Murach Chapter 3

How to Retrieve Data From a Single Table

Week 2, Lec 3

Knowledge Points in this lecture

- Base table values and calculated values
- Use column names in ORDER BY clause
- Column alias
- Concatenation operators in string expressions
- Operator precedence and parentheses in arithmetic expressions
- Scalar functions
- Dual Table
- DISTINCT keyword

The simplified syntax of the SELECT statement

```
SELECT select_list
FROM table_source
[WHERE search_condition]
[ORDER BY order_by_list]
```

The four clauses of the SELECT statement

- SELECT
- FROM
- WHERE
- ORDER BY

[ ] : Optional
WHERE clause and ORDER BY clause are optional and don’t have to appear in a query.

A simple SELECT statement

```
SELECT *
FROM invoices
```

	INVOICE_ID	VENDOR_ID	INVOICE_NUMBER	INVOICE_DATE	INVOICE_TOTAL	PAYMENT_TOTAL	CREDIT_TOTAL
1	1	34	QP58872	25-FEB-14	116.54	116.54	0
2	2	34	Q545443	14-MAR-14	1083.58	1083.58	0
3	3	110	P-0608	11-APR-14	20551.18	0	1200
4	4	110	P-0259	16-APR-14	26881.4	26881.4	0

```
(114 rows selected)
```

A SELECT statement that retrieves and sorts

```
SELECT invoice_number, invoice_date, invoice_total
FROM invoices
ORDER BY invoice_total
```

	INVOICE_NUMBER	INVOICE_DATE	INVOICE_TOTAL
1	25022117	24-MAY-14	6
2	24863706	27-MAY-14	6
3	24780512	29-MAY-14	6
4	21-4748363	09-MAY-14	9.95

```
(114 rows selected)
```

A SELECT statement that retrieves a calculated value

```
SELECT invoice_id, invoice_total,
       (credit_total + payment_total) AS total_credits
FROM invoices
WHERE invoice_id = 17
```

	INVOICE_ID	INVOICE_TOTAL	TOTAL_CREDITS
1	17	356.48	356.48

A SELECT statement that retrieves all invoices between given dates

```
SELECT invoice_number, invoice_date, invoice_total
FROM invoices
WHERE invoice_date BETWEEN '01-MAY-2014'
                       AND '31-MAY-2014'
ORDER BY invoice_date
```

	INVOICE_NUMBER	INVOICE_DATE	INVOICE_TOTAL
1	7548906-20	01-MAY-14	27
2	4-321-2596	01-MAY-14	10
3	4-327-7357	01-MAY-14	162.75
4	4-342-8069	01-MAY-14	10

(70 rows selected)

A SELECT statement that returns an empty result set

```
SELECT invoice_number, invoice_date, invoice_total
FROM invoices
WHERE invoice_total > 50000
```

Column specifications that use base table values

The * is used to retrieve all columns

```
SELECT *
```

Column names are used to retrieve specific columns

```
SELECT vendor_name, vendor_city, vendor_state
```

Base Table – table stored in the relational database

Column specifications that use calculated values

An arithmetic expression that calculates balance_due

```
SELECT invoice_number,
    invoice_total - payment_total - credit_total
    AS balance_due
```

A string expression that derives full_name

```
SELECT first_name || ' ' || last_name AS full_name
```

||: concatenation operator
Single quotes enclose string constants

Two SELECT statements that name the columns

A SELECT statement that uses the AS keyword

```
-- DATE is a reserved keyword.
-- As a result, it must be enclosed in quotations.
SELECT invoice_number AS "Invoice Number",
       invoice_date AS "Date",
       invoice_total AS total
FROM invoices
```

A SELECT statement that omits the AS keyword

```
SELECT invoice_number "Invoice Number",
       invoice_date "Date",
       invoice_total total
FROM invoices
```

The result set for both SELECT statements

	Invoice Number	Date	TOTAL
1	QP58872	25-FEB-14	116.54
2	Q545443	14-MAR-14	1083.58
3	P-0608	11-APR-14	20551.18
4	P-0259	16-APR-14	26881.4
5	MABO1489	16-APR-14	936.93

A SELECT statement that doesn't provide a name for a calculated column

```
SELECT invoice_number, invoice_date, invoice_total,
    invoice_total - payment_total - credit_total
FROM invoices
```

