

RĪGAS TEHNISKĀ UNIVERSITĀTE

Datorzinātnes, informācijas tehnoloģijas un enerģētikas fakultāte

2.praktiskais darbs

**Datu bāzes datu fiziskā sadalīšana (partīcijas)**

Izstrādāja: Natans Šalamberidze

171RMC203

Pārbaudīja: Profesors JānisEiduks

2023./2024. māc. Gads

Darba pamatuzdevums ir izpētīt SQL vaicājumu ātrdarbības uzlabošanu izmantojot tabulu un ineksu sadalīšanu nodalījumos jeb partīcijās. Tas būtiski atvieglina datu meklēšanas procesu, jo samazinās datu apjoms, kas jācaurskata.

1. Jādefinē 3 jaunas tabultelpas (tablespace), jāiegūst to metadati un jāveic iegūto metadatu semantikas komentēšanu.

2. Divās tabultēlpās jādefinē katrā pa vienai tabulai. Pirmās tabulas rakstu skaits >= 100 000, otrās tabulas rakstu skaits >= 1000. Tabulas ir loģiski saistītas. Datu ievadei jā lieto PL/SQL programmas un gadījumu vērtību ģenerēšanas funkcijas (iepriekšējais semestris).

2. Definēt tabulas ar: diapazona sadalījuma partīcijām, intervāla sadalījuma partīcijām, heš funkcijas sadalījuma partīcijām, saraksta sadalījuma partīcijām un atsauces sadalījuma partīcijām. Tabulš datus iekļaut no pirmajām definētajām tabulām (tā būs mazāk darba).

3. Izveidot 3 tipveida vaicājumus, kuri veic datu izgūšanu no tabulām ar par partīcijām un tabulām bez partīcijām. Vienā vaicājumā tiek iegūta viena vērtība (agregāts), otrā tiek iegūti apmēram 15 % rakstu un trešajā mazāk par 1% tabulas datu.

4.Veikt vaicājumus un iegūt vaicājumu novērtējumus COST. Datus WORD dokumentā iekļaut pārskatāmā tabulā, lai varētu veikt analīzi.

5.Turpinājumā jāizvērtē un jāpamato iegūtie rezultāti visiem partīciju veidiem..

6. Nesadalītai tabulai un sadalītām tabulām izveidot lokālos un globālos indeksus.

7. Visām indeksētajām tabulām veikt iepriekš definētoss vaicājumus. Iegūtos rezultātus ierakstīt WORD tabulā. Komentēt rezultātus un izdarīt to novērtējumu.

Pielikumā jāiekļauj minimālā formātā vaicājumu izpildes plāni.

Secinājumi. Ko eksperimentu dati ir parādījuši un obligāti, kāpēc ir šadi rezultāti (skaidrojums).

Datu ievades procedūras minimālā apjomā iekļaut beigās pielikumos.

**1) Jādefinē 3 jaunas tabultelpas (tablespace), jāiegūst to metadati un jāveic iegūto metadatu semantikas komentēšanu.**

CREATE TABLESPACE tablespace\_1

DATAFILE 'df\_1.dbf' SIZE 100M

EXTENT MANAGEMENT LOCAL UNIFORM SIZE 1M;

CREATE TABLESPACE tablespace\_2

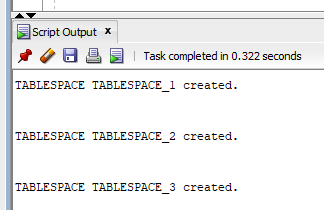
DATAFILE 'df\_2.dbf' SIZE 100M

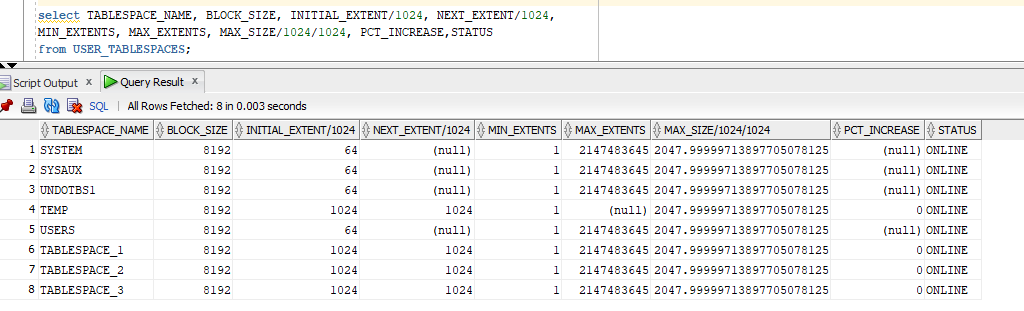
EXTENT MANAGEMENT LOCAL UNIFORM SIZE 1M;

