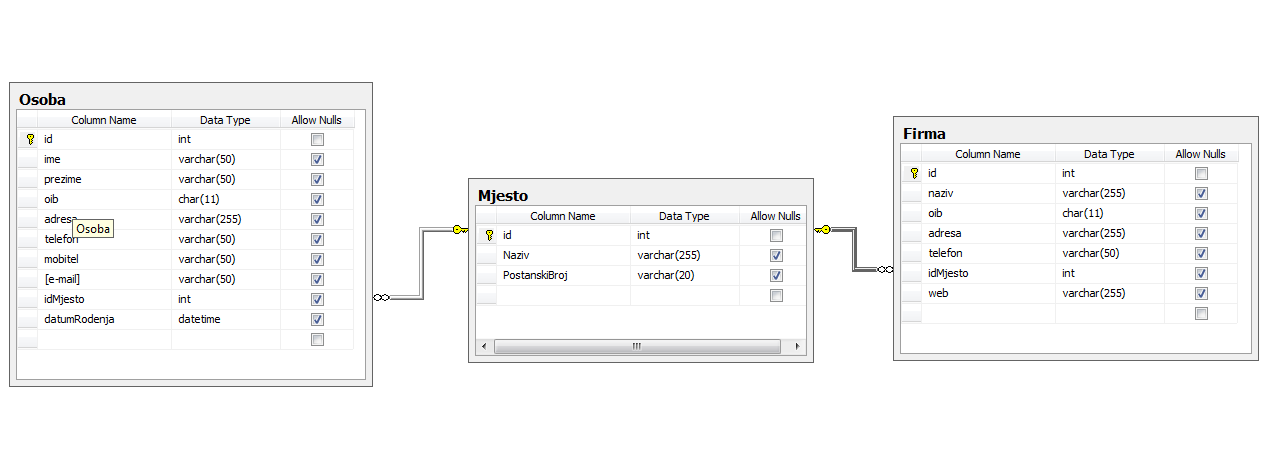
**Modeliranje – Dizajn baze na SQL serveru**

Otvorite Management studio i pristupite bazi pod nazivom fakultet\_korisnickoime (Za Peru Perića baza se zove fakultet\_peperic). Kreirajte u bazi tablice Osoba, Firma i Mjesto upotrebom standardnog designa. Kreirajte i primarne ključeve  


Tablica Osoba ima polja: id (int, identity), ime – varchar, prezime – varchar, oib – char(11), adresa, telefon, mobitel, e-mail (varchar), idMjesto – int i datumRodenja – date.

Tablica Firma ima polja: id, naziv, oib, adresa, telefon, idMjesto, web.

Tablica Mjesto ima polja: id, naziv, postanskiBroj

**Relacije**

1. **Relacija jedan na više (one-to-many)**

Kreirajte novi database diagram i iskoristite ga da povežete tablice u relacije (veza s tablicom Mjesto).

Unesite par zapisa u tablice preko management studija. Prvo unesite podatke u Mjesto, zatim u ostale tablice.

Isprobajte kako funkcionira referencijalni integritet:

* 1. Unesite u osobu ili firmu idMjesta koje ne postoji u tablici Mjesto
  2. Pokušajte izbrisati Mjesto koje se koristi u osobi ili mjestu

1. **Relacija više na više (many-to-many)**

Bazu proširimo sa još dvije tablice – Predmet i tablicom za vezu predmeta i osobe.

Predmet ima ova svojstva: id, naziv, ECTS (int), godina (int)

Kreirajte tablicu za vezu i uspostavite relaciju many-to-many između predmeta i osobe. Unesite par zapisa u obje tablice.

1. **Relacija jedan na jedan (one to one)**

Bazi dodajmo tablice – student i predavač.

Podaci koje želimo voditi za studente su: id, ime, prezime, adresa, oib, telefon, mobitel, e-mail, idMjesto, datumRodenja, datumUpisa, maticniBroj, napomena.

Podaci koje želimo voditi za predavače su: id, ime, prezime, adresa, oib, telefon, mobitel, e-mail, idMjesto, datumRodenja, datumZaposlenja, nastavnoZvanje, ziroRacun

