



UNIVERSITAT DE
BARCELONA

Módulo I:

Tarea – Plataforma como Servicio PaaS en AWS

Estudiante:

Heiner Romero Leiva

Máster en Big Data & Data Science

Noviembre, 2021

Descripción:

Has sido contratado como Data Engineer por una empresa dedicada al mundo del deporte. Dentro de su estrategia definida para los próximos años, está empezando un proyecto para migrar sus aplicaciones a la nube. Para ello, ha elegido Amazon Web Services (AWS) como proveedor de servicios en la nube.

La multinacional tiene en sus instalaciones una base de datos relacional **PostgreSQL**, donde almacenan resultados de diferentes deportes.

El CDO (Chief Data Officer) ha decidido migrar a AWS, utilizando el servicio **Amazon RDS** (**Amazon Relational DataBase Service**) por las ventajas que ofrece usar un servicio PaaS.

Para la migración, han hecho una exportación de las diferentes tablas a ficheros de texto. Una de ellas contiene los resultados de la Liga Española de fútbol.

Campo	Descripción	Valor Ejemplo
IdPartido	Identificador del partido	1
temporada	Temporada	1970-71
Jornada	Jornada	1
EquipoLocal	Nombre del equipo local	Athletic Club
EquipoVisitante	Nombre del equipo visitante	Barcelona
golesLocal	Goles del equipo local	1
golesVisitante	Goles del equipo visitante	1
fecha	Fecha del partido	12/09/1970
timestamp	Timestamp del registro	21942000.0

Las tareas que te han asignado son las siguientes:

- Crear una base de datos PostgreSQL en AWS utilizando el servicio Amazon RDS.

IMPORTANTE

Debes elegir la opción que indique “Free Tier”. Si eliges otra opción, AWS te cargará los costes asociados.

- Una vez creada la base de datos, crear una tabla para almacenar los datos.
- Importar los datos del archivo “Partidos.txt” en la tabla creada.
- Crear las consultas necesarias para poder responder a las siguientes preguntas:
 1. ¿Cuántos goles ha marcado el Barcelona?
 2. ¿Cuántos partidos han terminado 0-0?
 3. ¿En qué temporada se han marcado más goles?
 4. ¿Cuál es el equipo que tiene el record de meter más goles como local? ¿Y como visitante?
 5. ¿Cuál son las 3 décadas en las que más goles se metieron?
 6. ¿Qué equipo es el mejor local en los últimos 5 años?
 7. ¿Cuál es la media de victorias por temporada en los equipos que han estado menos de 10 temporadas en 1^a división?
 8. ¿Quién ha estado más temporadas en 1^a División: Barcelona o Real Madrid?
 9. ¿Cuál es el record de goles como visitante en una temporada del Real Madrid?
 10. ¿En qué temporada se marcaron más goles en Cataluña?
 - Goles marcados y recibidos por el Barcelona jugando de local.
 - Goles marcados y recibidos por el Espanyol jugando de local.

Pasos para crear una base de datos PostgreSQL en AWS utilizando el servicio Amazon RDS.

1. Ingresar a AWS con el login de usuario y la contraseña.
2. En la pestaña de búsqueda escribir: “RDS” y darle click.

The screenshot shows the AWS Management Console search results for "rds". The search bar at the top has "rds" typed into it. Below the search bar, there are two main sections: "Servicios" and "Características".

- Servicios:**
 - RDS: Servicio de bases de datos relacionales administrado
 - AWS Glue DataBrew: Herramienta de preparación de datos visuales para limpiar y normalizar datos para a...
 - Amazon OpenSearch Service (sucesor de Amazon Elasticsearch Service): Ejecute y escale clústeres de OpenSearch y Elasticsearch (sucesor de Amazon Elasti...
 - Application Discovery Service: Detecte el inventario y las dependencias de las aplicaciones en las instalaciones
- Características:**
 - Proxies: Característica RDS
 - Bases de datos: Característica RDS
 - Copias de seguridad automatizadas: Característica RDS
 - Panel: Característica RDS

On the right side of the search results, there is a sidebar with the heading "Manténgase conectado a los recursos de AWS en cualquier lugar" which includes a note about the mobile app supporting multiple regions. Below this, there are sections for "Explore AWS" featuring Amazon Redshift, AWS Fargate, and AWS Marketplace.

3. Dar click donde dice: “Crear base de datos”.

The screenshot shows the AWS RDS Management Console for creating a new database. The left sidebar is titled "Amazon RDS" and contains several navigation options: Panel, Bases de datos, Editor de consultas, Información sobre rendimiento, Instantáneas de, Copias de seguridad automatizadas, Instancias reservadas, Proxies, Grupos de subredes, Grupos de parámetros, Grupos de opciones, Zonas de disponibilidad personalizada, Versiones de motor personalizadas, Eventos, Suscripciones a eventos, Recomendaciones, and Actualización del certificado.

The main content area is titled "Amazon Aurora" and provides an overview: "Amazon Aurora es una base de datos de nivel empresarial compatible con MySQL y PostgreSQL. Está disponible a partir de <\$1/day. Aurora supports up to 64TB of auto-scaling storage capacity, 6-way replication across three availability zones, and 15 low-latency read replicas." It includes a prominent orange button labeled "Crear base de datos".

Below this, there are sections for "Recursos" (Resources) and "Información adicional" (Additional Information). The "Recursos" section lists various Amazon RDS resources used in the US East region, such as instances, storage, and VPC configurations. The "Información adicional" section links to documentation and other resources.

At the bottom, there is a "Crear base de datos" (Create database) button, a note about launching instances in the US East region, and a "Estado de los servicios" (Service status) section with a "Ver panel de estado del servicio" (View service status) link.

4. Seleccionar “creación estándar” y PostgreSQL.

The screenshot shows the AWS RDS Management Console interface. The user is in the 'Create database' section. In step 1, 'Creación estándar' is chosen. In step 2, 'PostgreSQL' is selected from the engine options. In step 3, the version 'PostgreSQL 12.6-R1' is chosen. The interface includes standard browser navigation and AWS service links at the top.

5. Escoger la opción de la capa gratuita.

The screenshot shows the continuation of the 'Create database' wizard. Step 4, 'Plantillas', shows 'Capa gratuita' selected. Step 5, 'Disponibilidad y durabilidad', shows 'Capa gratuita' selected under 'Opciones de implementación'. Step 6, 'Configuración', lists 'Identificador de instancias de bases de datos'. The interface remains consistent with the previous screenshot, showing the AWS logo and various service links at the top.

- Darle un nombre a la base de datos y crear una contraseña. Se debe dejar la instancia de db.t2.micro.

Identificador de instancias de bases de datos Información
Escriba un nombre para la instancia de base de datos. El nombre debe ser único en relación con todas las instancias de base de datos pertenecientes a su cuenta de AWS en la región de AWS actual.

database-1

El identificador de la instancia de base de datos no distingue entre mayúsculas y minúsculas, pero se almacena con todas las letras en minúsculas (como en "minInstanciadebd"). Restricciones: de 1 a 60 caracteres alfanuméricos o guiones. El primer carácter debe ser una letra. No puede contener dos guiones consecutivos. No puede terminar con un guion.

▼ Configuración de credenciales

Nombre de usuario maestro Información
Escriba un ID de inicio de sesión para el usuario maestro de la instancia de base de datos.

postgres

De 1 a 16 caracteres alfanuméricos. El primer carácter debe ser una letra.

Generación automática de contraseña
Amazon RDS puede generar una contraseña en su nombre, o bien puede especificar su propia contraseña.

Contraseña maestra Información

Restricciones: debe tener al menos 8 caracteres ASCII imprimibles. No puede contener ninguno de los siguientes caracteres: / (barra diagonal), \ (corillas simples), " (dóbles comillas) y @ (signo de arroba).

Confirmar contraseña Información

Clase de instancia de base de datos

Clase de instancia de base de datos Información

Clases estándar (incluye clases m)
 Clases optimizadas para memoria (incluye clases r y x)
 Clases con ráfagas (incluye clases t)

db.t2.micro
1 vCPU 1 GiB RAM Not EBS Optimized

Incluir clases de generación anterior

- Se deja por defecto las 20 GBs y el 1000 en el Umbral de almacenamiento máximo, asimismo se deja el default VPC.

Almacenamiento

Tipo de almacenamiento Información
SSD de uso general (gp2)
Rendimiento de referencia determinado por el tamaño del volumen

Almacenamiento asignado
20 GiB
(Mínimo: 20 GiB; máximo: 16,384 GiB) Un almacenamiento asignado mayor **puede mejorar** el rendimiento de IOPS.

Escalado automático de almacenamiento Información
Proporciona compatibilidad con el escalado dinámico para el almacenamiento de la base de datos en función de las necesidades de la aplicación.

Habilitar escalado automático de almacenamiento
Habilitar esta característica permitirá que el almacenamiento aumente una vez que el umbral especificado se supera.

Umbral de almacenamiento máximo Información
Los cargos se aplicarán cuando la base de datos escala automáticamente el umbral especificado.
1000 GiB
Mínimo: 21 GiB. Máximo: 16,384 GiB

Conectividad

Virtual Private Cloud (VPC) Información
La VPC que define el entorno de red virtual para esta instancia de base de datos.

Default VPC (vpc-0905b988741633cb)

Sólo se muestran las VPC con grupos de subredes de base de datos correspondientes.

Después de crear una base de datos, no puede cambiar su VPC.

Grupo de subredes Información
Grupo de subredes de base de datos que define las subredes e intervalos de IP que puede usar la instancia de base de datos en la VPC seleccionada.

default-vpc-0905b988741633cb

8. Se selecciona la opción: “Elegir existente” y debe dejar el acceso público marcado como un “si” de lo contrario no podrá conectarse a la base de datos luego.

Virtual Private Cloud (VPC) Información
La VPC que define el entorno de red virtual para esta instancia de base de datos.
Default VPC (vpc-09050988741653cd)

Después de crear una base de datos, no puede cambiar su VPC.

Grupo de subredes Información
Grupo de subredes de base de datos que definen las subredes e intervalos de IP que puede usar la instancia de base de datos en la VPC seleccionada.
default-vpc-09050988741653cd

Acceso público Información
 Sí Los dispositivos y las instancias de Amazon EC2 que están fuera de la VPC se pueden conectar a su base de datos. Elija uno o varios grupos de seguridad de VPC que especifiquen los dispositivos y las instancias EC2 dentro de la VPC que pueden conectarse a la base de datos.
 No RDS no asignará una dirección IP pública a la base de datos. Solo los dispositivos y las instancias de Amazon EC2 que están dentro de la VPC pueden conectarse a su base de datos.

Grupo de seguridad de VPC
Elija un grupo de seguridad de VPC para permitir el acceso a la base de datos. Asegúrese de que las reglas del grupo permiten el tráfico entrante adecuado.

Elegir existente Elija grupos de seguridad de VPC existentes
 Crear nuevo Crea un grupo de seguridad nuevo de VPC

Grupos de seguridad de VPC existentes
Elegir grupos de seguridad de VPC
default X

Zona de disponibilidad Información
Sin preferencia

9. En el Puerto de la base dejar el Puerto por defecto que es 5432.

Configuración adicional

Puerto de la base de datos Información
Puerto TCP/IP que la base de datos usará para las conexiones de las aplicaciones.
5432

Autenticación de bases de datos

Opciones de autenticación de bases de datos Información
 Autenticación con contraseña Se autentica con las contraseñas de las bases de datos.
 Autenticación de bases de datos con contraseña e IAM Se autentica con las credenciales de usuario y la contraseña de las bases de datos a través de usuarios y roles de AWS IAM.
 Autenticación Kerberos y con contraseña

10. En cuanto a las opciones de bases de datos se deja todo tal y como está por defecto.

The screenshot shows the AWS RDS Management Console interface. A new database instance is being created with the following settings:

- Nombre de base de datos inicial:** onlinereexercise
- Grupo de parámetros de base de datos:** default.postgres12
- Grupo de opciones:** default:postgres-12
- Copia de seguridad:** Enabled (checkbox checked)
- Periodo de retención de copia de seguridad:** 7 días
- Periodo de copia de seguridad:** Sin preferencia (radio button selected)
- Copiar las etiquetas en las instantáneas:** Enabled (checkbox checked)
- Replicación de copias de seguridad:** Deshabilitado (checkbox unchecked)
- Autenticación con contraseñas:** Enabled (radio button selected)

At the bottom, there are links for [Comentarios](#), [Español](#), and a large orange [Crear base de datos](#) button.

