DATABASE DESIGN & IMPLEMENTATION

OBJECTIVES

- ANATOMY OF AN SQL STATEMENT
- ARITHMETIC IN THE SELECT CLAUSE
- NULL VALUES IN ARITHMETIC
- COLUMN ALIASES
- CONCATENATION
- CONSTRUCT QUERY TO SORT A RESULT SET IN ASCENDING OR DESCENDING ORDER
- CONSTRUCT A QUERY TO ORDER A RESULT SET USING A COLUMN ALIAS
- CONSTRUCT A QUERY TO ORDER A RESULT SET FOR SINGLE OR MULTIPLE COLUMNS

ANATOMY OF A SQL STATEMENT

- SELECT IS ONE OF THE MOST IMPORTANT, IF NOT THE MOST IMPORTANT KEYWORD IN SQL.
- SELECT ALLOWS YOU TO SEARCH FOR SPECIFIC DATA IN A DATABASE
- THE SELECT STATEMENT MUST CONTAIN A SELECT CLAUSE AND A FROM CLAUSE
- THE LIST OF COLUMNS IN A SELECT CLAUSE ALLOWS YOU TO CONDUCTION PROJECTION I.E. COLUMNS IN A TABLE.
- THE WHERE CLAUSE ALLOWS YOU TO CONDUCTION SELECTION I.E. ROWS IN A TABLE.
- SELECT * FROM TABLENAME MEANS THAT YOU WANT TO SEE ALL COLUMNS FROM A TABLE

ARITHMETIC IN SELECT CLAUSE

- YOU CAN CONSTRUCT A SELECT CLAUSE THAT CONTAINS ARITHMETIC
- YOU MAY WANT TO MODIFY THE WAY DATA IS DISPLAYED, PERFORM CALCULATIONS ETC
- WE ARE NOT CREATING NEW COLUMNS FOR THESE CALCULATIONS OR CHANGING THE DATA IN THE DATABASE.
- THE RESULTS APPEAR ONLY IN THE OUTPUT

```
SELECT LAST_NAME, SALARY, SALARY = 300

FROM EMPLOYEE;
```

ARITHMETIC IN THE SELECT CLAUSE

- PRECEDENCE IS THE ORDER IN WHICH THE DATABASE MANAGEMENT SYSTEM EVALUATES THE OPERATORS IN THE SAME EXPRESSION.
- ORACLE EVALUATES OPERATORS WITH HIGHER PRECEDENCE FIRST * / + -
- ORACLE EVALUATES OPERATORS WITH EQUAL PRECEDENCE FROM LEFT TO RIGHT WITHIN AN EXPRESSION.
- YOU CAN USE PARENTHESES TO FORCE THE EXPRESSION WITHIN THE PARENTHESES TO BE EVALUATED FIRST.

```
SELECT LAST_NAME, SALARY, 12*SALARY + 100 FROM EMPLOYEES;

SELECT LAST_NAME, SALARY, 12*(SALARY +100) FROM EMPLOYEES;
```

NULL VALUES IN ARITHMETIC

- IN SQL, NULL IS NOT ZERO OR SPACE, IN SQL, ZERO IS A NUMBER AND SPACE IS A CHARACTER.
- IF ANY COLUMN VALUE IN AN ARITHMETIC EXPRESSION IS NULL, THE RESULT IS NULL.
- IF YOU TRY TO DIVIDE BY A NULL VALUE, THE RESULT IS NULL.
- IF YOU TRY TO DIVIDE BY ZERO YOU GET AN ERROR.

```
SELECT LAST_NAME, JOB_ID, SALARY, COMMISSION_PCT,
       SALARY*COMMISSION_PCT
FROM EMPLOYEES;
```

