ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

GUIA DE AUTOESTUDIO 2

SQL Basico

ALUMNOS:

Ignacio Andrés Castillo Rendon

Anderson Fabian Garcia Nieto

Laboratorio-Modelos de bases de datos
2024-2

DOCENTE:
MARIA IRMA DIAZ ROZO

Bogotá D.C-Colombia

INVESTIGACIÓN

A. NULL

¿Qué significa?

NULL es un término común en programación y bases de datos que, en términos simples, representa la ausencia de valor. Es decir, indica que en un campo específico no se ha asignado ningún dato o que el dato que debería estar allí no existe.

¿Por qué es importante NULL?

Indica falta de información: Cuando un campo es NULL, significa que no se tiene información al respecto. Esto es útil para representar datos faltantes o desconocidos.

Diferencia de "cero" o cadena vacía: NULL no es lo mismo que cero en un campo numérico, ni una cadena vacía en un campo de texto. Cero y una cadena vacía son valores, mientras que NULL indica la ausencia de valor.

Manejo de errores: En algunas operaciones, los valores NULL pueden generar resultados inesperados o errores. Por eso es importante manejarlos correctamente en la programación.

Null - Glosario de MDN Web Docs: Definiciones de términos relacionados con la Web | MDN. (2023, November 13). MDN Web Docs. https://developer.mozilla.org/es/docs/Glossary/Null

David-Engel. (n.d.). Tratamiento de valores NULL - ADO.NET Provider for SQL Server. Microsoft Learn. https://learn.microsoft.com/es-es/sql/connect/ado-net/sql/handle-null-values?view=sql-server-ver16

¿Resultado de operarlo con los diferentes tipos de operadores: aritméticos, lógicos y de comparación?

Operadores aritméticos:

Cualquier operación aritmética con NULL da como resultado NULL. Ejemplo: 5 + NULL = NULL.

Operadores lógicos:

NULL AND cualquier_valor = NULL

NULL OR cualquier_valor = cualquier_valor

NOT NULL = NULL

Operadores de comparación:

NULL = cualquier_valor → FALSO

NULL <> cualquier_valor → DESCONOCIDO

NULL IS NULL → VERDADERO

David-Engel. (n.d.). Tratamiento de valores NULL - ADO.NET Provider for SQL Server. Microsoft Learn. https://learn.microsoft.com/es-es/sql/connect/ado-net/sql/handle-null-values?view=sql-server-ver16

B. JUNTA

¿Cuáles son las diferencias entre junta interna y externa?

Característica	Junta Interna (Inner Join)	Junta Externa (Outer Join)
Filas mostradas	Solo coincidencias	Incluye filas sin coincidencias
Resultado vacío	Si no hay coincidencias	No es vacío, muestra filas nulas
Tipos	Solo uno (Inner Join)	Izquierda, Derecha, Completa

La junta interna (INNER JOIN) combina filas de dos o más tablas que cumplen con una condición de coincidencia, mientras que la junta externa (OUTER JOIN) combina filas de dos o más tablas y también incluye filas que no tienen coincidencias en una o ambas tablas. (Aria(2024))

¿Qué opciones se tienen para la junta interna?

- Inner Join simple: Combina dos tablas basándose en una condición de coincidencia.
- Inner Join con múltiples condiciones: Agregar más de una condición para la coincidencia usando el AND.
- Inner Join con alias: Renombrar las tablas para que se pueda hacer la consulta más legible.
- Inner Join con subconsultas: Se pueden usar subconsultas en la cláusula ON para hacer las coincidencias más complejas
- Inner Join con más de dos tablas: Se combinan más de dos tablas en una sola consulta.

¿Qué opciones se tienen para la junta externa?

- Junta externa izquierda: Muestra todos los registros de la tabla izquierda y los registros que coinciden en la tabla derecha.
- Junta externa derecha: Muestra todos los registros de la tabla derecha y los registros que coinciden en la tabla izquierda.
- Junta externa completa: Muestra los registros completos de ambas tablas, con coincidencias donde existan.
- Junta externa izquierda con múltiples condiciones: Se agregan más condiciones en la cláusula ON para hacer coincidencias más específicas.
- Junta externa con alias: Renombrar las tablas para que se pueda hacer la consulta más legible

- Junta externa con más de dos tablas: Se combinan más de dos tablas en una sola consulta.

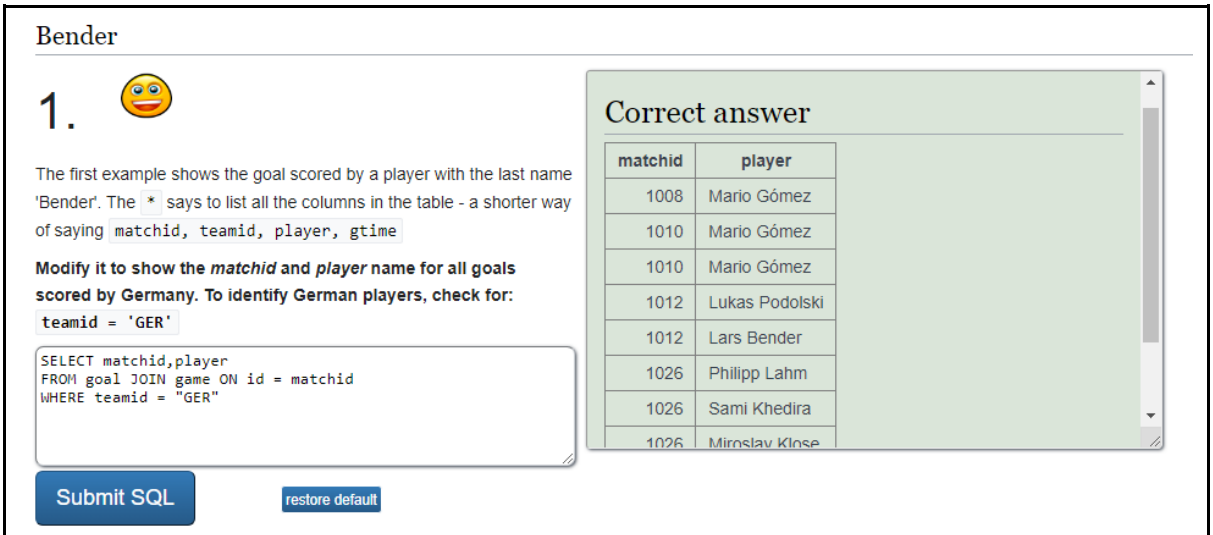
Fuente: *Combinar tablas y consultas - Soporte técnico de Microsoft*. (s. f.). https://support.microsoft.com/es-es/topic/combinar-tablas-y-consultas-3f5838bd-24a0-4832-9bc1-07061a1478f6#:~:text=En%20una%20combinaci%C3%B3n%20interna%2C%20no,los%20resultados%20de%20la%20consulta.

