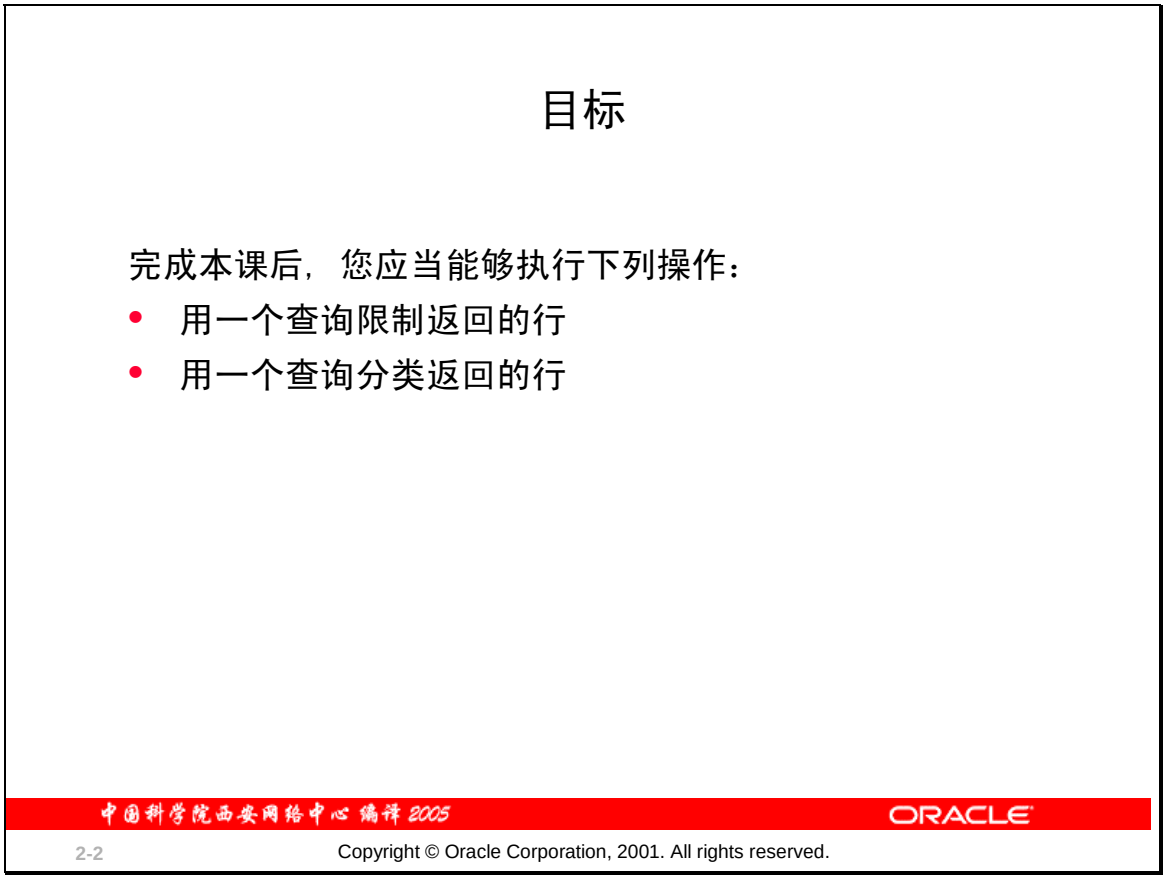
幻灯片 1

进度表：	时间	主题
	45 minutes	讲演
	30 minutes	练习
	75 minutes	总共

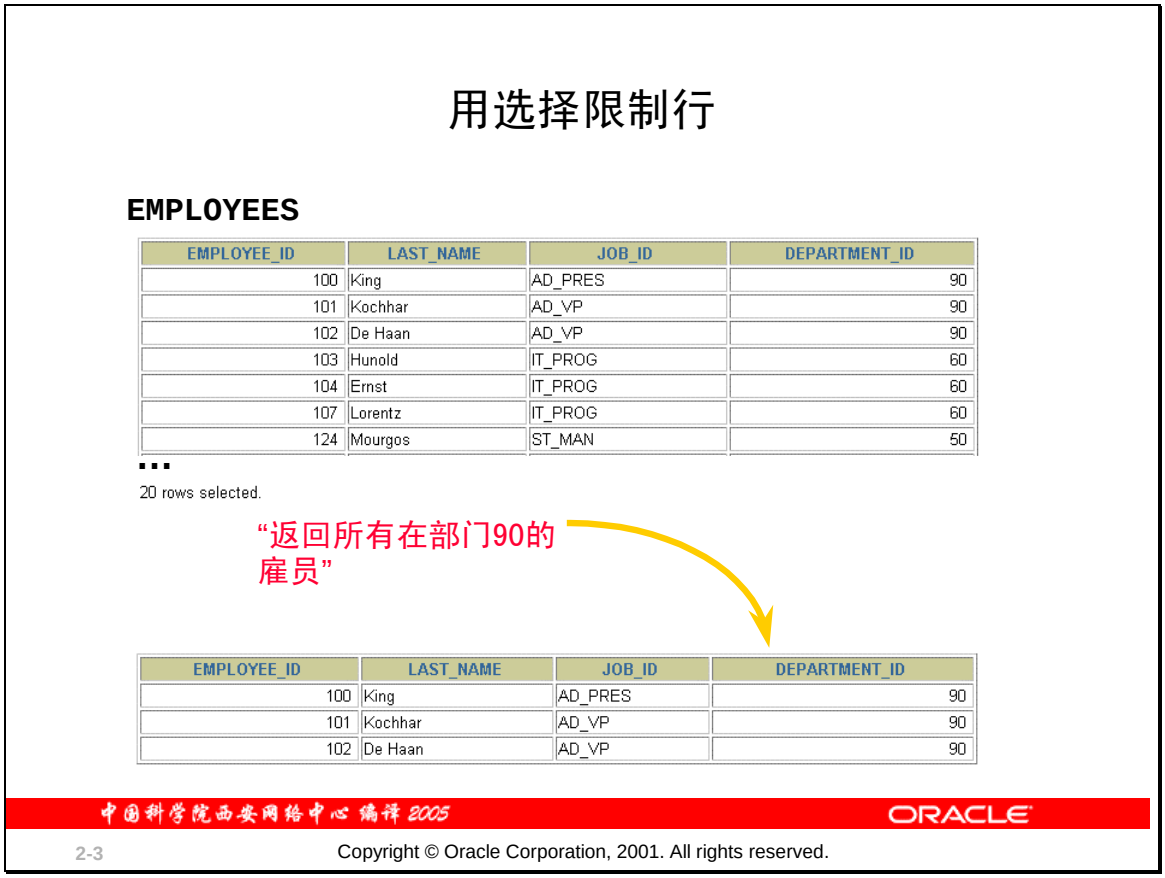
幻灯片 2



课程目标

从数据库取回数据的时候，你可能需要限制所显示的数据行，或者将显示的结果行排序。本课介绍用来完成这些工作的 SQL 语句。

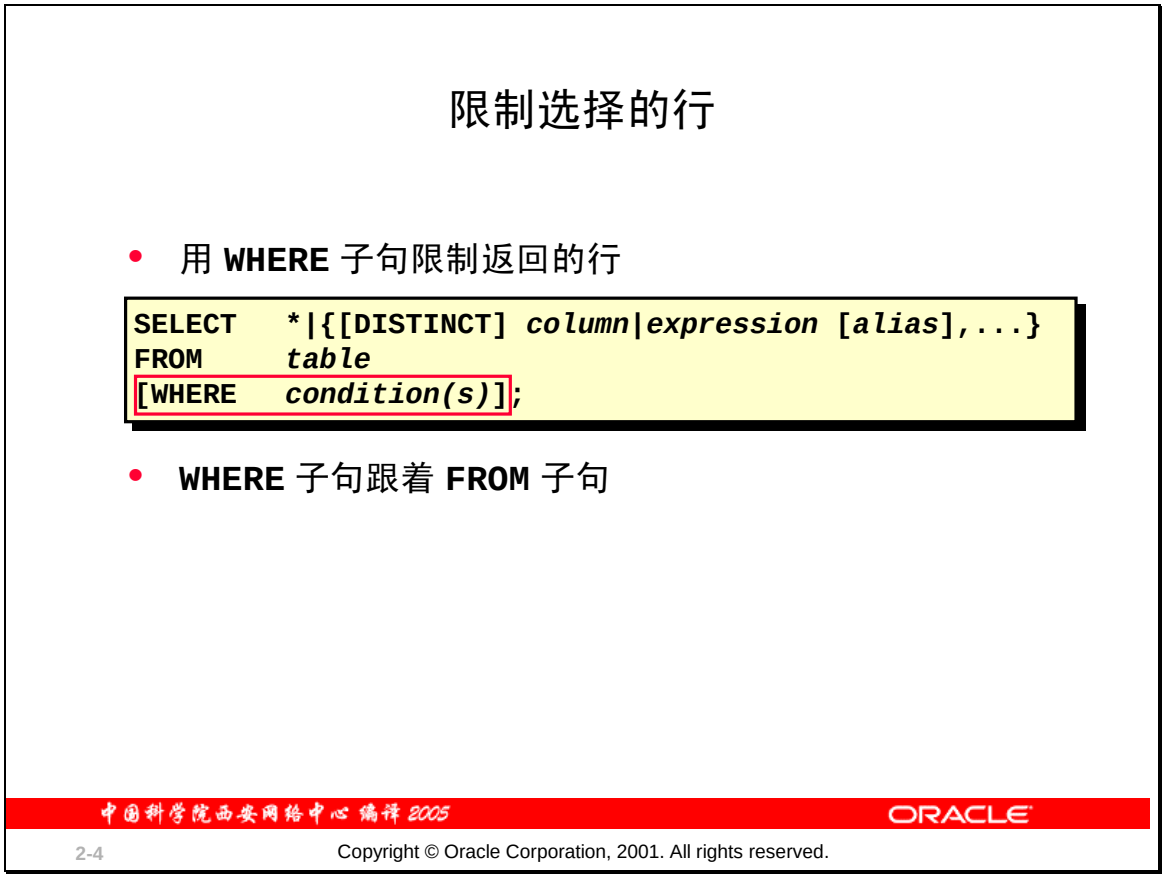
幻灯片 3



用选择限制行

在幻灯片的例子中，假定你想要显示部门 90 中所有的雇员。只返回 DEPARTMENT_ID 列值为 90 的行，这种限制方法基于 SQL 的 WHERE 子句。

幻灯片 4



限制选择的行

你能够用 WHERE 子句限制从查询返回的行。一个 WHERE 子句包含一个必须满足的条件，WHERE 子句紧跟着 FROM 子句。如果条件是 true，返回满足条件的行。

