May 2017

Function - Scalar & Aggregate

Buddha Purnima / Vesak Buddha Day (India / Malaysia / Singapore / Sri Lanka)

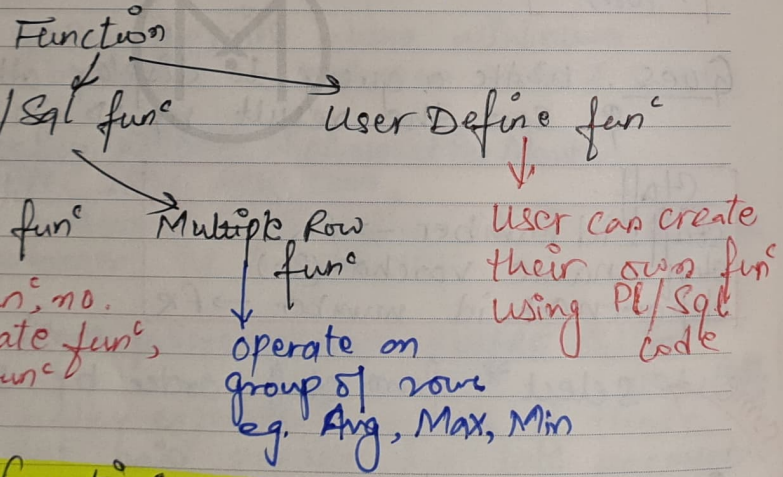
10 Wednesday

(130 - 235) Wk 19

Function :- Block of code that sometimes take arguments & always return a value.

SQL functions can be used to do the following:

1. Perform calculations on data
2. Modify individual data items
3. Manipulate o/p for groups of rows
4. Formate dates & no. for displays.
5. Convert column data types.



Single Row Function :-

↳ work in a single row & return one o/p per row.

↳ These func can appear in select list, where clauses & having clauses.

11 Thursday

(131 - 234) Wk 19

- Manipulates data items.
- Accepts arguments & returns one value.
- Acts on each row returned.
- Returns one result per row.
- May modify the data type.
- Can be nested.
- Accepts arguments which can be a column or an expression.
- fun_name [(arg1, arg2, ...)]

CHARACTER FUNCTION

↳ Single-row character funcs accept character data as i/p & can return both character & numeric values.

Function	Purpose.
① Lower(column/exp)	- converts alpha character values to lower case.
② Upper(column/exp)	- Convert alpha character values to upper case.
③ Initcap [column/exp]	- Convert alpha character values to upper case for the first letter of each word.
④ Concat(col1/exp1, col2/exp2)	- Concatenates two columns or expression values
⑤ Substr(col/exp, m, [n])	- Returns specified characters from the character value starting at character position m, n characters long.
⑥ Length(col/exp)	- Return the no. of characters in the expression

May 2017

CID	CNAME	PHONENO	Email	Address
1	Tom	9876	t@xyz.com	Chennai
2	John	8765	j@yza.com	Delhi

12 Friday
(132 - 233) Wk 19

Example :-

① SELECT LOWER(cname) FROM customer WHERE cid = 1;

Ans. o/p ⇒ Tom

② SELECT UPPER(cname) FROM Customer WHERE cid=1;

Ans o/p ⇒ TOM

③ SELECT INITCAP(cname) FROM customer WHERE cid=1;

Ans. o/p ⇒ Tom

④ SELECT CONCAT(Cname, Address) FROM customer WHERE cid=1;

Ans o/p ⇒ TomChennai

⑤ SELECT SUBSTR(cname, 1, 2) FROM Customer WHERE cid=1;

Ans o/p ⇒ To

SELECT SUBSTR(cname, 1) FROM Customer WHERE cid=1;

Ans o/p ⇒ Tom

SELECT SUBSTR(cname, -2, 2) FROM Customer WHERE cid = 1;

Ans o/p ⇒ om

⑥ SELECT LENGTH(cname) FROM Customer WHERE cid=1;

Ans. o/p ⇒ 3

NUMBER FUNCTION

13 Saturday
(133 - 232) Wk 19

① Abs(col|exp) → returns the absolute value of n

SELECT ABS(-5) FROM DUAL;
o/p → 5

② Ceil(col|exp) → returns the smallest integer value that is greater than or equal to a number.

