Izrada baze podataka muzejskih eksponenata u muzeju

Pučić, Anai

Sveučilište Jurja Dobrile u Puli
Fakultet informatike u Puli

Anai Pučić

Izrada baze podataka muzejskih eksponata u muzeju

Završni rad

Pula, rujan 2024.

Sveučilište Jurja Dobrile u Puli
Fakultet informatike u Puli

Anai Pučić

Izrada baze podataka muzejskih eksponata u muzeju

Završni rad

JMBAG: 0016133778, redoviti student
Studijski smjer: Informatika
Predmet: Baze podataka 2
Znanstveno područje: Društvene znanosti
Znanstveno polje: Informacijske i komunikacijske znanosti
Znanstvena grana: Informacijski sustavi i informatologija

Mentor: izv.prof.dr.sc. Goran Oreški
Komentor: mag.inf. Romeo Šajina

Pula, rujan 2024.

Sažetak

Predmet ovog rada je izrada baze podataka za muzej čija je osnovna namjena omogućiti neometan proces rada u muzeju, počevši od pružanja informacija posjetiteljima o vrsti muzeja, eksponatima, odjelima, događajima, realizaciji posjeta i naplati ulaznica, te osiguravanje podataka za različite vrste analiza. U sklopu sustava uvedene su provjere kako bi se osigurala zaštita podataka i integritet poslovnih procesa. Na primjer, napravljena je provjera koja osigurava da samo određeni zaposlenici mogu prodavati karte, može li zaposlenik stornirati karte, time ograničavajući broj storniranja na dnevnoj razini. Uvedeni su mehanizmi koji sprječavaju storniranje ili izmjenu finaliziranih računa, provjere prilikom obračuna popusta na ulaznice i dodjeljivanja eksponata događajima.

Projekt je dostupan na: https://github.com/anaip30/baza_podataka_muzej.git

Ključne riječi: baza podataka, muzeji, eksponati, eksponat, muzej, baza podataka za muzej, baza podataka muzejskih eksponata u muzeju

Abstract

The subject of this bachelor's thesis is the development of a database for a museum, aimed at ensuring a smooth operational process within the museum. This includes providing information to visitors about the type of museum, exhibits, departments, events, visit, facilitation, ticket sales, and ensuring data for varios types of analysis. The system has implemented checks to ensure data protection and the integrity of business processes. For instance, a verification mechanism has been established to ensure that only certain employees can sell tickets, and whether an employee can cancel tickets, with a daily limit on cancellations, Mechanisms have been introduced to prevent the cancellation or modification of finalized invoices, along with checks for applying discounts on tickets and assigning exhibits to events.

Keywords: database, museums, exhibits, exhibit, museum, database for museums, database of museum exhibits in a museum

Sadržaj

1. Uvod

U ovom završnom radu govorit će se o izradi baze podataka eksponenata u muzeju. Cilj je poboljšati upravljanje i prezentiranje podataka u muzejima, sam način upravljanja informacija za kupovanje karata i prezentiranje događaja. Glavni cilj rada je kreiranje baze koji omogućava efikasno i jednostavno upravljanje informacija o eksponentima, to uključuje njihov, opis, povijest, umjetnika, materijal eksponata i naravno o samim muzejima uključujući njihovo radno vrijeme, lokaciju, vrsti, o odjelu, zaposlenicima i posjetiteljima, događajima, recenzijama događaja i prodaji karata za događaje.

U samom radu koriste se metode dizajna i implementacije za bazu podataka. Tijek procesa izrade započelo se analizom potreba i zahtjeva korisnika. Napravio se je dizajn konceptualnog modela pomoću ER dijagrama, te logičkog modela kroz relacijske modele i za fizičku implementaciju baze podataka koristi se baza MySQL-a za prikaz EER (Enhanced Entity-Relationship) dijagrama. Baza MySQL je namijenjena izradi i upravljanjem baza podataka. Sama izrada baze podataka uključuje kreiranje relacije, definiranje veza između relacija i unos podataka u relacija.

Za izradu baze podataka za muzej, posvetilo se pažnji zaštiti podataka i dizajnu baze podataka za muzej sa kojom se omogućila manipulacija podataka za posjetitelje i zaposlenike.

Željene rezultate u ovom radu uključuju funkcionalnu bazu podataka koja omogućava muzejskim zaposlenicima upravljanje zbirkama i prodajom karata. Posjetiteljima se pruža bogatiji uvid u muzejske eksponate i događaje.

1.1. Ciljevi rada

Glavni cilj rada je dizajniranje i implementiranje baze podataka koja će zadovoljiti potrebe muzeja, a to je upravljanje eksponata, događaja i prodaja karata.

Prvi cilj je analiza potrebe korisnika. Identificiraju se ključni zahtjevi korisnika baze podataka, koje uključuju zaposlenike i posjetitelje. Drugi cilj je sam dizajn baze podataka. Razvit će se konceptualni i logički model baze podataka koji će organizirati informacije o eksponatima, muzejima i događajima. Potom treći cilj je implementacija fizičke relacije baze podataka. Koristit će se odgovarajuća baza za upravljanje bazama podataka. Dalje četvrti cilj je unos i održavanje podataka. Razvija se metoda za unos, ažuriranje i validaciju podataka. Cilj je osigurati točnost i povezanost informacija. Peti cilj je testiranje i evaluacija. Provodi se testiranje kako bi se osigurala operativnost i performans baze podataka. Evaluirati će se sustav na temelju povratnih informacija korisnika.

2. Pregled literature

2.1. Kratak pregled postojeće literature i istraživanja vezanih uz baze podataka u muzeju:

Postojeću literaturu o bazama podataka u muzeju koja je izabrana pokriva veliko područje tema. Uključujući razvoj i implementaciju baze, upravljanje muzejom i muzejskim zbirkama i prikaz digitalizaciju za muzeje. Ovo su neka od istraživanja i radova koje su izabrani:

Marty [3] u svom radu navodi način na koji muzeji koriste tehnologiju za upravljanje zbirkama i interakciju s publikom. Rad je pružio detaljan pregled razvoja informacijskih sustava za muzej. Rad naglašava važnost standardizacije podataka i sposobnost između različitih sustava. Iako je rad napisan 2005. godine, muzeji se i dan danas bave sa istim problemima. Ti problemi su u digitalizaciji podataka, točnost podataka i gubitak autorska prava.

Istraživanje od Addison [4] prikazuje korištenje sustava za očuvanje i prezentaciju kulturne baštine. Istraživanje prikazuje izazove s kojima se muzeji suočavaju pri spajanju tradicionalnih i digitalnih podataka i nudi rješenja za uspješnu implementaciju tih tehnologija.

3. Analiza potreba i zahtjeva

3.1. Opis poslovnog procesa

Baza podataka koja je napravljena za muzej je kao pametni sustav koji pomaže muzeju u boljem vođenju i organizaciji svih važnih informacija. Ovaj sustav ima jednostavan pregled svih podataka koji su povezani s muzejom, njegovim eksponatima, zaposlenicima, posjetiteljima, događajima i prodaji karata. Svatko tko radi u muzeju lako pronalazi informacije. U bazi su pohranjeni podaci o lokaciji muzeja, njegovom radnom vremenu, vrsti muzeja (na primjer povijesni, umjetnički i tako dalje), o samom muzeju, zaposlenicima muzeja, posjetiteljima, svim događajima koji se održavaju u muzeju ,recenzijama događaja, eksponatima i njihovim vrstama i o umjetnicima koji su izradili te eksponate. Evidentirano je koji eksponati su izloženi na određenim događajima i u kojim prostorijama muzeja. Funkcionalnosti sustava uključuju automatsko izračunavanje cijene računa za ulaznice, uključujući primjenu popusta i mogućnost storniranja računa u slučaju greške. Muzej može provjeriti koji eksponati će biti izloženi na određenom događaju, a sustav sprječava dvostruko izlaganje eksponata. Obračun plaća zaposlenika je automatiziran, a sustav prepoznaje koji događaji su najpopularniji prema recenzijama posjetitelja. Provjerava se koji zaposlenici imaju pravo prodavati karte. Sustav omogućava izvještavanje i analizu podataka. Može se vidjeti koliko je karata prodano, koji događaji su najbolje ocijenjeni ili koliko je posjetitelja posjetilo određeni muzej. Korištenjem

različitih provjera sustav osigurava da se podaci unose ispravno i da samo ovlašteni korisnici imaju pristup osjetljivim informacijama tako se osigurava sigurnost podataka.

3.2. Identifikacija ključnih korisnika baze podataka

Da bude baza podataka korisna i učinkovita, moraju se identificirati korisnici i njihove potrebe. Odredili su se korisnici koje su važni za izradu baze podataka u muzeju, a to su zaposlenici i posjetitelji.

Svaki zaposlenik ima svoje radno mjesto. Radna mjesta su: čistačica, zaštitar, znanstvenik, vodič i prodavač karata. Najveći fokus je na radno mjesto prodavač karata. Opisuje se kako prodavač karata upravlja prodajom karata i koje dozvole ima u smislu korištenja funkcionalnosti baze.

Prodavač karata upravlja prodajom karata, izdaje karte posjetiteljima muzeja, unosi potrebne podatke u bazu. Prodavač ima opciju birati između tri vrste karata, a to su: grupne, VIP ili obične. Prodavač može provjeriti raspoloživost karata za događaje i vremenske termine. Pohranjena procedura omogućava primjenu posebnih popusta na karte koje su unaprijed definirane. Na primjer za prodaju grupnih karata (30 karata) postoji mogućnost ostvarivanje popusta od 15%, a ako je posjetitelj kupio više od 15 karata ostvaruje pravo na popust od 10%. Na kraju radnog dana ili tijekom radnog dana, prodavač može izlistati izvještaj o ukupnom broju prodanih karata i ukupnom prihodu. U slučaju greške ili na zahtjev posjetitelja, prodavač može stornirati kartu i automatski ažurirati stanje u bazi podataka. Sve stornirane karte bilježe se u bazi prema vremenu storniranja. Status računa se u tom trenutku mijenja iz ‘T’ u ‘F’. Na takav način se vodi evidencija storniranih karata, bez brisanja iz baze. Baza ima ograničen broj storniranja karata po prodavaču. Postavljeno je u ograničenju na mogućnost storniranja 10 karata dnevno. Prema broju storniranih karata po prodavaču moguće je usporedba ostvarenih prihoda, te usporedba prodavača na radnom mjestu. Prodavač može unositi osnovne podatke o posjetiteljima, a to su ime, prezime, broj telefona i jezik. Ova funkcionalnost je korisna jer omogućava izdavanje personaliziranih karata i evidencija posjetitelja. Procedure i ograničenja osiguravaju da samo prodavač može izdavati i stornirati račune, te prodavati karte, a sve radnje se moraju evidentirati radi sigurnosti podataka i osobe.

Svaki posjetitelj mora pri kupnji karata dati svoje osobne podatke prodavaču. Prodavač upisuje potrebne podatke o posjetitelju, kao što su ime, prezime, broj telefona i jezik. Tijekom posjeta muzeju posjetitelj uz pomoć funkcionalnosti baze preko pogleda i upita može pristupiti informacijama pohranjeni u bazi o eksponatima i umjetnicima. Posjetitelji mogu davati povratne informacije i ocjenjivati izložbe, događaje i usluge muzeja. Informacije se spremaju u bazu podataka u cilju boljeg razumijevanju potreba posjetitelja i unaprjeđenju muzejskih usluga.

