Sistemi baze podataka

Vežbe – Ponavljanje SQL

Sadržaj

- Rad u učionici
- SQL - Primer
- SQL - Rekapitulacija
    - Jezik za definiciju podataka (DDL)
    - Jezik za manipulaciju podacima (DML)
    - Upitni jezik – SELECT naredba

Rad u učionici

Rad u učionici

- Baze podataka
  - studentska korisnička šema (user schema)
  - pod nazivom indXY
    - username: indrXY
    - password: ftn
  - Gde je ind oznaka studijskog programa, X broj indeksa, a Y godina upisa

Rad u učionici

- Podaci potrebni za konektovanje na bazu

	MI A2-1, MI A2-2, MI A2-3	Učionice računarskog centra	Kod kuće
Username	indXY[1]	indXY[1]	*[2]
Password	ftn	ftn	*[2]
Role	default	default	default
Host Name	192.168.18.9	192.168.7.204	localhost
Port	1522	1521	1521
Oracle SID	db2016	bp1	
Service name			xepdb1

1 – ind oznaka studijskog programa, X broj indeksa, a Y godina upisa
2 – username i password koji su postavljeni tokom izvršavanja skripta za kreiranje korisnika

SQL - Primer

Primer – relacioni model

- Radnik({Mbr, Ime, Prz, Sef, Plt, God,Pre}, {Mbr}),
- Projekat({Spr, Nap, Nar, Ruk}, {Spr}),
- Radproj({Spr, Mbr, Brc}, {Spr + Mbr}),

- Radproj[Mbr] ⊆ Radnik[Mbr],
- Radproj[Spr] ⊆ Projekat[Spr],

- Projekat[Ruk] ⊆ Radnik[Mbr],
- Null(Projekat, Ruk) = ⊥

- Radnik[Sef] ⊆ Radnik[Mbr],
- Null(Radnik, Sef) = T

Tabela radnik

- Mbr - matični broj radnika
- Ime - ime radnika
- Prz - prezime radnika
- Sef - maticni broj direktno nadređenog rukovodioca - radnika
- Plt - mesečni iznos plate radnika
- God - Datum rođenja radnika
- Pre – godišnja premija na platu radnika

- Obeležja Mbr, Ime, Prz ne smeju imati null vrednost. Plata ne sme biti manja od 500

Tabela projekat

- Spr - šifra projekta
- Nap - naziv projekta
- Nar - naručilac projekta
- Ruk - rukovodilac projekta

- Obeležja Spr i Ruk ne smeju imati null vrednost, dok obeležje Nap mora imati jedinstvenu vrednost

Tabela radproj

- Spr - šifra projekta
- Mbr - matični broj radnika
- Brc - broj časova nedeljnog angažovanja na projektu

- Sva tri obeležja ne smeju da imaju null vrednost

Tabela isplate_radnicima

Relacioni model:

- Isplate_radnicima({Mbr, Mesec, Godina, Datum_isplate, Iznos, Razlog_isplate}, {Mbr, Datum_isplate, Razlog_isplate})
- Isplate_radnicima[Mbr] ⊆ Radnik[Mbr],

Opis:

- Mbr - matični broj radnika
- Mesec - naziv meseca kada je isplata izvršena
- Godina - godina kada je isplata izvršena
- Datum_isplate - datum kada je isplata izvršena
- Iznos - koliko je sredstava isplaćeno
- Razlog_isplate - razlog zbog kog je vršena isplata

Dodatne napomene:

- Nijedno obeležje ne sme imati null vrednost.
- Razlog_isplate može uzimati jednu od sledećih vrednosti:
    - 'PLATA_DEO1', 'PLATA_DEO2', 'PUTNI_TROSKOVI', 'BONUS', 'DNEVNICA'

Jezik za definiciju podataka (DDL)

Kreiranje tabele

```
CREATE TABLE [šema.]<naziv_tabele>
(<naziv_kolone> <tip_podatka> [DEFAULT izraz] [, ...]
CONSTRAINT <naziv_ogranicenja>
<definicija_ograničenja> [, ...]);
```

- Šema - poklapa se sa nazivom korisnika
- DEFAULT opcija:
    - Specificira se predefinisana vrednost za kolonu, koja se koristi ukoliko se prilikom ubacivanja podataka izostavi vrednost za tu kolonu

SQL tipovi podataka

Tip podataka	Opis
VARCHAR2(size)	Niz karaktera promenljive dužine, maksimalne dužine size; minimalna dužina je 1, maksimalna je 4000.
CHAR(size)	Niz karaktera fiksne dužine od size bajtova; default i minimalna dužina je 1, maksimalna dužina je 2000.
NUMBER(p, s)	Broj ukupnog broja cifara p, od čega je s cifara iza decimalnog zareza; p može imati vrednosti od 1 do 38.
DATE	Vrednosti za vreme i datum.
LONG	Niz karaktera promenljive dužine do 2 GB. (za kompatibilnost sa starijim verzijama Oracle-a).
CLOB	Niz karaktera promenljive dužine do 4 GB.
BLOB	Binarni podaci do 4 GB.
BFILE	Binarni podaci smešteni u eksternom fajlu do 4 GB.
ROWID	Jedinstvena adresa vrste u tabeli.

Tabela faze_projekta – Domaći

- Kreirati tabelu faze_projekta
  - faze_projekta({Spr , Sfp, Rukfp, Nafp, Datp}, {Spr+ Sfp})
    - faze_projekta[Spr] ⊆ projekat[Spr],
    - faze_projekta[Rukfp] ⊆ radnik[Mbr]

- Sfp - šifra faze projekta,
- Spr - sifra projekta,
- Rukfp - rukovodilac faze projekta,
- Nafp - naziv faze projekta,
- Datp - datum početka faze projekta

- Obeležja Spr i Sfp ne smeju imati null vrednost.
- Obeležje Nafp mora imati jedinstvenu vrednost.

