Reporting Aggregated Data Using the Group Functions

Group functions introduction

What are Group Functions?
Group functions operate on sets of rows to give one result per group.

Types of Group Functions

Function	Description
AVG([DISTINCT|ALL] n)	Average value of n, ignoring null values
COUNT	Number of rows, where expr evaluates to something other than null (count all selected rows using *, including duplicates and rows with nulls)
MAX([DISTINCT|ALL] expr)	Maximum value of expr, ignoring null values
MIN([DISTINCT|ALL] expr)	Minimum value of expr, ignoring null values
STDDEV([DISTINCT|ALL]n)	Standard deviation of n, ignoring null values
SUM([DISTINCT|ALL] n)	Sum values of n, ignoring null values
LISTAGG	Orders data within each group specified in the ORDER BY clause and then concatenates the values of the measure column
VARIANCE([DISTINCT|ALL]n)	Variance of n, ignoring null values

The group function is placed after the SELECT keyword. You may have multiple group functions separated by commas.

Syntax:

group_function([DISTINCT|ALL] expr)

Guidelines for using the group functions:

- DISTINCT makes the function consider only nonduplicate values; ALL makes it consider every value, including duplicates. The default is ALL and, therefore, does not need to be specified.
- The data types for the functions with an expr argument may be CHAR, VARCHAR2, NUMBER, or DATE.
- All group functions ignore null values. To substitute a value for null values, use the NVL, NVL2, COALESCE, CASE, or DECODE functions.

Group functions (SUM, COUNT, MAX, MIN, AVG and more)

```
SELECT MAX(SALARY), MIN(SALARY)
FROM EMPLOYEES;
```

You can use max and min with varchar

```
SELECT MAX(first_name), MIN(first_name)
FROM EMPLOYEES;
```

You can use max and min with dates

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

```
SELECT SUM(salary), AVG(salary)
FROM EMPLOYEES;
```

Note: You cannot use SUM and AVG with varchar or dates

```
SELECT COUNT(*) FROM EMPLOYEES;
```

```
SELECT COUNT(1) FROM EMPLOYEES; -- this is equlas to COUNT(*)
```

```
SELECT COUNT(DEPARTMENT_ID)
FROM EMPLOYEES; -- null is not counted
```

COUNT(DEPARTMENT_ID)
106

```
SELECT COUNT(DISTINCT DEPARTMENT_ID) FROM EMPLOYEES;
```

COUNT(DISTINCT DEPARTMENT_ID)
11

You can handle null values using NVL function

```
SELECT COUNT(NVL(COMMISSION_PCT,0)) FROM EMPLOYEES;
```

```
SELECT COUNT(EMPLOYEE_ID)
FROM EMPLOYEES
WHERE DEPARTMENT_ID = 30;
```

LISTAGG Function

```
SELECT first_name
FROM EMPLOYEES
WHERE DEPARTMENT_ID = 30
ORDER BY first_name;
```

FIRST_NAME
Alexander
Den
Guy
Karen

Shelli
Sigal

```
SELECT LISTAGG(FIRST_NAME, ', ')
    WITHIN GROUP (ORDER BY FIRST_NAME) "Emp_list"
FROM EMPLOYEES
WHERE department_id = 30;
```

Emp_list
Alexander, Den , Guy, Karen, Shelli, Sigal

Group by Clause, Having Clause

```
SELECT SUM(SALARY), AVG(SALARY)
FROM EMPLOYEES;
```

SUM(SALARY)	AVG(SALARY)
691416	6461.81177

The following statement will give an error

```
SELECT DEPARTMENT_ID, SUM(SALARY)
FROM EMPLOYEES;
```

ORA-00937: not a single-group function
00937.00000 - "not a single-group group function"

To fix this error, we need to use GROUP BY function

```
SELECT DEPARTMENT_ID, SUM(SALARY)
FROM EMPLOYEES
GROUP BY DEPARTMENT_ID;
```

```
SELECT DEPARTMENT_ID, JOB_ID, SUM(SALARY)
FROM EMPLOYEES
GROUP BY DEPARTMENT_ID, JOB_ID
ORDER BY 1, 2;
```

DEPARTMENT_ID	JOB_ID	SUM(SALARY)
10	AD ASST	4400
20	MM MAN	13000
20	MK REP	6000
30	PU CLERK	13900
30	PU MAN	11000
40	HR REP	6500

The following statement will retrieve an error since the JOB_ID should be in GROUP_BY expression

```
SELECT DEPARTMENT_ID, JOB_ID, SUM(SALARY)
FROM EMPLOYEES
GROUP BY DEPARTMENT_ID
ORDER BY 1, 2;
```

ORA-00979: not a GROUP BY expression
00979, 00000 - "not a GROUP BY expression"
You cannot make GROUP BY expression using alias

```
SELECT DEPARTMENT_ID d, SUM(SALARY)
FROM EMPLOYEES
GROUP BY d;
```

ORA-00904: "D": invalid identifier
00904, 00000 - "%s:invalid identifier"

However, you can make order using alias

```
SELECT DEPARTMENT_ID d, SUM(SALARY)
FROM EMPLOYEES
GROUP BY DEPARTMENT_ID
ORDER BY d;
```

Sequence: WHERE → GROUP BY → ORDER BY

```
SELECT DEPARTMENT_ID, SUM(SALARY)
FROM EMPLOYEES
WHERE DEPARTMENT_ID > 30
GROUP BY DEPARTMENT_ID
ORDER BY DEPARTMENT_ID;
```

Important Note: You cannot use WHERE statement to restrict groups

```
SELECT DEPARTMENT_ID, SUM(SALARY)
FROM EMPLOYEES
WHERE SUM(SALARY) > 156400 -- this is not correct, you should use HAVING
GROUP BY DEPARTMENT_ID
ORDER BY DEPARTMENT_ID;
```

ORA-00934: group function is not allowed here
00934, 00000 - "group function is not allowed here"

```
SELECT DEPARTMENT_ID, SUM(SALARY)
FROM EMPLOYEES
GROUP BY DEPARTMENT_ID
HAVING SUM(SALARY) > 156400
ORDER BY DEPARTMENT_ID;
```

We can use HAVING statement before GROUP BY statement, but it is not recommended

```
SELECT DEPARTMENT_ID, SUM(SALARY)
FROM EMPLOYEES
HAVING SUM(SALARY) > 156400
GROUP BY DEPARTMENT_ID
ORDER BY DEPARTMENT_ID;
```

You can make nested group functions

```
SELECT DEPARTMENT_ID, SUM(SALARY)
FROM EMPLOYEES
GROUP BY DEPARTMENT_ID
ORDER_BY 1;
```

DEPARTMENT_ID	SUM(SALARY)
10	4400
20	19000

30	24900
40	6500
50	156400
60	28800
70	10000
80	304500
90	58000
100	51608
110	20308
(null)	7000

```
SELECT MAX(SUM(SALARY)) -- only 2 group functions can be nested
FROM EMPLOYEES
GROUP BY DEPARTMENT_ID
ORDER BY 1;
```

MAX(SUM(SALARY))
304500