Single-Row Functions

Practical 5

Lesson Objectives

- Describe various types of functions that are available in SQL
- Use single row functions in SELECT statements
- Describe the use of conversion functions

Using Scripts

- Run the following scripts before the practical.
  - **Northwoods.sql**
  - **HR.sql**

Dummy Table

- The DUAL is special one row, one column dummy table present by default in all Oracle databases.
- The owner of DUAL is SYS (SYS owns the data dictionary, therefore DUAL is part of the data dictionary.) but DUAL can be accessed by every user.

DESCRIBE DUAL;

Dummy Table

Try the following SQL statements:

```
SELECT 'ABCDEF12345'
FROM DUAL;
```

Result: ABCDEF12345

```
SELECT 15+10-5*5/5
FROM DUAL;
```

Result: 20

SQL Functions

- Two Types:
  - Single-Row Function
  - Multiple-Row Function

- Single-Row Function
  - Operate on single rows only and return one result per row.
  - **character, number**, **date**, **conversion,** general
- Multiple-Row Function
  - Can manipulate groups of rows to give one result per group of rows.
  - Also known as Group Function.

Single-Row Example

S_ID	S_LAST	S_FIRST	S_MI	S_ADDRESS	S_CITY	S_STATE	S_ZIP
1	Miller	Sarah	M	144 Windridge Blvd.	Eau Claire	WI	54703
2	Umato	Brian	D	454 St. John’s Place	Eau Claire	WI	54702
3	Black	Daniel		8921 Circle Drive	Bloomer	WI	54715
4	Mobley	Amanda	J	1716 Summit St.	Eau Claire	WI	54703
5	Sanchez	Ruben	R	1780 Samantha Court	Eau Claire	WI	54701
6	Connoly	Michael	S	1818 Silver Street	Elk Mound	WI	54712

Single Row:

- Round
- Trunc
- Sign
- Lower
- Upper
- Initcap
- Concat
- Substr
- Length
- Instr
- Replace
- Trim

Multiple-Row Example

COURSE_ID	CALL_ID	COURSE_NAME	CREDITS
1	MIS 101	Intro. To Info. Systems	3
2	MIS 301	Systems Analysis	3
3	MIS 441	Database Management	3
4	CS 155	Programming in C++	3
5	MIS 451	Web-Based Systems	3

Multiple Row Function:

- Sum
- Avg
- Count
- Max
- Min

Single-Row Number Functions

- ROUND
  - Rounds a value to a specified position.
- TRUNC
  - Truncates a value to a specified position.
- SIGN
  - SIGN function returns a value indicating the sign of a number.

Single-Row Number Functions

```
SELECT ROUND(45.927,2),
       ROUND (45.927,0),
       ROUND (45.927),
       ROUND(45.927,-1)
FROM DUAL;
```

```
ROUND(45.927,2) ROUND(45.927,0) ROUND(45.927) ROUND(45.927,-1)
--------------------------------------------------------------
      45.93            46              46              50
```

Single-Row Number Functions

```
SELECT TRUNC(45.927,2),
            TRUNC (45.927,0),
            TRUNC (45.927),
            TRUNC (45.927,-1)
FROM DUAL;
```

```
TRUNC(45.927,2) TRUNC(45.927,0) TRUNC(45.927) TRUNC(45.927,-1)
-----------------------------------------------------------------
      45.92              45            45                40
```

Single-Row Number Functions

```
SELECT s_id, s_last, SYSDATE - s_dob
FROM student;

SELECT s_id, s_last, (SYSDATE - s_dob) / 365.25
FROM student;

SELECT s_id, s_last,TRUNC((SYSDATE - s_dob) /365.25)
FROM student;
```

Single-Row Character Functions

- Case Manipulation
  - LOWER, UPPER, INITCAP
- Character Manipulation
  - CONCAT, SUBSTR, LENGTH, INSTR, REPLACE, TRIM

Single-Row Character Functions

- LOWER() - Returns the string with all characters converted to lowercase letters.
- UPPER() - Returns the string with all characters converted to uppercase letters.
- INITCAP() - Returns the string with only the first letter in uppercase for each word.

Single-Row Character Functions

- CONCAT() - Concatenates (joins) two strings.

- SUBSTR() - Returns a string, starting at the *start* position, and of the specified *length*.

- LENGTH() - Returns an integer representing the string length.

Single-Row Character Functions

- INSTR() - Searches a string for a substring and returns an integer indicating the position of the character in string that is the first character of this occurrence.

- REPLACE() - Returns the string with every occurrence of the *search string* replaced with the *replacement string*.

- TRIM() , LTRIM(), RTRIM()
  - Removes all specified characters either from the beginning or the ending of a string.