CREATE TABLESPACE tablespace\_3

DATAFILE 'df\_3.dbf' SIZE 100M

EXTENT MANAGEMENT LOCAL UNIFORM SIZE 1M;





Tabultelpas metadati:

TABLESPACE\_NAME: Nosaukums tabultelpai

BLOCK\_SIZE: Bloka izmērs tabultelpā (baitos)

INITIAL\_EXTENT/1024: Sākotnējā extēnta izmērs (kilobaitos)

NEXT\_EXTENT/1024: Nākamā extēnta izmērs (kilobaitos)

MIN\_EXTENTS: Minimālais extēntu skaits

MAX\_EXTENTS: Maksimālais extēntu skaits

MAX\_SIZE/1024/1024: Maksimālais tabultelpas izmērs (megabaitos)

PCT\_INCREASE: Procentuālais palielinājums extēntiem

STATUS: Tabultelpas statuss (ONLINE)

**2. Divās tabultēlpās jādefinē katrā pa vienai tabulai. Pirmās tabulas rakstu skaits >= 100 000, otrās tabulas rakstu skaits >= 1000. Tabulas ir loģiski saistītas. Datu ievadei jā lieto PL/SQL programmas un gadījumu vērtību ģenerēšanas funkcijas (iepriekšējais semestris).**

CREATE TABLE customer (

id NUMBER,

name VARCHAR2(100),

age NUMBER,

phone\_number VARCHAR2(20)

)tablespace tablespace\_1;

CREATE TABLE payments (

id NUMBER,

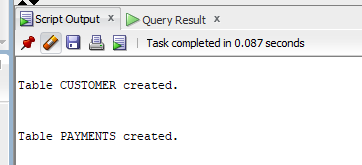
user\_id NUMBER,

invoice\_number VARCHAR2(200),

payment\_date DATE,

amount NUMBER

)tablespace tablespace\_2;



DECLARE

i NUMBER;

TYPE name\_array IS VARRAY(15) OF VARCHAR2(50);

TYPE surname\_array IS VARRAY(15) OF VARCHAR2(50);

first\_names name\_array := name\_array('Jānis', 'Pēteris', 'Māris', 'Andris', 'Āris', 'Artūrs', 'Miks', 'Lauris', 'Renārs', 'Raimonds', 'Kristaps', 'Edgars', 'Juris', 'Rihards', 'Andris');

surnames surname\_array := surname\_array('Bērziņš', 'Liepiņš', 'Ozoliņš', 'Saulītis', 'Kalniņš', 'Ķēniņš', 'Sējējs', 'Priedītis', 'Vītols', 'Liepa', 'Gulbis', 'Krūmiņš', 'Muižnieks', 'Lūsis', 'Šmits');

BEGIN

FOR i IN 1..1000 LOOP

INSERT INTO customer (id, name, age, phone\_number)

VALUES (i,

first\_names(TRUNC(DBMS\_RANDOM.VALUE(1, 15))) || ' ' || surnames(TRUNC(DBMS\_RANDOM.VALUE(1, 15))),

ROUND(DBMS\_RANDOM.VALUE(18, 80)),

TO\_CHAR('+3712' || ROUND(DBMS\_RANDOM.VALUE(1000000, 9999999)))

);

END LOOP;

FOR i IN 1..100000 LOOP

INSERT INTO payments (id, user\_id, invoice\_number, payment\_date, amount)

VALUES (i,

ROUND(DBMS\_RANDOM.VALUE(1, 1000)),

'Rekins - ' || TO\_CHAR(ROUND(DBMS\_RANDOM.VALUE(1000000, 9999999))),

SYSDATE - ROUND(DBMS\_RANDOM.VALUE(1, 365)),

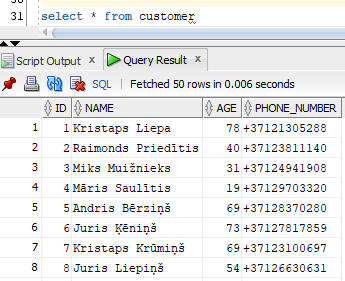
ROUND(DBMS\_RANDOM.VALUE(10, 1000), 2));

