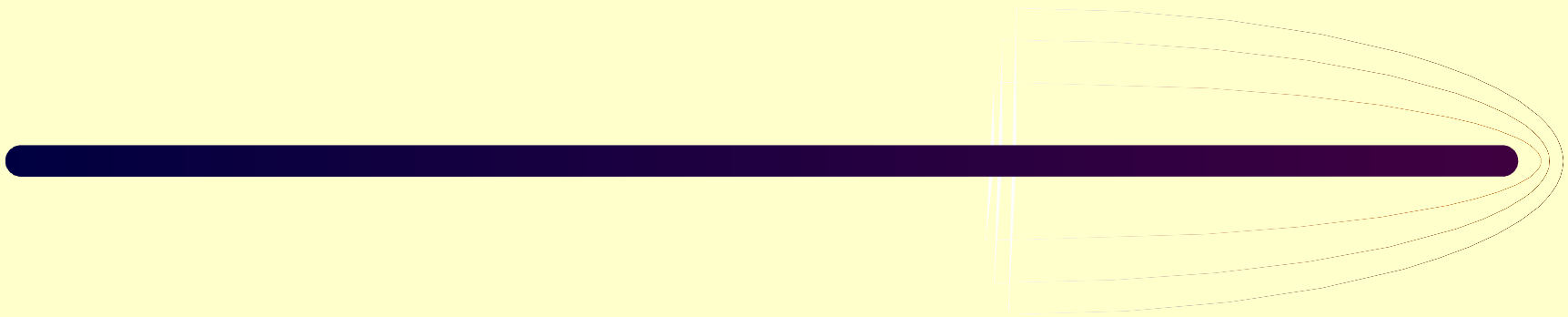
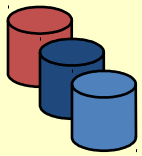
Práctica 2



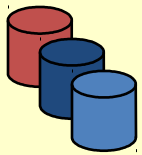
SENTENCIA SELECT COMPLETA

SELECT [ALL | DISTINCT | UNIQUE] <lista de selecciones>
FROM <nombre de tabla> [alias de tabla] [,...]
[WHERE <condición>]
[GROUP BY <lista-de columnas> [HAVING <condición>]]
[ORDER BY <nombre de columna> [ASC | DESC] [,...]]



RESUMEN

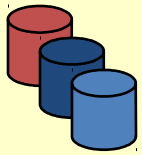
- **SELECT:** indica qué atributos o funciones se van a recuperar.
- **FROM:** especifica todas las relaciones (tablas) que se necesitan para la consulta (no las de las consultas “anidadas”)
- WHERE: especifica las condiciones, si es que son necesarias, para seleccionar las filas de esas tablas, incluyendo las condiciones de reunión.
- GROUP BY: especifica los atributos de agrupación, en tanto que HAVING indica una condición que deben cumplir los grupos seleccionados no las filas individuales. (**NO** siempre que se pone GROUP BY tiene que existir un HAVING).
- ORDER BY: especifica un orden para presentar el resultado de la consulta.



Orden de evaluación

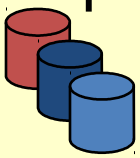
El **orden de evaluación** de las cláusulas de SELECT es el siguiente:

- **1º FROM** → selecciona las tablas implicadas
- **2º WHERE** → descarta las filas que no cumplan la condición impuesta
- **3º GROUP BY** → crea grupos con las filas que quedan tras el where, según las columnas de agrupamiento
- **4º HAVING** → descarta los grupos que no satisgan la condición impuesta
- **5º SELECT** → evalúa para cada grupo las expresiones de la lista de selección produciendo una sola fila por grupo
- **6º ORDER BY** → ordena los datos de acuerdo al criterio establecido
- Muestra los datos



Base de datos para ejemplos: semántica

- Deseamos controlar los proyectos en los que se trabaja en una empresa. Nos interesará conocer datos de los empleados así como de los departamentos a los que pertenecen. Así mismo, necesitaremos información de los proyectos en los que se trabaja así como un control del personal que trabaja en ellos.
- De los empleados necesitaremos saber: número de empleado, nombre, departamento al que pertenece y salario.
- De los departamentos dispondremos de su código, nombre y presupuesto.
- De cada proyecto conoceremos su número, nombre, horas previstas de ejecución y fecha de inicio
- El control de proyectos consistirá en conocer el número de horas dedicadas por los empleados en los diferentes proyectos en los que participe.



