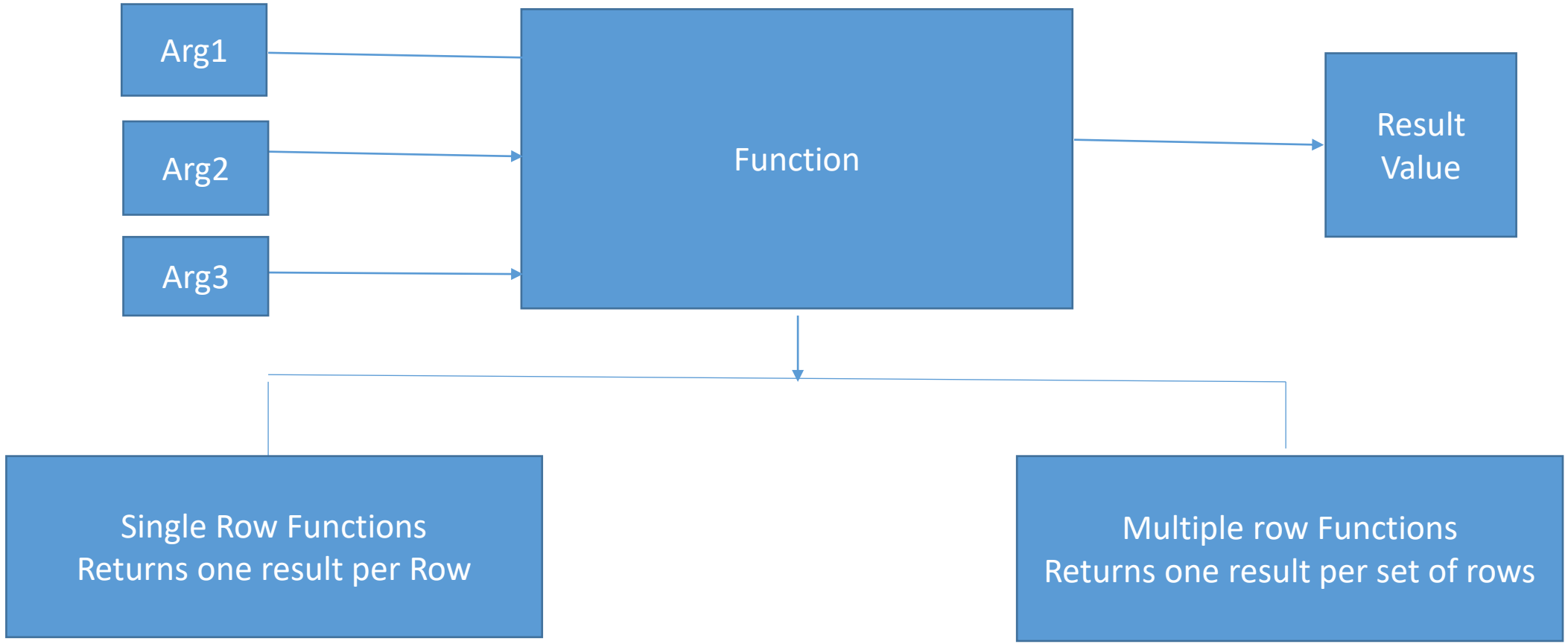
SQL Functions



Single Row Functions

- Manipulate Data Items
- Accept Arguments and return one value
- Return one result per row
- May modify data type
- Can be nested
- Accept arguments that can be column or an expression

Character Functions

- Case – manipulation functions

EX : LOWER , UPPER

- Character – manipulation functions

EX : CONCAT , SUBSTRING ,LEN, REPLACE , LEFT , RIGHT

Character Functions

- Case – manipulation functions

EX : LOWER , UPPER

LOWER (‘SQL Courses’) ---------- sql courses

UPPER (‘SQL Courses’) ---------- SQL COURSES

Character Functions

- Character – manipulation functions

EX : CONCAT , SUBSTRING ,LEN, REPLACE , LEFT , RIGHT

CONCAT (‘SQL ’ , ‘Courses’) ---------- SQL Courses

SUBSTRING(‘SQLCourses,1,3’) ---------- SQL

LEN(‘SQLCourses’) ---------- 10

REPLACE (‘Hello World’,’Hello’,’Ahmed’) ------- Ahmed World

LEFT (‘Ahmed’,3) ---------- ahm

RIGHT (‘Ahmed’,2) ---------- ed

Number Functions

- Number Functions

EX : ROUND , FLOOR , SQUARE , SQRT

ROUND (45.926 , 2) ---------- 45.93

FLOOR (45.926) ---------- 45

SQUARE (10) ---------- 100

SQRT (81) ---------- 9

Date Functions

- Date Functions

EX : GETDATE(), DateADD(),DateDIFF(),DatePART(),DateNAME()

,DAY() , MONTH() , YEAR()

GetDate() ---------- Date Of Today

DateAdd(day , 3 ,'4-12-2016') ---------- 4-15-2016

DateDiff(day,'1-1-2017','1-5-2017') ---------- 5

DatePart(dw,'12-28-2016') ---------- 4

DateName(dw,'12-28-2016')---------- 'WednesDay'