11. Se le da click al botón de “crear base de datos”.

The screenshot shows the final confirmation step before creating the database instance:

- Opciones de autenticación de bases de datos:** Autenticación con contraseñas (radio button selected)
- Costos mensuales estimados:** Summary of free tier usage and estimated monthly costs.
- Aviso legal:** A note stating that the user is responsible for ensuring they have the necessary rights for the service.

At the bottom, there are two buttons: [Cancelar](#) and the large orange [Crear base de datos](#) button.

12. Hay que esperar hasta que la base de datos se cree, este proceso puede tardar entre 2 a 4 minutos.

The screenshot shows the AWS RDS Management Console. On the left, there's a sidebar with various options like 'Panel', 'Bases de datos', 'Editor de consultas', etc. The main area has a green header bar stating 'Se ha creado correctamente la base de datos database-1.' Below it, a table lists the database details:

Identificador de base de datos	Rol	Motor	Región y AZ	Tamaño	Estado	CPU	Actividad actual	Mantenimiento	VPC
database-1	Instancia	PostgreSQL	us-east-1f	db.t2.micro	Disponible	7.00%	0.00 sessions		vpc-0

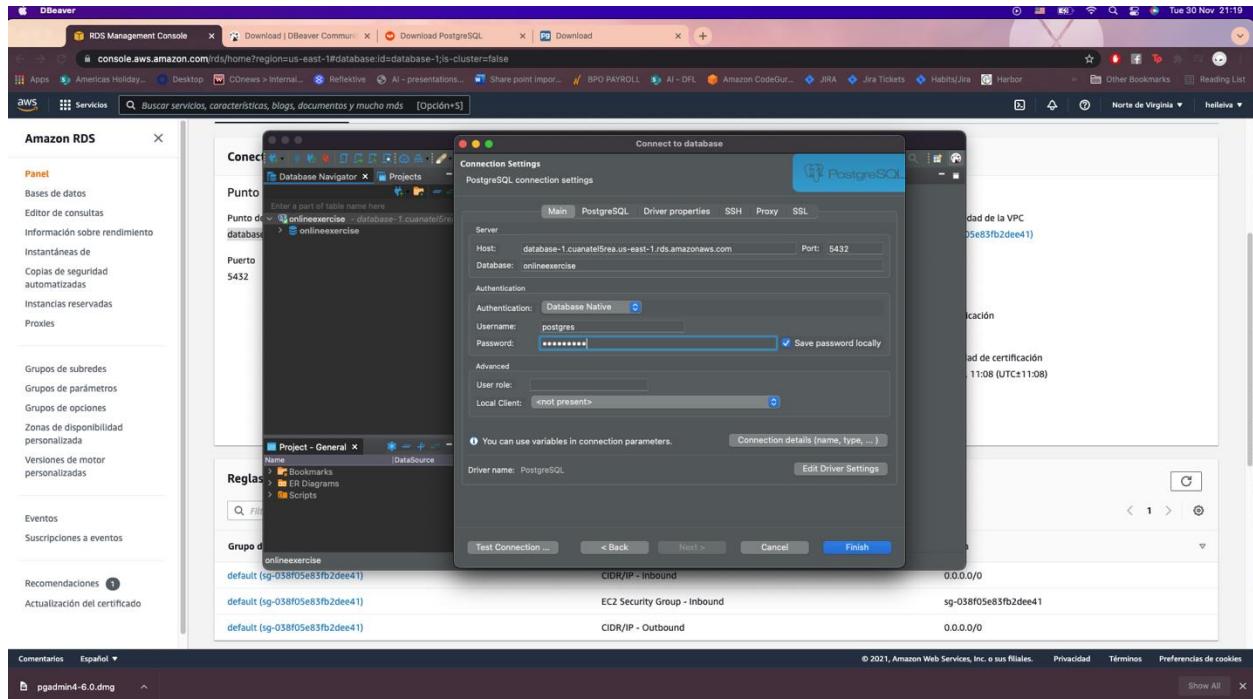
13. Una vez creada se deben editar las reglas de entrada y añadir el “Todos los TCP” y la etiqueta: “0.0.0.0/0” en caso de que no se haga este paso el interprete de SQL no reconocerá la conexión.

The screenshot shows the AWS EC2 Management Console. It's navigating through 'Grupos de seguridad' to a specific security group 'sg-038f05e83fb2dee41 - default'. The current page is 'Editar reglas de entrada'. There are two existing rules listed:

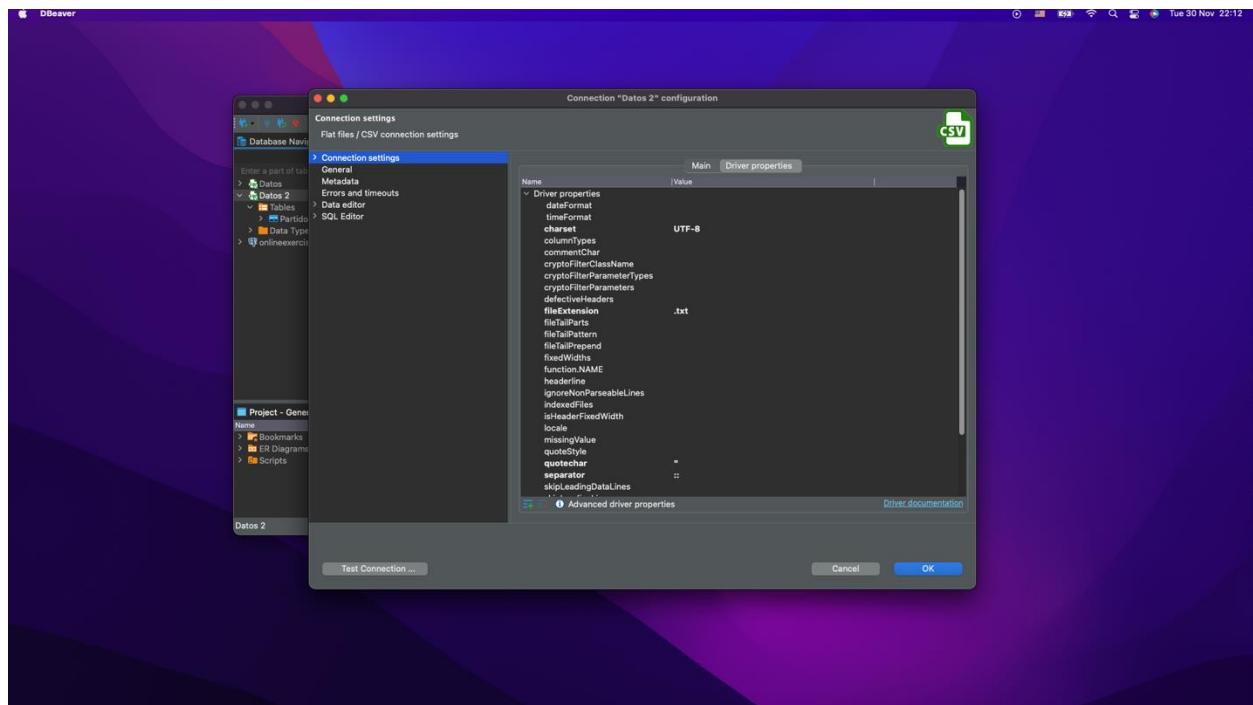
ID de la regla del grupo de seguridad	Tipo	Protocolo	Intervalo de puertos	Origen	Descripción: opcional
sgr-0367d9921a9180bf5	Todos los TCP	TCP	0 - 65535	Personalizada	0.0.0.0/0
sgr-0577df786d89a7be3	Todo el tráfico	Todo	Todo	Personalizada	sg-038f05e83fb2dee41

At the bottom, there are buttons for 'Cancelar', 'Previsualizar los cambios', and 'Guardar reglas'.

14. Finalmente se ingresa con el host, el nombre de la base de datos y el password y se le da “finish”.



15. Cuando se importe el archive, hay que cambiar el file extension a .txt y en separator colocar “::”.



16. Finalmente se tiene la base de datos conectada.

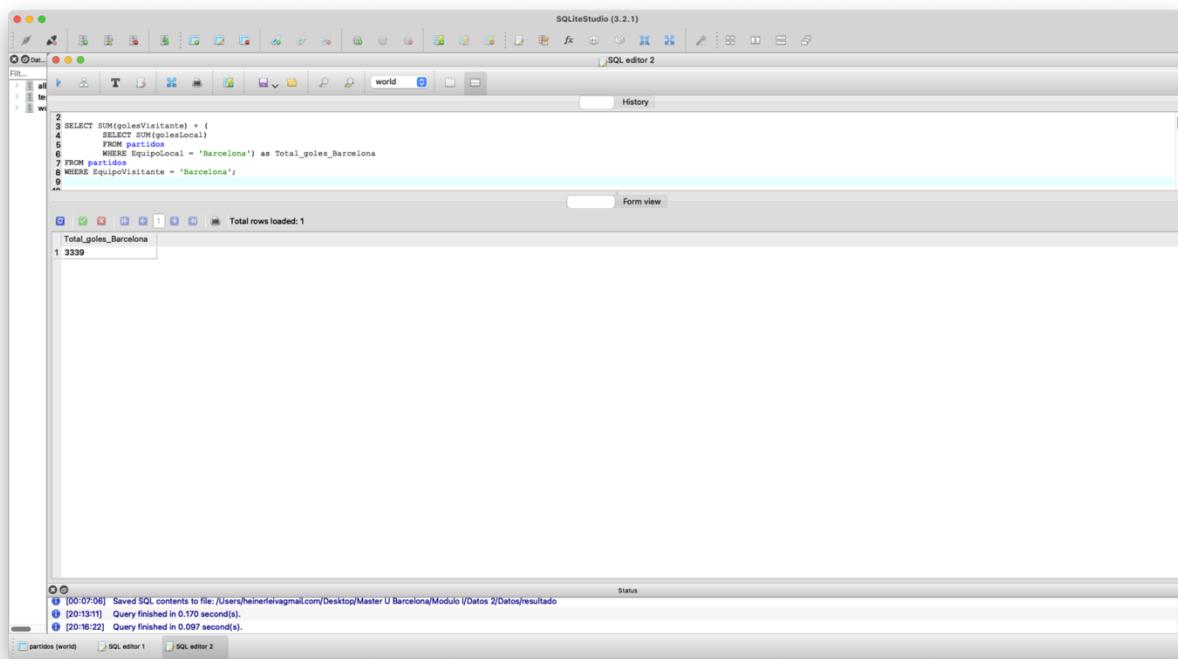
The screenshot shows the DBeaver 7.3.1 interface with the 'Partidos' table selected in the Database Navigator. The main window displays the 'Partidos' table grid with 31 rows of data. The columns are labeled: idPartido, mtemporada, mJornada, mEquipoLocal, mEquipoVisitaante, mgolesLocal, mgolesVisitaante, and mfec. The data includes various football teams from the 1970-71 season. The bottom status bar indicates 200 row(s) fetched in 1ms (+8ms). The Project - General tab in the sidebar shows a single dataSource entry.