Case Manipulation

```
SELECT LOWER('I Love SQL')
FROM DUAL;

SELECT UPPER('I Love SQL')
FROM DUAL;

SELECT INITCAP('I Love SQL')
FROM DUAL;
```

Using Case Manipulation Functions

```
SELECT term_id, term_desc, status
FROM term;

SELECT term_id, term_desc, INITCAP(status)
FROM term;
```

Using Case Manipulation Functions

```
SELECT s_last, s_first, s_dob
FROM student
WHERE s_last = 'mobley';
```

Result: no rows selected

```
SELECT s_last, s_first, s_dob
FROM student
WHERE LOWER(s_last) = 'mobley';
```

Character Manipulation

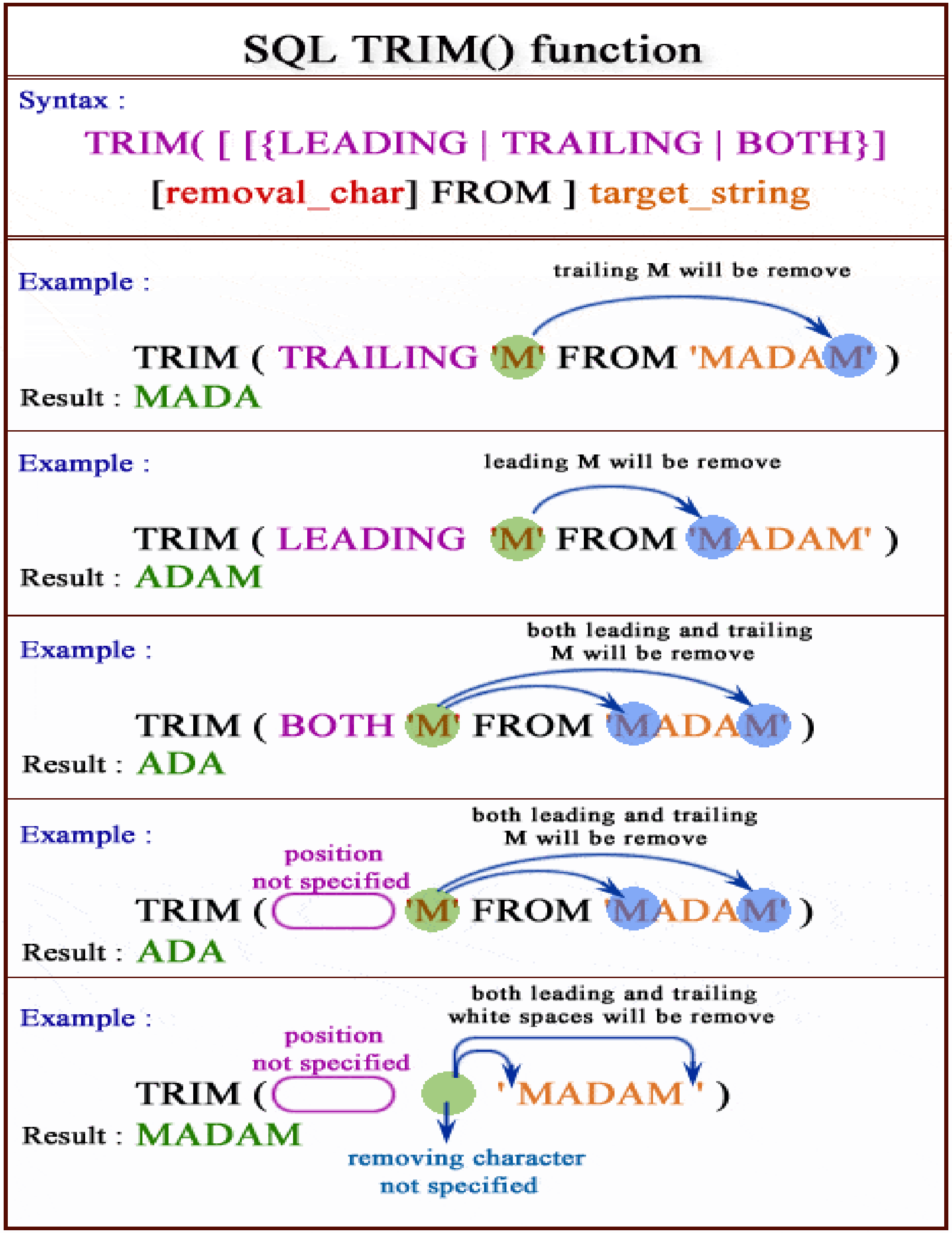
```
SELECT REPLACE('ABC','B','*')
FROM DUAL;

SELECT CONCAT('ABC','DEF')
FROM DUAL;

SELECT TRIM ('  ABC ')
FROM DUAL;

SELECT TRIM(LEADING '0' FROM '000123’)
FROM DUAL;
```

TRIM:



SQL TRIM() function

Syntax :

TRIM([[{LEADING | TRAILING | BOTH}] [removal_char] FROM] target_string

Example :

trailing M will be remove

TRIM (TRAILING 'M' FROM 'MADAM')

Result : MADA

Example :

leading M will be remove

TRIM (LEADING 'M' FROM 'MADAM')