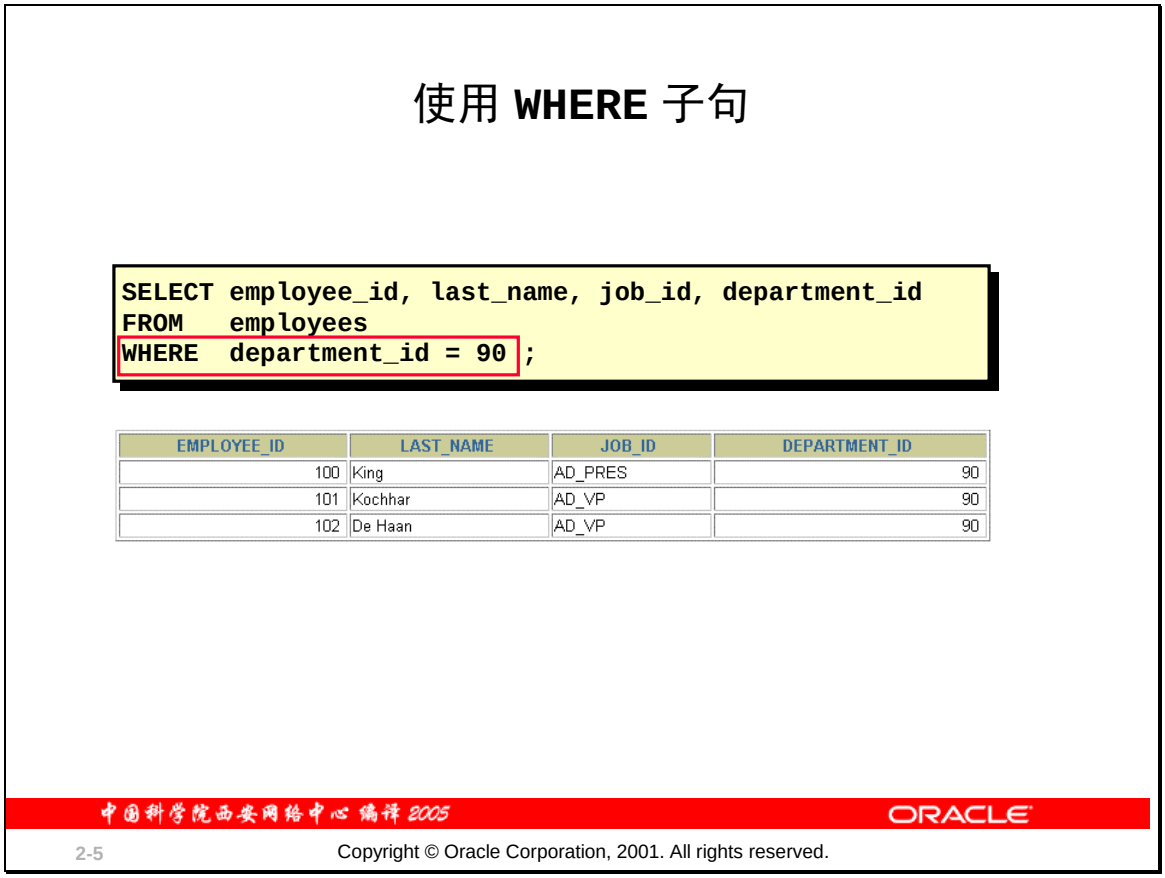
在语法中：

WHERE	限制查询满足条件的行
condition	由列名、表达式、常数和比较操作组成

WHERE 子句能够比较列值、文字值、算术表达式或者函数，WHERE 子句由三个元素组成：

- 列名
- 比较条件
- 列名、常量或值列表

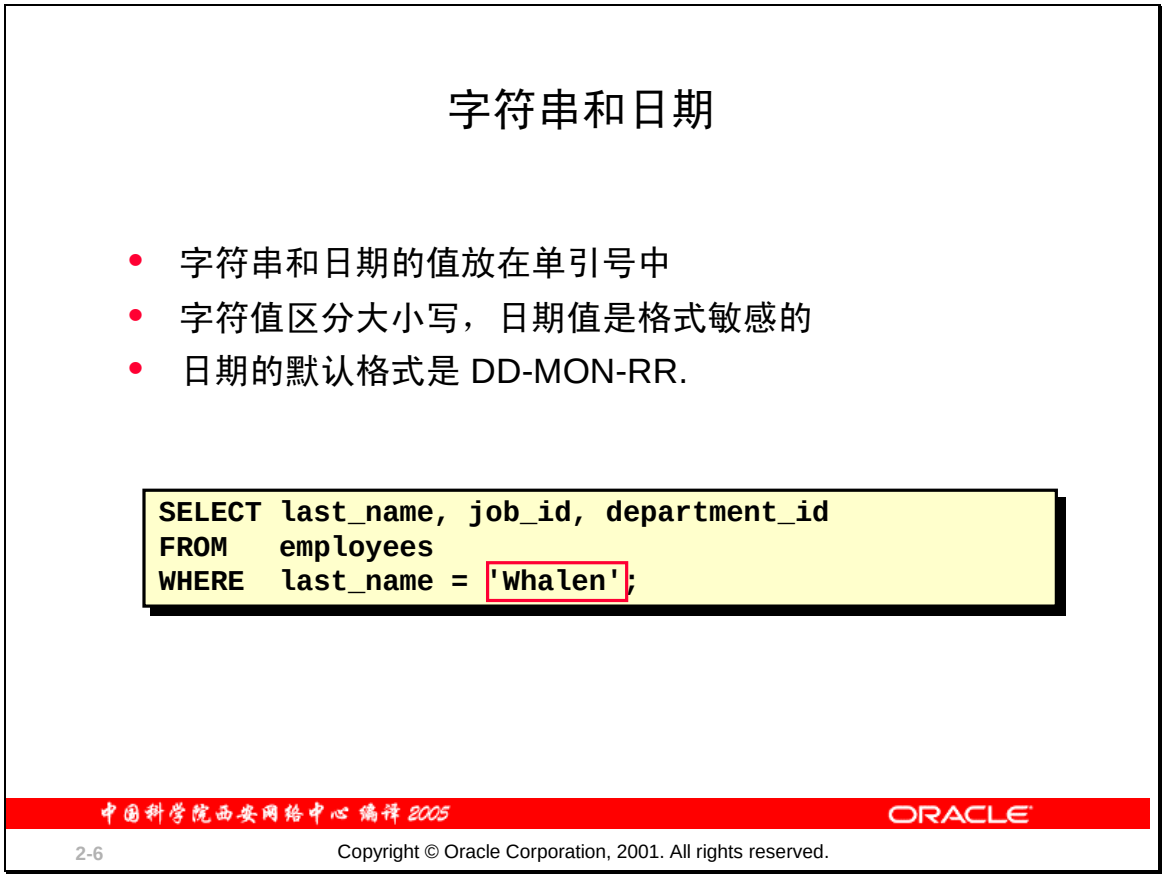
幻灯片 5



使用 WHERE 子句

在例子中，SELECT 语句取回所有雇员的 name，job ID 和 department 号，这些雇员的 department_id 是 90。

幻灯片 6



字符串和日期

在 WHERE 子句中字符串和日期必须包含在单引号 (' ') 中。但是，数字常数不应该包含在单引号中。

所有的字符搜索是大小写敏感的。在下面的例子中，没有行返回，因为 EMPLOYEES 表存储的所有名字都是大小写混合的：

```
SELECT last_name, job_id, department_id
FROM   employees
WHERE  last_name = 'WHALEN';
```

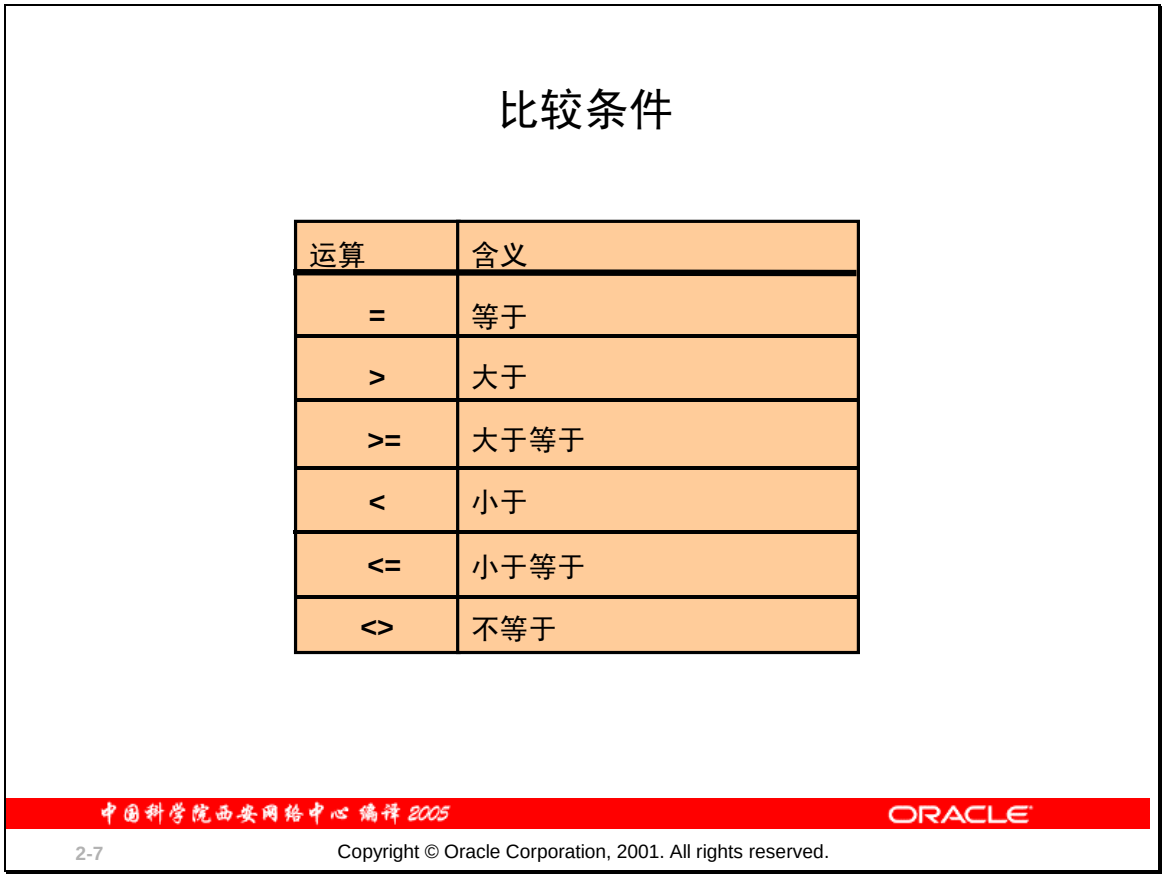
