

***Actividad 1: Resolver en SQL los ejercicios propuestos en “Ejemplos de funciones de agregación base de datos alumnos”:***

- SQL:

1. Nota media y desviación típica de la nota de acceso de los alumnos varones (redondeado a 2 decimales con round):

Hoja de Trabajo

Generador de Consultas






```
select round(avg(nota),2), round(stddev(nota),2) from alumnos where sexo='v';
```

Resultado de la Consulta x






SQL | Todas las Filas Recuperadas: 1 en 0,034 segundos

	ROUND(AVG(NOTA),2)	ROUND(STDDEV(NOTA),2)
1	6,85	1,33

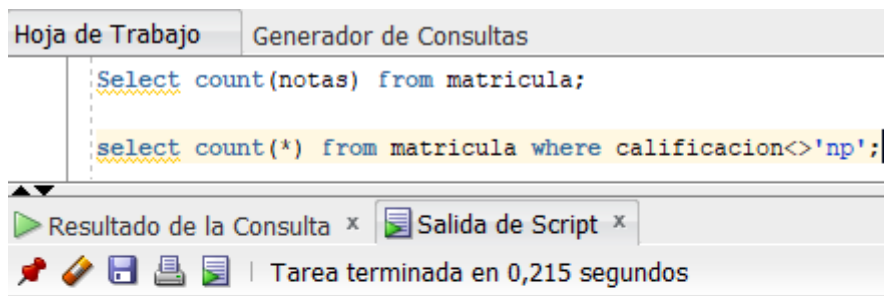
2. Número máximo y mínimo de créditos por asignatura:

Hoja de Trabajo		Generador de Consultas	
		<code>select min(cred+credpr),max(cred+credpr) from asigna;</code>	
<div>Resultado de la Consulta x Salida de Script x</div> <div>       Tarea terminada en 0,141 segundos</div>			
MIN (CREDIT+CREDPR)		MAX (CREDIT+CREDPR)	
-----		-----	
4,5		12	

3. Número de instancias de la tabla matrícula:

Hoja de Trabajo		Generador de Consultas	
		<pre>select count(*) from matricula;</pre> <pre>Select count(calificacion) from matricula;</pre>	
<div>Resultado de la Consulta x Salida de Script x</div> <div>       Tarea terminada en 0,216 segundos</div>			
<div>COUNT (*)</div> <div>-----</div> <div>240</div>			
<div>COUNT (CALIFICACION)</div> <div>-----</div> <div>240</div>			

4. Número de instancias de matrícula a las que se han presentado los alumnos:



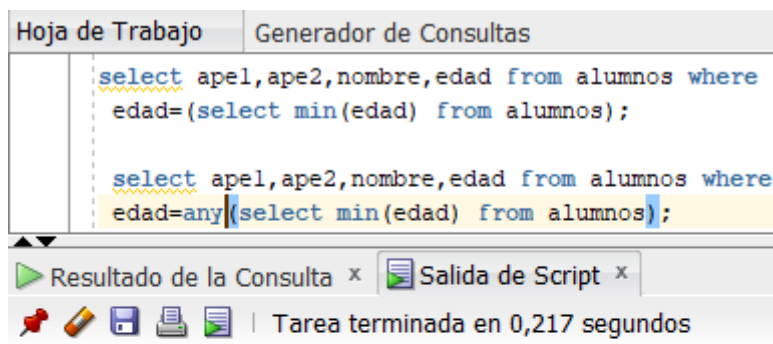
COUNT (NOTAS)

-----  
209

COUNT (\*)

-----  
209

5. Alumnos de edad mínima (hacedlo mediante un operador de agregación y comparar con formas anteriores):



APE1	APE2	NOMBRE	EDAD
Alvarez	Perez	Miguel	18

APE1	APE2	NOMBRE	EDAD
Alvarez	Perez	Miguel	18

## 6. Alumnos con nota de entrada menor o igual que la media de las notas de entrada:

Hoja de Trabajo | Generador de Consultas

```
select apel,ape2,nombre,nota from alumnos where nota<=(select avg(nota) from alumnos);
```

```
select apel,ape2,nombre,nota from alumnos where nota<=any(select avg(nota) from alumnos);
```

Salida de Script x

Tarea terminada en 0,259 segundos

APE1	APE2	NOMBRE	NOTA
Lopez	Perez	Jose	5,55
Alvarez	Perez	Miguel	6,55
Perez	Jimenez	Juan	4,55
Alvarez	Perez	Jose	5,55
Alvarez	Sanchez	Roque	6,55
Martinez	Perez	Antonio	5,55
Perez	Jimenez	Luis	6,87

7 filas seleccionadas.

