ALGEBRA

Pristup podacima iz programskog koda

Uvodno predavanje

Opće informacije

- Materijali kolegija će sadržavati primjere u programskim jezicima **C#** i **Python**
  - Prateći isključivo studijski program, trebali biste biti dobro upoznati s C# jezikom iz prethodnih kolegija
  - Python je vrlo koristan i praktičan programski jezik opće namjene koji se prvenstveno koristi za skriptiranje, podatkovnu znanost (čak i za Web!)

Opće informacije

- (neformalni) preduvjeti kolegija:
  - Uvod u baze podataka !
  - Oblikovanje baza podataka !
  - Objektno orijentirano programiranje !
  - Razvoj Web aplikacija (minimalno)

Ukoliko su koncepti obrađeni u kolegijima označenim s ! nejasni ili davno zaboravljeni, svakako ih ponovite (također sam otvoren za opciju organiziranja dodatnog sata ponavljanja ukoliko to bude potrebno)

ALGEBRA

O nastavniku

- **Borna Skračić**, pred.
- Podatkovni znanstvenik i istraživač
  - Član istraživačke skupine @ Sveučilište Algebra – Digitalno zdravlje
  - Voditelj EEG Data Portal projekta
  - 3+ godina iskustva rada u industriji (backend development, DevOps, …)
- Područje istraživanja
  - Primijenjeno strojno učenje i bioinformatika
  - Optimizacijske metode

ALGEBRA

INFORMACIJE O PROJEKTU

ALGEBRA

Informacije o projektu

- Ovo je **(individualni) projektni** kolegij!
  - Procedura je poznata, projekt je potrebno predati **3 dana prije datuma objavljenog na IE**
- Zadatak je podijeljen u dva dijela:
  - **Prvi projekt - `Dionis`** (ishodi 1, 4 i 5)
  - **Drugi project - `Apolon`** (ishodi 2 i 3)

Informacije o projektu

- Projektna rješenja mogu biti napisana u bilo kojem programskom jeziku (probajte se držati „klasičnih” OOP jezika)
- Primjeri vezani za prvi dio kolegija bit će pokazani u C#-u (napredni jezični koncepti i Entity Framework ORM-a)
- Primjeri vezani za drugi dio kolegija bit će pokazani u Python-u
- Koristit će se sljedeća rješenja za pohrane u oblaku:
  - Relacijske baze podataka: **Supabase** (Postgres instance)
  - Nerelacijske baze podataka: **MongoDB Atlas** (MongoDB instance)
  - Unstructured databases: **MiniO** bucket storage (S3 compatible)

Informacije o projektu

- Ukoliko planirate koristiti neki „egzotični“ jezik, najljepše vas molim vas da me kontaktirate kako bismo mogli razgovarati o daljnjim koracima! ☺

PRVI PROJEKT - APOLON

Prvi projekt

- Vaš zadatak je napisati **vlastitu implementaciju** *Object Relational Mapper* (ORM) alata
- Značajke koje morate implementirati:
  - Mapiranje klasa na tablice u bazi podataka (s pripadajućim mapiranjem podatkovnih tipova i podrškom za ograničenja - *constraints*)
  - Navigacijska svojstva za dohvat povezanih podataka (1-N, N-1, 1-1)*
  - Dobro definirano upravaljne poveznicama s bazom podataka kao i implementacija *Unit of Work* obrasca
  - Osnovne CRUD operacije koristeći pristup po želji (uključujući filtriranje i sortiranje – idealno korištenjme *flutent API**)
  - Stvaranje, izvršavanje i *roll back* migracija

ALGEBRA

Prvi projekt

- Demonstrirajte potrebne funkcionalnosti implementacijom **jednostavne web ili konzolne aplikacije** koja koristi vašu ORM implementaciju za CRUD operacije and zadanom shemom baze podataka
- Aplikacija bi trebala podržavati relacijski model podataka koji realizira scenarij upravljanja podacima o **pacijentima,** njihovom **poviješću bolesti** (medicinskim kartonima), **pregledima** i **receptima**