Result : ADAM

Example :

both leading and trailing M will be remove

TRIM (BOTH 'M' FROM 'MADAM')

Result : ADA

Example :

both leading and trailing M will be remove

position not specified

TRIM ('M' FROM 'MADAM')

Result : ADA

Example :

both leading and trailing white spaces will be remove

position not specified

TRIM (' MADAM ')

removing character not specified

Result : MADAM

Character Manipulation

```
SELECT SUBSTR('ABCDEF',2,3)
FROM DUAL;

SELECT LENGTH('ABCDEFG ')
FROM DUAL;

SELECT INSTR('ABCDEFG','C')
FROM DUAL;
```

Character Manipulation

```
SELECT bldg_code, room
FROM location
WHERE bldg_code = 'CR';

SELECT CONCAT(bldg_code, room)
FROM location
WHERE bldg_code = 'CR';
```

Concatenation Operator

- Links columns or character strings to other columns.
- Is represented by two vertical bars, ||

```
SELECT bldg_code, room
FROM location;
```

```
SELECT bldg_code || room
FROM location;
```

Practice 5.1

Write a SQL statement to show the following result:

```
Bldg No
------------------
CR-101
CR-103
CR-105
CR-202
BUS-105
BUS-211
BUS-402
BUS-404
BUS-421
BUS-424
BUS-433
```

Working with Dates

- The Oracle database stores dates in an internal numeric format: century, year, month, day, hours, minutes, and seconds.
- The default date format: DD-MON-RR.
  - Enables you to store 21st-century dates in the 20th century by specifying only the last two digits of the year
  - Enables you to store 20th-century dates in the 21st century in the same way

SYSDATE Function

- SYSDATE is a date function that returns the current database server date and time.

Single-Row Date Functions

- Add_months(date,n)
- Months_between (date1, date2)
- next_day(date, 'Day')
- Last_day (date)
- ROUND(date[, 'fmt'])
- TRUNC(date[, 'fmt'])

Single-Row Date Functions

```
SELECT ADD_MONTHS(sysdate, 2)
FROM dual;

SELECT MONTHS_BETWEEN(hire_date,
    SYSDATE)
FROM employees
WHERE employee_id=114;
```

Single-Row Date Functions

```
SELECT NEXT_DAY(sysdate, 'FRIDAY')
FROM dual;

SELECT LAST_DAY(sysdate)
FROM dual;
```

Practice 5.2

- You next appointment with the dentist is six months from today. On what day will you go to the dentist? Name the output “appointment”.

Single-Row Date Functions

```
SELECT ROUND(SYSDATE, 'MONTH') FROM DUAL;
SELECT ROUND(SYSDATE, 'YEAR') FROM DUAL;
```

- SELECT ROUND(to_date('15/07/2008','DD/MM/YYYY'), 'MONTH’) FROM dual;

```
ROUND(TO_
---------
01-JUL-08
```

- SELECT ROUND(to_date('16/07/2008','DD/MM/YYYY'), 'MONTH’) FROM dual;

```
ROUND(TO_
---------
01-AUG-08
```

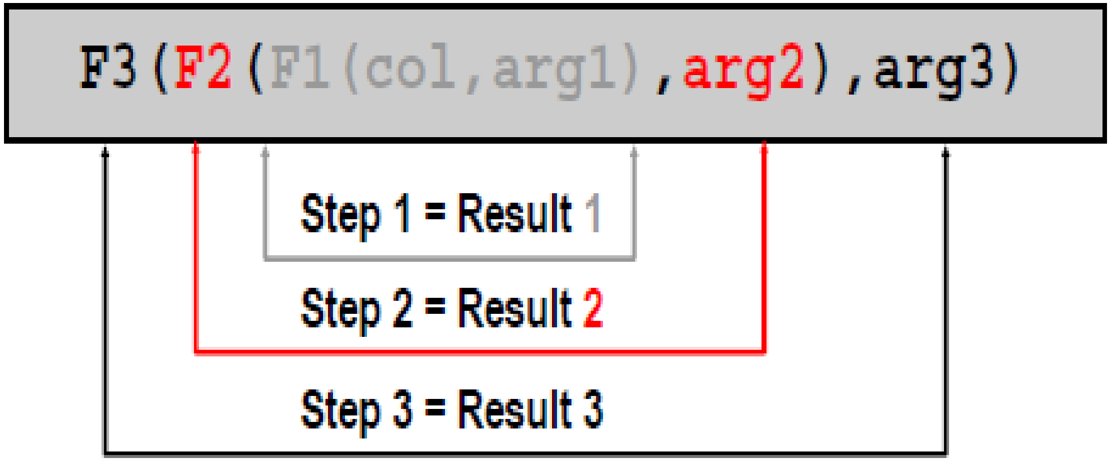
Single-Row Date Functions

```
SELECT TRUNC(SYSDATE, 'MONTH')
   FROM DUAL;


SELECT TRUNC (SYSDATE, 'YEAR')
   FROM DUAL;
```