APE1	APE2	NOMBRE	NOTA
Lopez	Perez	Jose	5,55
Alvarez	Perez	Miguel	6,55
Perez	Jimenez	Juan	4,55
Alvarez	Perez	Jose	5,55
Alvarez	Sanchez	Roque	6,55
Martinez	Perez	Antonio	5,55
Perez	Jimenez	Luis	6,87

7 filas seleccionadas.

## 7. Alumnos matriculados de más de 4 asignaturas en el curso académico 2016-2017:

Hoja de Trabajo | Generador de Consultas

```
select apel,ape2,nombre,edad from alumnos where 4<any(select count(*) from matricula where alumnos.dni=matricula.dni and curso_academico='2016-2017');
```

```
select apel,ape2,nombre,edad from alumnos where (select count(*) from matricula where alumnos.dni=matricula.dni and curso_academico='2016-2017') >4;
```

Salida de Script x

Tarea terminada en 0,223 segundos

APE1	APE2	NOMBRE	EDAD
Alvarez	Sanchez	Roque	19
Perez	Jimenez	Luis	26

APE1	APE2	NOMBRE	EDAD
Alvarez	Sanchez	Roque	19
Perez	Jimenez	Luis	26

8. Asignaturas que tienen o han tenido matriculados más de 25 alumnos, con independencia del curso académico:

Hoja de Trabajo    Generador de Consultas

```
select asi#,nombreas from asigna where 25<any(select count(*) from matricula where asi#=codasi#);
select asi#,nombreas from asigna where (select count(*) from matricula where asi#=codasi#)>25;
```

Salida de Script x

Tarea terminada en 0,227 segundos

```
ASI# NOMBREAS
-----
mab1 Modelos avanzados I
tec1 Tecnicas de Gestion I
tec2 Tecnicas de Gestion II
mab2 Modelos Avanzados II
```

```
ASI# NOMBREAS
-----
mab1 Modelos avanzados I
tec1 Tecnicas de Gestion I
tec2 Tecnicas de Gestion II
mab2 Modelos Avanzados II
```

9. Asignaturas con número máximo de créditos:

Hoja de Trabajo    Generador de Consultas

```
select asi#,nombreas,cred+credpr creditos from asigna where
(cred+credpr)=(select max(cred+credpr) from asigna);
```

Salida de Script x

Tarea terminada en 0,149 segundos

ASI# NOMBREAS	CREDITOS
tec1 Tecnicas de Gestion I	12
tec2 Tecnicas de Gestion II	12

**Actividad 2: De la relación de problemas Ejercicios Álgebra Relacional\_Cálculo Relacional resolver en Álgebra Relacional y SQL las cuestiones del ejercicio 4.**

a) Encontrar entre qué dos ciudades se realiza el viaje más largo.

\*AR:

$$\rho(Ruta) = R$$

$$\rho(Ruta) = Ru$$

$$\pi_{ciudad-sal, ciudad-llg} \left( (Ruta) - \pi_{R.km} \left( \left( \pi_{(R.km < Ru.km)} (R \times Ru) \right) \right) \right)$$

\*SQL:

select ciudad\_sal, ciudad\_llg from Ruta where Ruta.km = Any(  
select max(Ru.km) from Ruta Ru);

o'  
select ciudad\_sal, ciudad\_llg from Ruta where Ruta.km >= All(  
select Ru.km from Ruta Ru);

b) Listar los nombres de los conductores que hayan llevado todos los camiones de la empresa.

\*AR:

$$\pi_{nombre} (Conductor \bowtie (\pi_{ANE, matricula} (Viaje) \div \pi_{matricula} (Vehiculo)))$$

\*SQL:

select nombre from Conductor where not exists (  
select matricula from Vehiculo where not exists (  
select \* from Viaje where Vehiculo.matricula = Viaje.matricula and Conductor.ANE = Viaje.ANE));

o'  
select nombre from Conductor where not exists (  
select matricula from Vehiculo  
minus  
select matricula from Viaje where Conductor.ANE = Viaje.ANE);



c) Encontrar qué días de la semana se hacen viajes entre Granada y Sevilla por la mañana (antes de las 13h).

