Računarstvo i informatika

Katedra za računarstvo
Elektronski fakultet u Nišu

Sistemi baza podataka

Optimizacija upita

Letnji semestar 2015

Sadržaj

- Obrada upita
- Optimizacija upita
- Statistika baze podataka
- Indeksi
- Oracle optimizacija
- Oracle statistika
- Oracle indeksi
- Oracle execution plan

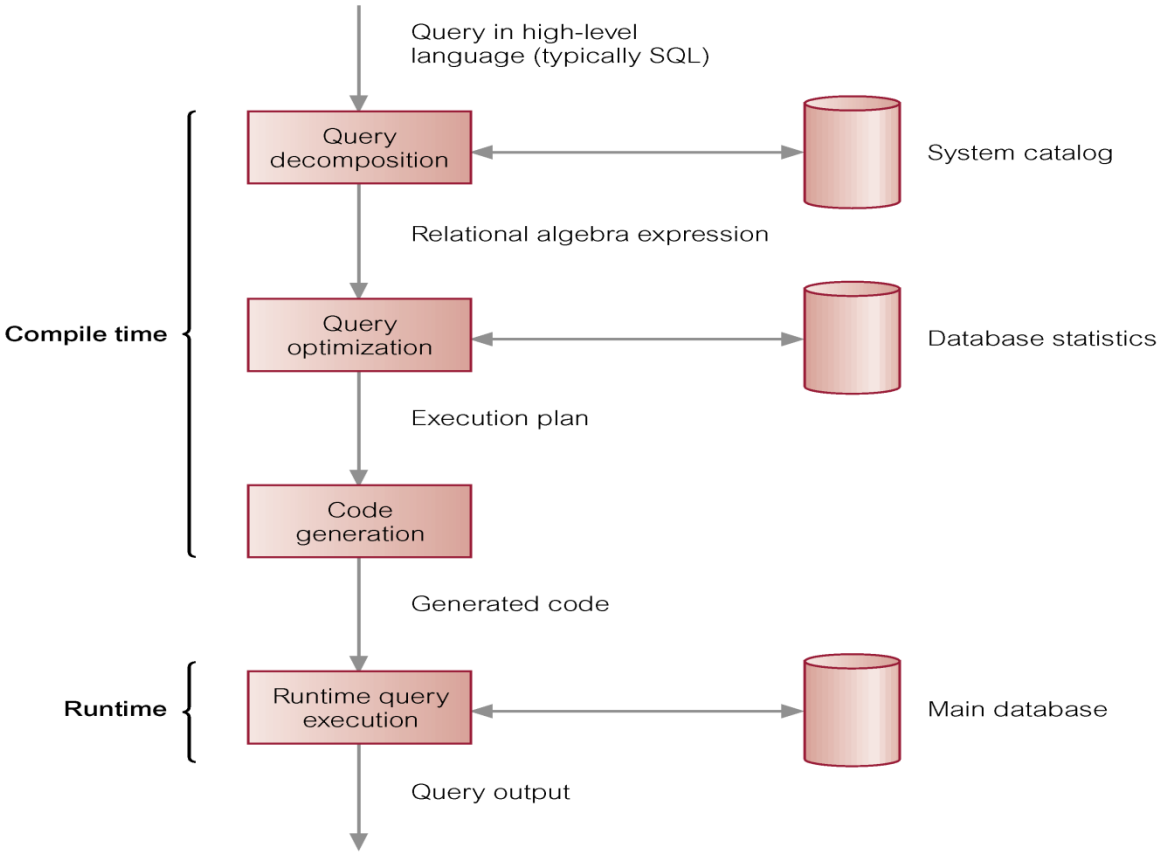
Obrada upita

- SQL pripada grupi deklarativnih jezika.
- Korisnik specificira šta treba uraditi (koje podatke treba pribavitit) a ne kako kako nešto uraditi (kako pristupiti podacima).
- Povećana univerzalna upotrebljivost SQL-a jer korisnik ne mora da vodi računa o optimizaciji izvršavanja upita.
- DBMS ima bolju kontrolu nad performansama funkcionisanja sistema.