SELECT CEIL(5.3) FROM DUAL;
o/p → 6

③ Floor(col|exp) → returns the largest integer value that is equal to or less than a number

SELECT FLOOR(5.8) FROM DUAL;
o/p → 5

④ Mod(m,n) → returns the remainder of m divide by n

SELECT MOD(10,2) FROM DUAL;
o/p → 0

⑤ Round(n[,m]) → returns a no. rounded to a certain no. of decimal places. m refers to decimal places

SELECT ROUND(10.678, 1) FROM DUAL; o/p → 10.7

SELECT ROUND(10.67) FROM DUAL; o/p → 11

⑥ Trunc(n[,m]) → returns a no. truncated to certain no. of decimal places.

SELECT TRUNC(10,678,1) FROM DUAL; o/p → 10.6

SELECT TRUNC(10.67) FROM DUAL; o/p → 10

May 2017

DATE FUNCTION

14 Sunday (134 - 231) Wk 20

↳ To get current system date use sysdate & select clause from dual

SELECT SYSDATE from Dual;

↳ Arithmetic opr can be performed with date

↳ Adding a no. with date increases the day.

eg. 10 May 2019 + 5 = 15 May 2019

↳ Subtract days from date

↳ Subtract 2 ~~day~~ dates gives no. of days in between.

(1) **Months_between** → no. of months b/w 2 dates

```
SELECT cid, cname, ROUND (months_between
       (sysdate , dob)/12) age FROM customer
       WHERE cid = 1;
```

o/p → 25

(2) **Add_months** → add calender months to date

```
SELECT ADD_MONTHS (SYSDATE, 4) FROM DUAL;
-- current date : 18-APR-19
```

o/p → 18-AUG-19

(3) **Next_date** → next date of the day specified

Monday 15 (120 - 230) Wk 20

```
SELECT NEXT_DATE (SYSDATE, 'monday')
FROM DUAL;
```

o/p → 23-APR-19

(4) **Last_day** → last day of month

```
SELECT LAST_DATE(SYSDATE) FROM DUAL;
```

o/p → 30-APR-19

(5) **Round** → Round date

```
SELECT ROUND(SYSDATE, 'year') FROM DUAL;
```

o/p → 01-JAN-19

```
SELECT ROUND(SYSDATE, 'month') FROM DUAL;
```

o/p → 01-MAY-19

(6) **Trunc** → Truncate date

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

o/p → 01-JAN-19

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

o/p → 01-APR-19

May 2017

16 Tuesday
(136 - 229) Wk 20

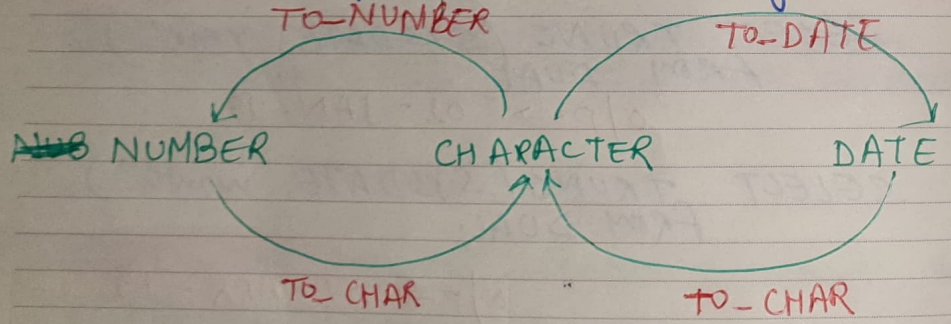
convert value from one datatype to another

CONVERSION FUNCTION

↳ Oracle server uses data of one data type where it expects data of a different data type.
↳ When this happen, oracle servers can automatically convert data to the expected data type.

