Agencija za
strukovno obrazovanje
i obrazovanje odraslih

OBRAZOVNI MATERIJAL ZA STRUČNO USAVRŠAVANJE NASTAVNIKA STRUKOVNIH PREDMETA

Modul: ***Usavršavanje u području struke: nova dostignuća i praćenje promjena – sektor elektrotehnike i računarstva***
Baze podataka i SQL jezik

Autor: ***Davorka Božičević***

Opis modula

<table>
<tr><th colspan="3">OSNOVNI PODATCI</th></tr>
<tr><td>Naziv modula</td><td colspan="2">Usavršavanje u području struke: nova dostignuća i praćenje promjena</td></tr>
<tr><td rowspan="3">Bodovna vrijednost i način izvođenja nastave</td><td>Kreditni bodovi</td><td>1</td></tr>
<tr><td>Broj sati vođene edukacije (uživo)</td><td>min. 12</td></tr>
<tr><td>Broj sati osobnih aktivnosti polaznika</td><td>max. 18</td></tr>
<tr><th colspan="3">CILJ MODULA</th></tr>
<tr><td colspan="3">Unaprijediti strukovne kompetencije nastavnika.</td></tr>
<tr><th colspan="3">OPIS MODULA</th></tr>
<tr><td colspan="3">Modul je generički namijenjen za predstavljanje novih dostignuća i promjena u struci i srodnim područjima i aspektima (npr. zakonska regulativa i sl.) nastavnicima koji bi ih trebali implementirati u vlastitoj praksi i nastavi.

Preporučeni sadržaj/struktura modula:
• izazovi i iskustva u vlastitoj strukovnoj/stručnoj praksi,
• nova znanja, tehnologije i dobre prakse u struci,
• primjeri svladavanja izazova u strukovnoj/stručnoj praksi (rješavanje problema) uz pomoć novih znanja, tehnologije i dobre prakse u struci,
• implementacija novih znanja, tehnologija i dobre prakse u vlastitu strukovnu/stručnu i nastavnu praksu,
• vrednovanje primjene novih znanja, tehnologija i dobre prakse u struci,
• prijenos novih znanja, tehnologija i dobre prakse na učenike i suradnike.</td></tr>
<tr><th colspan="3">ISHODI UČENJA ZA MODUL</th></tr>
<tr><td colspan="3">Nakon uspješno završenog modula polaznik će moći:
• objasniti inovacije/novine i unapređenja u struci,
• integrirati nova znanja, tehnologije i dobre prakse u vlastitu strukovnu/stručnu i nastavnu praksu i rješavanje problema,
• vrednovati korisnost i učinkovitost primjene novih znanja, tehnologija i dobre prakse u struci,
• osmisliti prijenos novih znanja, tehnologija i dobre prakse na učenike i suradnike.</td></tr>
<tr><th colspan="3">NAČIN VREDNOVANJA</th></tr>
<tr><th colspan="2">Elementi praćenja i provjeravanja</th><th>Opterećenje u kreditnim bodovima</th></tr>
<tr><td colspan="2">Vođena edukacija</td><td>0,4</td></tr>
<tr><td colspan="2">Samostalne aktivnosti polaznika</td><td>0,6</td></tr>
<tr><td colspan="2">Završno vrednovanje</td><td>0</td></tr>
</table>

Ukupno	**1**
KADROVSKI UVJETI	
Modul trebaju realizirati stručnjaci iz pojedinih obrazovnih sektora zaposleni na visokoškolskim institucijama (npr. za prijenos novih znanja, tehnologija), u realnom sektoru (npr. za prijenos tehnologija i dobre prakse) ili pak srodnim strukovnim školama (npr. primjeri dobre prakse implementacije novih dostignuća).	

Napomena: opis modula sastavni je dio Koncepta novog modela stručnog usavršavanja nastavnika strukovnih predmeta kojega je Agencija razvila u okviru ESF-ovog projekta

Sadržaj modula: IZAZOVI I ISKUSTVA U VLASTITOJ STRUKOVNOJ/STRUČNOJ PRAKSI
NOVA ZNANJA, TEHNOLOGIJE I DOBRE PRAKSE U STRUCI

BAZE PODATAKA – OSNOVNI POJMOVI

Ishod/i učenja koji se ostvaruju kroz sadržaj:

- objasniti inovacije/novine i unapređenja u struci
- integrirati nova znanja, tehnologije i dobre prakse u vlastitu strukovnu/stručnu i nastavnu praksu i rješavanje problema
- znati kolika je važnost čuvanja integriteta baza podataka
- upoznati se s fazama dizajniranja baze podataka
- naučiti modelirati entitete i veze
- naučiti kako provesti normalizaciju primjenom normalnih formi

Opis obrazovnog sadržaja:

Baze podataka koriste se kako bi olakšale računalni pristup podacima, učinile ga pouzdanijim, lakšim za pretraživanje te kako bi čitav proces pohranjivanja i pretraživanja podataka u aplikacijama učinile produktivnijim. Baze podataka definiramo kao skup međusobno povezanih podataka koji su pohranjeni u vanjskoj memoriji računala. Ti podaci su istovremeno dostupni raznim korisnicima i aplikacijskim programima. Njih možemo putem zajedničkog softvera mijenjati, čitati, brisati. Pritom korisnici i aplikacije ne moraju poznavati detalje fizičkog prikaza podataka, već se referenciraju na logičku strukturu baze. Za upravljanje bazom podataka koristimo DBMS sustav (Data Base Management System), odnosno sustavom upravljanja baze podataka (SUBP).
SUBP nam omogućuje oblikovanje fizičkog izgleda baze u odnosu na njenu logičku strukturu. Neki od najpoznatijih sustava su: Oracle, DB2, MySQL, Informix, PostgreSQL i SQL server. SUBP-om se koristimo kako bismo obavljali sve operacije s podacima u pojedinoj bazi, on također brine za sigurnost navedenih podataka te automatizira administrativne poslove s podacima u bazi. Svi podaci u bazi su logički organizirani prema jednom od modela podataka. Model podataka je skup pravila koja određuju kako može izgledati logička struktura baze, a odabir modela čini osnovu za koncipiranje, projektiranje i implementiranje baze.
Podaci u bazi su logički organizirani na temelju jednog od postojećih modela podataka.
Neki od modela kojima se SUBP može služiti su: relacijski model zasnovan na matematičkim pojmu relacije, prema kojem su podaci prikazani pravokutnim tablicama, mrežni model koji je predočen usmjerenim grafom, hijerarhijski model predočen stablom i s hijerarhijskim odnosom nadređeni-podređeni, te objektni model inspiriran objektno-orijentiranim programskim jezicima, u kojima je baza skup trajno pohranjenih objekata koji se sastoje od svojih internih podataka i operacija za rukovanje tim podacima koji su uređeni prema klasama.

Integritet baza podataka podrazumijeva čuvanje ispravnosti i konzistentnosti podataka. To očuvanje postiže se implementacijom metode provjere grešaka, validacijskih procedura i raznih ograničenja. Ograničenja koja postavljamo su ništa drugo nego pravila koja konzistentni podaci moraju zadovoljavati, a ako nisu zadovoljena, SUBP će nam poslati poruku o greški, te neće izvršiti traženu promjenu. Ograničenja koja SUBP može postaviti odnose se na čuvanje integriteta domene, integriteta unutar relacije, te referencijalnog integriteta.

S obzirom na to da su baze podataka uglavnom namijenjene većem broju korisnika u isto vrijeme, SUBP mora pažljivo koordinirati istovremeni rad s bazom, kako ne bi došlo do zloupotrebe i neovlaštenog pristupa podacima. Također, svaki korisnik treba imati osjećaj kako sam radi s bazom, iako istovremeno možda tri računala koriste jedan isti podatak. Rad korisnika u bazi se temelji na pokretanju unaprijed definiranih transakcija. Iako se jedna transakcija čini kao jedna operacija, ona se sastoji od nekoliko njih. Prema tome, stanja između pojedinih operacija podataka dovode do nekonzistentnog stanja u samoj bazi. Kako bi stanje baze ostalo konzistentno i kako bi njen integritet bio očuvan, transakcija koja se provodi mora biti u cijelosti izvršena ili uopće ne smije biti izvršena. Tako svaka započeta, a nedovršena transakcija, mora biti poništena.

Kako bi razumjeli važnost pravilne zaštite podataka, potrebno je shvatiti moguće posljedice krađe podataka. U današnjoj sveopćoj povezanosti raznih sustava i sve većoj digitalizaciji sadržaja ogroman broj informacija nalazi se pohranjen u bazama podataka. Neki od važnijih primjera:
1. Zdravstvene podatke kao što su digitalizirani zdravstveni kartoni (e-kartoni) koji sadrže anamnezu pojedinih pacijenata.
2. Financijski podaci kao što su osobni računi u bankama, ali i interni podaci raznih tvrtki o njihovom poslovanju.
3. Strogo povjerljivi podaci poput državnih tajni, informacija o kretanjima i planovima vojske također su na meti potencijalnih napada.
Najrašireniji skupina podataka su osobni podaci. Prema istraživanju tvrtke International Data Corporation indirektne i direktne aktivnosti pojedinaca stvorile su otprilike 70% digitalnih podataka 2019. godine. Osobni podaci imaju razne primjene, uključujući marketing, poboljšanje online iskustva, ali se također može koristiti za razna kriminalna djela kao što je krađa identiteta, uhođenje i razni drugi zločini na razini pojedinca i njegove obitelji.