Tabela faze_projekta – Domaći

```
CREATE TABLE faze_projekta (
    Spr integer not null,
    Sfp integer not null,
    Rukfp integer,
    Nafp varchar2(20),
    Datp date,
    CONSTRAINT faze_projekta_PK PRIMARY KEY (spr, sfp),
    CONSTRAINT faze_projekta_fk1 FOREIGN KEY (spr) REFERENCES projekat(spr),
    CONSTRAINT faze_projekta_fk2 FOREIGN KEY (rukfp) REFERENCES radnik(mbr),
    CONSTRAINT faze_projekta_uk UNIQUE(nafp)
);
```

Izmena definicije tabele

- ALTER TABLE
- Alter table iskaz služi za:
  - dodavanje nove kolone,
  - modifikaciju postojeće kolone,
  - definisanje podrazumevane vrednosti za novu kolonu,
  - brisanje kolone i
  - dodavanje oraničenja.

ALTER TABLE

```
ALTER TABLE <naziv_tabele>
ADD (<naziv_kolone> <tip_podatka> [DEFAULT izraz]
    [, <naziv_kolone> <tip_podatka>]...);

ALTER TABLE <naziv_tabele>
MODIFY (<naziv_kolone> <tip_podatka> [DEFAULT izraz]
        [, <naziv_kolone> <tip_podatka>]...);

ALTER TABLE <naziv_tabele>
DROP COLUMN (<naziv_kolone>);

ALTER TABLE <naziv_tabele>
ADD CONSTRAINT <naziv_ogranicenja>
<definicija_ogranicenja>;
```

Izmena definicije tabele – Domaći

- U tabelu faze_projekta dodati atribut:
  - Datz - datum završetka faze projekta
  - Datz ne sme biti manji od Datp

```
ALTER TABLE faze_projekta
     ADD datz date
     ADD CONSTRAINT dat_ch CHECK (datp<=datz);

ALTER TABLE faze_projekta
     ADD(datz date, CONSTRAINT dat_ch CHECK (datp<=datz));

ALTER TABLE faze_projekta
     ADD datz date;
ALTER TABLE faze_projekta
     ADD CONSTRAINT dat_ch CHECK (datp<=datz);
```

Brisanje definicije tabele

```
DROP TABLE <naziv_tabele>;
```

Brisanje definicije tabele – Domaći

- Izbrisati tabelu faze_projekta.

```
DROP TABLE faze_projekta;
```

Jezik za manipulaciju nad podacima (DML)

Ažuriranje baze podataka

- INSERT
- DELETE
- UPDATE

Ažuriranje baze podataka

- INSERT – dodavanje nove torke

```
INSERT INTO <naziv_tabele> [(<lista_obeležja >)]
VALUES (<lista_konstanti >) | SELECT …;
```

Ažuriranje baze podataka

- INSERT – dodavanje nove torke

```
INSERT INTO radnik (mbr, ime, prz, plt, sef, god)
VALUES (201, 'Ana', 'Jovic', 30000, null, '18-AUG-1971');

INSERT INTO projekat (spr, nap, ruk)
VALUES (90, 'P1', 201);

INSERT INTO radproj (mbr, spr, brc)
VALUES (201, 90, 5);
```

Ažuriranje baze podataka

- DELETE – brisanje postojećih torki

```
DELETE FROM <naziv_tabele>
[WHERE (<uslov_selekcije>)];
```

Ažuriranje baze podataka

- DELETE – brisanje postojećih torki

```
DELETE FROM radnik;

DELETE FROM radnik
WHERE mbr = 701;

DELETE FROM radproj;
```

Ažuriranje baze podataka

- UPDATE – modifikacija postojećih torki

```
UPDATE <naziv_tabele>
SET <obeležje>= <aritm_izraz>
{,<obeležje>= <aritm_izraz>}
[WHERE (<uslov_selekcije>)];
```

Ažuriranje baze podataka

- UPDATE – modifikacija postojećih torki

```
UPDATE radnik
SET plt = plt*1.2;

UPDATE radnik
SET plt = plt*1.2
WHERE mbr = 201;
```

Transakcija

- Najmanja jedinica obrade podataka, takva da
  - prevodi bazu podataka iz jedno u drugo (ne nužno različito) konzistentno stanje, s obzirom na implementirana ograničenja
  - sadrži operacije upita ili/i operacije ažuriranja podataka u bazi podataka
- Efekti izvođenja transakcije se, na kraju, u celosti
  - potvrđuju (COMMIT) i tada postaju vidljivi ostalim korisnicima u sistemu, ili
  - poništavaju (ROLLBACK) i ostavljaju obrađivani deo baze podataka u stanju kakvo je važilo neposredno pre početka njenog izvođenja