Obrada upita

- Obrada upita obuhvata niz aktivnosti čiji je cilj pribavljanje podataka iz baze podataka.
- Ciljevi:
  - transformacija upita napisanog korišćenjem nekog jezika visokog nivoa (npr. SQL) u validnu i efikasnu strategiju izvršavanja opisanu korišćenjem nekog jezika niskog nivoa
  - Izvršavanje rezultujuće strategije u cilju pribavljanja zahtevanih podataka

Obrada upita



Optimizacija upita

- Optimizacija upita predstavlja aktivnost izbora efikasnije strategije izvršavanja prilikom obrade upita.
- Transformacijom upita može se dobiti veći broj ekvivalentnih strategija izvršavanja.
- Optimizacijom upita bira se strategija koja minimizira upotrebu računarskih resursa (vreme izvršavanja, količina obrađenih podataka, operacije sa diskom, saobraćaj na mreži i sl.).
- Optimizacija upita se u velikoj meri bazira na postojanju statistike baze podataka.
- Statistika baze podataka čuva informacije o relacijama, atributima i indeksima.
- Preciznost i ažurnost statistike baze podataka u velikoj meri određuje kvalitet odabrane strategije za izvršavanje upita.

Optimizacija upita

- Postoje dva načina na koji se može vršiti optimizacija upita:
  - Dinamički, svaki put kada se upit izvršava
  - Statički, kada se upit prvi put prosledi na izvršenje
- Dinamička optimizacija upita
  - Prednosti: ažurnost svih informacija koje su neophodne prilikom izbora optimalne strategije izvršavanja
  - Nedostaci: degradacija performansi izvršavanja upita (povećava ukupno vreme izvršavanja upita), smanjen broj alternativa koje se obrađuju, smanjena verovatnoća pronalaženja najefikasnije strategije izvršavanja

Optimizacija upita

- Statička optimizacija upita
  - Prednosti: smanjen uticaj na performanse upita, povećan broj alternativa koje se razmatraju, veća verovatnoća nalaženja optimalne strategije izvršavanja
  - Nedostaci: strategija koja je bila optimalna u trenutku optimizacije upita ne mora biti optimalna (usled promene stanja sistema) u trenutku izvršavanja upita.
- Može se koristiti hibridni pristup
  - vrši se ponovna optimizacija upita kad god sistem detektuje da je statistika baze podataka značajno promenjena

Optimizacija upita

- Postoje dve glavne strategije za optimizaciju upita:
  - tehnika bazirana na korišćenju heurističkih pravila na osnovu kojih se određuje redosled operacija u upitu
  - tehnika bazirana na poređenju različitih strategija na osnovu njihove relativne cene.
- U praksi se ove dve strategije najčešće kombinuju kako bi se dobili najbolji rezultati.

Statistika baze podataka

- Statistika baze podataka igra izuzetno značajnu ulogu u procesu optimizacije upita (naročito kod tehnika koje se zasnivaju na proceni relativne cene operacija).
- Tipično se u bazi podataka čuvaju sledeće statističke informacije:

1. Za svaku tabelu
   a) Broj vrsta u tabeli
   b) Faktor blokiranja (broj vrsta koje staju u jedan blok)
   c) Broj blokova koji tabela zauzima
   d) Prosečna veličina vrste
2. Za svaku kolonu u tabeli
   a) Broj različitih vrednosti (DISTINCT values)
   b) Broj NULL vrednosti u koloni
   c) Minimalna i maksimalna vrednost kolone
   d) Procenjena selektivnost kolone (histogram)

Statistika baze podataka

3. Za svaki indeks
   a) Broj nivoa u indeksu
   b) Broj listova
4. Sistemska statistika
   a) I/O performanse i stepen iskorišćenja
   b) CPU performanse i stepen iskorišćenja

- Statistika baze podataka se čuva u sistemskim tabelama (meta podaci, meta tabele).

