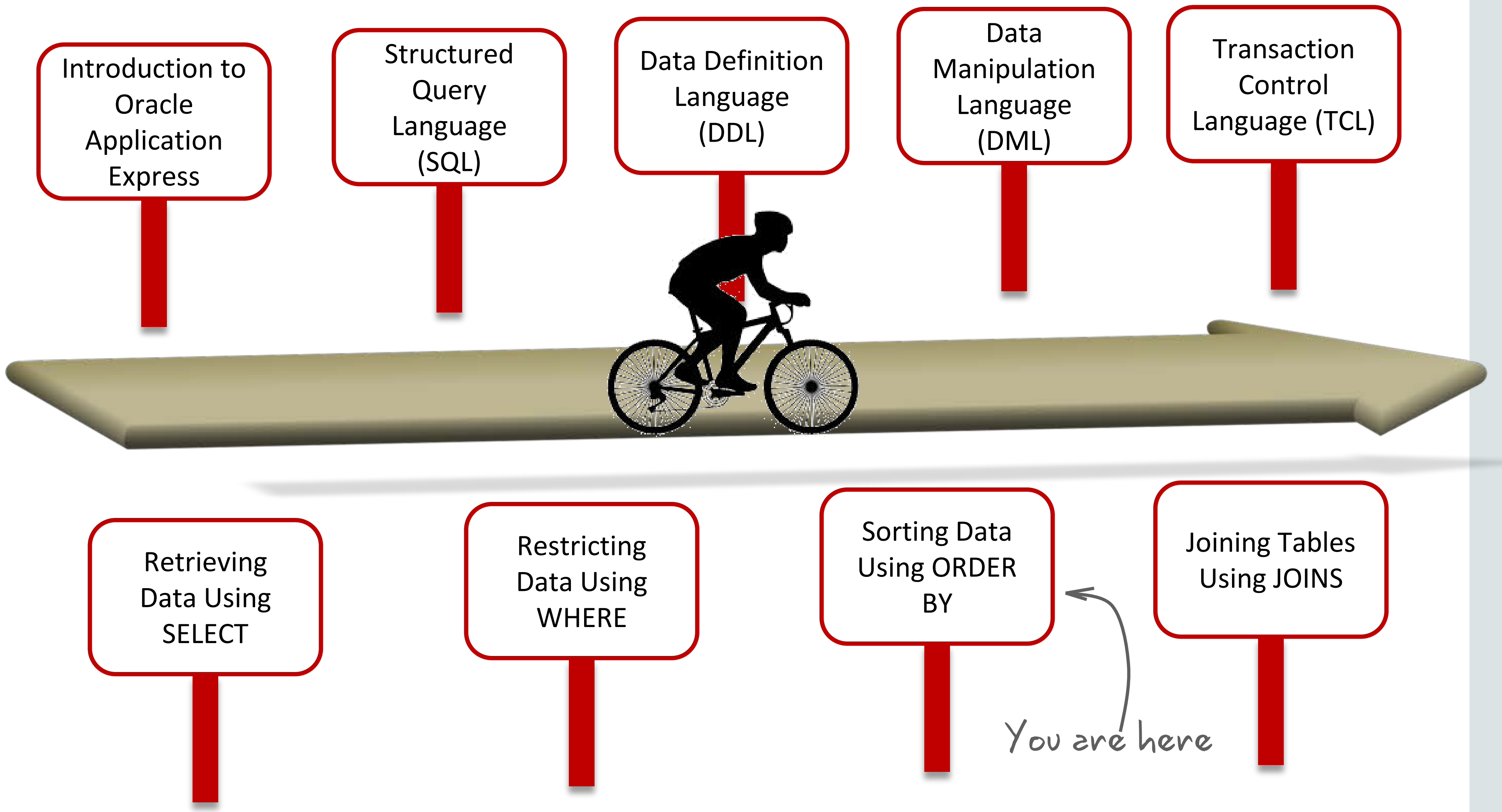
ORACLE®

ACADEMY

Database Foundations

6-8
Sorting Data Using ORDER BY

Roadmap


Introduction to Oracle Application Express
Structured Query Language (SQL)
Data Definition Language (DDL)
Data Manipulation Language (DML)
Transaction Control Language (TCL)
Retrieving Data Using SELECT
Restricting Data Using WHERE
Sorting Data Using ORDER BY
Joining Tables Using JOINS
You are here