PRÁCTICA

A. Realicen los ejercicios propuestos en los siguientes tutoriales. Utilice el motor MySQL

TUTORIALES

JOIN




```
SELECT matchid,player
FROM goal JOIN game ON id = matchid
WHERE teamid = "GER"
```

Game 1012

2.

From the previous query you can see that Lars Bender's scored a goal in game 1012. Now we want to know what teams were playing in that match.

Notice in the that the column `matchid` in the `goal` table corresponds to the `id` column in the `game` table. We can look up information about game 1012 by finding that row in the **game** table.

Show id, stadium, team1, team2 for just game 1012

```
SELECT id,stadium,team1,team2
FROM game
WHERE id = 1012
```

Submit SQL restore default

Correct answer

id	stadium	team1	team2
1012	Arena Lviv	DEN	GER

```
SELECT id,stadium,team1,team2
FROM game
WHERE id = 1012
```

JOIN

3.

You can combine the two steps into a single query with a `JOIN`.

```
SELECT *
  FROM game JOIN goal ON (id=matchid)
```

The **FROM** clause says to merge data from the goal table with that from the game table. The **ON** says how to figure out which rows in **game** go with which rows in **goal** - the **matchid** from **goal** must match **id** from **game**. (If we wanted to be more clear/specific we could say `ON (game.id=goal.matchid)`

The code below shows the player (from the goal) and stadium name (from the game table) for every goal scored.

Modify it to show the player, teamid, stadium and mdate for every German goal.

```
SELECT player,teamid,stadium,mdate
FROM game JOIN goal ON (id=matchid)
WHERE teamid LIKE "%GER%"
```

Submit SQL restore default

Correct answer

player	teamid	stadium	mdate
Mario Gómez	GER	Arena Lviv	9 June 2012
Mario Gómez	GER	Metalist Stadium	13 June 2012
Mario Gómez	GER	Metalist Stadium	13 June 2012
Lukas Podolski	GER	Arena Lviv	17 June 2012

```
SELECT player,teamid,stadium,mdate
FROM game JOIN goal ON (id=matchid)
WHERE teamid LIKE "%GER%"
```

Mario goals

4.

Use the same `JOIN` as in the previous question.

Show the team1, team2 and player for every goal scored by a player called Mario `player LIKE 'Mario%'`

```
SELECT team1,team2,player
FROM goal JOIN game ON id = matchid
WHERE player LIKE "%Mario%"
```

Submit SQL restore default

Correct answer

team1	team2	player
GER	POR	Mario Gómez
NED	GER	Mario Gómez
NED	GER	Mario Gómez
IRL	CRO	Mario Mandžukic
IRL	CRO	Mario Mandžukic
ITA	CRO	Mario Mandžukic
ITA	IRL	Mario Balotelli

```
SELECT team1,team2,player
FROM goal JOIN game ON id = matchid
WHERE player LIKE "%Mario%"
```

Ten minute goals

5.

The table `eteam` gives details of every national team including the coach. You can `JOIN` `goal` to `eteam` using the phrase `goal JOIN eteam on teamid=id`

Show `player` , `teamid` , `coach` , `gtime` **for all goals scored in the first 10 minutes** `gtime<=10`

```
SELECT player, teamid,coach, gtime
  FROM goal JOIN eteam ON teamid = id
 WHERE gtime<=10
```

Submit SQL restore default

Correct answer

player	teamid	coach	gtime
Petr Jirácek	CZE	Michal Bílek	3
Václav Pilar	CZE	Michal Bílek	6
Mario Mandžukic	CRO	Slaven Bilic	3
Fernando Torres	ESP	Vicente del Bosque	4

```
SELECT player, teamid,coach, gtime
  FROM goal JOIN eteam ON teamid = id
 WHERE gtime<=10
```

Coach Fernando

6.

To `JOIN` `game` with `eteam` you could use either `game JOIN eteam ON (team1=eteam.id)` or `game JOIN eteam ON (team2=eteam.id)`

Notice that because `id` is a column name in both `game` and `eteam` you must specify `eteam.id` instead of just `id`

List the dates of the matches and the name of the team in which 'Fernando Santos' was the team1 coach.

```
SELECT mdate,teamname
FROM game JOIN eteam ON team1=eteam.id
WHERE coach = "Fernando Santos"
```

Submit SQL restore default

Correct answer

mdate	teamname
12 June 2012	Greece
16 June 2012	Greece

```
SELECT mdate,teamname
FROM game JOIN eteam ON team1=eteam.id
WHERE coach = "Fernando Santos"
```

Players at the National Stadium

7.

List the player for every goal scored in a game where the stadium was 'National Stadium, Warsaw'

```
SELECT player
FROM goal JOIN game ON matchid = id
WHERE stadium = "National Stadium, Warsaw"
```

Submit SQL restore default

player
Robert Lewandowski
Dimitris Salpingidis
Alan Dzagoev
Jakub Blaszczykowski
Giorgos Karagounis
Cristiano Ronaldo
Mario Balotelli
Mario Balotelli
Mesut Özil

```
SELECT player
FROM goal JOIN game ON matchid = id
WHERE stadium = "National Stadium, Warsaw"
```

More difficult questions

8.

