2- Restricting and Sorting Data Objectives

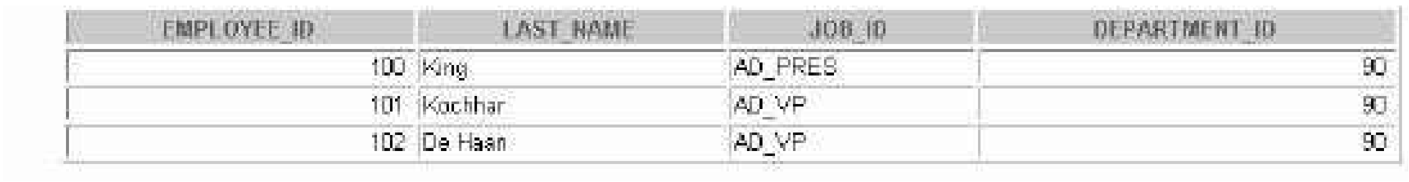
Objectives:
After completing this lesson, you should be able to do the following:
• Limit the rows retrieved by a query
• Sort the rows retrieved by a query

Limiting Rows Using a Selection

EMPLOYEES "retrieve all employees in department 90"
Limiting Rows Using a Selection

Limiting the Rows Selected

• Restrict the rows returned by using the WHERE clause.
SELECT *|{[DISTINCT] column|expression [alias],...} FROM table [WHERE condition(s)];
•The WHERE clause follows the FROM clause.



EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90

Using theWHERE Clause

SELECT employee_id, last_name, job_id, department_id
FROM employees
WHERE department_id = 90;

Character Strings and Dates

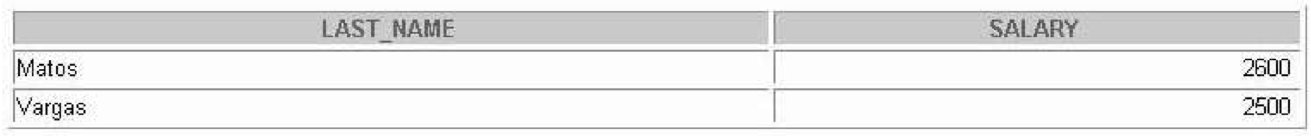
• Character strings and date values are enclosed in single quotation marks.
• Character values are case sensitive, and date values are format sensitive.
• The default date format is DD-MON-RR.

SELECT last_name, job_id, department_id FROM employees
WHERE last_name = 'Goyal';

Comparison Conditions

Operator	**Meaning**
=	**Equal to**
>	**Greater than**
>=	**Greater than or equal to**
<	**Less than**
<=	**Less than or equal to**
<>	**Not equal to**

Example
... WHERE hire_date='01-JAN-95'
... WHERE salary>=6000
... WHERE last_name='Smith'
An alias cannot be used in the WHERE clause.
Note: The symbol != and ^= can also represent the *not equal to* condition.



LAST_NAME	SALARY
Matos	2600
Vargas	2500

Using Comparison Conditions

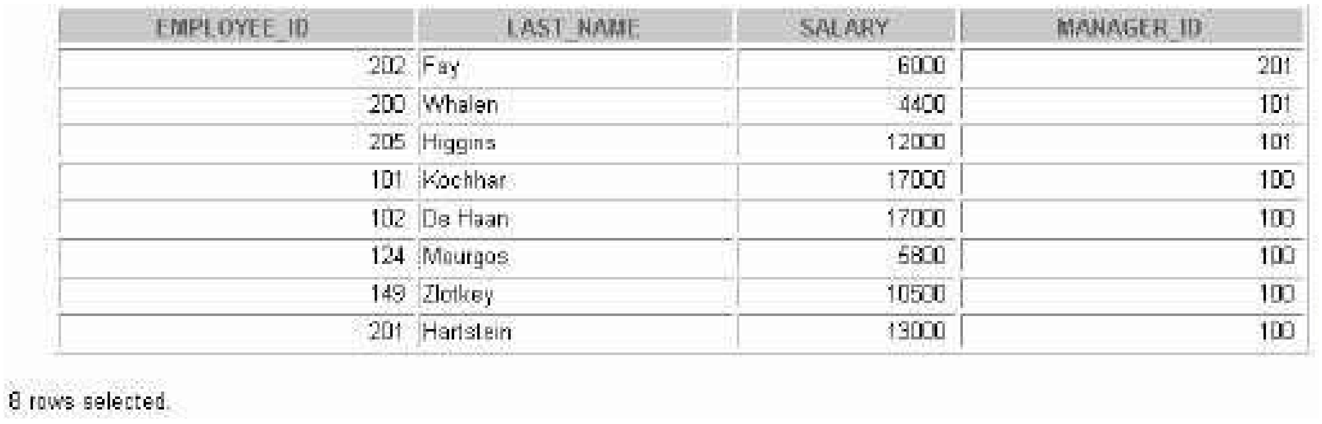
SELECT last_name, salary FROM employees
WHERE salary <= 3000;

