Upravljanje Oracle bazama podataka – Lab #7

Tablice, indeksi, ograničenja integriteta, klusteri, Undo i transakcije

U ovom labu nastavljate razvijati vašu bazu. Tablice i podaci koje kreirate za vrijeme ovog laba će vam trebati i za sljedeće labove te je stoga bitno uspješno završiti lab.

Zadaci laba

Potrebno je kreirati jednu ili više SQL skripti koje sadrže naredbe za kreiranje tablica, klustera i indeksa prema zadanoj specifikaciji. Kreirane skripte izvedite iz SQL*Plusa izvođenjem `start create_objects.sql` naredbe. Za vrijeme kreiranja tablica, indeksa i klustera moguće je da će ponestati prostora za pohranu u jednom ili više tablespace-ova. U tom slučaju potrebno je povečati prostor kojeg tablespace alocira. Skripte pohranite na sigurnu lokaciju ako budete trebali rekreirati bazu od početka.

1. Kreiranje tablica kao korisnik.

    1) Spojite se na bazu kao Lolek i kreirajte tablicu imena VENDOR unutar njegove sheme prema sljedećim zahtjevima:

        - Kolone, tipovi podataka, karakteristike i opis su dani tabelarno.
        - Podaci tablice trebaju biti pohranjeni u DATA tablespace.
        - Postavite PCTFREE=30 i nemojte postaviti PCTUSED. PCTTREE je parametar pohrane bloka koji određuje koliko prostora će ostati prazno u bloku baze za buduće izmjene. PCTUSED se ne koristi kad je uključeno automatsko upravljanje segmentima (ASSM) što je slučaj kod DATA tablespace-a.
        - Osigurajte da CREATE TABLE naredba kreira indeks za primarni ključ tablice, te pomoću ALTER TABLE dodajte ograničenje (constraint) koji provjerava je li `Account_Balance>=0`.
        - Odredite smislena imena za sva ograničenja tablice. Indeksi kreirani kao dio specifikacije ograničenja trebaju biti pohranjeni u IDX tablespace.
        - SQL s naredbama pohranite u skriptu imena `create_table_vendor.sql` te je pokrenite kroz SQL*Plus. Ako se kod pokretanja pojavi greška „*ORA-01536: space quota exceeded for tablespace 'DATA*“ istu riješite dodavanjem prostora korisniku (ALTER USER LOLEK QUOTA 200M ON DATA;).

Tablica VENDOR			
Naziv kolone	**Tip podatka**	**Karakteristike**	**Opis**
Vendor_ID	NUMBER	Primary Key	Identifikacijski broj prodavača
Vendor_Name	VARCHAR2(30)	Not Null	Ime prodavača
Street	VARCHAR2(30)	Not Null	Adresa prodavača
City	VARCHAR2(30)	Not Null	Grad prodavača
Zip_Code	INTEGER		Poštanski kod grada
Account_Balance	NUMBER(8,2)	>= 0	Stanje računa prodavača
Comments	VARCHAR2(30)		Komentar o prodavaču

```
CREATE TABLE LOLEK.VENDOR
(
  VENDOR_ID NUMBER,
  VENDOR_NAME      VARCHAR2(30 CHAR)           NOT NULL,
  STREET           VARCHAR2(30 CHAR)           NOT NULL,
  CITY             VARCHAR2(30 CHAR)            NOT NULL,
  ZIP_CODE         INTEGER,
  ACCOUNT_BALANCE  NUMBER(8,2),
  COMMENTS         VARCHAR2(30)
)
TABLESPACE DATA
PCTFREE    30
LOGGING
NOCOMPRESS;
COMMENT ON TABLE LOLEK.VENDOR IS 'Podaci prodavača';
COMMENT ON COLUMN LOLEK.VENDOR.VENDOR_ID IS 'Identifikacijski
broj prodavača';
COMMENT ON COLUMN LOLEK.VENDOR.VENDOR_NAME IS 'Ime prodavača';
COMMENT ON COLUMN LOLEK.VENDOR.STREET IS 'Adresa prodavača';
COMMENT ON COLUMN LOLEK.VENDOR.CITY IS 'Grad prodavača';
COMMENT ON COLUMN LOLEK.VENDOR.ZIP_CODE IS 'Poštanski kod
grada';
COMMENT ON COLUMN LOLEK.VENDOR.ACCOUNT_BALANCE IS 'Stanje
računa prodavača';
COMMENT ON COLUMN LOLEK.VENDOR.COMMENTS IS 'Komentar o
prodavaču';

ALTER TABLE LOLEK.VENDOR ADD(
CONSTRAINT VENDOR_ACC_BAL CHECK (ACCOUNT_BALANCE>=0));
```

