Kreiranje tabele

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

- 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

Naziv tabele i kolone

- mora početi slovom,
- mora biti između 1 i 30 znakova dužine,
- mora sadržati samo velika i mala slova, cifre, _, $ i #,
- ne sme se poklapati sa nazivom nekog drugog objekta koji je kreirao isti korisnik,
- ne sme biti rezervisana reč Oracle servera.
- Nazivi nisu case sensitive.

SQL tipovi podataka

Tip podatka	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 2 GB
– **BLOB**	binarni podaci do 4 GB
– **BFILE**	binarni podaci smešteni u eksternom fajlu do 4 GB
– **ROWID**	jedinstvena adresa vrste u tabeli

Tabela radnik

```
CREATE TABLE radnik
    (
     Mbr integer NOT NULL,
     Ime varchar2(20) NOT NULL,
     Prz varchar2(25) NOT NULL,
     Sef integer,
     Plt decimal(10, 2),
     Pre decimal(6, 2),
     God date NOT NULL,
     CONSTRAINT radnik_PK PRIMARY KEY (Mbr),
     CONSTRAINT radnik_FK FOREIGN KEY (Sef) REFERENCES
    Radnik (Mbr),
    CONSTRAINT radnik_CH CHECK (Plt>500)
    );
```

Tabela projekat

```
CREATE TABLE projekat
   (
    Spr integer not null,
    Ruk integer not null,
    Nap varchar2(30),
    Nar varchar2(30),
    CONSTRAINT projekat_PK PRIMARY KEY (Spr),
    CONSTRAINT projekat_FK FOREIGN KEY (Ruk)
    REFERENCES Radnik (Mbr),
    CONSTRAINT projekat_UK UNIQUE (Nap)
   );
```

Tabela radproj

```
CREATE TABLE radproj
   (
    Spr integer NOT NULL,
    Mbr integer NOT NULL,
    Brc integer NOT NULL,
    CONSTRAINT radproj_PK PRIMARY KEY (Spr, Mbr),
    CONSTRAINT radproj_rad_FK FOREIGN KEY (Mbr)
   REFERENCES radnik(Mbr),
    CONSTRAINT radproj_prj_FK FOREIGN KEY (Spr)
   REFERENCES projekat(Spr)
   );
```

Tabela faze_projekta

- Kreirati tabelu faze_projekta

faze_projekta({Spr , Sfp, Rukfp, Nafp, Datp}, {Spr+ Sfp})

faze_projekta[Spr] $\subseteq$ projekat[Spr],

faze_projekta[Rukfp] $\subseteq$ radnik[Mbr]

Tabela faze_projekta

- **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.

Izmena definicije tabele

- **ALTER TABLE**

  Alter table iskaz služi za:
  - dodavanje nove kolone,
  - modifikaciju postojeće kolone,
  - definisanje podrazumevane vrednosti za novu kolonu,
  - brisanje kolone
  - 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

- U tabelu faze_projekta dodati atribut:

  Datz - datum završetka faze projekta.

- **Datz** ne sme biti manji od **Datp**

Podaci za faze_projekta

- U tabelu faze_projekta dodati bar dve faze za jedan projekat i jednu za drugi projekat

Zadatak za vežbu

- Za svaki projekat prikazati sifru projekta, naziv projekta, ime i prezime rukovodioca projekta, prezime njegovog šefa, nazive faza projekta, imena i prezimena rukovodioca faza projekta. Ako projekat nije podeljen u faze napisati: nema faze.

Brisanje definicije tabele

DROP TABLE <naziv_tabele>;

Brisanje definicije tabele

- Izbrisati tabelu faze_projekta.

