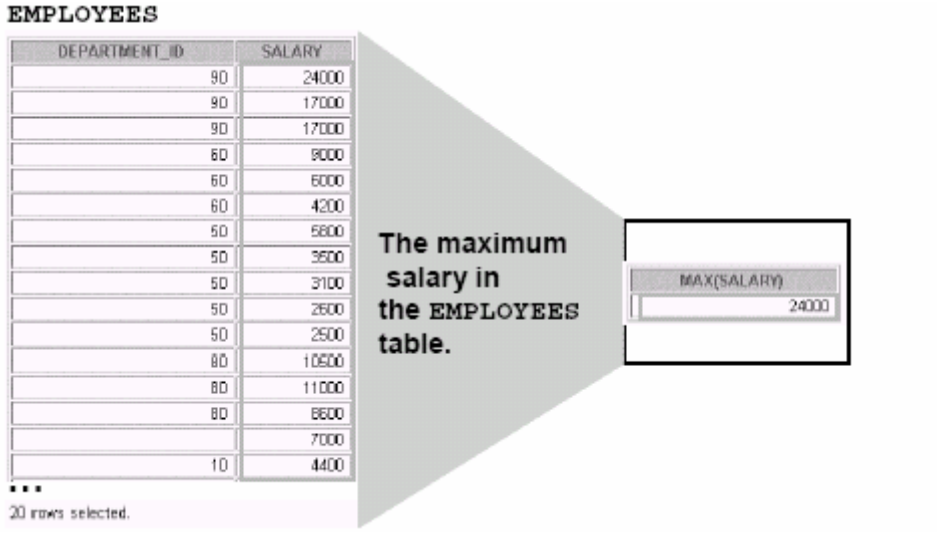
الفصل الخامس

Multiable Row Function

ماذا يعني ب "Grouping Function"؟

هي عبارة عن مجموعة من Functions تعمل علي مجموعة من الصفوف لتعرض نتيجة واحدة.
ويمكن أن تعمل علي كل بيانات الجدول أو علي جزء فقط من بيانات الجدول.



أنواع Group Functions

الوصف	الدوال
تقوم بحساب المتوسط الحسابى لمجموعة من الأرقام و تقوم بتجاهل القيمة الفارغة NULL .	AVG([DISTINCT|ALL]*n*)
تقوم بحساب عدد الحقول في عمود معين التي لا تحمل قيمةNULL.	COUNT({*|[DISTINCT|ALL]*expr*})
وهي اختصار كلمة MAXIMUM وهي تقوم بالبحث عن أكبر قيمة لمجموعة من القيم في عمود معين .	MAX([DISTINCT|ALL]*expr*)
وهي اختصار كلمة MINIMUM وهي تقوم بالبحث عن أقل قيمة موجودة في عمود معين.	MIN([DISTINCT|ALL]*expr*)
دالة الانحراف المعياري وهي اختصار لكلمة Stander deviation .	STDDEV([DISTINCT|ALL]*x*)
تقوم بحساب مجموع كل القيم الموجودة في عمود معين او فى مجموعة صفوف فقط.	SUM([DISTINCT|ALL]*n*)
تستخدم لحساب معدل الاختلاف .	VARIANCE([DISTINCT|ALL]*x*)

ملحوظة هامة جدا
كل Group Function تتجاهل NULL فيما عدا دالة COUNT اذا استخدمت مع *.
حيث عند استخدامها مع النجمة * اى (*) COUNT فهنا لا تتجاهل قيمة NULL.

الصيغة العامة لل Group Function

```
SELECT      [column,]  group function (column), ...
FROM        table
[WHERE      condition]
[GROUP BY   column]
[ORDER BY   column];
```

ملاحظة :
***تـرتبُ النّتيجةَ بترتيب تصاعدىِ عندما تستخدم فقرة GROUP BY. وللترتيب التنازلى نستخدم DESC فى فقرة ORDER BY.**

استخدام AVG و SUM
يمكنك استخدام دوال AVG و SUM للبيانات الرقمية.

```
SELECT  AVG(salary), MAX(salary),
        MIN(salary), SUM(salary)
FROM    employees
WHERE   job_id LIKE '%REP%';
```

AVG(SALARY)	MAX(SALARY)	MIN(SALARY)	SUM(SALARY)
8150	11000	6000	32600

المثال السابق يريد معرفة المتوسط الحسابى لمرتبات الموظفين و اعلى مرتب و اقل مرتب و مجموع المرتبات للموظفين .
*** يُمكنُ أتستخدام AVG و SUM و MIN و MAX مع الأعمدةِ التي يمكن أن تخزن بيانات رقمية.**

مثال اخر:

استخدام MIN وMAX
يمكنك استخدام MIN وMAX لأي نوع من البيانات.

```
SELECT  MIN(hire_date), MAX(hire_date)
FROM    employees;
```

MIN(HIRE_	MAX(HIRE_
17-JUN-87	29-JAN-00

يعرض المثال السابق اقدم موظف فى الشركة ذلك عن طريق استخدام MIN مع تاريخ التعين و يعرض ايضا احدث موظف فى الشركة عن طريق استخدام MAX .

```
SELECT MIN(last_name), MAX(last_name)
FROM   employees;
```

MIN(LAST_NAME)	MAX(LAST_NAME)
Abel	Zlotkey

ويعرض المثال السابق اسماء الموظفين وذلك عن طريق استخدام MAXو MIN فتقوم ال MAX بأظهار الاسم الذى يبدأ بأخر حرف هجائيا(Z). و تقوم ال MIN بأظهار الاسم الذى يبدأ بأول حرف هجائي (A).

لاحظ : كل FUNCTIONS تعمل علي القيم الرقمية فقط ماعدا MAX و MIN فمن الممكن أن يعملان مع التواريخ .

استخدام دالة COUNT

COUNT (*) تقوم بحساب عدد الصفوف في الجدول

```
SELECT COUNT(*)
FROM   employees
WHERE  department_id = 50;
```

COUNT(*)
5

عند استخدام النجمة (*) مع COUNT بدلا من اسم العمود فإنها تقوم بحساب عدد الصفوف الموجودة في الجدول .
ومن هنا نجد أن COUNT لها ثلاث أشكال :

1- COUNT(*)	**يقوم بحساب عدد الصفوف في الجدول ويتضمن الصفوف التي تحتوي قيمة فارغة NULL.**
2- COUNT(expr)	**فسوف تقوم بحساب عدد القيم التي لا تحتوي علي قيم فارغة NULL في الأعمدة التي حددت بواسطة expr.**
3- COUNT (DISTINCT expr)	**يقوم بحساب عدد القيم الفريدة اى الغير مكررة في الأعمدة المحددة.**

```
SELECT COUNT(commission_pct)
FROM   employees
WHERE  department_id = 80;
```

COUNT(COMMISSION_PCT)
3

يعرض المثال عددِ الموظفين في القسم 80 الذين يأخذون عمولة.
عند تحديد اسم عمود COUNT(commission_pct) مع COUNT فإنها تقوم بحساب عدد الصفوف التي لا تحتوي علي قيم فارغة NULL لهذا العمود وعدد الصفوف الموجودة والتي لا تحتوي علي NULL هم ثلاثة .

أعرض عددَ الأقسمِ الموجودة فى جدول الموظفين.

```
SELECT COUNT(department_id)
FROM   employees;
```

COUNT(DEPARTMENT_ID)
19

وتعتبر تلك النتيجة السابقة خاطئة لأنة قام بحساب التكرارت فى عمود DEPARTMENT_ID . ولمعالجة ذلك نقوم باستخدام DISTINCT كما فى المثال التالى .

استخدام DISTINCT

COUNT (DISTINCT expr) تستخدم لمنع احتساب الحقول التي تتكرر .اى تقوم بمنع Duplicate .

```
SELECT COUNT(DISTINCT department_id)
FROM   employees;
```

COUNT(DISTINCTDEPARTMENT_ID)
7

ملاحظة:استخدام DISTINCT لكى تمنع حِساب أي قِيَمِ مكررة فى العمودِ.

```
SELECT AVG(commission_pct)
FROM   employees;
```

AVG(COMMISSION_PCT)
.2125

كما ذكرنا ان جميع GROUP FUNTIONS تتجاهل قيم NULL فيما عدا COUNT(*) وعند حساب المتوسط لقيم العمود AVG(commission_pct) يتم تجاهل قيم NULL لبقية القيم و يتم حساب المتوسط عن طريق جمع قيم العمود و قسمته علي عدد الحقول التى بها قيمة فقط و بذلك يتم اعطاء نتيجة غير صحيحة للمتوسط الحسابى للعمولات commission وللتغلب علي هذه المشكلة نستخدم NVL.