idPartido	mtemporada	mJornada	mEquipoLocal	mEquipoVisitaante	mgolesLocal	mgolesVisitaante	mfec
1	1970-71	1	Athletic Club	Barcelona	1	1	12/09
2	1970-71	1	Las Palmas	Atletico de Madrid	1	1	12/09
3	1970-71	1	Real Madrid	Valencia	2	0	12/09
4	1970-71	1	Celta de Vigo	Sporting de Gijon	2	0	13/09
5	1970-71	1	Elche	Granada	1	1	13/09
6	1970-71	1	Espanyol	Sevilla	0	1	13/09
7	1970-71	1	Sabadell	Real Sociedad	0	0	13/09
8	1970-71	1	RCD Zaragoza	CD Malaga	0	0	13/09
9	1970-71	2	Valencia	Las Palmas	5	1	19/09
10	1970-71	2	Atletico de Madrid	Athletic Club	2	0	20/09
11	1970-71	2	Barcelona	Real Zaragoza	5	2	20/09
12	1970-71	2	Sporting de Gijon	Sabadell	3	2	20/09
13	1970-71	2	Granada	Espanyol	1	0	20/09
14	1970-71	2	CD Malaga	Celta de Vigo	2	1	20/09
15	1970-71	2	Real Sociedad	Elche	2	0	20/09
16	1970-71	2	Sevilla	Real Madrid	3	1	20/09
17	1970-71	3	Las Palmas	Athletic Club	1	1	26/09
18	1970-71	3	Real Madrid	Granada	3	2	26/09
19	1970-71	3	Valencia	Sevilla	0	1	26/09
20	1970-71	3	Celta de Vigo	Barcelona	1	1	27/09
21	1970-71	3	Elche	Sporting de Gijon	5	0	27/09
22	1970-71	3	Espanyol	Real Sociedad	0	0	27/09
23	1970-71	3	Sabadell	CD Malaga	5	2	27/09
24	1970-71	3	Real Zaragoza	Atletico de Madrid	0	1	27/09
25	1970-71	4	Athletic Club	Real Zaragoza	3	0	04/10
26	1970-71	4	Atletico de Madrid	Celta de Vigo	3	1	04/10
27	1970-71	4	Barcelona	Sabadell	4	1	04/10
28	1970-71	4	Sporting de Gijon	Espanyol	0	1	04/10
29	1970-71	4	Granada	Valencia	2	2	04/10
30	1970-71	4	Las Palmas	Sevilla	0	0	04/10
31	1970-71	4	CD Malaga	Elche	1	0	04/10

Ejercicios SQL

1. ¿Cuántos goles ha marcado el Barcelona?

Pregunta ambigua, porque no se especifica si como local o visitante, así que se hace un total general. El Barcelona ha anotado 3339 goles (tomando en cuenta goles locales y de visitante).



The screenshot shows the SQLiteStudio interface. In the top-left pane, there is a tree view with 'partidos' selected. In the bottom-left pane, there is a table named 'partidos (world)' with one row containing 'Total_goles_Barcelona' and '3339'. The status bar at the bottom shows three log entries: '[00:07:06] Saved SQL contents to file:/Users/heinerfeivagmail.com/Desktop/Master U Barcelona/Modulo I/Datos 2/Datos/resultado', '[20:13:11] Query finished in 0.170 seconds.', and '[20:16:22] Query finished in 0.097 seconds.'.

```
SELECT SUM(golesVisitante) + (SELECT SUM(golesLocal)
FROM partidos WHERE EquipoLocal = 'Barcelona') as Total_goles_Barcelona
FROM partidos
WHERE EquipoVisitante = 'Barcelona';
```

Query:

```
SELECT SUM(golesVisitante) + (SELECT SUM(golesLocal)
FROM partidos WHERE EquipoLocal = 'Barcelona') as Total_goles_Barcelona
FROM partidos
WHERE EquipoVisitante = 'Barcelona';
```

Forma alternativa:

Se puede obtener el mismo resultado diviendo el query por goles visitante y goles local:

The screenshot shows the SQLiteStudio interface. The SQL editor contains the following query:

```
1 -- ¿Cuántos goles ha marcado el Barcelona?
2 -- Fórmula alternativa
3 SELECT SUM(golesVisitante) AS Cantidad_goles_Visitante_Barcelona
4 FROM partidos
5 WHERE EquipoVisitante = 'Barcelona'
6
```

The results pane shows a single row of data:

Cantidad_goles_Visitante_Barcelona
1296

Total rows loaded: 1

The status bar at the bottom shows three completed queries with their execution times.

Query:

```
SELECT SUM(golesVisitante) AS Cantidad_goles_Visitante_Barcelona
FROM partidos
WHERE EquipoVisitante = 'Barcelona'
```

The screenshot shows the SQLiteStudio interface. The SQL editor contains the following query:

```
1 -- ¿Cuántos goles ha marcado el Barcelona?
2 -- Fórmula alternativa
3 SELECT SUM(golesLocal) AS Cantidad_goles_Local_Barcelona
4 FROM partidos
5 WHERE EquipoLocal = 'Barcelona'
```

The results pane shows a single row of data:

Cantidad_goles_Local_Barcelona
2043

Total rows loaded: 1

The status bar at the bottom shows three completed queries with their execution times.

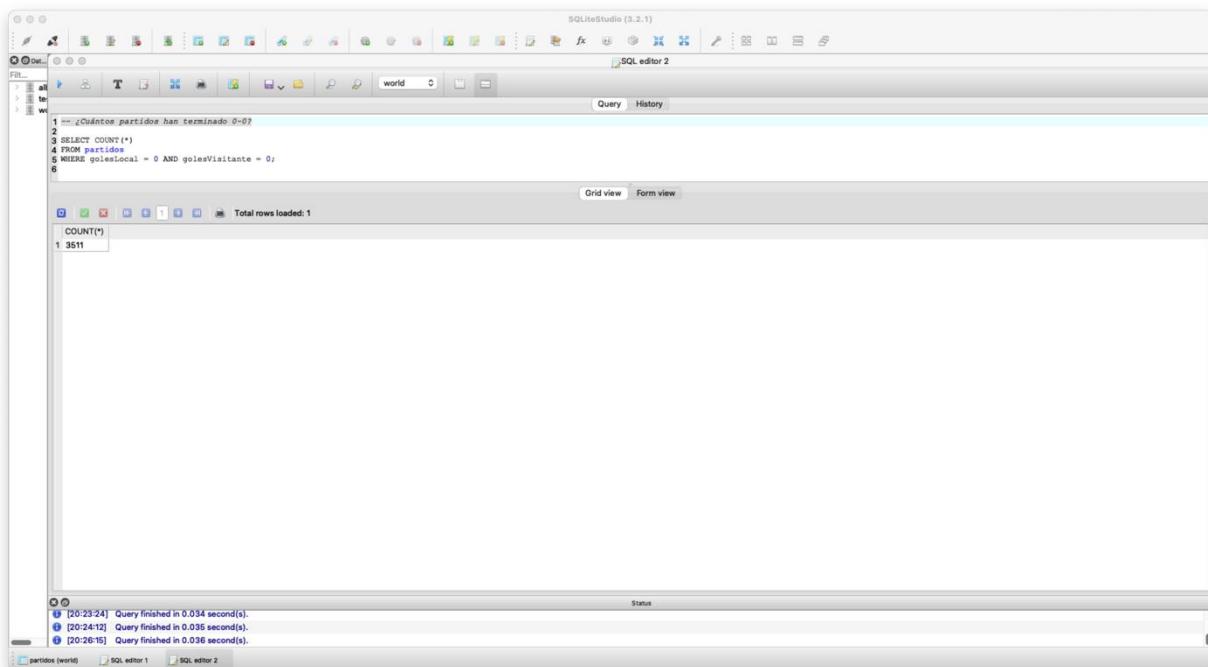
Query:

```
SELECT SUM(golesLocal) AS Cantidad_goles_Local_Barcelona
FROM partidos
WHERE EquipoLocal = 'Barcelona'
```

Si sumamos $2043 + 1296 = 3339$ goles anotados, lo mismo que nos dio la primera consulta con el sub select.

2. ¿Cuántos partidos han terminado 0-0?

La cantidad asciende a los 3511 partidos.



The screenshot shows the SQLiteStudio interface with the following details:

- Toolbar:** Standard SQL editor toolbar with various icons for file operations, database management, and query execution.
- Left Panel:** Shows a tree view of databases, tables, and other schema elements under the "world" database.
- SQL Editor:** Labeled "SQL editor 2". It contains the following SQL query:

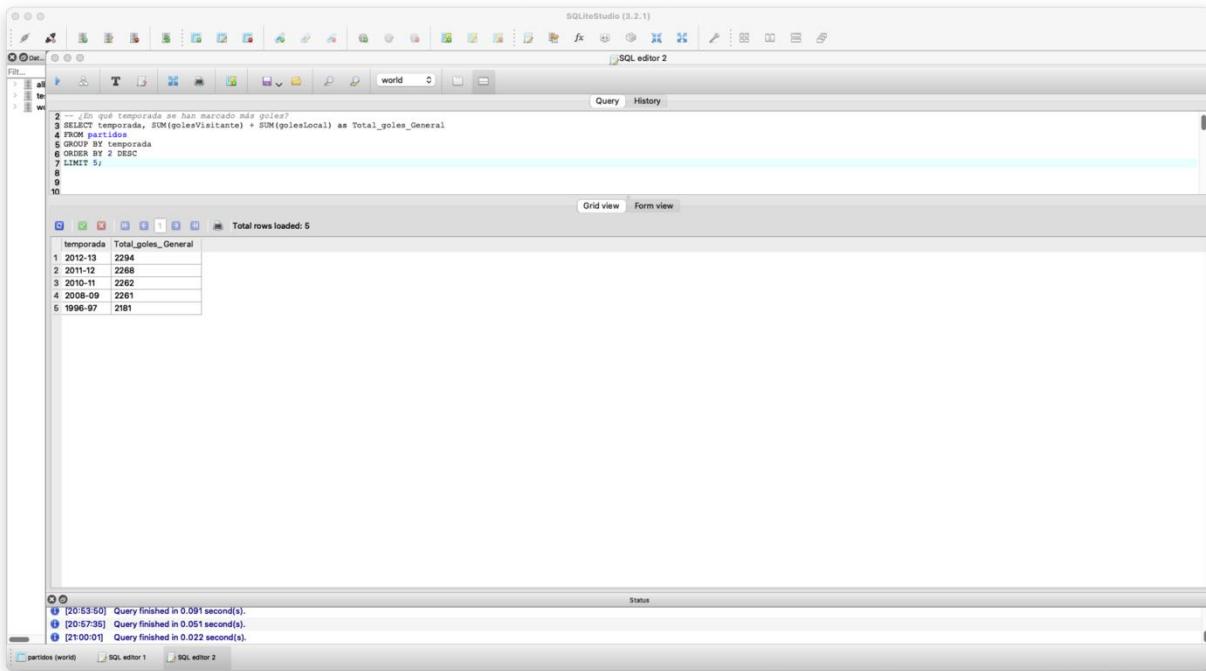
```
1 -- ¿Cuántos partidos han terminado 0-0?
2 SELECT COUNT(*)
3 FROM partidos
4 WHERE golesLocal = 0 AND golesVisitante = 0;
```
- Results View:** Shows the output of the query. It displays "Total rows loaded: 1" and a single row with the value "1 3511".
- Status Bar:** Shows the status message: "[20:23:24] Query finished in 0.034 second(s). [20:24:12] Query finished in 0.035 second(s). [20:26:15] Query finished in 0.036 second(s.)."
- Bottom Navigation:** Shows tabs for "partidos (world)", "SQL editor 1", and "SQL editor 2".

Query:

```
SELECT COUNT(*)
FROM partidos
WHERE golesLocal = 0 AND golesVisitante = 0
```

3. ¿En qué temporada se han marcado más goles?

Pregunta ambigua, se toman goles locales y visitantes. En la temporada de 2012-13 con 2294 goles respectivamente.



The screenshot shows the SQLiteStudio interface with the following details:

- Toolbar:** Standard database management toolbar.
- Menu Bar:** File, Edit, View, Tools, Database, Query, History, Help.
- SQL Editor:** Labeled "SQL editor 2". Contains the following SQL query:

```
1 /*En qué temporada se han marcado más goles?*/
2 SELECT temporada, SUM(golesVisitante) + SUM(golesLocal) as Total_goles_General
3 FROM partidos
4 GROUP BY temporada
5 ORDER BY 2 DESC
6 LIMIT 5;
```
- Results Grid:** Shows the results of the query in a grid format. The columns are "temporada" and "Total_goles_General". The data is as follows:

temporada	Total_goles_General
1 2012-13	2294
2 2011-12	2268
3 2010-11	2262
4 2008-09	2261
5 1996-97	2181

- Status Bar:** Shows three log entries: "[20:53:50] Query finished in 0.091 second(s).", "[20:57:35] Query finished in 0.051 second(s).", and "[21:00:01] Query finished in 0.022 second(s.)."
- Bottom Navigation:** Shows tabs for "partidos (world)", "SQL editor 1", and "SQL editor 2".