→ This datatype conversion can be done implicitly by oracle server. or explicitly by user.

Function	Purpose
→ To_char(number|date, [fmt])	Converts a no. or date to string.
→ To_date(char,[fmt])	Converts a string to a date
→ To_number(char,[fmt])	Convert a string to a number



May

June

Wednesday 17
(137 - 228) Wk 20

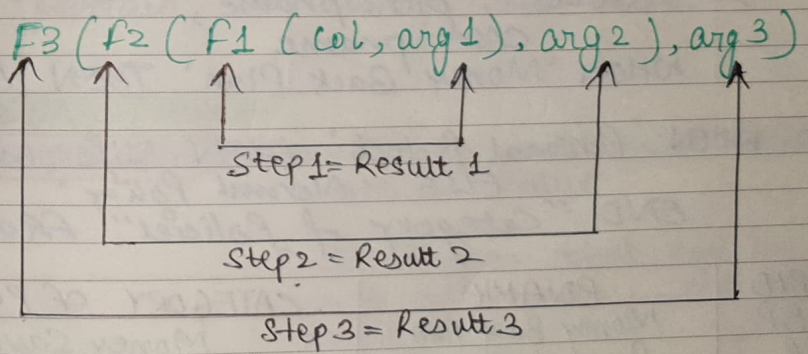
SELECT TO_CHAR(SYSDATE, 'dd month yyyy')
o/p → 18 APRIL 2019

SELECT TO_DATE('may-12-2019', 'mon-dd-yyyy')
FROM DUAL;
o/p → 12-MAY-19

SELECT TO_NUMBER('234.67', 999.99) FROM DUAL;
o/p → 234.67

NESTING FUNCTIONS

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



select cid, cname, round(months_between (sysdate, dob)/12) age from customer where cid = 1;

May 2017

18 Thursday
(138 - 227) Wk 20

CONDITIONAL EXPRESSION

Gives the use of IF-THEN-ELSE logic within SQL statement.

Two methods

CASE expression DECODE function

CASE EXPRESSION

↳ Oracle searches for the first WHEN & then 'THEN'.

- facilitates conditional inquiries by doing the work of an IF-THEN-ELSE statement.

```
>> SELECT pid, pname,
      CASE pname
   WHEN 'Money Back Plan' THEN 'Money Savings'
   WHEN 'Personal Protect' THEN 'Life Insurance'
      ELSE 'Normal Policy'
   END "Category of Policies" FROM policy;
```

PID	PNAME	CATEGORY OF POLICIES
MBP	Money Back Plan	Money Savings
PP	Personal Protect	Life Insurance

→ Conditional expression facilitates conditional inquiries by doing the work of a CASE or IF THEN-ELSE statement

DECODE FUNCTION

↳ func in oracle 8 is used to provide if-then-else type of logic to SQL.

↳ It is not available in MySQL or SQL Server

```
- SELECT pid, pname, DECODE(pname,
   'Money Back Plan', 'Money Saving',
   'Personal Protect', 'Life Insurance',
   'Normal Policy') "Category of Policies"
   FROM Policy;
```

GENERAL FUNCTION

These functions work with any data type & pertain to using null value.

NVL FUNCTION

1. NVL (exp1, exp2) → converts a null value to an actual value

- Data types that can be used are date, character, & number.
- Data types must match.

20 Saturday
(140 - 225) Wk 20

```
SELECT cid, pid, duedate,
paiddate, ~~paiddto~~ amount,
NVL(penalty, 0) penalty
FROM policyenrollment;
```

CID	PID	DUEDATE	PAIDDATE	AMOUNT	PENALTY
3	MBP	12-Dec-2017	11-Dec-2017	2000	0
1	PP	15-Mar-2018	12-Mar-2018	3000	0
2	PP	15-Feb-2018	22-Feb-2018	4000	200

(2) **NVL2 FUNCTION** → If exp1 is not null then return exp2. If exp2 is null then return exp3

