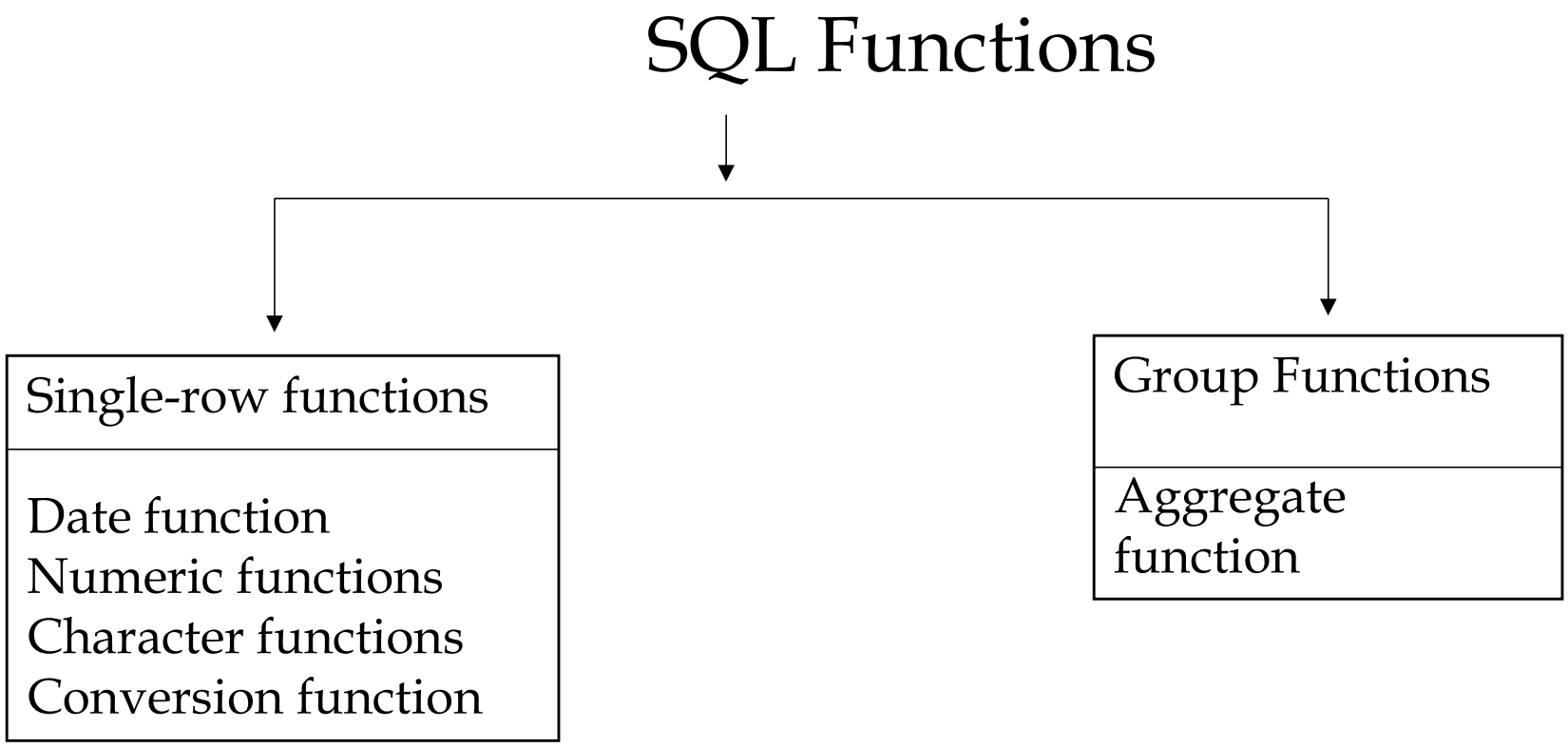
SQL FUNCTIONS



Aggregate functions

- It is used for grouping the data.
- It cannot be combined with another aggregate functions and it cannot be used with where clause.

- Sum()
- Avg()
- Max()
- Min()
- Count()

Aggregate functions

Syntax	Description
count (*), count (column name), count (distinct column name)	Returns number of rows
min (column name)	Min value in the column
max (column name)	Max value in the column
avg (column name)	Avg value in the column
sum (column name)	Sum of column values
stdev(column name)	Standard deviation value
variance(column name)	Variance value

Aggregate functions

- Select sum(sal) from emp;
- Select avg(sal) from emp;
- Select max(sal) from emp;
- Select min(sal) from emp;
- Select count(sal) from emp;

(it will count the number of non-empty values)

Group by Clause : The GROUP BY clause in SQL is used to arrange identical data into groups. It is typically used in conjunction with aggregate functions to perform operations on each group of data.

Specify the name of the attribute based on which we want to create the group. The columns in the SELECT statement must either be aggregated or included in the GROUP BY clause.

Find the total salary of the instructors in each school

```
select school_name, sum(salary) from employees group by
school_name;
```

Find the average salary and employee id of the instructors in each school

```
select emp_id, avg(salary) from employees group by
school_name, emp_id;
```

```
SELECT department_id, AVG(salary) AS average_salary FROM
employees GROUP BY department_id;
```

Having Clause : The Having clause in SQL is used to filter results after an aggregation has been performed by the GROUP BY clause.