	INVOICE_NUMBER	INVOICE_DATE	INVOICE_TOTAL	INVOICE_TOTAL-PAYMENT_TOTAL-CREDIT_TOTAL
1	QP58872	25-FEB-14	116.54	0
2	Q545443	14-MAR-14	1083.58	0
3	P-0608	11-APR-14	20551.18	19351.18
4	P-0259	16-APR-14	26881.4	0
5	MABO1489	16-APR-14	936.93	0

How to concatenate string data

```
SELECT vendor_city, vendor_state,
       vendor_city || vendor_state
FROM vendors
```

	VENDOR_CITY	VENDOR_STATE	VENDOR_CITY||VENDOR_STATE
1	Auburn Hills	MI	Auburn HillsMI
2	Fresno	CA	FresnoCA
3	Olathe	KS	OlatheKS
4	Fresno	CA	FresnoCA
5	East Brunswick	NJ	East BrunswickNJ

How to format string data using literal values

```
SELECT vendor_name,
    vendor_city || ', '
                || vendor_state
                || ' '
                || vendor_zip_code
    AS address
FROM vendors
```

	VENDOR_NAME	ADDRESS
1	Data Reproductions Corp	Auburn Hills, MI 48326
2	Executive Office Products	Fresno, CA 93710
3	Leslie Company	Olathe, KS 66061
4	Retirement Plan Consultants	Fresno, CA 93704
5	Simon Direct Inc	East Brunswick, NJ 08816

Literal value – constant value

How to include apostrophes in literal values

```
SELECT vendor_name || '''s address: ',
       vendor_city || ', '
                   || vendor_state
                   || ' '
                   || vendor_zip_code
FROM vendors
```

	VENDOR_NAME||'"S'ADDRESS:'	VENDOR_CITY||','||VENDOR_STATE||'"||VENDOR_ZIP_CODE
1	Data Reproductions Corp's address:	Auburn Hills, MI 48326
2	Executive Office Products's address:	Fresno, CA 93710
3	Leslie Company's address:	Olathe, KS 66061
4	Retirement Plan Consultants's address:	Fresno, CA 93704
5	Simon Direct Inc's address:	East Brunswick, NJ 08816

The arithmetic operators in order of precedence

*	Multiplication
/	Division
+	Addition
-	Subtraction

A SELECT statement that calculates balance due

```
SELECT invoice_total, payment_total, credit_total,
       invoice_total - payment_total - credit_total
           AS balance_due
FROM invoices
```

	INVOICE_TOTAL	PAYMENT_TOTAL	CREDIT_TOTAL	BALANCE_DUE
1	116.54	116.54	0	0
2	1083.58	1083.58	0	0
3	20551.18	0	1200	19351.18
4	26881.4	26881.4	0	0
5	936.93	936.93	0	0

A SELECT statement that uses parentheses

```
SELECT invoice_id,
    invoice_id + 7 * 3 AS order_of_precedence,
    (invoice_id + 7) * 3 AS add_first
FROM invoices
ORDER BY invoice_id
```

	INVOICE_ID	ORDER_OF_PRECEDENCE	ADD_FIRST
1	1	22	24
2	2	23	27
3	3	24	30
4	4	25	33
5	5	26	36

What determines the sequence of operations

- Same as in Java, C
- Operator precedence
- Parentheses

Scalar Functions

- Operate on a single value and returns a single value
- Opposite to Aggregate Functions operating on a set of values for data summary
- Examples in this chapter
  - SUBSTR
  - TO_CHAR
  - SYSDAE
  - ROUND
  - MOD
- More in Chapter 8

A SELECT statement that uses SUBSTR

```
SELECT vendor_contact_first_name,
vendor_contact_last_name,
    SUBSTR(vendor_contact_first_name, 1, 1) ||
    SUBSTR(vendor_contact_last_name, 1, 1) AS initials
FROM vendors
```

	VENDOR_CONTACT_FIRST_NAME	VENDOR_CONTACT_LAST_NAME	INITIALS
1	Cesar	Arodondo	CA
2	Rachael	Danielson	RD
3	Zev	Alondra	ZA
4	Salina	Edgardo	SE
5	Daniel	Bradlee	DB