3.3. Funkcionalni i nefunkcionalni zahtjevi

Prikazati će se zahtjevi baze koji se mogu podijeliti u funkcionalne i nefunkcionalne zahtjeve. Ian [6] piše da funkcionalni zahtjevi opisuju što bi baza trebala raditi. Opisivati ću sljedeće funkcionalne zahtjeve: izračunavanje iznosa računa, storniranje računa, provjera eksponata na događaju, obračun plaća zaposlenika, identifikacija popularnih događaja, provjera mogućnosti prodaje karata, izvještavanje i analitika, kontrola integriteta podataka i ograničenja. Sve ove funkcionalnosti su važne za muzej i zaposlenike. Nefunkcionalni zahtjevi omogućuju prikaz kako će baza raditi, to je objasnio Ian [6]. Govorit će se o nefunkcionalnom zahtjevu sigurnost i osiguravanje integriteta podataka baze podataka.

3.3.1. Funkcionalni zahtjevi

Izradom funkcionalnih zahtjeva u bazi podataka za upravljanje muzejskim poslovanjem osigurano je točnost podataka i omogućila pravilno izvršavanje poslova. Svaka se funkcionalnost izradila na način da osigura sigurnost sustava s ciljem sprječavanja potencijalnih grešaka i zloupotreba. U nastavku prikazujem opis postupka izrade i implementacije takvih funkcionalnih zahtjeva.

Izračunavanje iznosa računa

Za funkcionalnost izračunavanja napravljen je sustav koji računa ukupan iznos računa. Koristi se pristup gdje se prolazi kroz sve stavke računa. Izračunava se ukupni iznos na temelju podataka o cijeni i količini. Uveli su se posebni uvjeti, poput popusta od 15% u slučaju kupnje grupne karte u većoj količini i dodatni popust od 10% za kupnje od 15 ili više običnih karata. Na kraju sustav automatski ažurira ukupan iznos.

Storniranje računa

Napravljen je sustav koji provjerava ukupan broj prethodno storniranih računa po svakom zaposleniku na taj dan. Ako je broj storno računa veći od dozvoljenog ograničenja koji je definiran u sustavu, postupak storniranja se zaustavlja i sustav javlja grešku. Na takav način je osigurano da zaposlenik ne može zloupotrijebiti mogućnost storniranja računa iznad zadanog ograničenja. Ako su svi uvjeti u sustavu ispunjeni, sustav omogućuje storniranje računa i ažurirati će status u računu da je račun storniran. Na takav način se prati koji su računi stornirani.

Provjera eksponata na događaju

Napravljen je sustav koji provjerava da eksponat ne bude dodijeljen u više od jednog događaja u istom vremenskom razdoblju. Sustav provjerava da li je određeni eksponat već dodijeljen događaju u istom razdoblju. Ako se otkrije preklapanje, postupak se zaustavlja i javlja se greška. Na takav način je osigurano da eksponat ne bude na dva mjesta postavljen na različitim događajima u isto vrijeme.

Obračun plaća zaposlenika

Automatiziran je obračun plaća zaposlenika koji se provodi putem sustava za izračun plaća na temelju godina radnog staža. Sustav prolazi kroz sve zaposlenike, provjerava njihov radni staž i automatski dodjeljuje povišicu ako zaposlenik ima 5 ili više godina radnog staža, pri čemu povišica iznosi 10%. Nakon izračuna, sustav ažurira plaću zaposlenika u bazi podataka. Ovim procesom osiguralo se je pravedan i pravovremen obračun plaća.

Identifikacija popularnih događaja

Za pronalaženje najpopularnijih događaja napravljen je sustav koji koristi broj recenzija kao mjerilo popularnosti. Sustav vraća naziv događaja s najvećim brojem recenzija, što omogućuje muzeju da analizira svoje događaje i na temelju toga organizira buduće, kvalitetnije događaje.

Provjera mogućnosti prodaje karata

Napravljen je sustav koji provjerava ima li zaposlenik pravo prodavati karte. Ako zaposlenik ima radno mjesto "Prodavač karata" i povezan je s muzejom, moći će prodavati karte. Sustav osigurava da samo zaposlenici s odgovarajućim radnim mjestom mogu obavljati prodaju karata, čime se sprječava zloupotreba.

Izvještavanje i analitika

Za izvještavanje i analitiku napravljen je putem sustava koji omogućava pregled podataka. Sustav prikazuje najvažnije informacije o muzejima, prosječnim ocjenama događaja, posjetiteljima koji su kupili (‘VIP’) ulaznice, radnom vremenu muzeja, posjetiteljima i njihovim ulaznicama, zaposlenicima s određenim radnim mjestima, pregledima svih eksponata i umjetnika, ocjenama i komentarima izložbi i prikazuje ukupnu prodaju, troškove plaća i profit po godinama. Ovaj sustav omogućuje zaposlenicima muzeja jednostavan pristup važnim informacijama i tako mogu donosit odluke temeljene na podacima.

Kontrola integriteta podataka i ograničenja

Kontrolu potpunosti podataka je osigurana putem sustava koji sprječava neovlaštene promjene u bazi podataka. Napravljen je sustav koji sprječava umetanje, ažuriranje ili brisanje podataka. Ove kontrole sprječavaju umetanje podataka u račune koji su već finalizirani, ažuriranje finaliziranih računa ako određeni uvjeti nisu ispunjeni i brisanje finaliziranih računa. Sprječava se promjena statusa računa iz jednog stanja u drugo ako to nije dozvoljeno i ograničava se broj storniranih računa po zaposleniku na dnevnoj bazi. Napravljena je provjera koja sprječava umetanje eksponata ako je već povezan s nekim događajem, kao i kontrolu unosa podataka o zaposlenicima kako bi se osiguralo da samo zaposlenici s određenim radnim mjestima mogu obavljati određene radnje. Slične kontrole primjenjeno je na unose i ažuriranja stavki računa, sprječavajući dodavanje ili izmjenu podataka u slučajevima kada je račun već finaliziran. Ove provjere napravljene su s ciljem zaštite podataka, sprječavanja pogrešaka i zloupotrebe.

3.3.2. Nefunkcionalni zahtjevi

Za nefunkcionalne zahtjeve koriste se provjere na razini relacija, provjere točnosti podataka i ovlaštenja za korisnike. Uz pomoć tih provjera osigurava se da ispravni podaci budu pohranjeni u bazi podataka i samo da korisnici koji su ovlašteni mogu pristupiti osjetljivim podacima i izvršavati određene operacije. U nastavku se prikazuju te provjere.

Osiguravanje integriteta podataka

Foreign Keys (Strani ključevi):
Elmasri i Navathe [9] pišu o vanjskim ključevim (Foreign Keys) koji su postavljeni kako bi se osiguralo povezivanje između različitih relacija. Omogućava da vrijednosti u jednoj tablici odgovaraju vrijednostima u drugoj tablici, čime se osigurava pravilno povezivanje podataka i sprječavaju nelogičnosti u bazi podataka.

UNIQUE ograničenje:
Connolly i Begg [7] objašnjavaju UNIQUE ograničenje koje se koristi kako bi se osiguralo da stupci sadrže jedinstvene vrijednosti. Ovim ograničenjem sprječava se dupliciranje podataka, ključno je za održavanje jedinstvenih oznaka unutar baze. Time se sprječava dupliciranje podataka, što je jako važno za održavanje jedinstvenih oznaka, olakšava se pretraživanje i manipulaciju podataka.

NOT NULL ograničenje:

Connolly i Begg [7] objašnjavaju kako se ograničenja na razini relacija postavljaju kako bi spriječila pogreške u podacima i osigurala da svi podaci koji ulaze u bazu podataka zadovoljavaju određene uvjete. NOT NULL ograničenje omogućuje da određeni atribut ne ostane prazan u relaciji, što je ključno za potpunost podataka i olakšava pretraživanje unutar baze podataka.

CHECK ograničenje:

Silberschatz, Korth i Sudarshan [8] objašnjavaju kako CHECK ograničenje omogućuje postavljanje dodatnih uvjeta na vrijednosti unutar stupaca. Ova provjera osigurava da se u bazu unose samo ispravni podaci, sprječavajući unos nepravilnih ili logički neispravnih vrijednosti, čime se osigurava konzistentnost i ispravnost podataka u sustavu.

Napredne provjere:

Napredne provjere su provedene putem sustava koji osigurava dodatnu razinu validaciju podataka. Postavila sam kontrole koje provjeravaju ažuriranje, brisanje i dodavanje karata za korisnika te osiguravaju da on ima odgovarajuće ovlasti za te radnje. Implementirala sam sustav koji izračunava iznose na temelju postavljenih uvjeta, poput broja prodatih karata i datuma događaja.

Sigurnost:

Sigurnost sustava osiguran je postavljanjem prava pristupa i kontrolom nad operacijama koje različiti korisnici mogu izvršavati. Prikazat će se zaštita podataka koja se provodi putem različitih ovlaštenja.

Razine ovlaštenja:

Obični korisnici (na primjer prodavači karata):

Ovim korisnicima postavljena su pravila izvršavanja DML operacija ('INSERT', 'UPDATE', 'DELETE') nad određenim relacijama u bazi podataka. Omogućen im je rad s podacima o kartama, posjetiteljima i događajima. Imaju pravo na manipulaciju podacima koji su relevantni za njihovo svakodnevno poslovanje, ali nemaju pristup osjetljivim podacima ili administracijskim operacijama.

DB Admin korisnici:

Ovim korisnicima postavilo se je puno administracijskih prava nad bazom podataka. Imaju pravo na izradu i mijenjanje relacija, postavljanje novih ograničenja i kontrolu nad korisničkim ovlaštenjima. Samo administratori mogu izvršavati promjene u strukturi baze.

4. Modeliranje podataka

Modeliranje podataka proces je koji počinje analiziranjem zahtjeva na informacijski sustav, a završava izgradnjom baze podataka [6].

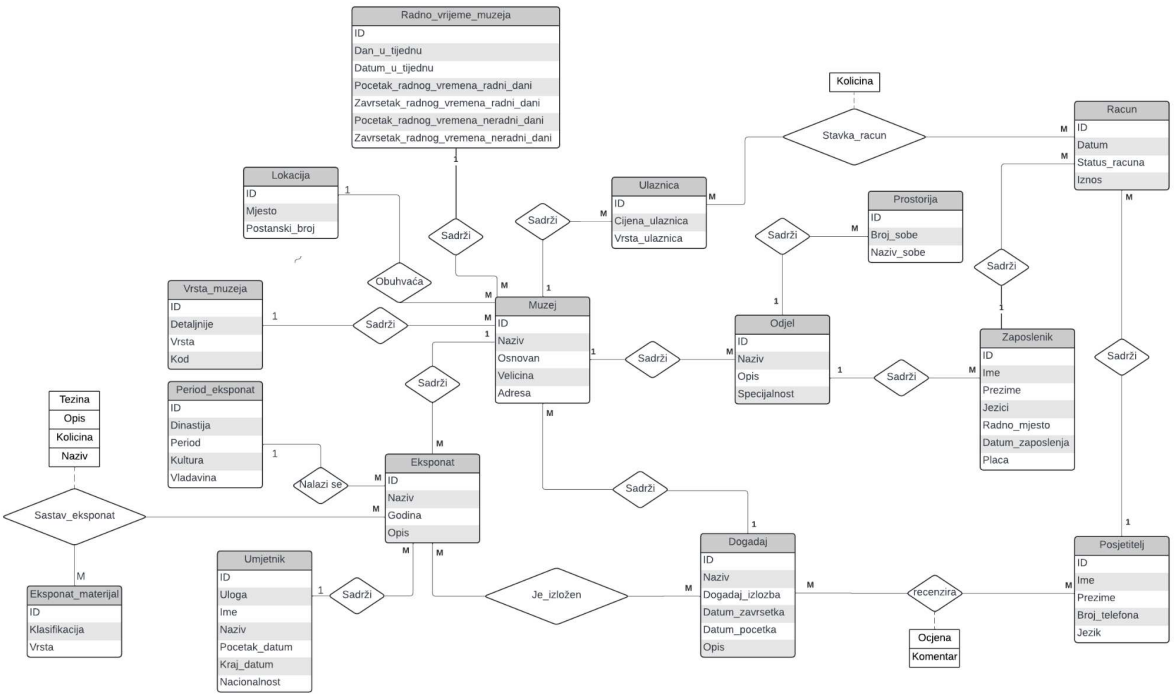
U modeliranim podacima prikazat će se konceptualni model koji predstavlja prvu fazu u dizajniranja baze podataka. Prikazati će se entitete i veze između njih. Nakon toga će se prikazati logički model koji je druga faza u dizajniranju baze podataka. Tu će se prikazati kako će baza izgledati u fizičkom dizajnu. Onda će se prikazati zadnju fazu, a to je fizičko modeliranje. Tu se izrađuje baza u MySQL-u i prikazuje uz pomoć MySQL Workbench dijagram relacija.

