Projekat iz Relacionih Baza Podataka

Tema:

Poslovanje Hotela

Student: Veljko Kanjevac, rt-22/22 Profesor: dr Gabrijela Dimić

Uvod

Cilj projekta je dizajn i implementacija relacione baze podataka za poslovanje hotela u okruženju Microsoft SQL Server / SSMS. Projekat obuhvata kompletan tok — od konceptualnog i logičkog modelovanja do fizičke implementacije. Odlikuju ga jednostavnost operativnog rada i podršku za izveštavanje. Baza je osmišljena da obuhvati ključne procese hotela: evidenciju gostiju, soba i zaposlenih, upravljanje rezervacijama, evidentiranje dodatnih usluga i plaćanja, kao i pregled raspoloživosti i obračuna troškova.

Prednosti ovakvog pristupa su višestruke:

- **Konzistentnost i integritet podataka**
- **Operativna efikasnost**
- **Transparentan obračun**
- **Skalabilnost i održavanje**
- **Podrška za izveštavanje i odluke**

Ukupno, projekat donosi pouzdanu osnovu za svakodnevni rad recepcije i menadžmenta, uz jasnu arhitekturu koja se može lako proširivati.

Korisnički scenario

Ova baza podataka napravljena je sa ciljem da zaposlenima u hotelu olakša poslovanje i omogući im da u svakom trenutku imaju uvid u glave paramere poslovanja. Baza je kimpelemtirana na način da bude jasna i jednostavna za koriscenje. Evo nekih primera kako je zaposleni mogu koristiti:

- **Recepcija:**

  - Radnici na recepciji su u mogućnosti da proverava slobodne sobe za "danas" preko pogleda view_SobeNaRaspolaganju.

  - Omogučava im se lakše pronalaženje slobodne obe koja se uklapa u one karakteritike koje gost želi. To se postiže korišćenjem procedure sp_PredloziSobu.

  - Moguće je evidentiranje dodatnih usluga tokom boravka korišćenjem procedure sp_DodajUslugu

  - Radnici mogu da prate stanje računa u odredjenoj sobi uz pomoć funkcija fn_RacunRezime (ukupan račun na kraju boravka uz mogućnost pregleda stavki) i fn_TrenutniTrosakSobe (gde se moze videti iznos koji je potrebno platiti za boravak u sobi do trenutka pozivanja fukcije)

  - Zaposleni imaju pregled svih rezervacija u budućnosti kao i onih koje su završene korišćenjem pogleda view_PregledRezervacija.

Značajno olakšava rad i radnicima koji rade u službi za finansije.

- **Finansije:**

- Sistem knjiži uplate procedurom **`sp_EvidentirajPlacanje`** (validirano i u transakciji).
- Usklađuje saldo i prati naplatu kombinovanjem **`fn_RacunRezime`** i pregleda rezervacija.

Radnici koji se bave menadžmentom mogu Izvlači pregled popunjenosti i strukturu prihoda iz postojećih pogleda/funkcija (spremno za izvoz u Excel) kao i da prate trendove dodatnih usluga i politike cena (oslanjajući se na standardizovane upite).

U praksi, recepcija i finansije svakodnevno rade kroz spremne poglede, funkcije i procedure, dok menadžment dobija brze i precizne preseke poslovanja — sve u jedinstvenoj bazi koja se lako održava i širi.

Funkcionalnost

1. Konceptualni model

● Gosti

○ **Opis:** Predstavlja fizička lica koja koriste usluge hotela.
○ **Atributi:** jedinstveni identifikator gosta; ime; prezime; kontakt podaci (telefon, email).
○ **Poslovna pravila:** po mogućnosti jedinstveni kontakt (telefon/email) po gostu; gost može imati više rezervacija kroz vreme; podaci se čuvaju istorijski.

● Sobe

○ **Opis:** Predstavlja smeštajne jedinice u hotelu.
○ **Atributi:** jedinstveni identifikator sobe; broj sobe (jedinstven u hotelu); sprat; tip kreveta; osnovna cena po noći; status (npr. slobodna, zauzeta, van upotrebe).
○ **Poslovna pravila:** soba može biti predmet mnogih rezervacija kroz različite datume; status se koristi operativno (raspoloživost, kvar, renoviranje).

● Zaposleni

○ **Opis:** Predstavlja osoblje koje radi u hotelu.