Prvi projekt

- **Ishod učenja 1**
  - Stvorite instancu Postgresa u oblaku i koristeći vašu implementaciju ORM-a omogućite izvršavanje DDL schema upita temeljenog na definiciji klasa (entiteta) u aplikativnom kodu– **10 bodova**
  - Implementirajte izvođenje CRUD u vašem ORM-u (uz filtriranje i sortiranje) **– 10 bodova**
- **Ishod učenja 4**
  - Osigurajte da shema relacijske baze podataka zadovoljava zahtjeve 3. normalne forme– **10 bodova**
  - Implement retrieval of related data using navigational properties – **10 bodova**
- **Ishod učenja 5**
  - Implementirajte strategiju upravljanja poveznica s bazom podataka kao i upravljanja transakcijskim kontekstom (*Unit of Work* obrazac) – **10 bodova**
  - Implementirajte funkcionalnosti vezane za upravljanje migracijama **– 10 bodova**

DRUGI PROJEKT - DIONIS

Drugi projekt

- Biološki podaci o pticama ☺

- Dostupni na: https://aves.regoch.net (podaci preuzeti s GBIF portala)

Drugi projekt

- Vaš je zadatak implementirati *pipeline* za obradu podataka ptica i njihovih opažanja koji se sastoji od **četiri koraka.** Glavni cilj ovog zadataka je generirati skup podataka koji sadrži informacije o pticama i njihovim opažanjima na temelju:
  - audio datoteka koje sadrže ptičji pjev
  - Ornitoloških podataka objavljenih na vanjskom servisu
- Za ovo rješenje obavezno je koristiti:
  - **MongoDB**
  - **MinIO**
- Sve ostalo po želji!

Drugi projekt

- Kako biste ostvarili maksimalni broj bodova, *pipeline* treba podijeliti na više manjih skripti orkestriranih bilo kojim alatom koji omogućuje izvršavanje s jednom ulaznom točkom (eng. *entrypoint*) te s mogućnošću definiranja opcionalnih parametara za vrijeme izvođenja

  Preporuka: **Snakemake**!

- Za dodatne bodove, potrebno je omogućiti izvršavanje skripte putem ručno pokrenutog **Github Actions** *workflowa* te vizualizaciju podataka

Drugi projekt

- Ishod učenja 2:
  - Procesiranje audio datoteka prijenosom u MinIO pohranu – **10 bodova**
  - Klasifikacija ptica temeljem audio datoteka te korištenjem dostpunog klasifikacijskog API-a – **10 bodova**
- Ishod učenja 3:
  - Dohvat podataka o pticama koristeći *web scraping* te pohrana podataka u MongoDB kolekciju – **10 bodova**
  - Čitanje poruka objavljenih na Kafka brokeru koje sadrže informacije o biološkim obzervacijama ptica – **5 bodova**
  - Implementacija filtriranja rezultatnog seta koritšenjem *fuzzy matching* pristupa te čišćenja i odgovarajućih transforamcija podataka – **5 bodova**

ALGEBRA

Obavezna priprema !!

WSL

Supabase

MongoDB Atlas

- Postgres documentation: https://www.postgresql.org/docs/
- C#: https://learn.microsoft.com/en-us/dotnet/csharp/
- Entity Framework: https://learn.microsoft.com/en-us/ef/core/
- Python: https://docs.python.org/3/
- Pandas: https://pandas.pydata.org/docs/getting_started/index.html

Priprema za prve vježbe

- Napravite besplatni račun na Supabase servisu
- Stvorite novu Postgres instancu
- Pokušajte ostvariti konekciju s bazom podataka kroz programski kod
- Ispišite listu tablica i njihove stupce koristeći *information_schema.columns* pogled (uključujući samo *public* shemu)

Pristup podacima iz programskog koda

UVODNO PREDAVANJE

Što su uopće podaci?