Statistika baze podataka

- Objekti u bazi podataka se stalno menjaju tako da je potrebno redovno ažurirati prikupljene podatke.
- Ukoliko bi se statistika ažurirala prilikom svake izmene baze podataka (SELECT, UPDATE, DELETE, INSERT) to bi dovelo do značajnog degradiranja performansi sistema.
- Alternativno se koristi pristup da DBMS ažurira statistiku periodično, tokom noći ili kada sistem nije opterećen.

Indeksi

- Kod relacionih baza podataka indeksi predstavljaju strukture podataka koje se koriste da ubrzaju operacije pretraživanja podataka i operacija sortiranja podataka.
- Za implementaciju indeksa se mogu koristiti različite strukture podataka:
  - Balansirana stabla traženja
  - B^+ stabla
  - Rasute tablice
- Indeks može biti definisan kao unique i non-unique (podrazumevano).
- Unique indeksi definišu ograničenje nad tabelom tako da vrednosti koje se indeksiraju moraju biti jedinstvene.

Indeksi

- Po svojoj arhitekturi indeksi mogu da budu:
  - Non-clustered – listovi stabla čuvaju referencu na blok koji sadrži odgovarajuću podatak
  - Clustered – listovi stabla često ne sadrže referencu već predstavljaju same podatke, blokovi na disku su uređeni u istom redosledu kao i čvorovi indeksa.
- Clustered indeksi postiži izuzetno dobre performanse kada se podacima pristupa sekvencijalno ili se pristupa opsegu podataka.

Indeksi

- Prilikom izvršavanja upita DBMS donosi odluku da li performanse mogu da se poboljšaju korišćenjem nekog indeksa.
- DBMS mora da održava sve indekse u bazi bez obzira da li se oni koriste ili ne.
- Za postizanje optimalnih performansi potrebno je obrisati sve indekse koji se ne koriste ili se koriste vrlo retko.
- Indeks koji poboljša performanse jednog upita može drastično da degradira performanse nekog drugog upita.
- Za analizu korišćenja indeksa koristi se plan izvršavanja – EXECUTION PLAN

Indeksi

- Preporuke za kreiranje indeksa:
  - Kolone koje se često koriste u WHERE klauzuli
  - Kolone koje se koriste za spajanje tabela
  - Kreirati indekse koji imaju veliku selektivnost (što manji procenat vrsta u tabeli koje za indeksirane kolone imaju istu vrednost)
  - Ne treba indeksirati kolone čije vrednosti se često menjaju
  - Voditi računa da indeksi degradiraju performanse operacija INSERT, UPDATE i DELETE
    - OLTP (Online Transaction Processing) baze podataka
    - Data warehouse i OLAP (Online Analytical Processing) baze podataka
  - Ne indeksirati kolone koje se u WHERE klauzuli pojavljuju kao argumenti funkcija ili operatora

Indeksi

- Preporuke za kreiranje kompozitnih indeksa:
  - Kolone koje se često javljaju zajedno u WHERE klauzuli povezane AND operatorom
  - Kolone koje se često zajedno selektuju pri čemu se jedna ili više tih kolona javlja u WHERE klauzuli
- Redosled kolona u kompozitnom indeksu utiče na to da li će DBMS izabrati taj indeks ili ne prilikom izvršavanja upita.
- Kolone koje se koriste u WHERE klauzuli treba da budu navedene na početku kompozitnog indeksa.

Oracle statistika

- Ukoliko se u bazi podataka kreira novi objekat ili se promeni količina podataka trenutna statistika baze podataka više ne odslikava pravo stanje sistema.
- Neažurna statistika baze podataka može u velikoj meri da oteža pa čak i onemogući pravilan izbor optimalnog plana izvršavanja upita.
- Korisnik baze podataka ima obavezu da obezbedi očuvanje ažurnosti statistike.
- Statistika baze podataka se kod Oracle-a čuva u Data dictionary tabelama ili u tabelama koje se namenski kreiraju u šemi korisnika.
- Data dictionary – kolekcija sistemskih (meta) tabela.