2) Pokretanjem skripte ispod napunite tablicu s 3M slučajno generiranih redaka.

```
INSERT INTO LOLEK.VENDOR
SELECT
  rownum,
  initcap(dbms_random.string('l',dbms_random.value(2,8))),
  initcap(dbms_random.string('l',dbms_random.value(2,8))),
  initcap(dbms_random.string('l',dbms_random.value(2,5))),
  dbms_random.value(10000,60000),
  round(dbms_random.value(1,100000),2),
  initcap(dbms_random.string('l',dbms_random.value(2,5)))
FROM
  (SELECT level FROM dual CONNECT BY level <= 3000000);

COMMIT;

3000000 rows created.
```

Pokrenite upit koji ispisuje sve retke tablice koji zadovoljavaju zadani uvjet

```
SQL> SELECT VENDOR_NAME FROM LOLEK.VENDOR WHERE VENDOR_ID=2934567;
VENDOR_NAME
------------------------------
Phkfmqbb
```

Kao DBA upotrijebite EXPLAIN PLAN naredbu i DBMS_XPLAN funkciju za generiranje i pregled plana izvođenja (EXECUTION PLAN) upita iz prethodnog koraka. Koja operacija pretraživanja se izvodi nad tablicom?

```
SQL> EXPLAIN PLAN FOR SELECT VENDOR_NAME FROM LOLEK.VENDOR
  2  WHERE VENDOR_ID=2934567;

Explained.

SQL> SELECT plan_table_output FROM
  2  table(DBMS_XPLAN.DISPLAY);

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------

Plan hash value: 2300692034

--------------------------------------------------------------------------------
| Id  | Operation         | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
--------------------------------------------------------------------------------
|   0 | SELECT STATEMENT  |        |   306 | 22950 |  7586   (1)| 00:01:32 |
|*  1 |  TABLE ACCESS FULL| VENDOR |   306 | 22950 |  7586   (1)| 00:01:32 |
--------------------------------------------------------------------------------

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

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------

   1 - filter("VENDOR_ID"=2934567)

Note
-----
   - dynamic sampling used for this statement (level=2)

17 rows selected.

SQL>
```

3) Kreirajte indeks nad kolonom VENDOR_ID. Za vrijeme kreiranja indeksa iz druge sesije pokrenite SELECT, UPDATE operacije nad tablicom. Koji tip scana je izveden nad tablicom? Koje DML operacije su dozvoljene nad podacima tijekom kreiranja indeksa? Poništite napravljene promjene nad podatcima.

```
--prva sesija
CREATE INDEX LOLEK.IDX_VENDOR_ID ON LOLEK.VENDOR
(VENDOR_ID)
TABLESPACE IDX;

--druga sesija
SELECT count(*) FROM LOLEK.VENDOR WHERE zip_code=23321;
UPDATE LOLEK.VENDOR SET zip_code=23322
      WHERE zip_code=23321;
ROLLBACK;
```

4) Izvođenjem upita ispod provjerite podržava li instalirana verzija ONLINE opciju za kreiranje indeksa.

```
SQL> SELECT parameter, value
```

```
  2  FROM v$option
  3  WHERE parameter='Online Index Build';

PARAMETER                VALUE
-----------------------  --------------------
Online Index Build       FALSE

SQL>
```