Other Comparison Conditions

Operator	**Meaning**
BETWEEN ...AND...	Between two values (inclusive)
IN(set)	Match any of a list of values
LIKE	Match a character pattern
IS NULL	Is a null value

Using the BETWEEN Condition

Use the BETWEEN condition to display rows based on a range of values.
SELECT last_name, salary
FROM employees
WHERE salary BETWEEN 2500 AND 3500;
Lower limit Upper limit
The BETWEEN Condition



EMPLOYEE_ID	LAST_NAME	SALARY	MANAGER_ID
202	Fay	6000	201
200	Whalen	4400	101
205	Higgins	12000	101
101	Kochhar	17000	100
102	De Haan	17000	100
124	Mourgos	5800	100
149	Zlotkey	10500	100
201	Hartstein	13000	100

8 rows selected.

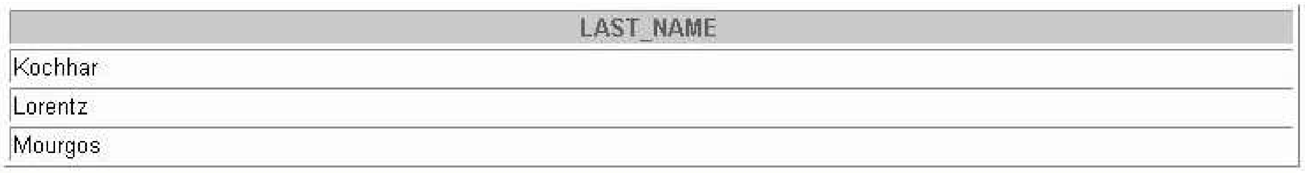
Using the IN Condition

Use the IN membership condition to test for values in
a list.

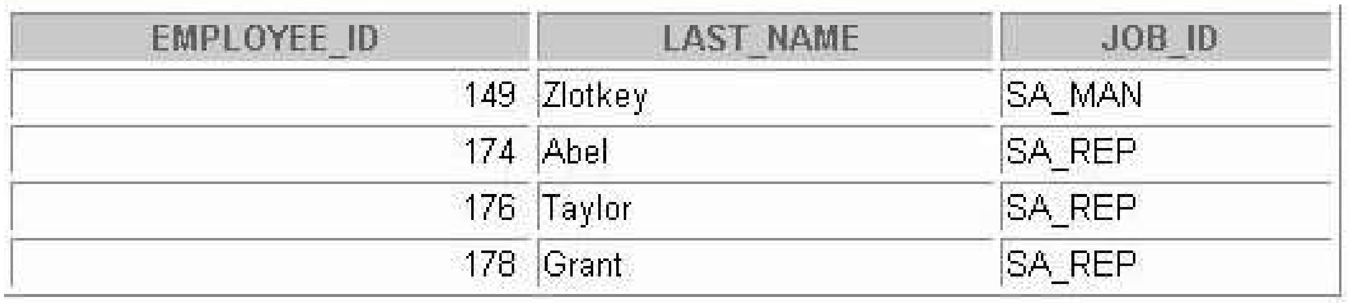
SELECT employee_id, last_name, salary, manager_id
FROM employees
WHERE manager_id IN (100, 101, 201);

Using the LIKE Condition

•Use the LIKE condition to perform wildcard
searches of valid search string values.
•Search conditions can contain either literal
characters or numbers:
– % denotes zero or many characters.
–_denotes one character.
SELECT first_name FROM employees WHERE first_name LIKE 'S%';



