Database Programming with SQL
kurs 2017 – database design and programming with sql
students slajdovi

15-1 Creating Views

- Da li svi zaposleni treba da imaju podjednaka prava za korišćenje podataka iz db ? Slična pitanja su deo posla DBA
- Kako se prave views – virtualne reprezentacije tabela nameštenih da ispunjavaju zahteve određenih korisnika

View

- Pogled, kao i tabela je db objekat; ali pogledi nisu realne tabele
- Pogledi su logičke reprezentacije postojećih tabela ili drugih pogleda
- Pogledi nemaju sopstvene podatke; oni funkcionišu kao prozor kroz koji se može videti ili izmeniti podatak u tabeli
- Tabele na osnovu kojih su pogledi bazirani se nazivaju “base“ (osnovne) tabele
- Pogled je upit smešten kao SELECT iskaz u DD

```
CREATE VIEW view_employees
AS SELECT employee_id,first_name, last_name, email
   FROM employees
   WHERE employee_id BETWEEN 100 and 124;
```

```
SELECT *
FROM view_employees;
```

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL
100	Steven	King	SKING
101	Neena	Kochhar	NKOCHHAR
102	Lex	De Haan	LDEHAAN
124	Kevin	Mourgos	KMOURGOS
103	Alexander	Hunold	AHUNOLD
104	Bruce	Ernst	BERNST
107	Diana	Lorentz	DLORENTZ

Why use Views ?

- Pogledi ograničavaju pristup podacima osnovne tabele pošto pogled može prikazati odabrane kolone iz tabele
- Pogled može da se koristi za smanjenje kompleksnosti izvršnih upita bazirano na komplikovanijim SELECT iskazima
- Npr, kreator pogleda može napraviti join iskaz koji vraća podatke iz više tabela
- Korisnik pogleda ne vidi kod ispod niti kako da ga napravi
- Korisnik, preko pogleda, interaguje sa db korišćenjem jednostavnih upita
- Pogledi se mogu koristiti za dobijanje podataka iz nekoliko tabela, čime se omogućava nezavisnost podataka za korisnike
- Korisnici mogu videti iste podatke na različite načine
- Pogledi daju grupama korisnika pristup podacima prema njihovim određenim dozvolama ili kriterijumima

Creating a View

- Za kreiranje pogleda, uneti podupit unutar CREATE VIEW iskaza
- Sintaksa za pogled iskaz:

```
CREATE [OR REPLACE] [FORCE| NOFORCE] VIEW view  [(alias [,
   alias]...)] AS subquery
[WITH CHECK OPTION [CONSTRAINT constraint]]
[WITH READ ONLY [CONSTRAINT constraint]];
```

OR REPLACE	Re-creates the view if it already exists.
FORCE	Creates the view whether or not the base tables exist.
NOFORCE	Creates the view only if the base table exists (default).
view_name	Specifies the name of the view.
alias	Specifies a name for each expression selected by the view's query.
subquery	Is a complete SELECT statement. You can use aliases for the columns in the SELECT list. The subquery can contain complex SELECT syntax.
WITH CHECK OPTION	Specifies that rows remain accessible to the view after insert or update operations.
CONSTRAINT	Is the name assigned to the CHECK OPTION constraint.
WITH READ ONLY	Ensures that no DML operations can be performed on this view.

- Primer:

```
CREATE OR REPLACE VIEW view_euro_countries
AS SELECT country_id, region_id, country_name, capitol
   FROM wf_countries
   WHERE location LIKE '%Europe';
```

```
SELECT * FROM view_euro_countries
ORDER BY country_name;
```

COUNTRY_ID	REGION_ID	COUNTRY_NAME	CAPITOL
22	155	Bailiwick of Guernsey	Saint Peter Port
203	155	Bailiwick of Jersey	Saint Helier
387	39	Bosnia and Herzegovina	Sarajevo
420	151	Czech Republic	Prague
298	154	Faroe Islands	Torshavn
49	155	Federal Republic of Germany	Berlin
33	155	French Republic	Paris
...	...	...	...