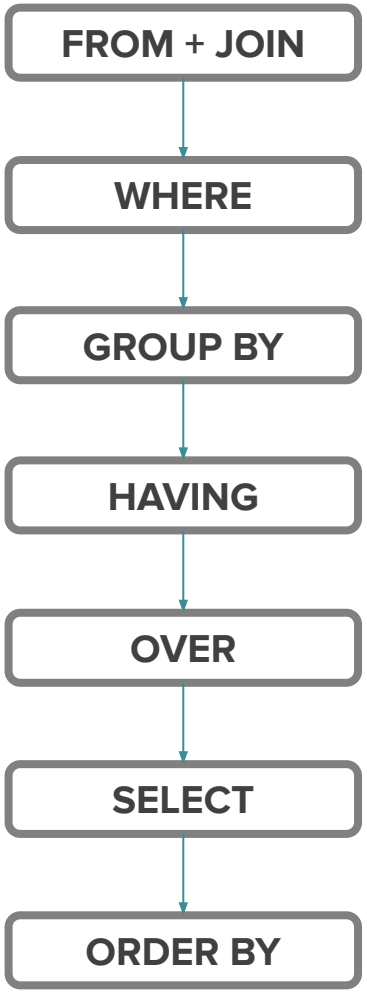
Upitni jezik – SELECT naredba

Izražavanje upita i osnovna struktura naredbe SELECT

- Sve vrste upita se u SQL-u izražavaju putem naredbe SELECT. Osnovna struktura SELECT naredbe je:

```
SELECT * | [DISTINCT] {kolona|izraz [[AS] alijas]}
FROM <lista_tabela>
[WHERE <uslov_selekcije>]
[GROUP BY <lista_obeležja>]
[HAVING <uslov_selekcije>]
[ORDER BY {kolona|izraz|alijas|redni broj [ASC|DESC]}];
```

Redosled izvršavanja klauzula



Zadatak za vežbu

- Izlistati mbr, ime, prz radnika koji rade na projektu sa šifrom 10, a ne rade na projektu sa šifrom 30.

```
SELECT mbr, ime, prz
FROM radnik
WHERE mbr IN
     (SELECT mbr FROM radproj WHERE spr = 10)
AND mbr NOT IN
     (SELECT mbr FROM radproj WHERE spr = 30);
```

Primer

- Izlistati nazive projekata na kojima se ukupno radi više od 15 časova.

```
SELECT nap
FROM projekat p, radproj rp
WHERE p.spr = rp.spr
GROUP BY p.spr, nap
HAVING SUM(brc) > 15;
```

Zadatak za vežbu

- Prikazati za svakog radnika mbr, prz, ime, ukupan broj projekata i ukupno angažovanje na projektima na kojima radi.

```
SELECT r.mbr, r.prz, r.ime, COUNT(*), SUM(rp.brc)
FROM radnik r, radproj rp
WHERE r.mbr=rp.mbr
GROUP BY r.mbr, r.prz, r.ime;
```

Zadatak za vežbu

- Izlistati nazive i šifre projekata na kojima je prosečno angažovanje veće od prosečnog angažovanja na svim projektima.

```
SELECT p.spr, p.nap
FROM projekat p, radproj rp
WHERE rp.spr=p.spr
GROUP BY p.spr, p.nap
HAVING AVG(brc) > (SELECT AVG(brc) FROM radproj);
```

Povezani upiti – Zadatak za vežbu

- Prikazati mbr, ime, prz, plt radnika čiji je broj sati angažovanja na nekom projektu veći od prosečnog broja sati angažovanja na tom projektu.

```
SELECT DISTINCT r.mbr, ime, prz, plt
FROM radnik r, radproj rp1
WHERE r.mbr = rp1.mbr AND rp1.brc > (SELECT AVG(brc)
                                     FROM radproj rp2
                                     WHERE rp2.spr=rp1.spr);
```

Prirodno spajanje (NATURAL JOIN)

- Spajanje se vrši na osnovu imena kolona
- Prikazati ime i prz radnika koji rade na projektu sa šifrom 30.

```
SELECT ime, prz
FROM radnik NATURAL JOIN radproj
WHERE spr=30;
```

Unutrašnje spajanje (INNER JOIN)

- Prikazati ime i prz radnika koji rade na projektu sa šifrom 30.

```
SELECT ime, prz
FROM radnik r INNER JOIN radproj rp ON r.mbr = rp.mbr
WHERE spr=30;
```

Spoljno spajanje (OUTER JOIN)

- Levo (LEFT)
- Desno (RIGHT)
- Potpuno (FULL)

Spoljno spajanje (LEFT OUTER JOIN)

- Prikazati mbr, ime i prz radnika i šifre projekata na kojima rade. Prikazati, takođe, iste podatke i za radnike koji ne rade ni na jednom projektu, pri čemu za šifru projekta treba, u tom slučaju, prikazati nedostajuću vrednost.

```
SELECT r.mbr,ime, prz, spr
FROM radnik r LEFT OUTER JOIN radproj rp ON r.mbr = rp.mbr;
```

Spoljno spajanje (LEFT OUTER JOIN)

- Prikazati mbr, ime i prz svih radnika i nazive projekata kojima rukovode. Ukoliko radnik ne rukovodi ni jednim projektom ispisati: ne rukovodi projektom.

```
SELECT r.mbr,ime, prz, NVL(nap, 'Ne rukovodi projektom') Projekat
FROM radnik r LEFT OUTER JOIN projekat p ON r.mbr=p.ruk;
```

Spoljno spajanje (RIGHT OUTER JOIN)

- Prikazati nazive svih projekata i mbr radnika koji rade na njima. Ukoliko na projektu ne radi ni jedan radnik ispisati nulu umesto matičnog broja.

```
SELECT NVL(rp.mbr, 0) "Mbr radnika", nap
FROM radproj rp RIGHT OUTER JOIN projekat p ON rp.spr=p.spr;

SELECT NVL(rp.mbr, 0) "Mbr radnika", nap
FROM radproj rp, projekat p
WHERE rp.spr(+)=p.spr;
```

Spoljno spajanje (FULL OUTER JOIN)

```
SELECT NVL(rp.mbr, 0) "Mbr radnika", nap
FROM radproj rp FULL OUTER JOIN projekat p ON rp.spr=p.spr;
```