○ **Atributi:** jedinstveni identifikator zaposlenog; ime; prezime; pozicija; kontakt podaci; datum zaposlenja; indikator aktivan/neaktivan.
○ **Poslovna pravila:** zaposleni mogu evidentirati rezervacije i dodatne usluge; istorija zaposlenih se čuva (ne brišu se zapisi već se deaktiviraju).

● **Rezervacije**

○ **Opis:** Predstavlja period boravka gosta u određenoj sobi.
○ **Atributi:** jedinstveni identifikator rezervacije; referenca na gosta; referenca na sobu; referenca na zaposlenog koji je evidentirao; datum prijave; datum odjave; broj noćenja; broj gostiju; status (rezervisano/prijavljen/odjavljen/otkazano); datum kreiranja.
○ **Poslovna pravila:** datumi moraju imati smisla (odjava posle prijave); broj noćenja je najmanje 1 i usklađen s datumima; za istu sobu rezervacije se ne smeju vremenski preklapati (osim dodirivanja na granici); jedna rezervacija može imati više plaćanja i više dodatnih usluga.

● Plaćanja

○ **Opis:** Evidentira uplate vezane za konkretnu rezervaciju.
○ **Atributi:** jedinstveni identifikator plaćanja; referenca na rezervaciju; iznos; metoda plaćanja (npr. gotovina, kartica, transfer); datum plaćanja; valuta.
○ **Poslovna pravila:** iznos je pozitivan; može postojati više delimičnih uplata; plaćanja se ne knjiže na otkazane rezervacije; zbir uplata ne bi trebalo da premaši ukupno zaduženje.

● Usluge

○ **Opis:** Evidentira dodatne usluge tokom boravka (npr. doručak, spa, minibar).
○ **Atributi:** jedinstveni identifikator usluge; referenca na rezervaciju; referenca na zaposlenog koji je uneo stavku; datum usluge; opis; količina; jedinična cena.
○ **Poslovna pravila:** količina > 0; cena ≥ 0; datum usluge u pravilu pripada periodu boravka; više usluga može biti vezano za jednu rezervaciju; svaku uslugu unosi jedan zaposleni.

2. Logički model

- **Gosti** :
  - `id_gosta` -> primarni ključ
  - Ostale kolone : ime, prezime, telefon, email

- **Sobe** :
  - `id_sobe` -> primarni ključ
  - Ostale kolone : broj_sobe, sprat, tip_kreveta, osnovna_cena, status

- **Zaposleni** :
  - `id_zaposlenog` -> primarni ključ
  - Ostale kolone : ime, prezime, pozicija, telefon, email, datum_zaposlenja, aktivan

- **Rezervacije** :
  - `id_rezervacije` -> primarni ključ
  - `id_gosta` -> spoljni ključ
  - `id_sobe` -> spoljni ključ
  - `id_zaposlenog` -> spoljni ključ
  - Ostale kolone : datum_prijave, datum_odjave, broj_nocenja, broj_gostiju, status, datum_kreiranja

● **Placanja** :
- ○ `id_placanja` -> primarni ključ
- ○ `id_rezervacije` -> spoljni ključ
- ○ Ostale kolone : iznos, metoda, datum_placanja, valuta

● **Usluge** :
- ○ `id_usluge` -> primarni ključ
- ○ `id_rezervacije` -> spoljni ključ
- ○ `id_zaposlenog` -> spoljni ključ
- ○ Ostale kolone : datum_usluge, opis_usluge, kolicina, jedinicna_cena

Veze i razlozi:

- **Gosti – Rezervacije -> 1 : N**
  Razlog: Jedan gost može imati više rezervacija tokom vremena, dok svaka rezervacija pripada tačno jednom gostu.

- **Sobe – Rezervacije -> 1 : N**
  Razlog: Jedna soba može biti predmet više rezervacija u različitim terminima, a svaka rezervacija se odnosi na jednu sobu.

