WHERE Clause to Limit Row Selection

WHERE Clause

- WHERE clause is optional
  - If used, it directly follows the FROM clause

```
SELECT *
FROM EMPLOYEES
WHERE department = 'IT';
```

- Restricts or limits rows returned from the table
- Contains a condition that must be met for a row to be returned

```
department = 'IT'
```

WHERE Clause

- The syntax for the WHERE clause is:

```
WHERE column_name comparison_condition comparison_value
```

```
WHERE department_id = 110;
```

Comparison Operators in the WHERE Clause

- Comparison operators can be used to compare one expression to a value or expression:

  = equal to

  > greater than

  >= greater than or equal to

  < less than

  <= less than or equal to

  <> not equal to (or != or ^=)

Using WHERE clause for simple search condition

- All character searches are case-sensitive
- What will probably be returned with

```
WHERE last_name = 'jones';
```

Example 9-2

Return customer_name and credit_limit for all rows WHERE city is equal to Chicago.

Data Set ds9_2

```
ID |CUSTOMER_NAME   |CITY     |CREDIT_LIMIT|
---|----------------|---------|------------|
101|Bargains Galore|Detroit  |      125000|
102|RedHot Discount|San Diego|      185500|
103|NewTown Deals  |Dallas   |      132500|
104|Ajax Sales     |Detroit  |      150000|
105|BigBox Direct  |Chicago  |      145000|
106|Mainstreet Inc |Dallas   |      200000|
107|Riverside Mfg  |Chicago  |      164000|
108|Cube Industries|Dallas   |      185000|
109|LowCost Shops  |San Diego|      155000|
110|Sterling Mfg   |Chicago  |      180000|
```

SQL Statement

```
SELECT customer_name, credit_limit
FROM ds9_2
WHERE city = 'Chicago';
```

Result Set

```
CUSTOMER_NAME|CREDIT_LIMIT|
-------------|------------|
BigBox Direct|      145000|
Riverside Mfg|      164000|
Sterling Mfg |      180000|
```

Using WHERE clause with a Primary Key

Example 9-3	Return customer_name for id 106 (primary key).

Data Set ds9_3

```
ID |CUSTOMER_NAME   |CITY      |
---|----------------|----------|
101|Bargains Galore|Detroit   |
102|RedHot Discount|San Diego|
103|NewTown Deals   |Dallas    |
104|Ajax Sales      |Detroit   |
105|BigBox Direct   |Chicago   |
```

SQL Statement

```
SELECT customer_name
FROM ds9_3
WHERE id = 104;
```

Result Set

```
CUSTOMER_NAME
-------------
Ajax Sales
```

A Second Look: Restricting Columns and Rows

What is the purpose of the SELECT clause?

Example 9-4 The column-list restricts which columns are included in the result set

Data Set ds9_4

```
ID |CUSTOMER_NAME   |CITY      |CREDIT_LIMIT|
---|----------------|----------|------------|
101|Bargains Galore|Detroit   |      125000|
102|RedHot Discount|San Diego|      185500|
103|NewTown Deals  |Dallas    |      132500|
104|Ajax Sales     |Detroit   |      150000|
105|BigBox Direct  |Chicago   |      145000|
```

SQL Statement

```
SELECT customer_name, city
FROM ds9_4;
```

Result Set

```
CUSTOMER_NAME   |CITY      |
----------------|----------|
Bargains Galore|Detroit   |
RedHot Discount|San Diego|
NewTown Deals  |Dallas    |
Ajax Sales     |Detroit   |
BigBox Direct  |Chicago   |
```

What is the purpose of the WHERE clause?

Example 9-5 The WHERE clause restricts which rows are included in the result set

Data Set ds9_5

```
ID |CUSTOMER_NAME   |CITY      |CREDIT_LIMIT|
---|----------------|----------|------------|
101|Bargains Galore|Detroit   |      125000|
102|RedHot Discount|San Diego|      185500|
103|NewTown Deals   |Dallas    |      132500|
104|Ajax Sales      |Detroit   |      150000|
105|BigBox Direct   |Chicago   |      145000|
```

SQL Statement

```
SELECT customer_name
FROM ds9_5
WHERE city = 'Detroit';
```

Result Set

```
CUSTOMER_NAME
--------------
Bargains Galore
Ajax Sales
```

Calculated Value in a WHERE Clause

Example 9-6

Return customer_name as "Customer" and available_credit (credit_limit – balance) for all rows where available credit is greater than 25,000.