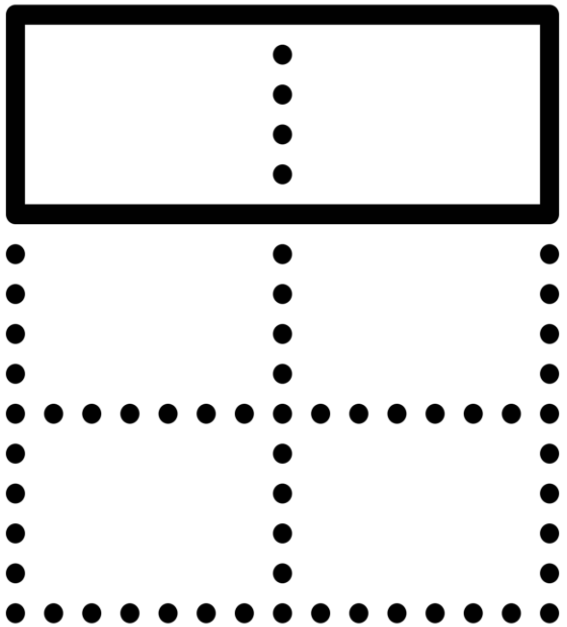
- Temelj za razvoj korisne programske podrške
- Ontološka podjela na **kod** i **podatke** ☺
- Obavezno izbjegavati nasumične *buzzworde* poput “data”, “data team”, “big data team”, “business intelligence”, “data scientist”, “business analytics”, “data analytics”…

Podaci - Informacije

	DATA	INFORMATION
Meaning	Data is raw, unorganized facts that need to be processed. Data can be something simple and seemingly random and useless until it is organized	When data is processed, organized, structured or presented in a given context so as to make it useful, it is called information
Example	Each student's test score is one piece of data	The average score of a class or of the entire school is information that can be derived from the given data

Odabir podataka

- Promatramo dvije metode:
  - Odabir podataka (redova)
  - Odabir atributa ili karakteristika (stupci)



Čišćenje podatka

- Čišćenje podataka znači uzeti u obzir specifičnosti podataka koji će biti korišteni

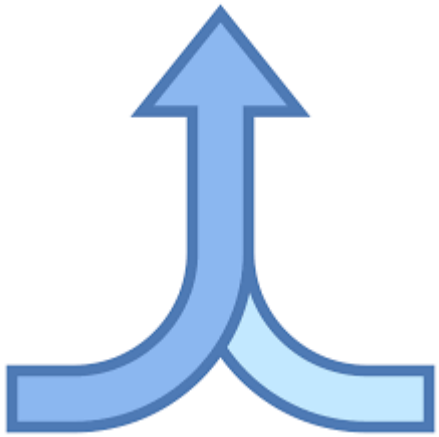
Data problem	Solution
Missing data	• Exclude rows or characteristics. • Fill blanks with an estimated value.
Data errors	• Use logic to manually discover errors and replace • Exclude characteristics
Coding inconsistencies	• Decide upon a single coding scheme, then convert and replace values
Missing or bad metadata	• Manually examine suspect fields and track down correct meaning

Stvaranje novih podataka

- Ponekad je slučaj da trebamo stvoriti novi podatke
  - Recimo, možemo dodati stupac je li objavljena transakcija za kupovinu produljene garancije
- Generalno postoje dva načina za stvaranje novih podataka:
  - Deriviranje atributa (stupaca ili karakteristika)
  - Generiranje redova

Integracija podataka

- Nije rijetka situacija da dani sustav treba koristiti više izvora podataka
- Dvije su osnovne metode integracije podataka:
  - Spajanje podataka sličnih redova, ali drugačijih atributa
  - Pridodavajući podatke koje integriraju jedan ili više setova podataka sa sličnim atributima, ali različitim zapisima

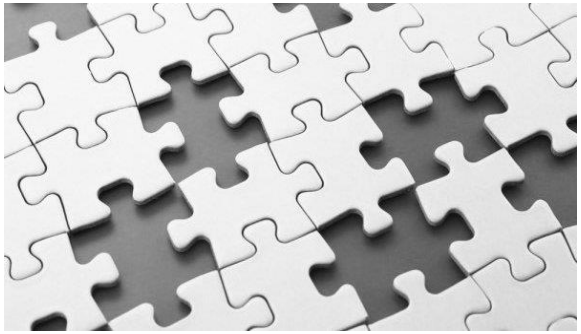


Postupci pripreme podataka

- **Čišćenje podataka**
  - Ispravljanje ”loših” podatke, filtriranje netočnih podataka iz skupa podataka, smanjivanje nepotrebnih detalja podataka
- **Transformacija podataka**
  - Podaci su konsolidirani kako bi rezultati korištenja podataka bili primjenjivi
- **Integracija**
  - Spajanje podataka iz više izvora
- **Normalizacija**
  - Izražavanje podataka (većinom numeričkih) u istoj skali, rasponu ili mjernoj jedinici
- **Imputacija podataka koji nedostaju**
  - Popunjavanje potrebnih podataka koji nedostaju s intuitivnim zamjenama
- **Identifikacija buke**
  - Detekcija nasumičnih pogrešaka i varijance u očitanjima