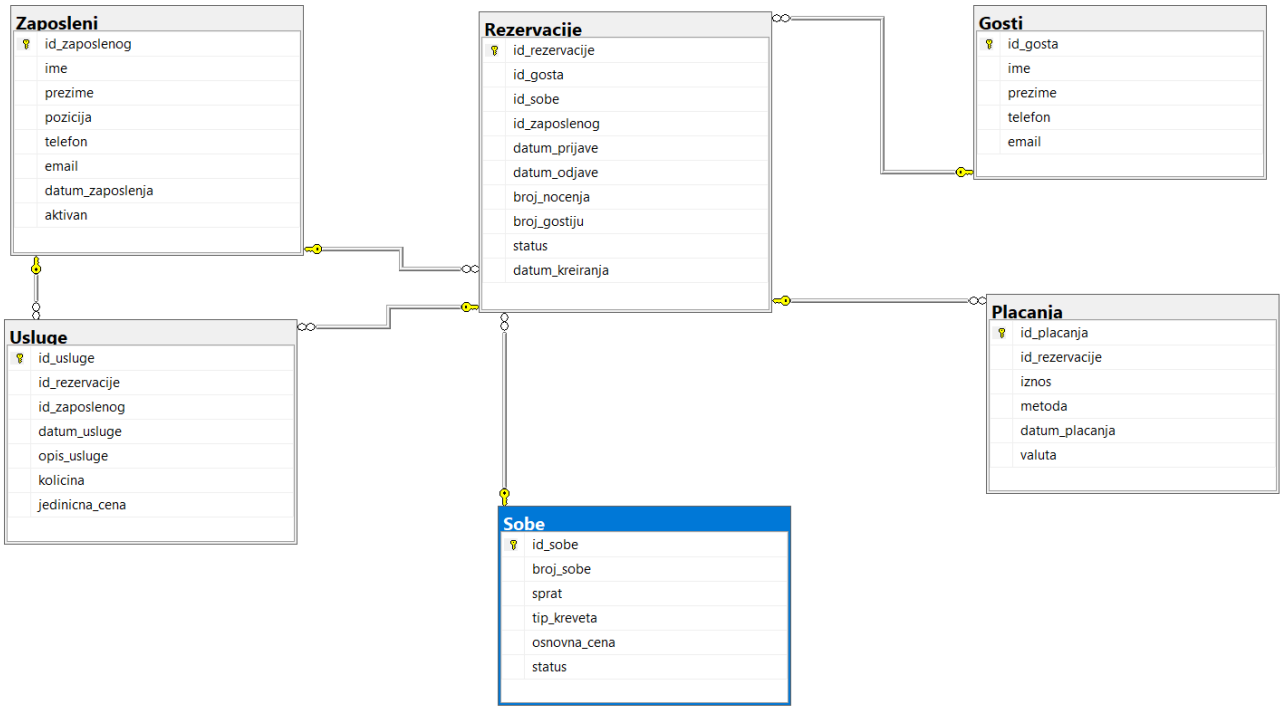
- **Zaposleni – Rezervacije -> 1 : N**
  Razlog: Jedan zaposleni može evidentirati više rezervacija; svaka rezervacija (u našem modelu) ima najviše jednog zaposlenog zaduženog za unos.

- **Rezervacije – Placanja -> 1 : N**
  Razlog: Za jednu rezervaciju može biti više delimičnih uplata; svako plaćanje je vezano za konkretnu rezervaciju.

- **Rezervacije – Usluge -> 1 : N**
  Razlog: Tokom boravka mogu se evidentirati brojne dodatne usluge; svaka usluga pripada jednoj rezervaciji.

- **Zaposleni – Usluge -> 1 : N**
  Razlog: Jedan zaposleni može uneti više stavki usluga; svaka usluga je unesena/overena od strane jednog zaposlenog.

Slika modela Baze podataka



Objekti baze podataka

1. Pogled – view_SobeNaRaspolaganju

```
CREATE OR ALTER VIEW dbo.v_SobeNaRaspolaganju
AS

WITH Aktivne AS (
    SELECT r.id_sobe
    FROM dbo.Rezervacije r
    WHERE r.status IN (N'rezervisano', N'prijavljen')
      AND CAST(GETDATE() AS date) >= r.datum_prijave
      AND CAST(GETDATE() AS date) <  r.datum_odjave
    GROUP BY r.id_sobe
)
SELECT
    s.id_sobe, s.broj_sobe, s.sprat, s.tip_kreveta,
    s.osnovna_cena, s.status
FROM dbo.Sobe s
LEFT JOIN Aktivne a ON a.id_sobe = s.id_sobe
WHERE a.id_sobe IS NULL
      AND (s.status IS NULL OR s.status NOT IN
      (N'zauzeta', N'van upotrebe'));
GO
```

Opis:

Ovaj prikazuje sve sobe koje su danas slobodne. Najpre, u CTE delu „Aktivne“ izdvajaju se sobe koje su danas zauzete nekom rezervacijom, i to samo rezervacije sa statusom „rezervisano“ ili „prijavljen“, pri čemu se današnji datum posmatra kao u intervalu od datuma prijave zaključno sa jučerašnjim datumom odjave (tj. dan odjave se smatra slobodnim). Zatim se lista svih soba spaja sa tim skupom zauzetih i zadržavaju se samo one sobe koje se ne nalaze u listi „Aktivne“, odnosno koje danas nisu pokrivene nijednom aktivnom rezervacijom. Na kraju se dodatno isključuju sobe koje su ručno označene kao „zauzeta“ ili „van upotrebe“, tako da rezultat čine isključivo sobe raspoložive za današnji dan, sa osnovnim podacima (ID, broj, sprat, tip kreveta, osnovna cena i status). Ako bi trebalo da se soba tretira kao zauzeta i na sam dan odjave, uslov u CTE delu može se promeniti tako da današnji datum bude manji ili jednak datumu odjave.

2.Funkcija fun_RezervacijeZaGosta

```
CREATE FUNCTION dbo.fn_RezervacijeZaGosta
(
    @id_gosta INT
)
RETURNS TABLE
AS
RETURN
(
    SELECT
        r.id_rezervacije, r.id_gosta,
        g.ime AS ime_gosta, g.prezime AS prezime_gosta,
        r.id_sobe, s.broj_sobe, r.datum_prijave,
        r.datum_odjave, r.broj_nocenja,
        r.broj_gostiju, r.status
    FROM dbo.Rezervacije r
    JOIN dbo.Gosti g ON g.id_gosta = r.id_gosta
    JOIN dbo.Sobe  s ON s.id_sobe  = r.id_sobe
    WHERE r.id_gosta = @id_gosta
);
GO
```

Opis:

Ova funkcija je inline tabelarna funkcija koja, za prosledjeni @id_gosta, vraća tabelarni rezultat sa svim rezervacijama tog gosta.

Kako radi: iz tabele Rezervacije uzima sve redove gde je r.id_gosta = @id_gosta, pa ih spaja sa Gosti (da bi vratio ime i prezime gosta) i sa Sobe (da bi vratio broj sobe). Rezultat sadrži sledeće kolone: id_rezervacije, id_gosta, ime_gosta, prezime_gosta, id_sobe, broj_sobe, datum_prijave, datum_odjave, broj_nocenja, broj_gostiju, status. Ako gost nema rezervacije (ili ne postoji), funkcija vraća 0 redova. Ne filtrira po statusu niti po datumu — vraća sve rezervacije tog gosta kakve god da su.

3. Procedura sp_PredloziSobu

```
CREATE PROCEDURE dbo.sp_PredloziSobu
    @od       DATE,
    @do       DATE,
    @tip_kreveta NVARCHAR(30) = NULL,
    @max_cena   DECIMAL(10,2) = NULL,
    @id_sobe INT OUTPUT,
    @poruka  NVARCHAR(200) OUTPUT
AS
BEGIN
    SET NOCOUNT ON;

    IF @od IS NULL OR @do IS NULL
    BEGIN
        RAISERROR (N'Parametri @od i @do su obavezni.', 11, 1);
        RETURN;
    END

    IF @do <= @od
    BEGIN
        RAISERROR (N'Datum odjave mora biti strogo veći od datuma prijave.', 11, 1);
        RETURN;
    END

    SET @id_sobe = NULL;

    SELECT TOP (1) @id_sobe = s.id_sobe
    FROM dbo.Sobe s
    WHERE (s.status IS NULL OR s.status NOT IN (N'van upotrebe'))
      AND (@tip_kreveta IS NULL OR s.tip_kreveta = @tip_kreveta)
      AND (@max_cena IS NULL OR s.osnovna_cena <= @max_cena)
      AND NOT EXISTS (
            SELECT 1
            FROM dbo.Rezervacije r
            WHERE r.id_sobe = s.id_sobe
              AND r.status IN (N'rezervisano', N'prijavljen')
              AND NOT (@do <= r.datum_prijave OR @od >= r.datum_odjave)
      )
    ORDER BY s.osnovna_cena, s.sprat, s.broj_sobe;

    IF @id_sobe IS NULL
        SET @poruka = N'Nema slobodnih soba koje ispunjavaju uslove u
        traženom terminu.';
    ELSE
        SET @poruka = N'Predložena soba je pronađena.';

END
GO
```

Opis:

Ova procedura pokušava da pronađe i predloži jednu slobodnu sobu za zadati termin. Prima obavezne datume prijave i odjave, a opciono možeš suziti izbor tipom kreveta i maksimalnom cenom. Na početku proverava da su datumi prosleđeni i da je odjava strogo posle prijave; u suprotnom prijavljuje grešku preko RAISERROR. Zatim postavlja izlazni parametar @id_sobe na NULL i traži prvu sobu koja ispunjava uslove: soba ne sme biti označena kao „van upotrebe“, ako je tražen tip kreveta mora da se poklopi, ako je zadat limit cene osnovna cena mora biti ≤ od tog limita, i ne sme postojati rezervacija iste sobe sa statusom „rezervisano“ ili „prijavljen“ koja se vremenski preklapa sa traženim intervalom. Preklapanje se proverava uslovom NOT (@do <= r.datum_prijave OR @od >= r.datum_odjave), što znači da je dozvoljen slučaj „check-out = check-in“ (dan odjave jedne rezervacije može biti isti kao dan prijave sledeće). Ako više soba ispunjava uslove, bira najjeftiniju (sortirano po osnovna_cena, pa sprat, pa broj_sobe) pomoću TOP (1). Na kraju, ako je soba nađena, @id_sobe dobija njen ID i @poruka se postavlja na „Predložena soba je pronađena.“; ako nije, @id_sobe ostaje NULL, a poruka kaže da nema slobodnih soba koje ispunjavaju uslove u traženom terminu.

4. Triger trg_Usluge_Insert

```
CREATE TRIGGER dbo.trg_Usluge_Insert
ON dbo.Usluge
AFTER INSERT
AS
BEGIN
    SET NOCOUNT ON;

    IF EXISTS (
        SELECT 1
        FROM inserted i
        WHERE (i.kolicina IS NULL OR i.kolicina <= 0)
           OR (i.jedinicna_cena IS NULL OR i.jedinicna_cena < 0)
           OR (i.datum_usluge IS NULL)
    )
    BEGIN
        RAISERROR (N'Neispravna stavka usluge: količina > 0, cena ≥ 0,
        datum_usluge nije NULL.', 11, 1);
        ROLLBACK TRANSACTION;
        RETURN;
    END

    IF EXISTS (
        SELECT 1
        FROM inserted i
        JOIN dbo.Rezervacije r ON r.id_rezervacije = i.id_rezervacije
        WHERE r.status IN (N'otkazano', N'odjavljen')
    )
    BEGIN
        RAISERROR (N'Nije dozvoljeno dodavanje usluge na otkazanu ili
        odjavljenu rezervaciju.', 11, 1);
        ROLLBACK TRANSACTION;
        RETURN;
    END
END
```

Opis:

Ovaj okidač se izvršava posle svakog INSERT nad tabelom Usluge i služi kao poslovna validacija koja po potrebi poništava unos.

Prvo proverava da li je u novounetim redovima količina prazna ili ≤ 0, da li je jedinična cena prazna ili < 0 i da li je datum usluge NULL. Ako ijedan red krši ova pravila, podiže grešku (RAISERROR) i radi ROLLBACK TRANSACTION, čime se ceo unos poništava. Zatim proverava status rezervacije na koju se usluga odnosi: ako je rezervacija „otkazano“ ili „odjavljen“, takođe podiže grešku i vraća transakciju. SET NOCOUNT ON samo sprečava ispis “(N row(s) affected)”. Triger radi i za višeredne inserte (ako jedan red ne valja, poništi se ceo batch). Ne menja nikakve iznose niti račune—isključivo sprečava neispravne unose prema navedenim pravilima.

Zaključak

Projekat je uspešno realizovao relacionu bazu podataka za poslovanje hotela u SQL Server-u, pokrivajući ključne procese: goste, sobe, rezervacije, usluge i plaćanja. Implementirani pogledi, funkcije, procedure i okidači obezbeđuju konzistentnost podataka, ubrzavaju operativni rad (provera dostupnosti, knjiženje usluga/plaćanja) i omogućavaju transparentan obračun i izveštavanje. Test podaci i jasno definisana pravila (statusi, datumi, broj noćenja) potvrđuju funkcionalnost i održivost rešenja. Arhitektura je modularna i spremna za proširenja—popusti, kategorije usluga, napredni izveštaji, korisničke uloge i eventualni indeksi za performanse—čime baza predstavlja čvrstu osnovu za dalji razvoj sistema.