Objectives

This lesson covers the following objectives:

- Use the `ORDER BY` clause to sort SQL query results
- Identify the correct placement of the `ORDER BY` clause within a `SELECT` statement
- Order data and limit row output by using the SQL `row_limiting_clause`
- Use substitution variables in the `ORDER BY` clause

Using the ORDER BY Clause

- Sort the retrieved rows with the ORDER BY clause:
  - ASC: Ascending order (default)
  - DESC: Descending order
- The ORDER BY clause comes last in the SELECT statement:

```
SELECT   last_name, job_id, department_id, hire_date
FROM     employees
ORDER BY hire_date ;
```

Sorting

- Sorting in descending order:

```
SELECT   last_name, job_id, department_id, hire_date
FROM     employees
ORDER BY hire_date DESC ;
```

1

- Sorting by column alias:

```
SELECT employee_id, last_name, salary*12 annsal
FROM   employees
ORDER BY annsal ;
```

2

Sorting

- Sorting by using the column's numeric position:

```
SELECT    last_name, job_id, department_id, hire_date
FROM      employees
ORDER BY  3;
```

3

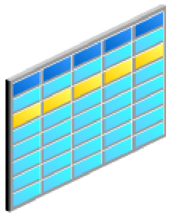
- Sorting by multiple columns:

```
SELECT last_name, department_id, salary
FROM   employees
ORDER BY department_id, salary DESC;
```

4

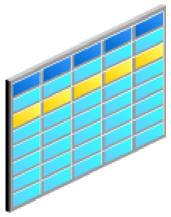
SQL `row_limiting_clause`

- The `row_limiting_clause` allows you to limit the rows that are returned by the query.
- Queries that order data and then limit row output are widely used and are often referred to as Top-N queries.
- You can specify the number of rows or percentage of rows to return with the `FETCH_FIRST` keywords.



SQL `row_limiting_clause`

- You can use the `OFFSET` keyword to specify that the returned rows begin with a row after the first row of the full result set.
- The `WITH TIES` keyword includes additional rows with the same ordering keys as the last row of the row-limited result set (you must specify `ORDER BY` in the query).
- You can specify the `ORDER BY` clause to ensure a deterministic sort order.



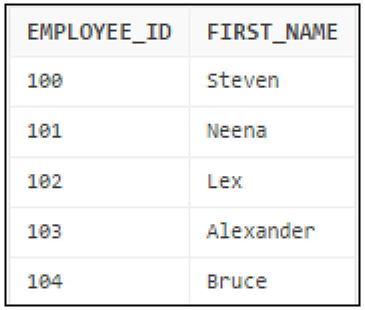
Using the SQL `row_limiting_clause` in a Query

You can specify the `row_limiting_clause` in the SQL SELECT statement by placing it after the ORDER BY clause.

```
subquery::=
{ query_block
  | subquery { UNION [ALL] | INTERSECT | MINUS } subquery
[ { UNION [ALL] | INTERSECT | MINUS } subquery ]...
  | ( subquery )
{
[ order_by_clause ]
[OFFSET offset { ROW | ROWS }]
[FETCH { FIRST | NEXT } [{ row_count | percent PERCENT }]
 { ROW | ROWS }
 { ONLY | WITH TIES }]
```

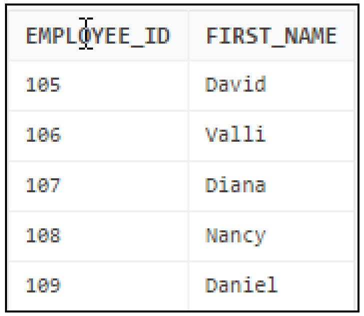
SQL row_limiting_clause: Example

```
SELECT employee_id, first_name
FROM employees
ORDER BY employee_id
FETCH FIRST 5 ROWS ONLY;
```



