**SQL: Database Programming with SQL: Section 4, 5: Single Row Functions Part I and II**

**4-1**

|  |  |
| --- | --- |
| DUAL table | Dummy table used to view results from functions and calculations |
| format | The arrangement of data for storage or display. |
| INITCAP | Converts alpha character values to uppercase for the first letter of  each word, all other letters in lowercase |
| Character functions | Functions that accept character data as input and can return both  character and numeric values |
| TRIM | Removes all specified characters from either the beginning or the  ending of a string |
| expression | A symbol that represents a quantity or a relationship between  quantities |
| Substitution variable | Functions that operate on single rows only and return one result  per row |
| UPPER | Converts alpha characters to upper case |
| input | Raw data entered into the computer |
| CONCAT | Concatenates the first character value to the second character  value; equivalent to concatenation operator (||) |
| knowledge | Data that is processed into information |
| LOWER | Converts alpha character values to lowercase. |
| RPAD | Pads the left side of a character, resulting in a right-justified value |
| SUBSTR | Returns specific characters from character value starting at a  specific character position and going specified character positions  long |
| REPLACE | Replaces a sequence of characters in a string with another set of  characters |
| INSTR | Returns the numeric position of a named string. |
|  | Returns the number of characters in the expression |
| LPAD | Pads the right-hand side of a character, resulting in a left- justified  value |

1. SELECT 'The Best Class' AS Title, CONCAT('Oracle ', 'Internet ', 'Academy') AS Description;
2. SELECT 'The Net' AS Title,

LOWER(SUBSTRING('Oracle Internet Academy', 7, 3)) AS Description;

1. 23 characters
2. Position 8
3. SELECT LPAD(‘Oracle’, 4 ,’\*’), (‘Internet’, 4 ,’\*’), (‘Academy’, 4 ,’\*’)

FROM DUAL;

1. SELECT RPAD('Oracle Internet Academy', ' ', '$$$');
2. SELECT 'The Best Class' AS Title, REPLACE('Oracle Internet Academy', 'Internet', '2013-2014') AS Description;
3. SELECT order\_date, LPAD(CAST(order\_total AS CHAR), LENGTH(order\_total) + 1, '$') AS TOTAL

FROM F\_ORDERS;

1. SELECT CONCAT('ZOE TWEE ', street\_address, ' ', city, ', ', state, ' ', zip\_code) AS ADDRESS

FROM F\_CUSTOMERS

WHERE first\_name = 'ZOE' AND last\_name = 'TWEE';

1. SELECT CONCAT(UPPER(SUBSTR(first\_name, 1, 1)), last\_name) AS Name, salary, department\_id

FROM EMPLOYEES

WHERE department\_id = :dept\_id;

1. SELECT department\_id, department\_name, location\_id

FROM DEPARTMENTS

WHERE department\_name = :department\_name;

1. SELECT \*

FROM EMPLOYEES

WHERE TO\_CHAR(hire\_date, 'MON') = UPPER(:month\_abbr);

**4-2**

|  |  |
| --- | --- |
| TRUNC | Used to terminate the column, expression, or value to a specified  number of decimal places |
| Number functions | These functions accept numeric input and return numeric values. |
| MOD | Returns the remainder of a division. |
| ROUND | Rounds the column, expression, or value to a set number of  decimal places |

1. SELECT last\_name, salary,

ROUND(salary / 1.55, 2) AS adjusted\_salary

FROM employees

WHERE employee\_id BETWEEN 100 AND 102;

1. SELECT last\_name, salary,

TRUNC(salary \* 0.05333, 2) AS adjusted\_salary

FROM employees

WHERE department = 80;

1. MOD (33873, 2);
2. SELECT

ROUND( 845.553, 1),

ROUND(30695.348, 2),

ROUND(30695.348, -2),

TRUNC(2.3454, 1)

FROM DUAL;

1. SELECT last\_name, salary,

salary / 3 AS salary\_divided\_by\_3

FROM employees

WHERE MOD(salary, 3) = 0;

1. SELECT MOD(34, 8) AS EXAMPLE

FROM DUAL;