Nesting Functions

- Single-row functions can be nested to any level.
- Nested functions are evaluated from deepest level to the least deep level.



```
F3(F2(F1(col,arg1),arg2),arg3)
```

Step 1 = Result 1

Step 2 = Result 2

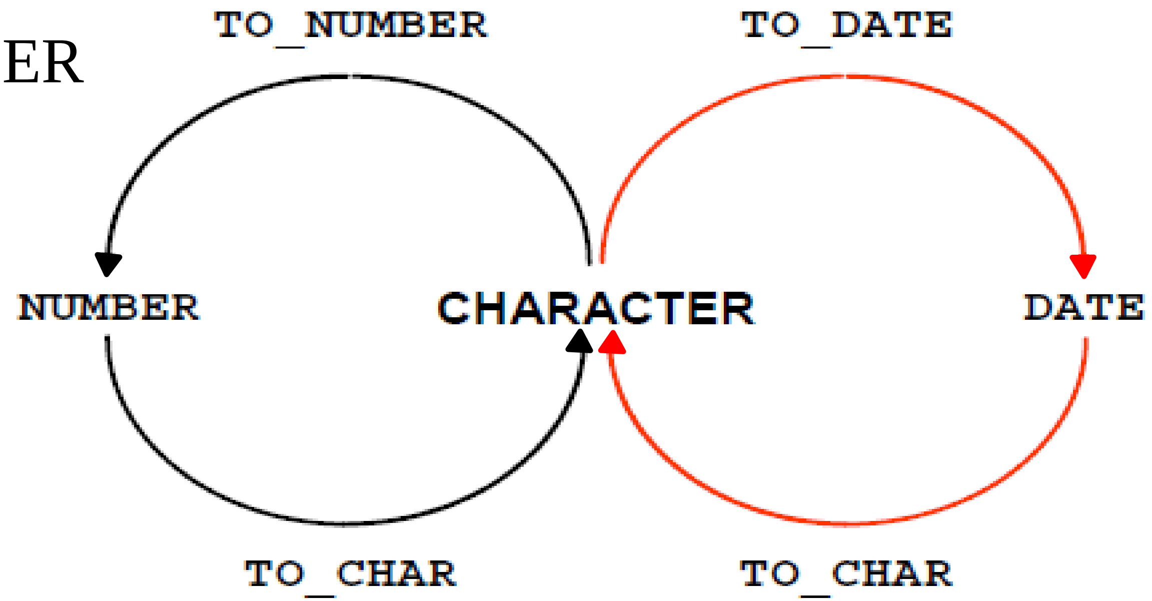
Step 3 = Result 3

Nesting Functions

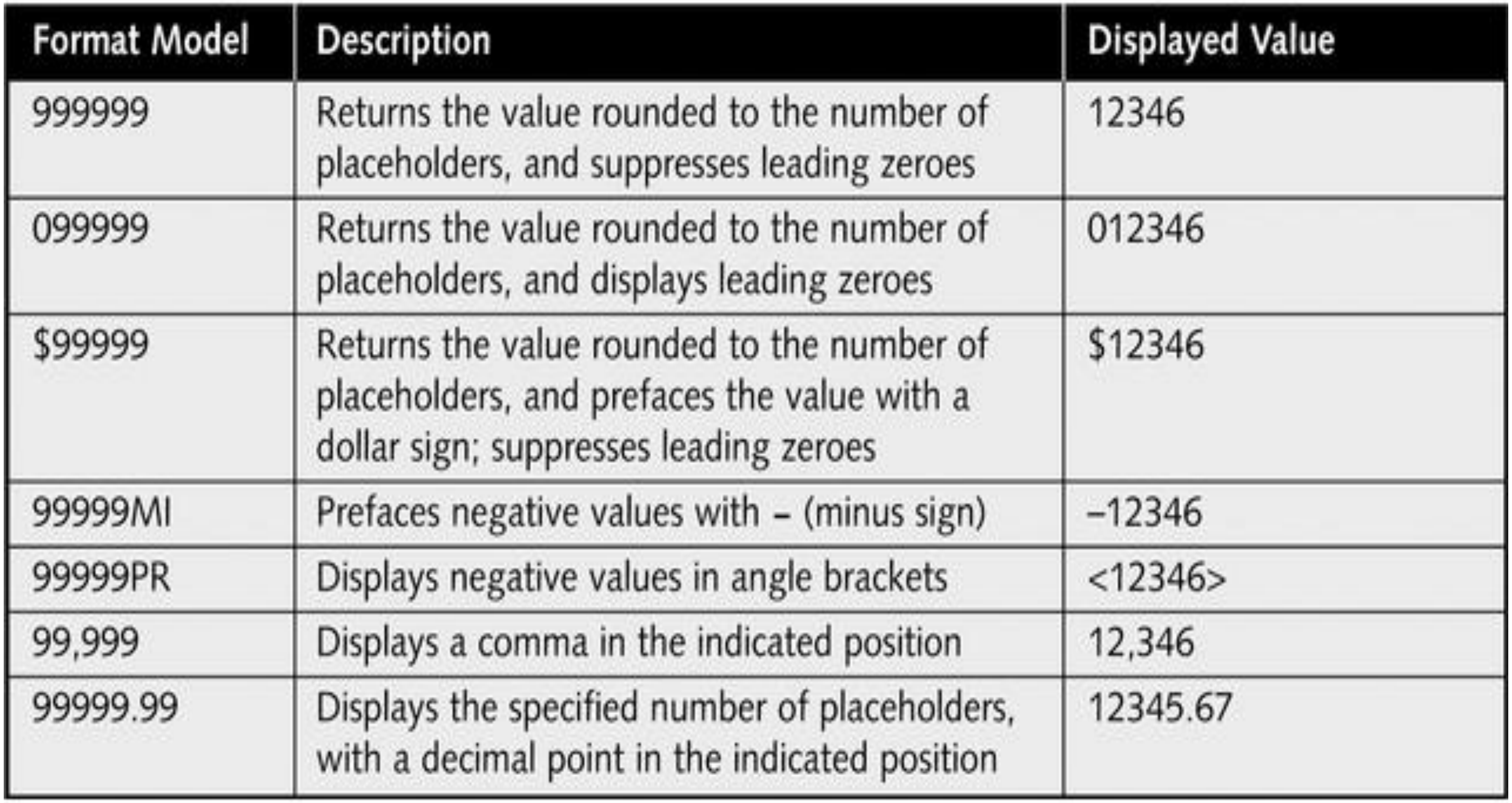
```
SELECT LENGTH(SUBSTR(course_name, 9, 5))
FROM COURSE;
```