- Kada se jednom kreira pogled od strane DBA, korisnik može napraviti upit nad pogledom kao da je u pitanju tabela

Guidelines for Creating a View

- Podupit koji definiše pogled može sadržati kompleksnu SELECT sintaksu
- Podupit koji definiše pogled ne treba da sadrži ORDER BY izaraz. ORDER BY izraz je specificiran kada se vrati podatak iz pogleda
- Može se koristiti OR REPLACE opcija za promenu definicije pogleda bez moranja za odstranjivanjem ili ponovnim dozvoljavanjem objektu privilegija koje su njemu bile prethodno dopuštene
- Alijasi se mogu koristiti za imena kolona u podupitu

CREATE VIEW Features

- Koriste se dve kvalifikacije pogleda: jednostavan i kompleksan
- Tabela sumira osobine svakog od njih:

Feature	Simple Views	Complex Views
Number of tables used to derive data	One	One or more
Can contain functions	No	Yes
Can contain groups of data	No	Yes
Can perform DML operations (INSERT, UPDATE, DELETE) through a view	Yes	Not always

Simple View

- Pogled prikazan u nastavku je primer jednostavnog pogleda
- Podupit izvlači podatke iz samo jedne tabele i nema join funkciju ili bilo koji funkciju grupe
- Pošto je to jednostavan pogled, INSERT, UPDATE, DELETE, MERGE operacije koje utiču na osnovnu tabelu mogu biti izvedene kroz pogled

```
CREATE OR REPLACE VIEW view_euro_countries
AS SELECT country_id, country_name, capitol
    FROM wf_countries
    WHERE location LIKE '%Europe';
```

- Imena kolona u SELECT iskazu mogu imati alijase kao na sledećem primeru
- Priemtiti da alijasi mogu takođe biti izlistani posle CREATE VIEW iskaza i pre SELECT podupita

```
CREATE OR REPLACE VIEW view_euro_countries
AS SELECT country_id AS "ID", country_name AS "Country",
      capitol AS "Capitol City"
   FROM wf_countries
   WHERE location LIKE '%Europe';
```

```
CREATE OR REPLACE VIEW view_euro_countries("ID", "Country",
    "Capitol City")
AS SELECT country_id, country_name, capitol
   FROM wf_countries
   WHERE location LIKE '%Europe';
```

- Moguće je napraviti pogled bez obzira da li postoji osnovna tabela
- Dodajući reč FORCE na CREATE VIEW iskaz kreira se pogled
- Kao DBA, ova opcija može biti korisna tokom razvoja db, posebno ako čekate za potrebne privilegije na referencirani objekat
- FORCE opcija će kreirati pogled bez obzira što ima nedostatke
- NOFORCE opcija je difolt kada se kreira pogled

Complex View

- Kompleksni pogledi su pogledi koji mogu da se sastoje od funkcija grupe i od join
- sledeći primer kreira pogled iz kojeg se dobija podatak iz dve tabele

```
CREATE OR REPLACE VIEW view_euro_countries
   ("ID", "Country", "Capitol City", "Region")
AS SELECT c.country_id, c.country_name, c.capitol, r.region_name
   FROM wf_countries c JOIN wf_world_regions r
   USING (region_id)
   WHERE location LIKE '%Europe';
```

```
SELECT *
FROM view_euro_countries;
```

Complex View

ID	Country	Capitol City	Region
375	Republic of Belarus	Minsk	Eastern Europe
48	Republic of Poland	Warsaw	Eastern Europe
421	Slovak Republic	Bratislava	Eastern Europe
36	Republic of Hungary	Budapest	Eastern Europe
90	Republic of Turkey	Ankara	Eastern Europe
40	Romania	Bucharest	Eastern Europe
373	Republic of Moldova	Chisinau	Eastern Europe
370	Republic of Lithuania	Vilnius	Eastern Europe
371	Republic of Latvia	Riga	Eastern Europe
372	Republic of Estonia	Tallinn	Eastern Europe
...	...	...	...

- Funkcija grupe se može dodati na kompleksni pogled iskaze

```
CREATE OR REPLACE VIEW view_high_pop
   ("Region ID", "Highest population")
AS SELECT region_id, MAX(population)
   FROM wf_countries
   GROUP BY region_id;
```

```
SELECT * FROM view_high_pop;
```

Region ID	Highest population
5	188078227
9	20264082
11	131859731
13	107449525
14	74777981
15	78887007
17	62660551
18	44187637
21	298444215
...	...

Modifying a View

- Za modifikovanje pogleda bez potrebe za odbacivanjem a zatim za ponovnim vraćanjem, koristiti OR REPLACE opciju u CREATE VIEW iskazu
- Stari pogled je zamenjen sa novom verzijom:

```
CREATE OR REPLACE VIEW view_euro_countries
AS SELECT country_id, region_id, country_name, capitol
   FROM wf_countries
   WHERE location LIKE '%Europe';
```

15-2 DML Operations and Views

- Pogledi omogućavaju korisnicima da naprave promene na tabelama ispod pogleda
- Kao DBA poželjno je stvoriti ograničenja na neke poglede nad podacima