- WHERE THERE WAS A NULL VALUE IN COMMISSION_PCT WOULD RESULT IN A NULL VALUE IN THE LAST COLUMN

COLUMN ALIASES

- AN ALIAS IS A WAY OF RENAMING A COLUMN HEADING IN THE OUTPUT.
- WITHOUT ALIASES, WHEN THE RESULT OF A SQL STATEMENT IS DISPLAYED, THE NAME OF THE COLUMNS DISPLAYED WILL BE THE SAME AS THE COLUMN NAMES IN THE TABLE OR A NAME SHOWING AN ARITHMETIC OPERATION SUCH AS 12*(SALARY +100)
- YOU WILL WANT YOUR OUTPUT TO DISPLAY IN A MORE USER FRIENDLY WAY

COLUMN ALIASES

- A COLUMN ALIAS:
  - RENAMES A COLUMN HEADING
  - IS USEFUL WITH CALCULATIONS
  - IMMEDIATELY FOLLOWS THE COLUMN NAME IN THE SELECT CLAUSE
  - MAY HAVE THE OPTIONAL AS KEYWORD BETWEEN THE COLUMN NAME AND ALIAS
  - REQUIRES DOUBLE QUOTATION MARKS IF THE ALIAS CONTAINS SPACES OR SPECIAL CHARACTERS, OR IS CASE-SENSITIVE

COLUMN ALIASES

```
SELECT * | COLUMN | EXPR [AS ALIAS], …
FROM TABLENAME;

SELECT LAST_NAME AS NAME,
      COMMISSION_PCT AS COMMISSION
FROM EMPLOYEES;

SELECT LAST_NAME “NAME”,
      COMMISSION_PCT “COMMISSION PERCENTAGE”
FROM EMPLOYEES;
```

CONCATENATION

- CONCATENATION MEANS TO CONNECT OR LINK TOGETHER IN A SERIES.
- THE SYMBOL IS 2 VERTICAL BARS SOMETIMES KNOWN AS PIPES
- VALUES ON EITHER SIDE OF THE PIPES ARE COMBINED TO MAKE A SINGLE OUTPUT COLUMN
- SYNTAX:

  STRING1 || STRING2 || STRINGN
- CONCATENATION IS USED TO PRODUCE READABLE DATA OUTPUT

```
SELECT DEPARTMENT_ID || ‘ ‘ || DEPARTMENT_NAME
FROM DEPARTMENTS;
```

CONCATENATION AND ALIASES

- COLUMN ALIASES ARE USEFUL WHEN USING THE CONCATENATION OPERATOR TO ENSURE THE HEADING IS READABLE

```
SELECT DEPARTMENT_ID || ‘ ‘ || DEPARTMENT_NAME AS “DEPARTMENT
INFO”

FROM DEPARTMENTS;
```

CONCATENATION AND LITERAL VALUES

- A LITERAL VALUE IS A FIXED DATA VALUE SUCH AS A CHARACTER, NUMBER OR DATE.
- ‘DOLLARS’ 1000 ‘JANUARY 1, 2009’ (NUMBER DO NOT NEED QUOTES)
- YOU CAN CREATE OUTPUT THAT LOOKS LIKE A SENTENCE OR STATEMENT.

```
SELECT LAST_NAME || ‘ HAS A MONTHLY SALARY OF ‘ || SALARY ||’
DOLLARS.’ AS PAY

FROM EMPLOYEES;
```

DISTINCT

- YOU WILL WANT TO ELIMINATE DUPLICATE ROWS
- FOR EXAMPLE IF YOU SELECT ALL THE DEPARTMENT ID’S FROM THE EMPLOYEES TABLE IT WILL OUTPUT MANY ROWS THAT ARE THE SAME DEPARTMENT ID
- IF YOU WANT TO JUST SEE ONE ROW FOR EACH UNIQUE DEPARTMENT ID THEN YOU USE DISTINCT

```
SELECT DISTINCT DEPARTMENT_ID

FROM EMPLOYEES;
```

- DISTINCT AFFECTS ALL LISTED COLUMNS, RETURNING ROWS THAT A UNIQUE ACROSS ALL COLUMNS. THE KEYWORD MUST APPEAR FIRST IN SELECT CLAUSE