5) Ponovite upit iz točke 2. Prethodno ispraznite SHARED POOL memoriju kako bi svi podatci učitani s diska u memoriju bili izbrisani. Koji tip scana je sad izveden nad tablicom? Usporedite vremena izvođenja upita.

```
SQL> ALTER SYSTEM FLUSH SHARED_POOL;

System altered.

SQL> SELECT VENDOR_NAME FROM LOLEK.VENDOR WHERE VENDOR_ID=2934567;

VENDOR_NAME
------------------------------
Phkfmqbb

SQL>

SQL> EXPLAIN PLAN FOR SELECT VENDOR_NAME FROM LOLEK.VENDOR
  2  WHERE VENDOR_ID=2934567;

Explained.

SQL> SELECT plan_table_output  FROM
  2  table(DBMS_XPLAN.DISPLAY);

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------
Plan hash value: 1473930698
--------------------------------------------------------------------------------

| Id  | Operation                   | Name               | Rows  | Bytes | Cost (%CPU)| Time     |

--------------------------------------------------------------------------------
PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------

|   0 | SELECT STATEMENT            |                    |   306 | 22950 |   115   (0)| 00:00:02 |

|   1 |  TABLE ACCESS BY INDEX ROWID| VENDOR             |   306 | 22950 |   115   (0)| 00:00:02 |

|*  2 |   INDEX RANGE SCAN          | IDX_VENDOR_ZIP_CODE | 14294 |       |     3   (0)| 00:00:01 |

--------------------------------------------------------------------------------
PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------

Predicate Information (identified by operation id):

---------------------------------------------------
   2 - access("VENDOR_ID"=2934567)

Note
-----
   - dynamic sampling used for this statement (level=2)

18 rows selected.
```

6) Dodajte PK nad kolonom VENDOR_ID. Prethodno izbrišite postojeći indeks nad kolonom.

```
DROP INDEX LOLEK.IDX_VENDOR_ID;

ALTER TABLE LOLEK.VENDOR
 ADD CONSTRAINT PK_VENDOR_ID
  PRIMARY KEY
  (VENDOR_ID)
  ENABLE VALIDATE;
```

7) Kreirajte IOT (index-organized table) imena PRODUCT sa kolonama kako je navedeno u tablici ispod prema istim pravilima koje su tražena za tablicu VENDOR. Skriptu za kreiranje IOT-a pohranite u datoteku `create_table_product.sql`.

Tablica PRODUCT			
Ime kolone	**Tip podatka**	**Karakteristike**	**opis**
Product_ID	NUMBER(5)	Primary Key	Identifikacijski broj proizvoda
Description	VARCHAR2(20)	Not Null	Opis proizvoda
Retail_Price	NUMBER(8,2)	>= 0	Željena prodajna cijena
Wholesale_Price	NUMBER(8,2)	>=0	Željena veleprodajna cijena

```
CREATE TABLE PRODUCT
(
  PRODUCT_ID       NUMBER(5),
  DESCRIPTION      VARCHAR2(20 BYTE)            NOT NULL,
  RETAIL_PRICE     NUMBER(8,2),
  WHOLESALE_PRICE  NUMBER(8,2),
  CONSTRAINT PRODUCT_ID
  PRIMARY KEY (PRODUCT_ID)
)
ORGANIZATION INDEX
LOGGING
TABLESPACE DATA
PCTFREE    30
INITRANS   2
MAXTRANS   255
STORAGE    (
            INITIAL          64K
            MINEXTENTS       1
            MAXEXTENTS       UNLIMITED
            PCTINCREASE      0
            BUFFER_POOL      DEFAULT
           )
NOPARALLEL
MONITORING;

COMMENT ON TABLE VENDOR IS 'Podaci proizvoda';
COMMENT ON COLUMN PRODUCT.PRODUCT_ID IS 'Jedinstveni broj proizvoda';
COMMENT ON COLUMN PRODUCT.DESCRIPTION IS 'Opis proizvoda';
COMMENT ON COLUMN PRODUCT.RETAIL_PRICE IS 'Željena prodajna cijena';
COMMENT ON COLUMN PRODUCT.WHOLESALE_PRICE IS 'Željena veleprodajna
cijena';

ALTER TABLE PRODUCT ADD(
CONSTRAINT PRODUCT_RET_PRICE CHECK (RETAIL_PRICE>=0));
```

```
ALTER TABLE PRODUCT ADD(
CONSTRAINT PRODUCT_WHOLESALE_PRICE CHECK (WHOLESALE_PRICE>=0));
```

8) Kreirajte indeksirani klaster imena INVOICE_AND_DETAILS koji će grupirati retke iz tablica INVOICE i INVOICE_DETAILS. Parametri klastera su sljedeći:

   - Postavite SIZE=150. SIZE određuje broj kluster ključeva i redaka koji mogu stati u klastrirani blok podataka (clustered data block).
   - Postavite PCTFREE=15 za klaster.
   - Pohranite kluster u DATA tablespace.