Create table as select

- Kreirati tabelu radnik2 koja ima iste kolone kao tabela radnik, pri čemu radnik2 sadrži samo podatke o radnicima koji imaju platu manju od 10000

CREATE TABLE radnik2 AS (SELECT * FROM radnik WHERE plt < 10000);

Tabela radnik2 neće imati indekse i sva ograničenja koja ima tabela radnik

Insert into select

- Dopuniti tabelu radnik2 podacima koji joj nedostaju iz tabele radnik,

INSERT INTO radnik2 (SELECT * FROM radnik WHERE plt>=10000);

- Proveriti da li je broj torki u tabeli radnik jednak broju torki u tabeli radnik2

- Izbrisati sadržaj i definiciju tabele radnik2

Klauzula WITH

- Eng. *Common Table Expressions - CTE*
- 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

- 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

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

```
with projinfo as (
  select spr, avg(brc) prosek
  from radproj group by spr)
select distinct r.mbr, r.ime, r.prz, r.plt
from radnik r, radproj rp, projinfo pi
where r.mbr=rp.mbr and rp.spr=pi.spr
group by r.mbr, r.ime, r.prz, r.plt, pi.spr
having avg(rp.brc)>(select prosek from projinfo pi2
where pi2.spr=pi.spr);
```

WITH – Zadatak

- Prikazati mbr, ime, prz, plt radnika čiji je broj sati angažovanja na nekom projektu veći od prosečnog angažovanja na svim projektima

```
with projinfo as (
  select spr, avg(brc) pros
  from radproj group by spr)
select distinct r.mbr, r.ime, r.prz, r.plt
from radnik r, radproj rp, projinfo pi
where r.mbr=rp.mbr and rp.spr=pi.spr
group by r.mbr, r.ime, r.prz, r.plt, pi.spr
having avg(rp.brc)>(select avg(pros) from projinfo);
```

WITH – Primer

- 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 – Zadatak

- Koliko je ukupno angažovanje svih šefova na projektima?

```
with 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),
angaz_sefova (mbr, prz, ime, brrad, brsat) as (
  select distinct 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)
select sum(brsat) as ukangsef
from angaz_sefova;
```

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 zameljuje 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.

DML operacije sa pogledima

- DML (Data Manipulation Language) operacije se mogu primenjivati na jednostavnim pogledima.
- Ako pogled sadrži **skupovne funkcije**, **group by** kaluzulu, **distinct** rezervisanu reč ili **rownum** rezervisanu reč, vrsta iz pogleda se ne može izbrisati.
- Isto važi i za modifikaciju podataka, s tim što dodatno važi i da se **kolone definisane izrazima** ne mogu modifikovati (npr, salary*12).
- U pogled se ne mogu dodavati podaci ako pogled sadrži **skupovne funkcije**, **group by** kaluzulu, **distinct** rezervisanu reč, **rownum** rezervisanu reč, **kolonu koja je definisana izrazom**, ili **not null** kolonu u baznoj tabeli koja nije selektovana u pogledu.
- Dodavanjem vrednosti u pogled, one se dodaju direktno u baznu tabelu.

Brisanje pogleda

DROP VIEW pogled;

Pogled

- 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;
```

Pogled

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

Pogled

```
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;
```

Pogled

- 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.

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;
```

Pogled

- Koliko je ukupno angažovanje svih šefova na projektima?

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

Sekvenca

- automatski generiše jedinstvene brojeve
- najčešće se koristi za kreiranje primarnih ključeva
- sekvenca se generiše i čuva nezavisno od tabele, tako da se jedna sekvenca može koristiti za više tabela

Sekvencer (Generator sekvence vrednosti)

```
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
```

Primer upotrebe sekvencera

```
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);
```

Primer upotrebe sekvencera

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

Tabele u Oracle bazi podataka

- korisničke tabele
  - kolekcije tabela koje kreira i održava korisnik
  - sadrže korisničke informacije
- **Data Dictionary** (rečnik podataka)
  - kolekcija tabela koje kreira i održava Oracle server
  - sadrže informacije baze podataka
  - vlasnik svih tabela u rečniku je SYS korisnik
  - informacije smeštene u rečniku podataka obuhvataju imena korisnika Oracle servera, privilegije dodeljene korisnicima, nazive objekata baze podataka, ograničenja.
  - postoji nekoliko kategorija pogleda rečnika podataka; svaka od njih ima odgovarajući prefiks:
    - USER_ - ovi pogledi sadrže informacije o objektima čiji je vlasnik korisnik
    - ALL_ - ovi pogledi sadrže informacije o svim tabelama (objektnim i relacionim) koje su dostupne korisniku
    - DBA_ - ovi pogledi su zabranjeni, tj. dostupni su samo korisnicima koji imaju DBA ulogu