1. SELECT

565.784 AS original\_paycheck,

ROUND(565.784, 2) AS rounded\_paycheck,

TRUNC(565.784, 2) AS truncated\_paycheck

SELECT

original\_paycheck,

rounded\_paycheck,

truncated\_paycheck,

ROUND(original\_paycheck - rounded\_paycheck, 3),

ROUND((original\_paycheck - rounded\_paycheck) \* 1000, 3),

ROUND((original\_paycheck - rounded\_paycheck) \* 100000, 3),

ROUND((original\_paycheck - rounded\_paycheck) \* 1000000, 3)

FROM paychecks;

**4-3**

|  |  |
| --- | --- |
| SYSDATE | A function that returns the current date and time of the database  server |
| ADD\_MONTHS | Add calendar months to date |
| LAST\_DAY | Last day of the month |
| NEXT\_DAY | Next day of the date specified |
| MONTHS\_BETWEEN | Number of months between due dates |

1. SELECT

ROUND(MONTHS\_BETWEEN(SYSDATE, event\_date)) AS months\_difference

FROM events

WHERE event\_name = 'Vigil wedding';

1. SELECT

ROUND((MONTHS\_BETWEEN(TO\_DATE('2023-08-15', 'YYYY-MM-DD'), TO\_DATE('2023-06-01', 'YYYY-MM-DD')) \* 30.5)) AS Days

FROM DUAL;

1. SELECT

ROUND(MONTHS\_BETWEEN(TO\_DATE('2023-12-31', 'YYYY-MM-DD'), TO\_DATE('2023-01-01', 'YYYY-MM-DD')) \* 30.5) AS days\_difference

FROM DUAL;

1. SELECT

ROUND(MONTHS\_BETWEEN(SYSDATE, TRUNC(SYSDATE, 'YYYY')) / 12) + TRUNC(SYSDATE, 'YYYY') AS rounded\_year,

TRUNC(SYSDATE, 'YYYY') + INTERVAL '1' YEAR \* ROUND(MONTHS\_BETWEEN(SYSDATE, TRUNC(SYSDATE, 'YYYY')) / 12) AS truncated\_year,

ROUND(MONTHS\_BETWEEN(SYSDATE, TRUNC(SYSDATE, 'MM'))) + TRUNC(SYSDATE, 'MM') AS rounded\_month,

TRUNC(SYSDATE, 'MM') AS truncated\_month

FROM DUAL;

1. SELECT

LAST\_DAY(TO\_DATE('2005-06-01', 'YYYY-MM-DD')) AS last\_day\_of\_june

FROM DUAL;

1. SELECT

ROUND(MONTHS\_BETWEEN(SYSDATE, TO\_DATE('1990-01-15', 'YYYY-MM-DD')) / 12) AS years\_difference

FROM DUAL

WHERE employee\_name = 'Bob Miller';

1. SELECT

ADD\_MONTHS(SYSDATE, 6) AS Appointment

FROM DUAL;

1. SELECT

LAST\_DAY(SYSDATE) AS Deadline

FROM DUAL;

1. SELECT

MONTHS\_BETWEEN(TO\_DATE('2024-01-01', 'YYYY-MM-DD'), TO\_DATE('2023-03-15', 'YYYY-MM-DD')) AS months\_difference

FROM DUAL;

1. SELECT

NEXT\_DAY(TO\_DATE('2023-06-12', 'YYYY-MM-DD'), 'FRIDAY') AS "First Friday"

FROM DUAL;

1. SELECT

MONTHS\_BETWEEN(TO\_DATE('2024-01-01', 'YYYY-MM-DD'), TO\_DATE('2023-06-12', 'YYYY-MM-DD')) AS months\_difference

FROM DUAL;

1. SELECT

ADD\_MONTHS(TO\_DATE('2023-06-12', 'YYYY-MM-DD'), 3) AS new\_date

FROM DUAL;

1. It could be important to manipulate data for dinancial forecasting and budgeting.

Extension exercises

1. SELECT

ROUND(86.678, 2)

FROM DUAL;

1. SELECT

UPPER(cd\_title) AS "DJs on Demand Collections"

FROM cd\_collection

WHERE cd\_number IN (90, 91);

1. SELECT

LOWER(last\_name) || UPPER(SUBSTR(first\_name, 1, 1)) AS "User Passwords"

