

Baze de date

Limbajul SQL

Teams: FI-AIA-2-Baze de date-2022-2023



THE **INFORMATION** COMPANY

Curs 5

Limbajul SQL

Limbajul SQL

Interogări SELECT pe o singură tabelă (partea III)

4.3. Funcții referitoare la mai multe înregistrări

4.3.1. Clauza **GROUP BY**

4.3.2. Excluderea grupurilor (clauza **HAVING**)

4.3.3. Imbricarea funcțiilor de grup

Tabele EMP si DEPT

Pentru exemplele din cursuri vom folosi tabela ***EMP***:















EMP													
Table	Data	Indexes	Model	Constraints	Grants	Statistics	UI Defaults	Triggers	Dependencies	SQL	REST	Sample Queries	
Add Column		Modify Column		Rename Column		Drop Column		Rename	Copy	Drop	Truncate	Create Lookup Table	Create App
Column Name				Data Type				Nullable			Default		Primary Key
EMPNO				NUMBER(4,0)				No			-		1
ENAME				VARCHAR2(50)				Yes			-		-
JOB				VARCHAR2(50)				Yes			-		-
MGR				NUMBER(4,0)				Yes			-		-
HIREDATE				DATE				Yes			-		-
SAL				NUMBER(7,2)				Yes			-		-
COMM				NUMBER(7,2)				Yes			-		-
DEPTNO				NUMBER(2,0)				Yes			-		-

Structura tablei ***EMP***.

Tabele EMP si DEPT

Pentru exemplele din cursuri vom folosi tabela ***EMP***:

EMP

Table	Data	Indexes	Model	Constraints	Grants	Statistics	UI Defaults	Triggers	Dependencies	SQL	REST	Sample Queries
Query	Count Rows	Insert Row	Load Data									
EDIT	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO				
	7839	KING	PRESIDENT	-	11/17/1981	34475.65	-	10				
	7698	BLAKE	MANAGER	7839	05/01/1981	19651.14	-	30				
	7782	CLARK	MANAGER	7839	06/09/1981	16893.06	-	10				
	7566	JONES	MANAGER	7839	04/02/1981	20513.04	-	20				
	7788	SCOTT	ANALYST	7566	12/09/1982	20685.39	-	20				
	7902	SMITH	ANALYST	7566	12/03/1981	20685.39	-	20				
	7369	SMITH	CLERK	7902	12/17/1980	5516.12	-	20				
	7499	ALLEN	SALESMAN	7698	02/20/1981	11032.2	300	30				
	7521	WARD	SALESMAN	7698	02/22/1981	8618.96	500	30				
	7654	SMITH	SALESMAN	7698	09/28/1981	8618.96	1400	30				
	7844	TURNER	SALESMAN	7698	09/08/1981	10342.72	0	30				
	7876	ADAMS	CLERK	7788	01/12/1983	7584.64	-	20				
	7900	JAMES	CLERK	7698	12/03/1981	6550.4	-	30				
	7934	MILLER	CLERK	7782	01/23/1982	8963.67	-	10				

Continutul initial al tablei ***EMP***.

Tabele EMP si DEPT

Table	Data	Indexes	Model	Constraints	Grants	Statistics	UI Defaults	Triggers	Dependencies	SQL	REST	Sample Queries
<div>Add Column</div> <div>Modify Column</div> <div>Rename Column</div> <div>Drop Column</div> <div>Rename</div> <div>Copy</div> <div>Drop</div> <div>Truncate</div> <div>Create Lookup Table</div> <div>Create App</div>												
Column Name				Data Type			Nullable		Default		Primary Key	
DEPTNO				NUMBER(2,0)			No		-		1	
DNAME				VARCHAR2(50)			Yes		-		-	
LOC				VARCHAR2(50)			Yes		-		-	

Structura tabelii *DEPT*.

DEPT

Table

Data

Indexes

Model

Constraints

Grants

Statistics

UI Defaults

Triggers

Dependencies

SQL

REST






Sample Queries

Query

Count Rows

Insert Row

Load Data

EDIT	DEPTNO	DNAME	LOC
	10	ACCOUNTING	NEW YORK
	20	RESEARCH	DALLAS
	30	SALES	CHICAGO
	40	OPERATIONS	BOSTON
	15	Departament IT	123

Continutul initial al tabelii *DEPT*.