Tabele u Oracle bazi podataka

- Upiti u rečniku podataka se postavljaju kao i svi ostali upiti.
- Prikazati nazive tabela čiji je vlasnik korisnik.

```
SELECT table_name
FROM user_tables;
```

- Prikazati različite tipove objekata čiji je vlasnik korisnik.

```
SELECT DISTINCT object_type
FROM user_objects;
```

- Prikazati tabele, poglede, sinonime i sekvence čiji je vlasnik korisnik.

```
SELECT *
FROM user_catalog;
```

Neke karakter funkcije

- LOWER(char) – za konvertovanje svih znakova u mala slova
- UPPER(char) – za konvertovanje svih znakova u velika slova
- INITCAP(char) – prvo slovo svake reči u nizu znakova pretvara u veliko slovo, a ostatak reči u mala slova
- SUBSTR(char, m [,n]) – koristi se za izdvajanje dela niza znakova
- TRIM(LEADING | TRAILING | BOTH trim_character FROM trim_source) – uklanja početne ili prateće znakove sa početka ili kraja niza znakova
- LENGTH(char) – vraća broj znakova u nizu

Neke karakter funkcije - primeri

LOWER (‘Sva mala slova')→ 'sva mala slova‘

UPPER ('Sva velika slova')→ ‘SVA VELIKA SLOVA‘

INITCAP(‘Velika početna slova')→ ' Velika Početna Slova‘

SUBSTR('DobroJutro',1,5) →'Dobro‘

TRIM('D' FROM 'DobroJutro') → 'obroJutro'

LENGTH('DobroJutro') → 10

Neke karakter funkcije - primer

```
SELECT Mbr, Prz, Ime
FROM Radnik
WHERE UPPER(Prz) = 'PETRIC';
```

Neke karakter funkcije – primer 2

- Prikazati radnike čije prezime na početku sadrži prva 3 slova imena, na primer:
  Petar Petric

```
SELECT * from radnik
WHERE prz LIKE
SUBSTR(IME,0,3) || '%';
```

Zadatak za vežbu

- Prikazati imena i prezimena radnika tako da se sva imena koja imaju poslednje slovo 'a', prikazuju bez poslednjeg slova.

SELECT TRIM(TRAILING 'a' FROM ime) FROM radnik;

Zadatak za vežbu

- Svim radnicima promeniti ime tako da poslednje slovo bude uvećano.
- Primer: AnA -> AnA, Marko -> MarkO

UPDATE radnik SET ime=
SUBSTR(ime,1,LENGTH(ime)-1) ||
UPPER(SUBSTR(ime,LENGTH(ime),1));

Neke funkcije za konverziju podataka

- TO_CHAR(d [, fmt]) – transformiše vrednosti tipa DATE u VARCHAR2, po izboru uz navedeni format datuma
- TO_CHAR(n [, fmt]) – transformiše vrednost brojčanog tipa u VARCHAR2, po izboru uz navedeni format broja
- TO_DATE(char [, fmt]) – za konvertovanje niza znakova u ekvivalentni datum
- TO_NUMBER(char [,fmt]) – za konvertovanje znakovnih vrednosti u numeričke

Zadatak za vežbu

- Za svakog radnika prikazati ime, prz, i projekte na kojima radi. Ako ne radi ni na jednom projektu, napisati ‘Ne radi na projektu’. Imena radnika prikazati velikim slovima, a prezimena malim.

```
SELECT UPPER(ime), LOWER(prz),
NVL(TO_CHAR(spr), 'Ne radi na projektu') broj_proj
FROM radnik LEFT OUTER JOIN radproj
on radnik.mbr = radproj.mbr;
```