4.1. Konceptualni dizajn baze podataka

Konceptualni model podataka opisuje strukturu podataka informacijskog sustava i predstavlja ključ razumijevanja informacijskog resursa organizacije. Konceptualno modeliranje polazi od specifikacija informacijskih zahtjeva, koje čine zahtjevi na strukturu podataka i zahtjeva za korištenjem podataka, a rezultira izrađenim konceptualnim modelom podataka [6].

Za prikaz konceptualnog modela koristi se ER(Entity-Relationship) dijagram. ER (Entity-Relationship) vrsta je dijagrama koja ilustrira kako se "entiteti" kao što su ljudi, objekti ili koncepti međusobno povezuju unutar sustava [2].

Slika 1 prikazuje obrađenu bazu podataka na konceptualnom nivou s prikazom entiteta, veza i atributa. Kad je izrađivan konceptualni dizajn donosilo su se odluke o tome koje atribute sačuvati u bazi podataka i kako ih organizirati u različite tablice.



Slika 1: ER dijagram za bazu podataka muzej

Iz opisa poslovnih procesa (sekcija) identificirano je 15 entiteta i njihovih atributa:

- Radno_vrijeme_muzeja - ID, Dan_u_tijednu, Datum_u_tijednu, Pocetak_radnog_vremena_radni_dani, Zavrsetak_radnog_vremena_radni_dani, Pocetak_radnog_vremena_neradni_dani, Zavrsetak_radnog_vremena_neradni_dani
- Lokacija - ID, Mjesto , Postanski_broj,
- Vrsta_muzeja - ID, Detaljnije, Vrsta, Kod
- Muzej - ID, Naziv, Osnovan, Velicina, Adresa
- Period_eksponat - ID, Dinastija, Period, Kultura, Vladavina
- Eksponat_materijal - ID, Klasifikacija, Vrsta
- Umjetnik - ID, Uloga, Ime, Naziv, Pocetak_datuma, Kraj_datuma, Nacionalnost
- Eksponat - ID, Naziv, Godina, Opis
- Ulaznica - ID, Cijena_ulaznica, Vrsta_ulaznica
- Racun - ID, Datum, Status_racuna, Iznos
- Zaposlenik – ID , Ime, Prezime, Jezici, Radno_mjesto, Datum_zaposlenja, Placa
- Posjetitelj - ID, Ime, Prezime, Broj_telefona, Jezik
- Dogadaj - ID, Naziv, Dogadaj_izlozba, Dogadaj_pocetak, Dogadaj_zavrsetak
- Odjel - ID, Naziv, Opis, Specijalnost
- Prostorija - ID, Broj_sobe, Naziv_sobe

Prikazani su entitete koji će biti u bazi podataka. Te sada će se prikazati njihove veze:
Svaki muzej može imati jedno radno vrijeme, ali radno vrijeme može biti primijenjeno na više muzeja. Lokacije mogu obuhvatiti više muzeja, dok se svaki muzej nalazi na jednoj specifičnoj lokaciji. Vrsta muzeja može se povezati s više muzeja i svaki muzej može imati jednu vrstu. Eksponati se povezuju s određenim periodom, dok jedan period može obuhvatiti više eksponata. Jedan eksponat može sadržavati različite materijale, a svaki materijal može biti povezan s više eksponata. Ulaznica može biti vezana uz više muzeja, a svaki muzej može prodavati jednu ulaznicu po posjetitelju. Račun može uključivati više posjetitelja, ali svaki posjetitelj može imati jedan račun za ulaznice. Posjetitelji mogu ostavljati više recenzija za različite događaje, događaj može primiti više recenzija od istog posjetitelja. Muzeji mogu organizirati više događaja, ali svaki događaj pripada jednom muzeju. Svaki muzej ima jedan odjel, dok odjeli mogu sadržavati više zaposlenika. Zaposlenici su smješteni u jedan odjel, a odjeli mogu imati više prostorija, dok je svaka prostorija dio jednog odjela. Zaposlenici mogu izdavati više računa, ali svaki račun izdaje jedan određeni zaposlenik. Događaji mogu uključivati više eksponata, eksponati mogu biti izloženi na više događaja. Umjetnici mogu biti povezani s više eksponata, ali svaki eksponat pripada jednom umjetniku.

4.2. Pretvorba konceptualnog modela u relacijski model

Nakon što je završeno sa konceptualnim dizajnom baze podataka putem ER dijagrama, slijedi proces pretvaranje modela u relacijski (logički) model. Relacijski model definira strukturu baze podataka u obliku relacija, povezane su vanjskim ključevima i sadrže atribute.

4.2.3. Kombinacija shema

Kombinacija shema je proces koji povezuje više tablica koje dolaze iz ER dijagrama u relacijski model. Cilj je provjeriti sve entitete i njihove veze, osigurati da su entiteti I atributi ispravno pretvoreni u relacije.

Pretvara se svaki entitet u relaciju, pri čemu se za svaki entitet postavlja primarni ključ. Potom se provjerava da su veze (relacije) između entiteta pravilno pretvorene u relacijske tablice. To se radi dodjeljivanjem vanjskih ključeva. Kada veza sadržava više na više vezu (M:M) veza se pretvara u relaciju. Kombinacija shema također podrazumijeva pregledavanje atributa kako bi se spriječila redundancija podataka. Definiraju se atributi za svaku tablicu, pri čemu se provjerava da atributi sadržavaju jednostavne, nedjeljive vrijednosti.

Pretvorba u relacijski model:

- Radno_vrijeme_muzeja (ID, Dan_u_tijednu, Datum_u_tijednu, Pocetak_radnog_vremena_radni_dani, Zavrsetak_radnog_vremena_radni_dani, Pocetak_radnog_vremena_neradni_dani, Zavrsetak_radnog_vremena_neradni_dani)
- Sadržaj_Radno_vrijeme_muzeja (ID_ Radno_vrijeme_muzeja, ID_Muzej)
- Lokacija (ID, Mjesto , Postanski_broj)
- Obuhvaća (ID_lokacija, ID_Muzej)
- Vrsta_muzeja (ID, Detaljnije, Vrsta, Kod)
- Sadrži_Vrsta_muzeja (ID_Vrsta_muzeja, ID_Muzej)
- Muzej (ID, Naziv, Osnovan, Velicina, Adresa)
- Period_eksponat (ID, Dinastija, Period, Kultura, Vladavina)
- Nalazi_se (ID_ Period_eksponat, ID_ Eksponat)
- Eksponat_materijal (ID, Klasifikacija, Vrsta)
- Sastav_eksponat (ID_ Eksponat, ID_ Eksponat_materijal, Tezina, Opis, Kolicina, Naziv)
- Umjetnik (ID, Uloga, Ime, Naziv, Pocetak_datuma, Kraj_datuma, Nacionalnost)
- Sadrži_Umjetnik (ID_ Eksponat, ID_Umjetnik)
- Eksponat (ID, Naziv, Godina, Opis)
- Sadrži_Eksponat (ID_ Eksponat, ID_Muzej)
- Ulaznica (ID, Cijena_ulaznica, Vrsta_ulaznica)
- Sadrži_Ulaznica (ID_Muzej, ID_Ulaznica)
- Racun (ID, Datum, Status_racuna, Iznos)
- Stavka_racun (ID_ Ulaznica, ID_ Racun, Kolicina)
- Zaposlenik (ID , Ime, Prezime, Jezici, Radno_mjesto, Datum_zaposlenja, Placa)
- Zaposlenik_Sadrži_Racun (ID_ Zaposlenik, ID_ Racun)
- Zaposlenik_Sadrži_Odjel (ID_ Zaposlenik, ID_Odjel)
- Posjetitelj (ID, Ime, Prezime, Broj_telefona, Jezik)
- Posjetitelj_sadrži_Racun (ID_ Posjetitelj, ID_ Racun)
- Recenzira (ID_ Posjetitelj, ID_ Dogadaj, Ocjena, Komentar)
- Dogadaj (ID, Naziv, Dogadaj_izlozba, Dogadaj_pocetak, Dogadaj_zavrsetak)
- Je_izlozen (ID_ Dogadaj, ID_ Eksponat)
- Odjel (ID, Naziv, Opis, Specijalnost)
- Sadrži_Odjel (ID_ Odjel, ID_Muzej)
- Prostorija (ID, Broj_sobe, Naziv_sobe)
- Sadrži_Prostorija (ID_ Odjel, ID_ Prostorija)

Nakon što se primjenila kombinacija shema proizlaze sljedeće relacije:

Veze u ER dijagramu koje imaju više na više vezu (M:M) pretvara se u relaciju. To su veze ‘Sastav_eksponat’ koju pretvara u relaciju ‘Sastav_eksponat’, veza ‘Je_izložen’ pretvara u relaciju ‘Eksponat_na_dogadaj’, veza ‘Recenzija’ u relaciju ‘Recenzija_dogadaj’ i veza ‘Stavka_racun’ u relaciju ‘Stavka_racun’.

- Muzej (ID_muzej, ID_odjel, ID_ulaznica, ID_radno_vrijeme_muzeja, ID_lokacija, ID_vrsta_muzeja, Naziv, Osnovan, Velicina, Adresa)
- Radno_vrijeme_muzeja (ID_radno_vrijeme_muzeja, Dan_u_tijednu,Datum_u_tijednu, Pocetak_radnog_vremena_radni_dani, Zavrsetak_radnog_vremena_radni_dani, Pocetak_radnog_vremena_neradni_dani, Pocetak_radnog_vremena_neradni_dani)
- Lokacija (ID_lokacija, Mjesto, Postanski_broj)
- Vrsta_muzeja (ID_vrsta_muzeja, Detaljnije, Vrsta, Kod)
- Period_eksponat (ID_period_eksponat, Dinastija, Period, Kultura, Vladavina)
- Umjetnik (ID_umjetnik, Uloga, Ime, Naziv, Pocetak_datuma, Kraj_datuma, Nacionalnost)
- Eksponat_materijal (ID_eksponat_materijal, Klasifikacija, Vrsta)
- Sastav_eksponat (ID_sastav_eksponat, ID_eksponat_materijal, Tezina, Opis, Kolicina, Naziv)
- Eksponat (ID_eksponat, ID_sastav_eksponat, ID_umjetnik, ID_muzej, Naziv, Godina, Opis)
- Eksponat_na_dogadaj (ID_eksponat_na_dogadaju, ID_Dogadaj, ID_eksponat)
- Dogadaj (ID_dogadaj, Naziv, Dogadaj_izlozba, Datum_zavrsetka, Datum_pocetka, Opis)
- Recenzija_dogadaj (ID_recenzija_dogadaj, ID_Dogadaj, ID_posjetitelj, Ocjena, Komentar)
- Posjetitelj (ID_posjetitelj, Ime, Prezime, Broj_telefona, Jezik)
- Račun (ID_racun, ID_posjetitelj, ID_zaposlenik, Datum, Status_racuna, Iznos)
- Zaposlenik (ID_Zaposlenik, Ime, Prezime, Jezici, Radno_mjesto, Datum_zaposlenja, Placa)
- Prostorija (ID_prostorija, Broj_sobe, Naziv_sobe)
- Odjel (ID_odjel, Naziv, Opis, Specijalnost)
- Stavka_racun (ID_stavka_racun, ID_racun, ID_ulaznica, Kolicina)
- Ulaznica (ID_ulaznica, Cijena_ulaznica, Vrsta_ulaznica)

Na kraju pretvaranje ER dijagrama u relacijski model nastale su 18 relacija.