Funcții

Funcțiile sunt o caracteristică importantă a SQL si sunt utilizate pentru:

- ✓ a realiza calcule asupra datelor
- ✓ a modifica date
- ✓ a manipula grupuri de înregistrări
- ✓ a schimba formatul datelor
- ✓ sau pentru a converti diferite tipuri de date

Funcții

Funcțiile se clasifică în două tipuri:

1. Funcții referitoare la o singură înregistrare
(single-row functions)
2. Funcții referitoare la mai multe înregistrări
(multiple-row functions)

Funcții

1. Funcții referitoare la o singură înregistrare (single-row functions):

1. funcții caracter
2. funcții numerice
3. funcții pentru data calendaristică si oră
4. funcții de conversie
5. funcții diverse

Funcții

2. Funcții referitoare la mai multe înregistrări (multiple-row functions):

- funcții totalizatoare sau funcții de grup

Funcții

Diferența dintre cele două tipuri de funcții este numărul de înregistrări pe care acționează:

- *Funcțiile referitoare la o singură înregistrare returnează un singur rezultat pentru fiecare rând al tabelului,*
- *pe când funcțiile referitoare la mai multe înregistrări returnează un singur rezultat pentru fiecare grup de înregistrări din tabelă.*

Funcții

O observație importantă este faptul că dacă se apelează o funcție **SQL** ce are un argument (parametru) egal cu valoarea **Null**, atunci în mod automat rezultatul va avea valoarea **Null**.

Singurele funcții care nu respectă această regulă sunt:

- **CONCAT**
- **DECODE**
- **DUMP**
- **NVL**
- **REPLACE**

Limbajul SQL

Cereri SELECT pe o tabela

4.1. Funcții

4.2. Funcții referitoare la o singură înregistrare

4.3. Funcții referitoare la mai multe înregistrări

4.3.1. Clauza **GROUP BY**

4.3.2. Excluderea grupurilor (clauza **HAVING**)

4.3.3. Imbricarea funcțiilor de grup

4.3. Funcții de grup

Funcțiile de grup sunt funcții care operează pe un set de rânduri pentru a da un rezultat pe întreg setul.

Parametrii și descrierea funcțiilor de grup.

Funcțiile de grup sunt:

1. **AVG**
2. **COUNT**
3. **MAX**
4. **MIN**
5. **STDDEV**
6. **SUM**
7. **VARIANCE**

4.3. Funcții de grup

Fiecare dintre aceste funcții acceptă anumiți parametri:

Funcția	Descriere
AVG([DISTINCT ALL]n)	Valoarea medie pentru grup, ignorand valorile nule
COUNT({* [DISTINCT ALL]expr})	Numarul de randuri unde expr evalueaza altceva in afara de null (folosind * sunt numarate toate randurile, incluzand duplicatele si pe cele cu valoare nula)
MAX([DISTINCT ALL]expr)	Valoarea maxima a expr , ignorand valorile nule
MIN([DISTINCT ALL]expr)	Valoarea minima a expr , ignorand valorile nule
STDDEV([DISTINCT ALL]x)	Deviatia standard pentru grup, ignorand valorile nule
SUM([DISTINCT ALL]x)	Suma valorilor pentru grup, ignorand valorile nule
VARIANCE([DISTINCT ALL]x)	Variatia pentru grup, ignorand valorile nule

4.3. Funcții de grup

DISTINCT face ca funcția să ignore valorile duplicat.

ALL face ca funcția să afișeze și valorile duplicat.

Valoarea implicită este **ALL**, deci nu este necesar să fie specificată.

Tipul de dată returnat de funcția **expr** poate fi **CHAR**, **VARCHAR2**, **NUMBER** sau **DATE**.

Toate funcțiile de grup ignoră valorile nule.

Pentru a lua în considerare și valorile nule se folosesc funcțiile **NVL**, **NVL2** sau **COALESCE**.

4.3. Funcții de grup

Sintaxa funcțiilor de grup:

```
SELECT [coloana,] functie_de_grup(coloana),  
...  
FROM tabel  
[WHERE conditie]  
[GROUP BY coloana]  
[HAVING conditie_de_grupare]  
[ORDER BY coloana];
```

Rezultatele sunt sortate implicit crescător. Pentru o ordonare descrescătoare se va folosi clauza **DESC** după **ORDER BY**.