Oracle statistika

- Za pregled statistika koriste se odgovarajući pogledi iz Data dictionary dela baze (USER, ALL ili DBA)

- ALL_TABLES
- ALL_OBJECT_TABLES
- ALL_TAB_STATISTICS
- ALL_TAB_COL_STATISTICS
- ALL_TAB_HISTOGRAMS
- ALL_INDEXES
- ALL_IND_STATISTICS
- ALL_CLUSTERS
- ALL_TAB_PARTITIONS
- ALL_TAB_SUBPARTITIONS
- ALL_IND_PARTITIONS
- ALL_IND_SUBPARTITIONS
- ALL_PART_COL_STATISTICS
- ALL_PART_HISTOGRAMS
- ALL_SUBPART_COL_STATISTICS
- ALL_SUBPART_HISTOGRAMS

Oracle statistika

- ANALYZE – ova naredba se može koristiti za prikupljanje statistike za određenu tabelu ili indeks.
- Statistika se može odrediti precizno ili se proceniti na osnovu određenog procenta podataka.

```
ANALYZE TABLE employees COMPUTE STATISTICS;
ANALYZE INDEX employees_pk COMPUTE STATISTICS;

ANALYZE TABLE employees ESTIMATE STATISTICS SAMPLE 100 ROWS;
ANALYZE TABLE employees ESTIMATE STATISTICS SAMPLE 15 PERCENT;
```

Oracle statistika

- DBMS_UTILITY – kolekcija funkcija koje omogućavaju rad sa statistikom baze podataka.
- Funkcije se mogu primeniti na čitavu šemu ili na pojedine tabele.

```
EXEC DBMS_UTILITY.analyze_schema('SCOTT','COMPUTE');
EXEC DBMS_UTILITY.analyze_schema('SCOTT','ESTIMATE', estimate_rows => 100);
EXEC DBMS_UTILITY.analyze_schema('SCOTT','ESTIMATE', estimate_percent => 15);

EXEC DBMS_UTILITY.analyze_database('COMPUTE');
EXEC DBMS_UTILITY.analyze_database('ESTIMATE', estimate_rows => 100);
EXEC DBMS_UTILITY.analyze_database('ESTIMATE', estimate_percent => 15);
```

Oracle statistika

- DBMS_STAT – kolekcija funkcija koje omogućavaju rad sa statistikom baze podataka.
- Oracle preporučuje korišćenje funkcija iz ovog paketa.

```
EXEC DBMS_STATS.gather_database_stats;
EXEC DBMS_STATS.gather_database_stats(estimate_percent => 15);

EXEC DBMS_STATS.gather_schema_stats('SCOTT');
EXEC DBMS_STATS.gather_schema_stats('SCOTT', estimate_percent => 15);

EXEC DBMS_STATS.gather_table_stats('SCOTT', 'EMPLOYEES');
EXEC DBMS_STATS.gather_table_stats('SCOTT', 'EMPLOYEES', estimate_percent => 15);

EXEC DBMS_STATS.gather_index_stats('SCOTT', 'EMPLOYEES_PK');
EXEC DBMS_STATS.gather_index_stats('SCOTT', 'EMPLOYEES_PK', estimate_percent => 15);

EXEC DBMS_STATS.delete_database_stats;
EXEC DBMS_STATS.delete_schema_stats('SCOTT');
EXEC DBMS_STATS.delete_table_stats('SCOTT', 'EMPLOYEES');
EXEC DBMS_STATS.delete_index_stats('SCOTT', 'EMPLOYEES_PK');
```