4.3. Analiza relacijskog modela

Nakon što je stvoren relacijski model, provjerava i osigurava se da relacije zadovoljavaju pravila normalnih formi (1NF, 2NF, 3NF) kako bi se izbjegla redundancija i osigurala konzistentnost podataka u bazi. Postupak normalizacije uključuje razdvajanje podataka u relacije. Postoje tri normalne forme: 1NF, 2NF i 3NF.

1NF:

Abraham [10] govori o 1NF (Prva normalna forma). 1NF zahtijeva da svaki stupac tablice mora sadržavati nedjeljive vrijednosti, to znači da unutar jednog stupca ne smiju postojati skupovi, nizovi ili popisi. Svaki stupac također mora sadržavati vrijednosti jednog tipa podataka. 1NF osigurava da su podaci strukturirani u tabelarnom formatu gdje je svaki unos u tablici jedinstven i svako polje sadrži samo jednu informaciju.

Baza podataka muzeja koju je napravljena je već u prvoj normalnoj formi (1NF). To znači da su svi podaci unutar tablica organizirani na način da svaka tablica zadovoljava osnovna pravila relacijskih baza podataka. Svaki atribut sadrži jednostavne i nedjeljive vrijednosti. Na primjer relacija "Eksponat", atributi "Naziv", "Godina" i "Opis" su nedjeljivi. Atribut "Naziv" sadrži samo naziv eksponata, a ne kombinaciju više podataka (na primjer naziv i opis u istom polju). Svaka tablica ima primarni ključ koji osigurava da nema dupliciranje redaka. Baza podatka je zbog toga lakša za pretraživanje, manipulaciju podataka i omogućava bolju organizaciju podataka.

2NF :

Abraham, Henry i Sudarshan [10] govore o 2NF (druga normalna forma). 2NF se nadograđuje na 1NF osiguravajući da svi atributi koji nisu ključevi potpuno zavisni o primarnom ključu. To znači da ne bi trebalo postojati djelomična ovisnost bilo kojeg stupca samo o djelu složenog primarnog ključa. Ako relacije sadrži složeni ključ, svaki stupac bez ključa mora ovisiti o cijelom primarnom ključu, a ne samo o njegovom podskupu.

Baza podataka muzeja također zadovoljava zahtjeve 2NF. Primjer iz baze je relacije "Stavka_Racun", koja je u 2NF. U ovoj relaciji imam složeni primarni ključ koji se sastoji od dva atributa: "ID_Racun" i "ID_Ulaznica".Atribut "Kolicina" ovisi isključivo o cijeloj kombinaciji "ID_Racun" i "ID_Ulaznica" primarnog ključa, a ne samo o jednom od njih. Na takav način se izbjegavaju djelomične zavisnosti.

3NF:

Abraham, Henry i Sudarshan pišu o trećoj normalna forma (3NF). 3NF dodatno proširuje zahtjeve druge normalne forme (2NF). Ona zahtijeva da svi atributi koji nisu dio primarnog ključa ovise samo o primarnom ključu i da nema tranzitivnih ovisnosti. Tranzitivna ovisnost nastaje kada neki atribut koji nije ključ ovisi o drugom atributu koji također nije ključ, a koji je neizravno povezan s primarnim ključem. Da bi se postigla 3NF, treba se osigurati da svaki atribut koji nije ključ izravno ovisi samo o primarnom ključu. Na takav način se izbjegava redundancija.

Prikaz netrivijalne zavisnosti za relacije:

1. Stavka_racun (ID_stavka_racun, ID_racun, ID_ulaznica, Kolicina)

ID_stavka_racun → (ID_racun, ID_ulaznica, Kolicina)

'ID_stavka_racun' je primarni ključ i funkcijski određuje sve ostale atribute u relaciji. Svaka stavka računa mora imati jedinstven identifikator, a na temelju tog ID-a može se dobiti sve ostale informacije. Netrivijalne funkcionalne zavisnosti su zadovoljene, jer primarni ključ (ID_stavka_racun) određuje sve druge atribute u relaciji.

2. Ulaznica (ID_ulaznica, Cijena_ulaznica, Vrsta_ulaznica)

ID_ulaznica → (Cijena_ulaznica, Vrsta_ulaznica)

'ID_ulaznica' je primarni ključ i funkcijski određuje sve ostale atribute (Cijena_ulaznica, Vrsta_ulaznica). Svaka ulaznica ima jedinstven ID, cijenu i vrstu ulaznice koji su povezani s tim ID-om. Netrivijalne funkcionalne zavisnosti su zadovoljene jer 'ID_ulaznica' jednoznačno određuje cijenu i vrstu ulaznice.

3. Račun (ID_racun, ID_posjetitelj, ID_zaposlenik, Datum, Status_racuna, Iznos)

ID_racun → (ID_posjetitelj, ID_zaposlenik, Datum, Status_racuna, Iznos)

'ID_racun' je primarni ključ i funkcijski određuje sve ostale atribute. Svaki račun ima jedinstveni ID, a na temelju tog ID-a može se dobiti sve informacije o posjetitelju, zaposleniku, datumu, statusu računa i iznosu. Netrivijalne funkcionalne zavisnosti su zadovoljene jer primarni ključ (ID_racun) jednoznačno određuje sve ostale atribute u relaciji.

4. Zaposlenik (ID_zaposlenik , Ime, Prezime, Jezici, Radno_mjesto, Datum_zaposlenja, Placa)

ID_zaaposlenik → (Ime, Prezime, Jezici, Radno_mjesto, Datum_zaposlenja, Placa)

‘ID_zaposlenik’ je primarni ključ koji identificira svakog zaposlenika. Na temelju ID-a može se odrediti sve ostale atribute: ime, prezime, jezike koje govori, radno mjesto, datum zaposlenja i plaću. Netrivijalne funkcionalne zavisnosti su zadovoljene jer primarni ključ jednoznačno određuje sve druge atribute.

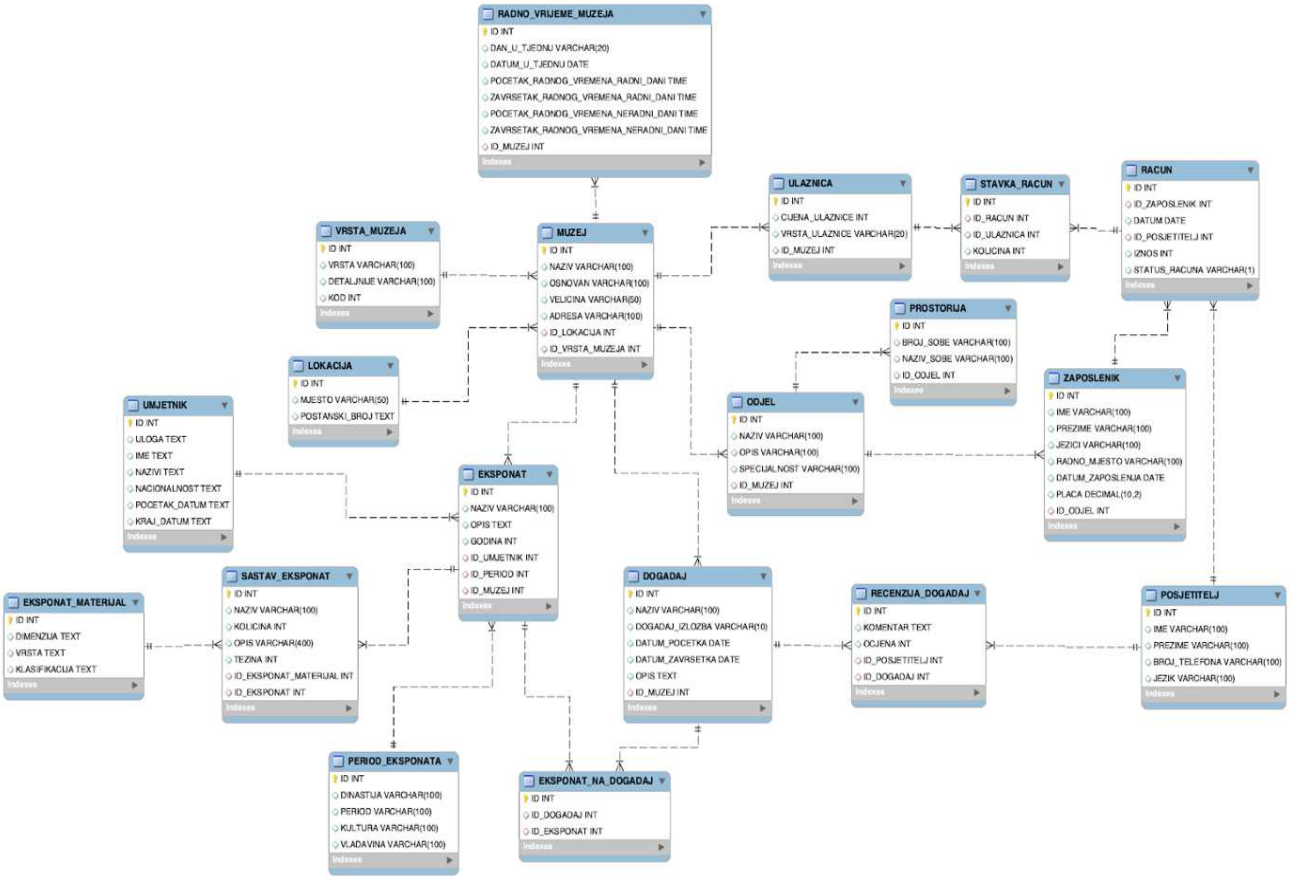
3. Eksponat (ID_eksponat, ID_sastav_eksponat, ID_umjetnik, ID_muzej, Naziv, Godina, Opis)

ID_eksponat → (ID_eksponat, ID_sastav_eksponat, ID_umjetnik, ID_muzej, Naziv, Godina, Opis)

‘ID_eksponat ‘ je primarni ključ i on jednoznačno identificira svaki eksponat. Na temelju ovog ID-a mogu se dobiti sve ostale informacije o eksponatu: njegov sastav, umjetnika, muzej u kojem se nalazi, naziv, godinu nastanka i opis. Netrivijalne funkcionalne zavisnosti su zadovoljene jer ‘ID_eksponat’ jednoznačno određuje sve ostale atribute. Svaki eksponat ima jedinstven identifikator i na temelju njega se može saznati sve potrebne informacije.

U svim danim relacijama netrivijalne funkcionalne zavisnosti su zadovoljene jer primarni ključevi u svakoj relaciji jednoznačno određuju sve ostale atribute, što je potrebno za održavanje funkcionalne konzistentnosti unutar baze podataka. Prema tome može se zaključiti da je 3NF zadovoljena.

Logički dizajn nakon provođenja normalnih formi:



Slika 2: EER (Enhanced Entity-Relationship) dijagram za bazu podataka muzej

4.4. Fizički dizajn baze podataka

Nakon logičkog dizajna, dolazi se na fizički dizajn.

