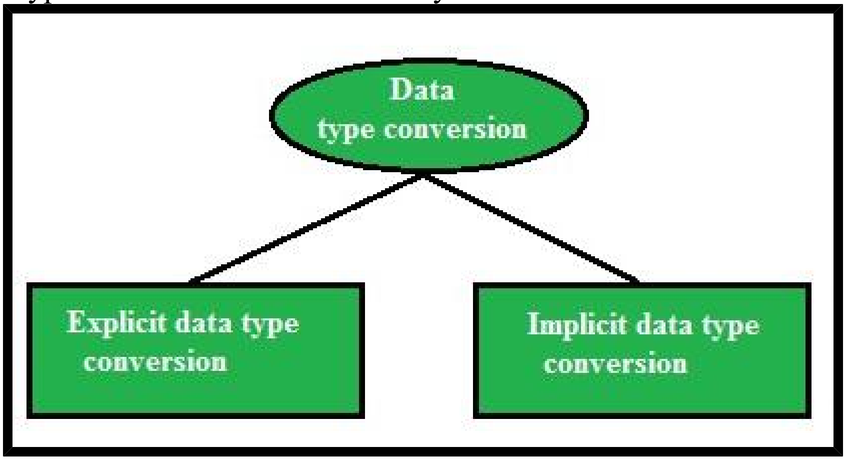
CONVERSION FUNCTIONS

When you define expressions and local variables then you should specify what type of data will be stored in those objects such as text data, money, dates, numbers, or characters.

- Strings Data types such as CHAR and VARCHAR.
- Decimal values such as FLOAT and REAL.
- Binary String such as BINARY.
- Date and Time Data Types such as DATE, TIME, TIMESTAMP, and DATETIME.
- Numeric Data types such as INT, DOUBLE, and BIGINT.
- MS Access Data Types such as TEXT, LONG, and BYTE. On the basis of this, there are two types of conversion in the Data first implicit types conversion and the second is explicit data type conversion. In implicit type conversion Server can automatically convert the data type from one type to another (i.e., VARCHAR TO CHAR and INT TO FLOAT) but in explicit data type conversion it can be done by the user side.



Implicit Data-Type Conversion

In this type of conversion, the data is converted from one type to another implicitly (by itself/automatically).

From	To
VARCHAR2 or CHAR	NUMBER
VARCHAR2 or CHAR	DATE
DATE	VARCHAR2
NUMBER	VARCHAR2

Query

```
CREATE TABLE employees (
   employee_id INT PRIMARY KEY,
   first_name VARCHAR(50),
   salary INT
);
INSERT INTO employees (employee_id, first_name, salary)
```

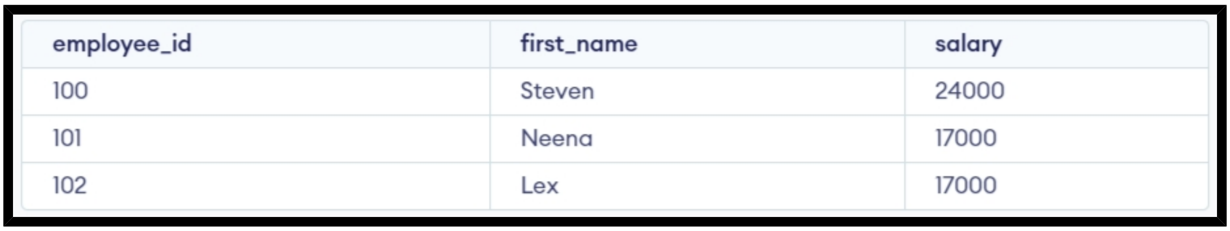
```
VALUES
  ( 100, 'Steven', 24000),
  ( 101, 'Neena', 17000),
  ( 102, 'Lex', 17000),
  ( 103, 'John', 11000),
  ( 104, 'Robert', 12000),
  ( 105, 'Leo', 10000);
```

1 - Query

Here, we want to retrieve the employee_id, first_name, and salary from the employees table whose salary is greater than 15000 then the query is

```
SELECT employee_id, first_name, salary
FROM employees
WHERE salary > 15000;
```

