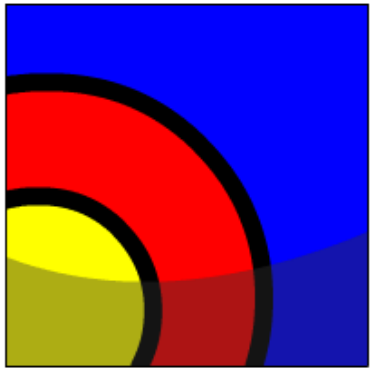
# COUNT, DISTINCT, NVL

# What Will I Learn?

**In this lesson, you will learn to:**



- Construct and execute a SQL query using the COUNT group function
- Use DISTINCT and the NVL function with group functions

# Why Learn It?

Being able to aggregate data using SQL functions enables businesses to do calculations that would otherwise have to be done by hand.


Remember the example of having to count all the students in your school? A daunting task! There just aren't enough hands to accomplish it manually.

Fortunately, the SQL group functions can easily process these types of requests.

# Tell Me / Show Me

## COUNT

COUNT(expression) returns the number of non-null values in the expression column

```
SELECT COUNT (YEAR)
FROM d_cds
WHERE year < 2001;
```

| COUNT (YEAR) |
|---|
| 5 |

COUNT(DISTINCT expression) returns the number of unique non-null values in the expression column.

```
SELECT COUNT (DISTINCT year)
FROM d_cds
WHERE year < 2001;
```

| COUNT (DISTINCT YEAR) |
|---|
| 4 |

# Tell Me / Show Me

## COUNT

Why are null values returned in the query shown? There are six comments listed, but the count function returned only five. Why?

```
SELECT comments
FROM d_play_list_items;
```

| COMMENTS |
|---|
| Play late |
| (null) |
| Play early |
| Play after cake cutting |
| Play first |
| Play for the father |

Because COUNT ignores the null value in the column.

```
SELECT COUNT(comments)
FROM d_play_list_items;
```

| COUNT (COMMENTS) |
|---|
| 5 |

# Tell Me / Show Me

## COUNT

COUNT(*) returns the number of rows in a table that satisfy the criteria of the SELECT statement.

For example, to find out how many of DJs on Demand's CDs were produced before 2001, COUNT can be used in the SELECT statement.

We use COUNT(*) when we want to make sure that we count all the rows, including those that may have nulls in one or more columns.

```
SELECT COUNT (*)
FROM d_cds
WHERE year < 2001;
```

| COUNT (*) |
| --- |
| 5 |

# Tell Me / Show Me

## DISTINCT

The keyword DISTINCT is used to return only nonduplicate values or combinations of nonduplicate values in a query.

Examine the query on the right. Without using the keyword DISTINCT, the query returned all of the year values from the DJs on Demand D_CDS table.

**SELECT year as 'CD Year'**
**FROM d_cds;**

| CD Year |
|---|
| 1997 |
| 2000 |
| 2002 |
| 1999 |
| 2000 |
| 2001 |
| 1998 |
| 2004 |

# Tell Me / Show Me

To eliminate duplicate rows, use the DISTINCT keyword as shown here.

Using the DISTINCT keyword returned all the CD years exactly once, with no duplicate values.

**SELECT DISTINCT year AS 'CD Year'**
**FROM d_cds;**

| CD Years |
|---|
| 1997 |
| 1998 |
| 1999 |
| 2000 |
| 2001 |
| 2002 |
| 2004 |

# Tell Me / Show Me

The keyword DISTINCT, when used in a query selecting more than one column, will return nonduplicate combinations of the columns. Examine the two results sets shown here. Can you tell which query used the DISTINCT keyword?

| YEAR | TITLE |
|---|---|
| 1997 | The Celebrants Live in Concert |
| 1998 | Graduation Songbook |
| 1999 | Songs from My Childhood |
| 2000 | Cape Diem |
| 2000 | Party Music for All Occasions |
| 2001 | Here Comes the Bride |
| 2002 | Back to the Shire |
| 2004 | Whirled Peas |

**SELECT DISTINCT year, title**
**FROM d_cds;**

| YEAR | TITLE |
|---|---|
| 1997 | The Celebrants Live in Concert |
| 2000 | Party Music for All Occasions |
| 2002 | Back to the Shire |
| 1999 | Songs from My Childhood |
| 2000 | Carpe Diem |
| 2001 | Here Comes the Bride |
| 1998 | Graduation Songbook |
| 2004 | Whirled Peas |

In this case, it's hard to tell, isn't it? The results set on the top was returned using the DISTINCT keyword. In both examples, there are no duplicate combinations of year and title even though there are duplicate years.

