# Povezava za prijavo seminarja na učilnici: [TUP seminarske naloge 2023-2024](https://unilj-my.sharepoint.com/:x:/g/personal/lukas_fri1_uni-lj_si/EbKN1g8fhBRHlHwO2gm_vksBWBSB7NbtNv_OvChQIbMNBw?e=ji3j7N)

Formirajte skupine po dva (2) študenta in se prijavite na zgornjo povezavo.

# 1. 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?

# 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, OpisDavcneStopnje, 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. PostgreSQL in PostgresML: podatkovna analitika 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>

# 4. Vgrajena podatkovna analitika 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](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>

# 5. 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](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>

# 6. Odprti podatki z 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>

# 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 dva milijona vrstic)!

Nekaj primerov:

* 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>
* 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/>

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