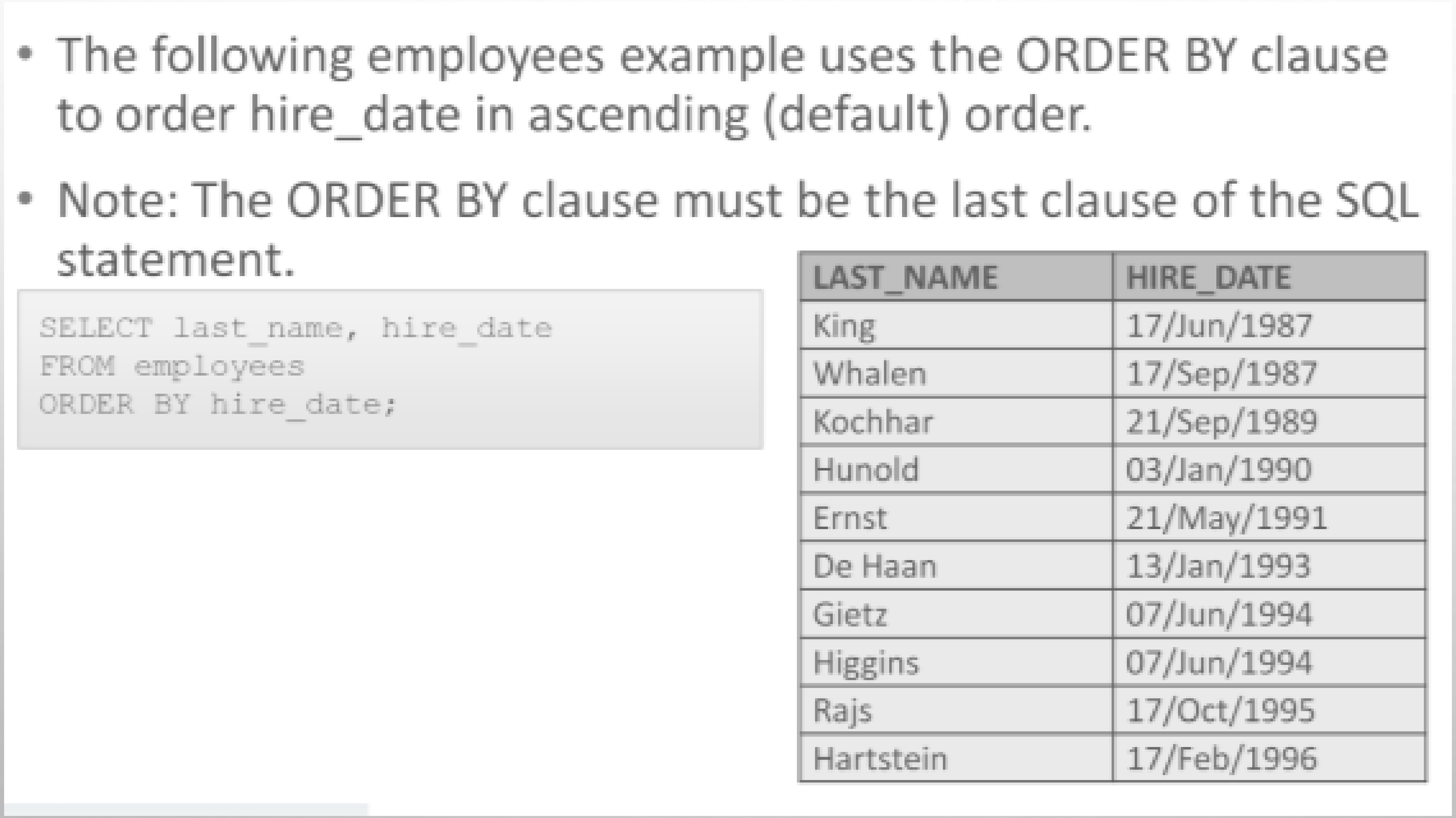
ORDER BY CLAUSE

- INFORMATION SORTED ASCENDING ORDER IS FAMILIAR TO US.
- IT'S WHAT MAKES LOOKING UP A NUMBER IN A PHONE BOOK, FINDING A WORD IN A DICTIONARY, OR LOCATING A HOUSE BY ITS STREET ADDRESS RELATIVELY EASY.
- SQL USES THE ORDER BY CLAUSE TO ORDER DATA.
- THE ORDER BY CLAUSE CAN SPECIFY SEVERAL WAYS IN WHICH TO ORDER ROWS RETURNED IN A QUERY.