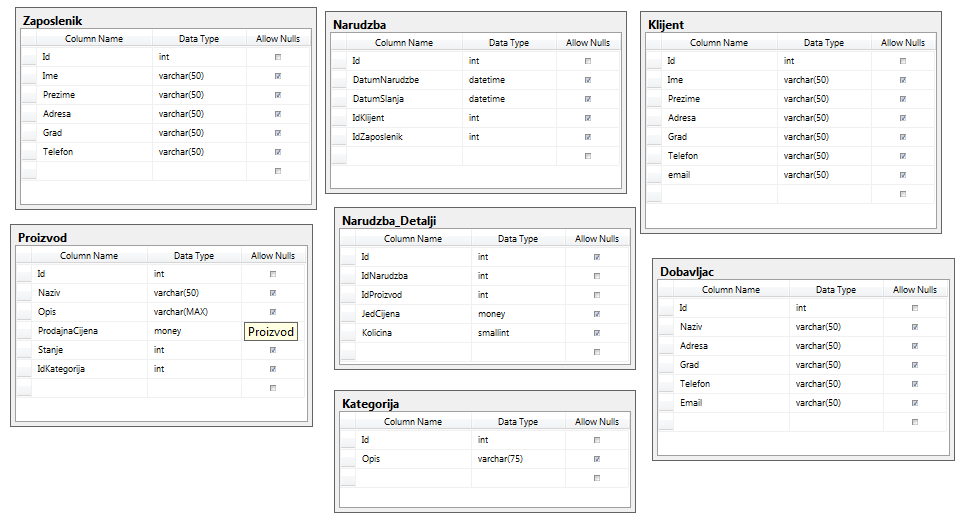
Dodajte nove tablice implementacijom relacije 1 na 1 tako da se izbjegne nepotrebno ponavljanje podataka. Isprobajte unos podataka u tablice.

**Check constrainti, defaulti i izračunata polja**

1. Postavite da su ime i prezime obavezna polja u osobi.
2. U tablicu osoba dodajte provjeru (check constraint) koji će onemogućiti unos u polje oib ako tekst nema točno 11 znakova. Koristite LEN funkciju SQL servera.
3. Dodajte i provjeru datuma rođenja tako da upisana osoba ne može biti mlađa od 18 godina (koristite funkcije DATEADD i GETDATE)
4. Dodajte u tablicu student default datumUpisa da bude današnji datum.
5. Dodajte u tablicu Osoba izračunato polje Godina koje će izračunavati starost iz upisanog datuma rođenja. Koristite SQL funkciju DATEDIFF i GETDATE().

**Vježba – struktura baze podataka**

1. Kreirajte bazu Narudzbe za evidenciju narudžbi nekog zamišljenog dućana. Tablice u bazi su prikazane na slici:



1. Dodajte primarne ključeve i relacije među tablicama. Tablica Dobavljac zasad nije u relaciji niti s jednom tablicom.
2. Postavite tablicu dobavljac u relaciju s proizvodom u dvije varijante:
   1. Proizvod ima samo jednog dobavljača
   2. Proizvod može imati više dobavljača
3. Dodajte sljedeća ograničenja na vrijednosti polja (obavezna polja i check constrainte):
   1. Naziv proizvoda u tablici Proizvod je obavezno polje
   2. Naziv dobavljača je obavezno polje
   3. Cijena proizvoda mora biti veća od 0 (u tablicama proizvod i narudzba\_detalji)
   4. Količina proizvoda na narudžbi ne smije biti 0 (smije biti negativan broj)
   5. Datum narudžbe i datum slanja u tablici Narudzba ne smiju biti u budućnosti
4. Dodajte u tablicu Narudzba izračunato polje VrijemePripreme koje predstavlja razliku Datuma slanja i datuma narudžbe.
5. Dodajte u tablicu Narudzba da je default vrijednost za polje DatumNarudzbe današnji dan.

**SELECT naredba (WHERE, ORDER BY)**

Osnovni oblik naredbe SELECT je:

SELECT polja ili \* FROM tablica WHERE uvjet ORDER BY polja [DESC]

Uvjeti mogu biti:

* >,<,=, <>, AND OR
* BETWEEN
* IN
* LIKE
* IS NULL

Koristeći bazu Narudzbe napišite sljedeće upite :

1. Izlistaj sve narudžbe zaprimljene između 1.8.2011 i 1.9.2013". Zadatak ima dva moguća rješenja. Jedno je korištenjem dva izraza spojenih logičkim operatorom AND a drugo je korištenjem operatora BETWEEN. Napišite obje varijante upita. Rezultate poredajte po datumu narudžbe. (782)
2. Izlistaj sve narudžbe poslane u 9. i 10. mjesecu 2011. Riješiti upotrebom BETWEEN, logičkih operatora i pomoću operatora IN. Sortirati po datumu slanja. (287)
3. Izlistaj sve proizvode iz kategorije 1 čija prodajna cijena na premašuje 300. Poredati po cijeni uzlazno. (11)
4. Izlistaj sve proizvode iz kategorija 2 ili 4 čija prodajna cijena pada u raspon od 3500-7500. (2)
5. Prikaži klijente čije prezime počinje sa P. Poredaj po prezimenu pa po imenu. (7)
6. Prikaži klijente čija adresa sadrži pojam Drive. (5)
7. Prikaži klijent koji imaju upisan grad (23)
8. Prikaži klijente koji nemaju upisan grad (5)

**SELECT sa izrazima**

Tekstualna polja se povezuju u izraze korištenjem operatora + uz eventualnu konverziju. Ako je u izrazu jedna vrijednost NULL, cijeli izraz je NULL što nekad nije željeno ponašanje.