Oracle 数据库以内部数字格式存储日期，表示为：世纪、年、月、日、小时、分和秒。默认的日期显示是 DD-MON-RR。

注：改变默认的日期格式的方法在后面的课程中介绍。

教师注释

一些学生可能会问，怎样覆盖大小写敏感，在后面的课程中，我们将介绍，用单行函数，例如，UPPER 和 LOWER 来覆盖大小写敏感。

幻灯片 7



比较条件

比较条件被用于一个表达式与一个值或与另一个表达式的比较。他们以下面的格式被用于 WHERE 子句中：

语法

```
... WHERE expr operator value
```

举例

```
... WHERE hire_date='01-JAN-95'
... WHERE salary>=6000
... WHERE last_name='Smith'
```

别名不能用在 WHERE 子句中。

注：符号 != 和 ^= 也能够表示 *不等于* 条件。

幻灯片 8

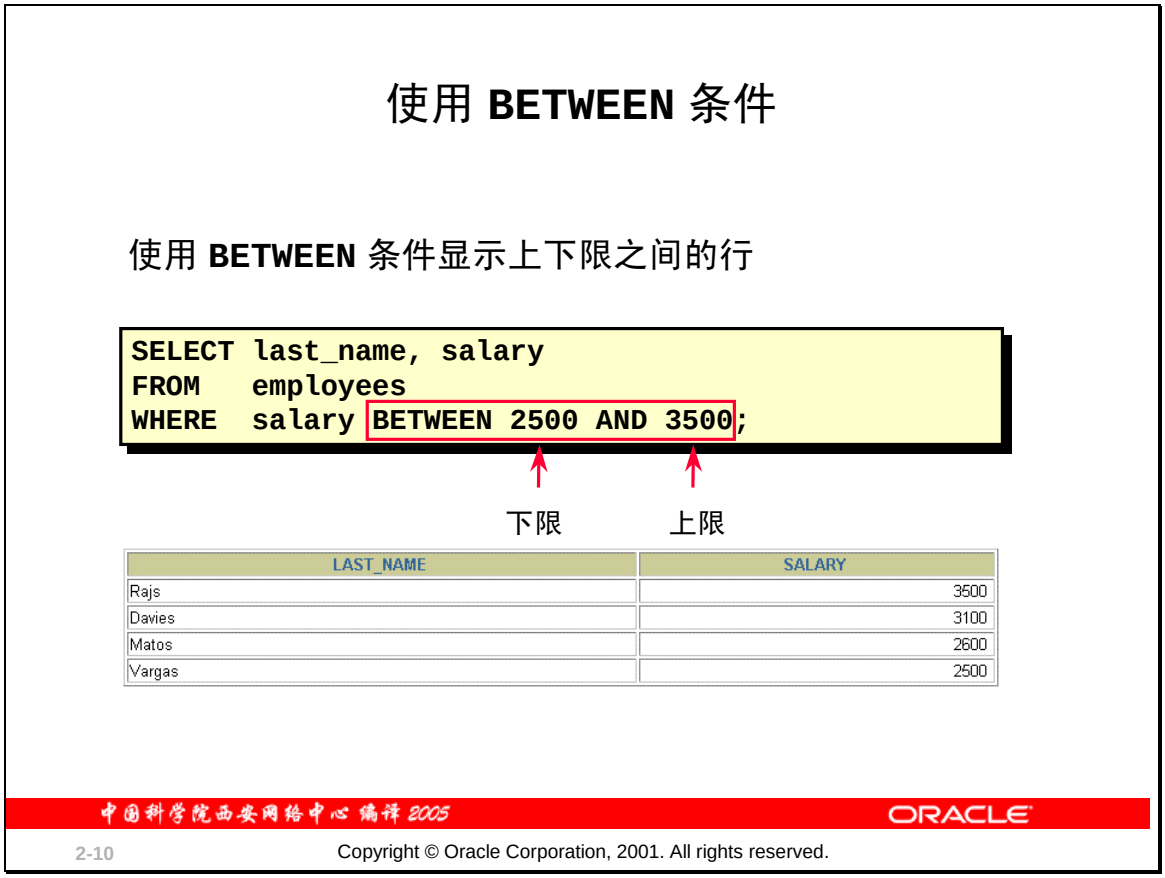
使用比较条件

在例子中，SELECT 从 EMPLOYEES 表语句取回 last name 和 salary，在这里这些雇员的薪水小于等于 3000。注意，有一个直接的值 3000 提供给 WHERE 子句。该直接值 3000 与 EMPLOYEES 表中的 SALARY 列的薪水值进行比较。