4.3. Funcții de grup

Exemplul 1: Afișarea salariului mediu, maxim, minim și suma tuturor salariilor angajaților cu funcție SALESMAN.

```
SELECT AVG(sal), MAX(sal), MIN(sal), SUM(sal)  
FROM EMP  
WHERE job = 'SALESMAN';
```

```
1  SELECT AVG(sal), MAX(sal), MIN(sal), SUM(sal)  
2  FROM EMP  
3  WHERE job = 'SALESMAN';
```

Results Explain Describe Saved SQL History			
AVG(SAL)	MAX(SAL)	MIN(SAL)	SUM(SAL)
1400	1600	1250	5600
1 rows returned in 0.01 seconds Download			

4.3. Funcții de grup

Exemplul 2 - Datele la care s-au făcut prima și ultima angajare.

```
SELECT MIN(hiredate), MAX(hiredate)
FROM EMP;
```

```
1  SELECT MIN(hiredate), MAX(hiredate)
2  FROM EMP;
```

Results		Explain	Describe	Saved SQL	History
MIN(HIREDATE)			MAX(HIREDATE)		
12/17/1980			01/12/1983		
1 rows returned in 0.00 seconds					
Download					

4.3. Funcții de grup

Exemplul 3 - Primul și ultimul nume de angajat în ordine alfabetică:

```
SELECT MIN(ename), MAX(ename)  
FROM EMP;
```

```
1  SELECT MIN(ename), MAX(ename)  
2  FROM EMP;
```

ResultsExplainDescribeSaved SQLHistory

MIN(ENAME)	MAX(ENAME)
ADAMS	WARD

1 rows returned in 0.00 secondsDownload

4.3. Funcții de grup

Funcția **COUNT**

Funcția **COUNT** are 3 formate:

COUNT(*)

COUNT(expr)

COUNT(DISTINCT expr)

4.3. Funcții de grup

- **COUNT(*)** întoarce numărul de rânduri dintr-o tabela care satisface criteriul de selecție, incluzând rândurile duplicat și rândurile conținând valori nule.
- Dacă clauza **WHERE** este introdusă, atunci **COUNT(*)** returnează numărul de rânduri care satisfac condiția din clauza **WHERE**.

4.3. Funcții de grup

- În contrast, funcția **COUNT(expr)** întoarce numărul de valori nenule din coloana specificată de **expr**.
- **COUNT(DISTINCT expr)** returnează numărul de valori distincte, nenule din coloana specificată de **expr**.

4.3. Funcții de grup

Exemplul 4

Numărul angajaților din departamentul cu id-ul 30.

```
SELECT COUNT(*)  
FROM EMP  
WHERE deptno = 30;
```

Column Name	Data Type	Nullable	Default	Primary Key
EMPNO	NUMBER(4,0)	No		1
ENAME	VARCHAR2(50)	Yes		
JOB	VARCHAR2(50)	Yes		
MGR	NUMBER(4,0)	Yes		
HIREDATE	DATE	Yes		
SAL	NUMBER(7,2)	Yes		
COMM	NUMBER(7,2)	Yes		
DEPTNO	NUMBER(2,0)	Yes		

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT		11/17/1981	5000		10
7698	BLAKE	MANAGER	7839	5/1/1981	2850		30
7782	CLARK	MANAGER	7839	6/9/1981	2450		10
7566	JONES	MANAGER	7839	4/2/1981	2975		20
7788	SCOTT	ANALYST	7566	12/9/1982	3000		20
7902	FORD	ANALYST	7566	12/3/1981	3000		20
7369	SMITH	CLERK	7902	12/17/1980	800		20
7499	ALLEN	SALESMAN	7698	2/20/1981	1600	300	30
7521	WARD	SALESMAN	7698	2/22/1981	1250	500	30
7654	MARTIN	SALESMAN	7698	9/28/1981	1250	1400	30
7844	TURNER	SALESMAN	7698	9/8/1981	1500	0	30
7876	ADAMS	CLERK	7788	1/12/1983	1100		20
7900	JAMES	CLERK	7698	12/3/1981	950		30
7934	MILLER	CLERK	7782	1/23/1982	1300		10

4.3. Funcții de grup

Rezultatul obtinut - Numărul angajaților din departamentul cu id-ul 30.

```
1  SELECT COUNT(*)  
2  FROM EMP  
3  WHERE deptno = 30;
```

Results	Explain	Describe	Saved SQL	History
				COUNT(*)
6				
1 rows returned in 0.00 seconds Download				

4.3. Funcții de grup

Exemplul 5

Numărul angajaților care iau comision din departamentul 30.

```
SELECT COUNT(comm)  
FROM EMP  
WHERE deptno = 30;
```

4.3. Funcții de grup

Rezultatul obtinut - Numărul angajaților care iau comision din departamentul 30:

```
1  SELECT COUNT(comm)
2  FROM EMP
3  WHERE deptno = 30;
```

Results	Explain	Describe	Saved SQL	History
COUNT(COMM)				
4				
1 rows returned in 0.00 seconds Download				

4.3. Funcții de grup

Exemplul 6 - Numărul de departamente din firma
(**varianta incorectă** și **varianta corectă**).