DML Statements and Views

- DML operacije INSERT, UPDATE i DELETE se mogu izvesti na jednostavnim pogledima
- Ove operacije se mogu iskoristiti za izmenu podataka u osnovnim tabelama koje leže ispod pogleda (underlying)
- Ako se kreira pogled koji omogućava korisnicima da vide zabranjene informacije korišćenjem WHERE iskaza, korisnici i dalje mogu izvesti DML operacije na svim kolonama pogleda
- Npr, pogled prikazan na desnoj strani je napravljen za menadžere sektora 50 iz employees db
- Namera ovog pogleda je da dozvoli menadžerima sektora 50 da vide informacije o njihovim zaposlenima

```
CREATE VIEW view_dept50
AS SELECT department_id, employee_id,first_name, last_name, salary
 FROM copy_employees
 WHERE department_id = 50;
```

```
SELECT * FROM view_dept50;
```

DEPARTMENT_ID	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY
50	124	Kevin	Mourgos	5800
50	141	Trenna	Rajs	3500
50	142	Curtis	Davies	3100
50	143	Randall	Matos	2600
50	144	Peter	Vargas	2500

Controling Views

- Korišćenje pogleda kao što je prikazano, moguće je uraditi INSERT, UPDATE i DELETE informacije za sve redove u pogledu, čak i ako ovo rezultuje stvaranjem reda koji više nije deo pogleda
- Ovo možda nije ono što je bila namera DBA kada je pogled kreiran
- Za kontrolu pristupa podacima, dve opcije se mogu dodati CREATE VIEW iskazu: WITH CHECK OPTION i WITH READ ONLY

Views with CHECK Option

- Pogled je definisan bez WITH CHECK OPTION:

```
CREATE VIEW view_dept50
AS SELECT department_id, employee_id, first_name, last_name, salary
   FROM copy_employees
   WHERE department_id = 50;
```

- Korišćenjem pogleda, employee_id 124 je promenjen u dept_id 90

```
UPDATE view_dept50
SET department_id = 90
WHERE employee_id = 124;
```

```
1 row(s) updated.
```

- Updejt je uspeo, čak iako ovaj zaposleni sada nije deo pogleda
- WITH CHECK OPTION osigurava da DML operacije izvedene na pogledu ostaju unutar domena pogleda

- Bilo koji pokušaj za promenom broja sektora za bilo koji red u pogledu neće uspeti pošto on ne ispunjava WITH CHECK OPTION ograničenje
- Primetiti u primeru ispod da je WITH CHECK OPTION CONSTRAINT dato ime view_dept50_check

```
CREATE OR REPLACE VIEW view_dept50
AS SELECT department_id, employee_id, first_name, last_name, salary
   FROM employees
   WHERE department_id = 50
WITH CHECK OPTION CONSTRAINT view_dept50_check;
```

- Ako pokušamo modifikovati red u pogledu koji će ga uzeti izvan domena pogleda, javiće se greška

```
UPDATE view_dept50
SET department_id = 90
WHERE employee_id = 124;
```

```
ORA-01402: view WITH CHECK OPTION where-clause violation
```

Views with READ ONLY

- Opcija WITH READ ONLY osigurava da se nikakve DML operacije neće desiti kroz pogled
- Bilo koji pokušaj za izvršenjem INSER, UPDATE, DELETE iskaza će rezultovati u Oracle server grešku

```
CREATE OR REPLACE VIEW view_dept50
AS SELECT department_id, employee_id, first_name, last_name, salary
 FROM employees
 WHERE department_id = 50
WITH READ ONLY;
```

DML Restrictions

- Jednostavni pogledi i kompleksni pogledi se razlikuju u njihovoj sposobnosti da omoguće DML operacije kroz pogled
- Za jednostavne poglede, DML operacije se mogu izvesti kroz pogled
- Za kompleksne poglede, DML operacije nisu uvek dozvoljene
- Sledeća tri pravila se moraju uzeti u obzir pri izvođenju DML operacija na pogledima
- Ne može se odstraniti red iz osnovne tabele ispod pogleda ako pogled sadrži bilo kole od sledećeg:
    - funkcije grupe
    - GROUP BY izraz
    - DISTINCT službenu reč
    - pseudokolonu ROWNUM službenu reč
- Ne mogu se modifikovati podaci kroz pogled ako pogled sadrži:
    - funkcije grupe
    - GROUP BY iskaz
    - DISTINCT službenu reč
    - pseudokolonu ROWNUM službenu reč
    - kolone definisane sa izrazima
- Ne može se dodati podatak kroz pogled ako pogled uključuje:
    - funkcije grupe

- GROUP BY iskaz
- DISTINCT službenu reč
- pseudokolonu ROWNUM službenu reč
- kolone definisane sa izrazima
- ako ne uključuje NOT NULL kolone u osnovnoj tabeli