其它比较条件

操作	含义
BETWEEN ...AND...	在两个值之间 (包含)
IN(set)	匹配一个任意值列表
LIKE	匹配一个字符模板
IS NULL	是一个空值

幻灯片 10



BETWEEN 条件

你可以用 BETWEEN 范围条件显示基于一个值范围的行。你指定的范围包含一个下限和一个上限。

幻灯片中的 SELECT 语句从 EMPLOYEES 表中返回薪水在$2,500 和$3,500 之间的那些雇员。

BETWEEN 条件包括指定的上下限值，必须先指定下限。

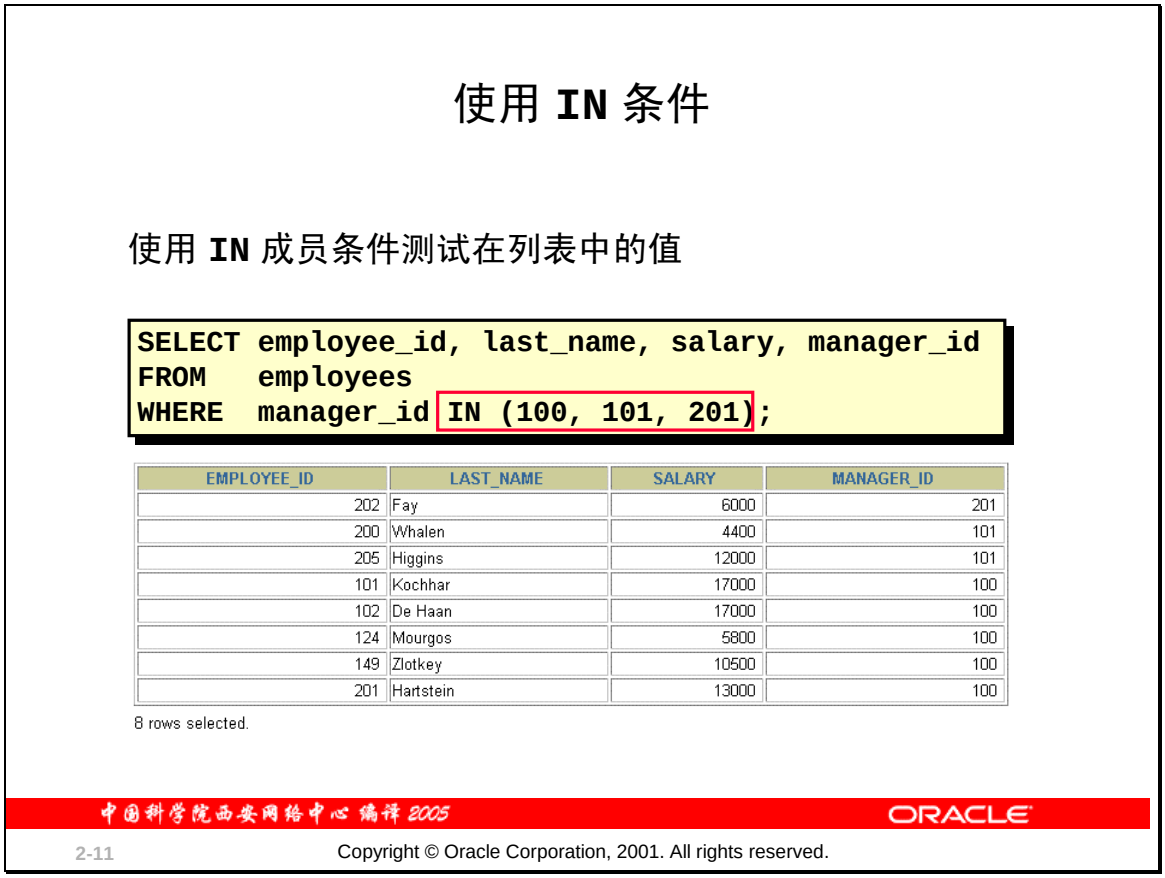
教师注释

强调在例子中也包括用 BETWEEN 运算符指定的值。解释 BETWEEN … AND …实际上是由 Oracle 服务器转变为一对 AND 条件：(a >= 下限) AND (a <= 上限)，所以使用 BETWEEN … AND … 并没有性能的提高，只是逻辑上简单。

演示：2_betw.sql

目的：举例说明 BETWEEN 运算符的使用。

幻灯片 11



IN 条件

用 IN 条件在指定的一组值中进行测试。IN 条件也就是 *成员条件*。

在幻灯片的例子中显示所有经理号为 100、101 或 201 的雇员的 employee numbers, last names, salaries 和经理的 employee numbers。

在 IN 条件中可以使用任何数据类型。下面的例子从 EMPLOYEES 表返回雇员信息行，这些雇员的名字包括在 WHERE 子句的名字列表中：

```
SELECT employee_id, manager_id, department_id
FROM   employees
WHERE  last_name IN ('Hartstein', 'Vargas');
```

如果 IN 条件中的成员是字符或日期，它们必须放在单引号 (' ') 中。

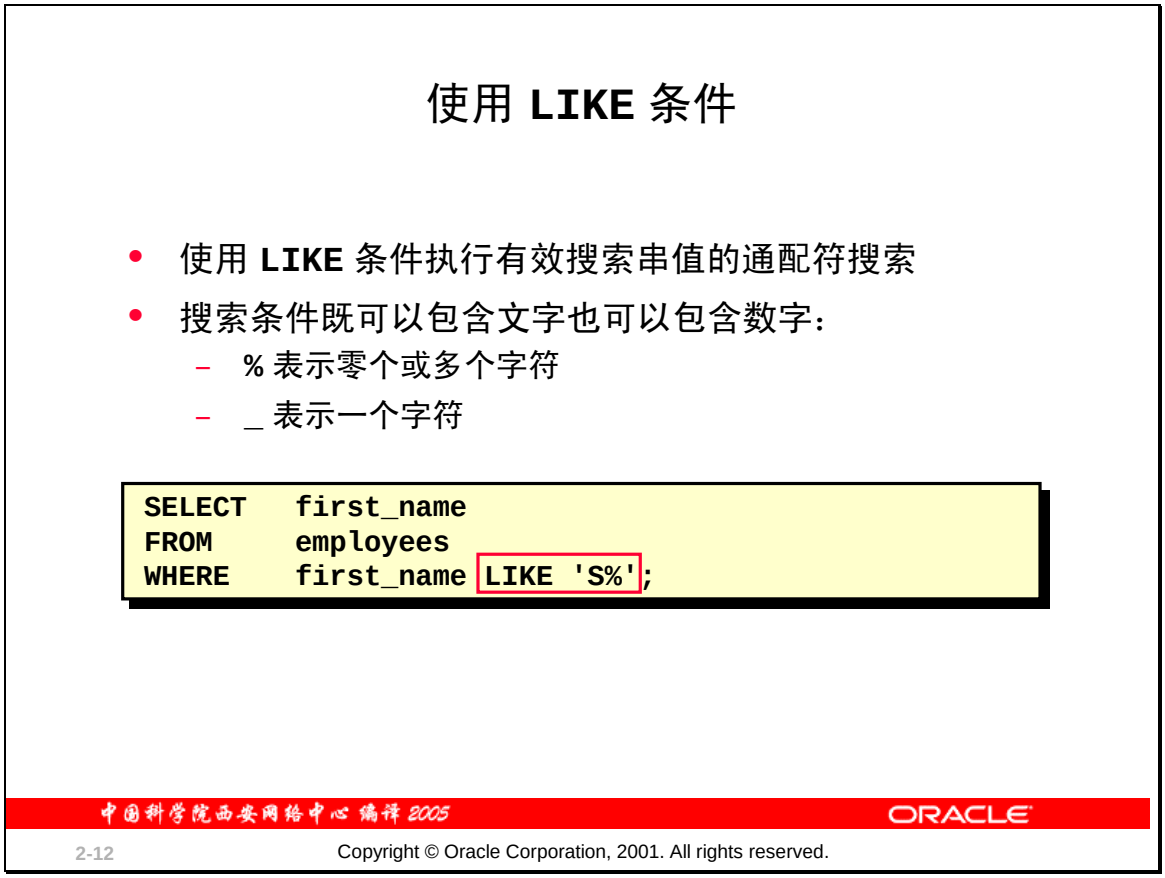
教师注释

解释 IN (...) 实际上是由 Oracle 服务器转变为一组 OR 条件：a = value1 OR a = value2 OR a = value3，所以使用 IN (...) 并没有得到性能的提高，只是逻辑上简单。