SELECT COUNT(deptno), **COUNT(DISTINCT** deptno)
FROM EMP;

```
1 SELECT COUNT(deptno), COUNT(DISTINCT deptno)
2 FROM EMP;
3
```

Results Explain Describe Saved SQL History	
COUNT(DEPTNO)	COUNT(DISTINCTDEPTNO)
14	3
1 rows returned in 0.02 seconds Download	

4.3. Funcții de grup

Exemplul 7 - Comisionul mediu în departamentul 30
(ignorând sau nu valorile nule):

```
SELECT AVG(comm), AVG(NVL(comm, 0))
FROM EMP
WHERE deptno = 30;
```

[illegible]

Limbajul SQL

Cereri SELECT pe o tabela

4.1. Funcții

4.2. Funcții referitoare la o singură înregistrare

4.3. Funcții referitoare la mai multe înregistrări

4.3.1. Clauza GROUP BY

4.3.2. Excluderea grupurilor (clauza **HAVING**)

4.3.3. Imbricarea funcțiilor de grup

4.3.1. Clauza GROUP BY

- Până acum toate funcțiile de grup au fost aplicate întregii tabele.
- Pentru a putea împărți tabela în grupuri mai mici se folosește clauza **GROUP BY**.
- Folosirea acesteia returnează informații sumare despre fiecare grup.

4.3.1. Clauza GROUP BY

- Folosind **GROUP BY** nu se pot extrage și coloane individuale, ci doar coloane ce rămân identice în tot grupul.
- Folosind **WHERE** se pot exclude rânduri, înaintea împărțirii lor în grupuri.
- Nu pot fi folosite aliasuri de coloane în clauza **GROUP BY**.
- Implicit, rândurile sunt sortate crescător după coloana (coloanele) specificate în **GROUP BY**.
- Acest lucru poate fi schimbat folosind **ORDER BY**.

4.3.1. Clauza GROUP BY

Exemplul 8 - Salariul mediu pe fiecare department:

```
SELECT deptno, AVG(sal)
FROM EMP
GROUP BY deptno;
```

Column Name	Data Type	Nullable	Default	Primary Key
EMPNO	NUMBER(4,0)	No		1
ENAME	VARCHAR2(50)	Yes		
JOB	VARCHAR2(50)	Yes		
MGR	NUMBER(4,0)	Yes		
HIREDATE	DATE	Yes		
SAL	NUMBER(7,2)	Yes		
COMM	NUMBER(7,2)	Yes		
DEPTNO	NUMBER(2,0)	Yes		

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT		11/17/1981	5000		10
7698	BLAKE	MANAGER	7839	5/1/1981	2850		30
7782	CLARK	MANAGER	7839	6/9/1981	2450		10
7566	JONES	MANAGER	7839	4/2/1981	2975		20
7788	SCOTT	ANALYST	7566	12/9/1982	3000		20
7902	FORD	ANALYST	7566	12/3/1981	3000		20
7369	SMITH	CLERK	7902	12/17/1980	800		20
7499	ALLEN	SALESMAN	7698	2/20/1981	1600	300	30
7521	WARD	SALESMAN	7698	2/22/1981	1250	500	30
7654	MARTIN	SALESMAN	7698	9/28/1981	1250	1400	30
7844	TURNER	SALESMAN	7698	9/8/1981	1500	0	30
7876	ADAMS	CLERK	7788	1/12/1983	1100		20
7900	JAMES	CLERK	7698	12/3/1981	950		30
7934	MILLER	CLERK	7782	1/23/1982	1300		10

4.3.1. Clauza GROUP BY

Rezultatul obtinut - Salariul mediu pe fiecare department:

[illegible]

4.3.1. Clauza GROUP BY

Exemplul 9

Salariul mediu pe fiecare departament, iar rezultatele ordonate după salariul mediu pe departament.

```
SELECT deptno, AVG(sal)
FROM EMP
GROUP BY deptno
ORDER BY AVG(sal);
```

4.3.1. Clauza GROUP BY

Rezultatul obtinut - Salariul mediu pe fiecare departament, iar rezultatele ordonate după salariul mediu pe departament.

[illegible]

4.3.1. Clauza GROUP BY

Gruparea după mai multe coloane.

Câteodată este necesară obținerea de rezultate pentru grupuri în alte grupuri.

Atunci în dreptul clauzei **GROUP BY** vom întâlni mai multe coloane.