Conversion Functions

- SQL provides three functions to convert a value from one data type to another.
  - TO_CHAR
  - TO_DATE
  - TO_NUMBER



Numerical Format Models



Format Model	Description	Displayed Value
999999	Returns the value rounded to the number of placeholders, and suppresses leading zeroes	12346
099999	Returns the value rounded to the number of placeholders, and displays leading zeroes	012346
$99999	Returns the value rounded to the number of placeholders, and prefaces the value with a dollar sign; suppresses leading zeroes	$12346
99999MI	Prefaces negative values with – (minus sign)	–12346
99999PR	Displays negative values in angle brackets	<12346>
99,999	Displays a comma in the indicated position	12,346
99999.99	Displays the specified number of placeholders, with a decimal point in the indicated position	12345.67

Using the TO_CHAR Function with Numbers

- Translates a value of NUMBER data type to VARCHAR2 data type.
- TO_CHAR(*field_name*, '*format_model*')

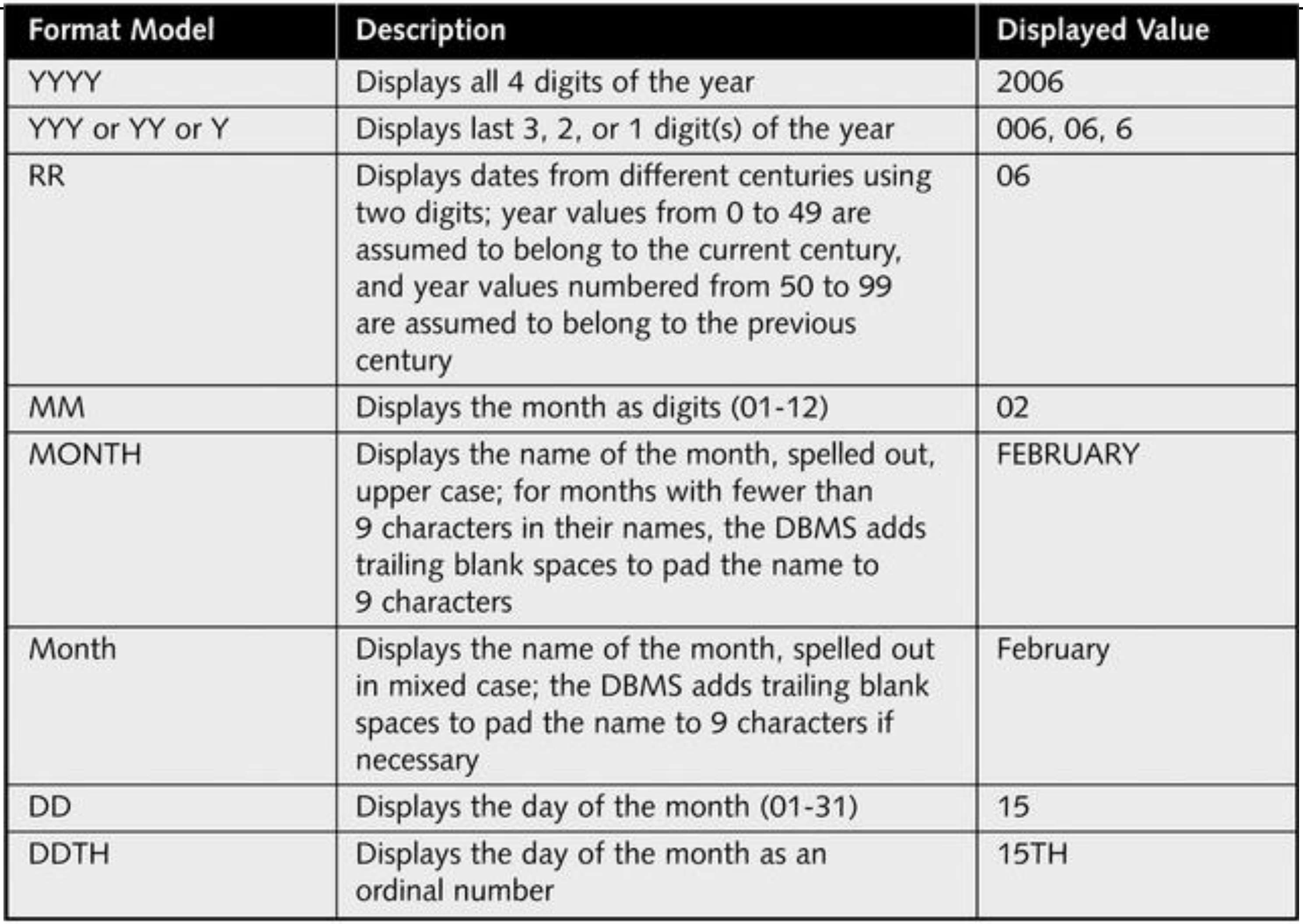
```
SELECT salary, TO_CHAR(salary, '$99,999.00') SALARY
FROM employees
WHERE last_name like 'E%';
```