Zadatak za vežbu

- Prikazati za sve radnike i projekte na kojima rade mbr, prz, ime, spr i nap. Za radnike koje ne rade ni na jednom projektu, treba prikazati mbr, prz, ime, dok za vrednosti obeležja spr i nap treba zadati, redom, konstante 0 i "Ne postoji". Urediti izlazni rezultat saglasno rastućim vrednostima obeležja mbr.

```
SELECT r.mbr, r.prz, r.ime, NVL(p.spr, 0) AS spr, NVL(p.nap, 'Ne postoji') AS nap
FROM radnik r, radproj rp, projekat p
WHERE r.mbr = rp.mbr (+) AND rp.spr = p.spr (+)
ORDER BY mbr;

SELECT r.mbr, r.prz, r.ime, NVL(p.spr, 0) AS spr, NVL(p.nap, 'Ne postoji') AS nap
FROM radnik r LEFT OUTER JOIN radproj rp ON r.mbr=rp.mbr LEFT OUTER JOIN projekat p ON rp.spr=p.spr
ORDER BY mbr;
```

Selekcioni izraz (CASE)

- Prosti (Simple) CASE:

```
CASE expr
     WHEN expr1 THEN return_expr1
     [ WHEN expr2 THEN return_expr2
     WHEN exprn THEN return_exprn]
     [ ELSE else_expr ]
END;
```

- Pretražujući (Searched) CASE:

```
CASE
     WHEN comparison_expr1 THEN return_expr1
     [ WHEN comparison_expr2 THEN return_expr2
     WHEN comparison_exprn THEN return_exprn]
     [ ELSE else_expr ]
END;
```

Zadatak za vežbu

- Za svakog radnika prikazati mbr, ime, prz, kao kategoriju kojoj pripada na osnovu visine plate. Kategorije po visini plate su sledeće:
  - Plata manja od 10000 – kategorija: **’mala primanja’**,
  - plata između 10000 i 20000 – kategorija: **’srednje visoka primanja’**,
  - plata između 20000 i 40000 – kategorija: **’visoka primanja’**,
  - plata veća od 40000 – kategorija: **’izuzetno visoka primanja’**.
- Takođe, radnike urediti prema kategoriji kojoj pripadaju, u redosledu od najniže ka najvišoj kategoriji po visini plate.

Zadatak za vežbu

```
SELECT mbr, ime, plt,
CASE
    WHEN plt < 10000 THEN 'mala primanja'
    WHEN plt >=10000 AND plt < 20000 THEN 'srednja primanja'
    WHEN plt >=20000 AND plt < 40000 THEN 'visoka primanja'
    ELSE 'izuzetno visoka primanja'
END AS visina_primanja
FROM radnik
ORDER BY
    CASE visina_primanja
        WHEN 'izuzetno visoka primanja' THEN 1
        WHEN 'visoka primanja' THEN 2
        WHEN 'srednja primanja' THEN 3
        ELSE 4
    END DESC, plt ASC;
```

Selekcioni izraz (CASE) – Napomene

- Može se iskoristiti gde god je dozvoljeno korišćenje izraza
  - Najčešće – u okviru liste kolona ili u okviru ORDER BY klauzule
- Ukoliko se ne iskoristi else, podrazumevana vrednost biće null
- Kod prostog CASE izraza, poređenje sa null vrednošću nije moguće – podrazumevano se koristi poređenje operatorom '=', pa je takav izraz uvek netačan

Zadatak za vežbu

- Za svakog radnika ispisati mbr, ime, prz, platu i mbr šefa. Pri ispisu treba obezbediti da su radnici uređeni saglasno visini plate, od najviše ka najnižoj, pri čemu bi direktor firme trebalo da se ispiše prvi.

```
SELECT mbr, ime, plt, sef
FROM radnik
ORDER BY
CASE
     WHEN sef IS NULL THEN 1
     ELSE 2
END, plt DESC;
```

Klauzula WITH

- CTE (engl. *Common Table Expressions*)
- Dodela naziva bloku podupita
- Blok može biti referenciran više puta unutar upita
  - najveća razlika u odnosu na SELECT u listi tabela
- Uvodi svojstvo sastavljanja (eng. Composability) u SQL
- Optimizacija upita
  - kao privremena tabela/umetnuti pogled
- Sintaksa

```
WITH naziv_upita AS (SELECT...);
```

WITH – Primer

- Prikazati za svakog radnika angažovanog na projektu mbr, prz, ime, spr i broj drugih radnika koji su angažovani na istom projektu.

```
SELECT r.mbr, r.ime, r.prz, rp1.spr, COUNT(rp2.mbr)-1 ostali
FROM radnik r, radproj rp1, radproj rp2
WHERE r.mbr=rp1.mbr AND rp1.spr=rp2.spr
GROUP BY r.mbr, r.ime, r.prz, rp1.spr;

WITH projinfo AS (
     SELECT rp.spr, COUNT(rp.mbr) AS rad_broj
     FROM radproj rp GROUP BY rp.spr)
SELECT r.mbr, r.ime, r.prz, rp.spr, pi.rad_broj-1 ostali
FROM radnik r, radproj rp, projinfo pi
WHERE r.mbr=rp.mbr AND rp.spr=pi.spr;
```

WITH – Zadatak za vežbu

- Prikazati za svakog radnika angažovanog na projektu mbr, prz, ime, spr i udeo u ukupnom broju časova rada na tom projektu (zaokruženo na dve decimale)

```
WITH projinfo AS (
     SELECT rp.spr, SUM(rp.brc) AS cas_suma
     FROM radproj rp
     GROUP BY rp.spr)
SELECT r.mbr, r.ime, r.prz, rp.spr, ROUND(rp.brc/pi.cas_suma, 2) udeo
FROM radnik r, radproj rp, projinfo pi
WHERE r.mbr=rp.mbr AND rp.spr=pi.spr;
```