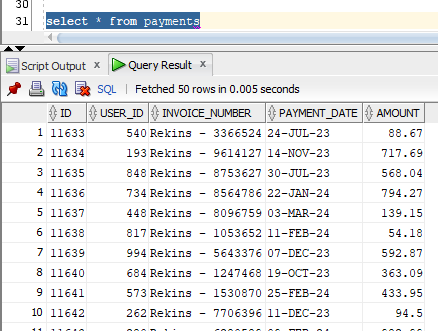
END LOOP;

COMMIT;

END;

/





**Definēt tabulas ar: diapazona sadalījuma partīcijām, intervāla sadalījuma partīcijām, heš funkcijas sadalījuma partīcijām, saraksta sadalījuma partīcijām un atsauces sadalījuma partīcijām. Tabulš datus iekļaut no pirmajām definētajām tabulām (tā būs mazāk darba).**

CREATE TABLE payments\_range\_part

tablespace tablespace\_3

PARTITION BY RANGE (amount)

(

PARTITION part\_1 VALUES LESS THAN (250),

PARTITION part\_2 VALUES LESS THAN (500),

PARTITION part\_3 VALUES LESS THAN (750),

PARTITION part\_4 VALUES LESS THAN (1001)

)

ENABLE ROW MOVEMENT

as (select \* from payments);

/

CREATE TABLE payments\_interval\_part

PARTITION BY RANGE(amount) INTERVAL (250)

(partition amount1 values less than (250) tablespace tablespace\_3)

ENABLE ROW MOVEMENT

as (select \* from payments);

/

create table payments\_hash\_part

partition by HASH (amount)

partitions 4

as select \* from payments;

/

CREATE TABLE payments\_list\_part

PARTITION BY LIST (amount)

(

PARTITION part\_1 VALUES (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 250),

PARTITION part\_2 VALUES (251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 500),

PARTITION part\_3 VALUES (501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 750),

PARTITION part\_4 VALUES (751, 752, 753, 754, 755, 756, 757, 758, 759, 760, 1000),

PARTITION part\_default values (DEFAULT)

)

as SELECT \* from payments;

/

CREATE TABLE customer\_range\_part

tablespace tablespace\_3

PARTITION BY RANGE (id)

(

PARTITION part\_1 VALUES LESS THAN (250),

PARTITION part\_2 VALUES LESS THAN (500),

PARTITION part\_3 VALUES LESS THAN (750),

PARTITION part\_4 VALUES LESS THAN (1001)

)

ENABLE ROW MOVEMENT

as (select \* from customer);

/

alter table customer\_range\_part add constraint PK\_CUST\_ID PRIMARY KEY (id);

/

create table payments\_ref\_part (

id, user\_id not null, invoice\_number, payment\_date, amount,

constraint payment\_cust\_fk foreign key(user\_id) references customer\_range\_part(id))

partition by reference (payment\_cust\_fk)

ENABLE ROW MOVEMENT

as select \* from payments;

**Izveidot 3 tipveida vaicājumus, kuri veic datu izgūšanu no tabulām ar par partīcijām un tabulām bez partīcijām. Vienā vaicājumā tiek iegūta viena vērtība (agregāts), otrā tiek iegūti apmēram 15 % rakstu un trešajā mazāk par 1% tabulas datu.**

select AVG(Amount) from payments; --agregāts

select \* from payments where amount < 155; -- ~15%

select \* from payments where amount > 990; -- ~1%

**4.Veikt vaicājumus un iegūt vaicājumu novērtējumus COST. Datus WORD dokumentā iekļaut pārskatāmā tabulā, lai varētu veikt analīzi.**

**5.Turpinājumā jāizvērtē un jāpamato iegūtie rezultāti visiem partīciju veidiem..**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tabula** | **Vaicājums ar agregātu** | **Vaicājums kas atgriež 15% ierakstu** | **Vaicājums kas atgriež 1% ierakstu** |
| payments | 174 | 174 | 174 |
| payments\_range\_part | 175 | 44 | 46 |
| payments\_interval\_part | 220 | 220 | 220 |
| payments\_hash\_part | 168 | 168 | 168 |
| payments\_list\_part | 169 | 169 | 169 |
| payments\_ref\_part | 175 | 175 | 175 |