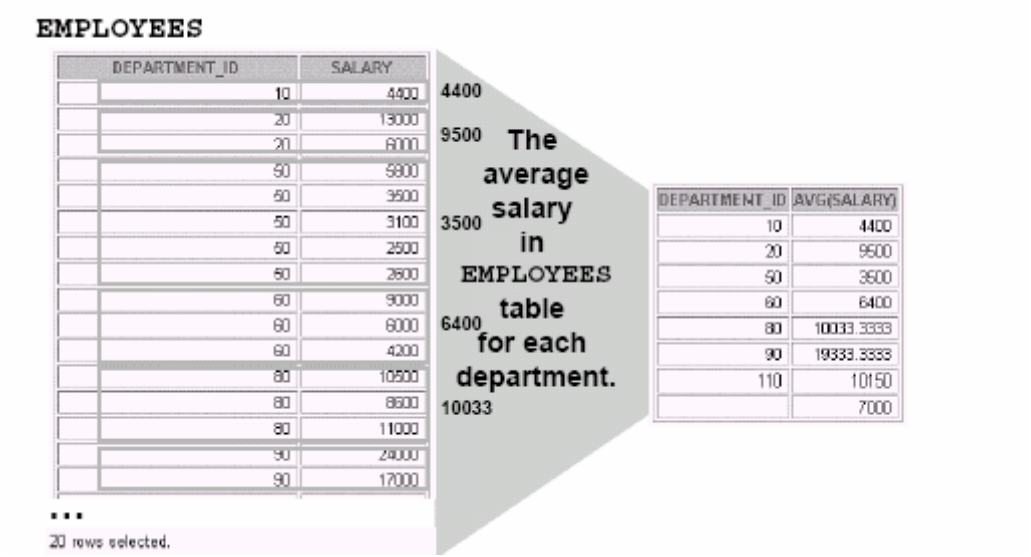
استخدام NVL مع GROUP FUNCTIONS

```
SELECT AVG(NVL(commission_pct, 0))
FROM   employees;
```

AVG(NVL(COMMISSION_PCT,0))
.0425

وبذلك يتم القسمة على عدد الحقول كلها وليس الحقول التى بها قيمة فقط.

استخدام GROUP BY



حتى الآن كل GROUP FUNCTIONS تتعامل مع الجدول كمجموعة واحدة.
ولكن باستخدام GROUP BY يمكنك من تقسيم البيانات.فيمكن حساب المتوسط للمرتبات بجدول الموظفين لكل قسم من الاقسام على حدة .

استخدام GROUP BY

```
SELECT        column, group_function(column)
FROM          table
[WHERE        condition]
[GROUP BY     group_by_expression]
[ORDER BY     column];
```

تقسم صفوف الجدول إلي مجموعات صغيرة بواسطة GROUP BY .
يمكنك استخدام GROUP BY كي تقسم صفوف الجداول إلي مجموعات.

بعض الشروط لاستخدام GROUP BY :

1. **اذا تم ادراج عمود فى جملة SELECT و أردت استخدام GROUP BY فلابد من ادراج تلك العمود المدرج فى جملة SELECT فى فقرة GROUP BY وبغير ذلك يحدث Error .**
2. **يمكن ان تستخدم فقرة WHERE لتحديد الصفوف المراد عرضها ذلك قبل استخدام فقرة GROUP BY .**
3. **لا يمكن استخدام الاسم المستعار (Alias) فى فقرة Group By .**
4. **يتم ترتيب الناتج بالترتيب التصاعدي .ويمكن أن تتجاوز هذا باستخدام الفقرة ORDER BY.**

استخدام GROUP BY

```
SELECT    department_id, AVG(salary)
FROM      employees
GROUP BY department_id ;
```

DEPARTMENT_ID	AVG(SALARY)
10	4400
20	9500
50	3500
60	6400
80	10033.3333
90	19333.3333
110	10150
	7000

8 rows selected

ففى المثال السابق يقوم بحساب المتوسط الحسابى للمرتبات لكل قسم على حدة. وذلك باستخدام Group by .
لاحظ استخدام نفس العمود بجملة SELECT فى جملة GROUP BY .
و ليس من الضروري تواجد اسم العمود المستخدم بجملة GROUP BY بجملة SELECT كما فى المثال التالى :

```
SELECT    AVG(salary)
FROM      employees
GROUP BY department_id ;
```

AVG(SALARY)
4400
9500
3500
6400
10033.3333
19333.3333
10150
7000

يمكن استخدام GROUP BY مع أكثر من عمود.
يجب ذكر جميع أسماء الأعمدة المدرجة في جملة SELECT بجملة GROUP By.
عند إغفال ذكر أحد الأعمدة في عبارة GROUP BY فتظهر رسالة خطأ ولا يتم تنفيذ الآمر.
مثال على ذلك:

Enter statements:

```
SELECT department_id dept_id,job_id,sum(salary) from employees
group by department_id
```

Execute | Save Script | Clear Screen | Cancel

```
SELECT department_id dept_id,job_id,sum(salary) from employees
                                *

ERROR at line 1:
ORA-00979: not a GROUP BY expression
```

نلاحظ ان هناك خطا عند استخدام جملة GROUP BY وذلك لعدم ادراج JOB_ID داخل فقرة GROUP BY .