Data Set ds9_6

```
ID  |CUSTOMER_NAME   |CREDIT_LIMIT|BALANCE   |
----|----------------|------------|----------|
101|Bargains Galore|      125000|110083.00|
102|RedHot Discount|      185500|120387.00|
103|NewTown Deals  |      132500|109635.00|
104|Ajax Sales     |      150000|118630.00|
105|BigBox Direct  |      145000|123122.00|
```

SQL Statement

```
SELECT customer_name AS "Customer",
       credit_limit - balance AS "Available Credit"
FROM ds9_6
WHERE credit_limit - balance > 25000;
```

Result Set

```
Customer        |Available Credit|
----------------|----------------|
RedHot Discount|        65113.00|
Ajax Sales     |        31370.00|
```

Alias Name Cannot be used in WHERE Clause

Example 9-7 Forcing an error by using an alias name on the WHERE clause

Data Set ds9_7

```
ID  |CUSTOMER_NAME   |CREDIT_LIMIT|BALANCE   |
----|----------------|------------|----------|
101|Bargains Galore|      125000|110083.00|
102|RedHot Discount|      185500|120387.00|
103|NewTown Deals  |      132500|109635.00|
104|Ajax Sales     |      150000|118630.00|
105|BigBox Direct  |      145000|123122.00|
```

SQL Statement

```
SELECT customer_name AS "Customer",
       credit_limit - balance AS "Available Credit"
FROM ds9_7
WHERE "Available Credit";
```

Result Set

```
SQL Error [42703]: [SQL0206] Column or global variable
Available Credit not found.
```

SQL Statement Order of Execution

Order	Clause	Function
1	FROM	Connects to database tables
2	WHERE	Filters or restricts rows
3	SELECT	Filters or restricts columns

SQL Comparison Operators in the WHERE Clause

- Additional SQL comparison operators that add functionality for retrieving specific sets of data:

SQL Comparison Operator	Description
IS NULL	Condition tests for NULL value
BETWEEN..AND	Returns rows based on a range of values
LIKE	Returns rows that match a character pattern
IN	Tests for values in a specified list of values

- NOT can be used in front of these operators

BETWEEN…AND Operator

- Returns a result set based on a range of values
- Lower-limit must be listed first

```
SELECT last_name, salary
  FROM employees
  WHERE salary BETWEEN 9000 AND 11000;
```

LAST_NAME	SALARY
Zlotkey	10500
Abel	11000
Hunold	9000

Inclusive Values

- Values specified with the BETWEEN condition are said to be inclusive
- Inclusive - The values returned include the lower-limit value and the upper-limit value

```
SELECT last_name, salary
  FROM employees
  WHERE salary BETWEEN 9000 AND 11000;
```

LAST_NAME	SALARY
Zlotkey	10500
Abel	11000
Hunold	9000

BETWEEN…AND Operator

- Both examples below are the same
- No performance benefit
- Simplicity in reading the code

```
SELECT title, year
FROM d_cds
WHERE year BETWEEN 1999 AND 2001;
```

or

```
WHERE year >= 1999 AND year <= 2001;
```

Example 9-8

Use the BETWEEN operator to return customer_name, and credit_limit for all rows where credit_limit is between 145000 and 180000 inclusive.

Data Set ds9_8

```
ID |CUSTOMER_NAME   |CREDIT_LIMIT|
---|----------------|------------|
101|Bargains Galore|      125000|
102|RedHot Discount|      185500|
103|NewTown Deals  |      132500|
104|Ajax Sales     |      150000|
105|BigBox Direct  |      145000|
106|Mainstreet Inc |      200000|
107|Riverside Mfg  |      164000|
108|Cube Industries|      185000|
109|LowCost Shops  |      155000|
110|Sterling Mfg   |      180000|
```

SQL Statement

```
SELECT customer_name, credit_limit
FROM ds9_8
WHERE credit_limit BETWEEN 145000 AND 180000;
```