Funkcija COALESCE pretvara NULL u neku drugu vrijednost tako da izraz ne postane NULL ako je vrijednost NULL.

1. Ispišite sve klijente tako da se ime i prezime ispisuje kao jedan stupac. Unesite nakon toga klijenta koji nema ime ili prezime pa pokrenite ponovno upit. Kako se ispisuje izraz? Poboljšajte izraz upotrebom funkcije COALESCE. (28-29)
2. Ispiši zapise iz tablice narudzba\_detalji tako da se osim polja JedCijena i Kolicina vidi i ukupni iznos koji se dobije množenjem ta dva polja (4231). Kako bismo poredali po ukupnom iznosu?
3. Ispiši zapise iz tablice Narudzba tako da se osim standardnih polja vidi i izračunato polje VrijemePripreme koje predstavlja razliku između datuma slanja i datuma narudžbe u danima. Riješite zadatak upotrebom funkcije CAST a nakon toga i upotrebom funkcije DATEDIFF. (947)

**SELECT – baza Agencija**

Koristeći bazu Agencija napišite sljedeće upite :

1. Ispiši klijente kojima poštanski broj završava na 006 (5)
2. Ispiši agente koji žive u Redmondu i rade za proviziju veću od 0,2 (1)
3. Ispiši angažmane koji su bili ugovoreni u 7. mjesecu 2016 i kojima je ugovorena cijena bila veća od 2000 (3)
4. Ispiši angažmane i uz svakog razliku u danima između početka i kraja (131)

**SELECT naredba – CROSS JOIN**

Upit koji sadrži više tablica koje nisu u relaciji rezultira tzv. CROSS JOIN-om kod kojeg se svaki zapis iz jedne tablice kombinira sa svakim zapisom iz druge tablice.

Napišite SELECT koji će izlistati sve proizvode iz tablice Proizvod. Koliko ima proizvoda?

Napišite također SELECT koji će izlistati kategorije. Koliko ih ima?

Napišite sljedeći SQL:

SELECT Proizvod.[Id], Proizvod.[Naziv], Proizvod.[Opis], Proizvod.[ProdajnaCijena], Proizvod.[Stanje], Proizvod.[IdKategorija], Kategorija.id, Kategorija.Opis

FROM [Proizvod] CROSS JOIN Kategorija

Koliko je izlistanih zapisa?

Ako bismo htjeli izlistati proizvode i kraj svakog naziv njegove kategorije, koje bismo zapise trebali povezati i po kojem polju?

**SELECT naredba – INNER JOIN**

Kod INNER JOIN-a tablice se povezuju preko polja koje je najčešće strani ključ i tipa int. Vraćaju se samo zapisi kod kojih zapis s tim poljem postoji u obje tablice. To je tzv. „strogi“ JOIN.

Na bazi agencija napišite sljedeće upite:

1. Ispiši podatke o agentu (ime, prezime) i datum početka i završetka angažmana koji je organizirao. Rezultate poredaj po datumu početka angažmana. (131)
2. Proširi upit iz točke a tako da se ispisuje i naziv izvođača koji je bio angažiran. (131)
3. Ispiši podatke o klijentu (ime, prezime), izvođaču kojeg je angažirao, datumu i vremenu početka i kraja angažmana, duljini angažmana u satima, ugovorenoj cijeni. Rezultate poredaj po klijentu a zatim po ugovorenoj cijeni silazno. (131)
4. Ispiši podatke o izvođaču (naziv, web, email) i podatke o njegovom angažmanu (datum, iznos) i to samo za izvođače kojima je žanr Standards ili Salsa. Rezultate poredati po nazivu izvođača a unutar naziva po datumu početka angažmana silazno. (50)
5. Ispiši agente i izvođače koji žive na istom poštanskom broju. (10)

**SELECT naredba – OUTER JOIN**

Kod OUTER JOIN-a zapis na mora postojati u obje tablice. To je tzv. „labavi“ JOIN.