The example query shows all goals scored in the Germany-Greece quarterfinal.

Instead show the name of all players who scored a goal against Germany.

HINT

Select goals scored only by non-German players in matches where GER was the id of either **team1** or **team2**.

You can use `teamid!='GER'` to prevent listing German players.

You can use `DISTINCT` to stop players being listed twice.

```
SELECT DISTINCT player
  FROM game JOIN goal ON matchid = id
    WHERE teamid != "GER" AND (team1 = "GER" OR team2 =
"GER")
```

Submit SQL restore default

Correct answer

player
Robin van Persie
Michael Krohn-Dehli
Georgios Samaras
Dimitris Salpingidis
Mario Balotelli

```
SELECT DISTINCT player
  FROM game JOIN goal ON matchid = id
    WHERE teamid != "GER" AND (team1 = "GER" OR team2 = "GER")
```

Goals by team

9.

Show teamname and the total number of goals scored.

COUNT and GROUP BY

```
SELECT teamname, COUNT(teamid)
FROM goal JOIN eteam ON teamid = id
GROUP BY teamname
```

Submit SQL restore default

Correct answer

teamname	COUNT(teamid)
Croatia	4
Czech Republic	4
Denmark	4
England	5
France	3
Germany	10
Greece	5

```
SELECT teamname, COUNT(teamid)
FROM goal JOIN eteam ON teamid = id
GROUP BY teamname
```

Goals by Stadium

10.

Show the stadium and the number of goals scored in each stadium.

```
SELECT stadium,COUNT(matchid)
FROM game JOIN goal ON id = matchid
GROUP BY stadium
```

Submit SQL restore default

Correct answer

stadium	COUNT(matchid)
Arena Lviv	9
Donbass Arena	7
Metalist Stadium	7
National Stadium, Warsaw	9
Olimpiyskiy National Sports Complex	14
PGE Arena Gdansk	13
Stadion Miejski (Poznan)	8

```
SELECT stadium,COUNT(matchid)
FROM game JOIN goal ON id = matchid
GROUP BY stadium
```

Matches for POL

11.

For every match involving 'POL', show the matchid, date and the number of goals scored.

```
SELECT matchid,mdate,COUNT(teamid)
  FROM game JOIN goal ON matchid = id
 WHERE (team1 = 'POL' OR team2 = 'POL')
GROUP BY mdate
ORDER BY matchid
```

Submit SQL restore default

Correct answer

matchid	mdate	COUNT(teamid)
1001	8 June 2012	2
1004	12 June 2012	2
1005	16 June 2012	1

```
SELECT matchid,mdate,COUNT(teamid)
  FROM game JOIN goal ON matchid = id
 WHERE (team1 = 'POL' OR team2 = 'POL')
GROUP BY mdate
ORDER BY matchid
```

Goals for GER

12. 

For every match where 'GER' scored, show matchid, match date and the number of goals scored by 'GER'

```
SELECT matchid,mdate,COUNT(*)
FROM game JOIN goal ON id = matchid
WHERE teamid = "GER"
GROUP BY matchid
```

Submit SQL restore default

Correct answer

matchid	mdate	COUNT(*)
1008	9 June 2012	1
1010	13 June 2012	2
1012	17 June 2012	2
1026	22 June 2012	4
1030	28 June 2012	1

```
SELECT matchid,mdate,COUNT(*)
FROM game JOIN goal ON id = matchid
WHERE teamid = "GER"
GROUP BY matchid
```

Goals for each match

13. 

List every match with the goals scored by each team as shown. This will use "CASE WHEN" which has not been explained in any previous exercises.

You will also have to use a LEFT JOIN to include the 0-0 games; again, this has not been explained in any previous exercises.

mdate	team1	score1	team2	score2
1 July 2012	ESP	4	ITA	0
10 June 2012	ESP	1	ITA	1
10 June 2012	IRL	1	CRO	3
...				

Notice in the query given every goal is listed. If it was a team1 goal then a 1 appears in score1, otherwise there is a 0. You could SUM this column to get a count of the goals scored by team1. **Sort your result by mdate, matchid, team1 and team2.**

```
SELECT mdate,
  team1,
  SUM(CASE WHEN teamid=team1 THEN 1 ELSE 0 END) AS score1
,team2
,SUM(CASE WHEN teamid=team2 THEN 1 ELSE 0 END) AS score2
 FROM game LEFT JOIN goal ON matchid = id
GROUP BY id,mdate,team1,team2
ORDER BY mdate,matchid,team1,team2
```