ORDER BY CLAUSE

- THE DEFAULT SORT ORDER IS ASCENDING.
- NUMERIC VALUES ARE DISPLAYED LOWEST TO HIGHEST.
- DATE VALUES ARE DISPLAYED WITH THE EARLIEST VALUE FIRST
- CHARACTER VALUES ARE DISPLAYED IN ALPHABETICAL ORDER
- NULL VALUES ARE DISPLAYED LAST IN ASCENDING ORDER AND FIRST IN DESCENDING ORDER
- NULLS FIRST SPECIFIES THAT NULL VALUES SHOULD BE RETURNED BEFORE NON-NULL VALUES.
- NULLS LAST FIRST SPECIFIES THAT NULL VALUES SHOULD BE RETURNED AFTER NON-NULL VALUES.
- YOU CAN SORT BY MORE THAN ONE COLUMN (SEPARATE WITH COMMAS).

ORDER BY CLAUSE

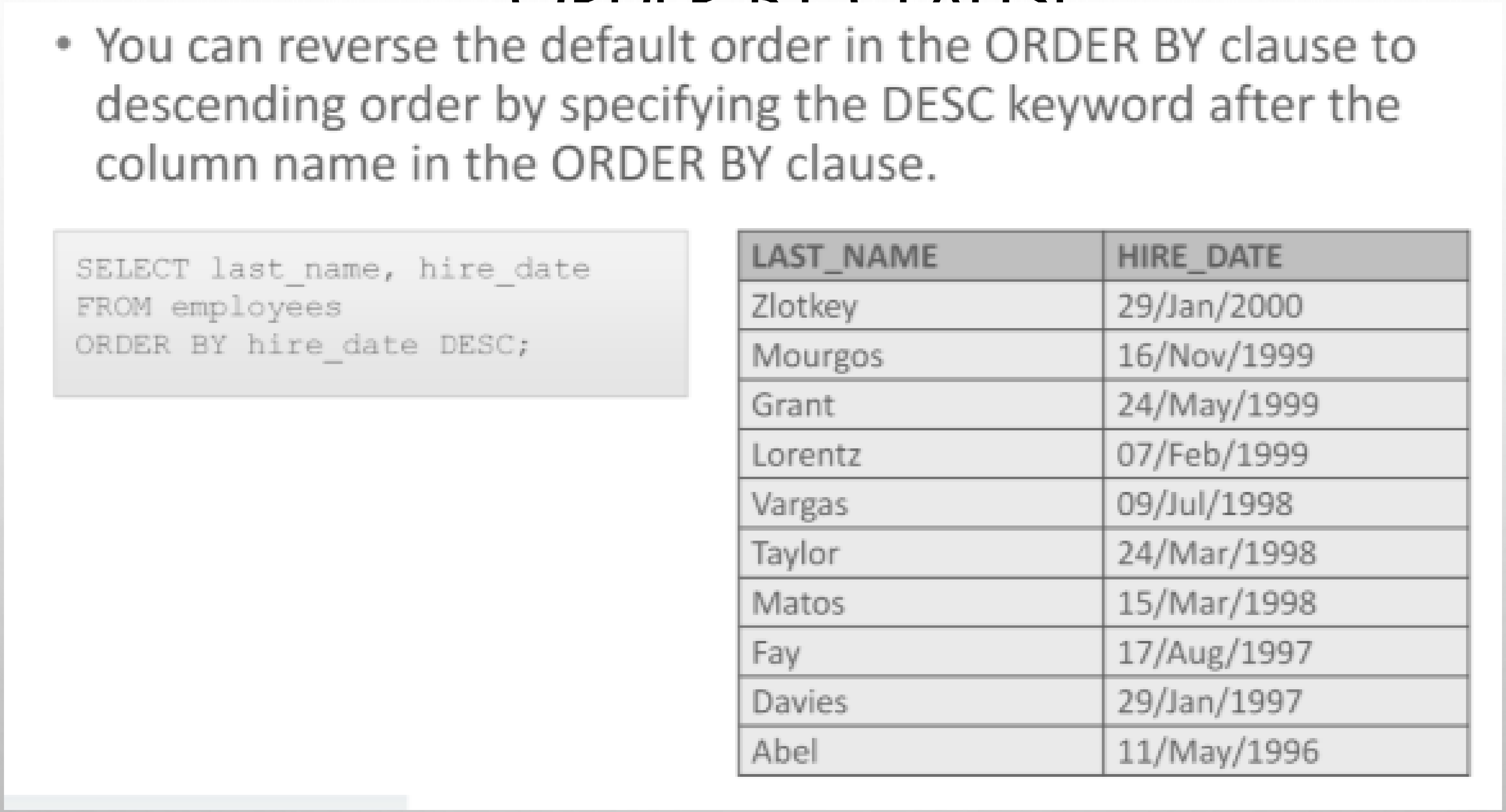


- The following employees example uses the ORDER BY clause to order hire_date in ascending (default) order.
- Note: The ORDER BY clause must be the last clause of the SQL statement.

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

LAST_NAME	HIRE_DATE
King	17/Jun/1987
Whalen	17/Sep/1987
Kochhar	21/Sep/1989
Hunold	03/Jan/1990
Ernst	21/May/1991
De Haan	13/Jan/1993
Gietz	07/Jun/1994
Higgins	07/Jun/1994
Rajs	17/Oct/1995
Hartstein	17/Feb/1996