Koristeći bazu Agencija napišite sljedeće upite:

1. Ispiši sve izvođače i njihove angažmane, dakle i one koji nemaju niti jedan angažman. (14)
2. Ispiši izvođače koji nisu imali niti jedan nastup. Riješi kao LEFT i kao RIGHT JOIN. (1)
3. Ispiši agente koji nisu obavili niti jedan angažman. (2)
4. Ispiši žanrove koje nitko ne preferira. (6)

**SELECT – ponavljanje**

**Baza Narudzbe**

1. Ispiši proizvode kojima cijena nije između 300 i 600 a stanje je veće od 30. Neka se uz svaki proizvod vidi i naziv kategorije. (4)
2. Ispiši proizvode kojima je cijena u rasponu od 100-300 ili od 1000-2000. Neka se vide sljedeći podaci: id, naziv, prodajna cijena i naziv kategorije. Poredaj po nazivu. (21)
3. Ispiši dobavljače koji u adresi imaju pojam ave (3)
4. Ispiši dobavljače koji imaju upisan e-mail. (6)
5. Ispiši klijente koji nemaju narudžbi (2)
6. Ispiši proizvode koji nisu naručivani (2)
7. Ispiši narudžbe koje su zaprimljene u 8 ili 12 mjesecu 2011. Uz podatke o narudžbi neka se vidi ime i prezime klijenta za kojeg je napravljena narudžba. Poredati po datumu silazno. (315)

**Baza Agencija**

1. Ispiši angažmane kojima je cijena u rasponu od 1000-3000 i koji su obavljeni u 8, 10 ili 12. mjesecu 2016. Neka se uz podatke o angažmanu vidi i ime i prezime agenta koji ga je ugovorio kao jedan stupac te naziv izvođača. (30)
2. Ispiši žanrove koje ne izvodi niti jedan izvođač (9)
3. Ispiši izvođače koji nemaju članove (1)
4. Ispiši izvođače i imena i prezimena članova za one izvođače koji sviraju žanr koji u nazivu ima riječ rock. Podaci o izvođaču će se ponavljati. (11)

**SELECT naredba – agregatne funkcije**

Agregatne funkcije izračunavaju totale na skupu zapisa. Najčešće korištene su: COUNT, SUM, AVG, MIN i MAX.

1. Koristeći bazu Agencija napišite sljedeće upite:
   1. Ispišite broj klijenata. (16)
   2. Ispišite ukupnu vrijednost svih angažmana. (151075)
   3. Ispišite datum najranije ugovorenog angažmana. (29.6.2016)
   4. Ispišite datum najkasnije ugovorenog angažmana (29.12.2016)
   5. Ispišite broj, ukupni iznos i prosječnu cijenu angažmana ugovorenih u 8. i 9. mjesecu 2016



* 1. "Ispišite broj i ugovorenu cijenu najranije ugovorenog angažmana."  
     
  2. Ispišite broj angažmana klijenata koji žive u mjestu Bellevue. (40)

1. Koristeći bazu WebShop napišite sljedeće upite:
   1. Ispišite prosječnu cijenu bicikala iz podkategorije Mountain bikes. (1624,99)
   2. Ispišite cijenu najskupljeg proizvoda iz podkategorije Handlebars. (93,06)
   3. Ispišite cijenu i naziv najskupljeg proizvoda u dućanu. (Minipump, 17100)
   4. Ispišite naziv i cijene proizvoda iz podkategorije Road bikes čija je cijena barem 20 % veća od prosječne cijene svih proizvoda te podkategorije (13)

**SELECT naredba – GROUP BY**

GROUP BY služi da se zapisi grupiraju po nekom kriteriju i onda se na toj grupi izvršavaju agregatne funkcije. Omogućuje izračun subtotala.