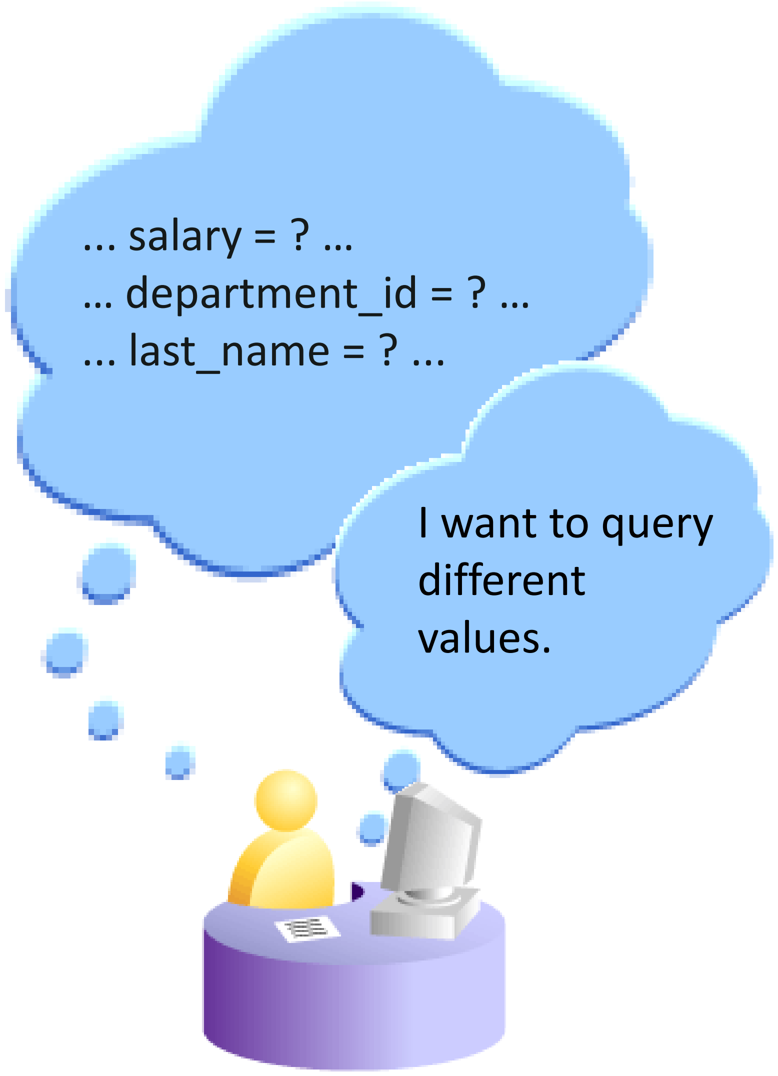
EMPLOYEE_ID	FIRST_NAME
100	Steven
101	Neena
102	Lex
103	Alexander
104	Bruce

```
SELECT employee_id, first_name
FROM employees
ORDER BY employee_id
OFFSET 5 ROWS FETCH NEXT 5 ROWS ONLY;
```



EMPLOYEE_ID	FIRST_NAME
105	David
106	Valli
107	Diana
108	Nancy
109	Daniel

Substitution Variables



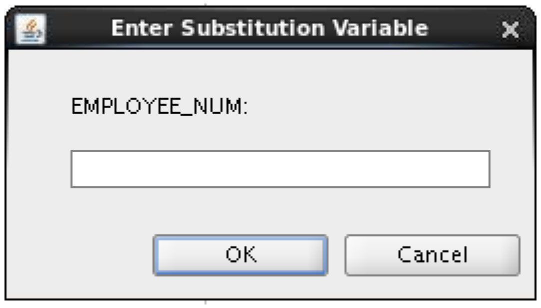
Substitution Variables

- Use substitution variables to temporarily store values with single-ampersand (&) and double-ampersand (&&) substitutions.
- Use substitution variables to supplement the following:
  - WHERE conditions
  - ORDER BY clauses
  - Column expressions
  - Table names
  - Entire SELECT statements