Base de datos para ejemplos: tablas

DEPARTAMENTO:

DEPT	DNOMBRE	BUDGET
D1	Marketing	10
D2	Contabilidad	12
D3	Recursos Humanos	5

EMPLEADO:

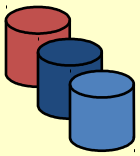
EMPNO	ENOMBRE	DEPT	SALARIO
E1	López	D2	500
E2	Goñi	D3	700
E3	Aramburu	D2	350
E4	Arrarás	D1	1000

PROYECTO:

NPROY	PNOMBRE	HORASPR	FECINI
1	Albaranes	300	2002/05/20
2	Acuse de pedidos	220	2003/01/05
3	Facturas	90	2002/11/13

CONTROL:

EMPNO	NPROY	HORASTR
E1	1	15
E1	3	30
E2	2	20
E3	1	50

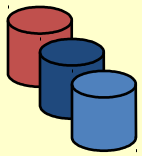


FUNCIONES DE AGREGACIÓN

- Las funciones de agregación son funciones que toman un conjunto de valores como entrada y producen **un único** valor de salida.
- Estas funciones se pueden usar en las cláusulas **SELECT** o **HAVING (NUNCA** en la cláusula **WHERE)**
- SQL proporciona cinco funciones de agregación:
  - Media: **avg**
  - Mínimo: **min**
  - Máximo: **max**
  - Total: **sum**
  - Cuenta: **count**

- Las funciones **sum, max, min y avg** se aplican a un conjunto o multiconjunto de valores numéricos

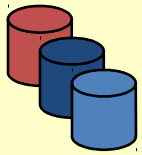
- Las funciones **max** y **min** se pueden usar con columnas que tengan dominios no numéricos si los valores del dominio tienen un orden total entre ellos.



FUNCIONES DE AGREGACIÓN

AVG([**DISTINCT** | **ALL**] *n*)	Valor promedio de *n*, ignorando nulos.
COUNT(* | [**DISTINCT** | **ALL**] *expr*)	Número de filas, en donde la *expresión* evalúa a un valor diferente de nulo. Si se coloca * cuenta todas las filas incluyendo duplicados y filas con nulos.
MAX([**DISTINCT** | **ALL**] *expr*)	Máximo valor de la expresión ignorando nulos.
MIN([**DISTINCT** | **ALL**] *expr*)	Mínimo valor de la expresión ignorando nulos.
SUM([**DISTINCT** | **ALL**] *n*)	Suma de valores de *n* ignorando nulos

- Los tipos de datos para ***expr*** son **CHAR**, **VARCHAR2**, **NUMBER** y **DATE**.
- Los tipos de datos para ***n*** son numéricos.



Ejemplos

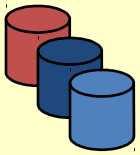
Consulta:

- Hallar la suma de los salarios de todos los empleados, el salario máximo, el salario mínimo y el salario medio.

```
SELECT  sum(SALARIO) SUMA, max(SALARIO) MAXIMO, min(SALARIO)
MINIMO, avg(SALARIO)  MEDIA
FROM EMPLEADO
```

Resultado:

SUMA	MAXIMO	MINIMO	MEDIA
2550	1000	350	637,5



Ejemplos

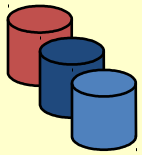
Consulta:

- Hallar de todos los empleados del departamento de Contabilidad la suma de los salarios, el salario máximo, el salario mínimo y el salario medio.

```
SELECT  sum(SALARIO) SUMA, max(SALARIO) MAXIMO, min(SALARIO)
MINIMO, avg(SALARIO) MEDIA
FROM EMPLEADO, DEPARTAMENTO
WHERE EMPLEADO.DEPT=DEPARTAMENTO.DEPT and
        DNOMBRE='Contabilidad'
```

Resultado:

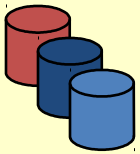
SUMA	MAXIMO	MINIMO	MEDIA
850	500	350	425



F. DE AGREGACIÓN- Consideraciones

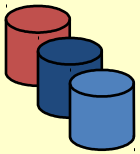
- La existencia de valores ***nulos*** complica el procesamiento de la funciones de agregación. Depende de la implementación de SQL. En general, las funciones de agregación tratan los valores nulos según la siguiente regla:
  - Todas las funciones de agregación excepto **count(*)** ignoran los valores nulos de los datos de entrada.
  - Como resultado de ignorar valores nulos, la colección de valores de entrada puede ser vacía.
  - El cálculo de **count** de una colección vacía se define como 0, y todas las demás funciones de agregación devuelven un valor nulo cuando se aplican sobre una colección de datos vacía

  El efecto de los valores nulos en algunas construcciones más complejas de SQL puede ser más sutil.
- También el uso de valores ***nulo*** en operaciones de comparación causa varias complicaciones, en algunas implementaciones el resultado de una comparación que involucre un valor nulo es un resultado desconocido, y en otras implementaciones el resultado es falso.