演示：2_in.sql

目的：举例说明 IN 操作符的使用。

幻灯片 12



LIKE 条件

你也许不总能知道要搜索的确切的值，你能够选择那些用 LIKE 条件匹配一个字符模板的行。字符模板匹配运算涉及通配符查询。有两个符号 % 和 _ 可以用来构造搜索串。

符号	说明
%	表示任意顺序的 0 个或多个字符
_	表示任意单个字符

幻灯片中的 SELECT 语句返回 EMPLOYEES 表中名字以一个大写字母 S 开始的雇员的名字，那些以小写字母 s 开始的名字不会返回。

LIKE 条件能够被用于某些 BETWEEN 比较。下面的例子显示所有在 1995 年 1 月和 1995 年 12 月之间进入本公司的雇员的名字和受雇日期：

```
SELECT last_name, hire_date
FROM   employees
WHERE  hire_date LIKE '%95';
```

幻灯片 13

使用 LIKE 条件

- 你可以组合模式匹配字符串

```
SELECT last_name
FROM   employees
WHERE  last_name LIKE '_o%';
```

LAST_NAME
Kochhar
Lorentz
Mourgos

- 你可以用 **ESCAPE** 标识符搜索实际的 % 和 _ 符号

通配符字符的组合

% 和 _ 符号能够被用来与文字字符组合。幻灯片中的例子显示所有名字中有一个 o 而且位于第二个位置的雇员的名字。

ESCAPE 选项

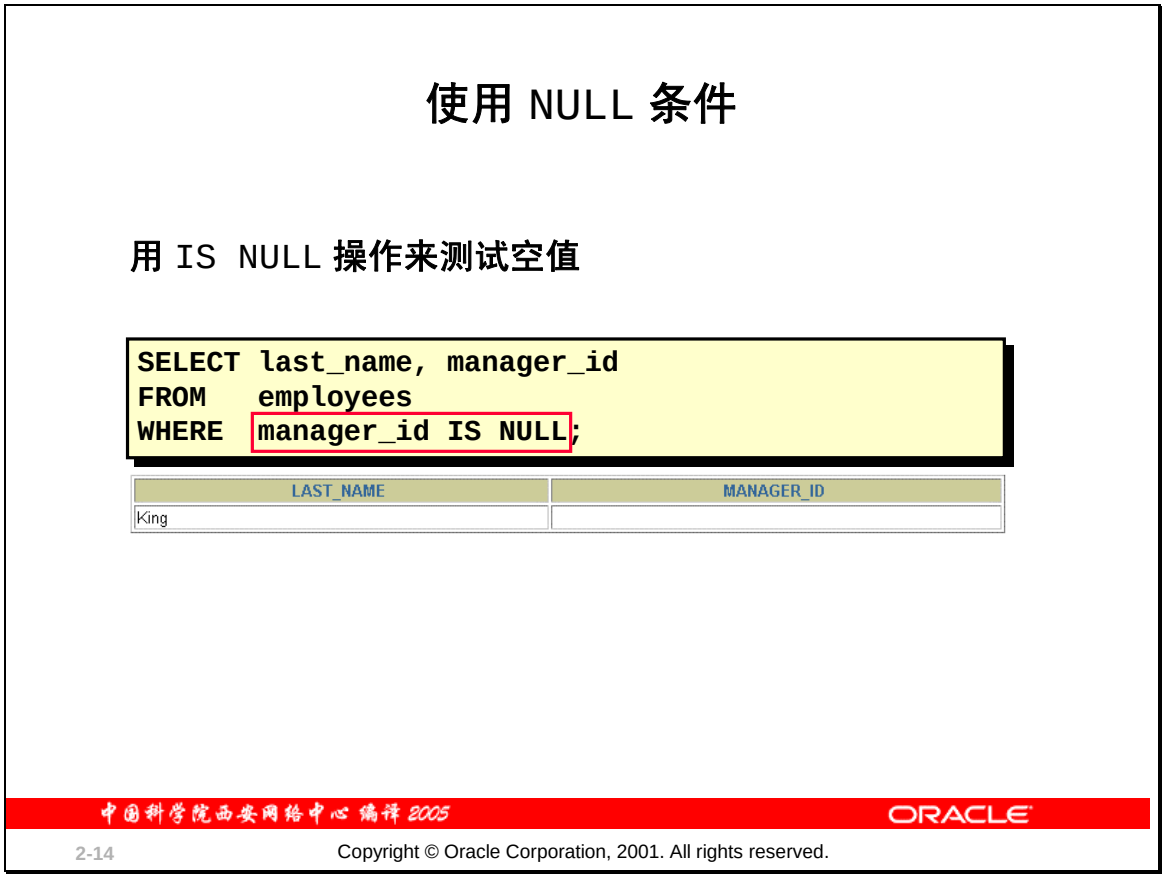
当你需要有一个确切匹配实际的字符 % 和 _ ，使用 ESCAPE 选项，该选项指定换码符是什么。如果你想要搜索包含‘SA_’的字符串，可以用下面的 SQL 语句：

```
SELECT employee_id, last_name, job_id
FROM   employees
WHERE  job_id LIKE '%SA\_%' ESCAPE '\';
```

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

ESCAPE 选项指定反斜线 (\) 为换码符，在模板中，换码符字符在下划线 (_) 的前面，原因是 Oracle 服务器逐字地解释字符串。

幻灯片 14



NULL 条件

NULL 条件，包括 IS NULL 条件和 IS NOT NULL 条件。

IS NULL 条件用于空值测试。空值的意思是难以获得的、未指定的、未知的或者不适用的。因此，你不能用 = ，因为 null 不能等于或不等于任何值。幻灯片中的例子取回所有没有主管经理的雇员的名字和他的主管经理。

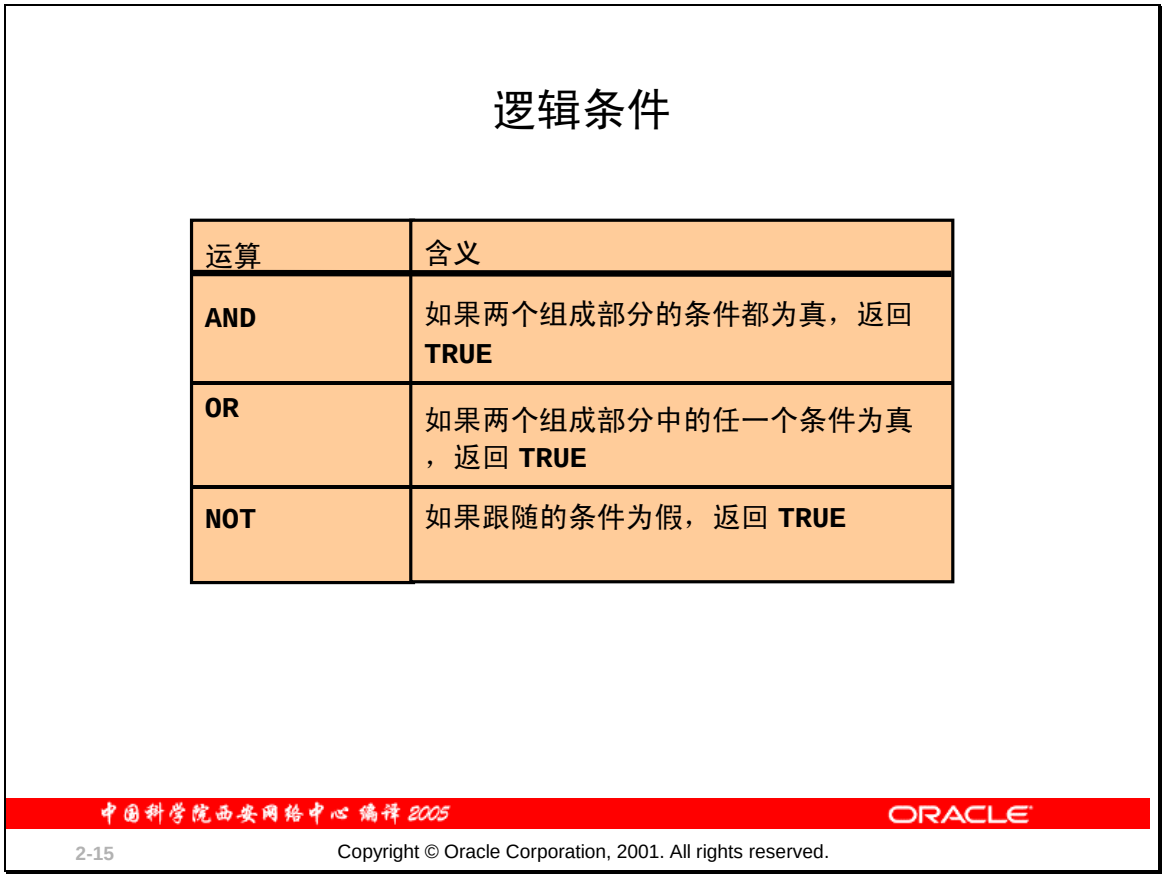
例如，显示所有没有佣金的雇员的 last name, job ID 和 commission，用下面的 SQL 语句：

```
SELECT last_name, job_id, commission_pct
FROM   employees
WHERE  commission_pct IS NULL;
```