Result Set

```
CUSTOMER_NAME|CREDIT_LIMIT|
-------------|------------|
Ajax Sales   |      150000|
BigBox Direct|      145000|
Riverside Mfg|      164000|
LowCost Shops|      155000|
Sterling Mfg |      180000|
```

NOT Keyword with BETWEEN…AND Operator

Example 9-9

Use the NOT BETWEEN operator to return customer_name, and credit_limit for all rows where credit_limit is not between 145000 and 180000

Data Set ds9_9

```
ID |CUSTOMER_NAME   |CREDIT_LIMIT|
---|----------------|------------|
101|Bargains Galore|      125000|
102|RedHot Discount|      185500|
103|NewTown Deals   |      132500|
104|Ajax Sales      |      150000|
105|BigBox Direct   |      145000|
106|Mainstreet Inc  |      200000|
107|Riverside Mfg   |      164000|
108|Cube Industries|      185000|
109|LowCost Shops   |      155000|
110|Sterling Mfg    |      180000|
```

SQL Statement

```
SELECT customer_name, credit_limit
FROM ds9_9
WHERE credit_limit NOT BETWEEN 145000 AND 180000;
```

Result Set

```
CUSTOMER_NAME   |CREDIT_LIMIT|
----------------|------------|
Bargains Galore|      125000|
RedHot Discount|      185500|
NewTown Deals   |      132500|
Mainstreet Inc  |      200000|
Cube Industries|      185000|
```

IN Operator

- Returns rows that match a value in a specified list
- "Membership condition"
- List must be in brackets/parentheses
- Values are separated by commas

```
SELECT city, state_province, country_id
FROM locations
WHERE country_id IN('UK', 'CA');
```

CITY	STATE_PROVINCE	COUNTRY_ID
Toronto	Ontario	CA
Oxford	Oxford	UK

Example 9-10

Use the IN operator to return customer_name, city, and credit_limit for all rows where city is Detroit or San Diego

Data Set ds9_10

```
ID |CUSTOMER_NAME   |CITY     |CREDIT_LIMIT|
---|----------------|---------|------------|
101|Bargains Galore|Detroit  |      125000|
102|RedHot Discount|San Diego|      185500|
103|NewTown Deals  |Dallas   |      132500|
104|Ajax Sales     |Detroit  |      150000|
105|BigBox Direct  |Chicago  |      145000|
106|Mainstreet Inc |Dallas   |      200000|
107|Riverside Mfg  |Chicago  |      164000|
108|Cube Industries|Dallas   |      185000|
109|LowCost Shops  |San Diego|      155000|
110|Sterling Mfg   |Chicago  |      180000|
```

SQL Statement

```
SELECT customer_name, city
FROM   ds9_10
WHERE  city IN ( 'Detroit', 'San Diego' );
```

Result Set

```
CUSTOMER_NAME   |CITY     |
----------------|---------|
Bargains Galore|Detroit  |
RedHot Discount|San Diego|
Ajax Sales     |Detroit  |
LowCost Shops  |San Diego|
```

Using OR Instead of IN Operator

- The IN operator could also be written as a set of OR conditions:

```
SELECT city, state_province, country_id
FROM locations
WHERE country_id IN('UK', 'CA');
…
WHERE country_id = 'UK' OR country_id = 'CA';
```

- The IN condition can be written using either syntax just as efficiently

LIKE Operator

- Two symbols, called wildcard characters, are used to to perform string pattern matching
  - Underscore (_) symbol represents one character
  - Percent (%) symbol represents any sequence of zero or more characters
- Allows the selection of rows that match either characters, dates, or number patterns

Example 9-12 Retrieve customer_name for all rows where customer name contains 'Street'

Data Set ds9_12

```
ID  |CUSTOMER_NAME       |
----|--------------------|
401 |Backstreet Inc.     |
402 |Streetlight Inc.    |
403 |Crosstown Mfg.      |
404 |Ben Street Inc.     |
405 |BigBox Direct       |
406 |Alex Streeting Inc. |
```

SQL Statement

```
SELECT customer_name
FROM   ds9_12
WHERE  customer_name LIKE '%Street%'
```

Result Set

```
CUSTOMER_NAME       |
--------------------|
Streetlight Inc.    |
Ben Street Inc.     |
Alex Streeting Inc. |
```

Example 9-13	Retrieve first_name for all rows where first_name begins with the letter 'A'