WITH – Zadatak za vežbu

- Prikazati mbr, ime i prz rukovodilaca projekata kao i ukupan broj radnika kojima rukovode na projektima

```
WITH rukovodilac AS (
     SELECT mbr, ime, prz, plt, spr
     FROM radnik, projekat WHERE mbr=ruk),
projinfo AS (
     SELECT spr, count(mbr) ljudi
     FROM radproj GROUP BY spr)
SELECT ru.mbr, ru.ime, ru.prz, SUM(pi.ljudi) ljudi
FROM rukovodilac ru, projinfo pi
WHERE ru.spr=pi.spr
GROUP BY ru.mbr, ru.ime, ru.prz;
```

WITH – Rekurzija

- Blok podupita pomoću WITH
- Blok sadrži dva upita vezana preko UNION ALL
  - prvi upit određuje početni skup podataka
  - drugi upit obezbeđuje rekurzivno proširenje skupa putem unije sa tekućim skupom
- Postupak se zaustavlja kada ne dođe do promene skupa prilikom proširenja

```
WITH naziv_upita(lista_obeležja) as
    (
        upit1
        UNION ALL
        upit2
    )
```

WITH – Rekurzija – Primer

- Prikazati za svakog radnika sve direktno i indirektno nadređene radnike.

```
WITH hijerarhija(mbr,sef) AS (
     SELECT mbr, sef
     FROM radnik
     UNION ALL
     SELECT r.mbr, h.sef
     FROM radnik r, hijerarhija h
     WHERE r.sef = h.mbr AND h.sef IS NOT NULL
)
SELECT * FROM hijerarhija ORDER BY mbr, sef;
```

WITH – Rekurzija – Zadatak za vežbu

- Prikazati za svakog radnika sve direktno i indirektno podređene radnike.

```
WITH hijerarhija(mbr,pod) AS(
     SELECT sef, mbr
     FROM radnik
     UNION ALL
     SELECT h.mbr, r.mbr
     FROM hijerarhija h, radnik r
     WHERE h.pod = r.sef AND h.mbr IS NOT NULL
)
SELECT * FROM hijerarhija ORDER BY mbr, pod;
```

WITH – Rekurzija – Top-down obilazak – Zadatak za vežbu

- Prikazati za svakog radnika mbr, ime, prezime, njegovog šefa i njegov nivo u hijerarhiji.

```
WITH hijerarhija(mbr, ime, prz, sef, nivo) AS (
     SELECT r.mbr, r.ime, r.prz, r.sef, 1 AS nivo
     FROM radnik r
     WHERE r.sef IS NULL

     UNION ALL

     SELECT r1.mbr, r1.ime, r1.prz, r1.sef, hr.nivo+1
     FROM radnik r1
     INNER JOIN hijerarhija hr ON r1.sef = hr.mbr
)
SELECT mbr, ime, prz, sef, nivo
FROM hijerarhija
ORDER BY nivo;
```

Pogledi

- Podupiti mogu se trajno sačuvati kao pogledi
  - podupiti koji se koriste kod CTE mogu se koristiti samo dok traje naredba
- Pogodnosti koje pruža korišćenje pogleda:
  - ograničavaju pristup bazi podataka
  - obezbeđuju nezavisnost podataka
  - obezbeđuju višestruke poglede nad istim podacima
  - mogu se brisati bez uklanjanja podataka u osnovnim tabelama
- CTE se koriste za jednokratne upite, dok se pogledi koriste kada se isti podupit često koristi

Kreiranje, izmena i brisanje definicije pogleda

```
CREATE [OR REPLACE] VIEW <naziv_pogleda> [(alias [, alias]...)]
AS podupit;
```

- Podupit koji se koristi za definisanje pogleda može biti kompleksan

Modifikacija pogleda

- Pogledi se modifikuju pomoću OR REPLACE opcije (kreira se pogled, a ako pogled sa tim imenom već postoji, nova definicija zamenjuje staru).
- Dakle, pogled može biti izmenjen bez brisanja postojećeg pogleda.
- Na primer, mogu se dodati alijasi za kolone u pogledu.

Kreiranje složenog pogleda

- Ukoliko se u upitu pomoću kog se kreira pogled nalaze skupovne funkcije (min, max, avg, sum, count) ili izrazi, u pogledu se moraju definisati alternativna imena za te kolone.

Brisanje pogleda

```
DROP VIEW pogled;
```

Pogledi – Zadatak za vežbu

- Napraviti pogled koji će za sve radnike prikazati samo njihova imena, prezimena i platu.

```
CREATE OR REPLACE VIEW plate_radnika (Ime, Prezime, Plata) AS
    SELECT Ime, Prz, Plt
    FROM radnik;
```

Pogledi – Zadatak za vežbu

- Napraviti pogled koji će za sve radnike prikazati Mbr i ukupan broj sati angažovanja radnika na projektima na kojima radi.

```
CREATE OR REPLACE VIEW angaz_po_radnicima (Mbr, SBrc) AS
     SELECT r.Mbr, NVL(SUM(rp.Brc), 0)
     FROM radnik r, radproj rp
     WHERE r.Mbr = rp.Mbr (+)
     GROUP BY r.Mbr;
```

Pogledi – Zadatak za vežbu

- Napraviti pogled koji će za svakog šefa (rukovodioca radnika) prikazati njegov matični broj, prezime, ime, ukupan broj radnika kojima šefuje i njegovo ukupno angažovanje na svim projektima, na kojima radi. Koristiti prethodno definisani pogled.

```
CREATE VIEW angaz_sefova (Mbr, Prz, Ime, BrRad, BrSat) AS
    SELECT r.Sef, r1.Prz, r1.Ime, COUNT(*), a.SBrc
    FROM radnik r, radnik r1, angaz_po_radnicima a
    WHERE r.Sef = r1.Mbr AND r.Sef = a.Mbr
    GROUP BY r.Sef, r1.Prz, r1.Ime, a.SBrc;
```