   Skripte za kreiranje objekata pohranite u datoteku `create_cluster_invoice_details.sql`.

```
CREATE CLUSTER INVOICE_AND_DETAILS
(INVOICE_ID       NUMBER(3))
PCTFREE    15
SIZE       150
TABLESPACE DATA;
```

   Kreirajte klaster indeks u IDX tablespace-u.

```
CREATE INDEX INVOICE_AND_DETAILS_IDX
ON CLUSTER INVOICE_AND_DETAILS
TABLESPACE IDX;
```

   Kreirajte INVOICE i INVOICE_DETAILS tablice unutar INVOICE_AND_DETAILS klastera. CREATE TABLE naredba treba kreirati primarni ključ za INVOICE i INVOICE_DETAILS tablice. Tablice moraju biti pohranjene u kluster te nametnut referencijalni integritet za sljedeće veze:

   - 1:V relacija između prodavatelja i faktura. Ako je izbrisan redak u tablici VENDOR, kaskadno se brišu retci u INVOICE tablici. Redak u INVOICE tablicu ne može biti umetnut bez ispravne asocijacije sa retkom u VENDOR tablici.
   - 1:V relacija između proizvoda i detalja fakture. Nije dozvoljeno brisanje retka iz PRODUCT tablice ako postoje povezani retci u INVOICE_DETAILS tablici (zabranjeno kaskadno brisanje). Redak u INVOICE_DETAILS tablici ne može biti umetnut ako ne postoji ispravna asocijacija sa retkom u PRODUCT tablici.
   - 1:V relacija između fakture i detalja fakture. Ako je izbrisan redak u INVOICE tablici, kaskadno se brišu retci u INVOICE_DETAILS tablici. Redak u INVOICE_DETAILS tablicu ne može biti umetnut bez ispravne asocijacije sa retkom u INVOICE tablici.
   - Nametnutim ograničenjima dodjelite asocijativna imena i osigurajte da indeksi budu pohranjeni u IDX tablespace-u.

Tablica INVOICE			
Ime kolone	**Tip podataka**	**Karakteristike**	**Opis**
Invoice_ID	NUMBER(3)	Primary Key	Identifikacijski broj fakture

Invoice_Date	DATE	Not Null	Datum fakture
Invoice_Amount	NUMBER(10,2)	>= 0	Ukupni iznos fakture
Promise_Date	DATE	Podrazumijevano TODAY + 14 od danas.	Obečani datum isporuke fakture
Vendor_ID	NUMBER(3)	Not Null	Identifikacijski broj prodavatelja (FK)