Example : Find the names and average salaries of all the departments where total salary is greater than 1000000

```
select department, avg(salary) from employees group_by
department having sum(salary) > 1000000.
```

Date functions

`Syntax	Description
add_months(date,no. of months)	Return the date after adding the number of months `Select add_months(doj,36) from emp;`
months_between (date1,date2)	Returns the numeric value of the difference between the months. `select months_between('01-may-83','01-jan-83') from dual;` MONTHS_BETWEEN('01-MAY-83','01-JAN-83') ---------- 4
round(date, [format])	Format – 'day', 'month' , 'year' rounded to the nearest format specified

```
SQL> select round(to_date('11-feb-1990'),'yyyy') from
dual;
```

01-JAN-90, here its rounded to the nearest year, if month before June(including June) to the nearest year. From July 01 onwards to the nearest year. That would be the next year jan 1st.

```
SQL> select round(to_date('11-jul-1990'),'mm') from dual;
```

01-JUL-90

```
SQL> select round(to_date('16-jul-1990'),'mm') from dual;
```

01-AUG-90

```
select round(to_date('22-feb-1990 15:35:32','dd-mm-yyyy
HH24:MI:SS'),'dd') from dual;
```

23-FEB-90

```
select round(to_date('22-feb-1990 05:35:32','dd-mm-yyyy
HH24:MI:SS'),'dd') from dual;
```

22-FEB-90

Syntax	Description

next_day(date, day)	Returns the next date of the day
	The next occurrence of Friday to the dob of an employee.
	
	`SELECT ssn, NEXT_DAY(dob, 'fri')`
	`FROM employee;`

trunc(date, [format])	Format – ‘day’, ‘month’ , ‘year’ Day – That day midnight Month – first day of the month Year – first day of the year
greatest(date1, date2,…)	Returns the latest date

```
SQL> select trunc(to_date('22-jul-1990','dd-mm-yyyy'),'mm')from dual;

SQL> select next_day(to_date('16-jun-1990'),'mon') from dual;

SQL> select greatest(to_date('01-jan-2023','dd-mm-yyyy'),to_date('01-feb-
2024','dd-mm-yyyy')) from dual;
```

Numeric functions

Syntax	Description
abs ( )	Returns the absolute value
ceil ( )	Rounds the argument
cos ( )	Cosine value of argument
cosh( )	Hyperbolic cos
exp ( )	Exponent value
floor( )	Truncated value
power (m,n)	n raised to m
mod(m,n)	Remainder of m / n
round (m,n)	Rounds m's decimal places to n
trunc(m,n)	Truncates m's decimal places to n
sqrt(m)	Square root value

Character functions

Syntax	Description
initcap (char)	Changes first letter to capital
lower (char)	Changes to lower case
upper (char)	Changes to upper case
ltrim (char, set)	Removes the set from left of char
rtrim (char, set)	Removes the set from right of char
replace(char, search string, replace string)	Replaces the search string to new
substring(char, m , n)	Returns chars from m to n length
lpad(char, length, special char)	Pads special char to left of char to Max of length
rpad(char, length, special char)	Pads special char to right of char to Max of length
length(char)	Length of string
||	Concatenation of strings

```
SELECT name || ' is ' || TO_CHAR(age) AS description FROM employee;
```

The Project names in Upper case.

```
SELECT UPPER(name) FROM project;
```

Department names with left padded stars.

```
SELECT LPAD(name,15,'*') department FROM department;
```

The first five characters of employee first names.

```
SELECT SUBSTR(fname,0,5) FROM employee;
```

The length of longest department name.

```
SELECT MAX(LENGTH(name)) FROM department;
```

Conversion Functions

Syntax	Description
to_char(date, format)	`select to_char(to_date('11-JUN-2024'),'dd-mm/yyyy')  from dual;`
to_date(char,format)	to_date (‘1991-08-02’,’yyyy-mm-dd’)
to_number(char)	Converts a numerical string to number `select to_number('34') from dual;`

Views

View is an database object, is a virtual table.

It is a subset of a table derived from the original large table for convenient manipulation. View doesn't store any data.

Difference between table and view is a table can store data while a view doesn't.

Can be used to simplify complex queries.

```
CREATE OR REPLACE VIEW <view name> AS <query>;

CREATE OR REPLACE VIEW <view name> (alias column name) AS
<query>;
```

To create read only view:

```
CREATE OR REPLACE VIEW <view name> AS <query> WITH READ
ONLY;
```

SQL> `create view tv as select * from t;`
View created.

SQL> `select * from tv;`

NO	NAME	SAL	HRA	DA	NET	DOJ	DPN
10	rash	320				12-SEP-89	15
20	jon	200				30-AUG-98	30

Insert into view

SQL> `insert into tv values (50,'wad',600,null,null,null,'18-mar-75',25);`

1 row created.

Delete from view

SQL> `delete from tv where no = 10;`

1 row deleted.

Modify the view

SQL> `update tv set sal = 400 where no = 20;`

1 row updated.