Submit SQL restore default

Correct answer

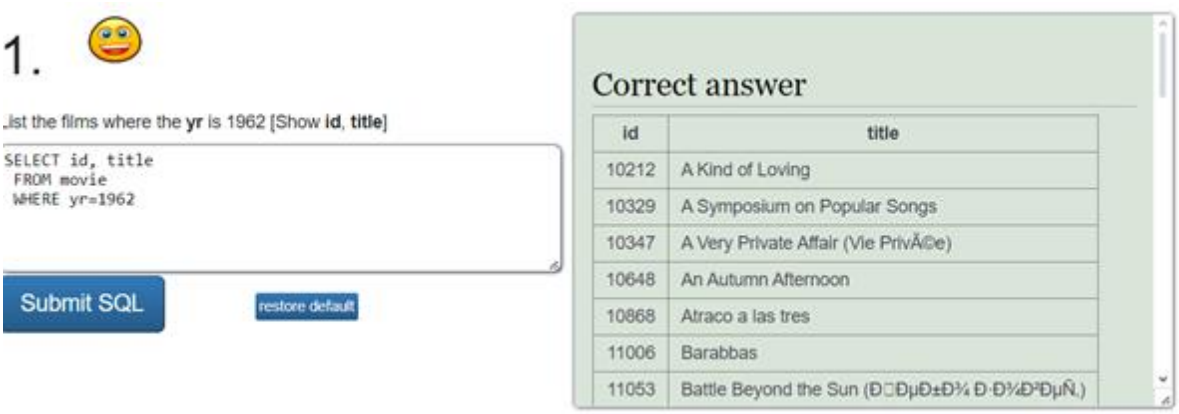
mdate	team1	score1	team2	score2
1 July 2012	ESP	4	ITA	0
10 June 2012	ESP	1	ITA	1
10 June 2012	IRL	1	CRO	3
11 June 2012	FRA	1	ENG	1
11 June 2012	UKR	2	SWE	1
12 June 2012	GRE	1	CZE	2
12 June 2012	POL	1	RUS	1

```
SELECT mdate,
  team1,
  SUM(CASE WHEN teamid=team1 THEN 1 ELSE 0 END) AS score1
,team2
,SUM(CASE WHEN teamid=team2 THEN 1 ELSE 0 END) AS score2
```

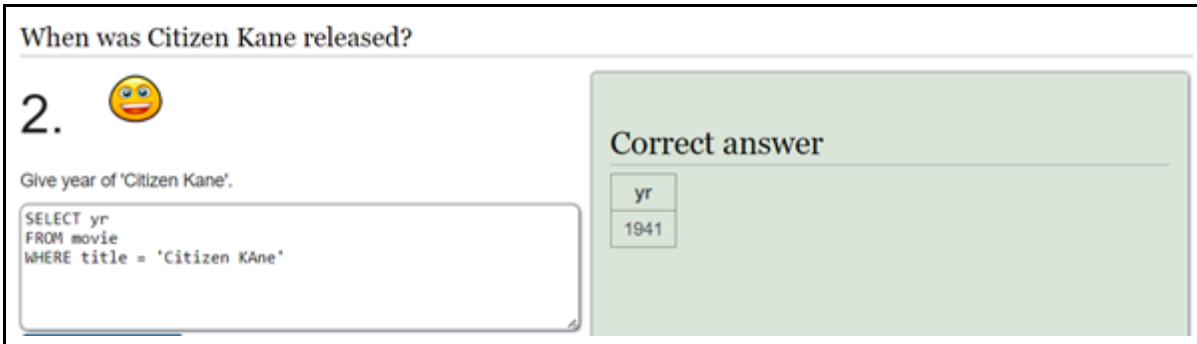
```
 FROM game LEFT JOIN goal ON matchid = id
GROUP BY id,mdate,team1,team2
ORDER BY mdate,matchid,team1,team2
```

MORE JOIN

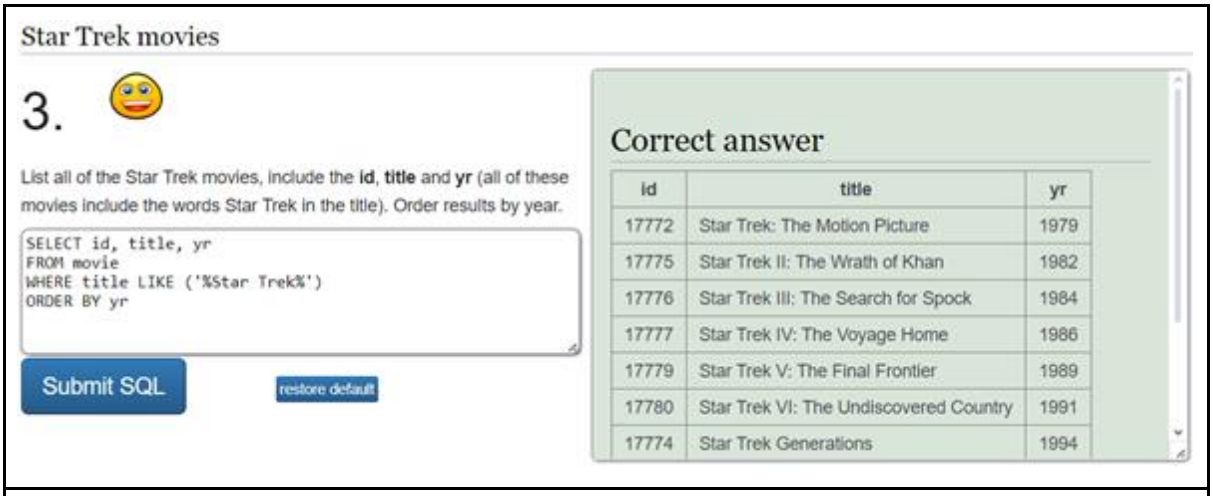
1.



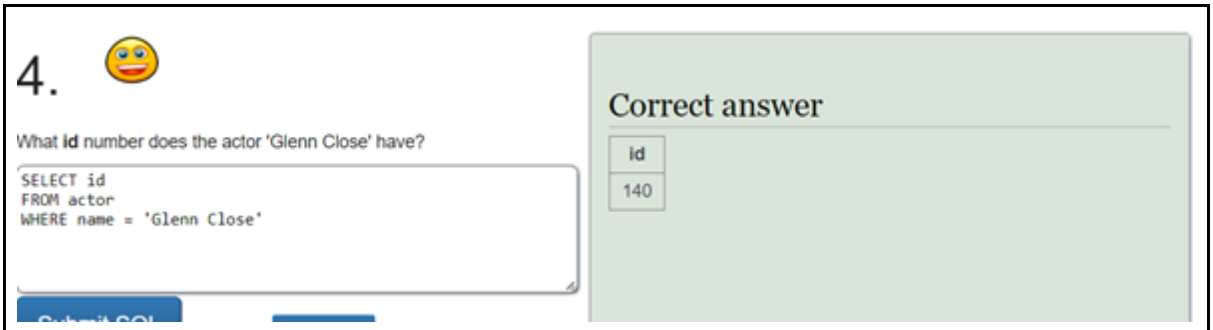

```
SELECT id, title
FROM movie
WHERE yr = 1962
```