Fizički model je opis stvarne fizičke organizacije podataka, točnije baze podataka realizirane na medijima za memoriranje podataka [6].

Prikaz primjera kreiranja relacije ‘ULAZNICA’ u bazi podataka:

```
CREATE TABLE ULAZNICA(
ID INTEGER  PRIMARY KEY,
CIJENA_ULAZNICE INTEGER,
VRSTA_ULAZNICE VARCHAR(20),
ID_MUZEJ INTEGER  NOT NULL,
FOREIGN KEY (ID_MUZEJ) REFERENCES MUZEJ (ID),
CHECK (VRSTA_ULAZNICE IN ('VIP', 'Grupna', 'Osnovna')),
CHECK (CIJENA_ULAZNICE BETWEEN 20.00 AND 50.00)
);
```

Slika 3: Kreiranje relacije ‘ULAZNICA’

Opisi atributa u relacijama:

LOKACIJA:
- ID
- MJESTO (grad, država)
- POSTANSKI_BROJ (poštanski broj mjesta)

VRSTA_MUZEJA:
- ID
- DETALJNIJE (opis kakav je muzej)
- VRSTA (koja je vrsta muzeja, na primjer povijesni, pomorski.)
- KOD (jedinstveni kod za vrstu muzeja)

MUZEJI:
- ID
- NAZIV (naziv muzeja)
- OSNOVAN (koje godine je muzej osnovan)
- VELICINA (velicina muzeja u metrima)
- ADRESA (adresa muzeja)
- ID_LOKACIJA (vanjski ključ relacije ‘Lokacija’)
- ID_VRSTA_MUZEJA (vanjski ključ relacije ‘Vrsta_muzej’)

RADNO_VRIJEME_MUZEJA:

- ID
- DAN_U_TJEDNU (dan u tjednu)
- DATUM_U_TJEDNU (datum u tjednu)
- POCETAK_RADNOG_VREMENA_RADNI_DANI (početak radnog vremena radni dani muzeja)
- ZAVRSETAK_RADNOG_VREMENA_RADNI_DANI (završetak radnog vremena radni dani muzeja)
- POCETAK_RADNOG_VREMENA_NERADNI_DANI (početak radnog vremena neradni dani muzeja)
- ZAVRSETAK_RADNOG_VREMENA_NERADNI_DANI (završetak radnog vremena neradni dani muzeja)
- ID_MUZEJ (vanjski ključ relacije 'MUZEJ')

POSJETITELJ:

- ID
- IME (ime posjetitelja)
- PREZIME (prezime posjetitelja)
- BROJ_TELEFONA (broj telefona posjetitelja)
- JEZIK (jezik kojim govori posjetitelj)

ULAZNICA:

- ID
- CIJENA_ULAZNICE (cijena ulaznice muzeja)
- VRSTA_ULAZNICE (vrsta ulaznice muzeja, VIP, grupna i obicna)
- ID_MUZEJ (vanjski ključ relacije 'MUZEJ')

ODJEL:

- ID
- NAZIV(naziv odjela)
- OPIS (opis odjela, što sadržava)
- SPECIJALNOST (u kojem području je odjel specijaliziran (Povijest umjetnosti, Prirodne znanosti, Etnologija, Arheologija, Tehnologija))
- ID_MUZEJ (vanjski ključ relacije 'MUZEJ')

ZAPOSLENIK :

- ID
- IME (ime zaposlenika)
- PREZIME (prezime zaposlenika)
- JEZICI (kojim jezicima govori zaposlenik)
- RADNO_MJESTO (radno mjesto zaposlenika)
- DATUM_ZAPOSLENJA (datum zaposlenja zaposlenika)
- PLACA (placa zaposlenika)
- ID_ODJEL (vanjski ključ relacije 'ODJEL')

RACUN:

- ID
- DATUM (datum izdavanja računa)
- IZNOS (iznos računa)
- STATUS_RACUNA (status računa može biti 'T' ili 'F', 'T' označava da je plaćen,a 'F' znači da je storniran)
- ID_POSJETITELJ (vanjski ključ relacije 'POSJETITELJ')

- ID_ZAPOSLENIK (vanjski ključ relacije ‘ZAPOSLENIK’)

STAVKA_RACUN:
- ID
- KOLICINA (kolicina kupljenih ulaznica)
- ID_RACUN (vanjski ključ relacije ‘RACUN’)
- ID_ULAZNICA (vanjski ključ relacije ‘ULAZNICA’)

DOGADAJ:
- ID
- NAZIV (naziv događaja)
- DOGADAJ_IZLOZBA (izložba događaja)
- DATUM_POCETKA (datum početka izložbe)
- DATUM_ZAVRSETKA (datum završetka izložbe)
- OPIS (opis događaja, koja je tematika, što će sve sadržavati događaj)
- ID_MUZEJ (vanjski ključ relacije ‘MUZEJ’)

PROSTORIJA:
- ID
- BROJ_SOBE (oznaka broj soba po prostorijama)
- NAZIV_SOBE (naziv sobe u prostoriji)
- ID_ODJEL (vanjski ključ relacije ‘ODJEL’)

RECENZIJA_DOGADAJ:
- ID
- KOMENTAR (komentar za događaj)
- OCJENA (ocjena događaja, od 1 do 5)
- ID_POSJETITELJ (vanjski ključ relacije ‘ODJEL’)
- ID_DOGADAJ (vanjski ključ relacije ‘POSJETITELJ’)

PERIOD_EKSPONAT:
- ID
- DINASTIJA (iz koje dinastije je eksponat)
- PERIOD (iz kojeg perioda je eksponat)
- KULTURA (iz koje kulture je eksponat)
- VLADAVINA (iz koje vladavine je eksponat)

UMJETNIK:
- ID
- ULOGA (označava da li je umjetnik kipar, pisac..)
- IME (ime umjetnika)
- NAZIV (puni naziv umjetnika)
- POCETAK_DATUMA (početak stvaranja eksponata)
- KRAJ_DATUMA (završetak stvaranja eksponata)
- NACIONALNOST (nacionalnost umjetnika)

EKSPONAT:
- ID
- NAZIV (naziv eksponata)
- GODINA (godina stvaranja eksponata)
- OPIS (opis što taj eksponat predstavlja)
- ID_UMJETNIK (vanjski ključ relacije ‘UMJETNIK’)
- ID_PERIOD (vanjski ključ relacije ‘PERIOD’)
- ID_MUZEJ (vanjski ključ relacije ‘MUZEJ’)

EKSPONAT_MATERIJAL:

- ID
- DIMENZIJA (veličina eksponata u centimetrima)
- VRSTA (predstavljaju različite vrste materijala ili tehnika korištenih u eksponatima (na primjer zlato, srebro))
- KLASIFIKACIJA (klasificiraju eksponate prema materijalima, tehnikama ili kategorijama (na primjer metal, staklo))

SASTAV_EKSPONAT:

- ID
- NAZIV (označava što eksponat predstavlja u umjetničkom djelu (na primjer Umjetnička Kreacija, Eksponat Evolucija Sastav Elementar))
- KOLICINA (količina sastava u eksponatu)
- OPIS (opis sastava u eksponatu)
- TEZINA (težina eksponata u kilogramima)
- ID_EKSPONAT_MATERIJAL (vanjski ključ relacije 'EKSPONAT_MATERIJAL ')
- ID_EKSPONAT (vanjski ključ relacije 'EKSPONAT')

EKSPONAT_NA_DOGADAJ

- ID
- ID_DOGADAJ (vanjski ključ relacije 'DOGADAJ')
- ID_EKSPONAT (vanjski ključ relacije 'EKSPONAT')

5. Implementacija baze podataka

5.1. Detalji o tehnologijama i alatima korištenim za implementaciju

Za implementaciju baze podataka eksponata u muzeju koristi se baza MySQL.
MySQL je najpopularnija baza podataka otvorenog koda na svijetu. Izvorno je razvijen za brzo rukovanje velikim bazama podataka i koristi se u vrlo zahtjevnim proizvodnim okruženjima već dugi niz godina [1].
U ovoj bazi podataka koristila se je za izradu relacije, upita, funkcija, procedura, okidača i pogleda.

5.2. Opis implementacije poslovnih pravila

Implementacija poslovnih pravila unutar baze podataka provdilo se je putem postavljanje validacija na razini tablica što je ključno za održavanje dosljednosti i integriteta podataka u bazama podataka, posebno u kontekstu muzejske baze. Ove validacije pomažu osigurati da se podaci unose i pohranjuju na način koji je u skladu s poslovnim pravilima i zahtjevima sustava. Ograničenja poput NOT NULL, UNIQUE, CHECK ograničenja, okidača i vanjskih ključeva (Foreign Keys) pružaju strukturu i pouzdanost u upravljanju podacima.

5.3. Opis postupka implementacije funkcionalnosti

Za implementaciju funkcionalnosti koriste se pohranjene procedure, funkcije, okidači, pogledi i upiti.

Alan [12] piše o pogledima ili view. Pogled je upit podatka. Koji za razliku od relacija ne uključuje pohranu podataka. To su privremene “relacije” koje ne uključuju pohranu podataka,već uzima od postojećih relacija podatke i prikazuje podatke kao upit.
James i Paul [11] objašnjavaju procedure, funkcije i okidače. Razlika između funkcije i procedure je ta što se funkcija koristi kao izraz stupaca naredba SELECT, te zbog toga može vratiti samo jednu vrijednost podataka (na primjer objekt) kada se pozove, a procedura može vratiti puno podataka ili u nekim slučajevima ništa. Okidači se koriste za sigurnost podataka u bazi. Pri umetanju, brisanju ili ažuriranju baze u slučaju ako zaposlenik želi napraviti neku promjenu provjerava ima li zaposlenik te ovlasti.

5.3.1. Pohranjene procedure (Stored Procedures)

Pohranjena procedura koja izračunava iznos računa:

Cilj ove procedure 'IZRACUN_IZNOSA_RACUNA' je automatizirati proces izračuna iznosa računa. Uzima se u obzir vrsta kupljenih ulaznica, količina i posebni uvjeti kao što su popusti za grupne ulaznice, popust ako je posjetitelj kupio više od 15 ulaznica i poskupljenje ulaznice ako je kupljena na dan događaja.