Pogledi – Zadatak za vežbu

- Koliko je ukupno angažovanje svih šefova na projektima?

```
SELECT SUM(BrSat) AS UkAngSef
FROM angaz_sefova;
```

Sekvencer

- Generator sekvence vrednosti
- Automatski generiše jedinstvene brojeve
- Najčešće se koristi za kreiranje primarnih ključeva (surogat ključevi)
- Sekvenca se generiše i čuva nezavisno od tabele, tako da se jedna sekvenca može koristiti za više tabela

```
CREATE SEQUENCE sequence
[INCREMENT BY n]
[START WITH n]
[{MAXVALUE n | NOMAXVALUE}]
[{MINVALUE n | NOMINVALUE}]
[{CYCLE | NOCYCLE}]
[{CACHE n | NOCACHE}]

ALTER SEQUENCE sequence ...

DROP SEQUENCE sequence ...
```

Sekvencer – Primer

```
CREATE SEQUENCE SEQ_Mbr
INCREMENT BY 10
START WITH 240
NOCYCLE
CACHE 10;
```

```
INSERT INTO radnik (Mbr, Prz, Ime, God)
VALUES (SEQ_Mbr.NEXTVAL, 'Misic', 'Petar', SYSDATE);

SELECT SEQ_Mbr.CURRVAL
FROM SYS.DUAL;
```

Analitičke funkcije

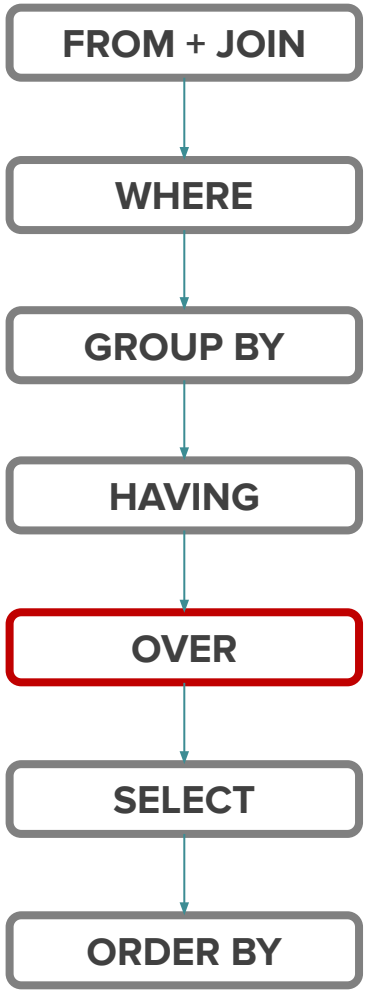
- Kod agregacionih funkcija i GROUP BY klauzule, na izlazu operacije grupisanja se za svaku grupu (podskup redova) dobija po jedan red.

```
SELECT rp.spr, AVG(rp.brc)
FROM radproj rp
GROUP BY rp.spr;
```

- Kod analitičkih funkcija, izračunavanja se takođe vrše na nivou podskupa redova, ali se njihovom primenom ne smanjuje broj redova na izlazu.

```
SELECT r.mbr, r.ime, r.prz, rp.spr, rp.brc, AVG(rp.brc) OVER() AS prosek_brc_ukupni
FROM radnik r INNER JOIN radproj rp ON r.mbr=rp.mbr;
```

Redosled izvršavanja klauzula



Analitičke funkcije

```
analitička_funkcija([arg]) OVER (analitički_uslovi)
```

- analitička_funkcija – AVG, MAX, MIN, COUNT, ROW_NUMBER, ...
- analitički_uslovi
  - sastoje se od sledećih delova:
    - [uslov_particionisanja] [uslov_uređivanja] [uslov_odabira_torki]
      - Uslov particionisanja definiše uslov za podelu ulaznog skupa torki u particije
      - Uslov uređivanja koristi se za uređivanje redova unutar definisanih particija
        - Bitno ako je zadata analitička funkcija osetljiva na redosled redova
      - Ukoliko se nijedan deo ne zada (prazan OVER), svi ulazni redovi biće deo iste particije

Analitičke funkcije – Primer

- Prikazati mbr, ime, prz radnika angažovanih na projektima. Pored toga, prikazati spr i brc projekata na kojima su angažovani, kao i prosečno angažovanje na tom projektu.

```
SELECT r.mbr, r.ime, r.prz, rp.spr, rp.brc,
    AVG(rp.brc) OVER (PARTITION BY rp.spr) AS prosek_brc_za_projekat
FROM radnik r INNER JOIN radproj rp ON r.mbr=rp.mbr;
```

Kumulativni zbir – Primer

- Prikazati mbr, datum isplate, razlog isplate, isplaćeni iznos, kao i kumulativnu sumu isplaćenog iznosa od početka 2023. godine za radnika sa matičnim brojem 70.

```
SELECT mbr, datum_isplate, razlog_isplate, iznos,
       SUM(iznos) OVER (ORDER BY datum_isplate) AS kumulativni_zbir
FROM isplate_radnicima
WHERE mbr = 70 AND godina = 2023
ORDER BY datum_isplate;
```

Analitičke funkcije – Kumulativni zbir

- Ukoliko se u analitičkim uslovima zada uslov uređivanja, zadata funkcija se računa kumulativno, saglasno uređenju redova.
- Može se eksplicitno definisati koji bi redovi trebalo da se uzmu u obzir prilikom ovakvog računanja
  - **uslov_odabira_torki** (engl. *windowing clause*)
  - RANGE BETWEEN početna_tačka AND krajnja_tačka
  - ROWS BETWEEN početna_tačka AND krajnja_tačka
  - Moguće vrednosti za početnu i krajnju tačku:
    - UNBOUNDED PRECEDING (samo početna)
    - UNBOUNDED FOLLOWING (samo krajnja)
    - CURRENT ROW