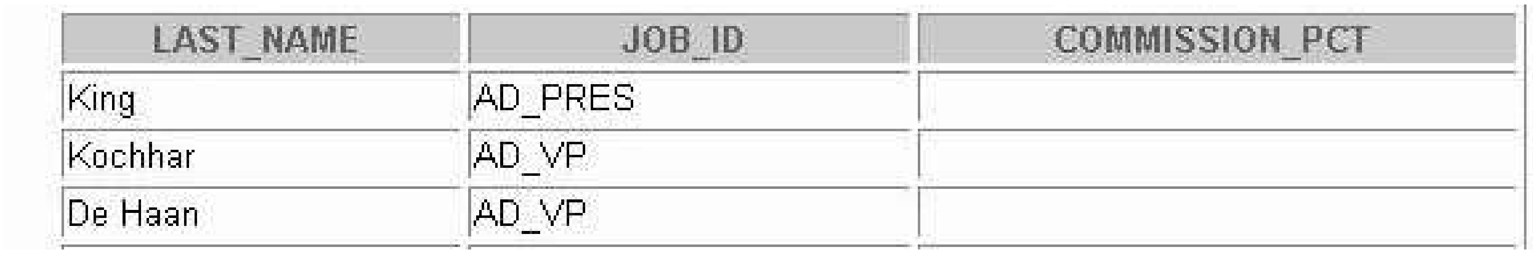
LAST_NAME
Kochhar
Lorentz
Mourgos



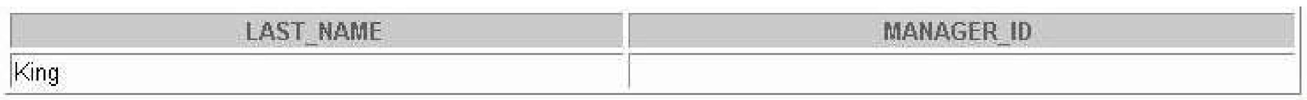
EMPLOYEE_ID	LAST_NAME	JOB_ID
149	Zlotkey	SA_MAN
174	Abel	SA_REP
176	Taylor	SA_REP
178	Grant	SA_REP

Using the LIKE Condition

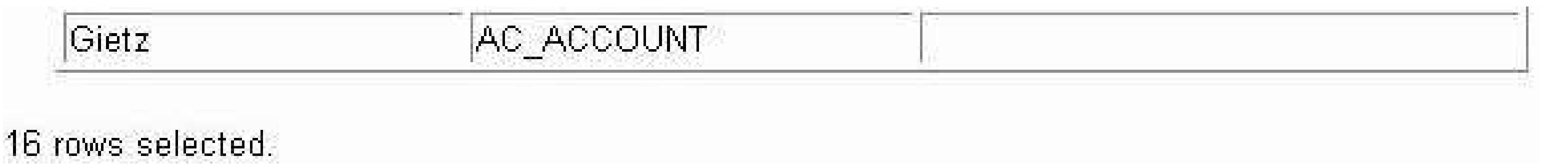
• You can combine pattern-matching characters.
SELECT last_name FROM employees
WHERE last_name LIKE '_o%';
• You can use the ESCAPE identifier to search for the
Actual % and _ symbols.



LAST_NAME	JOB_ID	COMMISSION_PCT
King	AD_PRES	
Kochhar	AD_VP	
De Haan	AD_VP	



LAST_NAME	MANAGER_ID
King	



Gietz	AC_ACCOUNT	

```
16 rows selected.
```

Using the NULL Conditions

Test for nulls with the IS NULL operator.

```
SELECT last_name, manager_id FROM   employees
WHERE  manager_id IS NULL;
```

Logical Conditions

Meaning	*Operator*
AND	Returns TRUE if *both* component conditions are true
OR	Returns TRUE If *either* component condition is true
NOT	Returns TRUE if the following condition is false

Using the AND Operator

AND requires both conditions to be true.