4.3.1. Clauza GROUP BY

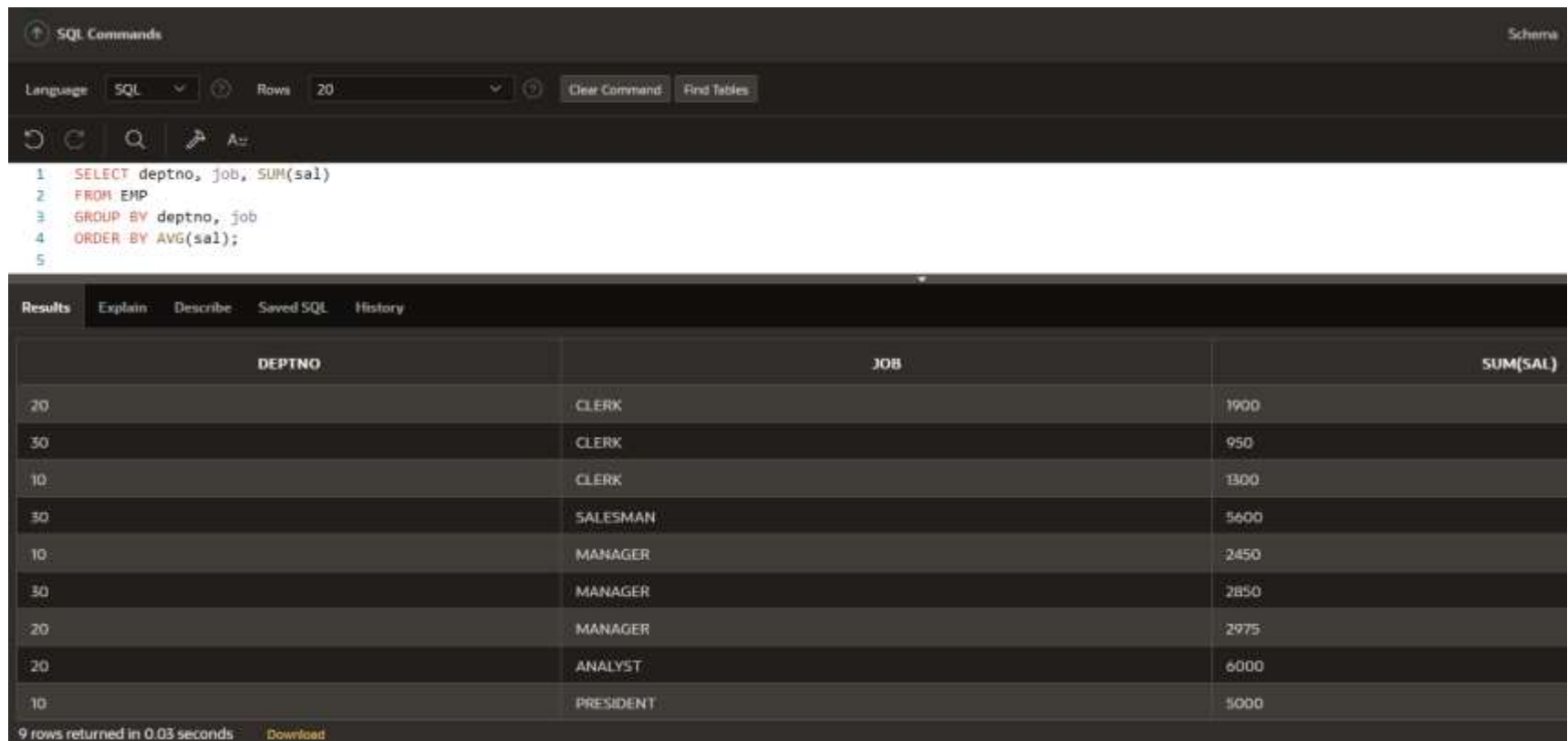
Exemplul 10

Salariul total pe fiecare departament si pe fiecare functie, iar rezultatele ordonate după salariul mediu pe departament.

```
SELECT deptno, job, SUM(sal)
FROM EMP
GROUP BY deptno, job
ORDER BY AVG(sal);
```

4.3.1. Clauza GROUP BY

Rezultatul obtinut - Salariul total pe fiecare departament si pe fiecare functie, iar rezultatele ordonate după salariul mediu pe departament.



The screenshot shows a SQL IDE interface. The top section is titled 'SQL Commands' and contains a query editor with the following SQL code:

```
1 SELECT deptno, job, SUM(sal)
2 FROM EMP
3 GROUP BY deptno, job
4 ORDER BY AVG(sal);
5
```

Below the query editor, the 'Results' tab is active, displaying a table with 9 rows. The table has three columns: DEPTNO, JOB, and SUM(SAL). The data is ordered by the average salary (AVG(sal)) for each department and job combination.