Dizajniranje baze podataka

Prije nego što se upustimo u kreiranje i rad s bazama podataka, dobro je proći barem neku fazu dizajniranja. Naravno, kada se radi s nekom manjom bazom, mnogo se toga može dizajnirati 'napamet', dovoljna će biti i skica na papiru. No, čim se krene raditi sa većim bazama, treba uložiti veći napor da bi se krajnja baza dobro napravila. Uvođenje baze podataka u neko poduzeće ili ustanovu predstavlja složeni zadatak koji zahtijeva timski rad stručnjaka raznih profila.
Taj se projekt koji se može podijeliti u pet faza:
- analiza potreba
- modeliranje podataka
- implementacija
- testiranje
- održavanje

Analiza potreba

Da bismo razvili bazu za neki sustav moramo prikupiti informacije o tom sustavu, bilo kroz proučavanje samog sustava bilo komunikacijom sa stručnjacima koji se njime služe. Tako se sustav i podaci koje je potrebno zapamtiti u bazi detaljno upoznaju, kao i način na koji se ti podaci prikupljaju. Rezultat ovog koraka treba biti početna dokumentacija sustava koja će služiti da bi se izveli daljnji koraci. Rezultat analize je dokument (pisan neformalno u prirodnom jeziku) koji se zove specifikacija potreba. Kada imamo takvu dokumentaciju krećemo na konceptualni dizajn. To znači da trebamo detaljno raščlaniti podatke koji čine bazu služeći se modelom visokog nivoa (koriste se tzv. ER dijagrami).

Modeliranje podataka

U ovom koraku identificiramo pripadnike (entitete) koji čine sustav i shvaćamo način na koji su povezani (tj. točno utvrđujemo njihovu međusobnu relaciju). Sljedeći je korak logički dizajn u kojem iz modela visokoga nivoa izvodimo model za implementaciju, tj. iz ER dijagrama kreiramo stvarnu bazu.

Implementacija

Na temelju sheme i uz pomoć dostupnog DBMS-a, fizički se kreira baza podataka na računalu. U DBMS-u postoje parametri kojima se može utjecati na fizičku organizaciju baze, a oni se podešavaju tako da se osigura efikasan rad najvažnijih transakcija. Baza se puni podacima. U fizičkom dizajnu, odlučujemo (točnije, administrator baze odlučuje) kako će baza biti zapisana (na disku, traci, na više diskova...) i na koje će se sve to načine događati. Te se odluke obično donose nakon razmišljanja o performansama i dostupnosti hardvera.

Testiranje

Korisnici testiraju rad s bazom i provjeravaju zadovoljava li ona svim zahtjevima. Nastoje se otkriti greške koje su se mogle potkrasti u svakoj od faza razvoja. Greške u ranijim fazama imaju teže posljedice. Na primjer, greška u analizi potreba dovodi do toga da transakcije možda korektno rade, ali ne i ono što korisnicima treba. Dobro bi bilo kad bi se takvi propusti otkrili prije implementacije. Zato se u novije vrijeme, prije prave implementacije, razvijaju i približni prototipovi baze podataka, koji se pokazuju korisnicima.

Održavanje

Održavanje se odvija nakon što baza već uđe u redovnu upotrebu. Sastoji se od popravaka grešaka koje nisu bile otkrivene u fazi testiranja, te od uvođenja promjena zbog novih zahtjeva korisnika i podešavanja parametara u DBMS-u u svrhu poboljšavanja performansi.

Modeliranje podataka

Relacijski model baze podataka zasnovan je na ideji da se cjelokupni skup podataka koji želimo da baza prati razdijeli na pravokutne tablice tzv. relacije. Svaka relacija ima svoje ime po kojem je razlikujemo od ostalih u istoj bazi. Jedan stupac relacije obično sadrži vrijednost jednog atributa (za entitet ili vezu), pa zato stupac poistovjećujemo s atributom. Atribut ima svoje ime po kojem ga razlikujemo od ostalih u istoj relaciji. Vrijednosti jednog atributa čine podaci istog tipa. Dakle, definiran je skup dopuštenih vrijednosti za atribut, koji se zove domena atributa. Vrijednost atributa mora biti jednostruka i jednostavna (ne da se rastaviti na dijelove). Vrijednost atributa nije uvijek obvezna za unos. Jedan redak relacije predstavlja jedan zapis entiteta.
Neka svojstva tablice u relacijskom modelu su:

- tablica je skup zapisa u bazi podataka
- tablica sadrži više stupaca
- svaki stupac ima jedinstven naziv
- svaki redak predstavlja jedan zapis nekog entiteta
- vrijednost nekog podatka dobije se na presjeku odgovarajućeg retka i odgovarajućeg stupca

Danas je činjenica da se RDBMS koristi u većini komercijalnih sustava. Preostaje još pitanje zašto je RDBMS bolji od ostalih modela? Prvo, pokazalo se da su i mrežni i hijerarhijski modeli usko vezani uz implementaciju baze, a relacijski model nije, pa je relacijski model popularniji izbor (kada je nešto usko vezano uz implementaciju, veliki problem postaju naknadne izmjene u strukturi, što uvelike otežava održavanje i proširivanje baze).

Glavna prednost relacijskoga modela u tome je što vrlo efikasno rješava dva velika problema koja se javljaju kod pohrane podataka: redundanciju (ponavljanje) i nekonzistentnost podataka.

Polaznici i tečajevi				
Šifra polaznika	**Ime polaznika**	**Mjesto stanovanja**	**Šifra tečaja**	**Naziv tečaja**
1	Ana Milić	Zagreb	P01	Osnove rada PC računala
2	Sanja Tarak	Split	P02	Microsoft Word
3	Mladen Gork	Osijek	D01	SQL - osnove
4	Ivana Matkić	Split	O01	Računalni operator – uredsko poslovanje
4	Ivana Matkić	Split	D05	Osnove i teorija C++
5	Marina Anić	Osijek	O02	Specijalist poslovne primjene računala
6	Ivica Limac	Split	P01	Osnove rada PC računala

Slika1: Tablica s podacima o polaznicima i tečajevima

U ovom primjeru imamo duple podatke o mjestu stanovanja Ivane Matkić i dva puta zapisanu njezinu šifru, ime i prezime. Zamislimo da sada svatko od npr. 10000 polaznika upiše po nekoliko tečajeva, koliku bismo hrpu nepotrebno dupliciranih podataka dobili! Ali, nije to sve: što ako se Ivana preseli iz Splita u Varaždin? Onda moramo locirati sva pojavljivanja Ivane Matkić u tablici (koja sada ima više od npr. 40.000 redaka), te promijeniti Split u Varaždin. Zar ne bi bilo bolje da smo morali tu informaciju promijeniti samo na jednome mjestu?
Kod takvih i sličnih problema pomaže nam postupak koji nazivamo normalizacijom.
Primarni ključ je podatak koji razlikuje jedan zapis od drugoga. Po njemu gledajući, svi su zapisi različiti. Dakle, primarni ključ je vrijednost stupca koja na jedinstven način određuje neki redak.
Primarni ključ javlja se u praksi, npr. JMBG, ISBN knjige, poštanski broj, bar_kod proizvoda i td. Polje primarni ključ čini jedinstvenim svaki zapis u tablici i time ih međusobno distancira. U relacijskim je bazama obavezno definiranje primarnoga ključa za svaku tablicu.

Modeliranje entiteta i veza

Entitet je objekt, pojava ili događaj o kojem želimo spremati podatke. To je nešto što se može identificirati, npr. student, kupac, proizvođač, polaznik seminara, servisiranje auta... Svaki entitet ima svoje karakteristike koje ga jednoznačno ili jedinstveno opisuju, a nazivaju se atributi. Na primjer, atributi kuće su kućni_broj, visina, širina, vrijednost ...
Za svaki entitet stvara se tablica ili relacija. Veza je nešto što veže dva ili više entiteta. Razlikujemo tri vrste binarnih veza i to zovemo funkcionalnost veze:
• 1 –1 veza jednom zapisu iz jedne tablice odgovara jedan i samo jedan zapis iz druge tablice; npr. veza JE_U_BRAKU_SA između entiteta MUŠKARAC i ŽENA
• 1 – n veza (ili 1:∞) jednom zapisu iz jedne tablice odgovara 0, 1 ili više zapisa iz druge; npr. veza PREDAJE između entiteta PROFESOR i KOLEGIJ (jedan profesor može predavati više kolegija na fakultetu)
• m – n veza (ili ∞:∞) jedan zapis prvog entiteta može biti u vezi s 0, 1 ili više primjeraka drugog entiteta, te također jedan zapis drugog entiteta može biti u vezi s 0, 1 ili više zapisa prvog entiteta; npr. veza UPISALA između entiteta OSOBA i SEMINAR.

Ako svaki zapis nekog entiteta mora sudjelovati u određenoj vezi, onda kažemo da taj entitet ima obavezno članstvo u toj vezi. Inače, entitet ima neobavezno članstvo. Na

primjer, između entiteta OSOBA i GRAD zadana je veza ŽIVI_U. OSOBA u vezi ŽIVI_U ima obavezno članstvo jer svaka osoba mora negdje stanovati. Rezultat ovog koraka je popisivanje svih veza i entiteta te njihovih atributa.