Tipovi podataka koji nedostaju

- Poznajemo više tipova:
  - **Null ili sistemske vrijednosti** – nonstring vrijednosti koje su ostavljene u bazi podataka ili datoteci te nisu definirane kao vrijednosti koje nedostaju
  - **Prazni stringovi i razmaci**- stringovi bez ”vidljivih” znakova
  - **Blank ili korisnički definirane vrijednosti koje nedostaju** - vrijednosti poput *nepoznato*, 99 ili –1 koje su eksplicitno definirane u izvoru



Jednostavne metode za podatke koji nedostaju

- Utvrdite jesu li vrijednosti u stupcima zaista potrebne
  - Provjerite ciljni sustav kako biste ustvrdili da polje treba imati vrijednost
  - Provjerite header stupca i podatkovni tip
  - Ukoliko je moguće, provjerite zahtjeva li izvorni sustav unos u polje

Popravak:

- Konstanta vrijednost
- Funkcija
- Kopiranje podatka iz drugog stupca
- Brisanje redova koji sadrže polje koje nedostaje
- Brisanje cijelog stupca
  - Možemo obrisati stupac ukoliko je podatak nepotreban ili neupotrebljiv*

Normalizacija podataka

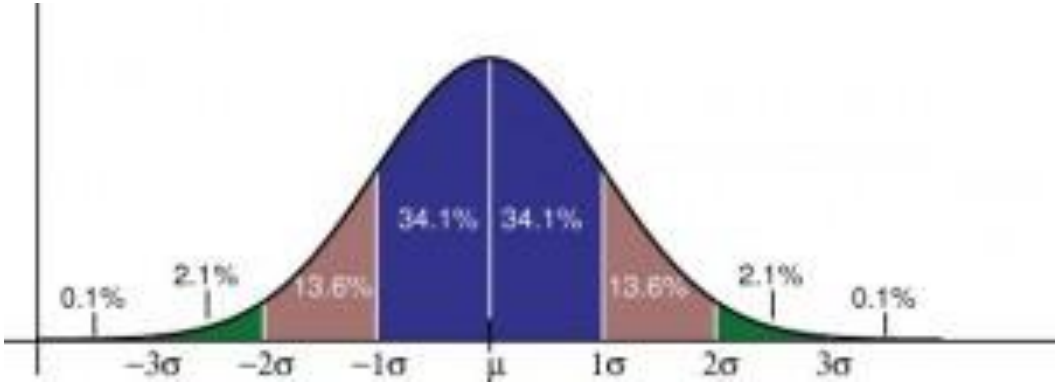
- **Min-max normalizacija** skalira numeričke vrijednosti **v** numeričkog atributa **A** u zadani raspon definiran kao [new min_A, new max_A]

$$v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A$$

- Z-score normalizacija …

Normalna distribucija

- Normalne distribucije su važne u statistici i često se koriste u prirodnim i društvenim znanostima za predstavljanje slučajnih varijabli s realnim vrijednostima čije distribucije nisu poznate
- Većina učenika će postići prosjek (3), dok će manji broj učenika postići 2 ili 4. Još manji postotak učenika postići će 1 ili 5
- Puno grupa prati ovaj uzorak:
  - Visina ljudi
  - Pogreške mjerenja
  - Krvni tlak
  - Ostvareni bodovi na ispitu
  - Plaće zaposlenika



ALGEBRA

Linearne transformacije

- U području znanstvenih otkrića i upravljanja strojevima, normalizacije možda neće biti dovoljne za prilagodbu podataka radi poboljšanja generiranog modela.
- U tim slučajevima agregiranje informacija sadržanih u različitim atributima može biti korisno
- Linearne transformacije temelje se na jednostavnim algebarskim transformacijama kao što su zbrojevi, prosjeci, rotacije, translacije…

Primjeri:

- cm → inčevi
- Fahreheit → Celsius

To je to (za sada ☺)

- Pitanja?