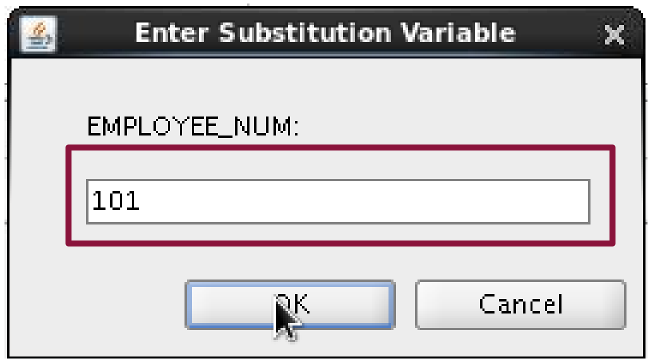
Using the Single-Ampersand Substitution Variable

Use a variable prefixed with an ampersand (&) to prompt the user for a value:

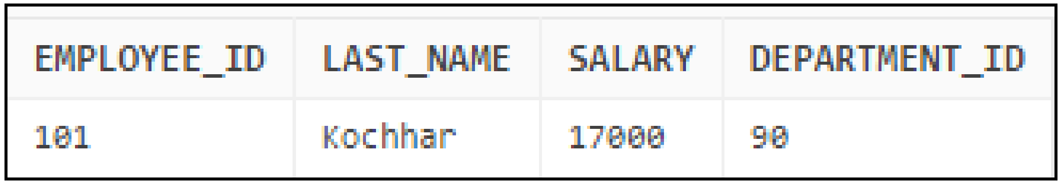
```
SELECT employee_id, last_name, salary, department_id
FROM   employees
WHERE  employee_id = &employee_num ;
```



Using the Single-Ampersand Substitution Variable



EMPLOYEE_NUM: 101

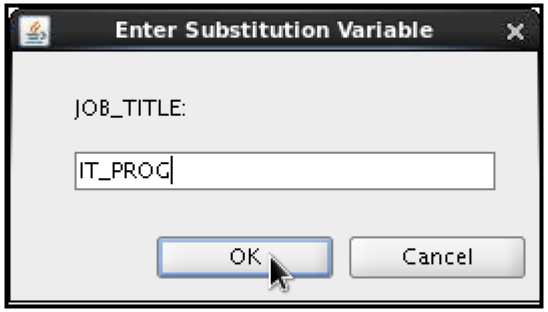


EMPLOYEE_ID	LAST_NAME	SALARY	DEPARTMENT_ID
101	Kochhar	17000	90

Character and Date Values with Substitution Variables

Use single quotation marks for date and character values:

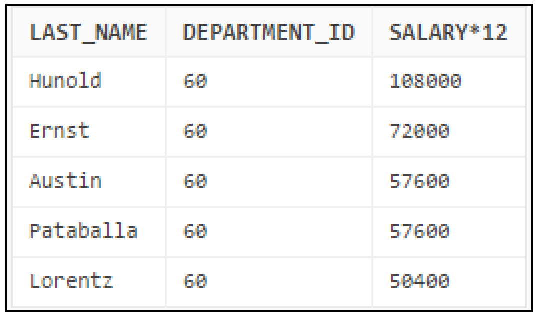
```
SELECT last_name, department_id, salary*12
FROM   employees
WHERE  job_id = '&job_title' ;
```



Enter Substitution Variable

JOB_TITLE:

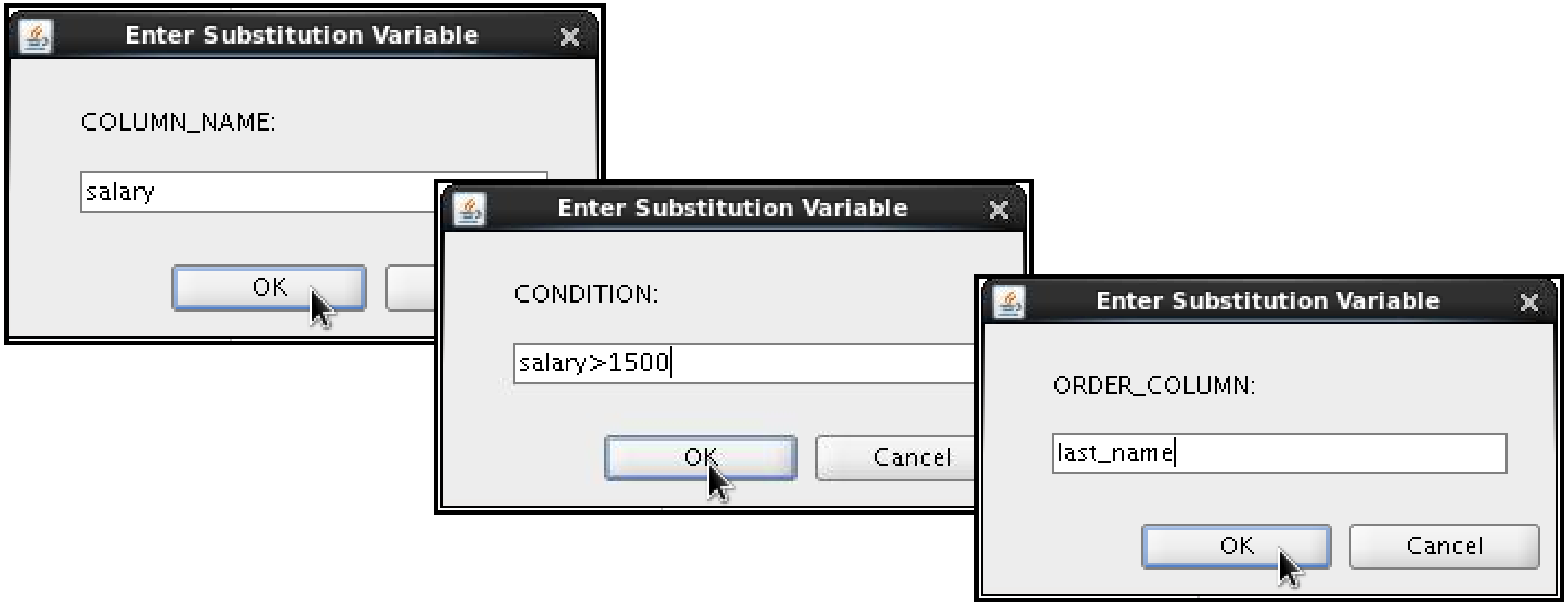
IT_PROG



LAST_NAME	DEPARTMENT_ID	SALARY*12
Hunold	60	108000
Ernst	60	72000
Austin	60	57600
Pataballa	60	57600
Lorentz	60	50400

Specifying Column Names, Expressions, and Text

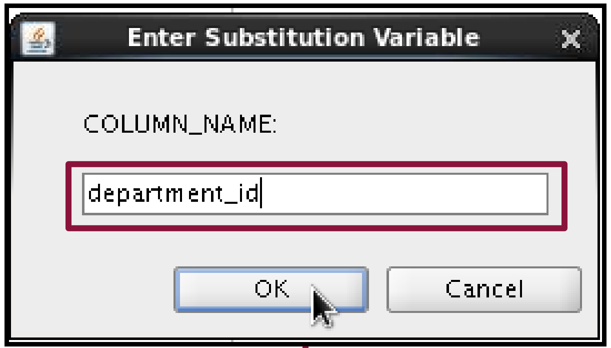
```
SELECT employee_id, last_name, job_id, &column_name
FROM   employees
WHERE  &condition
ORDER BY &order_column ;
```


Enter Substitution Variable
COLUMN_NAME:
salary
OK
Enter Substitution Variable
CONDITION:
salary>1500
OK
Cancel
Enter Substitution Variable
ORDER_COLUMN:
last_name
OK
Cancel

Using the Double-Ampersand Substitution Variable

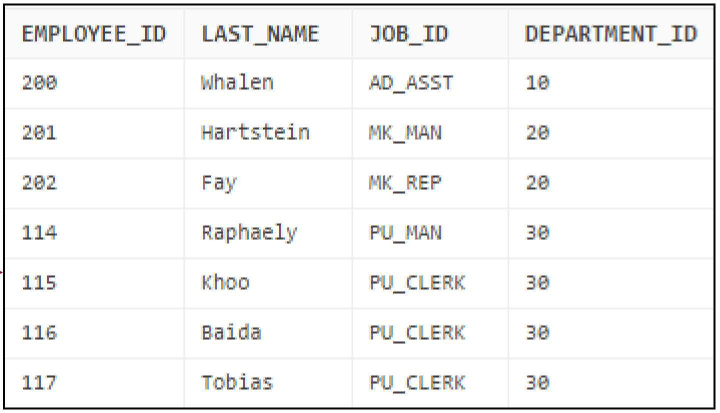
Use double ampersands (&&) if you want to reuse the variable value without prompting the user each time:

```
SELECT    employee_id, last_name, job_id, &&column_name
FROM      employees
ORDER BY &column_name ;
```