Normalizacija

Normalizacija je proces organiziranja podataka s ciljem minimalnog dupliciranja podataka, tj. proces kreiranja efikasne, pouzdane i fleksibilne baze podataka. Taj se postupak temelji na matematički dokazanim tvrdnjama, što znači da ćemo, budemo li ga slijedili, kao rezultat sigurno dobiti dobru bazu podataka (naravno, kao i sve drugo, i ovaj se postupak može loše provesti i dovesti do loših rezultata, no to se rijetko događa). Kako tablicu sa slike 1 možemo dovesti u dobro stanje, tj. normalizirali? Dovoljno je prethodnu tablicu razbiti na tri nove: jednu koja bi popisala polaznike, drugu koja bi popisala kolegije i treću, relacijsku tablicu, koja bi povezala prve dvije.

Polaznici			
Šifra polaznika	**Ime polaznika**	**Prezime polaznika**	**Mjesto stanovanja**
1	Ana	Milić	Zagreb
2	Sanja	Tarak	Split
3	Mladen	Gork	Osijek
4	Ivana	Matkić	Split
5	Marina	Anić	Osijek
6	Ivica	Limac	Split

Tečajevi	
Šifra tečaja	**Naziv tečaja**
P01	Osnove rada PC računala
P02	Microsoft Word
N01	SQL – osnove
O01	Računalni operator – uredsko poslovanje
O02	Specijalist poslovne primjene računala
O03	Grafički dizajner

Upisi	
Šifra polaznika	**Šifra tečaja**
1	P01
2	P02
3	N01
4	O01
5	O02
6	P01

Slika 2: Tablice nakon normalizacje za bazu podataka Polaznici i tečajevi

Ako Ivana želi upisati još jedan tečaj, tu ćemo informaciju jednostavno dodati kao novi redak u tablicu Upisi, a ostale tablice nećemo dirati. Ako se Ivana želi preseliti u Varaždin, tu ćemo informaciju promijeniti samo na jednome mjestu, u tablici Polaznici. Jasno je da smo time dobili puno, pogotovo u pogledu fleksibilnosti, integriteta podataka i efikasnosti. Do ovakvih rezultata dolazi se konzistentnom primjenom pravila koja se nazivaju normalne forme.

Postoji 6 normalnih formi:
1. Prva normalna forma (1NF)
2. Druga normalna forma (2NF)
3. Treća normalna forma (3NF)
4. Boyce-Coddova normalna forma (BCNF)
5. Četvrta normalna forma (4NF)
6. Peta normalna forma (5NF)

1NF zahtijeva da svaka vrijednost u stupcu koja se pojavljuje bude atomarna i da svi reci imaju isti broj polja. 1NF zapravo govori da se kao jedna vrijednost ne smije pojaviti skup podataka. Pravila prve normalne forme krše se najčešće tako da, primjerice, imate u jednom polju i ime i prezime polaznika, što komplicira stvari već prilikom sortiranja zapisa po npr. prezimenima. Rijetko kada se dogodi da tablica nije u 1NF, jer nam nekako iskustvo i intuicija odmah daju „dobru stvar".
2NF zahtijeva da bude zadovoljena 1NF, te da svi reci budu jednoznačno određeni i potpuno ovisni o primarnome ključu. Primjerice u prvoj tablici stupci Šifra tečaja i Naziv tečaja nemaju veze s primarnim ključem Šifra polaznika. Naravno, tečajevi postoje neovisno o polaznicima, postoji samo veza među njima (i to tek kada polaznik upiše neki tečaj).
3NF zahtijeva da bude zadovoljena 2NF, te da ne postoje tzv. tranzitivne ovisnosti o primarnome ključu, tj. svi stupci koji nisu primarni ključ moraju biti neovisni jedni o drugima. To znači da se ne smije dogoditi da neki podatak ovisi o nekome drugom, koji pak ovisi o primarnome ključu. To se nama dogodilo u prvoj tablici: ime tečaja ovisi o šifri tečaja koja pak ovisi o šifri polaznika. Najočitije kršenje treće normalne forme predstavljaju izračunata polja – problem ažuriranja podataka. Primjerice, u jednoj tablici postoje polja Cijena i Količina, a mi želimo pomnožiti te podatke i umnožak sačuvati u tablici. Problem se javlja ako se promijene Cijena ili Količina za neki proizvod jer se polje Ukupno automatski ne ažurira.
BCNF otprilike zahtijeva da, ako postoji stupac o kojem je neki drugi ovisan, onda on mora biti ključ u tablici. Ako je neka relacija u BCNF, onda je i u 3NF (naravno, i u 2NF i u 1NF), ali postoje situacije kada je relacija u 3NF, a nije u BCNF.
4NF i 5NF se u praksi rijetko koriste, čak se i BCNF ne koristi tako često jer se u većini situacija možemo „snaći" i problem riješiti drukčije. Naravno, idealno bi bilo dovesti bazu u 5NF, ali je s praktičnog stajališta to često nepotrebno. Naime, razbijanjem baze na mnogo malih tablica usporava se čitanje s medija, što nekad može biti lošije rješenje od nenormalizirane tablice. Osoba koja implementira bazu trebala bi isprobati i na temelju iskustva zaključiti dokle treba provesti normalizaciju.

Predloženi načini vrednovanja/ ostvarivanja ishoda obrazovnog sadržaja:

- ispit od 10 pitanja s višestrukim izborom
- sudjelovanje u raspravama na forumu na Loomenu
- sudjelovanje na Zoom platformi

Sadržaj modula: PRIMJERI SVLADAVANJA IZAZOVA U STRUKOVNOJ/STRUČNOJ PRAKSI (RJEŠAVANJE PROBLEMA) UZ POMOĆ NOVIH ZNANJA, TEHNOLOGIJE I DOBRE PRAKSE U STRUCI

IMPLEMENTACIJA NOVIH ZNANJA, TEHNOLOGIJA I DOBRE PRAKSE U VLASTITU STRUKOVNU/STRUČNU I NASTAVNU PRAKSU

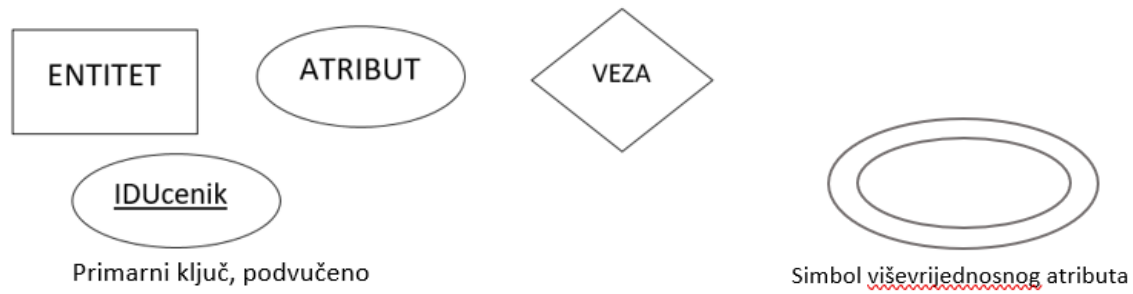
Pretvaranje ER dijagrama u relacijski model

Ishod/i učenja koji se ostvaruju kroz sadržaj:

- objasniti inovacije/novine i unapređenja u struci
- integrirati nova znanja, tehnologije i dobre prakse u vlastitu strukovnu/stručnu i nastavnu praksu i rješavanje problema
- znati napraviti ER dijagram
- znati pretvoriti ER dijagram u relacijsku shemu

Opis obrazovnog sadržaja:

ER dijagram je dijagram koji pokazuje relacije između entiteta u sustavu. Kratica 'ER dijagram' dolazi od engleskih riječi Entity Relationship Diagram. ER model dovoljno je jednostavan da ga ljudi različitih struka mogu razumjeti. Zato ER shema služi za komunikaciju projektanta baze podataka i korisnika, i to u najranijoj fazi razvoja baze.



Slika 3: Simboli ER dijagrama

Postojeći DBMS ne mogu direktno implementirati ER shemu, već zahtijevaju da se ona detaljnije razradi. Na ER dijagramu entiteti se prikazuju pravokutnicima, a veze među njima rombovima koji su s entitetima povezani bridovima.
Relacije među entitetima mogu biti unarne, binarne ili n-arne. Ako je relacija unarna to znači da je broj entiteta koji sudjeluju u relaciji jedan. U binarnoj relaciji sudjeluju dva, a u n-arnoj više od dva entiteta.
Kreiranje ER dijagrama je iterativan proces, što znači da kako bismo došli do konačne verzije, moramo nacrtati više „probnih" verzija, tj. više puta probati dok ne dobijemo odgovarajući zapis. Kada se skupi iskustva, broj probnih dijagrama se smanjuje. Nakon što dobijemo ER dijagram, pristupit ćemo izradi same baze. Baza se iz ER dijagrama dobiva pretvorbom dijagrama u tablice.

Objekt iz stvarnog svijeta koji se želi predstaviti u bazi podataka naziva se entitetom. Entitet može biti stvaran objekt ili osoba (kao npr. polaznik tečaja), ili pak apstraktni objekt

ili događaj (pohađanje tečaja). Tip entiteta određen je njegovim atributima. Tako je tip entiteta Polaznik određen atributima Ime i Prezime.

Za svaki tip entiteta koji postoji u modelu, stvara se tablica u bazi podataka. Stupci tablice odgovaraju atributima entiteta. Za svaki atribut potrebno je, osim imena atributa, definirati i koje vrijednosti on može poprimiti, tj. koji je njegov tip podatka. Jedan redak u tablici predstavlja pojedini entitet iz stvarnog svijeta, dakle jedan redak u tablici Polaznik predstavlja točno određenog polaznika s njegovim imenom i prezimenom.