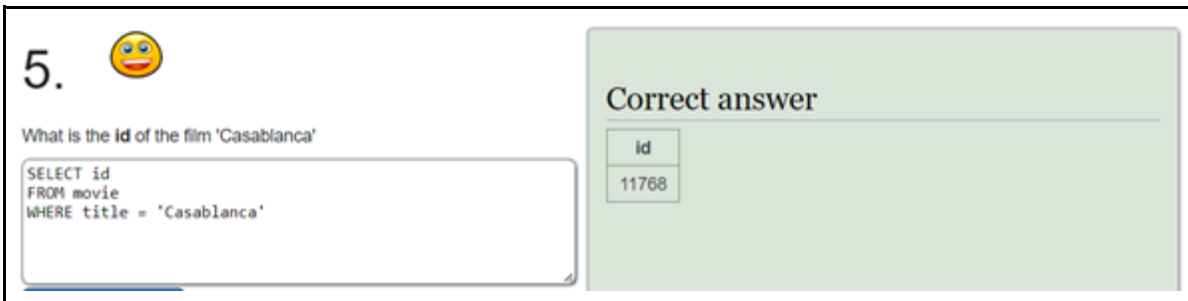

```
SELECT yr
FROM movie
WHERE title = 'Citizen Kane'
```



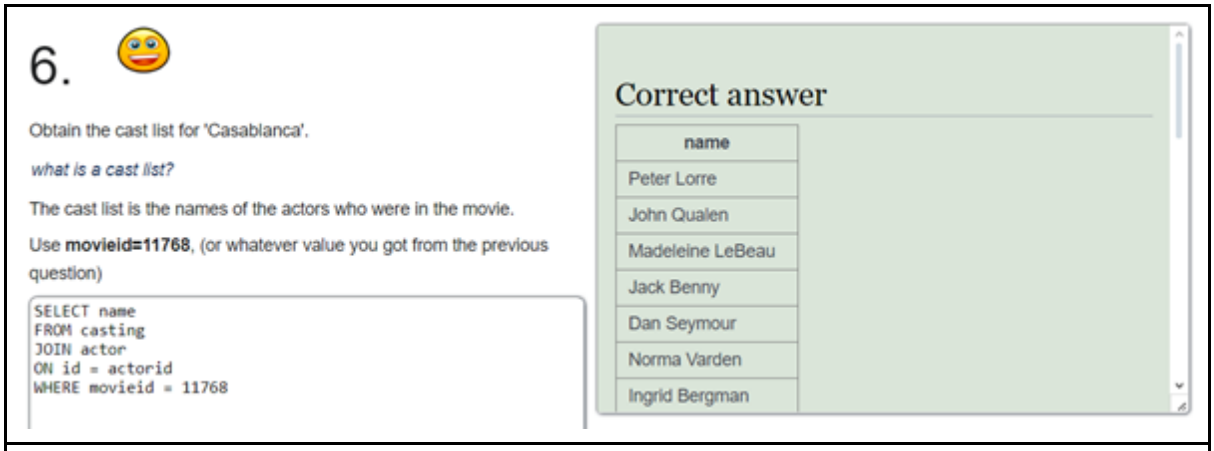
SELECT id, title, yr
FROM movie
WHERE title LIKE (‘%Star Trek%’)
ORDER BY yr



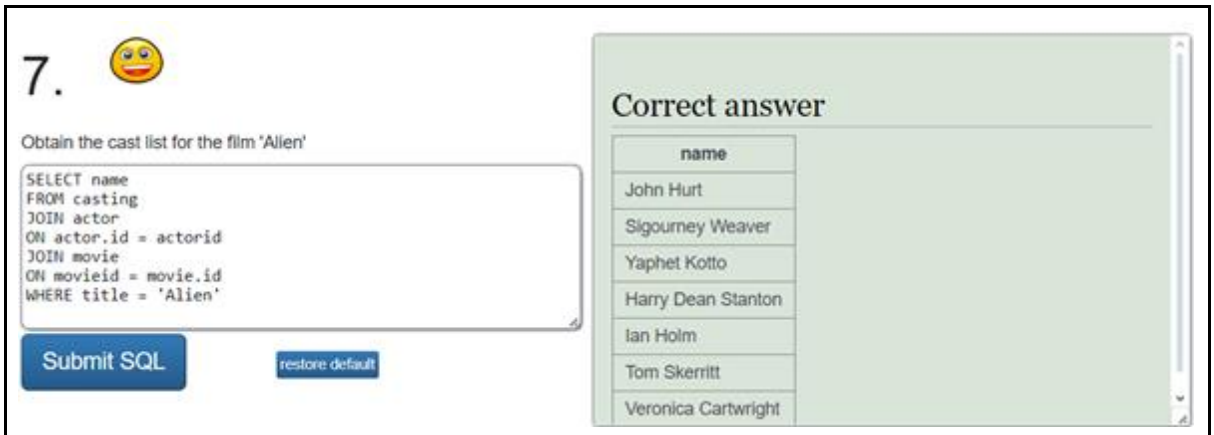
SELECT id
FROM actor
WHERE name = ‘Glenn Close’