Oracle indeksi

- Oracle podržava nekoliko tipova indeksa:
  - Normalni indeksi – podrazumevana je struktura B stabla
  - Funkcionalni indeksi – indeksi koji se formiraju nad vrednostima funkcija koje se primenjuju nad kolonama table
  - Bitmap indeksi – za svaku ključnu vrednost se formira bitmapa koja pamti da li određena vrsta sadrži tu vrednost ili ne
  - Particionisani indeksi – formira se particija za svaku ključnu vrednost
  - Domenski indeksi

- Oracle automatski kreira indekse za ograničenja tipa PRIMARY KEY i UNIQUE.
- U svim ostalim slučajevima korisnik je zadužen za kreiranje indeksa korišćenjem CREATE INDEX naredbe.

Oracle indeksi

```
CREATE [UNIQUE] INDEX index_name
  ON table_name (column1, column2, . column_n)
  [ COMPUTE STATISTICS ];
```

```
CREATE INDEX ord_customer_ix
  ON orders (customer_id);
```

Kreiranje indeksa nad jednom kolonom.

```
CREATE INDEX supplier_idx
  ON supplier (supplier_name, city);
```

Kreiranje kompozitnog indeksa.

```
CREATE INDEX supplier_idx
  ON supplier (supplier_name, city)
COMPUTE STATISTICS;
```

Eksplicitno generisanje statistike prilikom kreiranja indeksa.

Oracle indeksi

```
CREATE [UNIQUE] INDEX index_name
  ON table_name (function1, function2, . function_n)
  [ COMPUTE STATISTICS ];
```

```
CREATE INDEX supplier_idx
   ON supplier (UPPER(supplier_name));
```

Kreiranje indeksa baziranog na funkciji.

```
SELECT supplier_id, supplier_name, UPPER(supplier_name)
FROM supplier
WHERE UPPER(supplier_name) IS NOT NULL
ORDER BY UPPER(supplier_name);
```

Vrednost funkcije se koristi u WHERE klauzuli.

Oracle indeksi

```
ALTER INDEX index_name
  RENAME TO new_index_name;
```

Izmena imena indeksa.

```
ALTER INDEX index_name
  REBUILD COMPUTE STATISTICS;
```

Ažuriranje statistike indeksa.