\* AR:

$$\pi_{\text{día-sem}} \left( \pi_{\left( \underset{\vee}{\text{ciudad-sal} = \text{'Granada'} \wedge \text{ciudad-llg} = \text{'Sevilla'} \wedge \text{hora-llg} \leq 13 \right)} (\text{Ruta} \times \text{Prog-Viaje}) \right)$$

$$(\text{ciudad-sal} = \text{'Sevilla'} \wedge \text{ciudad-llg} = \text{'Granada'} \wedge \text{hora-llg} \leq 13)$$

\* SQL:

select día-sem from Prog-Viaje, Ruta where (ciudad-sal = 'Granada' and ciudad-llg = 'Sevilla' and hora-llg <= to\_date('13','HH'))  
or (ciudad-sal = 'Sevilla' and ciudad-llg = 'Granada' and hora-llg <= to\_date('13','HH'))

d) Encontrar las rutas que se hacen todos los días de la semana, suponiendo que hay viajes todos los días.

\* AR:

$$\rho(\text{Prog-Viaje}) = \text{PV}$$

$$\text{Ruta} \bowtie (\pi_{\text{Ruta\#, día-sem}} (\text{Prog-Viaje}) \div \pi_{\text{día-sem}} (\text{PV}))$$

\* SQL:

select \* from Ruta where not exists (

select día-sem from Prog-Viaje where not exists (

select \* from Prog-Viaje PV where Ruta.Ruta# = PV.Ruta# and Prog-Viaje.día-sem = PV.día-sem ))

ó  
select \* from Ruta where not exists (

select día-sem from Prog-Viaje

minus

select PV.día-sem from Prog-Viaje PV where Ruta.Ruta# = PV.Ruta# );

**Actividad 3: Crear la base de datos de Jugadores de Baloncesto (los scripts están en Prácticas SQL) y resolver en Álgebra y SQL los ejercicios de 3.60 a 3.64 del cuaderno de prácticas.**

3.60) Muestra la información disponible acerca de los encuentros de la liga.

\*AR:

$\pi_*(Encuentros)$

\*SQL:

select \* from Encuentros;

3.61) Muestra los nombres de los equipos y de los jugadores ordenados alfabéticamente.

\*AR:

$\pi_{nombreE, nombreJ}(Equipos \bowtie Jugadores)$

\*SQL:

select nombreE, nombreJ from Equipos, Jugadores where Jugadores.codE = Equipos.codE order by nombreE, nombreJ;

3.62) Muestra los jugadores que no tienen ninguna falta:

\*AR:

$\rho(Jugadores) = Jug$

$Jug \bowtie (\pi_{codJ}(Jugadores) - \pi_{codJ}(\sigma_{faltas > 0}(Alimaciones)))$

\*SQL:

select \* from Jugadores where codJ not in (select codJ from Alimaciones where faltas > 0);

o  
select \* from Jugadores where codJ in (

select codJ from Jugadores

minus

select codJ from Alimaciones where faltas > 0);

3.63) Muestra los compañeros del equipo del jugador que tiene por código  $x$  ( $\text{codJ} = 'x'$ ) y donde  $x$  es uno elegido por ti.

\* AR:

$$\rho(\text{Jugadores}) = \text{Ju}$$

$$\pi_{\text{codJ}}(\text{Ju}) \bowtie \left( \pi_{\text{codE}}(\text{Jugadores}) - \pi_{\text{codE}}(\pi_{\text{codJ} \neq 'x'}(\text{Jugadores})) \right)$$

\* SQL:

```
select codJ from Jugadores where codE in (
  select codE from Jugadores
  minus
  select codE from Jugadores where codJ <> 'x');
```

o

```
select codJ from Jugadores where codE not in ( select codE from Jugadores where codJ <> 'x');
```

o

```
select codJ from Jugadores where codE in (
  select codE from Jugadores where codJ in (
    select codJ from Jugadores where codJ = 'x'));
```

3.64) Muestra los jugadores y la localidad donde juegan (la de sus equipos).

\* AR:

$$\pi_{\text{nombreJ}, \text{localidad}}(\text{Equipos} \bowtie \text{Jugadores})$$

\* SQL:

```
select nombreJ, localidad from Equipos, Jugadores where Jugadores.codE = Equipos.codE;
```