Query:

```
SELECT temporada, SUM(golesVisitante) + SUM(golesLocal) as Total_goles_General
FROM partidos
GROUP BY temporada
ORDER BY 2 DESC
LIMIT 5;
```

4. ¿Cuál es el equipo que tiene el record de meter más goles como local? ¿Y cómo visitante?

A nivel local es el Real Madrid con 2054 goles.

The screenshot shows the SQLiteStudio interface with the following details:

- SQL Editor 2:** Contains the SQL query:

```
-- ¿Cuál es el equipo que tiene el record de meter más goles como local?
SELECT EquipoLocal, SUM(golesLocal) AS Total_goles_local
FROM partidos
GROUP BY EquipoLocal
ORDER BY 2 DESC;
```
- Table View:** Shows the results of the query in a grid format. The columns are "EquipoLocal" and "Total_goles_local". The data is as follows:

EquipoLocal	Total_goles_local
Real Madrid	2054
Barcelona	2043
Atlético de Madrid	1566
Valencia	1522
Athletic Club	1424
Real Zaragoza	1395
Sevilla	1384
RCD Sociedad	1366
Deportivo	1343
Betis	1328
Sporting de Gijón	1275
Espanyol	1272
Celta de Vigo	1252
Valladolid	1208
Rayo Vallecano	1203
Real Madrid C.F.	1142
Las Palmas	1141
Mallorca	1129
Osasuna	1098
Tenerife	1081
Elche	1060
Hércules	936
Murcia	927
Albacete de Huelva	880
Salamanca	860
Real Oviedo	857
Cádiz	815

- Status Bar:** Displays three error messages:
 - [2149-31] Error while executing SQL query on database 'world': misuse of aggregate function SUM()
 - [2149-46] Query finished in 0.066 second(s).
 - [2150-13] Query finished in 0.026 second(s).

Query:

```
SELECT EquipoLocal, SUM(golesLocal) AS Total_goles_local
FROM partidos
GROUP BY EquipoLocal
ORDER BY 2 DESC;
```

A nivel de visitante hay un empate, ya que tanto el Barcelona como el Real Madrid han obtenido la misma cantidad de goles como visitantes, 1296.

The screenshot shows the SQLiteStudio interface with a query window titled "SQL editor 2". The query is:

```
32 iY cdm.visitante?
33 SELECT EquipoVisitante, SUM(golesVisitante) AS Total_goles_Visitante
34 FROM partidos
35 GROUP BY EquipoVisitante
36 ORDER BY 2 DESC;
37
38
39
40
```

The results table has two columns: "EquipoVisitante" and "Total_goles_Visitante". The data is:

EquipoVisitante	Total_goles_Visitante
1 Barcelona	1296
2 Real Madrid	1296
3 Atlético de Madrid	988
4 Valencia	919
5 Betis	853
6 Sevilla	839
7 Real Sociedad	818
8 Real Zaragoza	807
9 Athletic Club	802
10 Celta de Vigo	775
11 Valladolid	775
12 Sporting de Gijón	767
13 Deportivo	765
14 Espanyol	726
15 Rayo Vallecano	711
16 Las Palmas	700
17 Mallorca	692
18 Racing de Santander	688
19 Elche	669
20 Tenerife	660
21 Osasuna	675
22 Recreativo de Huelva	661
23 Murcia	653
24 Hercules	632
25 Salamanca	510
26 Villarreal	502
27 Real Oviedo	490

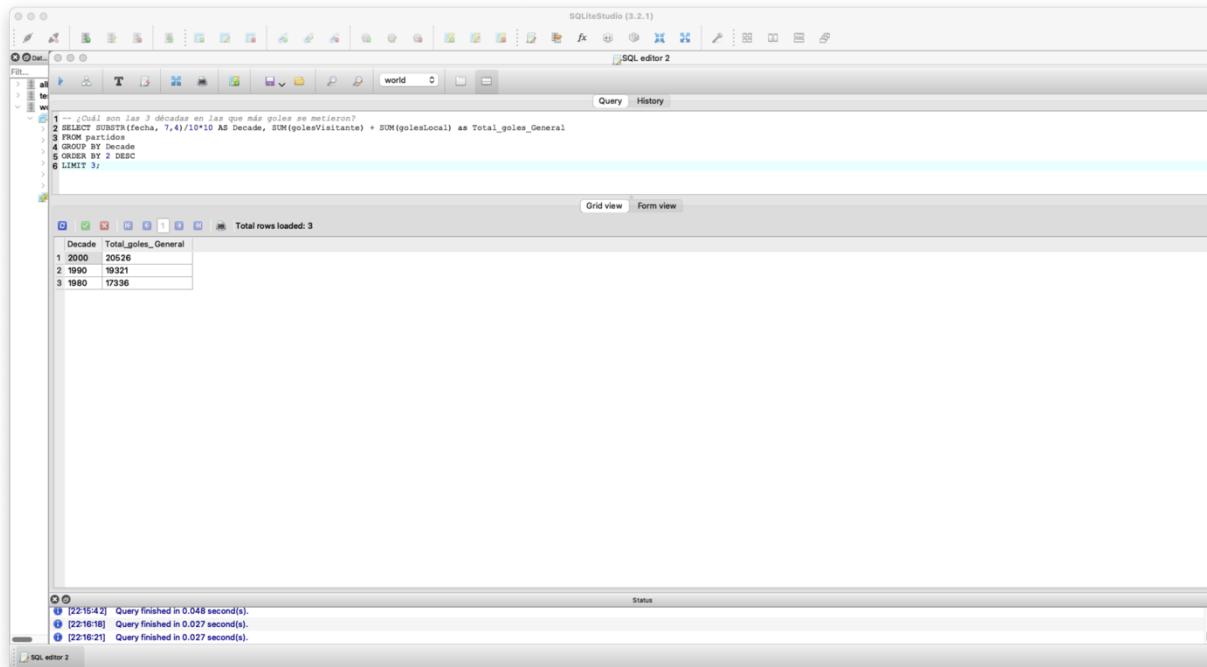
The status bar at the bottom shows three completed queries with execution times: "[2149-46] Query finished in 0.066 second(s).", "[2150-13] Query finished in 0.026 second(s).", and "[2152-04] Query finished in 0.044 second(s).".

Query:

```
SELECT EquipoVisitante, SUM(golesVisitante) AS Total_goles_Visitante
FROM partidos
GROUP BY EquipoVisitante
ORDER BY 2 DESC;
```

5. ¿Cuál son las 3 décadas en las que más goles se metieron?

Pregunta ambigua, se toman goles locales y de visitante como un todo. Corresponden a la década del 2000, 1990 y 1980 respectivamente.



The screenshot shows the SQLiteStudio interface with the following details:

- SQL Editor:** Contains the SQL query:

```
-- ¿Cuál son las 3 décadas en las que más goles se metieron?
SELECT SUBSTR(fecha, 7,4)/10*10 AS Decade, SUM(golesVisitante) + SUM(golesLocal) as Total_goles_General
FROM partidos
GROUP BY Decade
ORDER BY 2 DESC
LIMIT 3;
```
- Results Grid:** Displays the query results in a grid view.

Decade	Total_goles_General
1 2000	20526
2 1990	19321
3 1980	17336

Total rows loaded: 3
- Status Bar:** Shows three log entries indicating query completion times: [22:15:42] Query finished in 0.048 second(s), [22:16:18] Query finished in 0.027 second(s), and [22:16:21] Query finished in 0.027 second(s).

Query:

```
SELECT SUBSTR(fecha, 7,4)/10*10 AS Decade, SUM(golesVisitante) + SUM(golesLocal) as Total_goles_General
FROM partidos
GROUP BY Decade
ORDER BY 2 DESC
LIMIT 3;
```

6. ¿Qué equipo es el mejor local en los últimos 5 años?

Basado en lo que el profesor indicó, se toman los puntos como local de la siguiente forma: 3 si gana y 1 si empata, 0 si pierde y, además, la base de datos llegaba hasta el año 2015, por ende, se toman en cuenta los años desde el 2010-2015, dando como resultado el Barcelona.

The screenshot shows the SQLiteStudio interface with the following details:

- SQL Editor:** Contains the following SQL query:

```
1 -- ¿Qué equipo es el mejor local en los últimos 5 años?
2 SELECT EquipoLocal,
3       SUM(CASE
4           WHEN golesLocal > golesVisitante THEN 3
5           WHEN golesLocal = golesVisitante THEN 1
6           ELSE 0
7       END) AS Puntos_acumulados
8 FROM partidos
9 WHERE SUBSTR(fecha, 7,4) IN ('2010', '2011', '2012', '2013', '2014', '2015')
10 GROUP BY EquipoLocal
11 ORDER BY 2 DESC;
12
13
14
```
- Results View:** Displays a table titled "Total rows loaded: 57" with columns "EquipoLocal" and "Puntos_acumulados". The data is as follows:

EquipoLocal	Puntos_acumulados
1 Barcelona	286
2 Real Madrid	277
3 Atlético de Madrid	236
4 Valencia	228
5 Las Palmas	209
6 Villarreal	206
7 Girona	205
8 Sevilla	204
9 Real Sociedad	201
10 Numancia	198
11 Valladolid	198
12 Alcorcón	197
13 Betis	195
14 Elche	189
15 Deportivo de Gijón	189
16 Atlético Club	188
17 Córdoba	182
18 Celta de Vigo	179
19 Deportivo	177
20 Levante	174

- Status Bar:** Shows three log entries:
 - [23:19:51] Query finished in 0.027 second(s).
 - [23:27:45] Query finished in 0.049 second(s).
 - [23:43:06] Query finished in 0.039 second(s).

Query:

```
SELECT EquipoLocal,
       SUM(CASE
           WHEN golesLocal > golesVisitante THEN 3
           WHEN golesLocal = golesVisitante THEN 1
           ELSE 0
       END) AS Puntos_acumulados
  FROM partidos
 WHERE SUBSTR(fecha, 7,4) IN ('2010', '2011', '2012', '2013', '2014', '2015')
 GROUP BY EquipoLocal
 ORDER BY 2 DESC;
```

7. ¿Cuál es la media de victorias por temporadas en los equipos que han estado menos de 10 temporadas en 1^a división?

Esta pregunta lo que requiere es tener dos columnas una con equipos y la otra con la media de victorias siempre y cuando hayan jugado menos de 10 temporadas, así:

Primero se crea una vista que sea de Equipos:

The screenshot shows the MySQL Workbench interface with the SQL editor tab active. The code in the editor is as follows:

```
1 ----- VISTA DE EQUIPOS ---
2 CREATE VIEW Equipos
3 AS
4 SELECT DISTINCT(EquipoLocal) AS EQUIPOS
5 FROM Partido
6 ORDER BY 1 ASC;
7
8 -----
9
10
11 CREATE VIEW Ganes_Visitantes
12 AS
13 SELECT Temporada, EquipoVisitante,
14 SUM(CASE WHEN golesLocal < golesVisitante THEN 1 ELSE 0 END) AS Cantidad_goles_visitante,
15 COUNT(Temporada) AS Cantidad_partidos_visitante
16 FROM Partido AS p
17 INNER JOIN Equipos AS e
18 ON p.EquipoVisitante = e.Equipos
19 WHERE EquipoVisitante = E.Equipos
20 GROUP BY EquipoVisitante;
21
22
```

Below the editor, a results grid titled "Form view" displays the data from the "Equipos" view. The columns are labeled "EQUIPOS" and show the following list of teams:

EQUIPOS
1 AD Almeria
2 Alaves
3 Albacete
4 Alcorcon
5 Alcoyano
6 Algeciras
7 Alicante
8 Almeria
9 Alzira
10 Aragon
11 Athletic Club

The status bar at the bottom shows three log entries:

- [17:23:31] Error while executing SQL query on database 'world': table Equipos already exists
- [17:24:11] Query finished in 0.001 second(s).
- [17:24:16] Query finished in 0.045 second(s).