Using the TO_CHAR Function with Dates

- translates a value of DATE data type to VARCHAR2 data type.
- TO_CHAR(*date*, '*format_model*')
- The format model:
  - Must be enclosed by single quotation marks
  - Is case-sensitive
  - Can include any valid date format element
  - Is separated from the date value by a comma

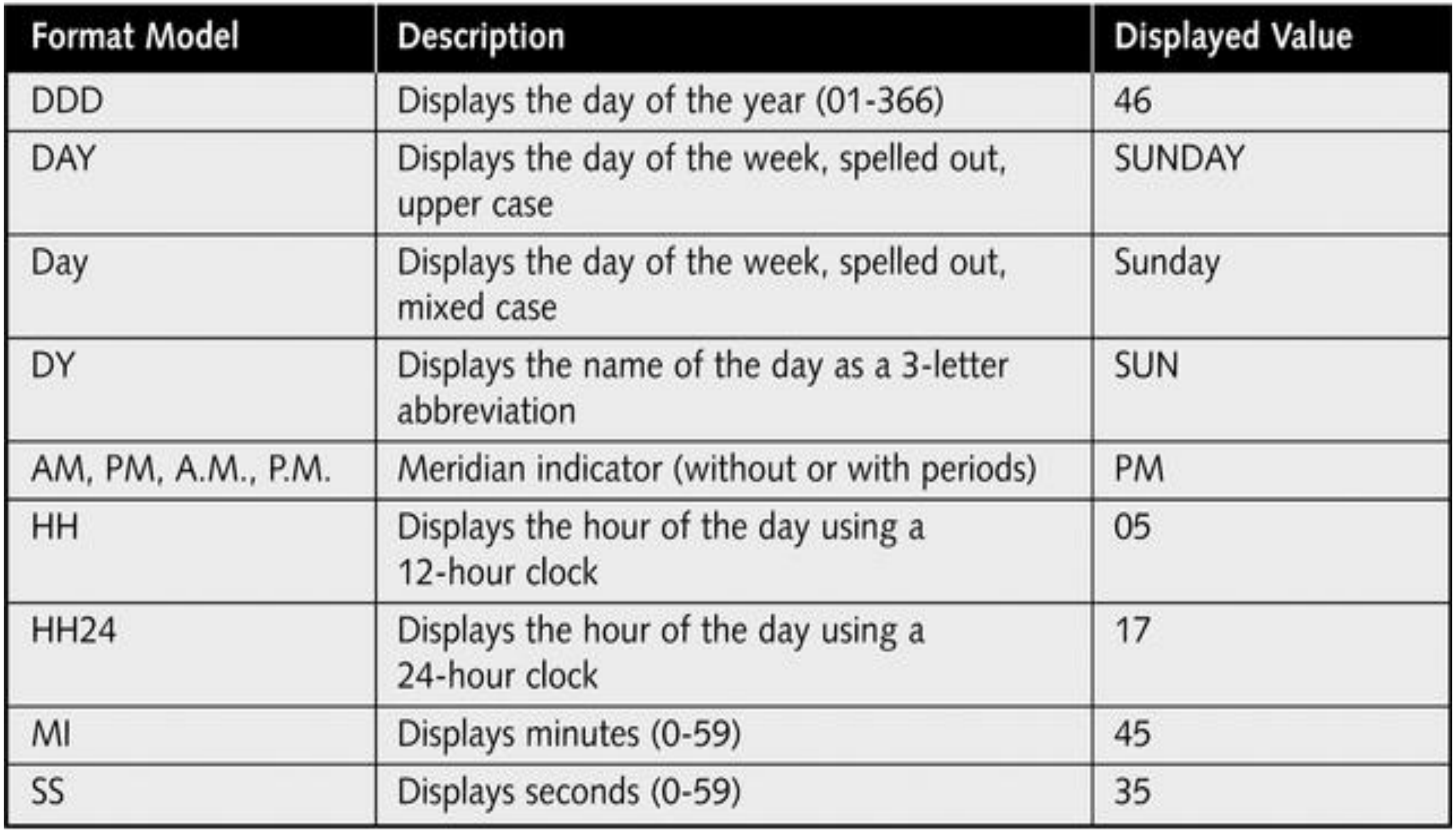
```
SELECT to_char(sysdate, 'YYYY' )
FROM dual;
```