Ņemot vērā izvēlētos tipveida vaicājumus, pozitīvo ietekmi uz vaicājuma izpildes laiku sniedza tikai diapazona vai intervālu sadalījums partīcijas. Vaicājumiem, kas atgriež 15% un 1% ierakstu, ir vienāda COST vērtība, jo abos gadījumos rezultāti atrodas vienā partīcijā.

Saraksta sadalījums nav piemērots dotam gadījumam, jo ir tika izvelēta kolona, kurai varbūt pārāk liels vērtību skaits, tāpēc rezultāts bija ne apmierinošs.

Citas sadalījuma metodēs būtiski neietekmēja veiktspēju pie dotiem vaicājumiem.

**6. Nesadalītai tabulai un sadalītām tabulām izveidot lokālos un globālos indeksus.**

Indekss nesadalītai tabulai:  
 CREATE INDEX idx\_amount\_global ON payments(amount);  
  
Indeksi sadalītam tabulas:

Diapazonu:

CREATE INDEX idx\_amount\_local\_partition\_1 ON payments\_range\_part(amount) LOCAL;

drop index idx\_amount\_local\_partition\_1;

CREATE INDEX idx\_amount\_global\_part ON payments\_range\_part(amount);

Intervālu:  
 CREATE INDEX idx\_amount\_local\_interval ON payments\_interval\_part(amount) LOCAL;

drop index idx\_amount\_local\_interval;

CREATE INDEX idx\_amount\_global\_inetrval ON payments\_interval\_part(amount);

Heš:

CREATE INDEX idx\_amount\_local\_hash ON payments\_hash\_part(amount) LOCAL

drop index idx\_amount\_local\_hash;

CREATE INDEX idx\_amount\_global\_hash ON payments\_hash\_part(amount);

Saraksta:

CREATE INDEX idx\_id\_local\_list ON payments\_list\_part(amount) LOCAL;

drop index idx\_id\_local\_list;

CREATE INDEX idx\_id\_global\_list ON payments\_list\_part(amount);

**7. Visām indeksētajām tabulām veikt iepriekš definētoss vaicājumus. Iegūtos rezultātus ierakstīt WORD tabulā. Komentēt rezultātus un izdarīt to novērtējumu.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Bez indeksiem** | | | **Ar indeksiem (local/global)** | | |
| **Tabula** | **Agregāts** | **15%** | **1%** | **Agregāts** | **15%** | **1%** |
| payments | 174 | 174 | 174 | -/63 | -/174 | -/174 |
| payments\_range\_part | 175 | 44 | 46 | 69/78 | 44/44 | 46/46 |
| payments\_interval\_part | 170 | 44 | 45 | 66/78 | 44/44 | 45/45 |
| payments\_hash\_part | 168 | 168 | 168 | 64/78 | 168/168 | 168/168 |
| payments\_list\_part | 169 | 188 | 188 | 64/78 | 188/188 | 188/188 |
| payments\_ref\_part | 175 | 175 | 175 | - | - | - |

Lokālus indeksus nesadalītiem tabulam nav iespējams izveidot.

Agregāti: Visās tabulās, kurās nav izmantoti indeksi, COST vērtība ir lielāka salīdzinājumā ar to, ja tiek izmantoti indeksi. Tas norāda uz to, ka piekļuve datiem tiek veikta caur pilnu tabulas skenēšanu. Arī, izmantojot lokālus indeksus tika sasniegta mazāka Cost vērtība visas sadalītas tabulās.

1% un 15% vaicājumi : COST vērtība ir vienāda ar un bez indeksiem, iespējams, tas notiek, jo tabulas ir pārāk vienkāršas un nav vērts veidot sadalījumu un indeksēšanu dažādiem kolonam, arī partīticijas atrodas vienā tabultelpā.

