Az ORACLE demo adatbázisa

Állítsa be a dátumok formátumát magyarra az `alter session set nls_date_format = 'YYYY.MM.DD';` paranccsal!

a) Listázza ki az összes olyan dolgozó vezetéknevét, fizetését és beosztásának azonosítóját, akinek a fizetése megegyezik a cégnél lévő minimális fizetéssel!

```
SELECT last_name, job_id, salary
FROM employees
WHERE salary = (SELECT min(salary) FROM employees);
```

```
1 SELECT last_name, job_id, salary
2 FROM employees
3 WHERE salary = (SELECT min(salary) FROM employees);
```

b) Az előző lekérdezést felhasználva készítse el azoknak a dolgozóknak a fizetés szerint csökkenő listáját (az utónevet és a fizetést megjelenítve), akik a legkevesebbet kereső dolgozóval azonos beosztásban dolgoznak!

```
SELECT last_name, salary
FROM employees
WHERE job_id = (
    SELECT job_id
    FROM employees
    WHERE salary = (SELECT min(salary) FROM employees)
)
ORDER BY 2 DESC;
```

```
1 SELECT last_name, salary
2 FROM employees
3 WHERE job_id = (
4     SELECT job_id
5     FROM employees
6     WHERE salary = (SELECT min(salary) FROM employees)
7 )
8 ORDER BY 2 DESC;
```

c) Listázza ki az ország neve szerint növekvő, azon belül a város neve szerint csökkenő ábécérendben az összes részleg nevét, a telephely országát, államát/tartományát és városát! Ha egy részleg telephelyéhez nem tartozik állam/tartomány, akkor ott a *-nincs-* szöveg jelenjen meg!

```
SELECT country_name, NVL(state_province, '-nincs-'),
       city, department_name
FROM (locations INNER JOIN countries USING(country_id))
    INNER JOIN departments USING(location_id)
ORDER BY 1, 3;
```

```
1  SELECT country_name, NVL(state_province, '-nincs-'),
2         city, department_name
3  FROM (locations INNER JOIN countries USING(country_id))
4      INNER JOIN departments USING(location_id)
5  ORDER BY 1, 3;
```

d) Készítsen lekérdezést, amely az egyes beosztásokhoz tartozó fizetési sávokat szemlélteti az ábrának megfelelően, 1000 dolláronként! Az alsó fizetési határig pont, utána a felső fizetési határig # jelzi a nagyságrendet. A lista a felső fizetési határ szerint csökkenő sorrendben jelenjen meg!

```
SELECT RPAD(job_id, 10) ||
    RPAD('.',ROUND(min_salary/1000)-1,'.') ||
    RPAD('#',ROUND(max_salary/1000)-ROUND(min_salary/1000),'#')
FROM jobs
ORDER BY max_salary DESC;
```

```
1  SELECT RPAD(job_id, 10) ||
2      RPAD('.',ROUND(min_salary/1000)-1,'.') ||
3      RPAD('#',ROUND(max_salary/1000)-ROUND(min_salary/1000),'#')
4  FROM jobs
5  ORDER BY max_salary DESC;
```

e) Listázza ki a *departments* táblában szereplő részlegek nevét, valamint az adott részlegről előléptetett dolgozók azonosítóját és a korábbi beosztás végdátumát! Azok a részlegek is jelenjenek meg, amelyekhez egyetlen előléptetett dolgozó sem tartozik! Rendezze a listát az részleg, azon belül a dolgozó azonosítója szerint!

```
SELECT department_name, employee_id, end_date
FROM job_history RIGHT JOIN departments USING(department_id)
ORDER BY 1, 2;
```

```
1  SELECT department_name, employee_id, end_date
2  FROM job_history RIGHT JOIN departments USING(department_id)
3  ORDER BY 1, 2;
```

A PL/SQL nyelv elemei

1. feladat: Nyissa meg a *View* menüt, majd a *Dbms Output*-ot! A megnyíló ablakban állítsa be a *connection*-t! Ha szükséges, akkor futtassa le a

```
set serveroutput on;
```

parancsot – ekkor az output nemcsak a *Dbms Output* ablakban, hanem a Script Output ablakban is látszani fog.