LAST_NAME	JOB_ID	COMMISSION_PCT
King	AD_PRES	
Kochhar	AD_VP	
De Haan	AD_VP	
...	...	...
Gietz	AC_ACCOUNT	

16 rows selected.

幻灯片 15



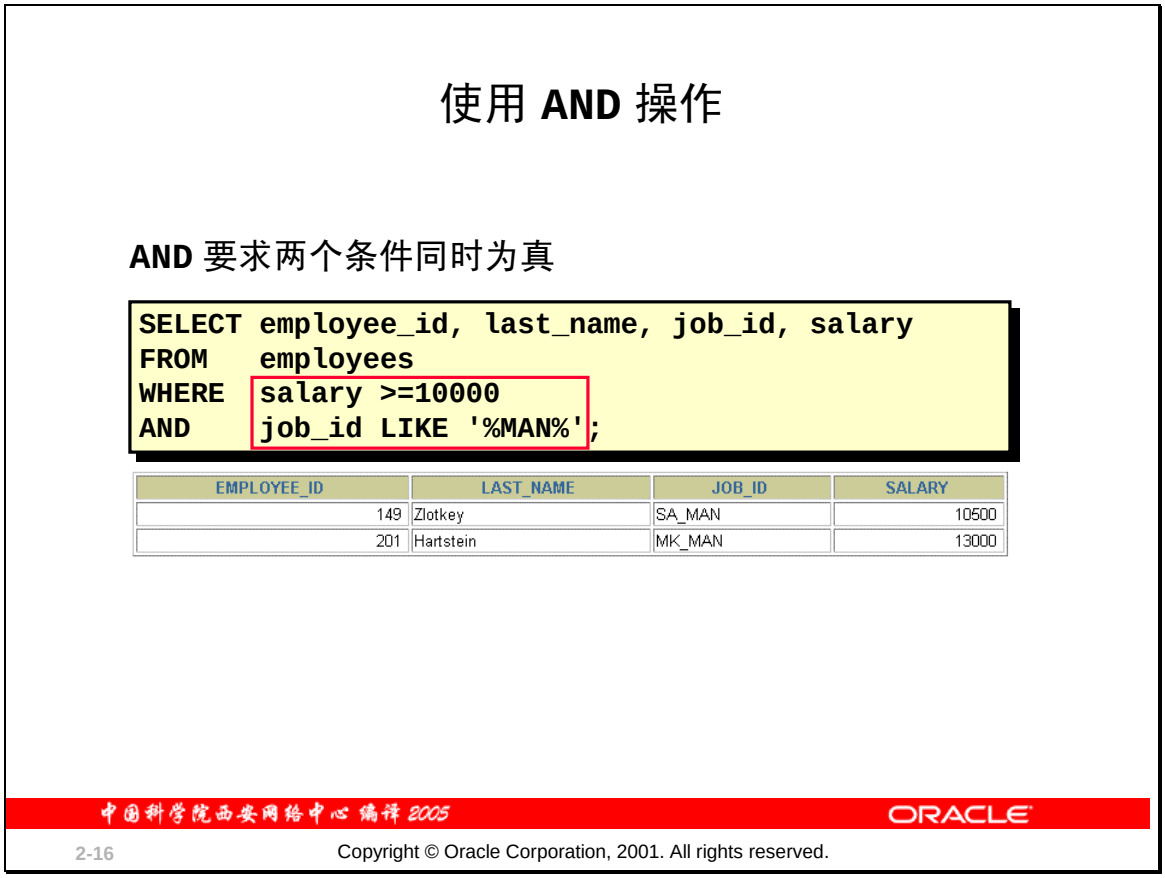
逻辑条件

逻辑条件组合两个比较条件的结果来产生一个基于这些条件的单个的结果，或者逆转一个单个条件的结果。当所有条件的结果为真时，返回行。SQL 的三个逻辑运算符是：

- AND
- OR
- NOT

至此所有的例子仅在 WHERE 子句中指定指定唯一的条件，你可以在 WHERE 子句中用 AND 和 OR 运算符使用多个条件。

幻灯片 16



AND 操作

在该例中，对于被选择的任何记录两个条件必须为真，因此，只有工作岗位包含字符串 MAN 并且收入大于等于$10,000 的那些雇员被选择。

所有字符搜索是大小写敏感的，如果 MAN 不是大写，则没有行返回。字符串必须放在引号中。

AND 真值表

下面的表显示了用 AND 组合两个表达式的结果：

AND	TRUE	FALSE	NULL
TRUE	TRUE	FALSE	NULL
FALSE	FALSE	FALSE	FALSE
NULL	NULL	FALSE	NULL

教师注释

演示：2_and.sql

目的：举例说明 AND 运算符的使用。

幻灯片 17

使用 OR 操作

OR 操作要求两者之一为真即可

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

EMPLOYEE_ID	LAST_NAME	JOB_ID	SALARY
100	King	AD_PRES	24000
101	Kochhar	AD_VP	17000
102	De Haan	AD_VP	17000
124	Mourgos	ST_MAN	5800
149	Zlotkey	SA_MAN	10500
174	Abel	SA_REP	11000
201	Hartstein	MK_MAN	13000
205	Higgins	AC_MGR	12000

8 rows selected.