1. Koristeći bazu Agencija napišite sljedeće upite:
   1. "Ispiši nazive agenata, zbroj svih ugovorenih cijena angažmana i ukupnu proviziju po agentu. Ispisati i agente bez angažmana a NULL vrijednosti pretvoriti u 0. (10)
   2. Ispiši agente koji su ugovorili angažmana vrijednosti veće od 10000. Uz ime agenta ispiši i taj iznos. (7)
   3. Ispiši za svakog klijenta ukupnu cijenu angažmana, datum prvog i datum zadnjeg angažmana. Null vrijednosti pretvori u 0 a kod datumskih ispisati „nema podataka“. Poredaj po cijeni silazno. (16)
   4. Ispiši za svaki žanr broj klijenata koji ga preferiraju. Za žanrove koje nitko ne preferira ispisati 0. (26)
   5. Ispiši za svaki žanr broj angažmana i ukupnu vrijednost. Zadatak riješi preko klijenata i preko izvođača.
   6. "Koji agenti su ugovorili angažmana u ukupnoj vrijednosti većoj od 3000 u prosincu 2016?. Ispisati naziv agenta i ukupnu vrijednost. " (4)
2. Koristeći bazu WebShop napišite sljedeće upite:
   1. Ispiši za svaku godinu u kojoj je dućan radio ukupni iznos narudžbi, minimalni iznos i maksimalni iznos na narudžbi. Poredati po iznosu narudžbi silazno. (3)
   2. Ispiši klijente (id, ime, prezime) i uz svakog ukupni iznos narudžbi koje je naručio te broj njegovih narudžbi. Pritom treba prikazati samo klijente čiji je ukupni iznos barem 20% veći od prosječnog iznosa narudžbe. (3237)
   3. Ispiši podkategorije i uz svaku prosječno vrijeme pripreme narudžbe za tu podkategoriju. Vrijeme pripreme je razlika datuma slanja i datuma narudžbe. Predvidi da ima podkategorija za čije proizvode nije bilo narudžbi. (37)
   4. Ispiši Narudžbe koje su imale samo jednu stavku. (9198)
   5. Izlistaj proizvode i uz svakog ukupnu količinu i novčani iznos koji je ušao na skladište. (509)

**SELECT naredba – tablični podupiti (IN, EXISTS)**

1. Koristeći bazu WebShop napišite sljedeće upite upotrebom običnog JOINA i nakon toga korištenjem tabličnih podupita IN i EXISTS.
   1. Ispiši proizvode iz podkategorija koje u nazivu sadrže riječi Bike (105)
   2. Ispiši narudžbe proizvoda iz podkategorije Wheels (720).
   3. Ispiši zaposlenike koji su izdavali narudžbe za proizvode iz podkategorija Headsets i brakes. (17)
2. Na bazi Agencija napišite sljedeće upite koristeći IN i EXISTS:
   1. Ispiši nazive agenata koji su posredovali za izvođače iz Redmonda (7)
   2. Ispiši nazive klijenata koji su angažirali izvođače koji sviraju žanr salsa (11)
   3. Ispiši agente koji su radili za klijente koji preferiraju žanrove koji u nazivu imaju riječ classic i to u periodu od 1.9.2016-31.10.2016 (5)

**SELECT naredba – skalarni podupiti**

1. Riješite sljedeće upite kao skalarne upite:
   1. "Ispiši proizvode i broj njihovih narudžbi". Zadatak se može riješiti kao GROUP BY pa prvo riješite kao tako a onda kao skalarni upit. Usporedite broj zapisa. (509)
   2. "Ispiši nazive kategorija zajedno sa brojem podkategorija i brojem proizvoda." Rezultate poredaj po broju proizvoda silazno. Da li se ovaj upit može riješiti kao GROUP BY? (5)
   3. "Ispiši podkategorije i uz svaku najvišu cijenu proizvoda iz podkategorije i naziv proizvoda sa najvišom cijenom. Zašto se ovakav upit ne može riješiti bez skalarnih podupita? (37)

**INSERT, UPDATE, DELETE**

1. Pomoću INSERT naredbe ubacite u tablicu Proizvod u bazi WebShop zapis sa sljedećim vrijednostima: NazivProizvoda = "Super bike", idPodkategorija = 1, Cijena = 2500. Kako biste odredili id proizvoda bez da gledate u tablicu koja je zadnja, najveća vrijednost?
2. Kreirajte u bazi tablicu NarudzbaArhiva i NarudzbaDetaljArhiva. Napišite SQL koji će kopirati narudžbe iz 2011. godine u arhivske tablice.
3. Modificirajte zadatak 2. tako da umjesto kopiranja zapise preselimo u arhivsku bazu. To znači da će oni nestati iz originalnih tablica.
4. Napišite SQL koji će svim proizvodima iz podkategorije Forks umanjiti cijenu za 10%.
5. Napišite SQL koji će umanjiti za 5% cijenu proizvodima koji nisu naručivani u 2012. godini.

**CREATE, ALTER, DROP**

1. Kreirajte skriptom tablicu Telefon sa sljedećom strukturom (koristite naredbu CREATE TABLE):
   1. id – primarni ključ
   2. idOsoba – int
   3. Broj – varchar(20)
   4. TipTelefona – varchar(20)
2. Napišite INSERT naredbe koje će ubaciti nekoliko zapisa u Telefon
3. Koristeći naredbu ALTER TABLE dodajte polje idTipTelefona (int).
4. Koristeći naredbu ALTER TABLE i ALTER COLUMN promijenite polje broj u znakovno polje sa 40 znakova
5. Koristeći CREATE TABLE napravite tablicu TipTelefona
6. Napišite INSERT naredbe koje će ubaciti nekoliko zapisa u tablicu TipTelefona
7. Koristeći ALTER TABLE i ADD CONSTRAINT dodajte referencijalni integritet između tablica TipTelefona i Telefon

Koristeći ALTER TABLE i DROP COLUMN uklonite suvišno polje TipTelefona.