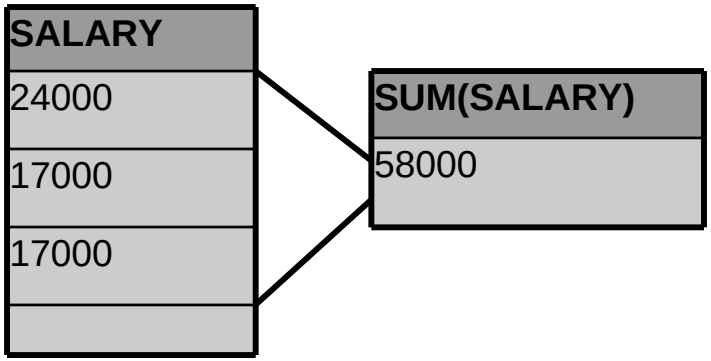
# Tell Me / Show Me

## Using DISTINCT

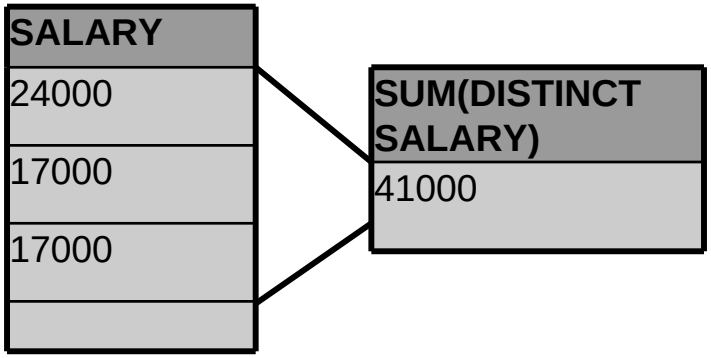
The keyword DISTINCT can be used with all group functions. Using DISTINCT makes the function consider only nonduplicate values.

Why do the two statements on the right produce different results?

```
SELECT SUM(salary)
FROM employees
WHERE department_id = 90;
```




```
SELECT SUM(DISTINCT salary)
FROM employees
WHERE department_id = 90;
```



# Tell Me / Show Me

When using DISTINCT with a group function such as COUNT, the result set will return the number of nonduplicate column values.

How many different jobs do our employees do ?

How many different salaries are there ?

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

| COUNT (DISTINCT job_id) |
|---|
| 12 |

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

| COUNT (DISTINCT salary) |
|---|
| 18 |

# Tell Me / Show Me

## NVL

Sometimes it is desirable to include null values in group functions. For example, knowing the average number of customer orders served each day could be used to judge how much food to order each month. Some days the restaurant is closed and there are no customers, but the owner has found that computing the average including every day is a better indicator than just counting the days with customers. The SELECT statement to include null values could be written starting with:

```
SELECT AVG(NVL(customer_orders, 0))
```

# Tell Me / Show Me

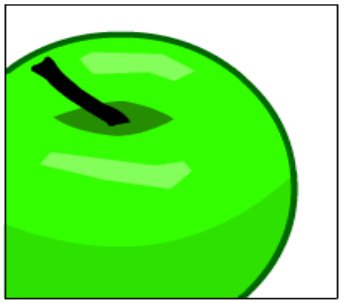
**Terminology**

Key terms used in this lesson include:

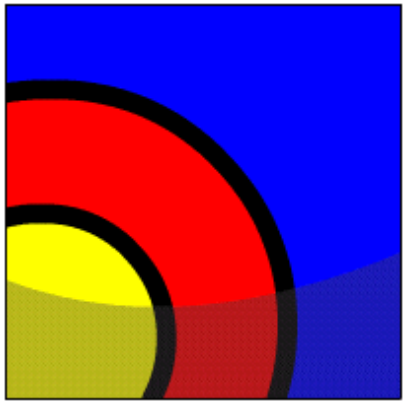
COUNT (expression)

COUNT (DISTINCT expression)

DISTINCT



# Summary



**In this lesson you have learned to:**

- Construct and execute a SQL query using the COUNT group function
- Use DISTINCT and the NVL function with group functions

# Summary

## Practice Guide

The link for the lesson practice guide can be found in the course resources in Section 0.