Kumulativni zbir – Primer

- Prikazati mbr, ime, prz radnika angažovanih na projektima. Pored toga, prikazati spr i brc projekata na kojima su angažovani, kao i kumulativnu sumu broja sati rada za radnike uređene od najmlađeg do najstarijeg.

```
SELECT r.mbr, r.ime, r.prz, rp.spr, rp.brc,
     SUM(rp.brc) OVER(partition by rp.spr ORDER BY god DESC
          ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS kumulativni_zbir
FROM radnik r INNER JOIN radproj rp ON r.mbr=rp.mbr;
```

Top n na nivou svake particije – Primer

- Za svaki projekat prikazati podatke o top 2 radnika sa najviše sati angažovanja.

```
WITH uredjeni_radnici AS (
     SELECT mbr, spr, brc, row_number() OVER(PARTITION BY spr ORDER BY brc DESC) AS rbr
     FROM radproj
     ORDER BY spr
)
SELECT r.mbr, r.ime, r.prz, ur.spr, ur.brc, ur.rbr
FROM uredjeni_radnici ur inner join radnik r ON r.mbr=ur.mbr
WHERE rbr <= 2
ORDER BY ur.spr, ur.rbr;
```

Funkcija row_number

- Ponašanje nalik onom viđenom kod ROWNUM pseudokolone
- Dodeljuje redne brojeve redovima unutar svake particije
- Funkcija je osetljiva na redosled redova
  - Izbacuje grešku ako se ne iskoristi uslov uređivanja
- Ako se ne navede uslov particionisanja u OVER, svi redovi će dobiti jedinstveni redni broj

Funkcije lead i lag

- Omogućavaju računanje vrednosti zadatog izraza za prethodni (lag) ili naredni (lead) red
- Funkcije su osetljive na redosled redova - mora se odrediti ko je prethodni, odnosno naredni
    - Izbacuje grešku ako se ne iskoristi uslov uređivanja

Računanje u odnosu na prethodnog/narednog – Primer

- Za svakog radnika prikazati razliku između njegove plate i plate radnika koji mu prethodi po visini plate, uređeno uzlazno po plati.

```
WITH susedne_plate AS (
SELECT ime, prz, plt, lag(plt) OVER (ORDER BY plt) plata_prethodnog
FROM radnik
)
SELECT ime, prz, plt, plata_prethodnog, plt-COALESCE(plata_prethodnog, 0) razlika_u_plati
FROM susedne_plate
ORDER BY plt;
```

Zadatak za vežbu

- Za svakog radnika prikazati koliko je dana prošlo između njegovog rođenja i rođenja njemu narednog radnika po datumu rođenja.

```
WITH susedni_po_god AS (
SELECT ime, prz, god, LEAD(god) OVER (ORDER BY god) god_narednog
FROM radnik
)
SELECT ime, prz, god, god_narednog, ROUND(COALESCE(god_narednog, SYSDATE)-god) razlika_god_u_danima
FROM susedni_po_god
ORDER BY god desc;
```

Zadatak za vežbu

- Za radnike koji imaju šefa ispisati mbr, ime, prz i rang po opadajućem iznosu plate u okviru istog nadređenog (šefa). Koristiti analitičku funkciju rank().

```
SELECT mbr, ime, prz, sef, plt,
     rank() OVER (PARTITION BY sef ORDER BY plt DESC) "Rang"
FROM radnik
WHERE sef IS NOT NULL;
```

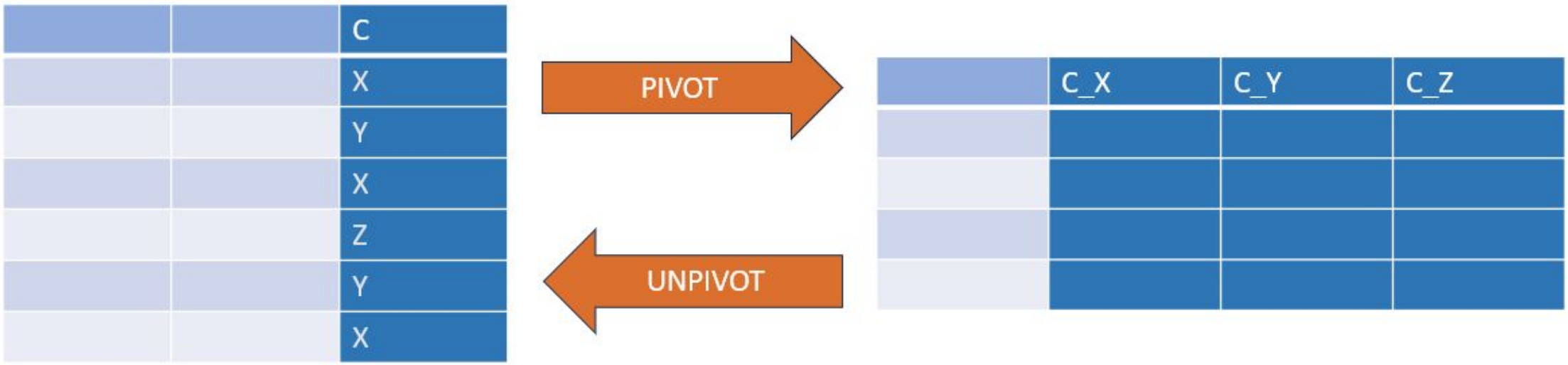
Zadatak za vežbu

- Za radnike angažovane na projektima ispisati mbr, ime, prz, spr i udeo broja časova njihovog angažmana u ukupnom broju časova na projektu.

```
SELECT r.mbr, ime, prz, spr,
     ROUND(brc/(SUM(brc) OVER (PARTITION BY spr)), 2) "Udeo"
FROM radnik r JOIN radproj rp ON r.mbr = rp.mbr;
```