DEPTNO	JOB	SUM(SAL)
20	CLERK	1900
30	CLERK	950
10	CLERK	1300
30	SALESMAN	5600
10	MANAGER	2450
30	MANAGER	2850
20	MANAGER	2975
20	ANALYST	6000
10	PRESIDENT	5000

At the bottom of the results section, it states '9 rows returned in 0.03 seconds' and provides a 'Download' button.

Limbajul SQL

Cereri SELECT pe o tabela

4.1. Funcții

4.2. Funcții referitoare la o singură înregistrare

4.3. Funcții referitoare la mai multe înregistrări

4.3.1. Clauza **GROUP BY**

4.3.2. Excluderea grupurilor (clauza **HAVING**)

4.3.3. Imbricarea funcțiilor de grup

4.3.2. Excluderea grupurilor (clauza **HAVING**)

Clauza **HAVING** funcționează în mare ca și clauza **WHERE**, diferența fiind că **HAVING** este folosit pentru a exclude anumite grupuri din rezultat, nu rânduri cum făcea **WHERE**.

Clauza **HAVING** poate fi folosită înainte de **GROUP BY**, însă este mai logic să fie folosită după.

Ordinea execuției va rămâne aceeași.

4.3.2. Excluderea grupurilor (clauza **HAVING**)

Exemplul 11

Salariul mediu pe fiecare departament unde acesta depășește 2000\$.

```
SELECT deptno, AVG(sal)
FROM EMP
HAVING AVG(sal) > 2000
GROUP BY deptno;
```

4.3.2. Excluderea grupurilor (clauza **HAVING**)

Rezultatul obtinut - Salariul mediu pe fiecare departament unde acesta depășește 2000\$.

[illegible]