Query:

```
CREATE VIEW Equipos
AS
SELECT DISTINCT(EquipoLocal) AS EQUIPOS
FROM Partido
ORDER BY 1 ASC;
```

Luego se crea una vista con la información de los partidos de visitantes y la cantidad de partidos de visitantes:

The screenshot shows the SQLiteStudio interface with the SQL editor tab active. The query window contains the following SQL code:

```
5 FROM partido
6 ORDER BY 1 ASC;
7
8 -----
9
10
11 CREATE VIEW Ganes_Visitantes
12 AS
13 SELECT temporada, EquipoVisitante,
14 SUM(CASE WHEN golesLocal < golesVisitante THEN 1 ELSE 0 END) AS Cantidad_goles_visitante,
15 COUNT(temporada) AS Cantidad_partidos_visitante
16 FROM partido AS p
17 INNER JOIN Equipos AS e
18 ON p.EquipoVisitante = E.Equipos
19 WHERE EquipoVisitante = E.Equipos
20 GROUP BY EquipoVisitante;
21
22 -----
23
24 CREATE VIEW Ganes_Locales
25 AS
26 SELECT temporada, EquipoLocal,
```

The results pane displays a table titled "Form view" with the following data:

	temporada	EquipoVisitante	Cantidad_goles_visitante	Cantidad_partidos_visitante
1	1981-82	AD Almeria	7	72
2	2014-2015	Alaves	101	488
3	2014-2015	Albacete	101	461
4	2014-2015	Alcorcon	29	100
5	2011-12	Alcoyano	3	21
6	2003-04	Algeciras	5	78
7	2008-09	Alicante	6	21
8	2014-2015	Almeria	69	298
9	1988-89	Alzira	2	19
10	1985-86	Aragon	1	19
11	2014-2015	Athletic Club	181	827

The status bar at the bottom shows three log messages:

- [17:25:12] Query finished in 0.003 second(s).
- [17:25:17] SQLiteStudio was unable to extract metadata from the query. Results won't be editable.
- [17:25:17] Query finished in 0.081 second(s).

Query:

```
CREATE VIEW Ganes_Visitantes
AS
SELECT temporada, EquipoVisitante,
SUM(CASE WHEN golesLocal < golesVisitante THEN 1 ELSE 0 END) AS Cantidad_goles_visitante,
COUNT(temporada) AS Cantidad_partidos_visitante
FROM partido AS p
INNER JOIN Equipos AS e
ON p.EquipoVisitante = E.Equipos
WHERE EquipoVisitante = E.Equipos
GROUP BY EquipoVisitante;
```

Enseguida se crea otra vista pero para Ganes de locales, con la cantidad de partidos locales

The screenshot shows the SQLiteStudio interface with the SQL editor tab active. The query window contains the following SQL code:

```
19 WHERE EquipoVisitante = E.Equipos
20 GROUP BY EquipoVisitante;
21
22 -----
23
24 CREATE VIEW Ganes_Locales
25 AS
26 SELECT temporada, EquipoLocal,
27 SUM(CASE WHEN golesVisitante > golesLocal THEN 1 ELSE 0 END) AS Cantidad_goles_local,
28 COUNT(temporada) AS Cantidad_partidos_locales
29 FROM partido AS p
30 INNER JOIN Equipos AS e
31 ON p.EquipoLocal = e.EQUIPOS
32 WHERE EquipoLocal = E.Equipos
33 GROUP BY EquipoLocal;
34
35
36 -----
37 CREATE VIEW Cotejo
38 AS
39 SELECT CAST((v.Cantidad_goles_visitante + l.Cantidad_goles_local) AS FLOAT)/CAST((v.Cantidad_partidos_visitante + l.Cantidad_partidos_locales) AS
40 FROM partido AS p
```

The results pane shows a table titled "Form view" with the following data:

	temporada	EquipoLocal	Cantidad_goles_local	Cantidad_partidos_locales
1	1981-82	AD Almeria	37	72
2	2014-2015	Alaves	244	487
3	2014-2015	Albacete	201	462
4	2014-2015	Alcorcon	57	103
5	2011-12	Alcoyano	6	21
6	2003-04	Algeciras	31	78
7	2008-09	Alicante	2	21
8	2014-2015	Almeria	132	297
9	1988-89	Alzira	7	19
10	1985-86	Aragon	8	19
11	2014-2015	Athletic Club	477	826

The status bar at the bottom shows three messages:

- [17:25:49] Query finished in 0.001 second(s).
- [17:25:53] SQLiteStudio was unable to extract metadata from the query. Results won't be editable.
- [17:25:53] Query finished in 0.073 second(s).

Query:

```
CREATE VIEW Ganes_Locales
AS
SELECT temporada, EquipoLocal,
SUM(CASE WHEN golesLocal > golesVisitante THEN 1 ELSE 0 END) AS Cantidad_goles_local,
COUNT(temporada) AS Cantidad_partidos_locales
FROM partido AS p
INNER JOIN Equipos AS e
ON p.EquipoLocal = e.EQUIPOS
WHERE EquipoLocal = E.Equipos
GROUP BY EquipoLocal;
```

Seguidamente se crea otra vista llamada cotejo en donde se obtiene la media por cada equipo

The screenshot shows the SQLiteStudio interface with the SQL editor tab open. The code in the editor is:

```
34
35
36 -----
37 CREATE VIEW Cotejo
38 AS
39 SELECT CAST((v.Cantidad_goles_visitante + l.Cantidad_goles_local) AS FLOAT)/CAST((v.Cantidad_partidos_visitante + l.Cantidad_partidos_loca) AS
40 FROM partido AS p
41 INNER JOIN Equipos AS e
42 ON p.EquipoLocal = e.EQUIPOS
43 INNER JOIN Ganes_Visitantes AS v
44 ON v.EquipoVisitante = e.EQUIPOS
45 INNER JOIN Ganes_Locales AS l
46 ON l.EquipoLocal = e.EQUIPOS
47 WHERE l.EquipoLocal = E.Equipos
48 GROUP BY E.Equipos;
49
50
51 -----
52
53
54 CREATE VIEW
55 Temporadas
```

The results pane shows a table titled "Form view" with the following data:

Promedio_general	EQUIPOS
0.3055555555555556	AD Almeria
0.35384615384615	Alaves
0.32719393282774	Albacete
0.42364532019704	Alcorcon
0.21428571428571	Alcayano
0.23076923076923	Algeciras
0.19047619047619	Alicante
0.33781512605042	Almeria
0.23684210526316	Alzira
0.23684210526316	Aragon
0.39806412583182	Athletic Club

The status bar at the bottom shows three messages:

- [17:26:18] Query finished in 0.002 second(s).
- [17:26:22] SQLiteStudio was unable to extract metadata from the query. Results won't be editable.
- [17:26:22] Query finished in 0.222 second(s).

Query:

```
CREATE VIEW Cotejo
AS
SELECT CAST((v.Cantidad_goles_visitante + l.Cantidad_goles_local) AS
FLOAT)/CAST((v.Cantidad_partidos_visitante + l.Cantidad_partidos_loca) AS FLOAT) AS
Promedio_general, E.EQUIPOS
FROM partido AS p
INNER JOIN Equipos AS e
ON p.EquipoLocal = e.EQUIPOS
INNER JOIN Ganes_Visitantes AS v
ON v.EquipoVisitante = e.EQUIPOS
INNER JOIN Ganes_Locales AS l
ON l.EquipoLocal = e.EQUIPOS
WHERE l.EquipoLocal = E.Equipos
GROUP BY E.Equipos;
```

Después se crea otra vista con las temporadas y la cantidad de veces que un equipo ha estado en cada una de ellas:

The screenshot shows the MySQL Workbench interface with the SQL editor tab active. The query window contains the following SQL code:

```
51 -----
52
53
54 CREATE VIEW
55 Temporadas
56 AS
57 SELECT DISTINCT(EquipoLocal) AS Equipos, COUNT(DISTINCT(temporada)) AS temporadas, temporada
58 FROM partido
59 GROUP BY Equipos;
60
61 -----
62
63 SELECT t.temporada, AVG(c.Promedio_general)AS Promedio_General
64 FROM Cotejo AS c
65 INNER JOIN Temporadas as t
66 ON c.Equipos = t.Equipos
67 WHERE t.temporadas < 10
68 GROUP BY t.temporada
69 ORDER BY t.temporada DESC;
70
71
```

The results pane displays a table titled "Form view" with the following data:

	Equipos	temporadas	temporada
1	AD Almeria	4	1981-82
2	Alaves	25	2014-2015
3	Albacete	23	2014-2015
4	Alcorcon	5	2014-2015
5	Alcoyano	1	2011-12
6	Algeciras	4	2003-04
7	Alicante	1	2008-09
8	Almeria	15	2014-2015
9	Alzira	1	1988-89
10	Aragon	1	1985-86
11	Athletic Club	45	2014-2015

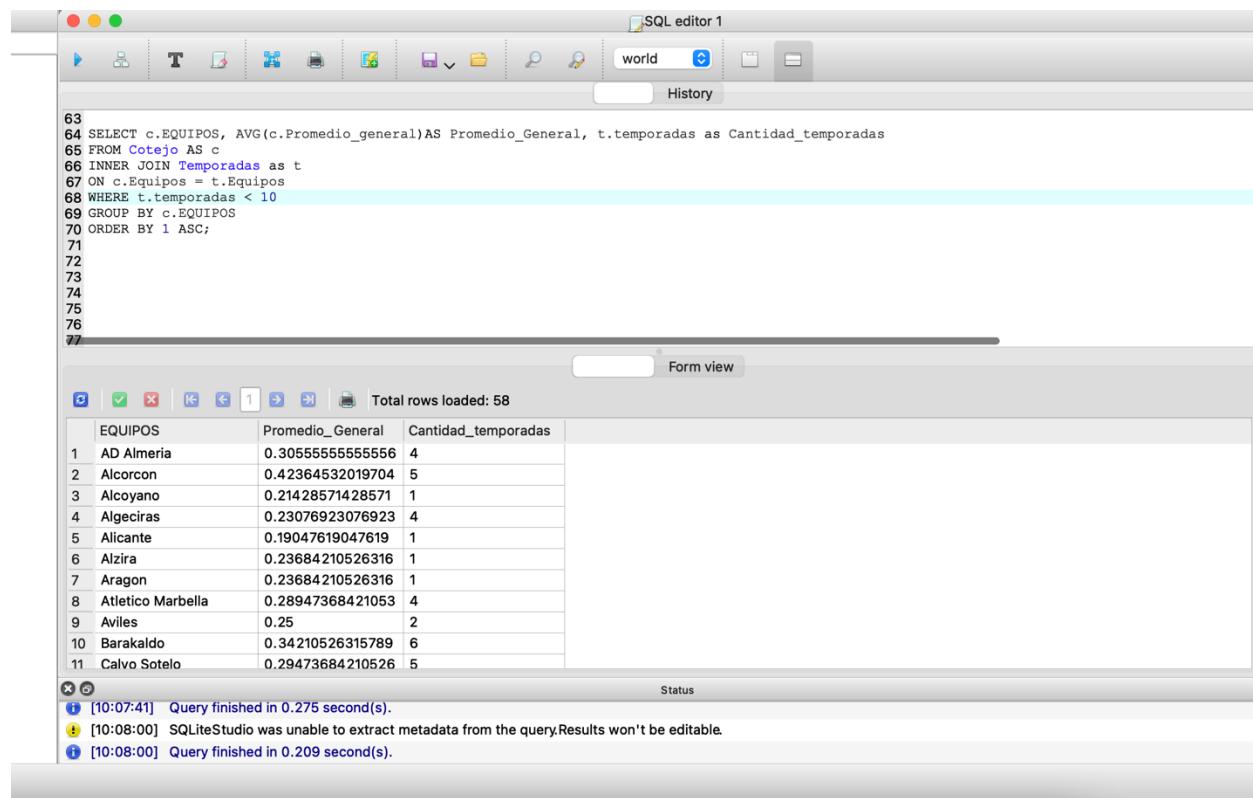
The status bar at the bottom shows three log entries:

- [17:26:57] Error while executing SQL query on database 'world': near "Temporadas": syntax error
- [17:27:04] Error while executing SQL query on database 'world': table Temporadas already exists
- [17:27:09] Query finished in 0.033 second(s).