Tablica INVOICE_DETAILS

Ime kolone	Tip podatka	Karakteristike	Opis
Invoice_ID	NUMBER(3)	Primary Key	Identifikacijski broj fakture
Product_ID	NUMBER(5)	Primary Key	Identifikacijski broj proizvoda
Quantity_Ordered	NUMBER(8,2)	>= 0	Naručena količina
Actual_Price	NUMBER(8,2)	>= 0	Prodajna cijena proizvoda

```
CREATE TABLE INVOICE
(
  INVOICE_ID       NUMBER(3)
  CONSTRAINT INVOICE_ID_PK PRIMARY KEY
  USING INDEX TABLESPACE IDX,
  INVOICE_DATE     DATE                        NOT NULL,
  INVOICE_AMOUNT   NUMBER(10,2),
  PROMISE_DATE     DATE                        DEFAULT SYSDATE + 14,
  VENDOR_ID        NUMBER(5)                   NOT NULL
)
CLUSTER INVOICE_AND_DETAILS(INVOICE_ID);

COMMENT ON TABLE INVOICE IS 'Podaci računa';
COMMENT ON COLUMN INVOICE.INVOICE_ID IS 'Identifikacijski broj fakture';
COMMENT ON COLUMN INVOICE.INVOICE_DATE IS 'Datum fakture';
COMMENT ON COLUMN INVOICE.INVOICE_AMOUNT IS 'Ukupni iznos fakture';
COMMENT ON COLUMN INVOICE.PROMISE_DATE IS 'Obečani datum isporuke fakture';
COMMENT ON COLUMN INVOICE.VENDOR_ID IS 'Identifikacijski broj prodavatelja';

ALTER TABLE INVOICE ADD (
  CONSTRAINT INVOICE_AMOUNT CHECK (INVOICE_AMOUNT>=0));

ALTER TABLE INVOICE ADD (
  CONSTRAINT INV_VENDOR_FK
  FOREIGN KEY (VENDOR_ID)
  REFERENCES VENDOR (VENDOR_ID));

CREATE TABLE INVOICE_DETAILS
(
  INVOICE_ID        NUMBER(3),
  PRODUCT_ID        NUMBER(5),
  QUANTITY_ORDERED  NUMBER(8,2),
  ACTUAL_PRICE      NUMBER(8,2)
)
CLUSTER INVOICE_AND_DETAILS(INVOICE_ID);

COMMENT ON TABLE INVOICE_DETAILS IS 'Detalji fakture';
COMMENT ON COLUMN INVOICE_DETAILS.INVOICE_ID IS 'Identifikacijski broj
fakture';
COMMENT ON COLUMN INVOICE_DETAILS.PRODUCT_ID IS 'Identifikacijski broj
proizvoda';
COMMENT ON COLUMN INVOICE_DETAILS.QUANTITY_ORDERED IS 'Naručena količina';
COMMENT ON COLUMN INVOICE_DETAILS.ACTUAL_PRICE IS 'Prodajna cijena
proizvoda';

ALTER TABLE INVOICE_DETAILS ADD (
```

```
  CONSTRAINT INV_DETAIL_QUANTITY CHECK (QUANTITY_ORDERED>=0));

ALTER TABLE INVOICE_DETAILS ADD (
  CONSTRAINT INV_DETAIL_ACT_PRICE CHECK (ACTUAL_PRICE>=0));

ALTER TABLE INVOICE_DETAILS ADD (
  CONSTRAINT INV_DETAIL_FK
  FOREIGN KEY (INVOICE_ID)
  REFERENCES INVOICE (INVOICE_ID));

ALTER TABLE INVOICE_DETAILS ADD (
  CONSTRAINT INV_DETAIL_PRODUCT_FK
  FOREIGN KEY (PRODUCT_ID)
  REFERENCES PRODUCT (PRODUCT_ID));
```

2. Unos testnih podataka za prodavatelja i proizvod i testiranje nametnutih ograničenja.

   1) Unesite nekoliko retka u tablicu prodavatelja.

```
INSERT INTO VENDOR (
 VENDOR_ID, VENDOR_NAME, STREET, CITY, ZIP_CODE, ACCOUNT_BALANCE, COMMENTS)
 VALUES (1, 'Tester 1', 'Sinjska', 'Split', '21000', 1000.50, 'Testni
prodavatelj 1');
```

      Zašto gornji kod podiže pogrešku? Pozivanjem MAX funkcije pronađite najveći VENDOR_ID.

```
SELECT MAX(VENDOR_ID) FROM VENDOR;
INSERT INTO VENDOR (
 VENDOR_ID, VENDOR_NAME, STREET, CITY, ZIP_CODE, ACCOUNT_BALANCE, COMMENTS)
 VALUES (3000001, 'Tester 1', 'Sinjska', 'Split', '21000', 1000.50, 'Testni
prodavatelj 1');
COMMIT;
```

   2) Unesite nekoliko redaka u tablicu proizvoda.

```
INSERT INTO PRODUCT
   (PRODUCT_ID, DESCRIPTION, RETAIL_PRICE, WHOLESALE_PRICE)
 VALUES
   (1, 'Proizvod 1', 100.10, 90.00);
```

   3) Pokušajte unijeti:

      - jedan redak u tablicu prodavatelja koja narušava nametnuto ograničenje za ACCOUNT_BALANCE. Zapišite poruku greške koja je podignuta!
      - jedan redak u tablicu prodavatelja koja narušava integritet primarnog ključa. Zapišite poruku greške koja je podignuta!
      - redak u tablicu prodavatelja koji narušava NOT NULL ograničenje. Zapišite poruku greške koja je podignuta!
      - jedan redak u tablicu proizvoda koji narušava nametnuto ograničenje za RETAIL_PRICE.
      - jedan redak u tablicu proizvoda koji narušava nametnuto ograničenje za WHOLESALE_PRICE.
      - jedan redak u tablicu proizvoda koji narušava integritet primarnog ključa.

   4) Unesite dva retka ispravnih faktura sa dva retka u detaljima faktura za svaku kreiranu fakturu.

```
INSERT INTO INVOICE
   (INVOICE_ID, INVOICE_DATE, INVOICE_AMOUNT, PROMISE_DATE, VENDOR_ID)
 VALUES
   (1, TO_DATE('04/10/2012 00:00:00', 'MM/DD/YYYY HH24:MI:SS'), 200.00,
   TO_DATE('04/24/2012 00:00:00', 'MM/DD/YYYY HH24:MI:SS'), 1);

INSERT INTO INVOICE_DETAILS
   (INVOICE_ID, PRODUCT_ID, QUANTITY_ORDERED, ACTUAL_PRICE)
 Values
   (1, 1, 1, 100.00);
COMMIT;
```

3. Pregled kreiranih segmenata.

   1) Ispišite sve segmente koje ste kreirali u ovom lab-u.

```
SQL> COLUMN segment_name FORMAT A30;
SQL> SELECT owner,segment_name,segment_type FROM dba_segments
  2  WHERE owner='LOLEK'
  3  ORDER BY segment_type ASC;

OWNER                          SEGMENT_NAME                   SEGMENT_TYPE
------------------------------ ------------------------------ ---------------
LOLEK                          INVOICE_AND_DETAILS            CLUSTER
LOLEK                          PK_VENDOR_ID                   INDEX
LOLEK                          PRODUCT_ID                     INDEX
LOLEK                          INVOICE_AND_DETAILS_IDX        INDEX
LOLEK                          INVOICE_ID_PK                  INDEX
LOLEK                          VENDOR                         TABLE

6 rows selected.

SQL>
```

   2) Ispišite početni blok ekstenta i veličinu ekstenta u blokovima za kluster objekt.

```
SQL> COLUMN owner FORMAT A6;
SQL> COLUMN segment_name FORMAT A20;
SQL> COLUMN segment_type FORMAT A7;
SQL> SELECT owner,segment_name,segment_type,extent_id,file_id,block_id,blocks
  2  FROM dba_extents
  3  WHERE owner='LOLEK' AND segment_name='INVOICE_AND_DETAILS';

OWNER  SEGMENT_NAME         SEGMENT  EXTENT_ID    FILE_ID   BLOCK_ID BLOCKS
------ -------------------- ------- ---------- ---------- ---------- -------
LOLEK  INVOICE_AND_DETAILS  CLUSTER          0          6      14098       5

SQL>
```

   3) Pronađite u kojem bloku se nalazi jedan redak INVOICE tablice.

```
SQL> SELECT
  2    dbms_rowid.rowid_relative_fno(rowid) REL_FNO,
  3    dbms_rowid.rowid_block_number(rowid) BLOCKNO,
  4    dbms_rowid.rowid_row_number(rowid) ROWNO,
  5    invoice_id,invoice_date,invoice_amount
  6    FROM lolek.invoice WHERE invoice_id = 1;

   REL_FNO    BLOCKNO      ROWNO INVOICE_ID INVOICE_ INVOICE_AMOUNT
---------- ---------- ---------- ---------- -------- --------------
         6      14102          0          1 10.04.12            200
SQL>
```