OR 操作

在例子中，两个条件之一为真的那些记录都被选择，因此，任何 job ID 中包含 MAN 或者收入大于等于$10,000 的雇员都被选择。

OR 真值表

下面的表显示了用 OR 组合两个表达式的结果：

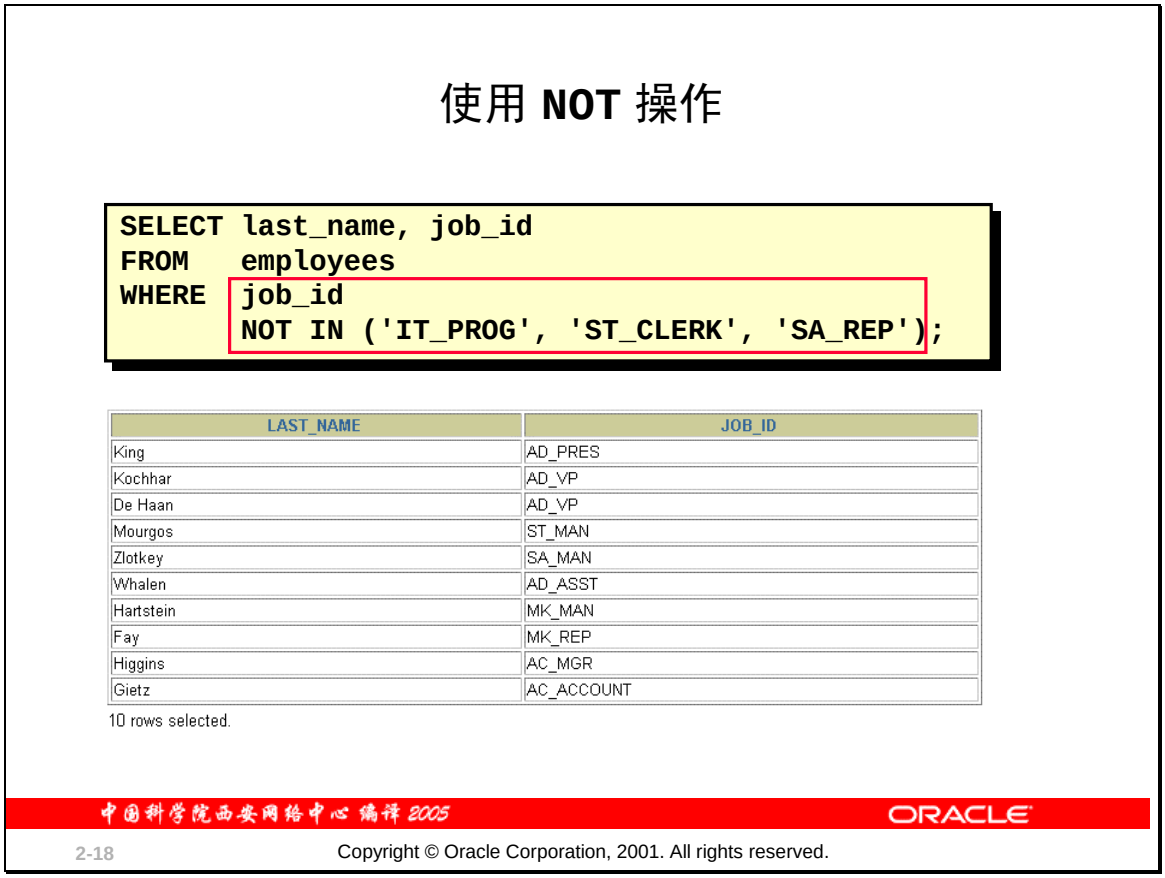
OR	TRUE	FALSE	NULL
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	NULL
NULL	TRUE	NULL	NULL

教师注释

示范：2_or.sql

目的：举例说明 OR 运算符的使用。

幻灯片 18



NOT 操作

幻灯片中的例子显示那些工作岗位不是 IT_PROG、ST_CLERK 或 SA_REP 的雇员的名字和工作岗位。

NOT 真值表

下面的表显示了应用 NOT 运算符到一个条件的结果：

NOT	TRUE	FALSE	NULL
	FALSE	TRUE	NULL

注：NOT 运算符也可以用于另一个 SQL 运算符，例如，BETWEEN、LIKE、和 NULL。

```
... WHERE  job_id    NOT  IN ('AC_ACCOUNT', 'AD_VP')
... WHERE  salary    NOT  BETWEEN  10000 AND  15000
... WHERE  last_name NOT  LIKE '%A%'
... WHERE  commission_pct  IS   NOT  NULL
```

幻灯片 19

优先规则

求值顺序	
1	算术运算
2	连字操作
3	比较操作
4	IS [NOT] NULL, LIKE, [NOT] IN
5	[NOT] BETWEEN
6	NOT 逻辑条件
7	AND 逻辑条件
8	OR 逻辑条件

使用圆括号改变优先规则

优先规则

优先规则定义表达式求值和计算的顺序，表中列出了默认的优先顺序。你可以用圆括号括住你想要先计算的表达式来覆盖默认的优先顺序。

幻灯片 20

优先规则

```
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
Abel	SA_REP	11000
Taylor	SA_REP	8600
Grant	SA_REP	7000

AND 操作优先的例子

在幻灯片的例子中，有两个条件：

- 第一个条件是 job_id 是 AD_PRES 并且薪水高于 15,000。
- 第二个条件是 job_id 是 SA_REP。

因此，该 SELECT 语句读作：

“选择符合下面条件的行，雇员是董事长 (president) ，并且收入超过$15,000，或雇员是销售代表。”

教师注释

演示：2_sal1.sql

目的：举例说明优先规则。

幻灯片 21

优先规则

使用圆括号强制优先权

```
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

使用圆括号

在上面的例子中有两个条件：

- 第一个条件是 job_id 是 AD_PRES 或者 SA_REP 。
- 第二个条件是薪水高于$15,000

因此，该 SELECT 语句读作：

“选择符合下面条件的行，雇员是董事长（president）或者销售代表，并且收入超过$15,000。”

教师注释

演示：2_sal2.sql

目的：举例说明优先规则。

幻灯片 22

ORDER BY 子句

在一个不明确的查询结果中排序返回的行。ORDER BY 子句用于分排序。如果使用了 ORDER BY 子句，它必须位于 SQL 语句的最后。你可以指定一个表达式，或者一个别名，作为排序条件。

语法

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

在语法中：

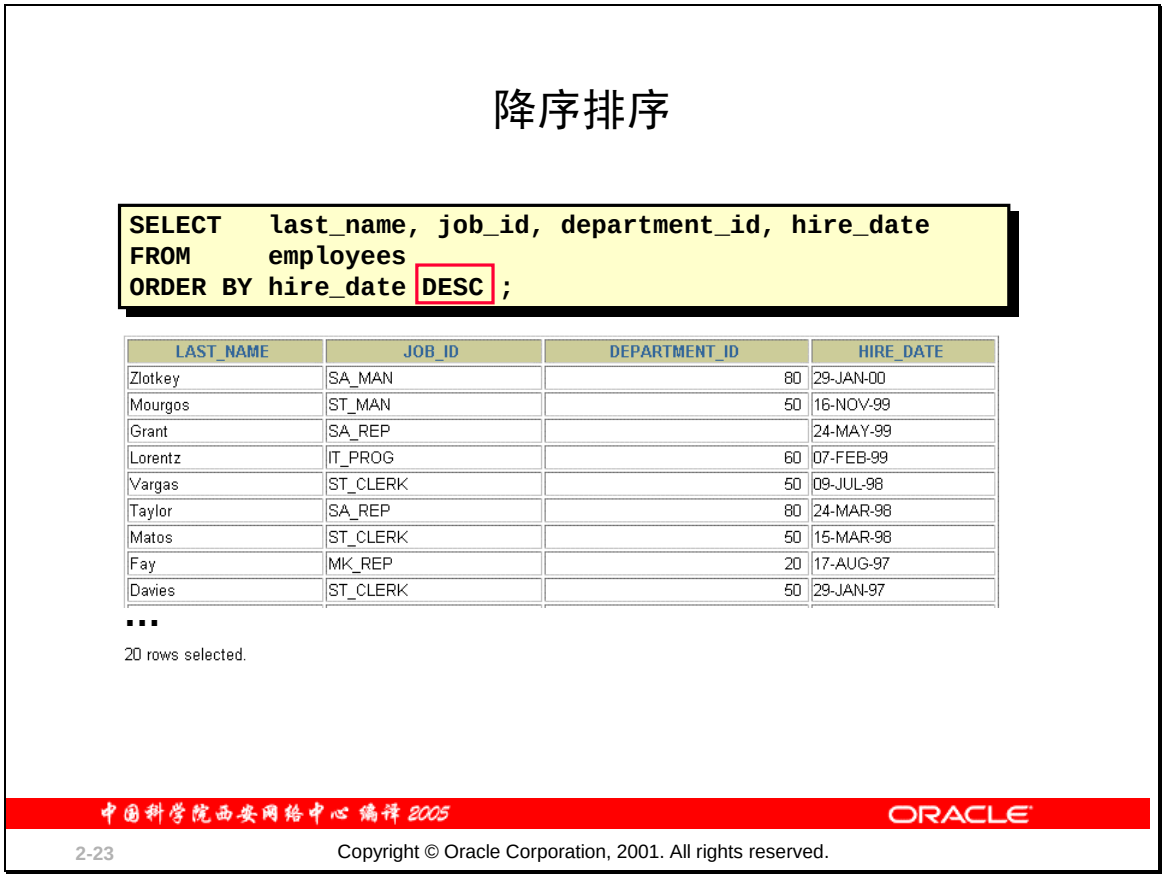
ORDER BY	指定排序显示返回的行
ASC	以升序排序行 (这是默认排序)
DESC	以降序排序行

如果未使用 ORDER BY 子句，排序次序就未定义，并且 Oracle 服务器可能对于相同查询的两次执行取回行的顺序不同。ORDER BY 子句以指定的顺序显示返回的行。