Cláusulas GROUP BY y HAVING

- La cláusula **GROUP BY** seguida de una lista de atributos permite agrupar las tuplas en grupos que tengan los mismos valores en todos los atributos de esa lista. Dichos atributos reciben el nombre de *atributos de agrupación*.
  - Importante: en la mayoría de los SGBDs exigen que las cláusulas **GROUP BY y SELECT** contengan los mismos atributos (columnas).
- La cláusula **HAVING** permite establecer una condición sobre los grupos de manera que sólo se seleccionan aquellos grupos que la cumplen



Ejemplos

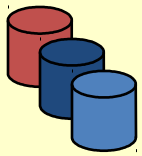
Consulta:

- Obtener los salarios medios por departamento indicando cuantos empleados hay en cada uno de ellos.

```
SELECT  DEPT, count(*) NUMEMP, avg(SALARIO) MEDIA
FROM EMPLEADO
GROUP BY DEPT
```

Resultado:

DEPT	NUMEMP	MEDIA
D2	2	425
D3	1	700
D1	1	1000



- Las filas de EMPLEADO se dividen en grupos y cada uno de ellos tiene el mismo valor en el atributo de agrupación DEPT.
- Las funciones **count** y **avg** se aplican a cada uno de esos grupos.

Ejemplos

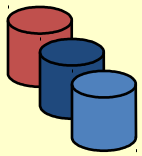
Consulta:

- De cada proyecto obtener su código, su nombre y el número de empleados que trabajan en él.

```
SELECT CONTROL.NPROY , PNOMBRE, COUNT(*)
FROM PROYECTO, CONTROL
WHERE PROYECTO.NPROY=CONTROL.NPROY
GROUP BY CONTROL.NPROY, PNOMBRE
```

Resultado:

CONTROL.NPROY	PNOMBRE	COUNT(*)
1	Albaranes	2
2	Acuse de pedidos	1
3	Facturas	1



- En la cláusula GROUP BY hay que poner CONTROL.NPROY, PNOMBRE obligatoriamente porque los grupos que queremos hacer incluyen los dos atributos, el código y nombre de proyecto.

Ejemplos

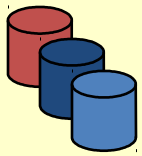
Consulta:

- De cada proyecto en el que trabajen al menos dos empleados, obtener su código, su nombre y el número de empleados que trabajan en él.

```
SELECT CONTROL.NPROY , PNOMBRE, COUNT(*)
FROM PROYECTO, CONTROL
WHERE PROYECTO.NPROY=CONTROL.NPROY
GROUP BY CONTROL.NPROY, PNOMBRE
HAVING COUNT(*)>=2
```

Resultado:

CONTROL.NPROY	PNOMBRE	COUNT(*)
1	Albaranes	2

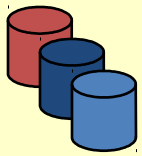


Consideraciones

- **CUIDADO:**
  - La cláusula **WHERE se ejecuta primero**, para seleccionar filas individuales.
  - La cláusula **HAVING se aplica después**, para seleccionar grupos de filas .

- **Ejemplo:**

```
SELECT DNOMBRE, COUNT(*)
FROM DEPARTAMENTO D, EMPLEADO E
WHERE D.DEPT=E.DEPT and SALARIO >400
GROUP BY DNOMBRE
HAVING COUNT(*) > 5
```



- ¿Qué realiza la consulta?

Consideraciones

- A primera vista puede parecer:

  "Obtener el número total de empleados cuyos sueldos sean mayores de 400 en cada departamento, pero sólo en el caso de departamentos que el número de empleados que trabajan sea mayor que 5"

- Cuando realmente es:

"seleccionar sólo los departamentos que tengan más de 5 empleados cuyo salario sea superior a 400"

  - Primero selecciona los empleados con sueldo mayor de 400
  - De esos, si en un departamento trabajan más de 5 empleados (con sueldo >400) selecciona dicho departamento como resultado de la consulta.