# SQL DML (2)

Bases de datos

# Operadores aritméticos y de concatenación

- Obtener el salario, aumentándolo un 5% para aquellos empleados que cobran menos de 1000

```
SELECT SAL*1.05
FROM EMP
WHERE SAL < 1000;
```

- Algo operado con NULL es NULL

| OPERADOR | |
|---|---|
| + | Suma |
| - | Resta |
| * | Multiplicación |
| / | División |

# Composición de consultas

- Operadores conjuntistas
  - UNION
  - INTERSECT
  - EXCEPT (diferencia)

```
consulta1
{UNION | INTERSECT | EXCEPT}  [ALL | DISTINCT]
consulta2
```

- Consultas con estructura compatible
- Selecciona empleados con salario mayor que 3000 o comisión mayor que 0

```
SELECT ename,sal,comm
FROM emp
where sal>3000
UNION
SELECT ename,sal,comm
FROM emp
where comm>0
```

# Funciones

- Pueden formar parte de una expresión
- Se pueden incluir en SELECT, WHERE y ORDER BY
- Numéricas
  - ABS(n) Valor absoluto
  - CEIL(n) Redondeo superior
  - FLOOR(n) Redondeo inferior
  - ROUND(n,decimales) Redondea al número más cercano, con los decimales especificados
  - MOD(m,n) Resto de la división entera
  - POWER(m,n) Potencia
  - SQRT(n) Raíz cuadrada
  - TRUNCATE(n,decimales) Trunca el número a la cantidad de decimales especificada

# Funciones

- De cadenas de caracteres
  - CONCAT(cad1,cad2) Devuelve las dos cadenas unidas
  - LOWER(cad) Pasa la cadena a minúsculas
  - UPPER(cad) Pasa la cadena a mayúsculas
  - LENGTH(cad) Devuelve la longitud de la cadena
- De fechas
  - NOW() Obtiene la fecha y hora actual
  - DATE_FORMAT(fecha,formato) Formatea la fecha

```
mysql> SELECT DATE_FORMAT('2009-10-04 22:23:00', '%W %M %Y'); -> 'Sunday October 2009'

mysql> SELECT DATE_FORMAT('2007-10-04 22:23:00', '%H:%i:%s'); -> '22:23:00'

mysql> SELECT DATE_FORMAT('1900-10-04 22:23:00', -> '%D %y %a %d %m %b %j'); -> '4th 00 Thu 04 10 Oct 277'

mysql> SELECT DATE_FORMAT('1997-10-04 22:23:00', -> '%H %k %I %r %T %S %w'); -> '22 22 10 10:23:00 PM 22:23:00 00 6'

mysql> SELECT DATE_FORMAT('1999-01-01', '%X %V'); -> '1998 52'

mysql> SELECT DATE_FORMAT('2006-06-00', '%d'); -> '00'
```

# Funciones

- De NULL
  - IFNULL(expr1,expr2)
    - Si expr1 no es NULL, devuelve expr1
    - Si es null, devuelve expr2

```
SELECT IFNULL(1,0);
+-------------+
| IFNULL(1,0) |
+-------------+
|           1 |
+-------------+

SELECT IFNULL(NULL,10);
+-----------------+
| IFNULL(NULL,10) |
+-----------------+
|              10 |
+-----------------+

SELECT IFNULL(1/0,10);
+----------------+
| IFNULL(1/0,10) |
+----------------+
|        10.0000 |
+----------------+

SELECT IFNULL(1/0,'yes');
+-------------------+
| IFNULL(1/0,'yes') |
+-------------------+
| yes               |
+-------------------+
```

# Funciones de agrupamiento o colectivas

- También llamadas agregadas, de columna, de resumen…
- Las funciones vistas hasta ahora obtienen un valor para cada fila

```
SELECT   SQRT(sal)        ==>  Para cada fila, obtiene la raíz cuadrada
FROM     emp                   del salario de cada empleado
```

- Las funciones de agrupamiento obtienen un valor que hace referencia a un conjunto de filas

```
SELECT   MAX(sal)         ==>  Obtiene una única fila con el máximo
FROM     emp                   de los salarios de los empleados
```

- Permiten ALL y DISTINCT: funcion([ALL | DISTINCT] expresion)
  - Si aparece DISTINCT se eliminan los valores repetidos antes de calcular la función

# Funciones de agrupamiento o colectivas

- SUM: suma
- AVG: media
- MAX: máximo
- MIN: mínimo
- COUNT: número de valores. Se puede usar como COUNT(*), que incluye NULL.

# Ejemplos

1.– Obtén la media de los salarios del departamento 30.

```
SELECT      AVG(sal)
FROM        emp
WHERE       deptno = 30
```

2.– Obtén la media de los salarios y el nombre alfabéticamente más alto de los empleados cuyo salario es mayor que 1500.

```
SELECT      AVG(sal), MAX(ename)
FROM        emp
WHERE       sal > 1500
```

3.– Obtén la media de los salarios de los empleados.

```
SELECT      AVG(sal)
FROM        emp
```

# Ejemplos

5.– !?

```
SELECT     MAX(sal), sal
FROM       emp
```

¿Cuál es el problema?

Ej.: Dada la tabla `t1` con una columna `C` y cuatro filas:

```
C
----
NULL
1
2
2
```

la siguiente consulta, obtendrá:

```
SELECT COUNT(*), COUNT(c), COUNT(ALL c), COUNT(DISTINCT c)
FROM   t1
```

| COUNT(*) | COUNT(C) | COUNT(ALLC) | COUNT(DISTINCTC) |
|---|---|---|---|
| 4 | 3 | 3 | 2 |