4. UNDO segmenti i transakcije.

   1) Spojite se kao DBA te pokrenite upit ispod.Upit čita sadržaj UNDO tablespace-a i zbraja po tipu ekstenta. Prikazuje se ekstenti u stanju ACTIVE, EXPIRED i UNEXPIRED dok oni FREE nisu prikazani.

```
SQL> SELECT status,
  2    round(sum_bytes / (1024*1024), 0) as MB,
  3    round((sum_bytes / undo_size) * 100, 0) as PERC
  4  FROM
  5  (
  6    SELECT status, sum(bytes) sum_bytes
  7    FROM dba_undo_extents
  8    GROUP BY status
  9  ),
 10  (
 11    SELECT sum(a.bytes) undo_size
 12    FROM dba_tablespaces c
 13      JOIN v$tablespace b ON b.name = c.tablespace_name
 14      JOIN v$datafile a ON a.ts# = b.ts#
 15    WHERE c.contents = 'UNDO'
 16      AND c.status = 'ONLINE'
 17  );
SQL>

STATUS            MB       PERC
--------- ---------- ----------
EXPIRED          386         57
UNEXPIRED         34          5
```

   Koliko ekstenata u MB je u stanju FREE?

   2) Ispišite aktivne transakcije.

```
SQL> SELECT a.sid, a.username, b.start_time, b.status,b.xidusn, b.used_urec,
  2  b.used_ublk FROM v$session a, v$transaction b
  3  WHERE a.saddr = b.ses_addr;

no rows selected
SQL>
```

   3) Spojite se na bazu kao korisnik LOLEK te napravite UPDATE bez komitiranja transakcije.

```
SQL> SELECT count(*) FROM vendor WHERE zip_code > 20000 AND zip_code < 22000;

  COUNT(*)
----------
    119695

SQL> UPDATE vendor SET zip_code = 21000
  2  WHERE zip_code > 20000 AND zip_code < 22000;

119695 rows updated.

SQL>
```

   4) Iz DBA sesije ponovite korake pod 1 i 2 i usporedite dobivene vrijednosti.

```
STATUS            MB       PERC
--------- ---------- ----------
ACTIVE            16          2
EXPIRED          378         56
UNEXPIRED         26          4
```

```
SQL> SELECT to_char(begin_time,'MM/DD/YY HH24:MI') begin,
  2  to_char(end_time,'MM/DD/YY HH24:MI') end,undoblks  FROM v$undostat;

BEGIN          END            UNDOBLKS
-------------- -------------- ----------
04/22/16 09:21 04/22/16 09:31       1304
04/22/16 09:11 04/22/16 09:21          2

SQL> SELECT a.sid, a.username, b.start_time, b.status,b.xidusn, b.used_urec,
  2  b.used_ublk FROM v$session a, v$transaction b
  3  WHERE a.saddr = b.ses_addr;

  SID USERNAME   START_TIME        STATUS  XIDUSN  USED_UREC  USED_UBLK
----- ---------- ----------------- ------- ------- ---------- ----------
  195 LOLEK      04/22/16 09:27:12 ACTIVE        8     119686       1304
```

5) Iz sesije LOLEK korisnika potvrdite transakciju.

```
SQL> COMMIT;

Commit complete.

SQL>
```

6) Iz DBA sesije provjerite aktivne transakcije i je li postoje ekstenti u statusu ACTIVE.

```
SQL> SELECT a.sid, a.username, b.start_time, b.status,b.xidusn, b.used_urec,
  2    b.used_ublk FROM v$session a, v$transaction b
  3    WHERE a.saddr = b.ses_addr;

no rows selected

SQL>
```

7) Kao LOLEK korisnik pokrenite upit ispod. Transakciju nemojte komitirati!

```
SQL> UPDATE VENDOR SET  ZIP_CODE = 41000 WHERE  ZIP_CODE >50000;

600448 rows updated.
```