Kako kod radi:
U proceduri se koriste podaci iz relacija 'RACUN', 'STAVKA_RACUN' i 'ULAZNICA'.
U proceduri se nalazi ulazni parametar 'P_ID_RACUN' koje identificiraju račun za koji treba izračunati iznos. Dohvaća se ID posjetitelja i datum računa iz relacije 'RACUN'. Računa se ukupan broj ulaznica na trenutnom računu i ukupan broj ulaznica koje je posjetitelj kupio od tog trenutka. Procedura onda provjerava je li datum računa unutar raspona određenog događaja,ako je cijena se uvećava za 50%. Za svaku stavku računa koristi se kursor koji iterira kroz stavke računa. Izračunava ukupan iznos za svaku stavku i dodaje ga ukupnom iznosu računa. Ako se radi o grupnoj ulaznici, a broj ulaznica je jednak 30 ili više, dobiva se popust od 15%. Kad završi iteracija procedura ona provjerava ukupan broj ulaznica i ako je broj ulaznica veći od 15 dobiva se popust od 10%. Na kraju procedura ažurira relaciju 'RACUN' s izračunatim ukupnim iznosom.

```
DELIMITER //

CREATE PROCEDURE IZRACUN_IZNOS_RACUNA(
    IN P_ID_RACUN INTEGER
)
BEGIN
    DECLARE V_CIJENA DECIMAL(10, 2);
    DECLARE V_VRSTA VARCHAR(20);
    DECLARE V_KOLICINA INTEGER;
    DECLARE V_ID_POSJETITELJ INTEGER;
    DECLARE V_IZNOS DECIMAL(10, 2);
    DECLARE V_TOTAL_IZNOS DECIMAL(10, 2) DEFAULT 0;
    DECLARE V_TOTAL_TIKET INTEGER DEFAULT 0;
    DECLARE V_TOTAL_TIKET_POSJETITELJ INTEGER DEFAULT 0;
    DECLARE V_DATUM DATE;
    DECLARE V_IS_EVENT_DAY BOOLEAN DEFAULT FALSE;
    DECLARE DONE INT DEFAULT FALSE;

    DECLARE CUR CURSOR FOR
        SELECT U.CIJENA_ULAZNICE, U.VRSTA_ULAZNICE, S.KOLICINA
        FROM STAVKA_RACUN S
        JOIN ULAZNICA U ON S.ID_ULAZNICA = U.ID
        WHERE S.ID_RACUN = P_ID_RACUN;

    DECLARE CONTINUE HANDLER FOR NOT FOUND SET DONE = TRUE;

    SELECT ID_POSJETITELJ, DATUM INTO V_ID_POSJETITELJ, V_DATUM FROM RACUN WHERE ID = P_ID_RACUN;
```

Slika 4: Prikaz SQL koda za pohranjenu proceduru 'IZRACUN_IZNOSA_RACUNA'

```
    SELECT COUNT(*) INTO V_TOTAL_TIKET_POSJETITELJ
    FROM STAVKA_RACUN S
    JOIN RACUN R ON S.ID_RACUN = R.ID
    WHERE R.ID_POSJETITELJ = V_ID_POSJETITELJ;

    SELECT COUNT(*) INTO V_TOTAL_TIKET
    FROM STAVKA_RACUN
    WHERE ID_RACUN = P_ID_RACUN;

    SELECT COUNT(*) > 0 INTO V_IS_EVENT_DAY
    FROM DOGADAJ
    WHERE V_DATUM BETWEEN DATUM_POCETKA AND DATUM_ZAVRSETKA;

    OPEN CUR;

    LOOPIC: LOOP
        FETCH CUR INTO V_CIJENA, V_VRSTA, V_KOLICINA;
        IF DONE THEN
            LEAVE LOOPIC;
        END IF;


        IF V_IS_EVENT_DAY THEN
            SET V_CIJENA = V_CIJENA * 1.5;
        END IF;

        IF V_VRSTA = 'Grupna' AND V_TOTAL_TIKET >= 30 THEN
             SET V_IZNOS = V_KOLICINA * V_CIJENA * 0.85;
        ELSE
            SET V_IZNOS = V_KOLICINA * V_CIJENA;
        END IF;

        SET V_TOTAL_IZNOS = V_TOTAL_IZNOS + V_IZNOS;
    END LOOP;

    CLOSE CUR;
    IF V_TOTAL_TIKET_POSJETITELJ > 15 THEN
        SET V_TOTAL_IZNOS = V_TOTAL_IZNOS * 0.90;
    END IF;

    UPDATE RACUN
    SET IZNOS = V_TOTAL_IZNOS
    WHERE ID = P_ID_RACUN;
END //

DELIMITER ;
```

Slika 5: Prikaz SQL koda za pohranjenu proceduru 'IZRACUN_IZNOSA_RACUNA'

Pohranjena procedura koja stornira račun:

Cilj ove procedure 'STORNO_RACUNA' je automatizirati proces storniranja računa. Uzima se u obzir ako je zaposlenik stornirao 10 karata na taj dan i ako je status računa 'F' storniranje karta se prekida.

Kako kod radi:
U proceduri se koriste podaci iz relacije 'RACUN'. U proceduri se nalazi ulazni parametar 'P_ID_RACUN' kojim se identificira račun kojeg treba obrisati. Dohvaća se status računa, ID zaposlenika iz relacije 'RACUN'. Provjerava se da li je status računa 'F', ako je storniranje karte se prekida i javlja se poruka 'Račun je izbrisan'. Ako status karte nije 'F' broji se koliko je zaposlenik na taj dan prodao karata, ako je broj jednak 10 ili više prekida se postupak storniranja i javlja se poruka 'Zaposlenik ne može stornirati više od 10 računa dnevno'. Ako se radnja nije prekinula račun se stornira.

```
DELIMITER //

CREATE PROCEDURE STORNO_RACUNA(IN P_ID_RACUN INTEGER)
BEGIN
    DECLARE P_STATUS_RACUNA VARCHAR(1);
    DECLARE P_ID_ZAPOSLENIK INTEGER;
    DECLARE P_PREKINI_BROJANJE INTEGER;

    SELECT STATUS_RACUNA, ID_ZAPOSLENIK INTO P_STATUS_RACUNA, P_ID_ZAPOSLENIK FROM RACUN WHERE ID = P_ID_RACUN;

    IF P_STATUS_RACUNA = 'F' THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Račun je izbrisan.';
    END IF;

    SELECT COUNT(*) INTO P_PREKINI_BROJANJE
    FROM RACUN
    WHERE ID_ZAPOSLENIK = P_ID_ZAPOSLENIK
    AND STATUS_RACUNA = 'F'
    AND DATE(DATUM) = CURDATE();

    IF P_PREKINI_BROJANJE >= 10 THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Zaposlenik ne može stornirati više od 10 računa dnevno.';
    END IF;

    UPDATE RACUN
    SET STATUS_RACUNA = 'F'
    WHERE ID = P_ID_RACUN;
    DELETE FROM RACUN WHERE ID = P_ID_RACUN;
END //

DELIMITER ;
```

Slika 6: Prikaz SQL koda za pohranjenu proceduru 'STORNO_RACUNA'

Pohranjena procedura koja obračuna plaće za zaposlenike:

Cilj ove procedure 'OBRACUN_PLACE' je automatizirati proces koji povećava plaće. Uzima se u obzir ako je zaposlenik radi 5 ili više godina tada se plaća uvećava za 10%.

Kako kod radi:
U proceduri se koriste podaci iz relacije 'ZAPOSLENIK'. U proceduri nema ulaznog parametra. Dohvaća se 'ID', 'DATUM_ZAPOSLENJA' i 'PLACA' iz relacije 'ZAPOSLENIK'. Za svakog zaposlenika koristi se kursor koji iterira kroz relaciju 'ZAPOSLENIK'. Provjerava se je li godina radnog staža veća ili jednaka od 5 godina. Te ako je plaća se uvećava za 10%. Na samom kraju se ažurira relacija 'ZAPOSLENIK u atribut 'PLACA'.

```
DELIMITER //

CREATE PROCEDURE OBRACUN_PLACE()
BEGIN
    DECLARE DONE INT DEFAULT 0;
    DECLARE P_ID INTEGER;
    DECLARE P_GODINA_STAZA INTEGER;
    DECLARE P_NOVA_PLACA DECIMAL(10, 2);

    DECLARE CUR CURSOR FOR
        SELECT ID, YEAR(CURDATE()) - YEAR(DATUM_ZAPOSLENJA) AS GODINESTAZA, PLACA
        FROM ZAPOSLENIK;

    DECLARE CONTINUE HANDLER FOR NOT FOUND SET DONE = 1;

    OPEN CUR;

    LOOPIC: LOOP
        FETCH CUR INTO P_ID, P_GODINA_STAZA, P_NOVA_PLACA;
        IF DONE THEN
            LEAVE LOOPIC;
        END IF;

        IF P_GODINA_STAZA >= 5 THEN
            SET P_NOVA_PLACA = P_NOVA_PLACA * 1.10;
        END IF;

        UPDATE ZAPOSLENIK
        SET PLACA = P_NOVA_PLACA
        WHERE ID = P_ID;
    END LOOP;

    CLOSE CUR;
END //

DELIMITER ;
```

Slika 7: Prikaz SQL koda za pohranjenu proceduru 'OBRACUN_PLACE'

5.3.2. Funkcija

Funkcija koja provjerava da li zaposlenik može prodati karate

Cilj ove funkcije ‘DAL_MOZE_PRODAT_KARTU’ je da provjerava je li određeni zaposlenik može prodati karte. Uzima se u obzir gdje radi zaposlenik i u kojem muzeju. Ova funkcija nam omogućava sprječavanje zloupotrebe prodaja karata.

Kako kod radi:

Napravljena je funkcija ‘DAL_MOZE_PRODAT_KARTU’ koja provjerava je li određeni zaposlenik može prodati kartu posjetitelju za određeni događaj u muzeju. Funkcija prima dva ulazna parametra: ID zaposlenika i ID muzeja. Zbog tih parametra se može vidjeti je li određeni zaposlenik može prodati kartu. U funkciji se koristi SQL upit koja ima varijablu ‘v_count’ za brojanje zapisa u tablici ‘ZAPOSLENIK’.Provjerava je li ID zaposlenika jednak unesenom parametru ‘P_ID_ZAPOSLENIK’, tako radi za ID muzej koji mora biti jednak s parametrom ‘P_ID_MUZEJ’. Također provjerava da zaposlenik ima radno mjesto "Prodavac karata". Ako se rezultati ovog upita zadovolje sprema se u varijablu ‘v_count’. Ako se u varijablu ‘v_count’ spremi 1 to znači da zaposlenik može prodavati karte, ako se spremi 0 znači da ne može prodavati karte. Na ovaj način je osigurano da samo određeni zaposlenik može prodavati karte i tako je spriječena zlouporaba prodaje.

```
DELIMITER //

CREATE FUNCTION DAL_MOZE_PRODAT_KARTU(P_ID_ZAPOSLENIK INTEGER, P_ID_MUZEJ INTEGER) RETURNS TINYINT(1)
BEGIN
    DECLARE v_count INTEGER;

    SELECT COUNT(*)
    INTO v_count
    FROM ZAPOSLENIK z
    JOIN ODJEL o ON z.ID_ODJEL = o.ID
    WHERE z.ID = P_ID_ZAPOSLENIK AND o.ID_MUZEJ = P_ID_MUZEJ AND z.RADNO_MJESTO = 'Prodavac karata';

    RETURN v_count > 0;
END //

DELIMITER ;
```

Slika 8: Prikaz SQL koda za funkciju ‘DAL_MOZE_PRODAT_KARTU’

Funkcija koja pokazuje koji je popularan događaj

Cilj ove funkcije 'POPULARAN_DOGADAJ' je da provjerava koji događaj je popularan. Uzima se u obzir naziv događaja. Ova funkcija nam omogućava analizu događaja.

Kako kod radi:
Napravljena je funkcija 'POPULARAN_DOGADAJ' koja provjerava koji događaj je popularan. Funkciju kada se pozove uzima potrebne podatke iz relacija 'DOGADAJ', 'MUZEJ' i 'RECENZIJE_DOGADAJ'. Onda spoji ID-ove i izračuna recenzije uz pomoć 'COUNT'. Poreda ih uzlazno i prikaze jedan događaj uz pomoć 'LIMIT'. Na takav način se može analizirati potrebne događaje koje bi se mogle poboljšati.

```
DELIMITER //

CREATE FUNCTION POPULARAN_DOGADAJ() RETURNS VARCHAR(100)
DETERMINISTIC
BEGIN
    DECLARE POP_DOG VARCHAR(100);

    SELECT DOGADAJ.NAZIV INTO POP_DOG
    FROM DOGADAJ
    JOIN MUZEJ ON MUZEJ.ID = DOGADAJ.ID_MUZEJ
    JOIN RECENZIJA_DOGADAJ ON DOGADAJ.ID = RECENZIJA_DOGADAJ.ID_DOGADAJ
    GROUP BY DOGADAJ.ID
    ORDER BY COUNT(RECENZIJA_DOGADAJ.ID) DESC
    LIMIT 1;

    RETURN POP_DOG;
END //

DELIMITER ;
```

Slika 9: Prikaz SQL koda za funkciju 'POPULARAN_DOGADAJ'

5.3.3. Triggers (Okidači):

Okidač za broja storniranih računa po zaposleniku

Cilj ovog okidača ‘PROVJERA_KOLIKO_JE_ZAPOSLENIK_STORNIRAO_RACUNA’ je da provjerava koliko puta je određeni zaposlenik stornirao račune na isti dan i spriječiti ga da to učini više od 10 puta dnevno.