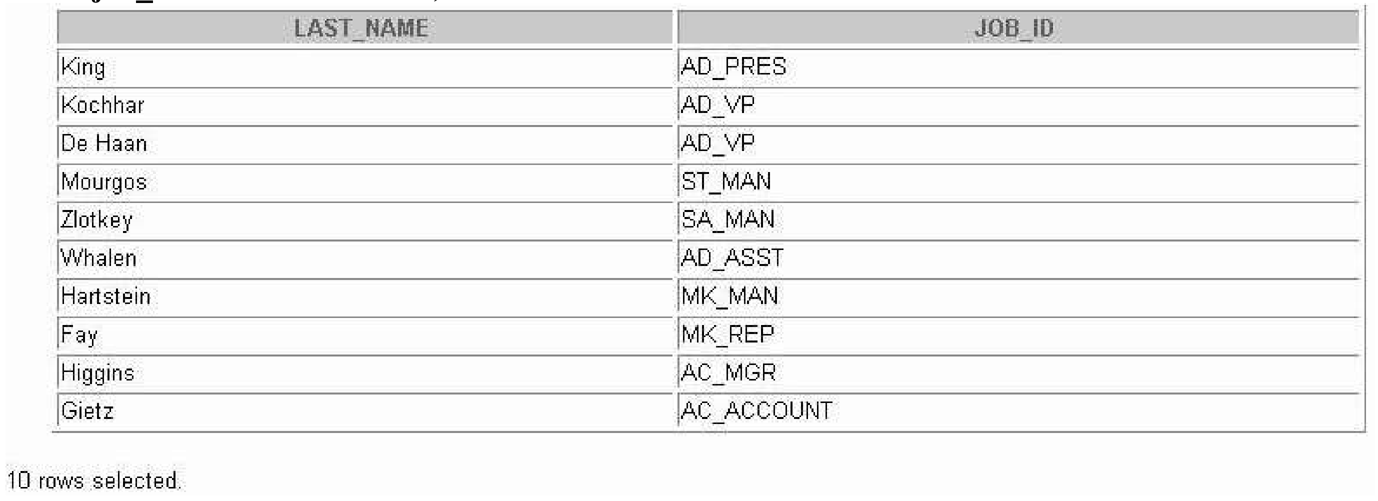
```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary >=10000
AND    job_id LIKE '%MAN%';
```

Using the OR Operator

OR requires either condition to be true.