Query:

```
CREATE VIEW
Temporadas
AS
SELECT DISTINCT(EquipoLocal) AS Equipos, COUNT(DISTINCT(temporada)) AS temporadas,
temporada
FROM Partido
GROUP BY Equipos;
```

Finalmente se ejecuta el siguiente query que obtendrá la media de los equipos que han estado en menos de 10 temporadas en primera division por cada una de los equipos y se adiciona una tercera columna de cotejo para saber que efectivamente, todos ellos han estado menos de 10 temporadas.



```

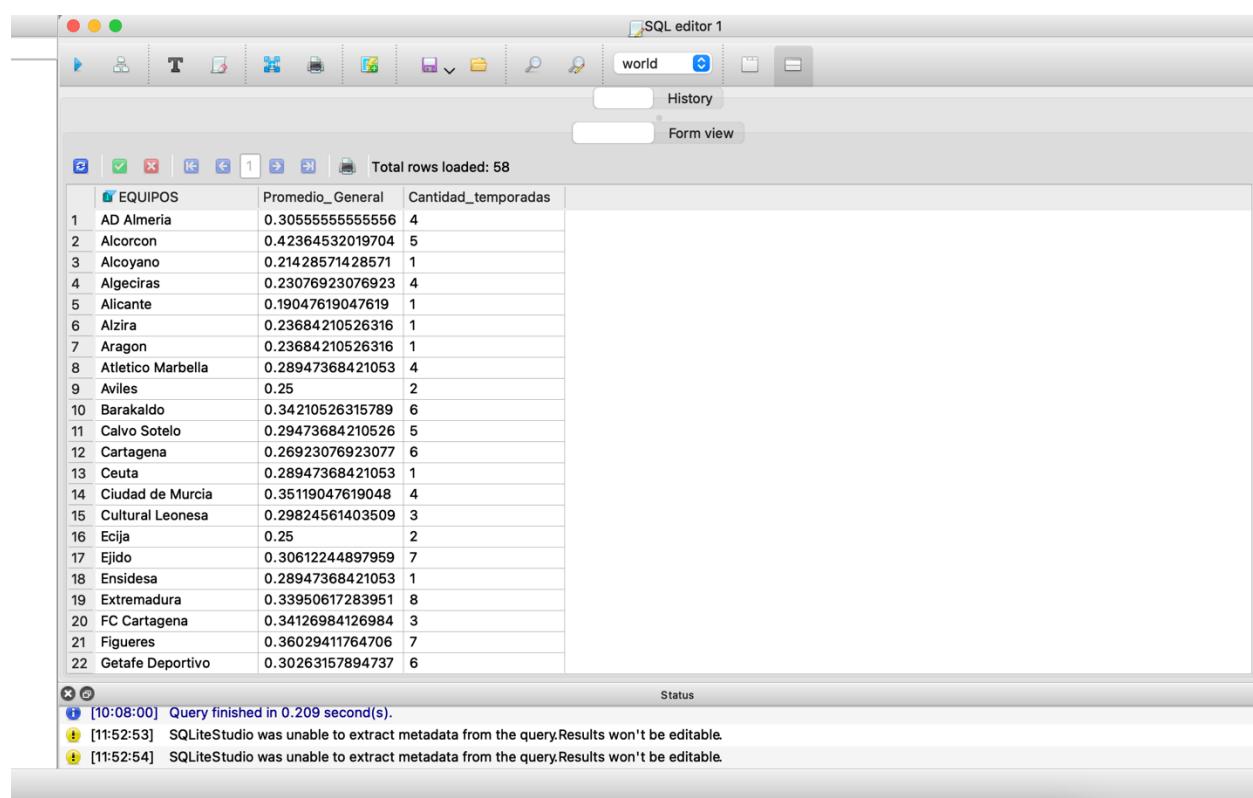
63
64 SELECT c.EQUIPOS, AVG(c.Promedio_general)AS Promedio_General, t.temporadas as Cantidad_temporadas
65 FROM Cotejo AS c
66 INNER JOIN Temporadas as t
67 ON c.Equipos = t.Equipos
68 WHERE t.temporadas < 10
69 GROUP BY c.EQUIPOS
70 ORDER BY 1 ASC;
71
72
73
74
75
76
77

```

Total rows loaded: 58

	EQUIPOS	Promedio_General	Cantidad_temporadas
1	AD Almería	0.305555555555556	4
2	Alcorcón	0.42364532019704	5
3	Alcoyano	0.21428571428571	1
4	Algeciras	0.23076923076923	4
5	Alicante	0.19047619047619	1
6	Alzira	0.23684210526316	1
7	Aragón	0.23684210526316	1
8	Atlético Marbella	0.28947368421053	4
9	Avilés	0.25	2
10	Barakaldo	0.34210526315789	6
11	Calvo Sotelo	0.29473684210526	5

[10:07:41] Query finished in 0.275 second(s).
[10:08:00] SQLiteStudio was unable to extract metadata from the query. Results won't be editable.
[10:08:00] Query finished in 0.209 second(s).



	EQUIPOS	Promedio_General	Cantidad_temporadas
1	AD Almería	0.305555555555556	4
2	Alcorcón	0.42364532019704	5
3	Alcoyano	0.21428571428571	1
4	Algeciras	0.23076923076923	4
5	Alicante	0.19047619047619	1
6	Alzira	0.23684210526316	1
7	Aragón	0.23684210526316	1
8	Atlético Marbella	0.28947368421053	4
9	Avilés	0.25	2
10	Barakaldo	0.34210526315789	6
11	Calvo Sotelo	0.29473684210526	5
12	Cartagena	0.26923076923077	6
13	Ceuta	0.28947368421053	1
14	Ciudad de Murcia	0.35119047619048	4
15	Cultural Leonesa	0.29824561403509	3
16	Ecija	0.25	2
17	Ejido	0.30612244887959	7
18	Ensidesa	0.28947368421053	1
19	Extremadura	0.33950617283951	8
20	FC Cartagena	0.34126984126984	3
21	Figuères	0.36029411764706	7
22	Getafe Deportivo	0.30263157894737	6

[10:08:00] Query finished in 0.209 second(s).
[11:52:53] SQLiteStudio was unable to extract metadata from the query. Results won't be editable.
[11:52:54] SQLiteStudio was unable to extract metadata from the query. Results won't be editable.

SQL editor 1

world

Total rows loaded: 58

History

Form view

EQUIPOS Promedio_General Cantidad_temporadas

23 Girona	0.3588850174216	7
24 Granada 74	0.23809523809524	1
25 Guadalajara	0.30952380952381	2
26 Huesca	0.3	5
27 Jaen	0.29432624113475	7
28 Langreo	0.28947368421053	2
29 Linares CF	0.27894736842105	5
30 Llagostera	0.34285714285714	1
31 Lorca	0.26315789473684	1
32 Lorca Deportiva	0.33333333333333	2
33 Lugo	0.29936305732484	4
34 Malaga B	0.24603174603175	3
35 Mallorca B	0.28571428571429	1
36 Merida	0.38135593220339	9
37 Mestalla	0.22368421052632	2
38 Mirandes	0.31932773109244	3
39 Mollerussa	0.07894736842105	1
40 Moscardo	0.15789473684211	1
41 Ontinyent	0.21052631578947	1
42 Orihuela	0.31578947368421	1
43 Ourense	0.23728813559322	6
44 Palamos	0.28947368421053	6

Status

[10:08:00] Query finished in 0.209 second(s).

[11:52:53] SQLiteStudio was unable to extract metadata from the query. Results won't be editable.

[11:52:54] SQLiteStudio was unable to extract metadata from the query. Results won't be editable.

SQL editor 1

world

Total rows loaded: 58

History

Form view

EQUIPOS Promedio_General Cantidad_temporadas

37 Mestalla	0.22368421052632	2
38 Mirandes	0.31932773109244	3
39 Mollerussa	0.07894736842105	1
40 Moscardo	0.15789473684211	1
41 Ontinyent	0.21052631578947	1
42 Orihuela	0.31578947368421	1
43 Ourense	0.23728813559322	6
44 Palamos	0.28947368421053	6
45 Palencia CF	0.31578947368421	4
46 Ponferradina	0.30049261083744	5
47 Pontevedra	0.29896907216495	5
48 Racing Ferrol	0.27322404371585	9
49 Real Burgos	0.2781954887218	7
50 Real Union	0.28571428571429	1
51 Sant Andreu	0.35714285714286	7
52 Sestao	0.32758620689655	9
53 Sevilla Atletico	0.19047619047619	2
54 Terrassa	0.29496402877698	7
55 Toledo	0.36330935251799	7
56 Universidad Las Palmas	0.19047619047619	1
57 Vecindario	0.21428571428571	1
58 Villarreal B	0.35714285714286	3

Status

[10:08:00] Query finished in 0.209 second(s).

[11:52:53] SQLiteStudio was unable to extract metadata from the query. Results won't be editable.

[11:52:54] SQLiteStudio was unable to extract metadata from the query. Results won't be editable.

Query:

```
SELECT c.EQUIPOS, AVG(c.Promedio_general)AS Promedio_General, t.temporadas as  
Cantidad_temporadas  
FROM Cotejo AS c  
INNER JOIN Temporadas as t  
ON c.Equipos = t.Equipos  
WHERE t.temporadas < 10  
GROUP BY c.EQUIPOS  
ORDER BY 1 ASC;
```

Dicho resultado se verificó a mano con los equipos de: “Alicante”, “Guadalajara”, “Terrassa”, “Aviles” y sí se obtiene el resultado correcto.

8. ¿Quién ha estado más temporadas en 1^a División: Barcelona o Real Madrid?

Hay un empate, ya que ambos han estado en 45 temporadas. Se opta por hacerlo de dos formas:

La primera:

The screenshot shows the SQLiteStudio interface with a SQL editor window containing the following query:

```
1 -- ¿Quién ha estado más temporadas en 1ª División: Barcelona o Real Madrid?
2 SELECT COUNT(DISTINCT(temporada)) AS Cantidad_temporadas, EquipoLocal, EquipoVisitante
3 FROM partidos
4 WHERE EquipoVisitante IN ('Real Madrid', 'Barcelona') AND EquipoLocal IN ('Real Madrid', 'Barcelona')
5 GROUP BY 2,3
6 ORDER BY 1 DESC;
```

The results are displayed in a table:

Cantidad_temporadas	EquipoLocal	EquipoVisitante
1	45	Real Madrid
2	45	Barcelona

The status bar at the bottom shows three completed queries with execution times: [00:24:26], [00:25:05], and [00:26:14].

Query:

```
SELECT COUNT(DISTINCT(temporada)) AS Cantidad_temporadas, EquipoLocal, EquipoVisitante
FROM partidos
WHERE EquipoVisitante IN ('Real Madrid', 'Barcelona') AND EquipoLocal IN ('Real Madrid', 'Barcelona')
GROUP BY 2,3
ORDER BY 1 DESC;
```

Forma alternativa:

The screenshot shows a MySQL Workbench interface. The top window is titled "SQL editor 1" and contains the following SQL code:

```
34  
35  
36  
37  
38  
39  
40  
41  
42 |  
43  
44  
45  
46  
47  
48 SELECT COUNT(DISTINCT(temporada)) AS Cantidad_temporadas, EquipoVisitante  
49 FROM Partido  
50 WHERE EquipoVisitante IN ('Real Madrid')  
51 ORDER BY 1 DESC;  
52  
53  
54
```

The results are displayed in a table below the editor:

Cantidad_temporadas	EquipoVisitante
1	45
	Real Madrid

The status bar at the bottom shows three log entries:

- [0:58:20] Query finished in 0.007 second(s).
- [0:59:00] Query finished in 0.005 second(s).
- [0:59:40] Query finished in 0.005 second(s).