Tipovi podataka

Baze podataka podržavaju velik broj tipova podataka. Ovdje su navedeni oni koji se najčešće koriste:

char - služi za prikazivanje standardnih ASCII znakova fiksne dužine

varchar - služi za prikazivanje standardnih ASCII znakova dužine

nchar - služi za prikazivanje Unicode znakova fiksne dužine

nvarchar - služi za prikazivanje Unicode znakova varijabilne dužine

money – valuta s preciznošću na 4 decimale

float – brojevi u rasponu od $-1,79*10^{308}$ do $1,79*10^{308}$

real – brojevi u rasponu od $-3,4*10^{38}$ do $3,4*10^{38}$

datetime – prikazuje datum i vrijeme

Navedeni tipovi podatka odnose se na sustav za upravljanje bazama podataka Microsoft SQL Server, mada većina ovih tipova ima svoje ekvivalente na drugim sustavima. Prilikom definicija stupca u tablici određuje se njegov tip podatka, odnosno kakve je vrijednosti moguće spremiti u tom polju.

Slijedi logička faza koja prikazuje logičku strukturu baze, odnosno organizaciju podataka u bazi. Relacijski model je sastavljen od relacija - tablica. Glavno obilježje relacijskog modela je uspostavljanje veza odnosno relacija između tablica. Pri povezivanju dviju tablica služimo se stranim ili vanjskim ključem.

Veza između entiteta uspostavlja se najčešće preko primarnih ključeva. Atribut preko kojeg se dva entiteta, odnosno dvije relacije povezuju, mora biti atribut i jednog i drugog entiteta. Jednom entitetu on je primarni, a drugom entitetu strani ključ.

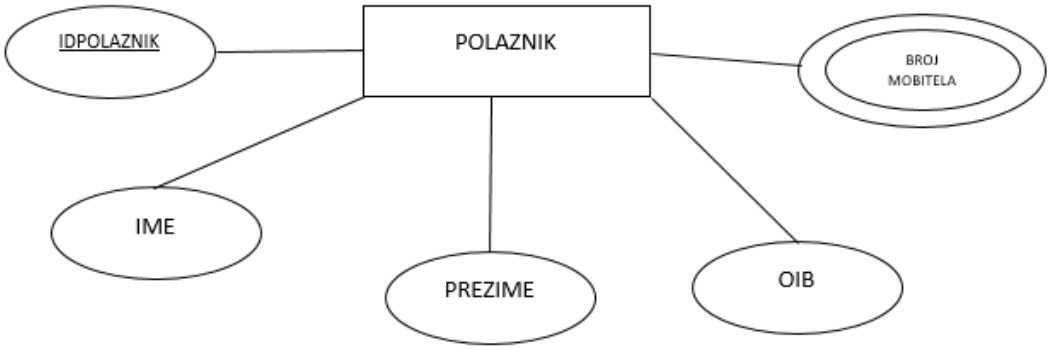
ER model posebnim transformacijskim pravilima pretvaramo u relacijski model. Tako se dobije logička shema baze podataka – ona daje stvarni raspored i stvarnu organizaciju podataka.

Poštujući pravila pretvaranja ER modela, dobije se relacijska shema koja sadrži pojedine sheme relacije.
Shema relacije sastoji se od naziva tablice, naziva atributa i podcrtanog primarnog ključa.

Pravila pretvorbe

Pravilo pretvaranja entiteta

Entitet iz ER modela postaje tablica u relacijskom modelu.
Naziv entiteta postaje naziv tablice, a svaki atribut ER modela postaje stupac tablice. Jedinstveni identifikacijski atribut entiteta postaje primarni ključ tablice.



Dijagram entiteta Polaznik

Polaznik			
IDPOLAZNIK	IME	PREZIME	OIB

Atribut BROJ MOBITELA je viševrijednosni atribut i postoji posebno pravilo za njegovo pretvaranje.

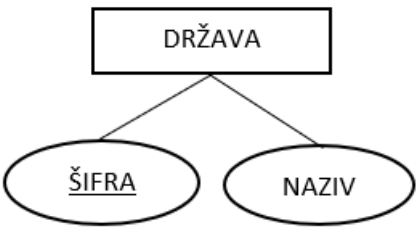
Shema relacije:

Polaznik (IDPOLAZNIK, IME, PREZIME, OIB)

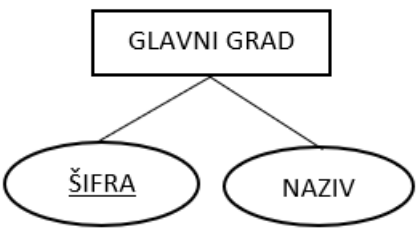
Pravilo pretvaranja veze 1:1

Postoje tri načina pretvorbe, ovisno o članstvu veze.

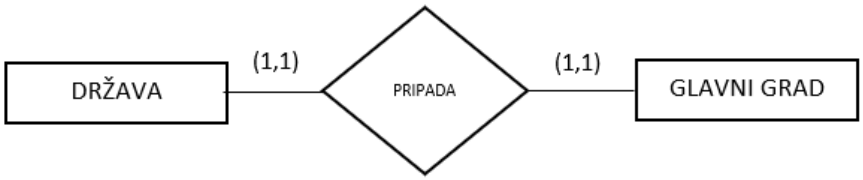
a) Ako su članstva za oba entiteta obvezna, entiteti se spajaju u jednu tablicu koja sadrži njihove atribute i zajednički primarni ključ.
Npr. entiteti Država i Glavni grad – članstvo i jednog i drugog entiteta je obvezno jer svaka država ima glavni grad i svaki glavni grad pripada državi.



Dijagram entiteta Država



Dijagram entitete Glavni grad



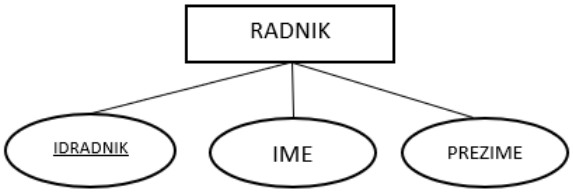
ER dijagram za Državu i Glavni grad

Država Grad		
ŠIFRA	DRŽAVA	GLAVNI GRAD

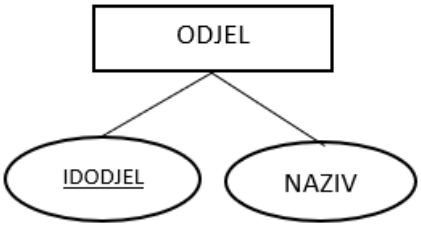
Shema relacije:

DržavaGrad (ŠIFRA, DRŽAVA, GLAVNI GRAD)

b) Ako je članstvo samo za jedan entitet obvezno, u taj entitet dodaje se strani ključ (dodaje se primarni ključ drugog entiteta koji nema obvezno članstvo)

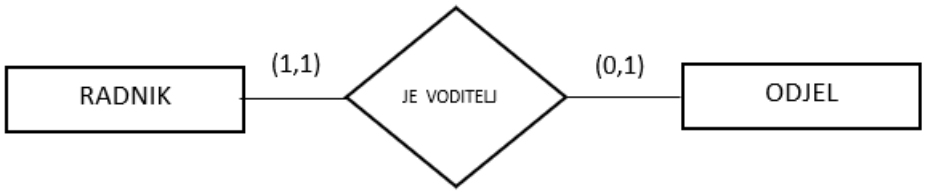


Dijagram entiteta Radnik



Dijagram entiteta Odjel

Entiteti Radnik i Odjel su povezani vezom *je voditelj.*
Članstvo entiteta Radnik je neobvezan jer svaki radnik ne mora biti voditelj.
Članstvo entiteta Odjel je obvezno jer svaki voditelj odjela mora biti radnik.



ER dijagram za Radnik i Odjel

Atribut *IDRadnik* dodaje se entitetu *Odjel* i postaje strani ključ u tablici *Odjel*.

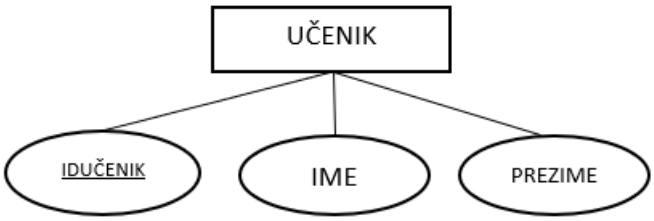
Radnik		
IDRADNIK	IME	PREZIME

Odjel		
IDODJEL	NAZIV ODJELA	IDRADNIK

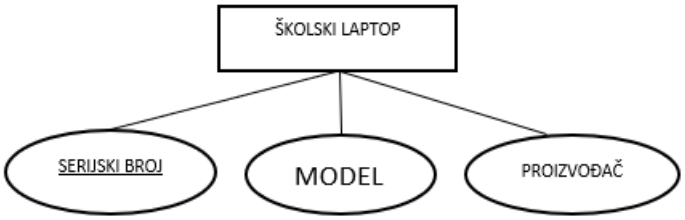
Shema relacije:

Radnik (IDRADNIK, IME, PREZIME)
Odjel (IDODJEL, NAZIV ODJELA, IDRADNIK)

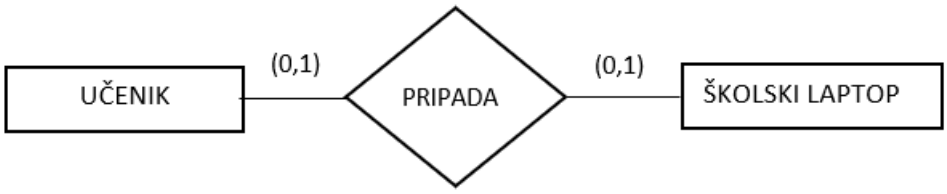
c) Ako su članstva za oba entiteta neobvezna, nastaje nova tablica koja sadrži primarne ključeve oba dva entiteta.