Kako kod radi:
Napravljen je okidač ‘PROVJERA_KOLIKO_JE_ZAPOSLENIK_STORNIRAO_RACUNA’ koja se aktivira prije svakog ažuriranja u relaciji ‘RACUN’ u atributu ‘status_racuna’. Okidač započinje varijablom ‘PREKID’ koja će spremiti broj računa koje je isti zaposlenik stornirao na isti dan. Okidač se aktivira samo kada se status računa mijenja u 'F', što označava "stornirano". Provjera se vrši uz pomoću SQL upita koji broji koliko puta je isti zaposlenik stornirao račune na isti dan koristi ID zaposlenika i filtrira račune s statusom 'F' na temelju datuma. Rezultat prebrojavanja sprema se u varijablu ‘PREKID’. Broj storniranih računa prelazi ili je jednak 10, okidač generira grešku koristeći ‘SIGNAL SQLSTATE’ prekida se izvršenje i vraća poruku: "Zaposlenik ne može stornirati više od 10 računa dnevno." Osigurava se da se ne može stornirati više od 10 računa dnevno za istog zaposlenika i sprječava se zloupotreba ili greške koje bi mogle nastati ako bi jedan zaposlenik imao mogućnost neograničeno stornirati račune.

```
DELIMITER //

CREATE TRIGGER PROVJERA_KOLIKO_JE_ZAPOSLENIK_STORNIRAO_RACUNA
BEFORE UPDATE ON RACUN
FOR EACH ROW
BEGIN
    DECLARE PREKID INT;


    IF NEW.STATUS_RACUNA = 'F' THEN

        SELECT COUNT(*) INTO PREKID
        FROM RACUN
        WHERE ID_ZAPOSLENIK = NEW.ID_ZAPOSLENIK
        AND STATUS_RACUNA = 'F'
        AND DATE(DATUM) = CURDATE();


        IF PREKID >= 10 THEN
            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Zaposlenik ne može stornirati više od 10 računa dnevno.';
        END IF;
    END IF;
END //

DELIMITER ;
```

Slika 10: Prikaz SQL koda za okidač ‘PROVJERA_KOLIKO_JE_ZAPOSLENIK_STORNIRAO_RACUNA’

Okidač za provjeru eksponata na događaju

Cilj ovog okidača 'PROVJERA_EKSPONATA_NA_DOGADAJU_U_INSERT' je da provjerava je li taj određeni eksponat već postavljen na određenom događaju.

Kako kod radi:
Napravljen je okidač 'PROVJERA_EKSPONATA_NA_DOGADAJU_U_INSERT' koji se aktivira prije svakog ubacivanja u relaciji 'EKSPONAT_NA_DOGADAJ'. Okidač započinje varijablama 'P_DATUM_POCETKA' i 'P_DATUM_ZAVRSETKA'. Okidač se aktivira kada se dodaje eksponat na događaj. Provjera se vrši uz pomoć SQL upita koji traži vremenski raspon od tog relacije 'DOGAĐAJ' atribute 'DATUM_POCETAK' i 'DATUM_ZAVRSETAK'. To se sprema u varijable 'P_DATUM_POCETKA' i 'P_DATUM_ZAVRSETKA'. Na kraju se zove procedura 'PROVJERI_EKSPONAT_NA_DOGAĐAJ' s varijablama 'ID_EKSPONAT', 'P_DATUM_POCETKA' i 'P_DATUM_ZAVRSETKA' koja provjerava je li se eksponat nalazi na određenom događaju.

```
DELIMITER //

CREATE TRIGGER PROVJERA_EKSPONATA_NA_DOGADAJU_U_INSERT
BEFORE INSERT ON EKSPONAT_NA_DOGADAJ
FOR EACH ROW
BEGIN
    DECLARE V_DATUM_POCETKA DATE;
    DECLARE V_DATUM_ZAVRSETKA DATE;


    SELECT DATUM_POCETKA, DATUM_ZAVRSETKA INTO V_DATUM_POCETKA, V_DATUM_ZAVRSETKA
    FROM DOGADAJ
    WHERE ID = NEW.ID_DOGADAJ;


    CALL PROVJERI_EKSPONAT_NA_DOGADAJU(NEW.ID_EKSPONAT, V_DATUM_POCETKA, V_DATUM_ZAVRSETKA);
END //

DELIMITER ;
```

Slika 11: Prikaz SQL koda za okidač 'PROVJERA_EKSPONATA_NA_DOGADAJU_U_INSERT'

5.3.4. View (Pogledi)

Pogled koji prikazuje zaradu po godinama

Cilj ovog pogleda 'PROFIT_PO_GODINAMA' je da prikazuje godišnju zaradu na temelju ukupne prodaje i ukupnih troškova plaća.

Kako kod radi:
Napravljen je pogled koji se naziva 'PROFIT_PO_GODINAMA' . Pogled uzima podatke iz dva pogleda:'UKUPNA_PRODAJA_PO_GODINAMA' i 'UKUPNI_TROSKOVI_PLACA_PO_GODINAMA', koristi se LEFT JOIN na temelju godine ('GODINA'). Uz pomoć funkcije 'COALESCE' ako nema podataka o troškovima plaća za određenu godinu, funkcija će koristiti vrijednost 0 umjesto NULL-a. Tako se sprječava da izračun bude neispravan. Rezultat na kraju je taj da prikazuje profit za svaku godinu uzimajući u obzir dostupne podatke o prodaji i troškovima plaća, to omogućava analizu financija po godinama.

```
CREATE VIEW PROFIT_PO_GODINAMA AS
SELECT UKUPNA_PRODAJA_PO_GODINAMA.GODINA, UKUPNA_PRODAJA_PO_GODINAMA.UKUPNA_PRODAJA,
COALESCE(UKUPNI_TROSKOVI_PLACA_PO_GODINAMA.UKUPNI_TROSKOVI_PLACA, 0) AS UKUPNI_TROSKOVI_PLACA,
(UKUPNA_PRODAJA_PO_GODINAMA.UKUPNA_PRODAJA - COALESCE(UKUPNI_TROSKOVI_PLACA_PO_GODINAMA.UKUPNI_TROSKOVI_PLACA, 0)) AS PROFIT
FROM UKUPNA_PRODAJA_PO_GODINAMA
LEFT JOIN UKUPNI_TROSKOVI_PLACA_PO_GODINAMA  ON UKUPNA_PRODAJA_PO_GODINAMA.GODINA = UKUPNI_TROSKOVI_PLACA_PO_GODINAMA.GODINA;
```

Slika 12: Prikaz SQL koda za pogled 'PROFIT_PO_GODINAMA'

Pogled koji prikazuje ukupnu prodaju po godinama

Cilj ovog pogleda 'UKUPNA_PRODAJA_PO_GODINAMA' je da prikazuje godišnju zaradu na temelju ukupne prodaje.

Kako kod radi:
Napravljen je pogled koji se naziva 'UKUPNA_PRODAJA_PO_GODINAMA'. Pogled uzima podatke iz relacije 'RACUN' , atribute 'DATUM' i 'IZNOS'. Godinu uzima sa atributom 'DATUM' pomoću funkcije YEAR(). Atribut 'IZNOS' se izračuna uz pomoć funkcije SUM(). Grupira po godini prodaje. Rezultat na kraju je prikaz prodaje po godini.

```sql
CREATE VIEW UKUPNA_PRODAJA_PO_GODINAMA AS
SELECT  YEAR(DATUM) AS Godina, SUM(IZNOS) AS UKUPNA_PRODAJA
FROM  RACUN
GROUP BY YEAR(DATUM);
```

Slika 13: Prikaz SQL koda za pogled 'UKUPNA_PRODAJA_PO_GODINAMA'

Pogled koji prikazuje ukupan trošak plaćanja po godinama

Cilj ovog pogleda 'UKUPNI_TROSKOVI_PLACA_PO_GODINAMA' je da prikazuje godišnje troškove plaća .

Kako kod radi:

Napravljen je pogled koji se naziva 'UKUPNI_TROSKOVI_PLACA_PO_GODINAMA'. Pogled uzima podatke iz relacije 'ZAPOSLENIK' , atribute 'DATUM_ZAPOSLENJA' i 'PLACA'. Godinu uzima sa atributom 'DATUM_ZAPOSLENJA' pomoću funkcije YEAR() i atribut 'PLACA' izračuna uz pomoć funkcije SUM() i to pomnoži sa 12. Grupira po godini zaposlenja. Rezultat na kraju je prikaz ukupnih troškova plaća po godinama.

```sql
CREATE VIEW UKUPNI_TROSKOVI_PLACA_PO_GODINAMA  AS
SELECT YEAR(DATUM_ZAPOSLENJA) AS Godina, SUM(PLACA * 12) AS UKUPNI_TROSKOVI_PLACA
FROM ZAPOSLENIK
GROUP BY YEAR(DATUM_ZAPOSLENJA);
```

Slika 14: Prikaz SQL koda za pogled 'UKUPNI_TROSKOVI_PLACA_PO_GODINAMA'

5.3.5. Upit

Upit koji prikazuje sve događaje i jednog posjetitelja koji je objavio komentar na pojedinom događaju

Cilj ovih upita je prikazati informacije o događajima zajedno s posjetiteljem koji je ostavio najviše recenzija za svaki događaj.

Kako prvi upit radi:
Prvi upit koristi podupit unutar 'WHERE' uvjeta za dohvaćanje posjetitelja s najviše recenzija za svaki događaj. Prvo se dohvaća relacija 'POSJETITELJ' koji je ostavio najviše recenzija za određeni događaj pomoću 'GROUP BY' i 'ORDER BY COUNT(*) DESC', gdje je 'COUNT(*)' broj recenzija, a 'LIMIT 1' osigurava da dohvaća samo posjetitelja s najvećim brojem recenzija. Ovaj rezultat se uspoređuje u vanjskom upitu pomoću 'WHERE' uvjeta kako bi se dobio posjetitelj za svaki događaj. 'SELECT' dio vraća sve podatke iz relacije 'DOGADAJ' i 'POSJETITELJ'.

Prednosti ovog pristupa su jednostavnost i lakoća razumijevanja, ali može biti manje optimiziran jer koristi podupit može usporiti izvođenje baze kada je baza velika.

```
SELECT D.*, P.*
 FROM DOGADAJ AS D, POSJETITELJ AS P
 WHERE P.ID = ( SELECT ID_POSJETITELJ
                FROM RECENZIJA_DOGADAJ
                WHERE ID_DOGADAJ = D.ID
                GROUP BY ID_DOGADAJ, ID_POSJETITELJ
                ORDER BY COUNT(*) desc
                LIMIT 1);
```

Slika 15: Prikaz SQL koda za upit

Kako drugi upit radi:
Drugi upit koristi 'JOIN' kako bi povezao podatke iz tablica 'DOGADAJ' i 'POSJETITELJ'. Prvo, koristi se podupit unutar 'JOIN' da bi se identificirao posjetitelj s najviše recenzija za svaki događaj. Unutar podupita, 'GROUP BY' se koristi kako bi se grupirale recenzije po događajima, a 'ORDER BY COUNT(*) DESC' osigurava da dohvati posjetitelja koji je ostavio najviše recenzija. 'LIMIT 1' osigurava da dohvati samo jednog posjetitelja s najviše recenzija. Ovaj podupit se povezuje s glavnim upitom pomoću 'JOIN', što omogućava efikasnije dohvaćanje podataka iz obje tablice.

