Using Conversion Functions and Conditional Expressions

Suman Sahil, Steve Simon

Conversion Functions

In some cases, the Server uses data of one type where it expects data of a different data type.

– **Implicit Data Type Conversion**

Supports internal type casting

– **Explicit Data Type Conversion**

It is recommended to perform explicit conversion instead of relying on software intelligence.

Implicit Data Type Conversion

- A VARCHAR2 or CHAR value can be implicitly converted to NUMBER or DATE type value.
- Similarly, a NUMBER or DATA type value can be automatically converted to character data.
- The implicit interconversion happens only when the character represents the a valid number or date type value respectively.

Implicit Data Type Conversion

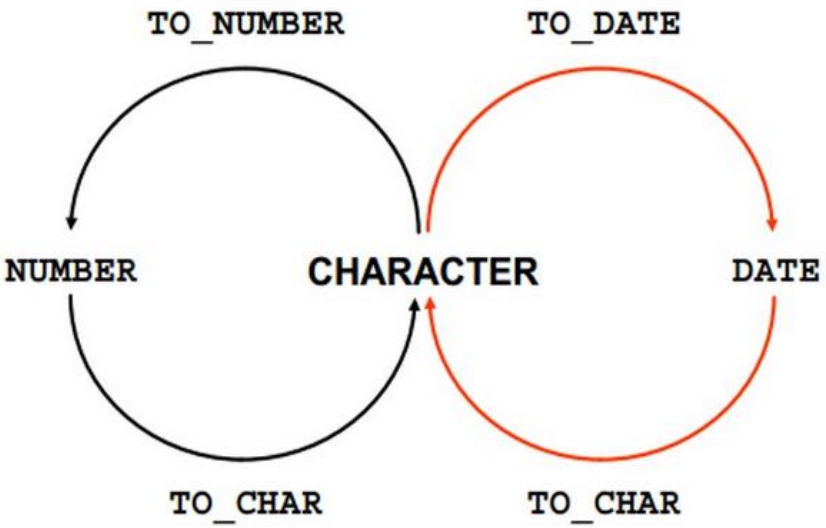
Implicit Data Type Conversion examples

- SQL Code

```
select *
from hospital
where hosp_id=218

select *
from hospital
where hosp_id='218'
```

Explicit Data Type Conversion



Explicit Data Type Conversion

- TO_CHAR function is used to typecast a numeric or date input to character type with a format model (optional).
- Syntax

  TO_CHAR(number1, [format], [nls_parameter])

Explicit Data Type Conversion

- SQL Code
  - select TO_CHAR (weight, '99.99') as weight
    from encounter where weight < 20
  - select TO_CHAR (admission_dt, 'MONTH DD YYYY') as admitted_date
    from encounter where weight < 20

Explicit Data Type Conversion

Using the TO_NUMBER and TO_DATE Functions :

- Convert a character string to a number format using the TO_NUMBER function

  TO_NUMBER (string1, [format], [nls_parameter])

- Convert a character string to a date format using the TO_DATE function

  TO_DATE(string1, [format_mask], [nls_language])

Explicit Data Type Conversion

- SQL Code
  - SELECT TO_NUMBER('5428.73', '9999.99')
    FROM DUAL

  - SELECT TO_DATE('January 5, 2019', 'MM/DD/YYYY')
    FROM DUAL

Conditional Functions

- Conditional functions like DECODE and CASE impose conditions in a SQL statement

- Decode is function is the SQL equivalence of IF..THEN..ELSE conditional statement.

DECODE (expression, search, result [, search, result]... [, default])

Conditional Functions

- CASE expressions is the SQL equivalence of IF..THEN..ELSE conditional statement.

```
CASE  [ expression ]
   WHEN condition_1 THEN result_1
   WHEN condition_2 THEN result_2
   ...
   WHEN condition_n THEN result_n
   ELSE result
END
```

Conditional Functions

- SQL Code
  - SELECT DECODE(NULL,NULL,'EQUAL','NOT EQUAL')
    FROM DUAL

    - SELECT enc_id, CASE WHEN weight < 50 THEN 'GRADE 1'
      WHEN weight > 50 AND weight < 200 THEN 'GRADE 2'
      ELSE 'GRADE 3'
      END
      FROM encounter

Conditional Functions

- IN:
- Checks whether a value is present within a set of values and can be used with WHERE, CHECK and creation of views.

- Syntax:
- WHERE column IN (x1, x2, x3 [,.....])

- SQL Code:
- SELECT * from encounter
  WHERE enc_id IN (495124258, 600334220)

Conditional Functions

- COALESCE : Returns the first non-null argument. Null is returned only if all arguments are null. It is often used to substitute a default value for null values when data is retrieved for display.

- Syntax:
  COALESCE(value [, ...])

- SQL Code:
- SELECT COALESCE(census_reg, '- NA -')
  from hospital where census_reg is null

Conditional Functions

- GREATEST: Returns the largest value from a list of any number of expressions.

- Syntax:
- GREATEST(expr1, expr2 [, ...])

- SQL Code:
- SELECT GREATEST('XYZ', 'xyz')
   from dual

Conditional Functions

- LEAST: Returns the smallest value from a list of any number of expressions.

- Syntax:
- LEAST(expr1, expr2 [, ...])

- SQL Code:

```
SELECT LEAST('XYZ', 'xyz')
from dual
```

Your homework

- Put your code and the output in a single PDF file

- Use Encounter Table
- Use case expression to classify age < 20 as ‘Group 1’, and age > 20 as ‘Group 2’

- Use hospital table
- Use coalesce function to return -1 for null values of teaching_ind in hospital table where census_reg = ‘West’