        - R/Rattle: https://rattle.togaware.com/
        - Python/Scikit-learn + GUI (če kdo kaj najde)
        - Python/Scikit-learn/Ipython widgets
        - Python/Scikit-learn iz Excela
        - DataMelt: https://datamelt.org/
        - H2O (open source): https://www.h2o.ai/products/h2o/
        - ELKI: https://elki-project.github.io/weka
        - Tableau: https://www.tableau.com/academic/students
        - ... lastna izbira (po dogovoru)
    - DB/DW/DL:
        - MinIO: https://min.io
        - Microsoft SQL Server Data Tools
        - CRISP-DM: https://en.wikipedia.org/wiki/Cross-industry_standard_process_for_data_mining

Vaša analiza naj vsebuje naslednje točke:

- Vnos podatkov v bazo
- Potrebna predpriprava podatkov za analitiko (znotraj baze)
- Analitično procesiranje podatkov (opis in uporaba razširitev jezika SQL)
- Ovrednotite tudi skalabilnost, hitrost in pomnilniško zahtevnost sistema