استخدام GROUP BY مع مجموعة من الأعمدة

```
SELECT   department_id dept_id, job_id, SUM(salary)
FROM     employees
GROUP BY department_id, job_id ;
```

DEPT_ID	JOB_ID	SUM(SALARY)
10	AD_ASST	4400
20	MK_MAN	13000
20	MK_REP	6000
50	ST_CLERK	11700
50	ST_MAN	5800
60	IT_PROG	19200
80	SA_MAN	10500
80	SA_REP	19600
90	AD_PRES	24000
90	AD_VP	34000
110	AC_ACCOUNT	8300
110	AC_MGR	12000
	SA_REP	7000

13 rows selected.

استعلامات غير صحيحة تستخدم Group Function
أي عمود أو تعبير في قائمة SELECT يجب أن يكون بفقرة GROUP BY.

```
SELECT department_id, COUNT (last_name)
FROM employees;
```

```
SELECT department_id, COUNT (last_name)
*
ERROR at line 1:
ORA-00937: not a single-group group function
```

ولمعالجة الخطأ السابق يجب ادراج فقرة GROUP BY متضمنة العمود الذى فى جملة SELECT كما فى المثال التالى .

Enter statements:

```
SELECT department_id, COUNT(last_name)
FROM employees
GROUP BY department_id
```

DEPARTMENT_ID	COUNT(LAST_NAME)
10	1
20	2
30	6
40	1
50	45
60	5
70	1
80	34
90	3
100	6
110	2
	1

استخدام Having لتحديد (شرط) على المجموعات

تستخدم HAVING لامكانية استخدام شرط مع Group Functions.
حيث لايمكن استخدام WHERE كشرط مع Group Functions .
كما سوف نرى فى المثال التالى:

```
SELECT    department_id, AVG(salary)
FROM      employees
WHERE     AVG(salary) > 8000
GROUP BY department_id;
```

```
WHERE  AVG(salary) > 8000
       *
ERROR at line 3:
ORA-00934: group function is not allowed here
```

لا يمكن استخدام WHERE مع Group Functions .
حيث ان where تعمل على تحديد الصفوف قبل تجميعها فى شكل مجموعات . لذلك نستخدم Having وهى تعمل داخل صفوف لكل مجموعة على حدى ويمكن استخدام Group Functions داخلها. كما يلى:

```
SELECT    department_id, AVG(salary)
FROM      employees
HAVING    AVG(salary) > 8000
GROUP BY department_id;
```

DEPARTMENT_ID	AVG(SALARY)
20	9500
80	10033.3333
90	19333.3333
110	10150

```
SELECT department_id, COUNT(last_name)
FROM employees
where
department_id=10
GROUP BY department_id
having avg(salary)>10
order by department_id
```

DEPARTMENT_ID	COUNT(LAST_NAME)
10	1

مثال أخر:

```
SELECT   department_id, MAX(salary)
FROM     employees
GROUP BY department_id
HAVING   MAX(salary)>10000 ;
```

DEPARTMENT_ID	MAX(SALARY)
20	13000
80	11000
90	24000
110	12000

لا يمكن استخدام WHERE لمنع مجموعة من المجموعات الناتجة حيث أن فقرة WHERE تستخدم لمنع الصفوف الفردية وليس صفوف المجموعة. ويتم استخدام عبارة HAVING بدلا من عبارة WHERE وذلك لإظهار مجموعات من البيانات دون الأخرى كما في المثال حيث لم تظهر المجموعة المستثناة وهي الإدارة 30 حيث انها لم تحقق الشرط و هو أن يكون متوسط مرتبها أكبر من 10000.

ملحوظة:

لايمكن استخدام الاسم المستعار (Alias) مع HAVING . كما فى المثال التالى:

```
Select job, max (sal) "MAX_SAL"
 From EMP
 Group by job
 Having MAX_SAL>100;
```

```
  Having MAX_SAL>100
         *

ERROR at line 4:
ORA-00904: "MAX_SAL": invalid identifier
```

ولمعالجة المثال السابق يجب عدم استخدام الاسم المستعار فى فقرة HAVING كما يلى:

```
Select job , MAX(SAL)  "MAX_SAL"
 From EMP
 Group by job
 Having MAX(SAL) >100;
```

JOB	MAX_SAL
ANALYST	3000
CLERK	1300
MANAGER	2975
PRESIDENT	5000
SALESMAN	1600

امكانية استخدام اكثر من FUNCTIONS كما يلى
فالمثال التالى يريد عرض أكبر متوسط للمرتبات بالنسبة للاقسام :

```
SELECT   MAX(AVG(salary))
FROM     employees
GROUP BY department_id;
```

MAX(AVG(SALARY))
19333.3333

فهنا تم حساب المتوسط لكل إدارة ثم بعد ذلك تم عرض المتوسط الأكبر.