Prednost ovog pristupa je efikasnost jer koristi ‘JOIN’ umjesto podupita, što može poboljšati performanse upita kod većih skupova podataka.

```
SELECT D.*, P.*
FROM DOGADAJ D
JOIN ( SELECT ID_DOGADAJ, ID_POSJETITELJ
        FROM RECENZIJA_DOGADAJ RD1
        WHERE ID_POSJETITELJ = ( SELECT ID_POSJETITELJ
                                 FROM RECENZIJA_DOGADAJ RD2
                                 WHERE RD2.ID_DOGADAJ = RD1.ID_DOGADAJ
                                 GROUP BY ID_POSJETITELJ
                                 ORDER BY COUNT(*) DESC
                                 LIMIT 1)
    ) AS TOP_RECENZIJA ON D.ID = TOP_RECENZIJA.ID_DOGADAJ
     JOIN  POSJETITELJ P  ON TOP_RECENZIJA.ID_POSJETITELJ = P.ID;
```

Slika 16: Prikaz SQL koda za upit

Može se zaključiti da oba upita imaju isti cilj, ali se razlikuju po načinu implementacije. Drugi upit je efikasniji jer koristi ‘JOIN’, koji je bolji za više podataka, dok prvi upit može biti jednostavniji za razumijevanje, ali manje je efikasan u performansama kod većih količina podataka.

6. Korištenje baze podataka

Prikazat ću korištenje (use-case) koji ću prikazati na način kako korisnici mogu koristiti aplikaciju za upravljanje bazom podataka u muzeju. Pokazati ću neke od zanimljivih SQL upita koje sam napravila.

Use Case 1: Unos karata na račun i potvrđivanje računa

Prodavač karata unosi ulaznice koje posjetitelj kupuje, potvrđuje se račun i nakon čega se izračunava ukupni iznos.

1. Unos karata na račun

Prodavač karata unosi u račun podatke ID ulaznice, ID posjetitelja. Iznos se kasnije upisuje uz pomoć procedure koje sada opisujemo. Stavlja se status računa na "T" (potvrđeno). Podaci se stavljaju u relaciju 'ULAZNICA', koja je povezana s relacijom 'STAVKA_RACUN'. U relaciji 'STAVKA RACUN' prodavač karata unosi ID račun, ID ulaznicu i količinu ulaznica.

```
INSERT INTO RACUN (ID, ID_ZAPOSLENIK, DATUM, ID_POSJETITELJ, IZNOS, STATUS_RACUNA) VALUES (1, 1, '2024-11-07', 1, NULL, 'T');
```

Slika 17: Prikaz SQL koda unos podataka u relaciju 'RACUN'

```
INSERT INTO STAVKA_RACUN (ID, ID_RACUN, ID_ULAZNICA, KOLICINA) VALUES (1, 1, 1, 10);
```

Slika 18: Prikaz SQL koda unos podataka u relaciju 'STAVKA_RACUN'

2. Potvrđivanje računa

Kada prodavač karata unese sve stavke, potvrđuje račun pozivom na proceduru. Pokreće se procedura 'IZRACUN_IZNOS_RACUNA' koja izračunava ukupni iznos računa na temelju unesenih stavki i uključuje sve primjenjive popuste. Kad prodavač karata potvrdi, procedura automatski izračuna ukupni iznos. Procedura ažurira u bazi i sprema konačni iznos računa u relaciju 'RACUN'.

```
CALL IZRACUN_IZNOS_RACUNA(1);
```

Slika 19: Prikaz SQL koda za pozivanje procedure 'IZRACUN_IZNOS_RACUNA'

Use Case 2: Analiza popularnosti događaja

Određeni zaposlenik ili menadžment muzeja želi analizirati popularan događaj kako bi mogao poboljšati buduće izložbe.

1. Identifikacija najpopularnijeg događaja

Napravljena je funkcija 'POPULARAN_DOGADAJ' . Funkcija vraća naziv najpopularnijeg događaja s najviše ocjena. Ova funkcija pomaže zaposlenicima procijeniti koji je događaj popularan. Funkcija vraća jedan popularan događaj.

POPULARAN_DOGADAJ()
Izložba A

Slika 20: Prikaz popularnog događaja

2. Prikaz prosječnih ocjena za sve događaje

Napravljen je pogled 'PROSJECNA_OCJENA_DOGADAJA'. Pogled vraća prosječne ocjenu za svaki događaj i pomaže zaposlenicima u procjeni koliko je uspješna pojedina izložba. Vraća podatke ID događaja i prosječnu ocjenu posjetitelja.

	ID_DOGADAJ	avg_rating
	5	5.0000
	6	3.0000
	7	3.0000

Slika 21: Prikaz prosječnog događaja

Use Case 3: Analiza prodaje i troškova

Napravljeni su pogledi koji omogućavaju financijskom odjelu muzeja pregled ukupne prodaje i profit po godinama.

1. Pregled ukupne prodaje po godinama

Pogled 'UKUPNA_PRODAJA_PO_GODINAMA' daje pregled ukupne prodaje za svaku godinu. Ovaj pogled nam vraća godinu i ukupnu prodaju ostvarenu u toj godini i omogućava financijskom odjelu praćenje financijskog ishoda za muzeje kroz godine.

	Godina	UKUPNA_PRODAJA
	2023	205
	2024	180
	2022	165
	2021	220

Slika 22: Prikaz ukupne prodaje po godinama

2. Izračun profita po godinama

Pogled 'PROFIT_PO_GODINAMA' omogućuje prikaz profita na temelju razlike između ukupne prodaje i troškova plaća za svaku godinu. Pogled vraća godinu, ukupnu prodaju, ukupne troškove plaća i izračunati profit. Omogućava muzeju analizu financija i da se mogu donesti odluke o budućim financijskim strategijama.

	GODINA	UKUPNA_PRODAJA	UKUPNI_TROSKOVI_PLACA	PROFIT
	2023	205	1092000.00	-1091795.00
	2024	180	660000.00	-659820.00
	2022	165	660000.00	-659835.00
	2021	220	660000.00	-659780.00

Slika 23: Prikaz ukupnog profita po godinama

7. Zaključak

U ovom radu uspješno su implementirani svi postavljeni funkcionalni zahtjevi za sustav upravljanja muzejskim poslovanjem. Korištenjem pohranjenih procedura, funkcija, pogleda i okidača osigurana je točnost podataka, automatizirani su ključni poslovni procesi i omogućavaju detaljniju analizu podataka. Procedura za izračunavanje iznosa računa omogućila je točan izračun karata s primjenom postavljenih popusta, a okidači poput brisanja računa i provjere eksponata na događaju osigurali su sigurnost podataka i spriječili zloupotrebe sustava. Dodatnu kontrolu nad podacima pružile su funkcije, a pogledi su omogućili zaposlenicima pristup informacijama potrebnim za donošenje kvalitetnih odluka.

Preporučuje se daljnje proširenje baze. Povezivanje prodaja ulaznica putem interneta ili aplikacija povećalo bi prodaju karata. Bilo bi korisno uvesti analitičke alate poput 'Tableau' i algoritme poput 'Linearne regresije' za predviđanje posjećenosti. Time bi se muzeju omogućilo bolje planiranje i optimizacija resursa. Preporučuje se izrada aplikacije ili web stranice za muzeje, gdje bi sve informacije o eksponatima bile dostupne na jednom mjestu. Uvođenjem ovih poboljšanja baza bi postala još bolja i prilagođena budućim potrebama muzeja, omogućujući nastavak uspješnog poslovanja i pružanje izvrsnog iskustva posjetiteljima. Dodatno proširenje baze moglo bi uključivati dodavanje relacije koja prati restauraciju eksponata. Kroz dodatne poglede ili procedure moguće je pratiti koji su eksponati u procesu restauracije te spriječiti njihovo sudjelovanje na događajima dok je proces u tijeku.

8. Literatura:

1. What is MySQL? Link na stranicu: https://www.oracle.com/in/mysql/what-is-mysql/ (Pristupljeno: 15.7.2024)

2. Što je dijagram odnosa entiteta (ERD)? Link na stranicu: https://www.lucidchart.com/pages/er-diagrams (Pristupljeno: 24.7.2024.)

3. Marty, P. F. (2007). Museum Informatics: People, Information, and Technology in Museums. Internet stranica: https://www.researchgate.net/publication/220141970_Museum_Informatics/link/59d98de3aca272e60966d776/download?_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6InB1YmxpY2F0aW9uIiwicGFnZSI6InB1YmxpY2F0aW9uIn19 (Pristupljeno: 17.7.2024.)

4. Addison, A. C. (2008). Digital Heritage: Applying Digital Imaging to Cultural Heritage. New York: Springer. Internet stranica: (https://academic.oup.com/dsh/article-abstract/23/2/244/1005805?login=false) (Pristupljeno: 17.7.2024.)

5. Mladen Varga (2022) Internet stranica: Baze podataka: Konceptualno, logičko i fizičko modeliranje podataka - Mladen Varga - Google Knjige (Pristupljeno: 24.7.2024.)

6.Ian Sommerville, Software Engineering, 9th Edition (2011.) Internet stranica: https://engineering.futureuniversity.com/BOOKS%20FOR%20IT/Software-Engineering-9th-Edition-by-Ian-Sommerville.pdf (Pristupljeno: 24.7.2024.)

7. Connolly, T. M., & Begg, C. E. (2014). Database Systems: A Practical Approach to Design, Implementation, and Management (6th ed.). Internet stranica: https://www.cherrycreekeducation.com/bbk/b/Pearson_Database_Systems_A_Practical_Approach_to_Design_Implementation_and_Management_6th_Global_Edition_1292061189.pdf (Pristupljeno: 25.7.2024.)

8. Silberschatz, A., Korth, H. F., & Sudarshan, S. (2011). Database System Concepts. (6th ed.) Internet stranica: https://dl.ebooksworld.ir/motoman/Pearson.Database.Systems.A.Practical.Approach.to.Design.Implementation.and.Management.6th.Global.Edition.www.EBooksWorld.ir.pdf (Pristupljeno: 26.7.2024.)

9.Elmasri, R., & Navathe, S. B. (2015). Fundamentals of Database Systems (7th ed.). Boston: Pearson. Internet stranica : https://www.auhd.edu.ye/upfiles/elibrary/Azal2020-01-22-12-28-11-76901.pdf (Pristupljeno: 28.7.2024.)

10 . Abraham Silberschatz , Henry F. Korth i S. Sudarshan (2010) DATABASE SYSTEM CONCEPTS (6th ed.). Internet stranica: https://opac.atmaluhur.ac.id/uploaded_files/temporary/DigitalCollection/YzAzNTEwNDk3NzJlYzA1ODJjMTFjZmVmYjExM2JiOWY1NTNmNzkzMw==.pdf (Pristupljeno: 29.7.2024.)

11. SQL: The Complete Reference" - James R. Groff, Paul N. Weinberg (2010). Internet stranica: https://ci-ceit.edu.ck/wp-content/uploads/2021/01/sql-the-complete-reference-third-edition-sep-2009.pdf (Pristupljeno: 1.8.2024.)

12. Alan Beaulie (2009.) Learning SQL (2th ed.). Internet stranica: https://www.r-5.org/files/books/computers/languages/sql/mysql/Alan_Beaulieu-Learning_SQL-EN.pdf (Pristupljeno: 25.8.2024.)

10. Popis slika