15-3 Managing Views

- Kada pogled više nije potreban ili treba da se modifikuje, postoje načini da se to izvede
- Treba znati kako se briše pogled, kreira inline pogled i konstruiše SELECT iskaz za proizvodnju sortirane liste podataka

Deleting a View

- Pošto pogled ne sadrži svoje podatke, odstranjivanje pogleda ne utiče na podatke u tabeli ispod pogleda
- Ako se pogled koristi za INSERT, UPDATE ili DELETE podataka u prošlosti, ovo promene na osnovnu tabelu ostaju
- Brisanje pogleda samo odstranjuje definiciju pogleda iz db
- Pogledi su smešteni kao SELECT iskazi u DD
- Samo kreator ili korisnik sa DROP ANY VIEW privilegijama može odstraniti pogled
- SQL sintaksa za odstranjivanje pogleda je: DROP VIEW viewname;

Inline Views

- Inline pogledi se takođe odnose kao podupiti unutar FROM iskaza
- Ubacuje se podupit u FROM iskaz kao da je podupit ime tabele
- Inline pogledi se najčešće koriste za pojednostavljenje kompleksnih upita odstranjivanjem join operacija i pakovanjem nekoliko upita u jedan
- Kao u sledećem primeru, FROM iskaz sadrži SELECT iskaz koji vraća podatak poput bilo kojeg SELECT iskaza
- Vraćeni podatak sa podupitom je dat kao alijas (d), koji se onda koristi u međudejstvu sa glavnim upitom za vraćanje izabranih kolona iz oba izvora upita

```
SELECT e.last_name, e.salary, e.department_id, d.maxsal
FROM employees e,
       (SELECT department_id, max(salary) maxsal
        FROM employees
        GROUP BY department_id) d
WHERE e.department_id = d.department_id
AND e.salary = d.maxsal;
```

- Inline pogled nalazi najveću platu za svaki sektor, a upit zatim prikazuje ime zaposlenog sa tom platom
- Jedan inline pogled mora imati alijas (u primeru je “d“) pošto funkcioniše poput imena tabele u FROM iskazu i “SELECT department_id, max(salary)...“ nije validno ime tabele

TOP-N-ANALYSIS

- Top-n-analiza je SQL operacija koja se koristi za rangiranje rezultata
- Da bi se koristila top-n-analiza je korisna kada treba vratiti 5 glavnih rekorda, ili top-n-records, od seta rezultata dobijenih iz upita

```
SELECT ROWNUM AS "Longest employed", last_name, hire_date
FROM employees
WHERE ROWNUM <=5
ORDER BY hire_date;
```

Longest employed	LAST_NAME	HIRE_DATE
1	King	17-Jun-1987
4	Whalen	17-Sep-1987
2	Kochhar	21-Sep-1989
3	De Haan	13-Jan-1993
5	Higgins	07-Jun-1994

- Rezultat ovog upita nisu šta se očekuje. Razlog za to se ORDER BY iskaz uvek realizuje poslednji, tako da redovi su poređani posle dobijanja brojeva.
- Top*-n-analiza upit koristi inline pogled (podupit) za vraćanje seta rezultata
- Može se koristiti ROWNUM u upitu za dodelu broja reda u set rezultata
- Glavni upit onda koristi ROWNUM za uređivanje podataka i vraća najviših pet:

```
SELECT ROWNUM AS "Longest employed", last_name, hire_date
FROM (SELECT last_name, hire_date
      FROM employees
      ORDER BY hire_date)
WHERE ROWNUM <=5;
```

- Korišćenjem inline pogleda, ORDER BY iskaz je izveden pre nego je ROWNUM dodat

Longest employed	LAST_NAME	HIRE_DATE
1	King	17-Jun-1987
2	Whalen	17-Sep-1987
3	Kochhar	21-Sep-1989
4	Hunold	03-Jan-1990
5	Ernst	21-May-1991

- In the example above, the inline view first selects the list of last_names and hire_dates of the employees:

```
(SELECT last_name, hire_date FROM employees...
```

- Then the inline view orders the years from oldest to newest.

```
...ORDER BY hire_date)
```

- Spoljni upit WHERE iskaza sekoristi za restrikciju broja vraćenih redova i mora koristiti < ili <= operator

```
SELECT ROWNUM AS "Longest employed", last_name, hire_date
FROM (SELECT last_name, hire_date
      FROM employees
      ORDER BY hire_date)
WHERE ROWNUM <=5;
```

Longest employed	LAST_NAME	HIRE_DATE
1	King	17-Jun-1987
2	Whalen	17-Sep-1987
3	Kochhar	21-Sep-1989
4	Hunold	03-Jan-1990
5	Ernst	21-May-1991