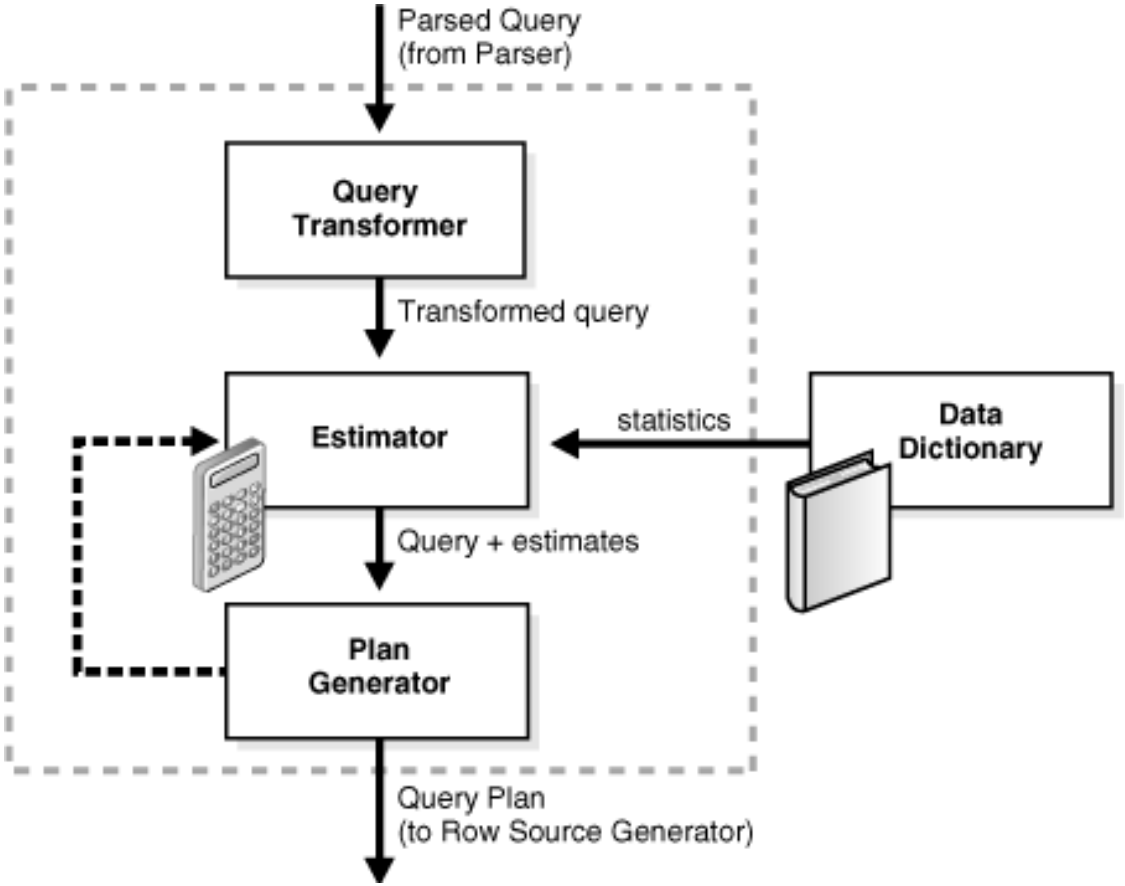
```
DROP INDEX index_name;
```

Brisanje indeksa.

Oracle Query optimizer

- Query optimizer je Oracle komponenta koja određuje optimalan plan za izvršavanje SQL naredbi.
- Optimizacija SQL naredbi se vrši u tri koraka:
  1. Određivanje mogućih planova izvršavanja određene SQL naredbe
  2. Za svaki plan se na osnovu statistike baze podataka vrši procena cene izvršanja SQL naredbe
  3. Poređenjem procenjenih cena izvršavanja bira se optimalan plan izvršavanja SQL naredbe.
- Osim procene cene izvršavanja SQL naredbi (Cost Based Optimization, CBO) Oracle Query optimizer koristi i skup heurističkih pravila.

Oracle Query optimizer



Oracle Query optimizer

- Prilikom procene cene izvršenja plana koriste se tri parametra:
  1. Selectivity – definiše procenat selektovanih vrsta u tabeli, pogledu ili rezultujućoj tabeli (rezultat spoja ili neke druge operacije)
  2. Cardinality – ukupan broj vrsta u tabeli, pogledu ili rezultujućoj tabeli (rezultat spoja ili neke druge operacije)
  3. Cost – cena izvršenja SQL naredbe u jedinicama obrade ili resursa (disk U/I, CPU vreme ili potreban memorija)

Oracle explain plan

- EXPLAIN PLAN – naredba koja prikazuje plan izvršenja koji je DBMS odabrao za SELECT, INSERT, UPDATE i DELETE naredbe.

EXPLAIN PLAN [INTO *table_name*] FOR *sql_command*;

- Oracle DBMS analizira specificiranu SQL komandu i rezultat te analize odnosno odabrani plan izvršenja smešta u specificiranu tabelu.
- Ukoliko se izostavi INTO *table_name* deo rezultat analize se smešta u tabelu PLAN_TABLE.
- ORACLE nudi skript UTLXPLAN.SQL koji se može koristiti za kreiranje tabele koja će prihvatiti rezultat EXPLAIN PLAN naredbe.

Oracle explain plan

```
EXPLAIN PLAN FOR
  SELECT last_name FROM employees;
```

Analiza izvršenja SQL komande.

```
EXPLAIN PLAN
  SET STATEMENT_ID = 'st1' FOR
SELECT last_name FROM employees;
```

Rezultatu analize se dodeljuje ID.

```
EXPLAIN PLAN
   SET STATEMENT_ID = 'st1'
   INTO my_plan_table
 FOR
SELECT last_name FROM employees;
```

Rezultatu analize se smešta u custom definisanu tabelu.