Date Format Models



Format Model	Description	Displayed Value
YYYY	Displays all 4 digits of the year	2006
YYY or YY or Y	Displays last 3, 2, or 1 digit(s) of the year	006, 06, 6
RR	Displays dates from different centuries using two digits; year values from 0 to 49 are assumed to belong to the current century, and year values numbered from 50 to 99 are assumed to belong to the previous century	06
MM	Displays the month as digits (01-12)	02
MONTH	Displays the name of the month, spelled out, upper case; for months with fewer than 9 characters in their names, the DBMS adds trailing blank spaces to pad the name to 9 characters	FEBRUARY
Month	Displays the name of the month, spelled out in mixed case; the DBMS adds trailing blank spaces to pad the name to 9 characters if necessary	February
DD	Displays the day of the month (01-31)	15
DDTH	Displays the day of the month as an ordinal number	15TH

Date Format Models



Format Model	Description	Displayed Value
DDD	Displays the day of the year (01-366)	46
DAY	Displays the day of the week, spelled out, upper case	SUNDAY
Day	Displays the day of the week, spelled out, mixed case	Sunday
DY	Displays the name of the day as a 3-letter abbreviation	SUN
AM, PM, A.M., P.M.	Meridian indicator (without or with periods)	PM
HH	Displays the hour of the day using a 12-hour clock	05
HH24	Displays the hour of the day using a 24-hour clock	17
MI	Displays minutes (0-59)	45
SS	Displays seconds (0-59)	35

Formatting Date/time

```
SELECT s_id, s_last, s_dob
FROM student;

SELECT s_id, s_last, TO_CHAR(s_dob, 'MM/YY')
   Month of Birth
FROM student;
```

Using the TO_DATE Functions

- Convert a character string to a date format using the TO_DATE function:
  - TO_DATE(*char*[, '*format_model*'])