FROM partners;

1. SELECT

UPPER(REPLACE(REPLACE('It''s a small world', 'It''s a small', 'HELLO'), 'world', 'WORLD')) AS converted\_string

FROM DUAL;

1. SELECT

REPLACE('fiddledeedee', 'fiddle', '') || REPLACE('fiddledeedum', 'dum', '') AS "Nonsense"

FROM DUAL;

1. SELECT

REPLACE('Mississippi', 'i', '$') AS replaced\_string

FROM DUAL;

1. SELECT

ROUND(5332.342, -2)

FROM DUAL;

1. SELECT

ROUND(3.14159, 2) AS converted\_number

FROM DUAL;

1. SELECT

ROUND(73.892, 1) AS converted\_number

FROM DUAL;

1. SELECT

NEXT\_DAY(ADD\_MONTHS(SYSDATE, 6), 'FRIDAY') AS "Future"

FROM DUAL;

1. SELECT

ADD\_MONTHS(SYSDATE, 120) AS "Future"

FROM DUAL;

1. SELECT

TO\_DATE((CASE WHEN MOD(EXTRACT(YEAR FROM SYSDATE), 4) = 0 THEN EXTRACT(YEAR FROM SYSDATE) + 4 ELSE EXTRACT(YEAR FROM SYSDATE) + (4 - MOD(EXTRACT(YEAR FROM SYSDATE), 4)) END) || '-02-29', 'YYYY-MM-DD') AS "Future"

FROM DUAL;

1. SELECT cd\_theme

FROM cd\_collection

WHERE cd\_theme LIKE '%ie%';

1. SELECT cd\_title, cd\_year

FROM cd\_collection

WHERE cd\_year > 2000 AND cd\_year < 2003;

1. SELECT employee\_id, hire\_date

FROM employees

WHERE hire\_date BETWEEN TO\_DATE('1997-01-01', 'YYYY-MM-DD') AND SYSDATE

ORDER BY hire\_date DESC;

**5-1**

|  |  |
| --- | --- |
| CHAR | Used for text and character data of fixed length, including  numbers, dashes, and special characters |
| Fm | Used to remove padded blanks or to suppress leading zeros |
| Conversion function | Functions that convert a value from one datatype to another |
| NUMBER | Used to store variable-length numeric data |
| VARCHAR2 | Used for character data of variable length, including numbers,  special characters, and dashes |
| DATE | Used for date and time values |
| TO\_CHAR | Converts dates or numbers to character strings with optional  formatting |
| Rr date format | Century value depends on the specified year and the last two  digits of the current yea |
| TO\_NUMBER | Converts a character string containing digits to a number with  optional formatting |
| Dd date format | Numeric day of the month |
| TO\_DATE | Converts a character string representing a date to a date value  with optional formatting |

1. SELECT last\_name,

TO\_CHAR(birth\_date, 'FMMonth DD, YYYY') AS formatted\_birthday

FROM employees -- Replace with the actual table name if different

ORDER BY last\_name;

1. SELECT

TO\_CHAR(TO\_DATE('January 3, 04', 'Month DD, YY'), 'DD-Mon-YYYY') AS formatted\_date

FROM DUAL;

1. SELECT

'The promotion began on the ' ||

TO\_CHAR(start\_date, 'FMDDth') || ' of ' ||

TO\_CHAR(start\_date, 'FMMonth YYYY') || '.' AS promotion\_message

FROM promotional\_menus

WHERE promo\_code = 110;

1. SELECT

'Today is the ' ||

TO\_CHAR(SYSDATE, 'FMDDth') || ' of ' ||

TO\_CHAR(SYSDATE, 'FMMonth') || ', ' ||

TO\_CHAR(SYSDATE, 'YYYY') AS formatted\_date

FROM DUAL;

1. SELECT employee\_id, employee\_name,

TO\_CHAR(salary, '$999,999.99') AS formatted\_salary

FROM employees

ORDER BY employee\_id;

1. SELECT first\_name, last\_name,

TO\_CHAR(salary, '$999,999.99') AS current\_salary,

TO\_CHAR(salary + 2000, '$999,999.99') AS "New Salary"

FROM employees

WHERE first\_name = 'Ellen' AND last\_name = 'Abel';