4.3.2. Excluderea grupurilor (clauza **HAVING**)

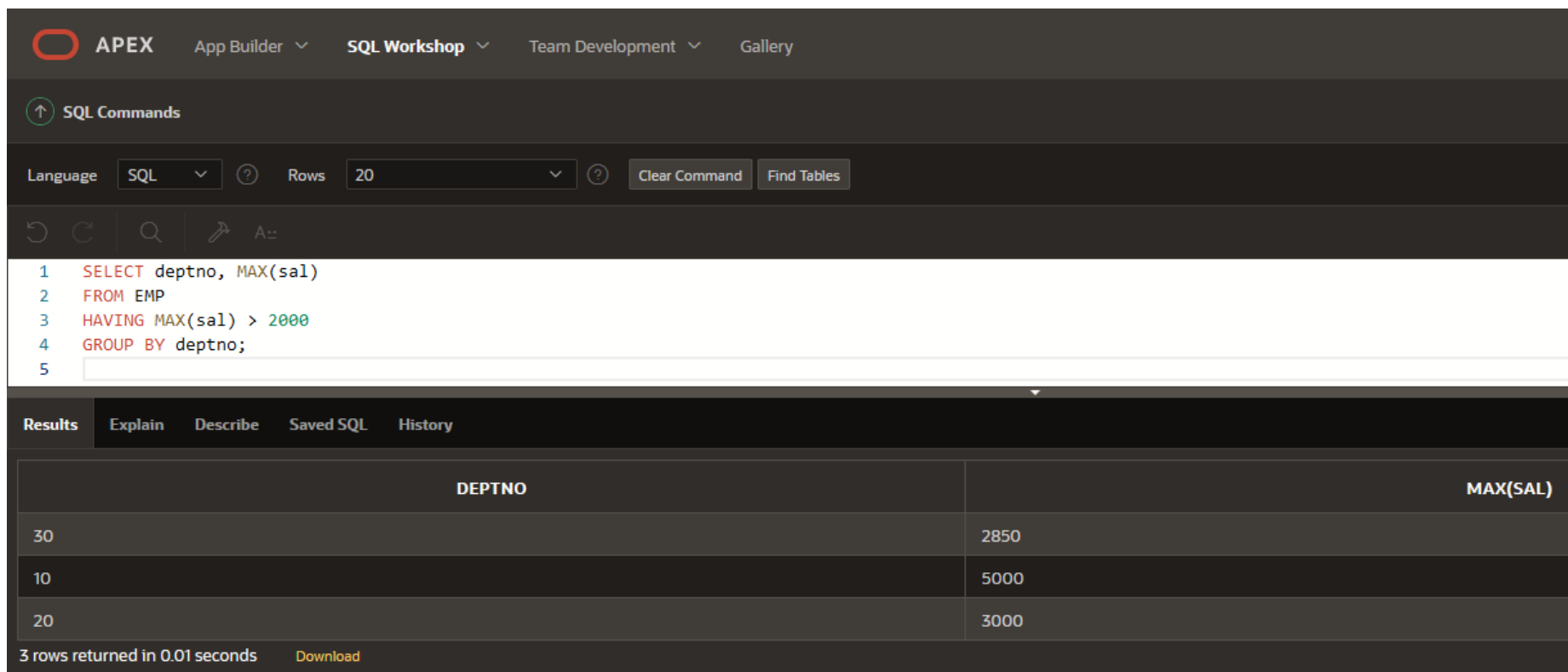
Exemplul 12

Salariul maxim pe fiecare departament unde acesta depășește 2000\$.

```
SELECT deptno, MAX(sal)  
FROM EMP  
HAVING MAX(sal) > 2000  
GROUP BY deptno;
```

4.3.2. Excluderea grupurilor (clauza **HAVING**)

Rezultatul obtinut - Salariul maxim pe fiecare departament unde acesta depășește 2000\$.



The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Commands' section is active, showing a query with 20 rows. The query is:

```
1 SELECT deptno, MAX(sal)
2 FROM EMP
3 HAVING MAX(sal) > 2000
4 GROUP BY deptno;
5
```

The 'Results' tab is selected, displaying a table with two columns: 'DEPTNO' and 'MAX(SAL)'. The table contains three rows of data:

DEPTNO	MAX(SAL)
30	2850
10	5000
20	3000

At the bottom, it states '3 rows returned in 0.01 seconds' and provides a 'Download' link.

4.3.2. Excluderea grupurilor (clauza **HAVING**)

Exemplul 13

Salariul total pe fiecare funcție, fără a lua în calcul **MANAGERII**, excluzând funcțiile cu suma salariilor sub 6000\$ cu ordonare după total.

```
SELECT job, SUM(sal)
FROM EMP
WHERE job != 'MANAGER'
GROUP BY job
HAVING SUM(sal) < 6000
ORDER BY SUM(sal);
```

