

SQL*Plus

Felhasználók:

- SYS: rendszergazda
- SCOTT: demonstrációs adatbázis, táblái: EMP (dolgozó), DEPT (osztály)
- "közönséges" felhasználók

Adatszótár:

- metaadatokat tartalmazó, csak olvasható táblák
- táblanév-prefixek:
 - o ALL: az adott felhasználó által elérhető összes objektum;
 - o USER: a felhasználó sémájában lévő objektumok;

Felhasználók listája: SELECT * FROM all_users;

A felhasználó tábláinak nevei: SELECT table_name FROM user_tables;

DUAL segéd tábla: egy bejegyzést tartalmazó teszt tábla

A tábla kiíratása: SELECT * FROM DUAL;

Teszt: SELECT 'a' || 'b' FROM DUAL;

Teszt 2: SELECT sysdate FROM DUAL;

SQL*Plus kezelői felület:

- háromféle parancsot adhatunk ki
 - o SQL parancsok (adatbázis műveletek)
 - o PL/SQL blokkok (adatbázis műveletek)
 - o SQL*Plus parancsok (opciók beállítása, az előző két parancstípus szerkesztése és tárolása, lekérdezések formázása)

SQL parancsbuffer: az utoljára begépelte utasítást tartalmazza (az SQL*Plus parancsok nem kerülnek bele)

- Enter: sortörés, de nem a parancs végrehajtása
- SQL-parancs lezárása:
 - o pontosvessző (;) vagy törtvonal (\): ezek nem kerülnek a bufferbe
 - o üres sor: ekkor nem hajtódik azonnal végre a parancs, de a bufferbe bekerül.

SQL*Plus parancsok után nem kell ;

SQL*Plus parancsok:

CLEAR SCREEN : képernyő törlése
CLEAR BUFFER : buffer törlése
EXIT : kilépés az SQL*Plusból
HELP parancsnev : egy SQL*Plus parancsról kérhetünk segítséget
HELP INDEX : felsorolja a parancsokat
SHOW ALL : SQL*Plus paraméterek listája
SHOW parameter : adott paraméter értéke
SET parameter érték : adott paraméter beállítása
DESCRIBE tablanév : adott tábla szerkezetének kiírása

- általában a parancsszavak rövidíthetők

Példák:

SHOW LINESIZE - sorméret
SHOW PAGESIZE - lapméret
SHOW NEWPAGE - két lap közötti üres sorok
SET LINESIZE 300
DESCRIBE dual
DESCRIBE emp

Manipuláló és futtató parancsok:

Listázzuk ki a DEMO felhasználóhoz tartozó táblákat:

```
SQL> SELECT owner, table_name
      2 FROM all_tables
      3 WHERE owner LIKE 'DEMO';
```

SAVE filename : buffer tartalmának mentése
SAVE filename REPLACE : létező file felülírása
EDIT filename : file megnyitása szerkesztésre
GET filename : file betöltése a bufferbe

Példa:

SAVE elso - létrehozza az elso.sql file-t a buffer tartalmával
EDIT elso - megnyitjuk szerkesztésre a file-t:

Javítsuk ki a DEMO nevet saját azonosítónkra, majd mentjük a file-t.

Töltsük be a file-t a bufferbe: GET elso
Futtassuk az utolsó utasítást: RUN
Listázzuk az utolsó utasítás 3. sorát: LIST 3

Cseréljük le a saját azonosítónkat a DEMO userre:
CHANGE /PBALAZS/DEMO

(a CHANGE /text alak az adott text szöveget törli a sorból)

Futtassuk az utasítást: RUN
Az utolsó sor végéhez fűzzük hozzá az or owner LIKE 'PBALAZS'
szöveget: APPEND or owner like 'PBALAZS'
Futtassunk: RUN
Töröljük a 3. sort:
LIST 3
DEL

Adjunk hozzá új sort a végére (jelenleg ez az aktuális sor):
INPUT WHERE owner like 'DEMO'

Mindig az aktuális sor után szűrődik be az új sor.
Beszúrás a legelejére: 0 (nulla) text

RUN csak a puffer tartalmát hajta végre (tehát egy utasítást).
Több utasítást (köztük akár SQL*Plus utasításokat is) tartalmazó
file-okat a START filename vagy @filename paranccsal
futtathatunk.

Nyelvi elemek:

Megjegyzések:

REMARK szoveg (PL/SQL blokkban nem használható)
-- szoveg (tetszőleges nem SQL*Plus utasítás után)
/* */ (többsoros megjegyzés)

Fontosabb adattípusok:

NUMBER(hossz,tizedes)
INTEGER - NUMBER-rel egyenértékű
CHAR(n) - n fix hosszú (alapból n=1) karaktersorozat
VARCHAR2(n) - változó, de max. n hosszúságú karaktersorozat
LONG -változó hosszúságú karaktersorozat
BINARY_FLOAT
BINARY_DOUBLE
DATE - dátum
TIMESTAMP - idő

Változók:

- rendszerváltozók (SQL*Plus paraméterek)
- felhasználói vagy helyettesítő változók (input)
- hozzárendelt változók (output PL/SQL blokkból)

DEFINE : definiált felhasználói változók listázása
DEFINE változo = ertek : változó definiálás
UNDEFINE változo : változó törlése
ACCEPT változo PROMPT 'szoveg' : változó értékének bekérése a szoveg megjelenítésével

Definiáljunk egy változót: DEFINE nevem = 'Balazs Peter' (alapból mindenkeppen string lesz, az ACCEPT paranccsal explicite is meg lehet adni a típust)

```
ACCEPT nev PROMPT 'mi a nev:'  
ACCEPT változo NUMBER PROMPT
```

Bekérés & paranccsal (a változó nem definiálódik):

```
SELECT owner,table_name FROM all_tables WHERE owner LIKE &name;  
Adja meg a(z) name értékét: 'DEMO'  
régi 1: select owner,table_name from all_tables where owner  
like &name  
új 1: select owner,table_name from all_tables where owner like  
'DEMO'
```

Bekérés && paranccsal (a változó definiálódik):

```
SELECT owner,table_name FROM all_tables WHERE owner LIKE &&name;  
Adja meg a(z) name értékét: 'DEMO'
```

Paraméter átadás a START paranccsal:

Ha a MYFILE tartalma az alábbi:

```
SELECT * FROM emp WHERE job='&1' AND sal='&2';
```

akkor a START MYFILE param1 param2 parancsot használjuk

Deklaráljunk egy hozzárendelt változót, majd kérdezzük le a deklarált változókat:

```
VARIABLE BP CHAR  
VARIABLE
```

Hozzárendelt változó értékének kiíratása: PRINT változonev

Feladat:

Hozzunk létre egy konyvek nevű file-t, amely futtatás után létrehoz egy táblát konyv néven és benne két adatot, majd kiírja azokat.

```
CREATE TABLE konyv
(
  cim VARCHAR2(60),
  szerzo VARCHAR2(60),
  isbn NUMBER(10)
);
```

```
INSERT INTO konyv VALUES
(
  'Elso konyv', 'PETIKE', 1234
);
```

```
INSERT INTO konyv VALUES
(
  'Masodik konyv', 'JANCSIKA', 5678
);
```

```
SELECT * FROM konyv;
```

Adjunk jogot a szomszédunknak a táblánk lekérdezésére:

```
GRANT SELECT ON konyv TO SZOMSZED;
```

Kérdezzük le a szomszéd tábláját:

```
SELECT * FROM SZOMSZED.konyv;
```

Vonjuk meg a jogot:

```
REVOKE SELECT ON konyv FROM SZOMSZED;
```