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary >= 10000
```

OR job_id LIKE '%MAN%';



LAST_NAME	JOB_ID
King	AD_PRES
Kochhar	AD_VP
De Haan	AD_VP
Mourgos	ST_MAN
Zlotkey	SA_MAN
Whalen	AD_ASST
Hartstein	MK_MAN
Fay	MK_REP
Higgins	AC_MGR
Gietz	AC_ACCOUNT

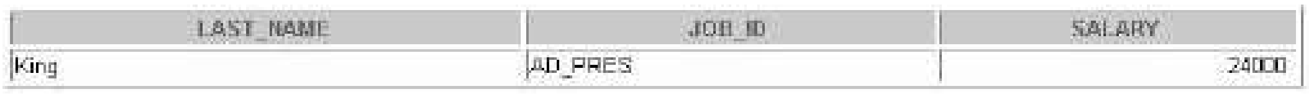
10 rows selected.

Using the NOT Operator

SELECT last_name, job_id
FROM employees
WHERE job_id NOT IN ('IT_PROG', 'ST_CLERK', 'SA_REP');

Rules of Precedence

Order Evaluated Operator
1 Arithmetic operators
2 Concatenation operator
3 Comparison conditions
4 IS [NOT] NULL , LIKE , [NOT] IN
5 [NOT] BETWEEN
6 NOT logical condition
7 AND logical condition
8 OR logical condition
Override rules of precedence by using parentheses**.**
SELECT last_name, job_id, salary
FROM employees
WHERE job_id = 'SA_REP'
OR job_id = 'AD_PRES'
AND salary > 15000;

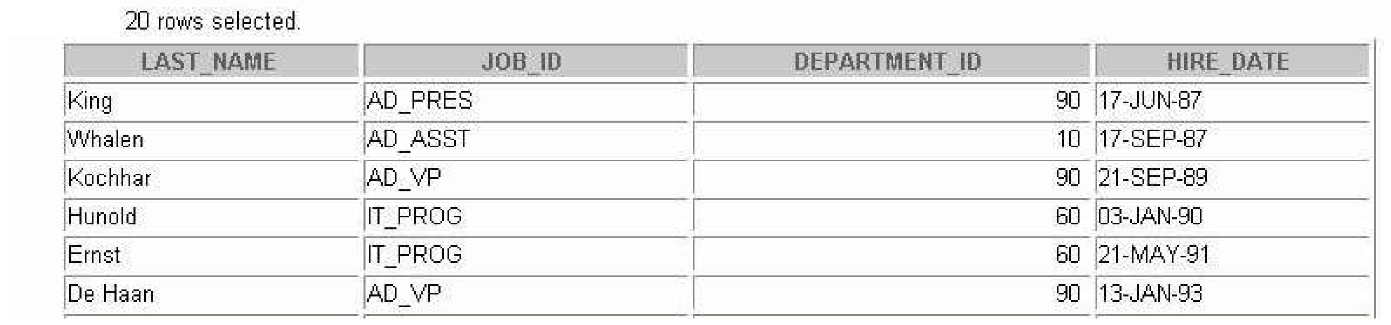


LAST_NAME	JOB_ID	SALARY
King	AD_PRES	24000

Use parentheses to force priority.

SELECT last_name, job_id, salary
FROM employees
WHERE (job_id = 'SA_REP'
OR job_id = 'AD_PRES')

AND salary > 15000;



20 rows selected.

LAST_NAME	JOB_ID	DEPARTMENT_ID	HIRE_DATE
King	AD_PRES	90	17-JUN-87
Whalen	AD_ASST	10	17-SEP-87
Kochhar	AD_VP	90	21-SEP-89
Hunold	IT_PROG	60	03-JAN-90
Ernst	IT_PROG	60	21-MAY-91
De Haan	AD_VP	90	13-JAN-93

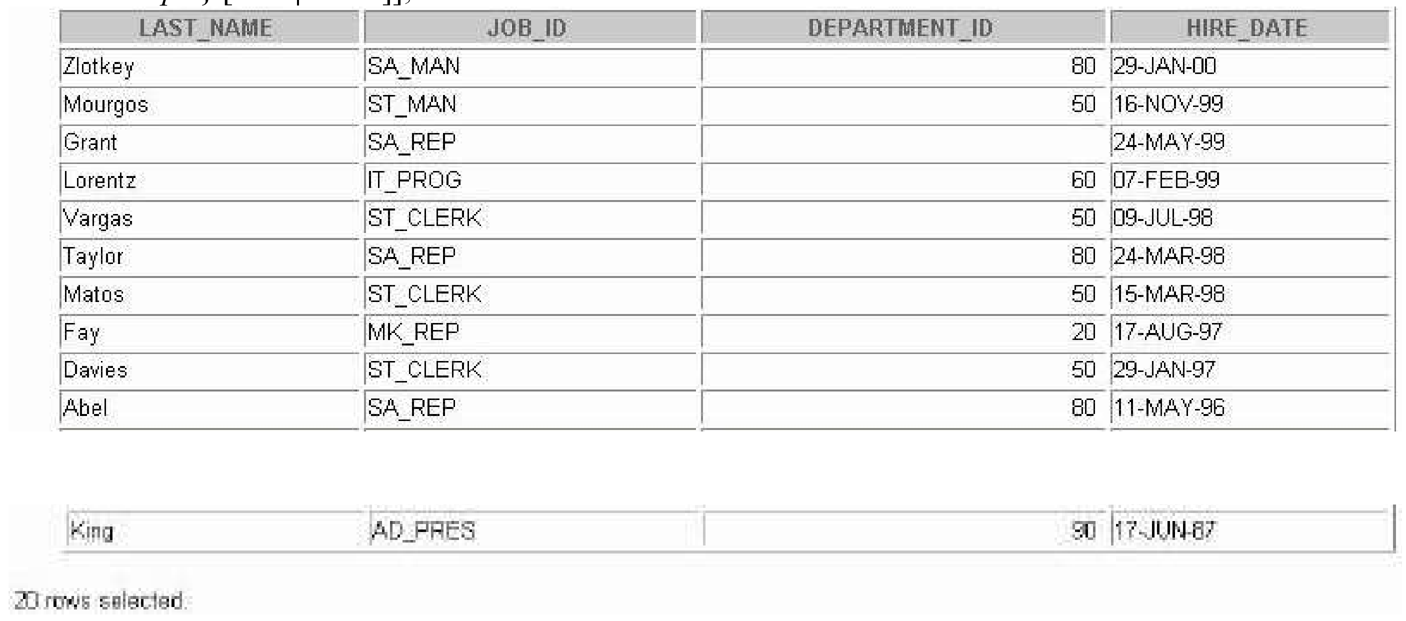
ORDER BY Clause

• Sort rows with the ORDER BY clause
– ASC : ascending order (the default order)
– 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;

Syntax

SELECT *expr* FROM *table* [WHERE *condition(s)*] [ORDER BY {*column*, *Expr* } [ASC|DESC]];

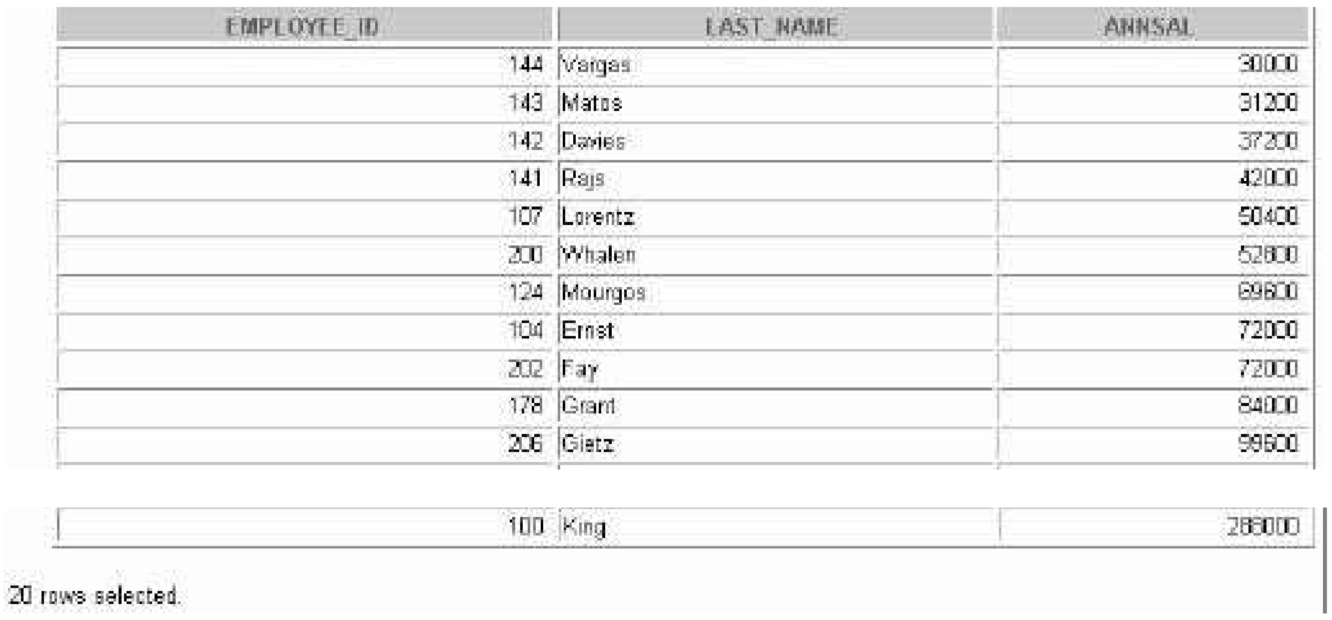