Data Set ds9_13

```
FIRST_NAME|LAST_NAME|
----------|---------|
Andrew    |North    |
Candice   |Wilson   |
andy      |davis    |
Amanda    |Miller   |
Marland   |Power    |
Jackie    |King     |
Chandler  |Jacob    |
```

SQL Statement

```
SELECT first_name
FROM   ds9_13
WHERE  first_name LIKE 'A%';
```

Result Set

```
FIRST_NAME|
----------|
Andrew    |
Amanda    |
```

Example 9-14 Retrieve first_name for all rows where first_name contains the letters 'and'

Data Set ds9_14

```
FIRST_NAME|LAST_NAME|
----------|---------|
Andrew    |North    |
Candice   |Wilson   |
andy      |davis    |
Amanda    |Miller   |
Marland   |Power    |
Jackie    |King     |
Chandler  |Jacob    |
```

SQL Statement

```
SELECT first_name
FROM   ds9_14
WHERE  first_name LIKE '%and%';
```

Result Set

```
FIRST_NAME|
----------|
Candice   |
andy      |
Amanda    |
Marland   |
Chandler  |
```

Example 9-15

Use underscore (_) symbol to retrieve first_name for all rows where first_name contains the letters 'and' in positions 2-4

Data Set ds9_15

```
FIRST_NAME|LAST_NAME|
----------|---------|
Andrew    |North    |
Candice   |Wilson   |
andy      |davis    |
Amanda    |Miller   |
Marland   |Power    |
Jackie    |King     |
Chandler  |Jacob    |
```

SQL Statement

```
SELECT first_name
FROM   ds9_15
WHERE  first_name LIKE '_and%';
```

Result Set

```
FIRST_NAME
----------
Candice
```

LIKE Operator

- Underscore (_) symbol represents one character
- Percent (%) symbol represents a string of zero or more characters

- Return all employees with last names beginning with any letter followed by an "o" and then followed by any other number of letters

LIKE Operator Example

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

- Which of the following last names would be returned from the above query?
  - 1. Sommersmith
  - 2. Oog
  - 3. Fong
  - 4. Mo

LIKE Operator Example

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

- Which of the following last names would be returned from the above query?
  - 1. Sommersmith
  - 2. Oog
  - 3. Fong
  - 4. Mo
- If you said ALL 4, you are correct!

LIKE Operator – ESCAPE option

- What if the % or _ symbols are not a wildcard but part of the value being searched
- The ESCAPE option is used to indicate that the _ or % symbol is part of the value, not a wildcard value
- The ESCAPE option character can be any character, but backslash (\) is recommended

LIKE Operator ESCAPE option

```
DROP TABLE categories;

CREATE TABLE categories(
     cat_id   VARCHAR(7)   NOT NULL,
     cat_name VARCHAR(50)  NOT NULL,
   CONSTRAINT categories_pk
     PRIMARY KEY(cat_id) );

INSERT INTO categories VALUES
('LHA-123', 'Large Home Appliances'),
('SHA_987', 'Small Home Appliances'),
('LKA-456', 'Large Kitchen Appliances'),
('SHA-285', 'Small Home Kitchen Appliances');

SELECT cat_id, cat_name FROM categories
WHERE cat_id LIKE '%\_%' ESCAPE '\';
```

```
CAT_ID  CAT_NAME
------- --------------------
SHA_987 Small Home Appliances
```

LIKE Operator – ESCAPE option: An Example

- Retrieve JOB_IDs from the employees table that contain the pattern _R

LAST_NAME	JOB_ID
King	AD_PRES
Higgins	AC_MGR
Abel	SA_REP
Taylor	SA_REP
Grant	SA_REP
Rajs	ST_CLERK
Davies	ST_CLERK
Matos	ST_CLERK
Vargas	ST_CLERK
Hunold	IT_PROG
Ernst	IT_PROG
Lorentz	IT_PROG
Fay	MK_REP

LIKE Operator – ESCAPE option: An Example

- Without the ESCAPE option, all employees that have an R in JOB_ID are returned

```
SELECT last_name, job_id
FROM employees
WHERE job_id LIKE '%_R%';
```