Dijagram entiteta Učenik



Dijagram entiteta Školski laptop



ER dijagram za Učenik i Školski laptop

UČENIK		
IDUČENIK	IME	PREZIME

ŠKOLSKI LAPTOP		
SERIJSKI BROJ	MODEL	PROIZVOĐAČ

PRIPADA	
IDUČENIK	SERIJSKI BROJ

Shema relacije:

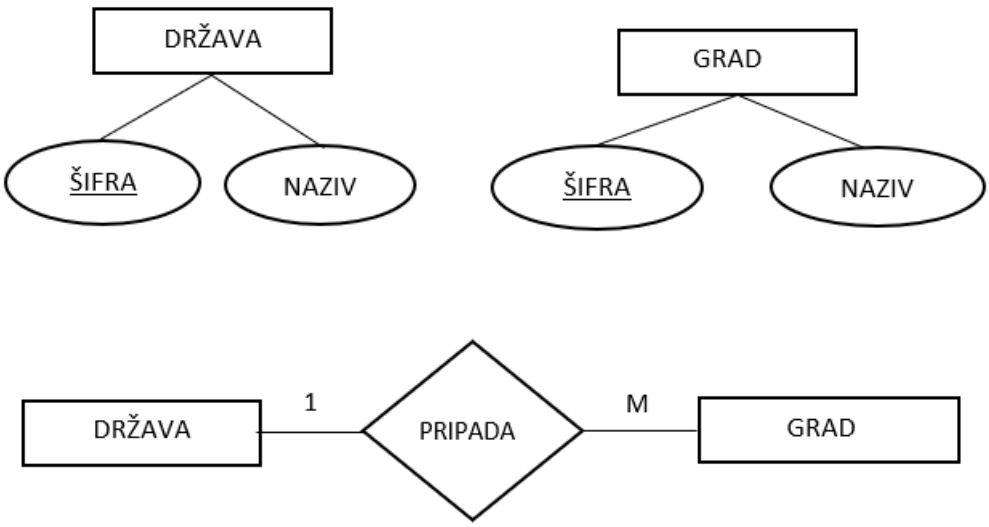
UČENIK (IDUČENIK, IME, PREZIME)
ŠKOLSKI LAPTOP (SERIJSKI BROJ, MODEL, PROIZVOĐAČ)
PRIPADA (IDUČENIK, SERIJSKI BROJ)

Pravilo pretvaranja veze 1:M

Veza 1:M između dva entiteta pretvara se tako da jedan entitet ostaje nepromijenjen, dok se u drugi entitet dodaje primarni ključ prvog entiteta i on jest strani ključ.

Primjer - entiteti Država i Grad koji su povezani vezom 1:M.

Jedna država ima više gradova, a jedan grad pripada točno jednoj državi.



Dijagrami entiteta i ER dijagram Država i Grad

DRŽAVA	
ŠIFRA DRŽAVE	NAZIV

GRAD		
ŠIFRA GRADA	NAZIV	ŠIFRA DRŽAVE

Shema relacije:

DRŽAVA (ŠIFRA DRŽAVE, NAZIV)
GRAD (ŠIFRA GRADA, NAZIV, ŠIFRA DRŽAVE)

Atribut ŠIFRA DRŽAVE u tablici GRAD je strani ključ.

Pravilo pretvaranja veze M:M

Za pretvorbu veze M:M između dva entiteta uvodi se nova tablica koja sadrži primarne ključeve oba entiteta i oni čine složeni primarni ključ nove tablice.
Primjer – entiteti Država i Rijeka su povezani vezom M:M jer jedna država ima više rijeka, a jedna rijeka teče kroz više država.

Dijagrami entiteta i ER dijagram Država i Rijeka

DRŽAVA	
ŠIFRA	NAZIV

RIJEKA		
OZNAKA	NAZIV	DULJINA

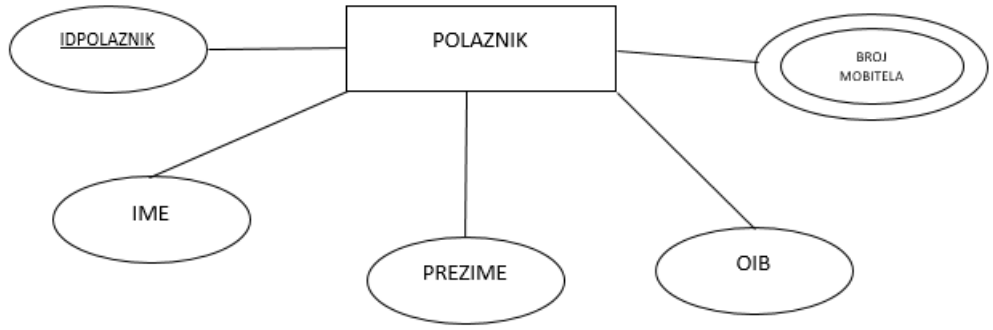
IMA	
ŠIFRA DRŽAVE	OZNAKA RIJEKE

Shema relacije:

DRŽAVA (ŠIFRA, NAZIV)
RIJEKA (OZNAKA, NAZIV, DULJINA)
IMA (ŠIFRA DRŽAVE, OZNAKA RIJEKE)

Pretvaranje viševrijednosnog atributa

Ako entitet ima viševrijednosni atribut, onda se stvara nova tablica nazvana nazivom atributa. Zatim se gleda veza između tablice entiteta i novostvorene tablice. Ovisno o funkcionalnosti veze, odabranim transformacijskim pravilom pretvara se u relacijski model.



Dijagram entiteta Polaznik

Prema 1. pravilu nastaje tablica:

Polaznik			
IDPOLAZNIK	IME	PREZIME	OIB

Viševrijednosni atribut BROJ MOBITELA postaje naziv nove tablice kojoj proizvoljno zadajemo atribute.

BROJ MOBITELA	
IDBROJ	VRIJEDNOST

Veza između Polaznika i Broj mobitela: jedan polaznik može imati više mobitela. Jedan broj mobitela može pripadati samo jednom polazniku (po korisničkom zahtjevu). Veza je 1:M pa primjenjujemo 3. transformacijsko pravilo. U tablicu Polaznik dodajemo strani ključ *IDBroj*.

Polaznik				
IDPOLAZNIK	IME	PREZIME	OIB	IDBROJ

BROJ MOBITELA	
IDBROJ	VRIJEDNOST

Predloženi načini vrednovanja/ ostvarivanja ishoda obrazovnog sadržaja:

- ispit od 10 pitanja s višestrukim izborom
- sudjelovanje u raspravama na forumu na Loomenu
- sudjelovanje na Zoom platformi

Sadržaj modula: PRIMJERI SVLADAVANJA IZAZOVA U STRUKOVNOJ/STRUČNOJ PRAKSI (RJEŠAVANJE PROBLEMA) UZ POMOĆ NOVIH ZNANJA, TEHNOLOGIJE I DOBRE PRAKSE U STRUCI

IMPLEMENTACIJA NOVIH ZNANJA, TEHNOLOGIJA I DOBRE PRAKSE U VLASTITU STRUKOVNU/STRUČNU I NASTAVNU PRAKSU

Kreiranje baze podataka

Ishod/i učenja koji se ostvaruju kroz sadržaj:

- objasniti inovacije/novine i unapređenja u struci
- integrirati nova znanja, tehnologije i dobre prakse u vlastitu strukovnu/stručnu i nastavnu praksu i rješavanje problema
- primijeniti naredbe SQL jezika u kreiranju baze i tablica
- postaviti ograničenja – CHECK, NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY

Opis obrazovnog sadržaja:

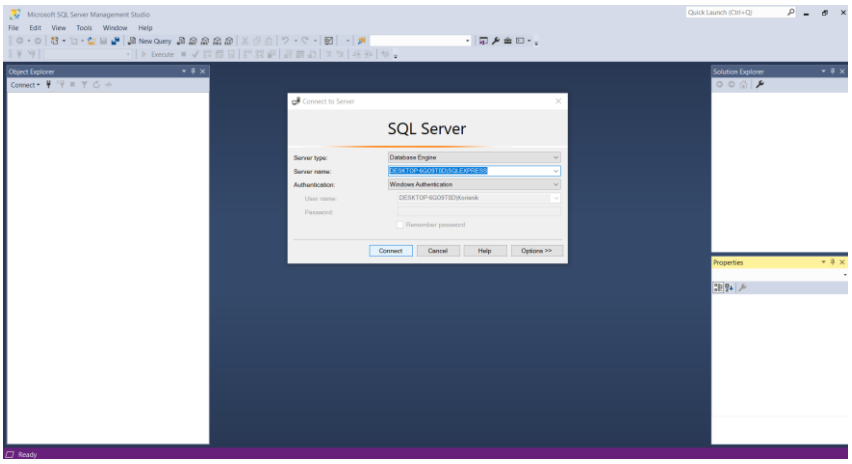
SQL jezik

SQL je kratica za Structured Query Lanquage (strukturirani upitni jezik) – jezik za postavljanje upita nad bazama podataka. Originalno razvijen u IBM-u 70-ih godina prošlog stoljeća, SQL je postao glavni jezik za rad s bazama podataka. Jezik SQL je napisan s namjerom da bude lako razumljiv i jednostavan za korištenje. On je jezik deklarativnog tipa – navodi se što se želi dobiti, a ne daju se konkretne instrukcije kako to dobiti (kao što je slučaj kod jezika proceduralnog tipa).
U upotrebi se nalazi više različitih sustava za upravljanje bazama podataka (database management systems).
Ovdje ćemo koristiti sustav za upravljanje bazama podataka Microsoft SQL Server, koji je pogodan i za velike poslovne sustave kao i za male baze podataka. Radi se o komercijalnom sustavu za koji se plaća licenca, ali koji je dostupan i u besplatnoj (Express) verziji. Osim njega, u širokoj su upotrebi i sustavi za upravljanje bazama podataka otvorenog koda - MySQL (najčešće se upotrebljava za web stranice i manje projekte) i PostgreSQL (koji je pogodan za ozbiljne primjene na velikim sustavima). U velikim poslovnim sustavima često se koristi sustav za upravljanje bazama podataka Oracle, a u upotrebi su još i sustavi Informix, Ingres i DB2. Iako je definiran SQL standard, svaki od sustava ima određene varijacije i nadogradnje na osnovni SQL jezik.

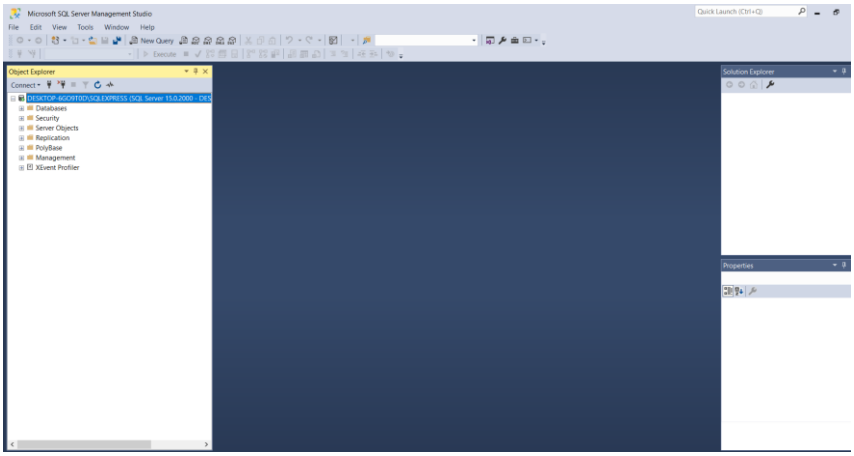
Primjer:
Napraviti bazu podataka *Polaznici_i_tecajevi* u kojoj će se nalaziti popis polaznika, popis tečajeva i popis polaznika koji su upisali određeni tečaj. Koristit će se naredbe SQL jezika sa MS SQL Server Managenent Studio programom.

Uputa:
Instalirati Microsoft SQL Server Express 2019 i Microsoft SQL Server Management Studio.
Poveznica za instaliranje programa Microsoft SQL Server Express 2019:
https://www.microsoft.com/en-us/download/details.aspx?id=101064

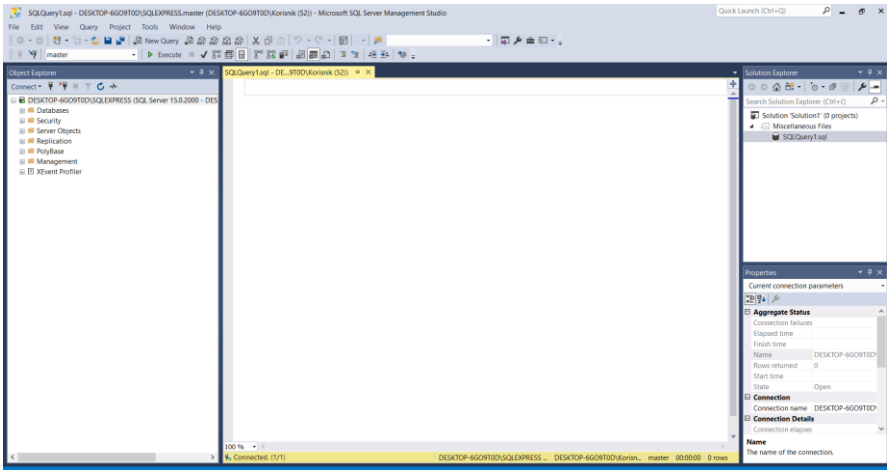
Poveznica za instaliranje programa Microsoft SQL Server Management Studio:
https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15



Program Microsoft SQL Server Management Studio je veza između korisnika i baze podataka. Nakon pokretanja potrebno je izabrati Connect.



Izabere se New Query i dobije se slijedeće sučelje.



Naredbe SQL jezika se nakon unosa pokreću tako da se odabere Execute (F5).

Nakon provedene normalizacije dobiju se slijedeće tablice:

Polaznici			
Šifra polaznika	**Ime polaznika**	**Prezime polaznika**	**Mjesto stanovanja**
1	Ana	Milić	Zagreb
2	Sanja	Tarak	Split
3	Mladen	Gork	Osijek
4	Ivana	Matkić	Split
5	Marina	Anić	Osijek
6	Ivica	Limac	Split

Tečajevi	
Šifra tečaja	**Naziv tečaja**
P01	Osnove rada PC računala
P02	Microsoft Word
N01	SQL – osnove
O01	Računalni operator – uredsko poslovanje
O02	Specijalist poslovne primjene računala
O03	Grafički dizajner

Upisi	
Šifra polaznika	**Šifra tečaja**
1	P01
2	P02
3	N01
4	O01
5	O02
6	P01

Slika 4: Tablice nakon normalizacije za bazu podataka Polaznici_i_tecajevi

ER dijagram:

Naredbe CREATE TABLE, DROP TABLE i ALTER TABLE

U sklopu SQL jezika koriste se naredbe CREATE, ALTER i DROP.

One su dio DDL (Data Definition Language) jezika koji služi za definiranje podataka.

Kreiranje baze i tablice

CREATE – kreira neki objekt u bazi (npr. tablicu)

Opći oblik za kreiranje baze podataka:

```
CREATE DATABASE ime_baze_podataka
GO
```
Npr.
```
CREATE DATABASE Polaznici_i_tecajevi
GO
```

Opći oblik za kreiranje tablice:

```
USE ime_baze _podataka
CREATE TABLE ime_tablice(
naziv stupca    tip,
…
)
GO
```

Npr.

```
USE Polaznici_i_tecajevi
CREATE TABLE polaznici (
sifra_polaznika        INT,
ime                    NCHAR(50),
prezime                NCHAR(50),
mjesto_stanovanja      NCHAR(25)
)
GO
```

Promjena postojećih objekata

ALTER – dopušta promjenu postojećeg objekta

Možemo mijenjati tablicu (dodavati, brisati, mijenjati stupce), poglede i druge objekte baze.

Npr. u tablicu Polaznici dodajemo još jedan stupac, npr. broj_telefona kao cijeli broj.

```
USE Polaznici_i_tecajevi
ALTER TABLE polaznici
        ADD broj_telefona    INT
GO
```

Možda nije najbolje broj telefona pamtiti kao cijeli broj, zgodnije ga je upamtiti kao tekst jer sadrži crtice, zagrade, pa promijenimo tip stupca u tekstualni.

```
USE Polaznici_i_tecajevi
ALTER TABLE polaznici
        ALTER COLUMN  broj_telefona       CHAR(20)
GO
```

Brisanje baze i njenih objekata

Naredbom DROP brišemo postojeći objekt iz baze, a naredbom DROP DATABASE brišemo cijelu bazu podataka.

Obrišimo stupac broj_telefona iz tablice polaznici.

```
USE Polaznici_i_tecajevi
ALTER TABLE polaznici
        DROP COLUMN  broj_telefona
GO
```

Isto tako možemo izbrisati i cijeli tablicu:

```
USE Polaznici_i_tecajevi
DROP TABLE polaznici
GO
```

Ili cijelu bazu:

```
USE master
DROP DATABASE Polaznici_i_tecajevi
GO
```

Pripazite, jer brisanje je nepovratno.

Važan dio vezan uz kreiranje baza su ograničenja (engl. constraints).
Ograničenja služe za da bi se osigurao integritet baze podataka. Prilikom kreiranja baze za neki stupac uvedemo pravila po kojima se podaci unose u taj stupac.

NOT NULL ograničenje koristi se kada želimo spriječiti unošenje NULL vrijednosti u stupac. Nema smisla dodati nekog polaznika u tablicu ako mu istovremeno ne upišemo šifru, ona mora biti odmah generirana.

```
USE Polaznici_i_tecajevi
DROP TABLE polaznici
GO

CREATE TABLE polaznici (
sifra_polaznika      INT NOT NULL,
ime                  NCHAR(50) NOT NULL,
prezime              NCHAR(50) NOT NULL,
mjesto_stanovanja    NCHAR(25)
)
GO
```

Sada nam stupci sifra_polaznika, ime i prezime imaju zabranu unosa NULL vrijednosti, tj. prilikom dodavanja novog retka u tablicu obavezno moramo unijeti sva tri polja. Mjesto_stanovanja može sadržavati i NULL vrijednost, tj. kao da se može eksplicitno napisati:

```
mjesto_stanovanja NCHAR(25)   NULL
```

Ako ništa ne napišemo, NULL je pretpostavljena vrijednost ovog ograničenja.