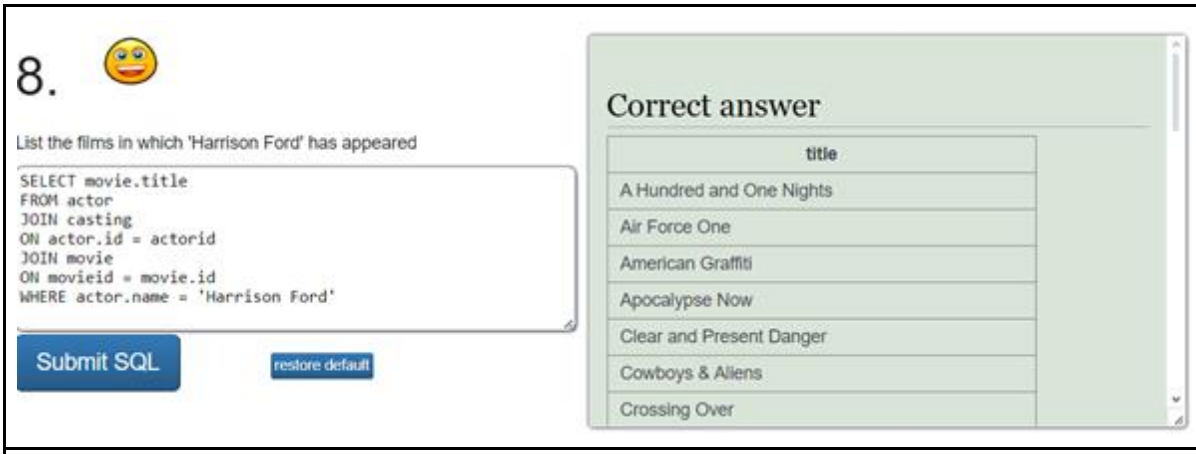
SELECT id
FROM movie
WHERE title = ‘Casablanca’



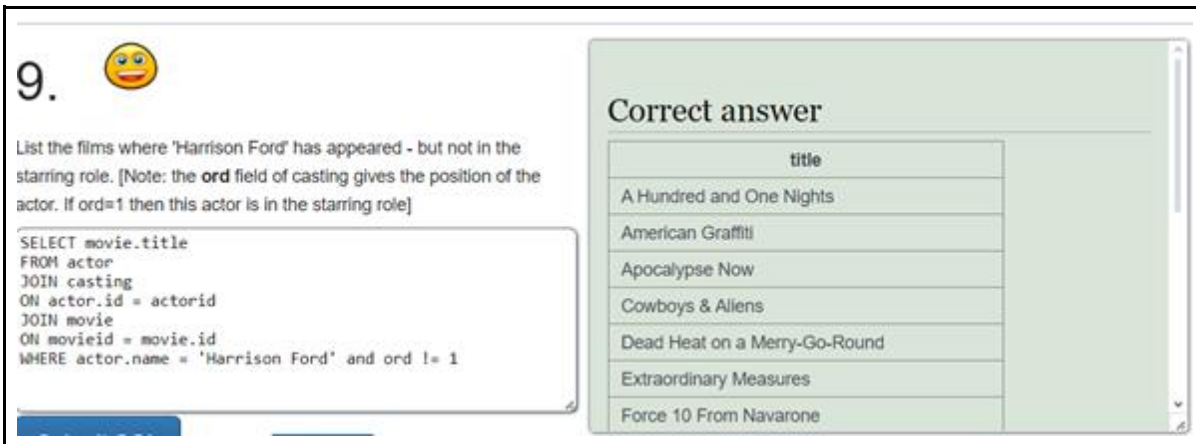

```
SELECT name
FROM casting
JOIN actor
ON id = actorid
WHERE movieud = 11768
```



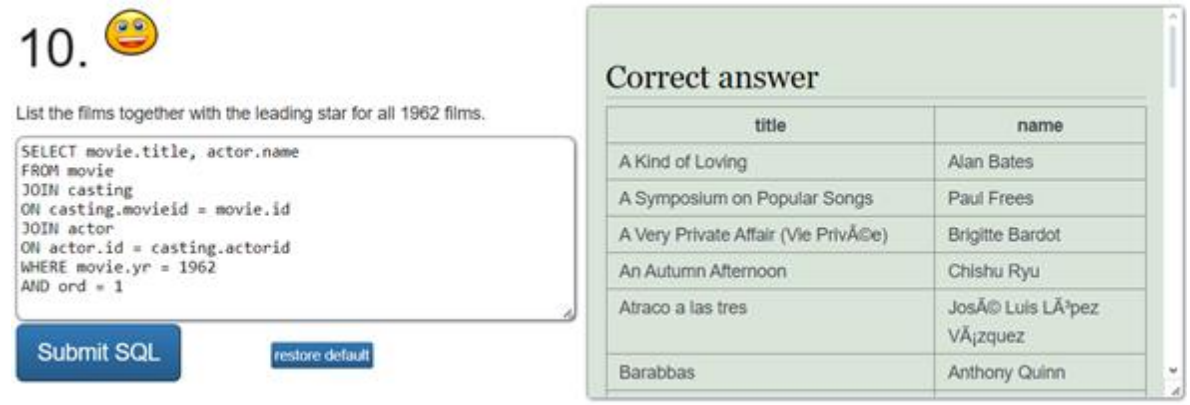

```
SELECT name
FROM casting
JOIN actor
ON actor.id = actorid
JOIN movie
ON movieid = movie.id
WHERE title = ‘Alien’
```