LAST_NAME	JOB_ID
King	AD_PRES
Higgins	AC_MGR
Abel	SA_REP
Taylor	SA_REP
Grant	SA_REP
Rajs	ST_CLERK
Davies	ST_CLERK
Matos	ST_CLERK
Vargas	ST_CLERK
Hunold	IT_PROG
Ernst	IT_PROG
Lorentz	IT_PROG
Fay	MK_REP

LIKE Operator – ESCAPE option: An Example

- With the ESCAPE option, only employees that have an _R in JOB_ID are returned

```
SELECT last_name, job_id
FROM employees
WHERE job_id LIKE '%\_R%' ESCAPE '\';
```

```
LAST_NAME  JOB_ID
---------  ------
Abel       SA_REP
Taylor     SA_REP
Grant      SA_REP
Fay        MK_REP
```

IS NULL, IS NOT NULL Operators

IS NULL, IS NOT NULL

- NULL = unknown
- Allows a test for unknown
  - All the dates in June that do not have an event scheduled
  - Clients that do not have an email address
- Requires use of IS NULL operator

NULL Column Value in WHERE Clause

- When comparing NULLS you CANNOT use the equals (=) or not equals (<>) operators

```
SELECT last_name, manager_id
  FROM employees
  WHERE manager_id = NULL;   - will not work
```

- The query will run, but returns no data, as the actual value of NULL is unknown, so how can we check if something is equal or not equal to a value that is unknown

IS NULL, IS NOT NULL

- `IS NULL` tests for data that is unknown (contains NULL)

- `IS NOT NULL` tests for the existence of data; That is, data that contains a value (not NULL)

Example 9-17

Return company name for all customers that have not been assigned a discount rate. That is, discount is unknown or NULL.

Data Set ds9_17

```
ID |CUSTOMER_NAME   |CITY      |DISCOUNT|
---|----------------|----------|--------|
101|Bargains Galore|Detroit   |    0.07|
102|RedHot Discount|San Diego|    0.10|
103|NewTown Deals   |Dallas    |    NULL|
104|Ajax Sales      |Detroit   |    0.10|
105|BigBox Direct   |Chicago   |    NULL|
```

SQL Statement

```
SELECT customer_name
FROM   ds9_17
WHERE  discount IS NULL;
```

Result Set

```
CUSTOMER_NAME|
-------------|
NewTown Deals|
BigBox Direct|
```

Example 9-18

Return company name for all customers that have been assigned a discount rate. That is, discount is NOT NULL.

Data Set ds9_18

ID	CUSTOMER_NAME	CITY	DISCOUNT
101	Bargains Galore	Detroit	0.07
102	RedHot Discount	San Diego	0.10
103	NewTown Deals	Dallas	NULL
104	Ajax Sales	Detroit	0.10
105	BigBox Direct	Chicago	NULL

SQL Statement

```
SELECT customer_name, discount
FROM   ds9_18
WHERE  discount IS NOT NULL;
```

Result Set

CUSTOMER_NAME	DISCOUNT
Bargains Galore	0.07
RedHot Discount	0.10
Ajax Sales	0.10

Remember: NULL Values

A common error is using

= NULL

which does not raise an error but also does not return any rows

Comparison Operators in the WHERE Clause

- Comparison operators can be used in all the following ways in the WHERE clause:

```
SELECT * FROM employees
WHERE job_id = 'IT_PROG';

SELECT * FROM employees
WHERE salary > 10500;

SELECT * FROM employees
WHERE hire_date > '1995-01-15';
```

Logical Operators

- NOT Returns TRUE if the condition is false
- AND Returns TRUE if both conditions are true
- OR Returns TRUE if either condition is true

- Evaluated in order of NOT, AND, OR

Logical Operators

- Logical Operators: NOT, AND, OR
  - AND and OR combine the result of two or more conditions to produce a single result
  - NOT can be one condition
  - A row is returned ONLY IF the overall result of the condition is TRUE

AND Operator

Example 9-19 Use the AND operator to return a list of customers that are located in Dallas and have a credit limit greater than 150,000

Data Set ds9_19

ID	CUSTOMER_NAME	CITY	CREDIT_LIMIT
101	Bargains Galore	Detroit	125000
102	RedHot Discount	San Diego	185500
103	NewTown Deals	Dallas	132500
104	Ajax Sales	Detroit	150000
105	BigBox Direct	Chicago	145000
106	Mainstreet Inc	Dallas	200000
107	Riverside Mfg	Chicago	164000
108	Cube Industries	Dallas	185000

