Tutoriat 4

WITH; EXISTS; ROWNUM;

WITH

Probabil una dintre cele mai utile clauze WITH ne permite sa definim o cerere si sa-i folosim rezultatul ca si cand ar fi un tabel. Aceasta clauza ne ajuta atunci cand avem nevoie de rezultatul unei cereri in mai multe locuri si pentru a ne putea structura programul in bucati mai mici si mai usor de implementat. Sintaxa clauzei WITH este:

```
1.  WITH nume_cerere1 AS ( SELECT ...),
2.       nume_cerere2 AS ( SELECT ...),
3.       ...
4.       nume_cerereN AS ( SELECT ...)
5.  SELECT ...
6.  FROM nume_cerere1, nume_cerere2, ..., nume_cerereN;
```

Asa cum se vede, in clauza WITH pot fi create si folosite orice numar de cereri. De asemenea in interiorul unei cereri pot fi folosite toate cererile definite anterior. Sa exemplificam acest lucru afisand toti angajatii din tabelul EMPLOYEES al caror nume incepe cu litera „A” si au salariul mai mare de 5000:

```
1.  WITH selecteaza_angajati AS(SELECT *
2.                              FROM employees
3.                              WHERE upper(last_name) LIKE 'A%'),
4.       verifica_salariu AS (SELECT *
5.                            FROM  selecteaza_angajati
6.                            WHERE salary > 5000)
7.  SELECT last_name, hire_date, salary
8.  FROM verifica_salariu;
```

LAST_NAME	HIRE_DATE	SALARY
Ande	24-03-2000	6400
Abel	11-05-1995	11000

Pentru rezolvare m-am folosit de o cerere noua pentru fiecare conditie pentru a exemplifica cum acestea pot fi folosite una in interiorul alteia.

De asemenea o astfel de cerere nu are nevoie neaparata de o clauza WITH, ea poate fi scrisa si in clauza FROM a cererii principale:

```
SELECT ...
FROM ( SELECT ...) nume_cerere1,
     ( SELECT... ) nume_cerere2,
      ...,
     ( SELECT... ) nume_cerereN;
```

Acest lucru nu este incurajat daca trebuie declarate mai multe cereri deoarece incarca codul destul de mult.

ATENTIE: Daca declar cererile in clauza FROM atunci nu voi mai avea acces din interiorul unei cereri la toate cererile declarate anterior decat prin join in clauza principala!

Sa rescriem problema anterioara folosind acest tip de declarare:

```
SELECT verifica_salariu.last_name, verifica_salariu.hire_date, verifica_salariu.salary
FROM (SELECT *
      FROM employees
      WHERE upper(last_name) LIKE 'A%') selecteaza_angajati,
     (SELECT *
      FROM  employees
      WHERE salary > 5000) verifica_salariu
WHERE selecteaza_angajati.employee_id = verifica_salariu.employee_id;
```

LAST_NAME	HIRE_DATE	SALARY
Ande	24-03-2000	6400
Abel	11-05-1995	11000

Asa cum putem vedea rezultatul este acelasi, singura diferenta fiind join-ul dintre cele 2 tabele.

Clauza WITH si subcererile sunt foarte utile in scrierea unei cereri. Acestea ne permit sa imitam intr-o oarecare masura functiile din limbaje cum ar fi C++ sau Python pentru a putea structura mult mai bine o cerere complexa.

EXISTS

EXISTS este operatorul ce ne permite sa verificam daca rezultatul unei subcereri este sau nu vid. Daca in subcerere se afla macar un rezultat atunci operatorul va intoarce TRUE, altfel FALSE.

Sa exemplificam utilizarea lui afisand numele tututror departamentelor unde exista macar un anagajat ce castiga mai mult de 12000.

```
1.  SELECT department_name
2.  FROM departments d
3.  WHERE EXISTS ( SELECT *
4.                 FROM employees e
5.                 WHERE e.department_id = d.department_id
6.                   AND salary > 12000);
```

Department_name
Marketing
Executive
Sales

Pentru fiecare departament subcererea intoarce toti angajatii ce au salariul mai mare decat 12000 si in cazul in care nu exista se va intoarce un tabel vid. Cu ajutorul lui EXISTS afisez toate departamentele ce au macar un angajat ce respecta conditia.

De asemenea negarea poate fi facuta folsind operatorul NOT EXISTS.

ROWNUM

Daca dorim afisarea unui anumit numar de linii dintr-o cerere putem folosi operatorul ROWNUM pentru a specifica cate linii dorim sa afisam astfel:

```
1.  SELECT *
2.  FROM tabel
3.  WHERE ROWNUM <= numar_linii;
```

In cazul acesta cererea va afisa numai primele „numar_linii” linii din tabelul „tabel”

IMPORTANT: Din cauza ordinii executarii clauzelor intr-o cererer toate clauzele ce se executa dupa clauza WHERE unde se afla ROWNUM vor fi aplicate numai asupra liniilor ce au ramas in tabel dupa ROWNUM! Astfel anumite clauze(cum ar fi ORDER BY) trebuie facute inainte (intr-o cerere cu clauza WITH de exemplu).

Ordinea executarii clauzelor intr-o cerere SQL:

ORDER	CLAUSE	FUNCTION
1	from	Choose and join tables to get base data.
2	where	Filters the base data.
3	group by	Aggregates the base data.
4	having	Filters the aggregated data.
5	select	Returns the final data.
6	order by	Sorts the final data.
7	limit	Limits the returned data to a row count.

Sa exemplificam acest lucru prin afisarea celor mai mari 5 salarii din tabelul EMPLOYEES:

```
1.  SELECT salary
2.  FROM employees
3.  WHERE ROWNUM <= 5
4.  ORDER BY salary DESC;
```

Salary
13000
6000
4400
2600
2600

Aceasta cerere este gresita. Exista angajati cu salarii mai mari in tabel, insa, ce s-a intamplat a fost ca, deoarece WHERE are prioritate in fata lui ORDER BY, prima data au fost selectate primele 5 linii din tabel, apoi ordonate dupa salariu. O rezolvare corecta este:

```
1.  SELECT salary
2.  FROM (SELECT *
3.        FROM employees
4.        ORDER BY salary DESC)
5.  WHERE ROWNUM <= 5;
```

Salary
24000
17000
17000
14000
13500

Ma folosesc de o cerere in clauza FROM ce imi intoarce tabelul deja sortat din care, in clauza WHERE, preiau numai primele 5 linii;

Exercitii se pot gasi pe Drive -> BD 2018-2019 -> Laborator5.