Oracle explain plan

- Najvažnije kolone u PLAN_TABLE tabeli su:
  - OPERATION – naziv operacije koja se izvršava
  - OPTIONS – parametri operacije koja se izvršava
  - OBJECT – objekat nad kojim se izvršava operacija
  - ID – ID operacije
  - PARENT_ID – ID roditeljske operacije

Oracle explain plan

- Vrednosti koje se mogu naći u koloni OPERATION:

1. DELETE STATEMENT
2. INSERT STATEMENT
3. SELECT STATEMENT
4. UPDATE STATEMENT
5. AND-EQUAL
6. CONNECT BY
7. CONCATENATION
8. COUNT
9. DOMAIN INDEX
10. FILTER
11. FIRST ROW
12. FOR UPDATE
13. HASH JOIN
14. INDEX
15. INLIST ITERATOR
16. INTERSECTION
17. MERGE JOIN
18. MINUS
19. NESTED LOOPS
20. PARTITION,
21. REMOTE
22. SEQUENCE
23. SORT
24. TABLE ACCESS
25. UNION
26. VIEW

Oracle explain plan

```
SELECT ID, PARENT_ID, OBJECT_NAME, OPERATION, OPTIONS
    FROM PLAN_TABLE;
```

Prikazuje sadržaj PLAN_TABLE tabele

ID	PARENT_ID	OBJECT_NAME	OPERATION	OPTIONS
0			SELECT STATEMENT	
1	0		SORT	GROUP BY
2	1		NESTED LOOPS	
3	2	RADNIK	TABLE ACCESS	FULL
4	2	SEKTOR	TABLE ACCESS	BY INDEX ROWID
5	4	SYS_C005431	INDEX	UNIQUE SCAN

6 rows selected

Oracle explain plan

```
SELECT PLAN_TABLE_OUTPUT FROM
    TABLE(DBMS_XPLAN.DISPLAY());
```

Prikazuje sadržaj PLAN_TABLE tabele

```
PLAN_TABLE_OUTPUT
------------------------------------------------------------------------------------------
Plan hash value: 2491799117

-------------------------------------------------------------------------------------------
| Id  | Operation                     | Name        | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |             |     2 |    74 |     5  (20)| 00:00:01 |
|   1 |  SORT GROUP BY                |             |     2 |    74 |     5  (20)| 00:00:01 |
|   2 |   NESTED LOOPS                |             |     2 |    74 |     4   (0)| 00:00:01 |
|*  3 |    TABLE ACCESS FULL          | RADNIK      |     4 |    88 |     3   (0)| 00:00:01 |
|   4 |    TABLE ACCESS BY INDEX ROWID| SEKTOR      |     1 |    15 |     1   (0)| 00:00:01 |
|*  5 |     INDEX UNIQUE SCAN         | SYS_C005431 |     1 |       |     0   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

   3 - filter("BROD">4)
   5 - access("BROD"="BROJOD")
       filter("BROJOD">4)

19 rows selected
```

Oracle optimizer hints

- Optimizer hints omogućavaju korisniku da preuzme kontrolu od DBMS-a prilikom izbora plana za izvršavanje SQL naredbi.
- Optimizer hints mogu da se odnose na:
  - Tabele
  - Grupu tabela
  - Blok (deo) SQL naredbe
  - SQL naredbu

Oracle optimizer hints

{DELETE|INSERT|MERGE|SELECT|UPDATE} /*+ hint [text] [hint[text]]... */

{DELETE|INSERT|MERGE|SELECT|UPDATE} --+ hint [text] [hint[text]]...

```
SELECT /*+ INDEX (employees emp_department_ix)*/
       employee_id, department_id
  FROM employees
 WHERE department_id > 50;
```

Forsira se plan izvršenja koji koristi indeks emp_department_ix