SQL Statement

```
SELECT customer_name, credit_limit
FROM   ds9_19
WHERE  city = 'Dallas' AND  credit_limit > 150000;
```

Result Set

CUSTOMER_NAME	CREDIT_LIMIT
Mainstreet Inc	200000
Cube Industries	185000

OR Operator

- OR condition - the result set contains rows that satisfy either one of the OR conditions

Example 9-20

Use the OR operator to return a list of customers that are located in Dallas OR customers that have a credit limit greater than 150,000

Data Set ds9_20

```
ID |CUSTOMER_NAME   |CITY     |CREDIT_LIMIT|
---|----------------|---------|------------|
101|Bargains Galore|Detroit  |      125000|
102|RedHot Discount|San Diego|      185500|
103|NewTown Deals  |Dallas   |      132500|
104|Ajax Sales     |Detroit  |      150000|
105|BigBox Direct  |Chicago  |      145000|
106|Mainstreet Inc |Dallas   |      200000|
107|Riverside Mfg  |Chicago  |      164000|
108|Cube Industries|Dallas   |      185000|
```

SQL Statement

```
SELECT customer_name, city, credit_limit
FROM   ds9_20
WHERE  city = 'Dallas' OR  credit_limit > 150000;
```

Result Set

```
CUSTOMER_NAME  |CITY     |CREDIT_LIMIT|
---------------|---------|------------|
RedHot Discount|San Diego|      185500|
NewTown Deals  |Dallas   |      132500|
Mainstreet Inc |Dallas   |      200000|
Riverside Mfg  |Chicago  |      164000|
Cube Industries|Dallas   |      185000|
```

Order of Operations with Logical Operators

ORDER	OPERATORS
1	Arithmetic + - * /
2	Concatenation ||
3	Comparison <, <=, >, >=, <>
4	IS (NOT) NULL, LIKE, (NOT) IN
5	(NOT) BETWEEN
6	NOT
7	AND
8	OR

Order of Operations – What Happens First

- In what order are the expressions evaluated?

```
SELECT first_name, last_name, job_id, salary * 1.05 AS raise
FROM employees
WHERE job_id IN('ST_CLERK', 'MK_REP') AND first_name LIKE 'P%'
      OR last_name LIKE '%z%';
```

Order of Operations – What Happens First

- First, the AND condition is evaluated, so all employees with job_id ST_CLERK or MK_REP, AND who have a first name starting with "P" are returned
- The OR clause is then evaluated and returns employees whose last name contains "z"

```
SELECT first_name, last_name, job_id, salary * 1.05 AS raise
FROM employees
WHERE job_id IN('ST_CLERK', 'MK_REP') AND first_name LIKE 'P%'
      OR last_name LIKE '%z%';
```

Order of Operations – What Happens First

```
SELECT first_name, last_name, job_id, salary * 1.05 AS raise
FROM employees
WHERE job_id IN('ST_CLERK', 'MK_REP') AND first_name LIKE 'P%'
      OR last_name LIKE '%z%';
```

FIRST_NAME	LAST_NAME	JOB_ID	RAISE
William	Gietz	AC_ACCOUNT	8715.0000
Peter	Vargas	ST_CLERK	2625.0000
Diana	Lorentz	IT_PROG	4410.0000
Pat	Fay	MK_REP	6300.0000

Order of Operations – What Happens First

- In this example, the order of the OR and AND have been reversed from the previous slide
- The order of operations is:
  1. first_name starts with "P" AND last_name contains "z". Both these conditions must be met to be returned
  2. Any instance of employees with job_id ST_CLERK or MK_REP will be returned

```
SELECT first_name, last_name, job_id, salary * 1.05 AS raise
FROM employees
WHERE job_id IN('ST_CLERK', 'MK_REP') OR first_name LIKE 'P%'
      AND last_name LIKE '%z%';
```