2. feladat: Írja ki a kijelzőre (dbms_output), hogy „*Helló, világ!*”!

```
BEGIN
  dbms_output.put_line('Helló, világ');
END;
```

3. feladat: Szükség esetén futtassa le az *adatb_ii_gyak02_create__demo.sql* állományt a demó adatbázis betöltéséhez! Ezután készítsen egy *utasításblokkot*, amely felhasználói bemenetként megkérdezi valamelyik részleg nevét (*department_name*), válaszul a *Dbms Outputra* küldi az osztály dolgozóinak számát és átlagfizetését! Mivel a *salary* mező NUMBER (8,2) beállítású, az átlagfizetést is két tizedesre kerekítve jelenítse meg!

```
ACCEPT reszleg PROMPT 'A részleg neve: ';
DECLARE
  átlagfizetés employees.salary%TYPE;
  létszám      NUMBER;
BEGIN
  SELECT COUNT(salary), ROUND(AVG(salary), 2)
    INTO létszám, átlagfizetés
    FROM departments INNER JOIN employees USING(department_id)
    WHERE UPPER(department_name) = UPPER('&reszleg');
  DBMS_OUTPUT.PUT_LINE(INITCAP('&reszleg') || ' részleg létszáma: ' ||
              létszám || ' fő; összfizetése: ' || átlagfizetés);
END;
```

```
1  ACCEPT reszleg PROMPT 'A részleg neve: ';
2  DECLARE
3    átlagfizetés employees.salary%TYPE;
4    létszám      NUMBER;
5  BEGIN
6    SELECT COUNT(salary), ROUND(AVG(salary), 2)
7      INTO létszám, átlagfizetés
8      FROM departments INNER JOIN employees USING(department_id)
9      WHERE UPPER(department_name) = UPPER('&reszleg');
10   DBMS_OUTPUT.PUT_LINE(INITCAP('&reszleg') || ' részleg létszáma: ' ||
11               létszám || ' fő; összfizetése: ' || átlagfizetés);
12 END;
```

4. feladat: Készítsen egy *utasításblokkot*, amely bemenő felhasználói bemenetként megkérdezi valamelyik részleg nevét! Az utasításblokk *formázottan* jelenítse meg a részleg azon dolgozóinak *vezetéknevét*, *beosztás-azonosítóját* és *fizetését* (név szerint rendezve), akik ezen részleg átlaga feletti fizetéssel rendelkeznek! A megoldáshoz használjon *kurzort*!

```
ACCEPT reszleg PROMPT 'A részleg neve: ';
DECLARE
  CURSOR emp_cur IS
    SELECT last_name, job_id, salary
    FROM employees INNER JOIN departments USING(department_id)
    WHERE UPPER(department_name) = UPPER('&reszleg') AND salary > (
      SELECT AVG(salary)
      FROM employees INNER JOIN departments USING(department_id)
      WHERE UPPER(department_name) = UPPER('&reszleg')
    )
    ORDER BY last_name;
  emp_rec emp_cur%ROWTYPE;
BEGIN
  FOR emp_rec IN emp_cur LOOP
    DBMS_OUTPUT.PUT_LINE(RPAD(INITCAP(emp_rec.last_name), 15) ||
                         RPAD(emp_rec.job_id, 12) ||
                         LPAD(emp_rec.salary, 8));
  END LOOP;
END;
```

```
1  ACCEPT reszleg PROMPT 'A részleg neve: ';
2  DECLARE
3    CURSOR emp_cur IS
4      SELECT last_name, job_id, salary
5      FROM employees INNER JOIN departments USING(department_id)
6      WHERE UPPER(department_name) = UPPER('&reszleg') AND salary > (
7        SELECT AVG(salary)
8        FROM employees INNER JOIN departments USING(department_id)
9        WHERE UPPER(department_name) = UPPER('&reszleg')
10     )
11     ORDER BY last_name;
12   emp_rec emp_cur%ROWTYPE;
13 BEGIN
14   FOR emp_rec IN emp_cur LOOP
15     DBMS_OUTPUT.PUT_LINE(RPAD(INITCAP(emp_rec.last_name), 15) ||
16                          RPAD(emp_rec.job_id, 12) ||
17                          LPAD(emp_rec.salary, 8));
18   END LOOP;
19 END;
```

5. feladat:

a) Készítsen *utasítás-sorozatot* (nem blokkot), amely
- létrehoz egy *nézettáblát* az összes telephely (*locations*) *azonosítójáról*, *közelebbi címéről*, *városáról* és *országáról* (a nézettábla neve legyen *telephelyLista*);
- egy *SELECT* paranccsal megjeleníti a nézet teljes tartalmát *ország*, *város*, azon belül a *közelebbi cím* szerint rendezve;
- végül eldobja a nézettáblát!

```
CREATE OR REPLACE VIEW telephelyLista AS
  SELECT location_id, street_address, city, country_name
  FROM locations INNER JOIN countries USING(country_id);
SELECT * FROM telephelyLista
  ORDER BY country_name, city, street_address;
DROP VIEW telephelyLista;
```

```
1 CREATE OR REPLACE VIEW telephelyLista AS
2   SELECT location_id, street_address, city, country_name
3   FROM locations INNER JOIN countries USING(country_id);
4 SELECT * FROM telephelyLista
5   ORDER BY country_name, city, street_address;
6 DROP VIEW telephelyLista;
```