1 - Output

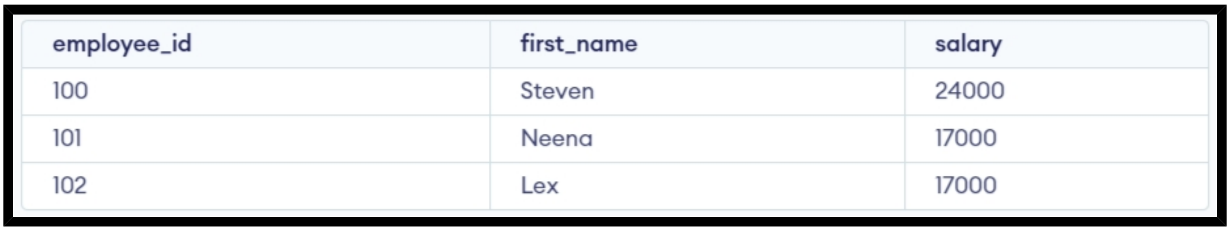


employee_id	first_name	salary
100	Steven	24000
101	Neena	17000
102	Lex	17000

2 - Query

```
SELECT employee_id, first_name, salary
FROM employees
WHERE salary > '15000';
```

2 - Output

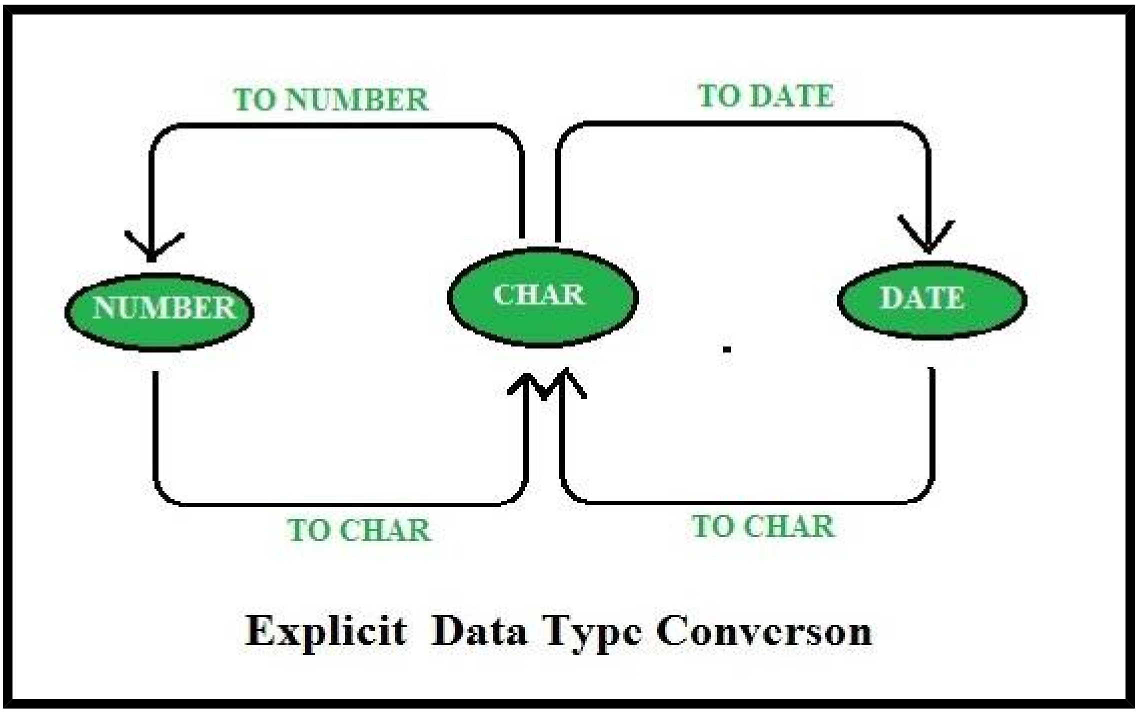


employee_id	first_name	salary
100	Steven	24000
101	Neena	17000
102	Lex	17000

Here we see the output of both queries came out to be the same, in spite of the 2nd query using ‘15000’ as text, it is automatically converted into an int data type.