Day('4-3-2017') -3 Month ('4-5-2016') - 4 Year('12-5-2016') -2016

Convert Function & Nested Functions

Convert (varchar(50), 123) ---------- ‘123’

Select concat (Employeeid , UPPER(SUBSTRING(EmployeeName,1,4))))

Using the NULLIF() function

- The NULLIF() function accepts two parameters. If they are equal, then it returns a null; otherwise, it returns the first parameter.

```
select NULLIF(col1,col2) from null_tables

If col=col2 it will return null ,otherwise will return col1 value.
```

Using the ISNULL() function

- This function accepts a single expression and a substitution value. If the source is not equal to null, then the ISNULL() function passes the value on. However, if the source is null, then the second parameter is substituted for the null.
- ISNULL(source_expression, replacement_value)

```
SELECT FirstName, LastName, ISNULL(Nickname,'none')
FROM Customer

If nickname is null ,it will be replaced by 'none'
```

Using the COALESCE() function

- COALESCE() accepts a list of expressions or columns and returns the first non-null value, as follows:
- COALESCE(expression, expression, ...)

```
select id,coalesce(col1,col2,col3,0) from null_tables

Will return first non null value
```

What are Group Functions

- Group Functions operate On sets of Rows

To give one result per group

Ex : Maximum Salary In Employee Table

- Types : Avg ,Min,Max,Sum,Count

Group Function Syntax

Select [column,] , group_function(column)

From table

[Where Conditins]

[Group by column]

[Order by column]

Example On Functions

```
Select Min(salary) , Max(salary) , Avg(salary) , Sum(salary)
From Employee
Where City = ‘cairo’
```

```
Select Min(hiredate) , Max(hiredate)
From Employee
```

Using Count Function

- Count(*) returns Number of Rows In Table
- Count(column) returns number of rows with non null values for Expression
- Count (distinct [Column]) returns number of distinct non null values of the expression

Group Functions And NULL Values

- Select Avg(salary)
  From Employee

- Select avg(isnull(salary,0))
  From Employee

Creating Groups of Data

- Calculate Average Salary For Employees On Each Department
- All Columns in Select List that are not in group functions Must be in Group By Clause
- The Group By Column does not have to be in Select List

Ex : Select Departmentid ,Avg (salary)
From Employee
Group By Departmentid

Ex : Select Avg (salary)
From Employee
Group By Departmentid

Group By Functions On Multiple Columns

- Select departmentid , jobid , sum(salary)

  From Employee

  group by departmentid , jobid

Filtering grouped results

- uses the HAVING clause to filter the groups

```
select Dept_Id,Salary,count(*) from Instructor
where Salary is not null
group by Dept_Id,Salary
having  count(*) >8
```

Nesting Group Functions

- Display Maximum Average Salary

```
Select departmentid,max (avg(salary))
From Employee
Group By departmentid
```

Join Types

- Obtaining data From more than one Table Need to Join Between these Tables.
- Cross Join
- Inner Join
- Outer Join
- Self Join
- Not Equal Join

Inner Join Syntax

- It Select rows from the two tables that have equal values in matched columns.
- Two Columns Must have same Data type in two tables
- Select Empname,deptname
  from Employee , department
  where employee.deptid = department.deptid

- Select Empname,deptname
  from Employee [inner] join department
  on employee.deptid = department.deptid

Qualifying Ambiguous Column Names

- Use Table Prefix To qualify column names that are in multiple tables
- Use table prefixes to improve performance
- Use column aliases to distinguish column that have identical names but reside in different tables

Wrong Example

- Select Empname,deptname,deptid

from Employee [inner] join department

on employee.deptid = department.deptid

Solution

- Select Employee.Empname,department.deptname,employee.deptid

from Employee [inner] join department

on employee.deptid = department.deptid

Using Table Aliases

- Use Table Aliases to simplify queries
- Use table aliases to improve performance

- Select E.Empname,D.deptname,E.deptid
  from Employee E [inner] join department D
  on E.deptid = D.deptid

Self Join Syntax

- To find Name Of employee’s manager , you need to join Employees table to itself

- Select E.Empname,M.Empname

  from Employee E , Employee M

  where E.managerid = M.employeeid

NonEquiJoins

- A nonequijoin is a join condition containing something other than an equality operator

- Select E.Empname , E.salary, G.Level

  from Employee E , grades G

  where E.salary between G.lowsal and G.highsal

Self Join Syntax

- To find Name Of employee’s manager , you need to join Employees table to itself

- Select E.Empname,M.Empname

  from Employee E , Employee M

  where E.managerid = M.employeeid

Outer Join

- Left Outer Join
- Select E.Empname,D.Deptname

  from Employee E Left outer Join Department D

  On E.Deptid = D.Deptid

- Right Outer Join
- Select E.Empname,D.Deptname

  from Employee E Right outer Join Department D

  On E.Deptid = D.Deptid

- Full Outer Join
- Select E.Empname,D.Deptname

  from Employee E Full outer Join Department D

  On E.Deptid = D.Deptid

Cartesian Products

- A Cartesian product is formed when :
- A join condition is omitted
- A join condition is invalid
- All rows in the first table that are joined with all rows in the second table

To Avoid Cartesian Product , always include valid join condition

Select Empname , Deptname

From Employee cross join Department