Enter Substitution Variable

COLUMN_NAME: department_id



EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
200	Whalen	AD_ASST	10
201	Hartstein	MK_MAN	20
202	Fay	MK_REP	20
114	Raphaely	PU_MAN	30
115	Khoo	PU_CLERK	30
116	Baida	PU_CLERK	30
117	Tobias	PU_CLERK	30

Using the DEFINE Command

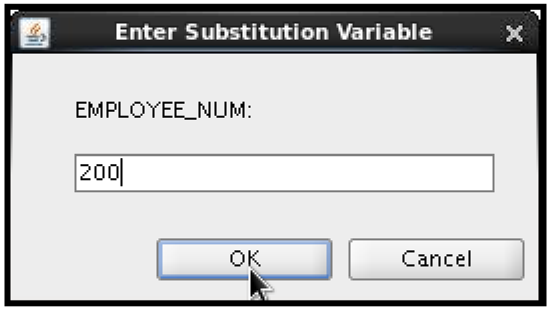
- Use the DEFINE command to create and assign a value to a variable.
- Use the UNDEFINE command to remove a variable.

```
DEFINE employee_num = 200

SELECT employee_id, last_name, salary, department_id
FROM   employees
WHERE  employee_id = &employee_num ;

UNDEFINE employee_num
```

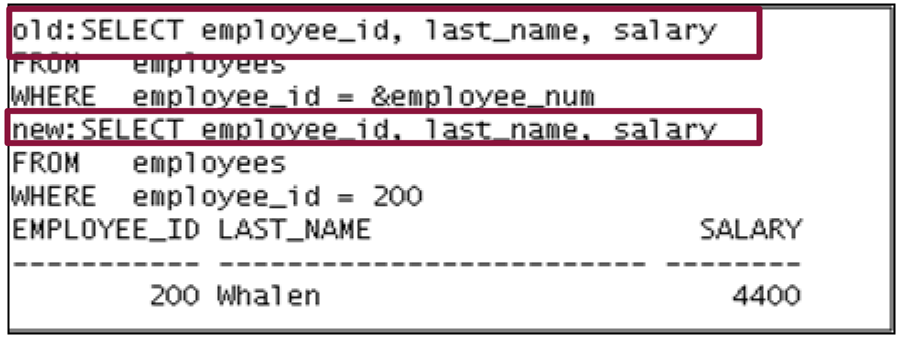
Using the VERIFY Command

Use the VERIFY command to toggle the display of the substitution variable before and after SQL Developer replaces substitution variables with values:

```
SET VERIFY ON
SELECT employee_id, last_name, salary
FROM   employees
WHERE  employee_id = &employee_num;
```



EMPLOYEE_NUM: 200

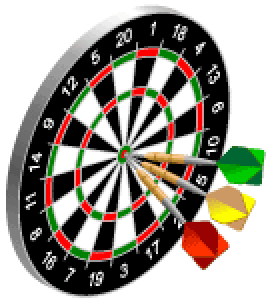


```
old:SELECT employee_id, last_name, salary
FROM   employees
WHERE  employee_id = &employee_num
new:SELECT employee_id, last_name, salary
FROM   employees
WHERE  employee_id = 200
EMPLOYEE_ID LAST_NAME                    SALARY
----------- ------------------------- ---------
        200 Whalen                         4400
```

Summary

In this lesson, you should have learned how to:

- Use the `ORDER BY` clause to sort SQL query results
- Identify the correct placement of the `ORDER BY` clause within a `SELECT` statement
- Order data and limit row output by using the SQL `row_limiting_clause`
- Use substitution variables in the `ORDER BY` clause



ORACLE® ACADEMY