LAST_NAME	JOB_ID	DEPARTMENT_ID	HIRE_DATE
Zlotkey	SA_MAN	80	29-JAN-00
Mourgos	ST_MAN	50	16-NOV-99
Grant	SA_REP		24-MAY-99
Lorentz	IT_PROG	60	07-FEB-99
Vargas	ST_CLERK	50	09-JUL-98
Taylor	SA_REP	80	24-MAR-98
Matos	ST_CLERK	50	15-MAR-98
Fay	MK_REP	20	17-AUG-97
Davies	ST_CLERK	50	29-JAN-97
Abel	SA_REP	80	11-MAY-96
King	AD_PRES	90	17-JUN-87

20 rows selected.

Sorting in Descending Order

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



EMPLOYEE_ID	LAST_NAME	ANNSAL
144	Vargas	30000
143	Matos	31200
142	Davies	37200
141	Rajs	42000
107	Lorentz	50400
200	Whalen	52800
124	Mourgos	69600
104	Ernst	72000
202	Fay	72000
178	Grant	84000
206	Gietz	99600
100	King	288000

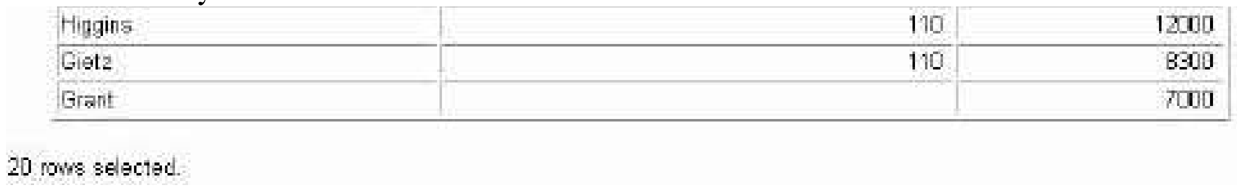
20 rows selected.

Sorting by Column Alias

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

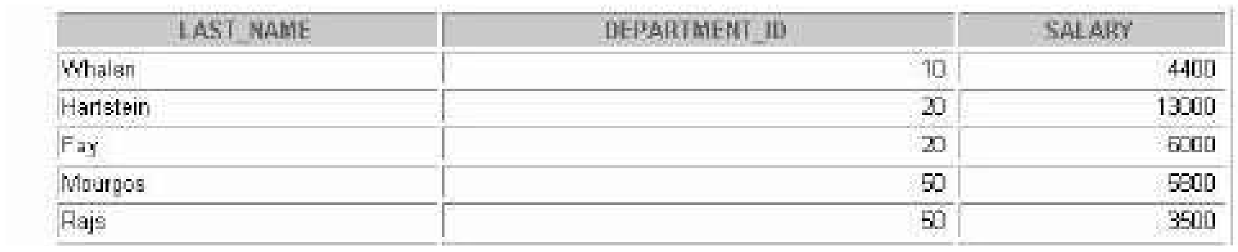
Sorting by Column Aliases

You can use a column alias in the ORDER BY clause. The slide example sorts the data by annual salary.



Higgins	110	12000
Gietz	110	8300
Grant		7000

20 rows selected.



LAST_NAME	DEPARTMENT_ID	SALARY
Whalen	10	4400
Hartstein	20	13000
Fay	20	6000
Mourgos	50	5800
Rajs	50	3500

Sorting by Multiple Columns

•The order of ORDER BY list is the order of sort.

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

•You can sort by a column that is not in the SELECT list.

Practice 1

1. Create a query to display the last name and salary of employees earning more than $12,000.
Place your SQL statement in a text file named lab2_1.sql . Run your query.
2. Create a query to display the employee last name and department number for employee number 176.