**Secinājumi. Ko eksperimentu dati ir parādījuši un obligāti, kāpēc ir šadi rezultāti (skaidrojums).**

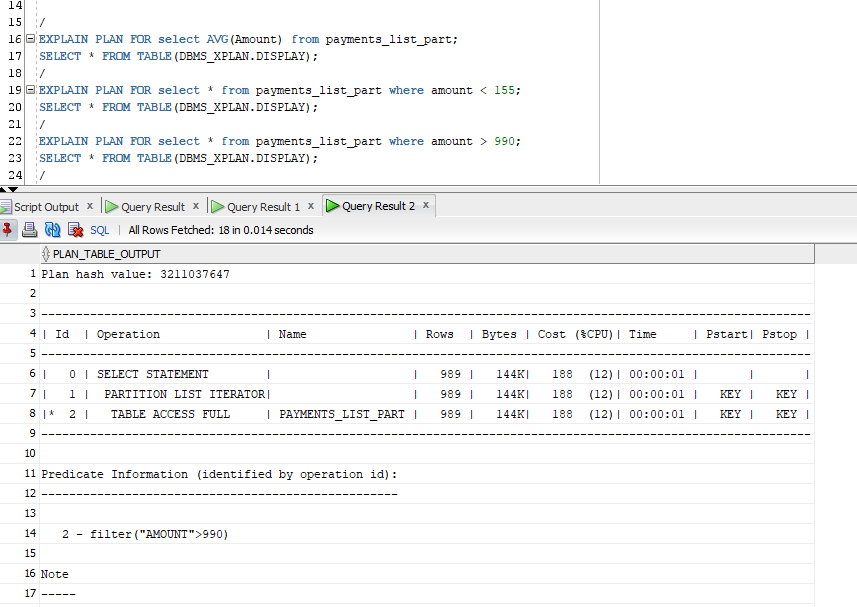
Secinājumi:

Tabulu definēšana un datu ievade: Tika izveidotās trīs jaunas tabultelpas un iegūstas to metadati. Divas tabulas tika veiksmīgi izveidotas un aizpildītas ar datiem, izmantojot PL/SQL programmu. Izveidotas tabulas tika sadalītas dažādos veidos - diapazona, intervālu, heš un saraksta sadalījumos. Šīs partīcijas ietekmēja vaicājumu izpildes veiktspēju un COST vērtības, jo dažādas partīciju stratēģijas sniedza atšķirīgus rezultātus atkarībā no izmantotā vaicājuma. Izveidojot lokālos un globālos indeksus, bija redzama pozitīva ietekme uz vaicājumu izpildes laiku, it īpaši, ja tiek izmantoti pareizi izvēlēti indeksi. Lokālie indeksi sniedza labāku veiktspēju salīdzinājumā ar globālajiem indeksiem. Indeksu izmantošana var ievērojami uzlabot vaicājumu izpildes veiktspēju, it īpaši, ja tiek izvēlēti piemēroti indeksi atbilstoši konkrētam datu modelim un vaicājumu raksturam.

Kopumā eksperimentu dati parādīja, ka pareizi izvēlēti partīciju un indeksu veidi var būtiski uzlabot datu bāzes veiktspēju un efektivitāti, ļaujot efektīvāk pārvaldīt un izmantot datu resursus.

**Pielikumā jāiekļauj minimālā formātā vaicājumu izpildes plāni.**

Izpildēs plānu iegūšanai tika izmantots zemāk noradīts PL/SQL fragments.  
  
Vajadzīgas tabulas nosaukums tika ievietots vaicājumos un trīs vaicājumi tika palaisti. Rezultāta tika izvadīti 3 plāni.



select AVG(Amount) from payments; --agregāts

select \* from payments where amount < 155; -- ~15%

select \* from payments where amount > 990; -- ~1%

/

payments

payments\_range\_part

payments\_interval\_part

payments\_hash\_part

payments\_list\_part

customer\_range\_part

payments\_ref\_part

/

EXPLAIN PLAN FOR select AVG(Amount) from payments\_list\_part;

SELECT \* FROM TABLE(DBMS\_XPLAN.DISPLAY);

/

EXPLAIN PLAN FOR select \* from payments\_list\_part where amount < 155;

SELECT \* FROM TABLE(DBMS\_XPLAN.DISPLAY);

/

EXPLAIN PLAN FOR select \* from payments\_list\_part where amount > 990;

SELECT \* FROM TABLE(DBMS\_XPLAN.DISPLAY);