ORDER BY CLAUSE

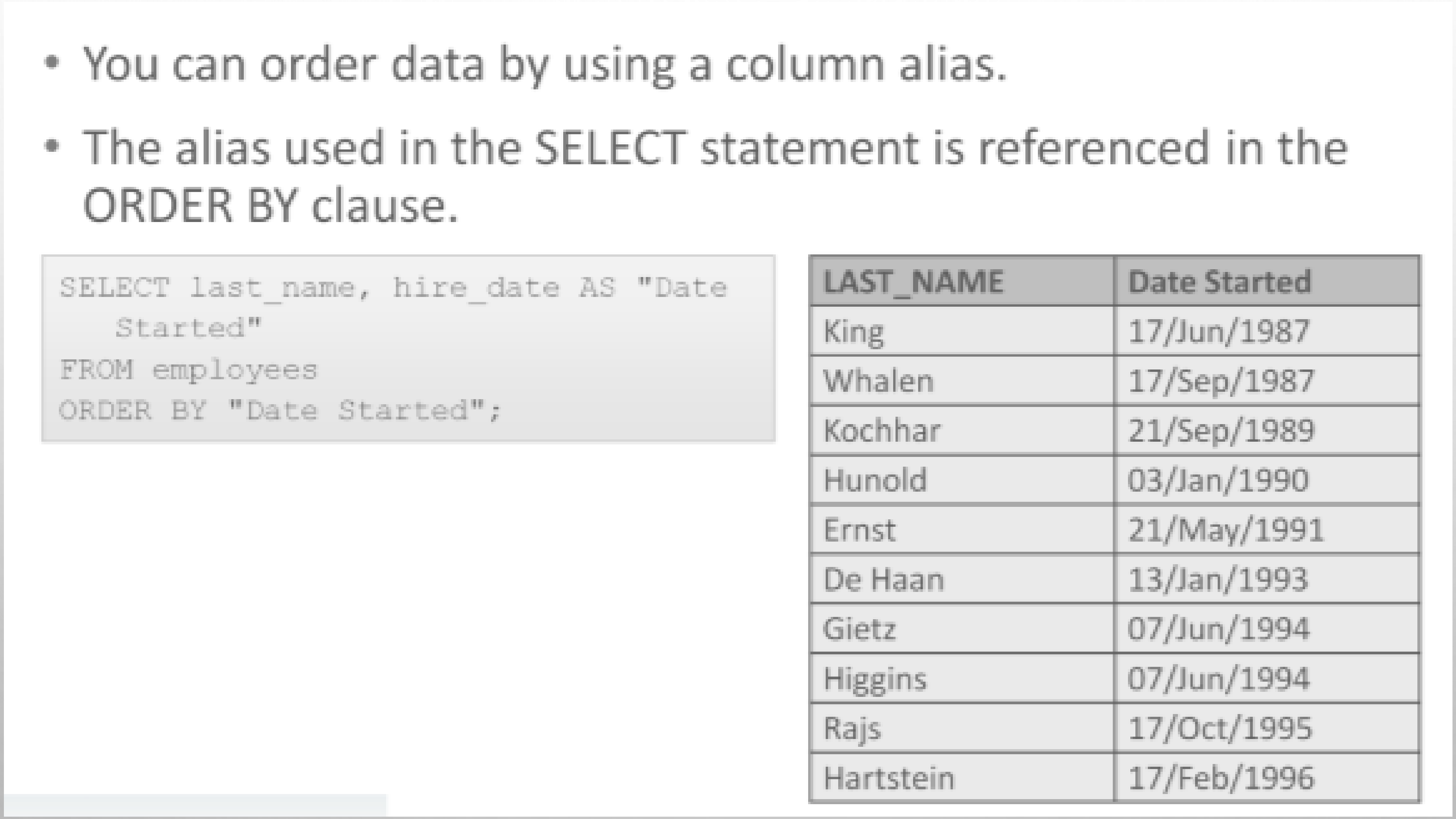


- You can reverse the default order in the ORDER BY clause to descending order by specifying the DESC keyword after the column name in the ORDER BY clause.

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

LAST_NAME	HIRE_DATE
Zlotkey	29/Jan/2000
Mourgos	16/Nov/1999
Grant	24/May/1999
Lorentz	07/Feb/1999
Vargas	09/Jul/1998
Taylor	24/Mar/1998
Matos	15/Mar/1998
Fay	17/Aug/1997
Davies	29/Jan/1997
Abel	11/May/1996