CHECK ograničenje koristi se kada želimo ograničiti raspon vrijednosti koji se može unijeti u neki stupac.
Npr. šifra može biti broj između 1 i 1000.

1.način:

```
USE Polaznici_i_tecajevi
DROP TABLE polaznici
GO
CREATE TABLE polaznici (
sifra_polaznika      INT, NOT NULL
ime                  NCHAR(50) NOT NULL,
```

```
prezime             NCHAR(50) NOT NULL,
mjesto_stanovanja   NCHAR(25)
CONSTRAINT chk_sifra CHECK (sifra BETWEEN 1 AND 1000)
)
GO
```

2. način:

```
USE Polaznici_i_tecajevi
DROP TABLE polaznici
GO
CREATE TABLE polaznici (
sifra_polaznika     INT  NOT NULL
                        CHECK (sifra BETWEEN 1 AND 1000),
ime                 NCHAR(50) NOT NULL,
prezime             NCHAR(50) NOT NULL,
mjesto_stanovanja   NCHAR(25)
)
GO
```

Razlika je u tome što u prvom slučaju kreiramo imenovanu instancu ograničenja koju kasnije možemo referencirati, a na drugi način dobijemo neimenovano ograničenje.

UNIQUE ograničenje koristi se kada želimo da se u navedeni stupac neka vrijednost može unijeti najviše jednom. NULL se smatra posebnom vrijednosti, pa slijedi da u tom stupcu smijemo imati najviše jednu NULL vrijednost.

PRIMARY KEY ograničenje koristi se kada želimo da dani stupac bude primarni ključ tablice. Primarni ključ jedinstveno određuje jedan redak u tablici, pa očito on mora biti UNIQUE. Ali za razliku od UNIQUE ograničenja ne možemo unijeti NULL vrijednost.

Svaka bi tablica trebala imati primarni ključ. Kako u tablici može postojati više stupaca koji jedinstveno određuju neki redak (takav stupac onda zovemo ključ kandidat), svaka njihova kombinacija može biti primarni ključ.
Primarni ključ može biti i više stupaca, što ovisi o potrebama i samom problemu. Preporuka je da primarni ključ bude što kraća, cjelobrojna vrijednost. U našem primjeru primarni ključ treba biti stupac *sifra_polaznika*.

Kreiranje tablice polaznici:

```
USE Polaznici_i_tecajevi
CREATE TABLE polaznici (
sifra_polaznika     INT  NOT NULL
                        CHECK (sifra BETWEEN 1 AND 1000)
                    PRIMARY KEY,
ime                 NCHAR(50) NOT NULL,
prezime             NCHAR(50) NOT NULL,
mjesto_stanovanja   NCHAR(25)
)
GO
```

Prilikom dodavanja novog retka u tablicu moramo unijeti jedinstvenu ne – NULL vrijednost za stupac *sifra_polaznika*.

FOREIGN KEY (strani ključ) ograničenje označava relaciju između tablica, tj. vrijednost u nekom stupcu tablice bira se iz nekog stupca druge tablice. Zapravo, strani ključ u tablici pokazuje na ključ kandidat u drugoj tablici. Ako redak sadrži strani ključ, onda na to mjesto ne možemo ubaciti vrijednost (osim NULL) koja se ne pojavljuje kao vrijednost u drugoj tablici.

Kreiranje tablice tecajevi:

```
USE Polaznici_i_tecajevi
CREATE TABLE tecajevi(
        sifra_tecaja        char(3) NOT NULL PRIMARY KEY,
        naziv_tecaja        nvarchar (50) NOT NULL
)
GO
```

Kreiranje tablice upisi:

```
USE Polaznici_i_tecajevi
CREATE TABLE upisi(
        sifra_polaznika     INT
                            FOREIGN KEY
                                REFERENCES polaznici(sifra),
        sifra_tecaja        CHAR(3)
                            FOREIGN KEY
                                REFERENCES tecajevi (sifra_tecaja),
        CONSTRAINT          PK_sifre
                                PRIMARY KEY (sifra_polaznika, sifra_tecaja)
)
GO
```

U ovoj tablici dva stupca zajedno čine primarni ključ (tj. obje šifre zajedno su kandidat za ključ), pa to moramo reći kao posebnu naredbu.

Predloženi načini vrednovanja/ ostvarivanja ishoda obrazovnog sadržaja:

- ispit od 10 pitanja s višestrukim izborom
- sudjelovanje u raspravama na forumu na Loomenu
- sudjelovanje na Zoom platformi

Sadržaj modula: PRIMJERI SVLADAVANJA IZAZOVA U STRUKOVNOJ/STRUČNOJ PRAKSI (RJEŠAVANJE PROBLEMA) UZ POMOĆ NOVIH ZNANJA, TEHNOLOGIJE I DOBRE PRAKSE U STRUCI

IMPLEMENTACIJA NOVIH ZNANJA, TEHNOLOGIJA I DOBRE PRAKSE U VLASTITU STRUKOVNU/STRUČNU I NASTAVNU PRAKSU

Korištenje baze podataka

Ishod/i učenja koji se ostvaruju kroz sadržaj:

- objasniti inovacije/novine i unapređenja u struci
- integrirati nova znanja, tehnologije i dobre prakse u vlastitu strukovnu/stručnu i nastavnu praksu i rješavanje problema
- koristiti naredbu SELECT
- postavljati upite nad više tablica

Opis obrazovnog sadržaja:

Dohvaćanje podataka – naredba SELECT

Nakon što je popunjeno nekoliko tablica, podatke iz njih nekako treba dohvatiti. Primjerice, u našoj maloj bazi zanimat će nas tko je sve upisao neki tečaj, koliko je polaznika upisalo koji tečaj, postoji li tečaj koji nitko nije upisao, je li neki polaznik upisao više od dva tečaja i slično.

Opći oblik naredbe SELECT:

```
SELECT
      [ DISTINCT ]
      [ TOP n ]
popis_stupaca
          [ INTO nova_tablica ]
FROM ime_tablice
          [ WHERE izraz_za_traženje ]
          [GROUP BY izraz_za_grupianje ]
          [ HAVING  izraz_za_traženje ]
          [ ORDER BY izraz_za_sortiranje [ ASC | DESC ] ]
```

Npr.

```
USE Polaznici_i_tecajevi
SELECT * FROM polaznici
GO
```

- za popis stupaca se ovdje koristi zvjezdica(*) što znači „svi stupci"

```
USE Polaznici_i_tecajevi
SELECT ime, prezime, mjesto_stanovanja FROM polaznici
GO
```

Ispisat će se vrijednosti za ime, prezime i mjesto_stanovanja iz tablice polaznici

```
USE Polaznici_i_tecajevi
SELECT * FROM polaznici, upisi, tecajevi
GO
```
Ispisati će se sve vrijednosti iz tri tablice, polaznici, upisi i tecajevi

```
USE Polaznici_i_tecajevi i
SELECT * FROM polaznici ORDER BY prezime
GO
```
Ispisati će se podaci sortirani po prezimenu

Sortiranje može biti uzlazno ASC ili silazno DESC
```
USE Polaznici_i_tecajevi
SELECT * FROM polaznici ORDER BY mjesto_stanovanja DESC
GO
```
Ispisati će se podaci silazno po mjestu stanovanja.

Unaprijed izabrana vrsta sortiranja je rastući redoslijed, pa se riječ ASC ne mora navoditi.

Postavljanje kriterija:
```
USE Polaznici_i_tecajevi
SELECT * FROM polaznici WHERE mjesto_stanovanja ='Split'
GO
```
Ispisati će se samo oni koji žive u Splitu.

```
USE Polaznici_i_tecajevi
SELECT * FROM polaznici WHERE mjesto_stanovanja='Split' OR 'Karlovac'
GO
```
Ispisati će se oni koji žive u Splitu i Karlovcu. Koristi se operator OR da bi se dva uvjeta povezala.

```
USE Polaznici_i_tecajevi
SELECT TOP 3 * FROM polaznici
GO
```
Ispisati će se prva 3 podatka

```
USE Polaznici_i_tecajevi
SELECT prezime FROM polaznici WHERE prezime LIKE 'M%'
GO
```
Ispisati će se polaznici kojima prezime počinje sa slovom M.

Upiti nad više tablica

Ako trebamo pogledati podatke iz više tablica, koristiti ćemo spajanje tablica.
Spajanja ima nekoliko vrsta:

- svatko sa svakom (cross-join)
- unutarnje spajanje (inner join)
- lijevo vanjsko spajanje (left outer join)
- desno vanjsko spajanje (right outer join)
- puno vanjsko spajanje (fullouter join)

Prva i najjednostavnija verzija spajanja je da spojimo svaki redak iz jedne tablice sa svakim retkom iz druge. Tako dobivamo spajanje svatko sa svakim ili cross-join.