Query:

```
SELECT COUNT(DISTINCT(temporada)) AS Cantidad_temporadas, EquipoVisitante
FROM Partido
WHERE EquipoVisitante IN ('Real Madrid')
ORDER BY 1 DESC;
```

The screenshot shows a Mac OS X desktop environment. At the top is a window titled "SQL editor 1" containing a SQL query and its results. Below the window is a system status bar displaying the date and time, battery level, signal strength, and other system icons.

```
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49 SELECT COUNT(DISTINCT(temporada)) AS Cantidad_temporadas, EquipoLocal
50 FROM partido
51 WHERE EquipoLocal IN ('Real Madrid')
52 ORDER BY 1 DESC;
53
54
```

Cantidad_temporadas	EquipoLocal
1	45
	Real Madrid

Total rows loaded: 1

Status:

- [01:57:36] Query finished in 0.013 second(s).
- [01:58:20] Query finished in 0.007 second(s).
- [01:59:00] Query finished in 0.006 second(s).

Query:

```
SELECT COUNT(DISTINCT(temporada)) AS Cantidad_temporadas, EquipoLocal
FROM partido
WHERE EquipoLocal IN ('Real Madrid')
ORDER BY 1 DESC;
```

The screenshot shows the MySQL Workbench interface. The SQL editor window contains the following query:

```
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49 SELECT COUNT(DISTINCT(temporada)) AS Cantidad_temporadas, EquipoVisitante  
50 FROM partido  
51 WHERE EquipoVisitante IN ('Barcelona')  
52 ORDER BY 1 DESC;  
53  
54  
55  
56
```

The results pane shows a single row of data:

Cantidad_temporadas	EquipoVisitante
1 45	Barcelona

The status bar at the bottom indicates "Total rows loaded: 1". Below the results, the status bar shows three completed queries with their execution times: [0:59:00], [0:59:40], and [0:00:43].

Query:

```
SELECT COUNT(DISTINCT(temporada)) AS Cantidad_temporadas, EquipoVisitante  
FROM partido  
WHERE EquipoVisitante IN ('Barcelona')  
ORDER BY 1 DESC;
```

The screenshot shows the MySQL Workbench interface. The top window is titled "SQL editor 1" and contains the following SQL code:

```
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49 SELECT COUNT(DISTINCT(temporada)) AS Cantidad_temporadas, EquipoLocal  
50 FROM partido  
51 WHERE EquipoLocal IN ('Barcelona')  
52 ORDER BY 1 DESC;  
53  
54  
55  
56
```

The results grid below the code shows one row of data:

Cantidad_temporadas	EquipoLocal
1 45	Barcelona

Total rows loaded: 1

The status bar at the bottom indicates three query executions:

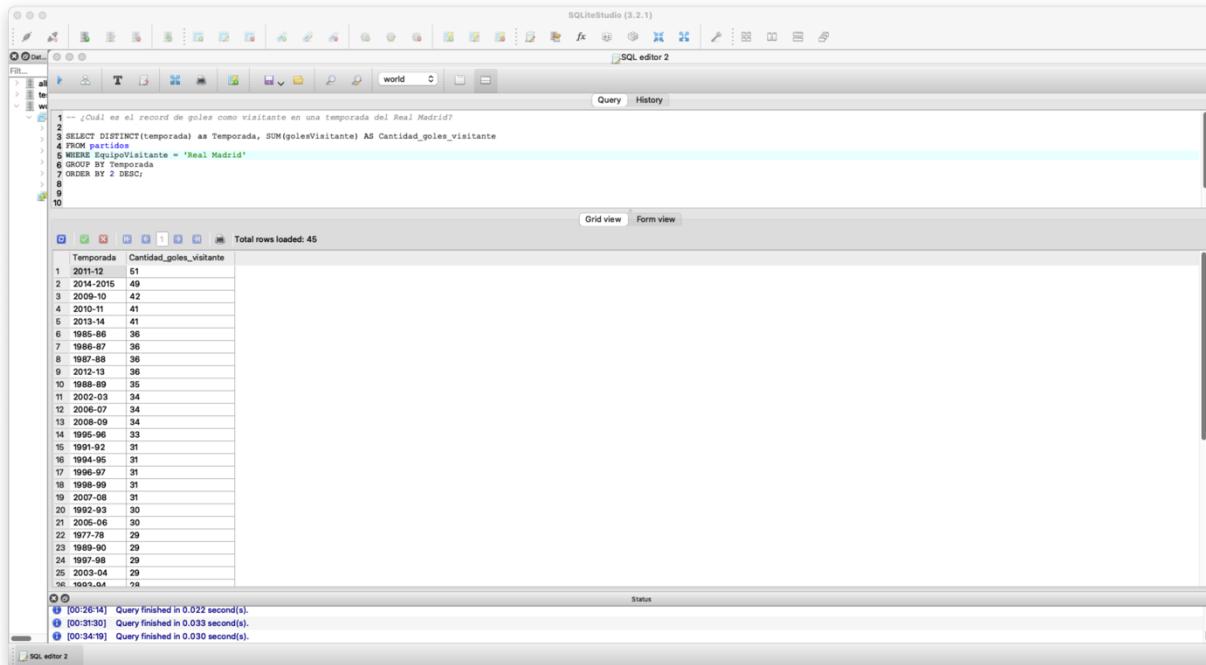
- [01:59:40] Query finished in 0.005 second(s).
- [02:00:43] Query finished in 0.016 second(s).
- [02:01:25] Query finished in 0.006 second(s).

Query:

```
SELECT COUNT(DISTINCT(temporada)) AS Cantidad_temporadas, EquipoLocal  
FROM partido  
WHERE EquipoLocal IN ('Barcelona')  
ORDER BY 1 DESC;
```

9. ¿Cuál es el record de goles como visitante en una temporada del Real Madrid?

Esta pregunta también estaba ambigua, porque no dice en cuál temporada o si se quiere saber en general, así que se hace de forma general y la respuesta es en la temporada 2011-12 con 51 goles de visitante.



The screenshot shows the SQLiteStudio interface with a SQL editor window containing the following query:

```
1 -- ¿Cuál es el record de goles como visitante en una temporada del Real Madrid?
2
3 SELECT DISTINCT(temporada) as Temporada, SUM(golesVisitante) AS Cantidad_goles_visitante
4 FROM partidos
5 WHERE EquipoVisitante = 'Real Madrid'
6 GROUP BY Temporada
7 ORDER BY 2 DESC;
```

The results table shows the following data:

	Temporada	Cantidad_goles_visitante
1	2011-12	51
2	2014-2015	49
3	2009-10	42
4	2013-14	41
5	2010-14	41
6	1985-86	36
7	1986-87	36
8	1987-88	36
9	2012-13	36
10	1988-89	35
11	2002-03	34
12	2006-07	34
13	2007-08	34
14	1990-91	33
15	1991-92	31
16	1994-95	31
17	1996-97	31
18	1998-99	31
19	2007-08	31
20	1992-93	30
21	2005-06	30
22	1977-78	29
23	1987-88	29
24	1987-98	29
25	2003-04	29
26	1993-94	28

The status bar at the bottom shows three completed queries with execution times: [0:26:14], [0:31:30], and [0:34:19].

Query:

```
SELECT DISTINCT(temporada) as Temporada, SUM(golesVisitante) AS Cantidad_goles_visitante
FROM partidos
WHERE EquipoVisitante = 'Real Madrid'
GROUP BY Temporada
ORDER BY 2 DESC;
```

10. ¿En qué temporada se marcaron más goles en Cataluña?

Esta pregunta es ambigua, porque no especifica si goles locales y visitantes, o solo goles en general y si solo los equipos jugando como locales o visitantes o en general, así que se opta por hacerla como goles totales en general de los equipos de Barcelona y Espanyol tal y como el profesor indicó en el foro, por ende se hizo en varios pasos.

1. Se crea una vista con la cantidad de goles locales y se toma en cuenta solo goles locales de Barcelona y Espanyol jugando estos como locales.

The screenshot shows the SQLiteStudio interface. At the top, there is a toolbar with various icons. Below it is a menu bar with 'File', 'Edit', 'View', 'Tools', 'Help'. The main area is titled 'SQL editor 1' and contains the following SQL code:

```
1 CREATE VIEW Cantidad_goles_Locales
2 AS
3 SELECT DISTINCT(temporada) as Temporada, SUM(golesLocal) AS Cantidad_goles_Locales
4 FROM partido
5 WHERE EquipoLocal IN ('Barcelona', 'Espanyol')
6 GROUP BY Temporada
7 ORDER BY 2 DESC;
```

Below the code, a message says 'Total rows loaded: 45'. A table is displayed with two columns: 'Temporada' and 'Cantidad_goles_Locales'. The data is as follows:

Temporada	Cantidad_goles_Locales
1 2011-12	104
2 1976-77	93
3 1996-97	93
4 1986-87	90
5 1988-90	89
6 1993-94	89
7 2008-09	89
8 2012-13	88
9 2013-14	87
10 2014-2015	82
11 1998-99	79
12 2010-11	79
13 1978-79	78
14 2001-02	78

At the bottom, the status bar shows three messages:

- [14:40:08] SQLiteStudio was unable to extract metadata from the query. Results won't be editable.
- [14:40:09] Query finished in 0.016 second(s).
- [14:42:01] Query finished in 0.041 second(s).

Query:

```
CREATE VIEW Cantidad_goles_Locales
AS
SELECT DISTINCT(temporada) as Temporada, SUM(golesLocal) AS Cantidad_goles_Locales
FROM partido
WHERE EquipoLocal IN ('Barcelona', 'Espanyol')
GROUP BY Temporada
ORDER BY 2 DESC;
```

2. Se crea otra vista con la cantidad de goles visitantes y se toma en cuenta solo goles visitante de Barcelona y Espanyol jugando estos como visitantes.

The screenshot shows the MySQL Workbench interface. The SQL editor window contains the following code:

```

25
26 CREATE VIEW Cantidad_goles_visitante
27 AS
28 SELECT DISTINCT(temporada) as Temporada, SUM(golesVisitante) AS Cantidad_goles_visitante
29 FROM partido
30 WHERE EquipoVisitante IN ('Barcelona', 'Espanyol')
31 GROUP BY Temporada
32 ORDER BY 2 DESC;
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50

```

The results grid shows the data loaded from the query:

Temporada	Cantidad_goles_visitante
1 2012-13	70
2 2014-2015	65
3 2008-09	62
4 2010-11	62
5 1993-94	61
6 1996-97	60
7 1995-96	59
8 1998-99	57
9 2006-07	57
10 1991-92	56
11 2011-12	56
12 1997-98	55
13 2003-04	54
14 2013-14	54

The status bar at the bottom indicates the following execution times:

- [14:40:09] Query finished in 0.016 second(s).
- [14:42:01] Query finished in 0.041 second(s).
- [14:42:48] Query finished in 0.038 second(s).

Query:

```

CREATE VIEW Cantidad_goles_visitante
AS
SELECT DISTINCT(temporada) as Temporada, SUM(golesVisitante) AS Cantidad_goles_visitante
FROM partido
WHERE EquipoVisitante IN ('Barcelona', 'Espanyol')
GROUP BY Temporada
ORDER BY 2 DESC;

```

3. Finalmente se hace un join para unir ambas vistas y obtener la suma total de goles por temporada, dando como resultado que fue en la temporada de 2011-12 con 160 goles totales.

The screenshot shows the SQLiteStudio interface. The top window is the SQL editor titled "SQL editor 1" with the query:

```

52 SELECT l.Temporada, (l.Cantidad_goles_Locales + v.Cantidad_goles_visitante) AS Goles_totales_Cataluna
53 FROM Cantidad_goles_Locales AS l
54 INNER JOIN Cantidad_goles_visitante AS v
55 ON l.Temporada = v.Temporada
56 GROUP BY 1
57 ORDER BY 2 DESC;
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77

```

The bottom window is the results grid titled "Form view" with the heading "Total rows loaded: 45". The data is as follows:

Temporada	Goles_totales_Cataluna	
1	2011-12	160
2	2012-13	158
3	1996-97	153
4	2008-09	151
5	1993-94	150
6	2014-2015	147
7	2010-11	141
8	2013-14	141
9	1998-99	136
10	1995-96	135
11	1989-90	133
12	1976-77	130
13	1991-92	130
14	1986-87	129

The status bar at the bottom shows three messages:

- [14:42:48] Query finished in 0.038 second(s).
- [14:43:16] SQLiteStudio was unable to extract metadata from the query. Results won't be editable.
- [14:43:16] Query finished in 0.045 second(s).