ORDER BY CLAUSE

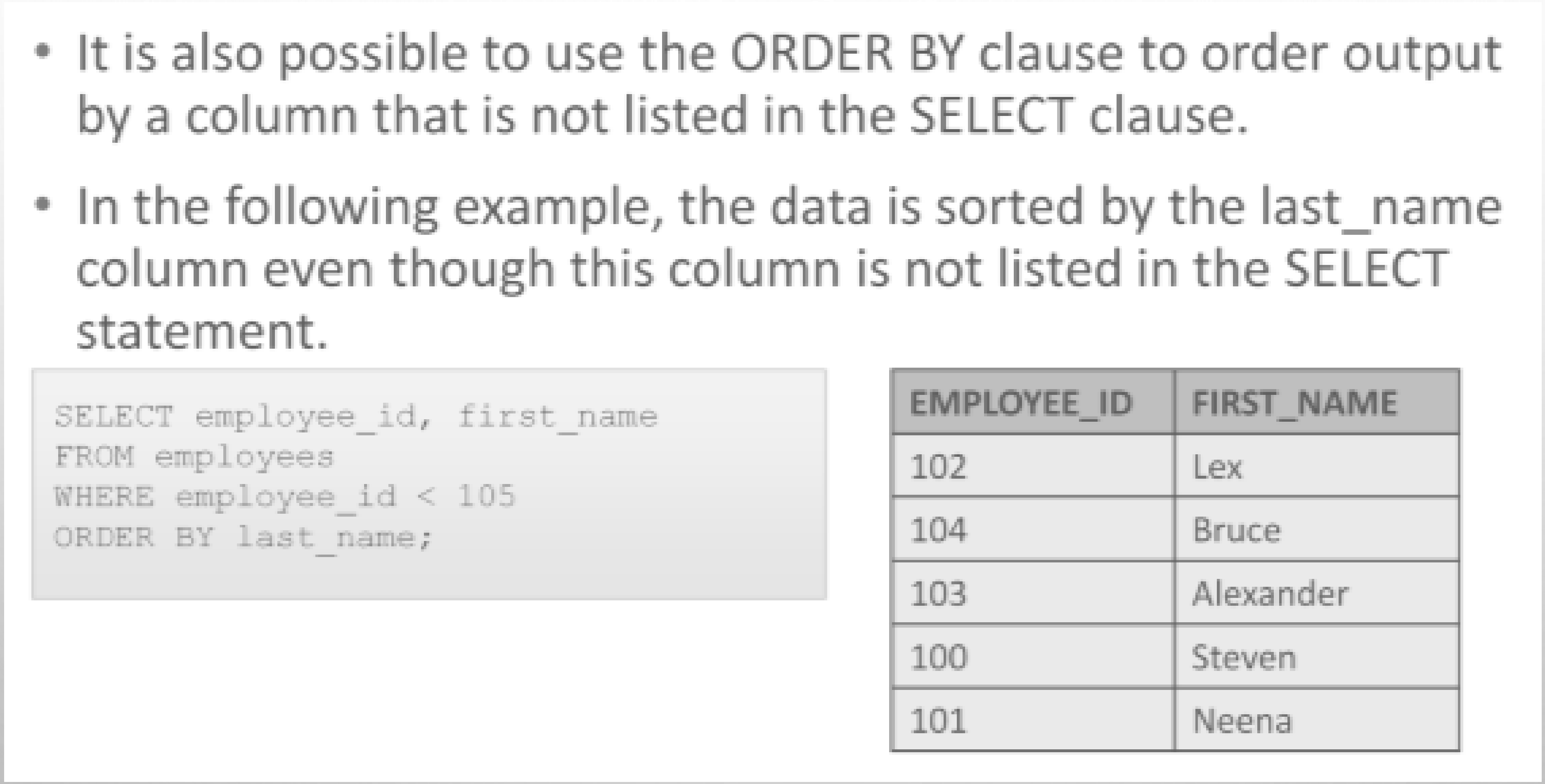


- You can order data by using a column alias.
- The alias used in the SELECT statement is referenced in the ORDER BY clause.

```
SELECT last_name, hire_date AS "Date
   Started"
FROM employees
ORDER BY "Date Started";
```

LAST_NAME	Date Started
King	17/Jun/1987
Whalen	17/Sep/1987
Kochhar	21/Sep/1989
Hunold	03/Jan/1990
Ernst	21/May/1991
De Haan	13/Jan/1993
Gietz	07/Jun/1994
Higgins	07/Jun/1994
Rajs	17/Oct/1995
Hartstein	17/Feb/1996

ORDER BY CLAUSE



- It is also possible to use the ORDER BY clause to order output by a column that is not listed in the SELECT clause.
- In the following example, the data is sorted by the last_name column even though this column is not listed in the SELECT statement.

```
SELECT employee_id, first_name
FROM employees
WHERE employee_id < 105
ORDER BY last_name;
```

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

ORDER OF EXECUTION

- THE ORDER OF EXECUTION OF A SELECT STATEMENT IS AS FOLLOWS:
  - FROM CLAUSE: LOCATES THE TABLE THAT CONTAINS THE DATA
  - WHERE CLAUSE: RESTRICTS THE ROWS TO BE RETURNED
  - SELECT CLAUSE: SELECTS FROM THE REDUCED DATA SET THE COLUMNS REQUESTED
  - ORDER BY CLAUSE: ORDERS THE RESULT SET

PRACTICE

- WRITE A SELECT STATEMENT THAT OUTPUTS THE FOLLOWING:

Partner Name	Area of Expertise	Expense Amount
Jennifer cho	Weddings	
Jason Tsang		
Allison Plumb	Event Planning	30000

- THE TABLE IS D_PARTNERS WITH COLUMNS: FIRST_NAME, LAST_NAME, EXPERTISE, AND AUTH_EXPENSE_AMT, ORDER THE OUTPUT STARTING WITH THE LARGEST EXPENSE AMOUNT.