Pivot i Unpivot

- Klauzule koje se mogu koristiti za “transponovanje” sadržaja tabela
  - Pivot omogućava transponovanje redova u kolone
    - “Uređeniji” prikaz agregiranih vrednosti u odnosu na GROUP BY
  - Unpivot omogućava transponovanje kolona u redove



Pivot

```
SELECT lista_izraza
FROM lista_tabela
PIVOT (
        pozivi_agregacionih_funkcija_nad_kolonama
        FOR obeležje_pivotiranja
        IN (lista_vrednosti_obeležja_pivotiranja)
    )
```

- `pozivi_agregacionih_funkcija_nad_kolonama` - prilikom transponovanja, vrši se agregiranje podataka
  - Implicitni GROUP BY **nad svim obeležjima koja nisu korišćena u klauzuli**
- `obeležje_pivotiranja` - definiše obeležje na osnovu čijih vrednosti se vrši pivotiranje
- `lista_vrednosti_obeležja_pivotiranja` - definiše listu vrednosti obeležja pivotiranja za koje bi trebalo da se izvrši transponovanje
  - Svaka vrednost u listi rezultovaće kreiranjem jedne ili više novih kolona
  - Kreira se nova kolona za svaku kombinaciju vrednosti obeležja pivotiranja i pozvanih agregacionih funkcija
    - Imena kolona se automatski određuju na osnovu kombinacije alijasa

Pivot - Primer

- Za svakog radnika prikazati ukupan iznos isplaćen za svaki pojedinačni razlog isplate, kao i koliko je puta vršena isplata za svaki razlog isplate. Ukupan iznos isplate i broj isplata po razlozima prikazati u vidu kolona.

```
SELECT *
FROM
     (SELECT mbr, razlog_isplate, iznos FROM isplate_radnicima)
PIVOT (
     SUM(iznos) ukupno,
     COUNT(razlog_isplate) broj_isplata
     FOR razlog_isplate IN ('PLATA_DEO1' AS PLATA_DEO1, 'PLATA_DEO2' AS PLATA_DEO2,
          'PUTNI_TROSKOVI' AS PUTNI_TROSKOVI, 'BONUS' AS BONUS, 'DNEVNICA' AS DNEVNICA)
)
ORDER BY mbr;
```

Pivot - Zadatak za vežbu

- Za svakog radnika prikazati iznos isplaćen za svaki mesec u prvoj polovini 2023. godine. Isplate po mesecima prikazati u vidu kolona.

```
SELECT *
FROM (
     SELECT mbr, mesec, godina, iznos
     FROM isplate_radnicima
     WHERE godina = 2023
     )
PIVOT (
     SUM(iznos) ukupno
     FOR mesec
     IN ('January', 'February', 'March', 'April', 'May', 'June')
)
ORDER BY mbr;
```

Unpivot

```
SELECT lista_izraza
FROM lista_tabela
UNPIVOT [INCLUDE | EXCLUDE NULLS](
        lista_ciljnih_obeležja_za_vrednosti
        FOR kategorizaciono_obeležje
        IN (lista_naziva_obeležja_unpivotiranja)
    )
```

- `lista_ciljnih_obeležja_za_vrednosti -` obeležja koja će biti kreirana kako bi se čuvale vrednosti za unpivotirane kolone
- `kategorizaciono_obeležje` - obeležje koje će biti kreirano kako bi se čuvale vrednosti kategorija
- `lista_naziva_obeležja_unpivotiranja-` definiše listu obeležja za koje bi trebalo da se izvrši transponovanje
  - Za svaku vrednost u listi kategorizaciono obeležje će dobiti novu vrednost
  - Poželjno dodeliti alijase kako bi vrednosti kategorizacionog obeležja bile smislene

Unpivot - Primer

- Kreirati tabelu isplate_pivotirane

```
CREATE TABLE isplate_pivotirane AS
SELECT *
FROM
     (SELECT mbr, razlog_isplate, iznos FROM isplate_radnicima)
PIVOT (
     SUM(iznos) ukupno,
     COUNT(razlog_isplate) broj_isplata
     FOR razlog_isplate IN ('PLATA_DEO1' AS PLATA_DEO1, 'PLATA_DEO2' AS PLATA_DEO2,
          'PUTNI_TROSKOVI' AS PUTNI_TROSKOVI, 'BONUS' AS BONUS, 'DNEVNICA' AS DNEVNICA)
)
ORDER BY mbr;
```

Unpivot - Primer

- Na osnovu sadržaja tabele ISPLATE_PIVOTIRANE, ispisati koliko je za svaku kategoriju razloga_isplate radnicima isplaćeno sredstava. Svaka kategorija trebalo bi da bude ispisana kao poseban red u izveštaju.

```
SELECT *
FROM (SELECT MBR, PLATA_DEO1_UKUPNO, PLATA_DEO2_UKUPNO, PUTNI_TROSKOVI_UKUPNO ,
BONUS_UKUPNO, DNEVNICA_UKUPNO FROM ISPLATE_PIVOTIRANE)
UNPIVOT EXCLUDE NULLS (
    IZNOS
    FOR RAZLOG
    IN (
        PLATA_DEO1_UKUPNO AS 'PLATA_DEO1', PLATA_DEO2_UKUPNO AS 'PLATA_DEO2',
        PUTNI_TROSKOVI_UKUPNO AS 'PUTNI_TROSKOVI', BONUS_UKUPNO AS 'BONUS',
        DNEVNICA_UKUPNO AS 'DNEVNICA'
    )
);
```

Kraj!

Hvala na pažnji!