Unutarnje spajanje (inner join) je dio SELECT izraza koji dohvaća zapise iz više tablica i daje jedan skup zapisa kao rješenje upita. Kako se kod spajanja stupaca iz različitih tablica često dogodi da dva stupca imaju isto ime, moramo ih na neki način razlikovati. Uobičajeno je da se koristi imenovanje stupaca unutar tablice: imeTablice.imeStupca. Znači, prvo se navede ime tablice, pa se stavi točka, pa se navede ime stupca. Na taj smo način jednostavno odredili na koji točno stupac mislimo prilikom pisanja upita.

Unutrašnje spajanje radi na temelju uspoređivanja vrijednosti u stupcima. Liievo vanisko spajanje (left outer join) kao rezultantni skup vraća sve retke iz lijeve tablice i odgovarajuće retke iz desne tablice. Ako neki redak iz lijeve tablice nema odgovarajući redak u desnoj, onda mu se pridružuje NULL redak (redak koji za sve vrijednosti ima NULL).

Desno vanisko spajanje (right outer join) kao rezultantni skup vraća sve retke iz desne tablice i odgovarajuće retke iz lijeve. Ako za neki redak ne postoji odgovarajući, pridružuje mu se NULL redak.

Potpuno vanjsko spajanje (full outer join) vraća sve retke iz lijeve i sve iz desne tablice tako da spoji one koji odgovaraju prema kriteriju spajanja, a one koji ne odgovaraju (i iz lijeve i iz desne tablice) dopunjava NULL vrijednostima.

Predloženi načini vrednovanja/ ostvarivanja ishoda obrazovnog sadržaja:

- ispit od 10 pitanja s višestrukim izborom
- sudjelovanje u raspravama na forumu na Loomenu
- sudjelovanje na Zoom platformi

Sadržaj modula: PRIJENOS NOVIH ZNANJA, TEHNOLOGIJA I DOBRE PRAKSE NA UČENIKE I SURADNIKE
VREDNOVANJE PRIMJENE NOVIH ZNANJA, TEHNOLOGIJA I DOBRE PRAKSE U STRUCI

Ishod/i učenja koji se ostvaruju kroz sadržaj:

- vrednovati korisnost i učinkovitost primjene novih znanja, tehnologija i dobre prakse u struci
- osmisliti prijenos novih znanja, tehnologija i dobre prakse na učenike i suradnike...
- upoznati se sa prednostima korištenja e-scenarija poučavanja i projektne nastave
- dizajnirati bazu podataka

Opis obrazovnog sadržaja:

Scenariji poučavanja su materijali u kojima su ponuđene inovativne i maštovite ideje kako provesti nastavne aktivnosti suvremenim pedagoškim metodama uz primjenu odgovarajućih digitalnih sadržaja i alata.
Naglasak je pri tome uvijek na ideji (aktivnosti), a učitelju i učenicima ostavlja se sloboda u njihovoj primjeni na različite i maštovite načine. Primjena informacijsko-komunikacijske tehnologije sastavni je dio koncepta scenarija poučavanja, no digitalni alati uvijek su pri tome svrhoviti i u funkciji ostvarivanja ishoda poučavanja nastavnog sadržaja.
Osnovni je cilj scenarija poučavanja staviti učenika u središte nastavnoga procesa i potaknuti ga na istraživanje, razmišljanje, samostalno zaključivanje i djelovanje.

Nastavnik izrađuje digitalne sadržaje. Korištenjem digitalnih obrazovnih sadržaja nastavnik ima mogućnost pripreme i organizacije nastave na fleksibilan, kreativan i inovativan način u skladu s potrebama i mogućnostima učenika koje poučava. U kombinaciji s korištenjem obrazovnih tehnologija, opreme i softvera, digitalni obrazovni sadržaji omogućuju primjenu suvremenih metoda učenja i poučavanja, učenja usmjerenog na učenika, autonomiju nastavnika u odabiru metoda i strategija za postizanje ishoda učenja i fleksibilniju strukturu nastavnog sata.
Ima nekoliko metoda kojima se može postići aktivnost učenika.
Jedna od njih je *Obrnuta učionica*, jedan od popularnijih suvremenih pristupa u obrazovanju. Obuhvaća primjenu informacijsko-komunikacijske tehnologije i različitih metoda, što omogućuje personalizaciju učenja, aktivnu primjenu znanja i razvijanje kritičkog mišljenja. Ona se provodi tako da učenici novo gradivo, koje učitelj pripremi na internetu, obrađuju kod kuće, a na satu se ono uvježbava, o njemu se raspravlja, istražuje. Dakle, učenici su aktivno uključeni u nastavni proces, samostalno istražuju i sudjeluju s drugim učenicima, dok je učitelj samo mentor koji ih u tome usmjerava.
Obrnuta nastava ima svoje prednosti i nedostatke, no važno je istaknuti da se njome poboljšava interakcija između učenika i učitelja, učenici su angažiraniji i potiče ih se na suradničko učenje.
Suradničko učenje je strategija poučavanja tijekom koje učenje proizlazi iz zajedničkog rada učenika, a podrazumijeva rješavanje složenih problemskih zadataka i rad na projektima.
Projektna nastava je složeni oblik nastave u kojoj učenici zajedno s učiteljem obrađuju neku temu kroz različite aktivnosti s ciljem postizanja točno određenog cilja u određenom vremenskom periodu.
Prednosti uporabe informacijsko-komunikacijske tehnologije i provođenja projektne nastave su mnogobrojne.

Radi se u grupama od 6 učenika.
Učenici odlaze u ustanovu za koju rade bazu, razgovaraju s zaposlenicima i prikupljaju podatke. Nakon toga sastavljaju specifikaciju potreba korisnika. Izrađuju ER model podataka i pretvaraju ga u relacijsku shemu. Implementiraju taj model na računalu, tj. izrađuju fizički model baze podataka. Unose podatke. Slijedi postupak pretraživanja podataka. Na kraju se testira baza i izrađuje dokumentacija koja je potrebna za održavanje baze podataka. Učenici koriste IKT tehnologiju.

Zadatak: Dizajnirati i kreirati bazu podataka.

Postupiti po uputama koje su u sadržajima ovog stručnog skupa.

Prijedlozi:

- Neko poduzeće sadrži nekoliko odjela u kojima zapošljava radnike. Svaki odjel ima svog šefa koji je ujedno i zaposlenik poduzeća. Kreirajte bazu podataka Zaposlenici_i_odjeli.

- Trgovina ima nekoliko skladišnih mjesta po kojima grupira svoje proizvode. Npr. cigle i crijepovi idu na jedno skladišno mjesto, drvena građa (daske, grede ...) na drugo skladišno mjesto. Svako skladišno mjesto ima nekoliko radnika koji na njemu rade i svako ima svog voditelja. Napraviti bazu podataka Skladište.

- Potrebno je osmisliti i realizirati bazu podataka Hotel koja može učinkovito poslužiti za potrebe recepcije hotela. U fazi analize, utvrđeno je da korisnik želi imati ažurnu evidenciju soba (broj sobe, broj ležajeva, tip kupaone, TV, bar,...), rezervacija, gostiju i njihovih računa (hrana, piće, hotelske usluge,...) za period boravka u hotelu. Osnovni zahtjevi korisnika su slijedeći: omogućiti uvid u trenutno slobodne i/ili zauzete sobe, prikaz stanja rezervacija i slobodnih soba.

- Potrebno je osmisliti i realizirati bazu podataka Teniski klub. U fazi analize, utvrđeno je da teniski klub raspolaže s određenom količinom tenis terena (što otvorenih, što zatvorenih).Vremenska jedinica naplate korištenja terena je jedan sat po terenu (bez obzira na broj igrača). Postoje različite tarife ovisno o dobu dana (prije podne, poslije podne, navečer), a cijena za zatvorene terene je 25% veća od one za otvorene terene. Klub raspolaže s popisom članova iz kojega je vidljivo da li je član podmirio godišnju članarinu ili ne. Oni koji nisu platili članarinu nemaju pravo rezervacije terena, oni koji su platili članarinu imaju pravo rezervirati teren 10 puta mjesečno. Osnovni zahtjevi korisnika su slijedeći: omogućiti vršenje rezervacije terena uz provjeru prava na rezervaciju, dati uvid u slobodne termine.

- Potrebno je kreirati bazu podataka Vozni_park koja će pamtiti podatke o vozilima i njihovim vlasnicima. Za svako vozilo potrebno je pamtiti broj šasije koja se uvijek sastoji od deset znakova, model automobila, godinu proizvodnje te njegovu boju koja nije obavezna za unos. O vlasnicima vozila tj. osobama potrebno je pamtiti grad u kojem žive i to njegov naziv i poštanski broj, njihovo ime, prezime, OIB koji mora biti jedinstven u tablici te datum rođenja koji nije obavezan za unos. Za svako

vozilo je potrebno pamtiti njegovog vlasnika. Jedno vozilo može imati samo jednog vlasnika dok jedna osoba može posjedovati vise vozila.

- Baza podataka za sustav Natjecanja

- Baza podataka za Knjižnicu

Svaki polaznik neka pošalje svoj primjer baze podataka koju je napravio s DBMS-om MS SQL Server.

Pokazati kako je koristio samu bazu podataka.

Slijedila bi rasprava o tim rješenjima kao i vrednovanje napravljenih baza podataka.

Predloženi načini vrednovanja/ ostvarivanja ishoda obrazovnog sadržaja:

- postavljanje na forumu primjera napravljene baze podataka
- međusobno ocjenjivanje kreiranih baza podataka
- sudjelovanje u raspravama na forumu na Loomenu
- sudjelovanje na Zoom platformi