Salariul total pe fiecare funcție

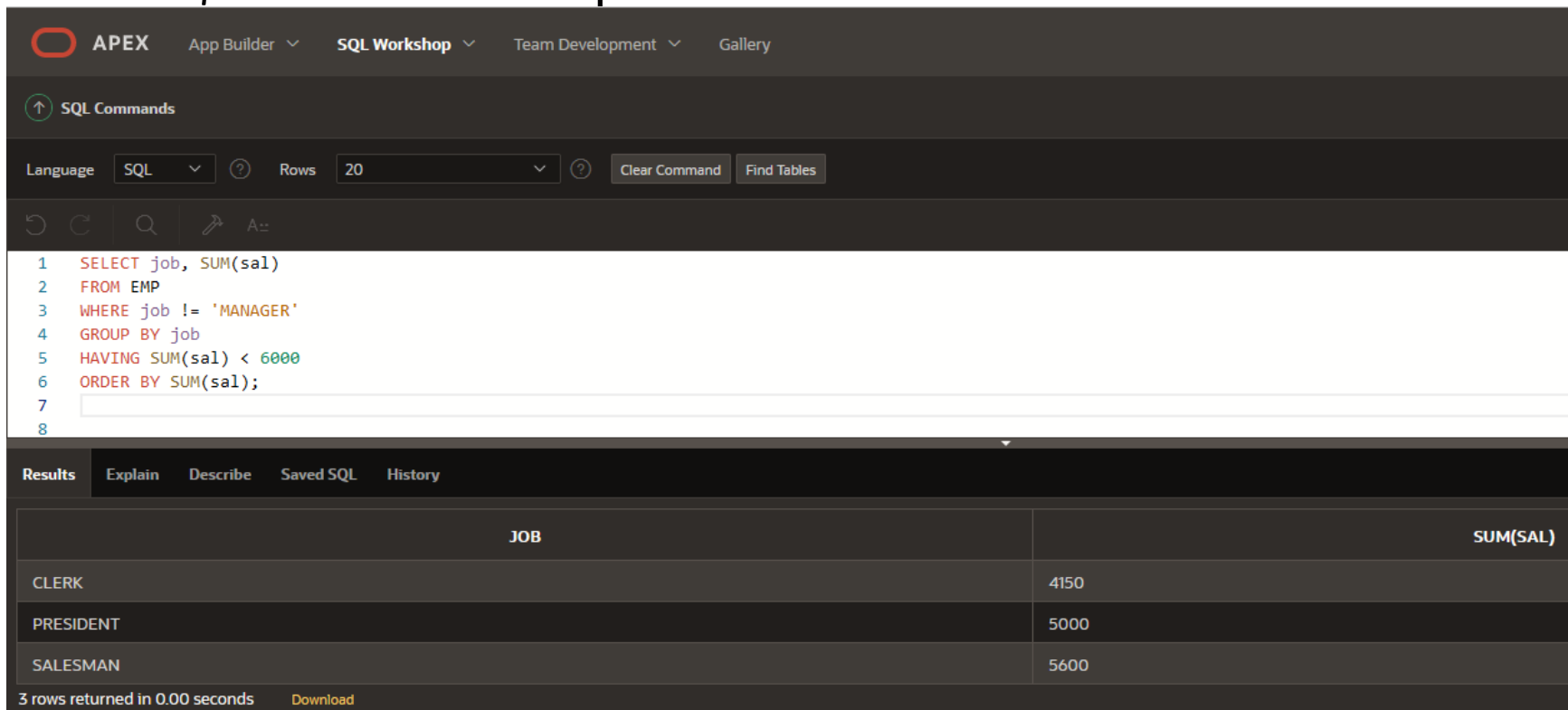
fără a lua în calcul **MANAGERII**

excluzând funcțiile cu suma salariilor sub 6000\$

cu ordonare după total

4.3.2. Excluderea grupurilor (clauza **HAVING**)

Rezultatul obtinut - Salariul total pe fiecare funcție, fără a lua în calcul MANAGERII, excluzând funcțiile cu suma salariilor sub 6000\$ cu ordonare după total.



The screenshot shows the APEX SQL Workshop interface. The SQL Commands tab is active, displaying the following query:

```
1 SELECT job, SUM(sal)
2 FROM EMP
3 WHERE job != 'MANAGER'
4 GROUP BY job
5 HAVING SUM(sal) < 6000
6 ORDER BY SUM(sal);
7
8
```

The Results tab shows the output of the query, which is a table with two columns: JOB and SUM(SAL). The table contains three rows of data:

JOB	SUM(SAL)
CLERK	4150
PRESIDENT	5000
SALESMAN	5600

At the bottom of the Results tab, it indicates "3 rows returned in 0.00 seconds" and provides a "Download" link.

Limbajul SQL

Cereri SELECT pe o tabela

4.1. Funcții

4.2. Funcții referitoare la o singură înregistrare

4.3. Funcții referitoare la mai multe înregistrări

4.3.1. Clauza **GROUP BY**

4.3.2. Excluderea grupurilor (clauza **HAVING**)

4.3.3. Imbricarea funcțiilor de grup

Ordinea de executie a functiilor de grup

Serverul **Oracle** execută funcțiile de grup într-o anumită ordine:

1. Selecția rândurilor ce respectă clauza **WHERE**
2. Gruparea rândurilor obținute, respectând clauza **GROUP BY**
3. Calcularea rezultatelor funcțiilor de grup pentru fiecare grup în parte
4. Eliminarea grupurilor ce nu respectă clauza **HAVING**
5. Ordonarea rezultatelor respectând clauza **GROUP BY**.

Ordinea de executie a functiilor de grup

- Ordinea de execuție are o importanță foarte mare, deoarece are un impact direct asupra vitezei.
- Cu cât mai multe înregistrări pot fi eliminate utilizând clauza **WHERE**, cu atât mai puțin va dura gruparea și operațiile ce urmează.
- Dacă o cerere **SQL** este concepută să elimine înregistrări/grupuri doar folosind clauza **HAVING**, atunci ar fi bine de încercat dacă este posibil și prin clauza **WHERE**. De obicei, totuși, această rescriere nu va fi posibilă.

4.3.3. Imbricarea functiilor de grup

Funcțiile de grup pot fi imbricate cu o adâncime de 2.

Exemplul 14

Salariul mediu maxim.

```
SELECT MAX(AVG(sal))  
FROM EMP  
GROUP BY deptno;
```

4.3.3. Imbricarea functiilor de grup

Funcțiile de grup pot fi imbricate cu o adâncime de 2.
Rezultatul obtinut - Salariul mediu maxim.

[illegible]

Referințe bibliografice

- 1) <https://docs.oracle.com/cloud/help/ro/analytics-cloud/ACUBI/GUID-4CBCE8D4-CF17-43BD-AAEF-C5D614A8040A.htm#BILUG672>
- 2) https://www.tutorialspoint.com/sql_certificate/using_single_row_functions.htm
- 3) <https://www.w3resource.com/sql-exercises/>

Întrebări?