Zadatak za vežbu

- Za svakog radnika prikazati datum rođenja u formatu yyyy/mm/dd

SELECT TO_CHAR(god,'yyyy/mm/dd') FROM radnik;

Analitičke funkcije

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

Primer:

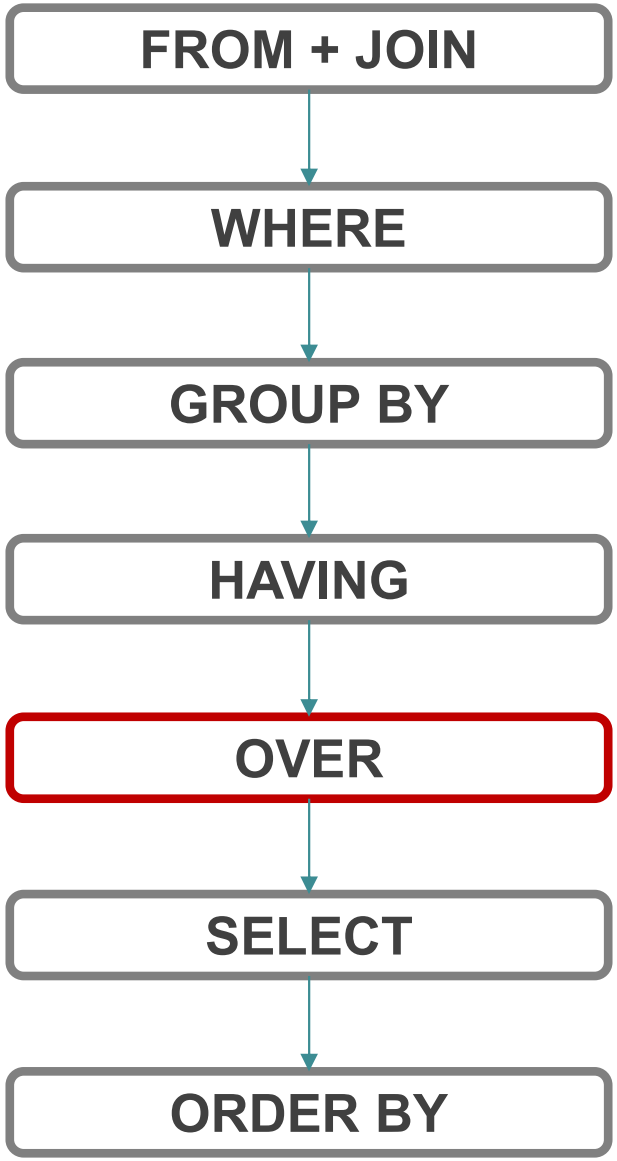
```
select rp.spr, avg(rp.brc)
from radproj rp
group by rp.spr;
```

Analitičke funkcije

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

```
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

- Opšti oblik sintakse:

analitička_funkcija([arg]) OVER (analitički_uslovi)

  - analitička_funkcija – AVG, MAX, MIN, COUNT, ROW_NUMBER,...
  - analitički_uslovi
    - sastoje se iz 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

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;
```

Primer – kumulativni zbir

- 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) as kumulativni_zbir
from radnik r inner join radproj rp on
    r.mbr=rp.mbr;
```

Primer – 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 (eng. *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

Primer – kumulativni zbir

- 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;
```

Primer - top n na nivou svake particije

Za svaki projekat prikazati podatke o top 2 radnika sa najvećim angažmanom

```
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
  - Baca grešku ako se ne iskoristi uslov uređivanja
- Ako se ne navede uslov particionisanja u OVER, svi redovi će dobiti jedinstveni redni broj

Zadatak

- 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 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;
```