教师注释

让学生知道 ORDER BY 子句在查询完成后执行，除非使用了 FOR UPDATE 子句，ORDER BY 子句应放在最后。

幻灯片 23



数据的默认排序

默认的排序顺序是升序：

- 对于数字值，小的只值在前面显示—例如，1 – 999。
- 对于日期，早的日期在前面显示—例如，01-JAN-92 在 01-JAN-95 。
- 对于字符值，依字母顺序显示—例如，*A* 第一，*Z* 最后。
- 对于空值，升序排序时显示在最后，降序排序时显示在最前面。

反转默认排序

为了显示倒序排序的行，在 ORDER BY 子句的列名后面指定 DESC 关键字。幻灯片中的例子按照雇员的受雇日期从近到远排序。

教师注释

让学生知道，也能够以 SELECT 列表中的列号排序。下面的例子用 salary 以逆序排序输出：

```
SELECT    last_name, salary
FROM      employees
ORDER BY  2  DESC;
```

幻灯片 24

用列别名排序

能够在 ORDER BY 子句中使用列别名。幻灯片的例子用年薪排序数据。

教师注释

在内部，SELECT 语句的执行顺序如下：

- FROM 子句
- WHERE 子句
- SELECT 子句
- ORDER BY 子句

幻灯片 25

多列排序

- **ORDER BY** 列表的顺序就是排序的顺序

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

LAST_NAME	DEPARTMENT_ID	SALARY
Whalen	10	4400
Hartstein	20	13000
Fay	20	6000
Mourgos	50	5800
Rajs	50	3500
Davies	50	3100
Matos	50	2600
Vargas	50	2500

...

20 rows selected.

- 你可以排序一个不在 **SELECT** 列表中的列

用多列排序

你可以用多列排序查询结果。排序的限制是所给表中的列数。

在 ORDER BY 子句中，多个指定的列名之间用逗号分开。如果你想要倒许序排序一个列，在该列名后面指定 DESC。你也可以用没有包括在 SELECT 子句中的列排序。

例子

显示所有雇员的名字和薪水。先用部门号顺序排序结果，再用薪水逆序排序。

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

教师注释

以升叙排序显示 DEPARTMENT_ID 列，同时以降序排序显示 SALARY 列。

幻灯片 26

小结

在本课中，您应该已经学会如何：

- 使用 **WHERE** 子句限制输出行
  - 使用比较条件
  - 使用 **BETWEEN**, **IN**, **LIKE**, 和 **NULL** 条件
  - 应用逻辑 **AND**, **OR**, 和 **NOT** 操作
- 使用 **ORDER BY** 子句排序输出的行

```
SELECT      *|{[DISTINCT] column|expression [alias],...}
FROM        table
[WHERE      condition(s)]
[ORDER BY   {column, expr, alias} [ASC|DESC]];
```


小结

在本可中，你应该已经学会关于限制和排序 SELECT 语句返回的行。你也应该已经学会实现各种运算和使用条件。

幻灯片 27

练习 2 概览

本章练习包括下面的主题：

- 选择数据并且改变行显示的顺序
- 用 **WHERE** 子句限制行
- 用 **ORDER BY** 子句排序行

练习 2 概览

本章的作业包括使用 WHERE 子句和 ORDER BY 子句的多种练习。

幻灯片 28

练习 2

1. 创建一个查询，显示收入超过 $12,000 的雇员的名字和薪水。将 SQL 语句存到文件 lab2_1.sql 中，运行该查询。

LAST_NAME	SALARY
King	24000
Kochhar	17000
De Haan	17000
Hartstein	13000

```
SELECT last_name, salary
FROM employees
WHERE salary > 12000;
```

2. 创建一个查询，显示雇员号为 176 的雇员的名字和部门号。

LAST_NAME	DEPARTMENT_ID
Taylor	80

```
SELECT last_name, department_id
FROM employees
WHERE employee_id = 176;
```

3. 修改 lab2_1.sql 文件，显示所有薪水不在 5000 和 12000 之间的雇员的名字和薪水。将 SQL 语句存到文件 lab2_3.sql 中。

LAST_NAME	SALARY
King	24000
Kochhar	17000
De Haan	17000
Lorentz	4200
Rajs	3500
Davies	3100
Matos	2600
Vargas	2500
Whalen	4400
Hartstein	13000

10 rows selected.

```
SELECT last_name, salary
FROM employees
WHERE salary NOT BETWEEN 5000 AND 12000;
```

4. 显示受雇日期在 1998 年 2 月 20 日 和 1998 年 5 月 1 日 之间的雇员的名字、岗位和受雇日期。按受雇日期顺序排序查询结果。

LAST_NAME	JOB_ID	HIRE_DATE
Matos	ST_CLERK	15-MAR-98
Taylor	SA_REP	24-MAR-98

```
SELECT last_name, job_id, hire_date
FROM employees
WHERE hire_date BETWEEN '20-Feb-1998' AND '01-May-1998'
ORDER BY hire_date;
```

5. 显示所有在部门 20 和 50 中的雇员的名字和部门号，并以名字按字母顺序排序。

LAST_NAME	DEPARTMENT_ID
Davies	50
Fay	20
Hartstein	20
Matos	50
Mourgos	50
Rajs	50
Vargas	50

7 rows selected.

```
SELECT last_name, department_id
FROM employees
WHERE department_id IN (20, 50)
ORDER BY last_name;
```

6. 修改 lab2_3.sql 列出收入在 $5,000 和 $12,000 之间，并且在部门 20 或 50 工作的雇员的名字和薪水。将列标题分别显示为 Employee 和 Monthly Salary，将 lab2_3.sql 保存为 lab2_6.sql。运行 lab2_6.sql 中的语句。

LAST_NAME	HIRE_DATE
Higgins	07-JUN-94
Gietz	07-JUN-94

```
SELECT last_name "Employee", salary "Monthly Salary"
FROM employees
WHERE salary BETWEEN 5000 AND 12000
AND department_id IN (20, 50);
```

7. 显示每一个在 1994 年受雇的雇员的名字和受雇日期。

LAST_NAME	HIRE_DATE
Higgins	07-JUN-94
Gietz	07-JUN-94

```
SELECT last_name, hire_date
FROM employees
WHERE hire_date LIKE '%94';
```

8. 显示所有没有主管经理的雇员的名字和工作岗位。

LAST_NAME	JOB_ID
King	AD_PRES

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

9. 显示所有有佣金的雇员的名字、薪水和佣金。以薪水和佣金的降序排序数据。

LAST_NAME	SALARY	COMMISSION_PCT
Abel	11000	.3
Zlotkey	10500	.2
Taylor	8600	.2
Grant	7000	.15

```
SELECT last_name, salary, commission_pct
FROM employees
WHERE commission_pct IS NOT NULL
ORDER BY salary DESC, commission_pct DESC;
```

如果你有时间，完成下面的练习：

10. 显示所有名字中第三个字母是 a 的雇员的名字。

LAST_NAME
Grant
Whalen

```
SELECT last_name
FROM employees
WHERE last_name LIKE '__a%';
```

11. 显示所有名字中有一个 a 和一个 e 的雇员的名字。

LAST_NAME
De Haan
Davies
Whalen
Hartstein

```
SELECT last_name
FROM employees
WHERE last_name LIKE '%a%'
AND last_name LIKE '%e%';
```

如果你想要额外的挑战，完成下面的练习：

12. 显示所有工作是销售代表或者普通职员，并且薪水不等于 $2,500、$3,500 或 $7,000 的雇员的名字、工作和薪水。

LAST_NAME	JOB_ID	SALARY
Davies	ST_CLERK	3100
Matos	ST_CLERK	2600
Abel	SA_REP	11000
Taylor	SA_REP	8600

```
SELECT last_name, job_id, salary
FROM employees
WHERE job_id IN ('SA_REP', 'ST_CLERK')
AND salary NOT IN (2500, 3500, 7000);
```

13. 修改 lab2_6.sql 显示所有佣金总计为 20% 的雇员的名字、薪水和佣金。保存 lab2_6.sql 为 lab2_13.sql。再运行 lab2_13.sql 中的语句。

Employee	Monthly Salary	COMMISSION_PCT
Zlotkey	10500	.2
Taylor	8600	.2

```
SELECT last_name "Employee", salary "Monthly Salary",
commission_pct
FROM employees
WHERE commission_pct = .20;
```