Using the TO_DATE Functions in Search Expressions

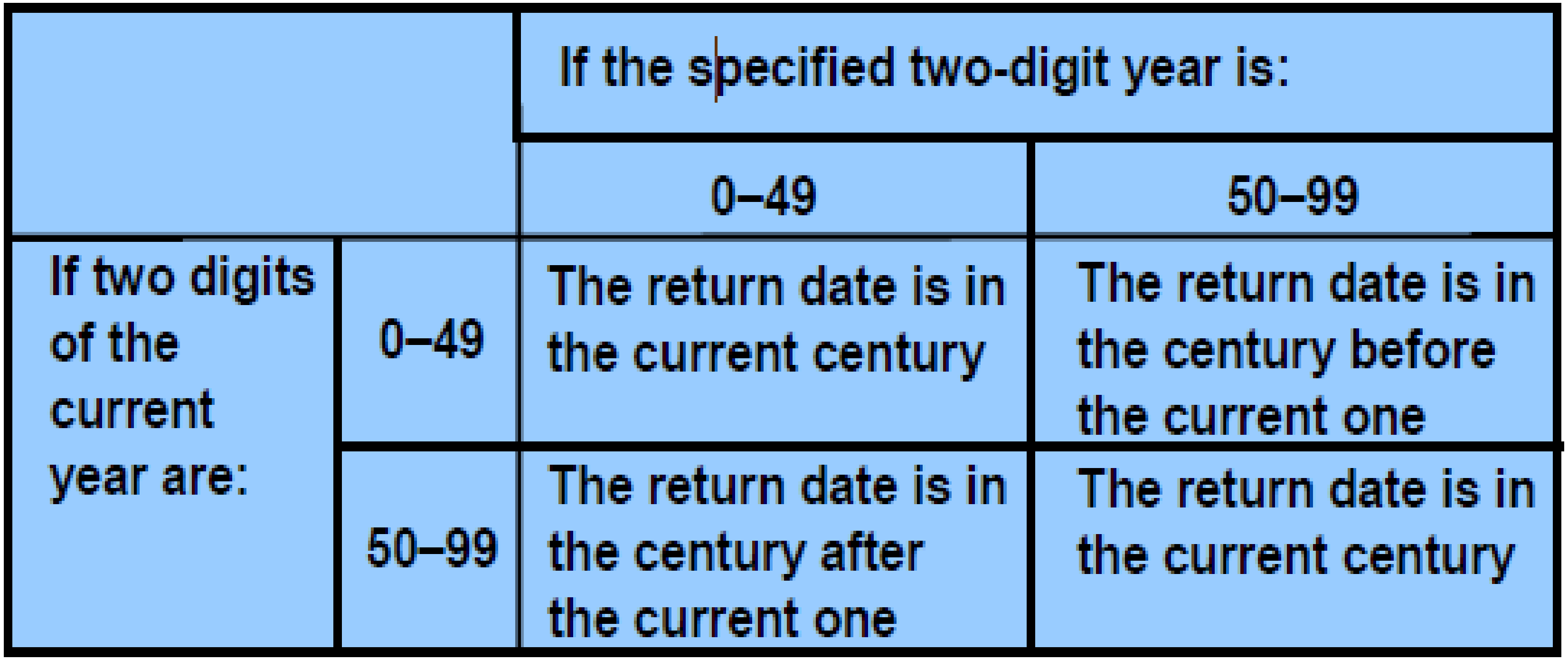
```
SELECT s_id, s_first, s_dob
FROM student
WHERE s_dob = '07/14/1985';
```

```
 *
ERROR at line 2:
ORA-01843: not a valid
month
```

```
SELECT s_id, s_first, s_dob
FROM student
WHERE s_dob = TO_DATE('07/14/1985', 'MM/DD/YYYY');
```

```
SELECT s_id, s_first, s_dob
FROM student
WHERE s_dob = TO_DATE('07/14/85', 'MM/DD/YY');
```

```
SELECT s_id, s_first, s_dob
FROM student
WHERE s_dob = TO_DATE('07/14/85', 'MM/DD/YYYY');
```



		If the specified two-digit year is:	
		0–49	50–99
If two digits of the current year are:	0–49	The return date is in the current century	The return date is in the century before the current one
	50–99	The return date is in the century after the current one	The return date is in the current century

Current Year	Specified Date	RR Format	YY Format
1995	**02-Nov-76**	1976	1976
1995	**02-Nov-14**	2014	1914
2001	**02-Nov-76**	1976	2076
2001	**02-Nov-14**	2014	2014

Using the TO_DATE Functions in Search Expressions

```
SELECT s_id, s_first, s_dob
FROM student
WHERE s_dob = TO_DATE('07/14/1985', 'MM/DD/RRRR');

SELECT s_id, s_first, s_dob
FROM student
WHERE s_dob = TO_DATE('07/14/85', 'MM/DD/RR');

SELECT s_id, s_first, s_dob
FROM student
WHERE s_dob = TO_DATE('07/14/85', 'MM/DD/RRRR');
```

Using the TO_DATE Functions in Search Expressions

Oracle default date format :

DD-MON-YYYY / DD-MON-RR

```
SELECT s_id, s_first, s_dob
FROM student
WHERE s_dob = '14-07-1985';

SELECT s_id, s_first, s_dob
FROM student
WHERE s_dob = '14-JUL-1985';

SELECT s_id, s_first, s_dob
FROM student
WHERE s_dob = '14-JUL-85';
```

Practice 5.3

- Write a query to retrieve to produce the following output.

```
Birthday
-----------------------------------------------------------------
Miller Sarah's birthday is on  14, July, 1985.
Umato Brian's birthday is on  19, August, 1985.
Black Daniel's birthday is on  10, October, 1982.
Mobley Amanda's birthday is on  24, September, 1986.
Sanchez Ruben's birthday is on  20, November, 1986.
Connoly Michael's birthday is on  4, December, 1986.
```

Using the TO_NUMBER Functions

- Convert a character string to a number format using the TO_NUMBER function:
  - TO_NUMBER(*char*[, '*format_model*'])

Using the TO_NUMBER Function

'character' , 'number format'

- SELECT TO_NUMBER('1210.73', '9999.99')

  FROM dual;

  Result: 1210.73

- SELECT TO_NUMBER('546', '999')

  Result: 546

- SELECT TO_NUMBER(room)

  FROM location;

Do it yourself

1. Retrieve the student last name who were born in ‘September’.
2. Get all employees last name and hired date, who were hired in ‘Jan’.
3. Get all employees last name and hired date, who were hired, arrange the hired date in alphabetical order in month regardless of year.

- Try the exercise given.