Iz DBA sesije provjerite aktivne transakcije i ekstente.

```
SQL> SELECT to_char(begin_time,'MM/DD/YY HH24:MI') begin,
  2  to_char(end_time,'MM/DD/YY HH24:MI') end,undoblks  FROM v$undostat;

BEGIN          END            UNDOBLKS
-------------- -------------- ----------
04/22/16 10:41 04/22/16 10:51      12540
04/22/16 10:31 04/22/16 10:41          0
04/22/16 10:21 04/22/16 10:31          4
04/22/16 10:11 04/22/16 10:21          0
04/22/16 10:01 04/22/16 10:11          2
04/22/16 09:51 04/22/16 10:01         16
04/22/16 09:41 04/22/16 09:51          0
04/22/16 09:31 04/22/16 09:41          0
04/22/16 09:21 04/22/16 09:31       1307
04/22/16 09:11 04/22/16 09:21          2

10 rows selected.

SQL>  SELECT a.sid, a.username, b.start_time, b.status,b.xidusn, b.used_urec,
  2   b.used_ublk FROM v$session a, v$transaction b
  3   WHERE a.saddr = b.ses_addr;
```

```
  SID USERNAME  START_TIME         STATUS   XIDUSN USED_UREC  USED_UBLK
----- --------- ------------------ -------- ------ ---------- ----------
  195 LOLEK     04/22/16 10:42:17  ACTIVE        6    1200850      11600
```

```
SQL> select status,
  2    round(sum_bytes / (1024*1024), 0) as MB,
  3    round((sum_bytes / undo_size) * 100, 0) as PERC
  4  from
  5  (
  6    select status, sum(bytes) sum_bytes
  7    from dba_undo_extents
  8    group by status
  9  ),
 10  (
 11    select sum(a.bytes) undo_size
 12    from dba_tablespaces c
 13      join v$tablespace b on b.name = c.tablespace_name
 14      join v$datafile a on a.ts# = b.ts#
 15    where c.contents = 'UNDO'
 16      and c.status = 'ONLINE'
 17  );

STATUS            MB       PERC
--------- ---------- ----------
ACTIVE           104         15
EXPIRED           84         13
UNEXPIRED         30          4
```

Iz sesije LOLEK korisnika napravite ROLLBACK transakcije te nakon toga usporedite ACTIVE i UNEXPIRED ekstente.

```
SQL> select status,
  2    round(sum_bytes / (1024*1024), 0) as MB,
  3    round((sum_bytes / undo_size) * 100, 0) as PERC
  4  from
  5  (
  6    select status, sum(bytes) sum_bytes
  7    from dba_undo_extents
  8    group by status
  9  ),
 10  (
 11    select sum(a.bytes) undo_size
 12    from dba_tablespaces c
 13      join v$tablespace b on b.name = c.tablespace_name
 14      join v$datafile a on a.ts# = b.ts#
 15    where c.contents = 'UNDO'
 16      and c.status = 'ONLINE'
 17  );

STATUS            MB       PERC
--------- ---------- ----------
EXPIRED           84         13
UNEXPIRED        134         20
```

5. Kreiranje kopije tablice.

   1) Spojite se kao BOLEK korisnik i kreirajte kopiju tablice LOLEK.PRODUCT. Eventualne greške otklonite dodijeljivanjem potrebnih privilegija BOLEK-u.

```
SQL> CREATE TABLE product_copy AS SELECT * FROM lolek.product;
CREATE TABLE product_copy AS SELECT * FROM lolek.product
                                           *
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> CREATE TABLE product_copy AS SELECT * FROM lolek.product;
```

```
CREATE TABLE product_copy AS SELECT * FROM lolek.product
                                           *
ERROR at line 1:
ORA-01031: insufficient privileges

SQL> CREATE TABLE product_copy AS SELECT * FROM lolek.product;
CREATE TABLE product_copy AS SELECT * FROM lolek.product
                                           *
ERROR at line 1:
ORA-01536: space quota exceeded for tablespace 'DATA'

SQL> CREATE TABLE product_copy AS SELECT * FROM lolek.product;

Table created.

SQL>
```

6. Izmjena strukture tablice.

   1) Spojite se kao korisnik LOLEK te dodajte tablici PRODUCT kolonu SALE_PRICE tipa NUMBER(8,2).

```
SQL> ALTER TABLE PRODUCT
  2   ADD (SALE_PRICE  NUMBER(8,2));

Table altered.
```

   2) Pokrenite upit koji ažurira sve retke u tablici tako da za dodanu kolonu pohranjuje vrijednost polja WHOLESALE_PRICE uvečanu za 20%;

```
SQL> UPDATE PRODUCT SET SALE_PRICE=1.2*WHOLESALE_PRICE;

1 row updated.

SQL> COMMIT;

Commit complete.
```