Order of Operations – What Happens First

- Adding brackets/parenthesis changes the way the WHERE clause is evaluated, and the rows returned
- The order of operations is:
  1. The values in the brackets are selected
  2. All instances of the values in the brackets AND contain the letter "z" in their last_name are returned

```
SELECT first_name, last_name, job_id, salary * 1.05 AS raise
FROM employees
WHERE ( job_id IN('ST_CLERK', 'MK_REP') OR first_name LIKE 'P%' )
      AND last_name LIKE '%z%';
```

Example 9-21 Order of Operations

Return city, customer name, credit limit, and balance based on the following conditions:

- Customers in Chicago that have a balance >= 120000
- Any customer with a credit limit >= 185000

Data Set ds9_21

```
ID |CUSTOMER_NAME   |CITY     |CREDIT_LIMIT|BALANCE   |
---|---------------|---------|------------|---------|
101|Bargains Galore|Detroit  |      125000|110083.00|
102|RedHot Discount|San Diego|      185500|120387.00|
103|NewTown Deals  |Dallas   |      132500|109635.00|
104|Ajax Sales     |Detroit  |      150000|118630.00|
105|BigBox Direct  |Chicago  |      145000|123122.00|
106|Mainstreet Inc |Dallas   |      200000|116588.00|
107|Riverside Mfg  |Chicago  |      164000|110230.00|
108|Cube Industries|Dallas   |      185000|119525.00|
109|LowCost Shops  |San Diego|      155000|106687.00|
110|Sterling Mfg   |Chicago  |      180000|122429.00|
```

SQL Statement

```
SELECT  city, customer_name, credit_limit, balance
FROM  ds9_21
WHERE ( city = 'Chicago' AND balance >= 120000 )
   OR credit_limit >= 185000
ORDER BY city;
```

Result Set

```
CITY     |CUSTOMER_NAME   |CREDIT_LIMIT|BALANCE
---------|---------------|------------|---------
Chicago  |BigBox Direct  |      145000|123122.00
Chicago  |Sterling Mfg   |      180000|122429.00
Dallas   |Mainstreet Inc |      200000|116588.00
Dallas   |Cube Industries|      185000|119525.00
San Diego|RedHot Discount|      185500|120387.00
```

NOT Operator to Negate a Condition

- The NOT operator returns rows that do NOT satisfy the condition in the WHERE clause

Example 9-22

Use NOT operator to negate a search condition that returns a list of customers that are not located in Dallas

Data Set ds9_22

```
ID |CUSTOMER_NAME   |CITY     |
---|----------------|---------|
101|Bargains Galore|Detroit  |
102|RedHot Discount|San Diego|
103|NewTown Deals   |Dallas   |
104|Ajax Sales      |Detroit  |
105|BigBox Direct   |Chicago  |
106|Mainstreet Inc  |Dallas   |
```

SQL Statement

```
SELECT customer_name, city
FROM   ds9_22
WHERE  NOT (city = 'Dallas');
```

Result Set

```
CUSTOMER_NAME   |CITY     |
----------------|---------|
Bargains Galore|Detroit  |
RedHot Discount|San Diego|
Ajax Sales      |Detroit  |
BigBox Direct   |Chicago  |
```

NOT IN Operator

Example 9-23

Return company name and city for customers not located in Dallas, Chicago, and Detroit

Data Set ds9_23

```
ID |CUSTOMER_NAME   |CITY     |
---|---------------|---------|
101|Bargains Galore|Detroit  |
102|RedHot Discount|San Diego|
103|NewTown Deals  |Dallas   |
104|Ajax Sales     |Detroit  |
105|BigBox Direct  |Chicago  |
106|Mainstreet Inc |Dallas   |
107|Riverside Mfg  |Chicago  |
108|Cube Industries|Dallas   |
109|LowCost Shops  |San Diego|
110|Sterling Mfg   |Chicago  |
```

SQL Statement

```
SELECT customer_name, city
FROM   ds9_23
WHERE  city NOT IN ('Dallas', 'Chicago', 'Detroit');
```

Result Set

```
CUSTOMER_NAME  |CITY     |
---------------|---------|
RedHot Discount|San Diego|
LowCost Shops  |San Diego|
```

WHERE Clause Rules

- Character strings and dates are enclosed in single quotation marks (' ')
- Numbers are not enclosed in single quotation marks
- All character searches are case-sensitive
- An alias cannot be used in the WHERE clause – Why?


Q & A
Questions &
Answers