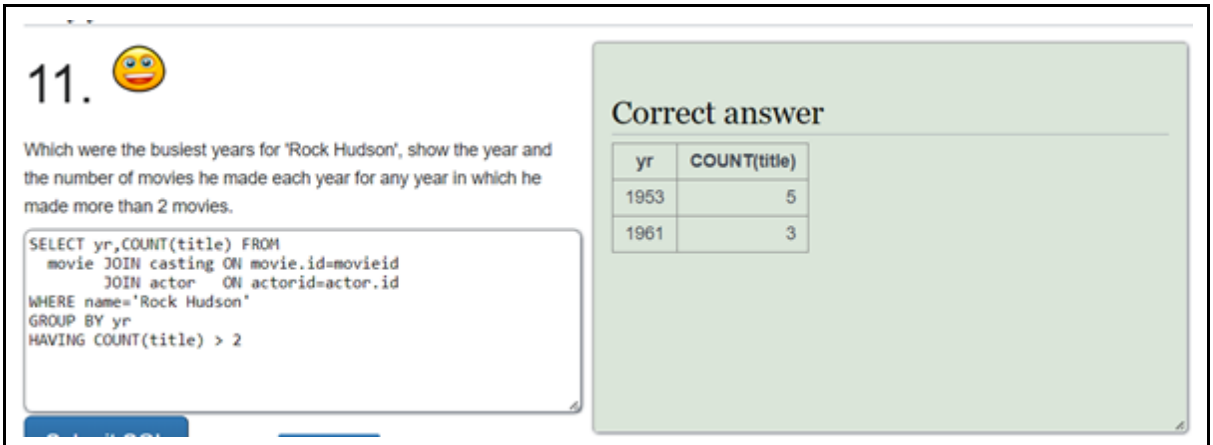

```
SELECT movie.title
FROM actor
JOIN casting
ON actor.id = actorid
JOIN movie
ON movieid = movie.id
WHERE actor.name = 'Harrison Ford'
```




```
SELECT movie.title
FROM actor
JOIN casting
ON actor.id = actorid
JOIN movie
ON movieid = movie.id
WHERE actor.name = 'Harrison Ford' and ord != 1
```




```
SELECT movie.title, actor.name
FROM movie
JOIN casting
ON casting.movieid = movie.id
JOIN actor
ON actor.id = casting.actorid
WHERE movie.yr = 1962
AND ord = 1
```



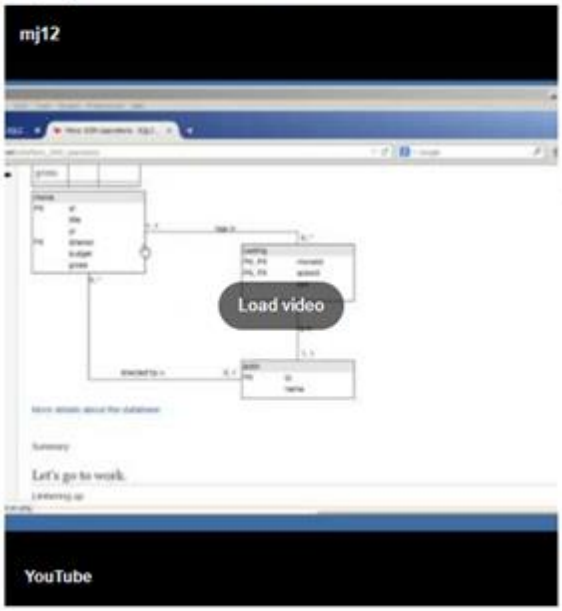

```
SELECT yr, COUNT(title) FROM
    movie JOIN casting ON movie.id = movieid
            JOIN actor on actorid = actor.id
WHERE name = 'Rock Hudson'
GROUP BY yr
HAVING COUNT(title) >2
```

Lead actor in Julie Andrews movies

12.

List the film title and the leading actor for all of the films 'Julie Andrews' played in.

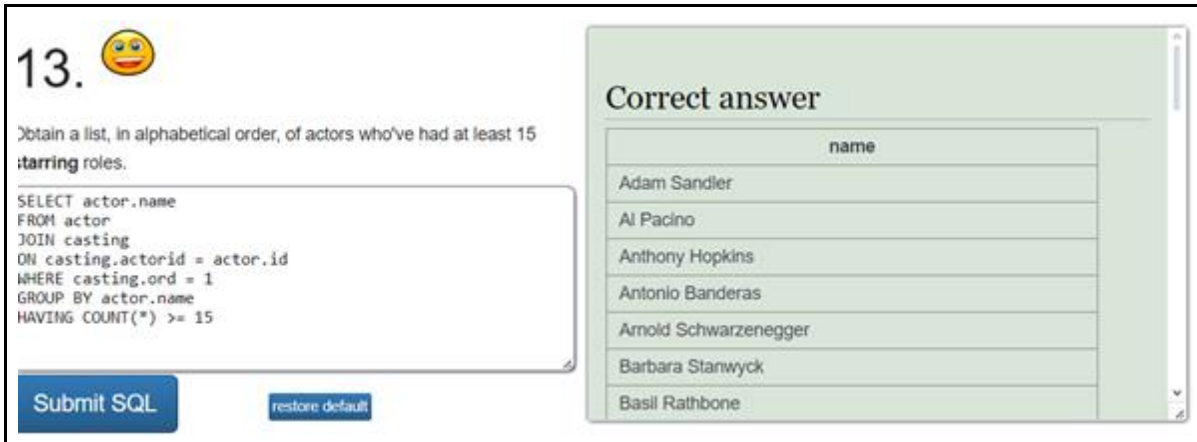
Did you get "Little Miss Marker twice"?