3. Modify lab2_1.sql to display the last name and salary for all employees whose salary is not in the range of $5,000 and $12,000. Place your SQL statement in a text file named lab2_3.sql
4. Display the employee last name, job ID, and start date of employees hired between February 20, 1998, and May 1, 1998. Order the query in ascending order by start date.
5. Display the last name and department number of all employees in departments 20 and 50 in alphabetical order by name.
6. Modify lab2_3.sql to list the last name and salary of employees who earn between $5,000 and $12,000, and are in department 20 or 50. Label the columns Employee and Monthly Salary , respectively. Resave lab2_3.sql as lab2_6.sql . Run the statement in lab2_6.sql .

7. Display the last name and hire date of every employee who was hired in 1994.
8. Display the last name and job title of all employees who do not have a manager.
9. Display the last name, salary, and commission for all employees who earn commissions. Sort data in descending order of salary and commissions.

If you have time, complete the following exercises:

10. Display the last names of all employees where the third letter of the name is an *a.*
11. Display the last name of all employees who have an *a* and an *e* in their last name.

If you want an extra challenge, complete the following exercises:

12. Display the last name, job, and salary for all employees whose job is sales representative or stock clerk and whose salary is not equal to $2,500, $3,500, or $7,000.
13. Modify lab2_6.sql to display the last name, salary, and commission for all employees whose commission amount is 20%. Resave lab2_6.sql as lab2_13.sql . Rerun the statement in lab2_13.sql.