**Programiranje SQL servera – T-SQL**

**Vježbe – varijable i IF naredba**

1. Napišite skriptu koja će ispisati tekst: "Ova skripta vrti se na serveru xxx. Verzija SQL servera je yyyyyyyyyyyyyyyy", gdje je xxx naziv servera a yyyyy verzija. Obje vrijednosti možete dobiti korištenjem pripadnih sistemskih varijabli. UPUTA: Radite tako da deklarirate lokalnu varijablu @poruka (sa DECLARE) i zatim joj postavite vrijednost korištenjem konstanti i sistemskih varijabli (sa SET)
2. Napišite skriptu koja će ispisati dvije poruke, ovisno o verziji service packa na serveru. Ako je postavljena zadnja verzija service packa, ispisat će se poruka 'Server je up to date.', ako nije, treba ispisati poruku 'Server treba patchati sa novijom verzijom service packa. Trenutna verzija xx, treba yy'. Zadatak treba podržavati i SQL server 2012 i SQL server 2014.   
     
   UPUTA: u zadatku treba koristiti funkciju SERVERPROPERTY koja prima kao parametar naziv propertya koji tražimo. SERVERPROPERTY('productversion') vraća verziju u formatu brojki i točkica (npr. '8.00.2039'), dok SERVERPROPERTY('productlevel') vraća nivo service packa (npr. 'SP2'). Također, treba koristiti činjenicu da verzije SQL servera 2012 imaju na početku brojku 11, dok verzije SQL servera 2014 imaju na početku brojku 12 (koristiti funkciju SUBSTRING). I za kraj, ne manje važno, posljednja verzija service packa za SQL 2012 je SP4, dok je za SQL2014 zadnji SP2.
3. Napišite skriptu koja će ispisati razliku između današnjeg datuma i datuma zadnjeg računa u tablici Ulaz u bazi WebShop (u danima). Koristite funkciju GETDATE i DATEDIFF. DATEDIFF ima oblik DATEDIFF ( datepart , startdate , enddate ) gdje je datepart d ako se radi o danima, startdate je početni a enddate završni datum za koji gledamo razliku. Međurezultate pohranite u lokalnim varijablama.

**CASE naredba**

1. Napišite skriptu koja će ispisati podatke o osobi i umjesto polja Email ispisati „Nema“ ako je vrijednost u polju 0, „Ima“ ako je vrijednost u polju 1 i „U postupku“ ako je vrijednost u polju 2. Koristite naredbu CASE.
2. Napišite skriptu koja će za trenutni datum ispisati kvartal u kojem se nalazi upotrebom naredbe CASE.

**DECLARE, SET, CASE, sistemske varijable: ponavljanje**

1. Napišite skriptu koja će pročitati sistemske varijable @@CPU\_BUSY i @@TIMETICKS i na temelju toga izračunati broj sekundi od kad je server aktivan. Broj sekundi se dobiva po formuli @@CPU\_BUSY \* @@TIMETICKS / 1000000 (Dijelimo sa 1.000.000 jer bi inače rezultat bio u mikro sekundama).   
     
   Nakon što izračuna sekunde, skripta će pronaći datum i vrijeme pokretanja servera tako što će oduzeti izračunate sekunde od trenutnog vremena. Koristiti funkcije DATEADD i GETDATE. Krajnji rezultat je ispis teksta „Server je zadnji puta pokrenut xxxxx“ gdje je xxxx datum i vrijeme pokretanja servera.
2. Napišite skriptu koja će u bazi WebShop pročitati broj narudžbi u zadnjih mjesec dana. Nakon toga će ispisati tekst „U zadnjih mjesec dana bilo je n narudžbi“ gdje je n broj narudžbi ili „U zadnjih mjesec dana nije bilo narudžbi“ ako je broj narudžbi 0.
3. Napišite SELECT naredbu koja će izlistati podatke o narudžbi (Id, DatumNarudbe, DatumSlanja, Ime i prezime klijenta) + stupac u kojem će pisati tekst online ili offline. U zadnjem stupcu će pisati „online“ ako je u polju Online 1, a ako je u tom polju 0 onda će pisati „offline“. Koristiti CASE.
4. Napišite skriptu koja će ispisati broj dana za zadani mjesec. Zadatak riješite na dva načina korištenjem obje verzije naredbe CASE.