NVL2(exp1, exp2, exp3)

```
SELECT cid, pid, duedate, NVL2(Paiddate,
'Paid', 'Not Paid') status FROM
policyenrollment;
```

CID	PID	DUEDATE	STAUS
3	MBP	12-Dec-2017	Paid
1	PP	15-Mar-2018	Paid
2	PP	15-Feb-2018	Not Paid

(3) **COALESE FUNCTION** → Returns the first not null expressions in the expression list.

```
SELECT cid, cname, COALESE(emailid,
TO_CHAR(phoneno), address) contact
FROM customer;
```

CID	CNAME	CONTACT
1	Tom	9876523190
2	John	john@yahoo.com
3	Ram	ram@gmail.com
4	Tiny	chennai

(4) **NULLIF** → Compares two expressions & returns null if they are equal or the first expression if they are not equal.

AGGREGATE OR GROUP FUNCTION

↳ Group functions operate on the entire set of rows on the table and gives one result for the entire set.
→ Group functions can be nested to a depth of two.

Various group function :- MAX, MIN, SUM, AVG, COUNT

Group functions ignore the null values.

AGGREGATE FUNCTION :-

Ques. I want to know what is the maximum & minimum penalty amount paid for late payment. Is there any way to get this information?

Sunday 21
(141 - 224) Wk 21

22 Monday
(142 - 223) Wk 21

```
SELECT MAX(penalty) as maximum,
MIN(penalty) as minimum
FROM policyenrollment;
```

MAXIMUM	MINIMUM
200	0

GROUP BY CLAUSE

- Until now, all the group fun(s) have treated the table as one large group of info.
- At times the table needs to be divided with information into smaller groups.
- This can be done by using the group by GROUP BY clause & with this column fun' results in single value for each group.
- If a group func is included in a SELECT clause along with other non grouped columns, then these non grouped column's should appear in the GROUP BY clause.
- SQL compiler generates an error if the non grouped columns are not included in the GROUP BY clause.
- The GROUP BY column does not have to be in the SELECT list.
- A column alias cannot be used in the GROUP BY clause.

Tuesday 23
(143 - 222) Wk 21

Example.

Ques. I want to know what is the max & min penalty amount paid for late payment each customer. Is there any way to get this result?

```
SELECT CID, MAX(penalty) as maximum,
MIN(penalty) as minimum FROM
policyenrollment GROUP BY CID;
```

HAVING CLAUSE

- The WHERE clause cannot be used to restrict groups. The group fun(s) cannot be in the WHERE clause.
- Conditions for groups must be specified in the HAVING clauses. HAVING clause of select group satisfy certain conditions.
- Having can specify any column func on any column in a table being queried. This column needs not be in SELECT list.

Ques. I want to know who all have paid a total premium amount greater than Rs. 2000. Is there any way to get this result?

```
SELECT CID, sum(amount) as maximum FROM
policyenrollment GROUP BY CID HAVING sum(amount)
>2000;
```

May 2017

24 Wednesday

(144 - 221) Wk 21

Things To remember:

1.) Group func can't be used in where clause

2.) All columns in the SELECT list must be in the GROUP BY clause apart from the group func.

3.) The GROUP BY clause divides the rows in a table into groups.

4.) We can use the group func(s) to return summary info. for each group.

JOINS & SUBQUERY IN SQL

Thursday 25

(145 - 220) Wk 21

JOINS - INTRODUCTION

Ques. I want to display customer name & name of the policy taken by each customer.

```
SELECT cname, pname FROM customer c
    JOIN policy p ON c.cid = p.cid;
```

1. A join is a query that combines records from two or more tables.
2. A join will be performed whenever multiple tables appear in the FROM clause of the query.
3. The select list of the query can select any columns from any of these tables
4. If join condition is omitted or invalid then a Cartesian product is formed. (all the rows in the first table are joined to all rows in second table).
5. If any 2 of these tables have a column name in common, then must qualify these columns throughout the query with table alias name to avoid ambiguity.
6. Most join queries contain at least one join condition either in the FROM clause or in the WHERE clause.