SUBSTR(string, start_position, substring_length):

- start_position: 0 or 1 means first letter

A SELECT statement that uses TO_CHAR

```
SELECT 'Invoice: # '
       || invoice_number
       || ', dated '
       || TO_CHAR(payment_date, 'MM/DD/YYYY')
       || ' for $'
       || TO_CHAR(payment_total)
       AS "Invoice Text"
FROM invoices
```

	Invoice Text
1	Invoice: # QP58872, dated 04/11/2014 for $116.54
2	Invoice: # Q545443, dated 05/14/2014 for $1083.58
3	Invoice: # P-0608, dated for $0
4	Invoice: # P-0259, dated 05/12/2014 for $26881.4
5	Invoice: # MABO1489, dated 05/13/2014 for $936.93

TO_CHAR function

- || : must operate on character data
- TO_CHAR(original_data, optional_format):
- original_data: NUMBER or DATE or CHAR
- optional_format: depends on the original data; may be omitted

A SELECT statement that uses the SYSDATE and ROUND functions

```
SELECT invoice_date,
    SYSDATE AS today,
    ROUND(SYSDATE - invoice_date)  AS invoice_age_in_days
FROM invoices
```

	INVOICE_DATE	TODAY	INVOICE_AGE_IN_DAYS
1	18-JUL-14	19-JUL-14	1
2	20-JUN-14	19-JUL-14	29
3	14-JUN-14	19-JUL-14	35

SYSDATE, ROUND functions

- SYSDATE
  - Current local date and time in DB server
  - No parameters; No parentheses; Error if add ()
- Date1 – Date2:
  - The number of days since Date1; may be fraction
- ROUND(number)
  - Round number to a whole number

A SELECT statement that uses the MOD function

```
SELECT invoice_id,
    MOD(invoice_id, 10) AS Remainder
FROM invoices
ORDER BY invoice_id
```

	INVOICE_ID	REMAINDER
9	9	9
10	10	0
11	11	1

MOD(number, divisor)

- Same as in Java: number%divisor
- Remainder of number/divisor

A SELECT statement that uses the Dual table

```
SELECT 'test'   AS test_string,
       10-7     AS test_calculation,
       SYSDATE AS test_date
FROM Dual
```

	TEST_STRING	TEST_CALCULATION	TEST_DATE
1	test	3	28-MAY-14

Dual table

- Automatically created and made available to users
- Used for testing expressions consists of literal values, operators, functions

A SELECT statement that returns ALL rows

```
SELECT vendor_city, vendor_state
FROM vendors
ORDER BY vendor_city
```

	VENDOR_CITY	VENDOR_STATE
1	Anaheim	CA
2	Anaheim	CA
3	Ann Arbor	MI
4	Auburn Hills	MI
5	Boston	MA

```
(122 rows selected)
```

Default: SELECT ALL…

- SELECT returns ALL rows (including duplicates) that meet conditions in WHERE clause.

A SELECT statement with no duplicate rows

```
SELECT DISTINCT vendor_city, vendor_state
FROM vendors
ORDER BY vendor_city
```

	VENDOR_CITY	VENDOR_STATE
1	Anaheim	CA
2	Ann Arbor	MI
3	Auburn Hills	MI
4	Boston	MA
5	Brea	CA

```
(53 rows selected)
```

Must place DISTINCT right after SELECT.

The syntax of the WHERE clause with comparison operators

```
WHERE expression_1 operator expression_2
```

The comparison operators

- =
- >
- <
- <=
- >=
- <> or !=

Examples of WHERE clauses that retrieve…

Vendors located in Iowa

```
WHERE vendor_state = 'IA'
```

Invoices with a balance due (two variations)

```
WHERE invoice_total - payment_total - credit_total > 0
```

```
WHERE invoice_total > payment_total + credit_total
```

Vendors with names from A to L

```
WHERE vendor_name < 'M'
```

Invoices on or before a specified date

```
WHERE invoice_date <= '31-MAY-14'
```

Invoices on or after a specified date

```
WHERE invoice_date >= '01-MAY-14'
```

Invoices with credits that don't equal zero

```
WHERE credit_total <> 0
```

Character values: case sensitive;
Date literal: '31-MAY-14'; must enclosed within single quotes