**Okidači (triggeri)**

1. U tablice Narudzba i NarudzbaDetalj dodajte dva datumska polja – VrijemeKreiranja i VrijemePromjene. Zatim kreirajte INSERT trigger koji će kod ubacivanja novog zapisa u polje VrijemeKreiranja zapisati trenutni datum i vrijeme. Provjerite da li trigger ispravno radi (ubacite novi zapis u tablicu – ručno preko Management studia ili pisanjem INSERT naredbe)
2. Napišite UPDATE trigger koji će ažurirati polje VrijemePromjene kod svake promjene zapisa u tablici Narudzba ili NarudzbaDetalj.
3. Napišite DELETE trigger koji će obrisati račun i pripadne stavke u arhivskim tablicama (NarudzbaArhiva, NarudzbaDetaljArhiva) kada se obriše zapis iz tablice Narudzba.
4. Kreirajte u bazi WebShop tablicu Log koja ima sljedeću strukturu:

|  |  |
| --- | --- |
| idLog | Int, identity |
| NazivTablice | Varchar(128) |
| Operacija | smallint |
| idZapis | Int |

Tablica Log sprema zapise o svakoj promjeni u tablicama Narudzba i NarudzbaDetalj. Npr. Ako je dodana nova narudžba, mora se kreirati novi zapis u tablici Log s vrijednostima NazivTablice = 'Narudzba', Operacija = 1, idZapis = id nove narudžbe. Ako je zapis promijenjen, sve je isto osim što je Operacija = 2. Ako je zapis obrisan, operacija = 3. Napišite skriptu koja će kreirati tablicu Log i zatim trigger(e) koji će implementirati navedenu funkcionalnost.

**Triggeri – ponavljanje**

1. Dodajte tablici Proizvod polja VrijemeKreiranja i VrijemePromjene i napišite triggere koji će ih ažurirati.
2. Kreirajte tablicu ObrisaniProizvodi sa sljedećem strukturom:

|  |  |
| --- | --- |
| Id | Int, identity |
| idProizvod | int |
| Naziv | Varchar(50) |
| DatumVrijeme | datetime |

Napišite trigger koji će upisati zapis u tablicu ObrisaniProizvodi kad korisnik obriše zapis iz tablice Proizvod.

1. Kreirajte tablicu ProizvodPromjene sa sljedećem strukturom:

|  |  |
| --- | --- |
| Id | Int, identity |
| idProizvod | int |
| StaraCijena | Money |
| NovaCijena | Money |
| DatumVrijeme | datetime |

Napišite trigger koji će upisati zapis u ovu tablicu kad se ubaci novi zapis ili promijeni postojeći zapis u tablici proizvod, ali samo ako je došlo do promjene cijene. Kod unosa novog zapisa, StaraCijena je null.

1. Kreirajte tablicu ProizvodArhiva koja ima istu strukturu kao tablica proizvod + dodatno polje DatumVrijeme. Napišite trigger koji ubacuje stare vrijednosti zapisa iz tablice Proizvod kod ažuriranja i brisanja proizvoda i bilježi datum/vrijeme kad se to dogodilo.

**View**

1. Kreirajte sljedeće viewove u bazi WebShop:

* **vNarudzbeStarijeOdTriMjeseca** – prikazuje broj narudžbe, datum i ukupni iznos. Poredani su po datumu narudžbe uzlazno.
* **vKategorijeProizvoda** – prikazuje naziv kategorije, broj proizvoda u kategoriji i naziv i cijenu najjeftinijeg i najskupljeg proizvoda. Kategorije su poredane po abecedi.

1. Kako bi glasili sljedeći upiti (koristiti viewove iz zadatka 1)?

* "Prikaži narudžbe starije od tri mjeseca čiji je ukupni iznos manji od 300 Kn."
* "Kod kojih kategorija proizvoda je odnos cijene najskupljeg i najjeftinijeg proizvoda manji od 5"