b) Az előző utasítás-sorozatot a nézet eldobását megelőzően egészítse ki olyan *utasításblokkal*, amely a nézettábla adataiból formázottan kiírja az ország nevét és az országban található telephelyek számát, az ország neve szerint rendezve!

```
CREATE OR REPLACE VIEW telephelyLista AS
  SELECT location_id, street_address, city, country_name
  FROM locations INNER JOIN countries USING(country_id);
SELECT * FROM telephelyLista
  ORDER BY country_name, city, street_address;
DECLARE
  orszag countries.country_name%TYPE;
  thsz   NUMBER(3);
  CURSOR country_cur IS
    SELECT country_name, COUNT(*) AS telephelyek_száma
    FROM telephelyLista
    GROUP BY country_name
    ORDER BY country_name;
BEGIN
  OPEN country_cur;
  FETCH country_cur INTO orszag, thsz;
  WHILE country_cur%FOUND LOOP
    DBMS_OUTPUT.PUT_LINE(RPAD(orszag, 14) || ': ' || LPAD(thsz, 3));
    FETCH country_cur INTO orszag, thsz;
  END LOOP;
  CLOSE country_cur;
END;
/
DROP VIEW telephelyLista;
```

```
1  CREATE OR REPLACE VIEW telephelyLista AS
2    SELECT location_id, street_address, city, country_name
3    FROM locations INNER JOIN countries USING(country_id);
4  SELECT * FROM telephelyLista
5    ORDER BY country_name, city, street_address;
6  DECLARE
7    orszag countries.country_name%TYPE;
8    thsz   NUMBER(3);
9    CURSOR country_cur IS
10     SELECT country_name, COUNT(*) AS telephelyek_száma
11     FROM telephelyLista
12     GROUP BY country_name
13     ORDER BY country_name;
14 BEGIN
15   OPEN country_cur;
16   FETCH country_cur INTO orszag, thsz;
17   WHILE country_cur%FOUND LOOP
18     DBMS_OUTPUT.PUT_LINE(RPAD(orszag, 14) || ': ' || LPAD(thsz, 3));
19     FETCH country_cur INTO orszag, thsz;
20   END LOOP;
21   CLOSE country_cur;
22 END;
23 /
24 DROP VIEW telephelyLista;
```

6. feladat: Készítsen másolatot az *employees* tábláról *dolgozo* néven! A *dolgozo* táblára írjon olyan *utasításblokkot*, amely *50 dollárral* növeli minden olyan *50-es részlegazonosítójú* dolgozó fizetését, aki *5000 dollárnál kevesebbet* keres!

```
CREATE TABLE dolgozo
  AS SELECT * FROM employees;
DECLARE
  CURSOR dolg_cur IS
    SELECT employee_id, last_name, salary
    FROM dolgozo
    WHERE department_id = 50 AND salary < 5000
    FOR UPDATE NOWAIT;
  az dolgozo.employee_id%TYPE;
  nev dolgozo.last_name%TYPE;
  fiz dolgozo.salary%TYPE;

BEGIN
  OPEN dolg_cur;
  LOOP
    FETCH dolg_cur INTO az, nev, fiz;
    EXIT WHEN dolg_cur%NOTFOUND;
    fiz := fiz + 50;
    UPDATE dolgozo
      SET salary = fiz
      WHERE CURRENT OF dolg_cur;
    DBMS_OUTPUT.PUT_LINE(nev || ': ' || fiz);
  END LOOP;
```

```
  CLOSE dolg_cur;
END;


CREATE OR REPLACE PROCEDURE atlfelettReszleg(rnev
departments.department_name%TYPE) AS
  rid departments.department_id%TYPE;
  CURSOR c1 IS
    SELECT last_name, job_id, salary
    FROM departments INNER JOIN employees USING(department_id)
    WHERE salary > (
        SELECT AVG(salary)
        FROM departments INNER JOIN employees USING(department_id)
        WHERE UPPER(department_name) = UPPER(rnev)
      )
      AND UPPER(department_name) = UPPER(rnev)
    ORDER BY last_name;
  r1 c1%ROWTYPE;
BEGIN
  SELECT department_id
    INTO rid
    FROM departments
    WHERE UPPER(department_name) = UPPER(rnev);
  FOR r1 IN c1 LOOP
    DBMS_OUTPUT.PUT_LINE(r1.last_name || '(' || r1.job_id || '): ' ||
r1.salary);
  END LOOP;
EXCEPTION
  WHEN No_Data_Found
    THEN DBMS_OUTPUT.PUT_LINE('Nincs ilyen részleg!');
END atlfelettReszleg;
/
```