1. SELECT

TO\_CHAR(start\_date, 'Day') AS day\_of\_week,

TO\_CHAR(start\_date, 'DD-Mon-YYYY') AS date

FROM f\_promotional\_menus -- Replace with the actual table name if different

WHERE promo\_code = 110;

1. SELECT

TO\_CHAR(TO\_DATE('25-Dec-2004', 'DD-Mon-YYYY'), 'FMMonth DDth, YYYY')

UPPER(TO\_CHAR(TO\_DATE('25-Dec-2004', 'DD-Mon-YYYY'), 'FMMonth DDth, YYYY')),

TO\_CHAR(TO\_DATE('25-Dec-2004', 'DD-Mon-YYYY'), 'FMDDth FMMonth, YYYY')

FROM DUAL;

1. SELECT

TO\_CHAR(low\_range, '$999,999.99') AS formatted\_low\_range,

TO\_CHAR(high\_range, '$999,999.99') AS formatted\_high\_range

FROM d\_packages;

1. SELECT

TO\_DATE('JUNE192004', 'FXMonthYYYY') AS converted\_date

FROM DUAL;

1. Implicit conversion is automatic and happens without user intervention (e.g., concatenating a number with a string). Explicit conversion requires specific commands to convert data types (e.g., using TO\_DATE or CAST)
2. It can help with data integration, increase accuracy in reporting, help businesses leverage their data more effectively.

**5-2**

|  |  |
| --- | --- |
| NVL | Converts nulls to an actual value |
| COALESCE | Returns the first non-null expression in the list |
| NVL2 | Examines the first expression; if the first expression is not null, it  returns the second expression; if the first expression is null, it  returns the third expression |
| NULLIF | Compares two expressions; if they are equal, the function returns  null; if they are not equal, the function returns the first expression |

1. SELECT promo\_name, start\_date,

CASE

WHEN end\_date IS NOT NULL THEN 'end in two weeks'

ELSE TO\_CHAR(SYSDATE, 'DD-Mon-YYYY')

END AS end\_date

FROM f\_promotional\_menus;

1. SELECT last\_name,

NVL(overtime\_rate, 0) AS "Overtime Status"

FROM employees;

1. SELECT last\_name,

NVL(overtime\_rate, 5.00) AS "Overtime Rate"

FROM employees;

1. SELECT last\_name,

CASE

WHEN manager\_id IS NULL THEN 9999

ELSE manager\_id

END AS "Manager ID"

FROM employees;

1. C
2. If last\_name is not NULL, the query returns the value of last\_name and if last\_name is NULL, the query returns the manager\_id converted to a character string
   1. SELECT first\_name,last\_name,

TO\_CHAR(hire\_date, 'Month') AS month\_of\_hire

FROM EMPLOYEES;

* 1. SELECT first\_name, \_name,

NULLIF(TO\_CHAR(hire\_date, 'Month'), 'September') AS month\_of\_hire

FROM

EMPLOYEES;

1. SELECT first\_name,

NVL(specialty, 'No Specialty') AS specialty

FROM d\_partners;

**5-3**

|  |  |
| --- | --- |
| DECODE | Compares an expression to each of the search values |
| CASE | An if-then-else expression whose value depends on the truth-  value of a Boolean expression |
| Conditional expression | Implements conditional processing within a SQL statement; it  meets the ANSI standard |

1. SELECT

CASE

WHEN duration = 2 THEN 'shortest'

WHEN duration = 10 THEN 'longest'

ELSE TO\_CHAR(duration) -- Assuming you want to keep other durations as they are

END AS "Play Times"

FROM d\_songs;

1. SELECT department\_id, last\_name, salary,

CASE

WHEN department\_id = 10 THEN salary \* 1.25

WHEN department\_id = 90 THEN salary \* 1.5

WHEN department\_id = 130 THEN salary \* 1.75

ELSE salary

END AS "New Salary"

FROM employees;

1. SELECT first\_name, last\_name, manager\_id, \_pct,

CASE

WHEN manager\_id IS NULL THEN

NVL(commission\_pct, 99999)

ELSE manager\_id

END AS Review

FROM employees

WHERE department\_id IN (80, 90);