**Stored procedure**

1. U bazi webshop kreirajte stored proceduru proc\_NarudzbeKorisnika koja će vratiti popis svih narudžbi nekog klijenta. IdKlijent se predaje proceduri kao parametar a ona vraća sve podatke o narudžbi kao i zbroj iznosa za svaku narudžbu. Narudžbe su poredane po datumu uzlazno.
2. Proširite proceduru iz zadatka 1 tako da osim što vraća popis narudžbi ujedno i vraća zbroj svih iznosa narudžbi za dotičnog klijenta.
3. Proceduri iz zadatka 1 treba dodati sljedeću logiku:
   1. Ako je idKlijent predan (nije NULL), vrati samo narudžbe tog klijenta
   2. Ako je idKlijent NULL, vrati sve datoteke
4. Istu proceduru parametrizirajte tako da se može zadati raspon datuma i login prodavača. Npr. možda bismo htjeli vratiti samo narudžbe od 1.1.2009 do 1.6.2009 koje je napravio prodavač s loginom betsy0. Kod raspona će očito trebati dva parametra – od i do. Pritom treba dozvoliti da je bilo koji od parametara NULL i shodno tome postaviti upit. Npr. ako se zada od=1.1.2009 i do = NULL, to znači da se traže narudžbe iza 1.1.2009 a da završni datum nije bitan.   
   Isto tako, kod logina mora biti omogućeno da se parametar uopće ne zada – u tom slučaju ne filtrirati narudžbe po loginu.
5. U bazi napišite proceduru proc\_OsobaInsert koja će služiti za ubacivanje nove osobe u tablicu. Parametri procedure će biti analogni poljima u tablici osoba, osim što će idosoba biti izlazni parametar, preko njega će procedura pozivatelju vratiti id novog zapisa. UPUTA: U proceduri ćete morati izračunati prvi sljedeći id, dodijeliti ga nekoj varijabli i onda ga predati INSERT naredbi koja će ubaciti vrijednosti u tablicu. Isprobajte proceduru u Management studiu.
6. Napišite verziju procedure iz zadatka 5 koja bi radila u situaciji da je idOsoba identity. UPUTA: U INSERT naredbi se ne ubacuje idOsoba, samo ostala polja. Nakon INSERT naredbe u proceduri treba napisati naredbu SET @idOsoba = SCOPE\_IDENTITY(), gdje je @idOsoba izlazni parametar procedure. Funkcija SCOPE\_IDENTITY vraća zadnji generirani identity unutar procedure a to je upravo id novo unesenog proizvoda.

**Korisnički definirane funkcije**

1. Napišite korisničku funkciju **fn**\_**Kvartal** koja iz zadanog datuma vraća redni broj kvartala. Nakon toga napišite upit koji će izlistati narudžbe i za svaki datum narudžbe ispisati kvartal koristeći funkciju **fn**\_**Kvartal**.
2. Napišite korisničku funkciju **fn\_ImePrezime** koja iz tri parametra (titula, ime , prezime) vraća rezultat u obliku titula prvo\_slovo\_imena prezime npr. za titula= Ms. Ime=Jane prezime=Greer vraća Ms. J. Greer.

**Ponavljanje – view, procedure i funkcije**

1. Kreirajte sljedeće viewove u bazi WebShop:

* **vProizvodiNarucivaniViseOd3puta** – prikazuje proizvode koji su naručivani više od 3 puta. Stupci koji se vide su svi podaci iz tablice Proizvod + ukupni iznos narudžbi za svaki proizvod i Datum zadnje narudžbe. Za dobivanje ukupnog iznosa i datuma zadnje narudžbe preporuča se skalarni podupit.   
    
  Napišite skripte sa sql upitima koji koriste ovaj view a koji vraćaju:  
    
  - proizvode kojima je ukupni iznos narudžbe veći od 1000  
  - proizvode kojima je datum zadnje narudžbe u drugoj polovici godine
* **vProizvodiStanje** – prikazuje proizvode i uz svaki broj narudžbi, broj ulaza, naručeni iznos (sa narudžbi) i nabavljeni iznos (sa ulaza).   
    
  Napišite skriptu koja će koristeći view ispisati proizvode kojima je naručeni iznos veći od nabavljenog iznosa tj. koji su u minusu na skladištu.

1. Napišite stored proceduru koja će imati funkcionalnost kao view iz zadatka 1 ali uz slanje parametara za iznos narudžbe i datum zadnje narudžbe.
2. Napišite stored proceduru koja će koristiti funkciju fn\_Kvartal da dobije kvartal datuma narudžbe. Procedura će imati parametar kvartal i vraćat će samo narudžbe u zadanom kvartalu ili sve ako je parametar null.
3. Proširite proceduru iz prethodnog zadatka tako da se može zadati raspon kvartala.
4. U bazi WebShop napravite stored proceduru proc\_Ulazi. Procedura će vraćati popis ulaza prema zadanim parametrima:
   1. Raspon za datumNarudzbe
   2. Raspon za datumSlanja
   3. idZaposlenik

Kroz poseban izlazni parametar procedura će vraćati ukupni financijski ulaz za promatrane parametre.

Svi parametri su opcioni.

1. Napisati i skriptu za testiranje stored procedure.