WITH - Rekurzija

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

- 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 – Zadatak

- Prikazati za svakog radnika sve direktno i indirektno podređene radnike, ako nema podređenih prikazati null umesto oznake podređenog

```
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 r.mbr, h.pod from hijerarhija h, radnik r
where r.mbr = h.mbr(+) order by mbr, pod;
```

WITH – Rekurzija – Zadatak

- Prikazati za svakog radnika oznaku šefa

```
with hijerarhija(mbr,imeprz,sef) as
( select mbr, ime||' '||prz, sef
  from radnik
  where sef is null
  union all
  select r.mbr, r.ime||' '||r.prz, h.mbr
  from radnik r, hijerarhija h
  where r.sef = h.mbr )
select * from hijerarhija;
```

WITH – Rekurzija – SEARCH

- klauzula SEARCH
- definiše poredak redova
  - BREADTH FIRST, DEPTH FIRST
- BY – poredak redova na istom nivou
- SET – vrednost pseudo-obeležja po redosledu redova
  - pseudo-obeležje automatski postaje deo rezultata

SEARCH BREADTH | DEPTH FIRST
BY *lista_obeležja*
SET *pseudo-obeležje*

WITH – Rekurzija – Primer

- Prikazati hijerarhiju rukovođenja po nivoima

```
with hijerarhija(mbr,imeprz,sef,nivo) as
( select mbr, ime||' '||prz, sef, 1 as nivo
  from radnik
  where sef is null
  union all
  select r.mbr, r.ime||' '||r.prz, h.mbr, nivo+1
  from radnik r, hijerarhija h
  where r.sef = h.mbr)
search depth first by imeprz set poredak
select nivo, rpad(' ',3*nivo)||imeprz as imeprz,
mbr, sef
from hijerarhija
order by poredak;
```

WITH – Rekurzija – Primer

- Prikazati hijerarhiju rukovođenja po nivoima

```
with hijerarhija(mbr,imeprz,sef,nivo) as
( select mbr, ime||' '||prz, sef, 1 as nivo
  from radnik
  where sef is null
  union all
  select r.mbr, r.ime||' '||r.prz, h.mbr, nivo+1
  from radnik r, hijerarhija h
  where r.sef = h.mbr)
search breadth first by imeprz set poredak
select nivo, rpad(' ',3*nivo)||imeprz as imeprz,
mbr, sef
from hijerarhija
order by poredak;
```

WITH – Rekurzija – Primer

- Prikazati za svakog radnika lanac rukovođenja

```
with hijerarhija(imeprz,mbr,sef,lanac,glavni) as
( select ime||' '||prz, mbr, sef,
   '/'||ime||' '||prz as lanac,
    ime||' '||prz as glavni
    from radnik
    where sef is null
    union all
    select r.ime||' '||r.prz, r.mbr, r.sef,
    h.lanac||'/'||r.ime||' '||r.prz as lanac, h.glavni
    from radnik r, hijerarhija h
    where r.sef = h.mbr )
select imeprz,mbr,sef,lanac,glavni from hijerarhija;
```

WITH – Rekurzija – CYCLE

- klauzula CYCLE
- označava cikluse u rekurziji
  - prema proveri zadate liste obeležja
- oznaka prisustva ili odsustva ciklusa
  - pseudo-obeležje automatski postaje deo rezultata
  - jedan karakter

CYCLE *lista_obeležja*
SET *pseudo_obeležje*
TO *oznaka_ciklusa*
DEFAULT *oznaka_odsustva_ciklusa*

WITH – Rekurzija – Primer

- Prikazati za svakog radnika lanac rukovođenja, uz proveru postojanja ciklusa

```
with hijerarhija(imeprz,mbr,sef,lanac,glavni) as
( select ime||' '||prz, mbr, sef,
   '/'||ime||' '||prz as lanac,  ime||' '||prz as glavni
   from radnik where sef is null
   union all
   select r.ime||' '||r.prz, r.mbr, r.sef,
   h.lanac||'/'||r.ime||' '||r.prz as lanac, h.glavni
   from radnik r, hijerarhija h where r.sef = h.mbr )
search breadth first by imeprz set poredak
cycle mbr set ciklus to 'x' default 'o'
select imeprz,mbr,sef,lanac,glavni, ciklus from
hijerarhija;
```