Explicit Data-Type Conversion

In this type of conversion, the data is converted from one type to another explicitly (by the user). simply we can say, users define the type to which the expression is to be converted.



TO_CHAR Function

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

Syntax

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

Using the TO_CHAR Function with Dates

Syntax

```
TO_CHAR(date, 'format_model')
```

The format model:

- Must be enclosed in single quotation marks and is case sensitive.
- Can include any valid date format element in the query.
- Has an fm element to remove padded blanks or suppress leading zeros.
- Is separated from the date value by a comma.

Example

```
SELECT employee_id,
TO_CHAR(hire_date, 'MM/YY') AS Month_Hired
FROM employees
WHERE last_name = 'Higgins';
```

Output

EMPLOYEE_ID	MONTH_HIRED
205	06/94

Elements of the Date Format Model

ELEMENT	DESCRIPTION
YYYY	Full-year in Numbers
YEAR	Year spelled out
YY	Two-digit value of year
MM	Two-digit value for the month
MONTH	Full name of the month
MON	Three Letter abbreviation of the month
D	Number of Days in a Week
DY	Three-letter abbreviation of the day of the week
DAY	Full Name of the Day
DD	Numeric day of the month

Date Format Elements – Time Formats

Use the formats listed in the following tables to display time information and literals and to change numerals to spelled numbers.

ELEMENT	DESCRIPTION
AM or PM	Meridian indicator
A.M. or P.M.	Meridian indicator with periods
HH or HH12 or HH24	Hour of day, or hour (1-12), or hour (0-23)
MI	Minute 0-59
SS	Second 0-59
SSSSS	Second past Mid Night 0-86399

Other Formats

ELEMENT	DESCRIPTION
/ . ,	Punctuation is reproduced in the result
"of the"	The quoted string is reproduced in the result

Specifying Suffixes to Influence Number Display

ELEMENT	DESCRIPTION
TH	Ordinal Number (for example DDTH for 4TH)
SP	Spelled out number (for example DDSP for FOUR)
SPTH or THSP	Spelled out ordinal numbers (for example DDSPTH for FOURTH)

Example

```
SELECT last_name,
TO_CHAR(hire_date, 'fmDD Month YYYY')
   AS HIREDATE
FROM employees;
```

Output

LASTNAME	HIREDATE
Austin	25 January 2005
Shubham	20 June 2004
Nishant	15 January 1999
Ankit	15 July 1995
Vanshika	5 August 2004
Kusum	10 June 1994
Faviet	11 March 2005
King	9 April 1996

Using the TO_CHAR Function with Numbers

Syntax

```
TO_CHAR(number, 'format_model')
```

These are some of the format elements you can use with the TO_CHAR function to display a number value as a character :

ELEMENT	DESCRIPTION
9	Represents a number
0	Forces a zero to be displayed
$	Places a floating dollar sign
L	Uses the floating local currency symbol

.	Prints a decimal point
,	Prints a thousand indicator

Example

```
SELECT TO_CHAR(salary, '$99,999.00') AS SALARY
FROM employees
WHERE last_name = 'Ernst';
```

Output

SALARY
$5000

Using the TO_NUMBER and TO_DATE Functions :

Convert a character string to a number format using the **TO_NUMBER** function:

```
TO_NUMBER(char[, 'format_model'])
```

Convert a character string to a date format using the **TO_DATE** function:

```
TO_DATE(char[, 'format_model'])
```

These functions have an **fx** modifier. This modifier specifies the exact matching for the character argument and date format model of a **TO_DATE** function.

Example

```
SELECT last_name, hire_date
FROM employees
WHERE hire_date = TO_DATE('May 24, 1999', 'fxMonth DD, YYYY');
```

Output

LASTNAME	HIREDATE
Kumar	24-MAY-1999