Query:

```

SELECT      l.Temporada,      (l.Cantidad_goles_Locales      +      v.Cantidad_goles_visitante)      AS
Goles_totales_Cataluna

FROM Cantidad_goles_Locales AS l

INNER JOIN Cantidad_goles_visitante AS v

ON l.Temporada = v.Temporada

GROUP BY 1

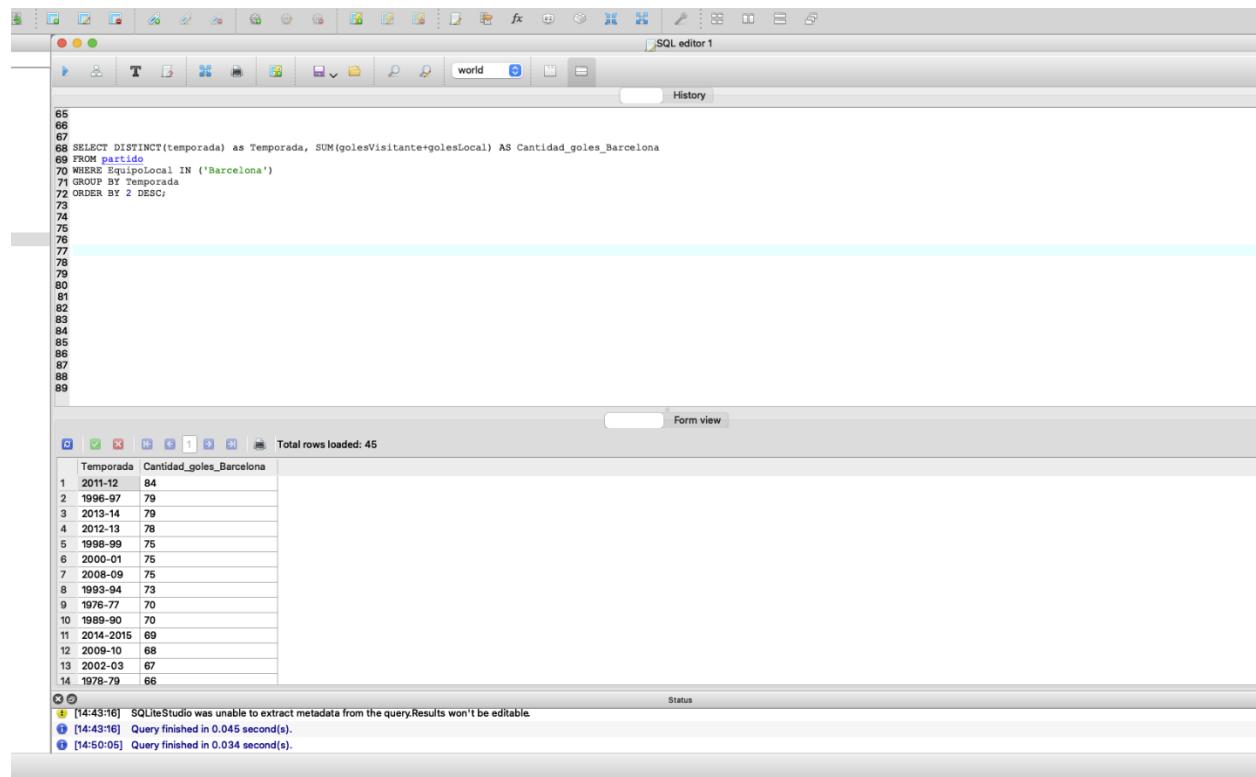
ORDER BY 2 DESC;

```

- Goles marcados y recibidos por el Barcelona jugando de local.

Esta pregunta también es ambigua, porque no hace referencia a ninguna temporada ni si se quiere el total, así que se hace para los dos escenarios.

Si se requiere por la temporada:



The screenshot shows the SQLiteStudio interface. The top part is the SQL editor window titled "SQL editor 1" containing the following SQL code:

```

65
66
67
68 SELECT DISTINCT(temporada) as Temporada, SUM(golesVisitante+golesLocal) AS Cantidad_goles_Barcelona
69 FROM partido
70 WHERE EquipoLocal IN ('Barcelona')
71 GROUP BY Temporada
72 ORDER BY 2 DESC;
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89

```

The bottom part shows the results grid with the following data:

	Temporada	Cantidad_goles_Barcelona
1	2011-12	84
2	1996-97	79
3	2013-14	79
4	2012-13	78
5	1998-99	75
6	2000-01	75
7	2008-09	75
8	1993-94	73
9	1976-77	70
10	1989-90	70
11	2014-2015	69
12	2009-10	68
13	2002-03	67
14	1978-79	66

The status bar at the bottom left shows the message: "SQLiteStudio was unable to extract metadata from the query. Results won't be editable." and the status bar at the bottom right shows the message: "[14:43:16] Query finished in 0.045 second(s).".

Query:

```

SELECT      DISTINCT(temporada)      as      Temporada,      SUM(golesVisitante+golesLocal)      AS
Cantidad_goles_Barcelona

FROM partido

WHERE EquipoLocal IN ('Barcelona')

GROUP BY Temporada

ORDER BY 2 DESC;

```

Si se requiere por la cantidad total:

The screenshot shows a MySQL Workbench interface. The top window is titled "SQL editor 1" and contains the following SQL code:

```
61
62
63
64
65
66
67 SELECT SUM(golesVisitante+golesLocal) AS Cantidad_goles_Barcelona
68 FROM partido
69 WHERE EquipoLocal IN ('Barcelona')
70
71
72
73
74
75
76 |
77
78
79
80
81
82
83
84
85
86
```

The bottom window shows the results of the query:

Cantidad_goles_Barcelona
1 2666

Total rows loaded: 1

In the status bar at the bottom, there are three log entries:

- [14:54:44] Query finished in 0.007 second(s).
- [14:55:32] Query finished in 0.006 second(s).
- [14:57:10] Query finished in 0.005 second(s).

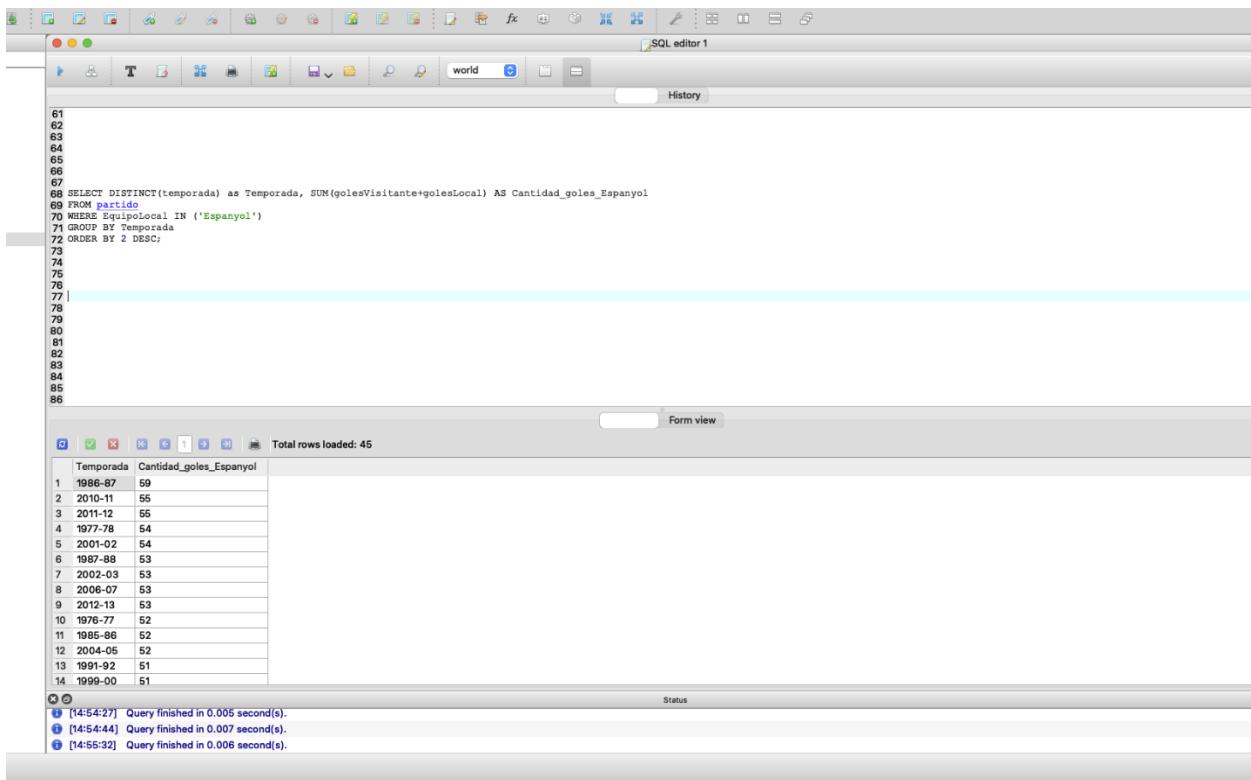
Query:

```
SELECT SUM(golesVisitante+golesLocal) AS Cantidad_goles_Barcelona
FROM partido
WHERE EquipoLocal IN ('Barcelona')
```

- Goles marcados y recibidos por el Espanyol jugando de local.

Esta pregunta también es ambigua, porque no hace referencia a ninguna temporada ni si se quiere el total, así que se hace para los dos escenarios.

Si se requiere por temporada:



The screenshot shows a MySQL Workbench environment. The SQL editor tab is active, displaying the following query:

```

61
62
63
64
65
66
67
68 SELECT DISTINCT(temporada) as Temporada, SUM(golesVisitante+golesLocal) AS Cantidad_goles_Espanyol
69 FROM partido
70 WHERE EquipoLocal IN ('Espanyol')
71 GROUP BY Temporada
72 ORDER BY 2 DESC;
73
74
75
76
77 |
78
79
80
81
82
83
84
85
86

```

The results grid below the editor shows the following data:

Temporada	Cantidad_goles_Espanyol
1986-87	59
2010-11	55
2011-12	55
4 1977-78	54
5 2001-02	54
6 1987-88	53
7 2002-03	53
8 2006-07	53
9 2012-13	53
10 1976-77	52
11 1986-86	52
12 2004-05	52
13 1991-92	51
14 1999-00	51

The status bar at the bottom indicates three completed queries:

- [14:54:27] Query finished in 0.005 second(s).
- [14:54:44] Query finished in 0.007 second(s).
- [14:55:32] Query finished in 0.006 second(s).

Query:

```

SELECT      DISTINCT(temporada)      as      Temporada,      SUM(golesVisitante+golesLocal)      AS
Cantidad_goles_Espanyol
FROM partido
WHERE EquipoLocal IN ('Espanyol')
GROUP BY Temporada
ORDER BY 2 DESC;

```

Si se requiere a nivel general:

The screenshot shows a MySQL Workbench interface. The top window is titled "SQL editor 1" and contains the following SQL code:

```
63
64
65
66
67
68 SELECT SUM(golesVisitante+golesLocal) AS Cantidad_goles_Espanyol
69 FROM partido
70 WHERE EquipoLocal IN ('Espanyol')
71
72
73
74
75
76
77
78 |
79
80
81
82
83
84
85
86
87
```

The bottom window is titled "Form view" and displays the results of the query:

Cantidad_goles_Espanyol
1 2036

Total rows loaded: 1

Status bar at the bottom shows three log entries:

- [14:54:17] Query finished in 0.033 second(s).
- [14:54:27] Query finished in 0.005 second(s).
- [14:54:44] Query finished in 0.007 second(s).

Query:

```
SELECT SUM(golesVisitante+golesLocal) AS Cantidad_goles_Espanyol
FROM partido
WHERE EquipoLocal IN ('Espanyol')
```

----- FIN -----