WITH – Rekurzija – Zadatak

- Prikazati sve podređene za radnika 70

```
with hijerarhija(imeprz,mbr,sef,lanac,glavni) as
( select ime||' '||prz, mbr, sef,
   '/'||ime||' '||prz as lanac,
    ime||' '||prz as glavni
    from radnik where mbr = 70
    union all
    select r.ime||' '||r.prz, r.mbr, r.sef,
    h.lanac||'/'||r.ime||' '||r.prz as lanac, h.glavni
    from radnik r, hijerarhija h where r.sef = h.mbr )
search breadth first by imeprz set poredak
cycle mbr set ciklus to 'x' default 'o'
select imeprz,mbr,sef,lanac,glavni, ciklus from
hijerarhija;
```

WITH – Rekurzija – Zadatak

- Promeniti šefa radnika 70 da bude radnik 140

```
update radnik
set sef = 140
where mbr = 70;
```

WITH – Rekurzija – Zadatak

- Ponovo prikazati sve podređene za radnika 70

```
with hijerarhija(imeprz,mbr,sef,lanac,glavni) as
( select ime||' '||prz, mbr, sef,
  '/'||ime||' '||prz as lanac,  ime||' '||prz as glavni
  from radnik  where mbr = 70
  union all
  select r.ime||' '||r.prz, r.mbr, r.sef,
  h.lanac||'/'||r.ime||' '||r.prz as lanac, h.glavni
  from radnik r, hijerarhija h
  where r.sef = h.mbr )
search breadth first by imeprz set poredak
cycle mbr set ciklus to 'x' default 'o'
select imeprz, mbr, sef, lanac, glavni, ciklus
from hijerarhija;
```

WITH – Rekurzija – Zadatak

- Poništiti promenu šefa radnika 70

  rollback;

SPARSE MATRICE

Sparse matrice

- Predstavljaju matrice gde većina elemenata sadrži vrednost 0.
- Velike sparse matrice se pojavljuju u naučnim proračunima prilikom rešavanja parcijalnih diferencijalnih jadnačina

Sparse matrice

- Primer množenja dve sparse matrice

$$\begin{bmatrix} 0&1&0&0&9 \\ 0&0&3&0&0 \\ 0&0&0&2&0 \\ 0&0&0&0&0 \\ 0&0&0&0&0 \end{bmatrix} \times \begin{bmatrix} 1&1&0&0&0 \\ 0&0&0&0&0 \\ 0&7&0&0&0 \\ 0&0&2&0&0 \\ 0&0&0&0&0 \end{bmatrix} = \begin{bmatrix} 0&0&0&0&0 \\ 0&2&1&0&0&0 \\ 0&0&4&0&0 \\ 0&0&0&0&0 \\ 0&0&0&0&0 \end{bmatrix}$$

Množenje sparse matrica

```
SELECT a.row_num, b.col_num,
SUM(a.value*b.value)
FROM a, b
WHERE a.col_num = b.row_num
GROUP BY a.row_num, b.col_num;
```

Sabiranje sparse matrica

```
SELECT a.row_num, a.col_num, a.value
FROM a
WHERE NOT EXISTS (SELECT 0 FROM b
WHERE a.col_num = b.col_num and
a.row_num = b.row_num);
```

Sabiranje sparse matrica

```
UNION
SELECT b.row_num, b.col_num, b.value
FROM b
WHERE NOT EXISTS (SELECT 0 FROM a
WHERE a.col_num = b.col_num and
a.row_num = b.row_num);
```

Sabiranje sparse matrica

```
UNION
SELECT a.row_num, a.col_num, a.value +
b.value
FROM a,b
WHERE a.col_num = b.col_num and
a.row_num = b.row_num;
```

KRAJ PREZENTACIJE O SQL-U