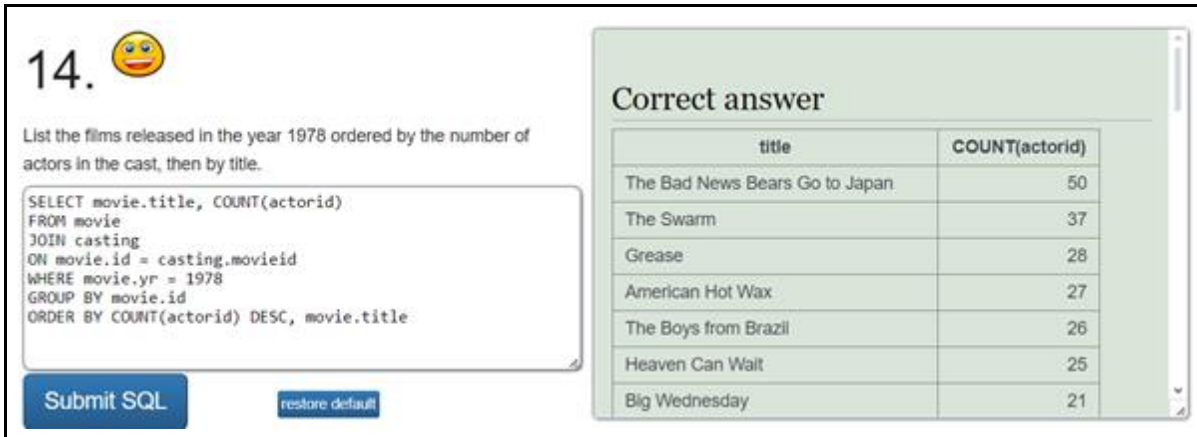
Relative Values	Julie Andrews
S.O.B.	Julie Andrews
Shrek the Third	Mike Myers
Star!	Julie Andrews
The Americanization of Emily	James Garner
The Pink Panther Strikes Again	Peter Sellers
The Princess Diaries	Anne Hathaway
The Princess Diaries 2: Royal Engagement	Anne Hathaway
The Sound of Music	Julie Andrews
The Tamarind Seed	Julie Andrews
Thoroughly Modern Millie	Julie Andrews

```
SELECT DISTINCT m.title, a.name
FROM (SELECT movie.*
      FROM movie
      JOIN casting
      ON casting.movieid = movie.id
      JOIN actor
      ON actor.id = casting.actorid
      WHERE actor.name = 'Julie Andrews') AS m
JOIN (SELECT actor.*, casting.movieid AS movieid
      FROM actor
      JOIN casting
      ON casting.actorid = actor.id
      WHERE casting.ord = 1) as a
ON m.id = a.movieid
ORDER BY m.title
```

```
SELECT DISTINCT m.title, a.name
FROM (SELECT movie.*
      FROM movie
      JOIN casting
      ON casting.movieid = movie.id
      JOIN actor
      ON actor.id = casting.actorid
      WHERE actor.name = 'Julie Andrews') AS m
JOIN (SELECT actor.*, casting.movieid AS movieid
      FROM actor
      JOIN casting
      ON casting.actorid = actor.id
      WHERE casting.ord = 1) as a
ON m.id = a.movieid
ORDER BY m.title
```




```
SELECT actor.name
FROM actor
JOIN casting
ON casting.actorid = actor.id
WHERE casting.ord = 1
GROUP BY actor.name
HAVING COUNT(*) >= 15
```




```
SELECT movie.title, COUNT(actorid)
FROM movie
JOIN casting
ON movie.id = casting.movieid
WHERE movie.yr = 1978
GROUP BY movie.id
ORDER BY COUNT(actorid) DESC, movie.title
```

15.

List all the people who have worked with 'Art Garfunkel'.

```
SELECT a.name
  FROM (SELECT movie.*
          FROM movie
          JOIN casting
            ON casting.movieid = movie.id
          JOIN actor
            ON actor.id = casting.actorid
         WHERE actor.name = 'Art Garfunkel') AS m
  JOIN (SELECT actor.*, casting.movieid
          FROM actor
          JOIN casting
            ON casting.actorid = actor.id
         WHERE actor.name != 'Art Garfunkel') as a
    ON m.id = a.movieid
```

Correct answer

name
Mark Ruffalo
Ryan Phillippe
Mike Myers
Neve Campbell
Salma Hayek
Sela Ward
Breckin Meyer

```
SELECT a.name
  FROM (SELECT movie.*
          FROM movie
          JOIN casting
            ON casting.movieid = movie.id
          JOIN actor
            ON actor.id = casting.actorid
         WHERE actor.name = 'Art Garfunkel') AS m
  JOIN (SELECT actor.*, casting.movieid
          FROM actor
          JOIN casting
            ON casting.actorid = actor.id
         WHERE actor.name != 'Art Garfunkel') as a
    ON m.id = a.movieid
```

USING NULL

1.

List the teachers who have NULL for their department.

Why we cannot use =

You might think that the phrase dept=NULL would work here but it doesn't - you can use the phrase `dept IS NULL`

That's not a proper explanation.

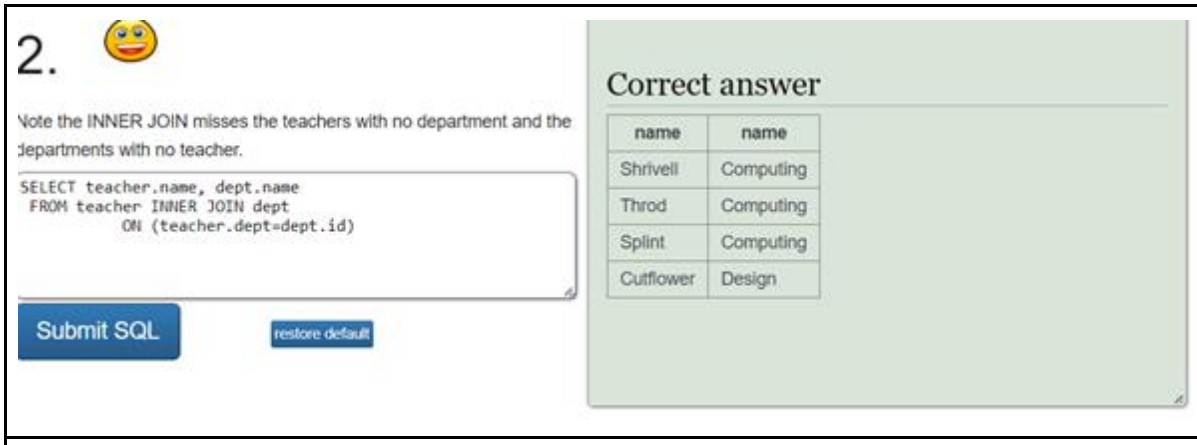
No it's not, but you can read a better explanation at Wikipedia:NULL.

```
SELECT name
FROM teacher
WHERE dept is NULL
```

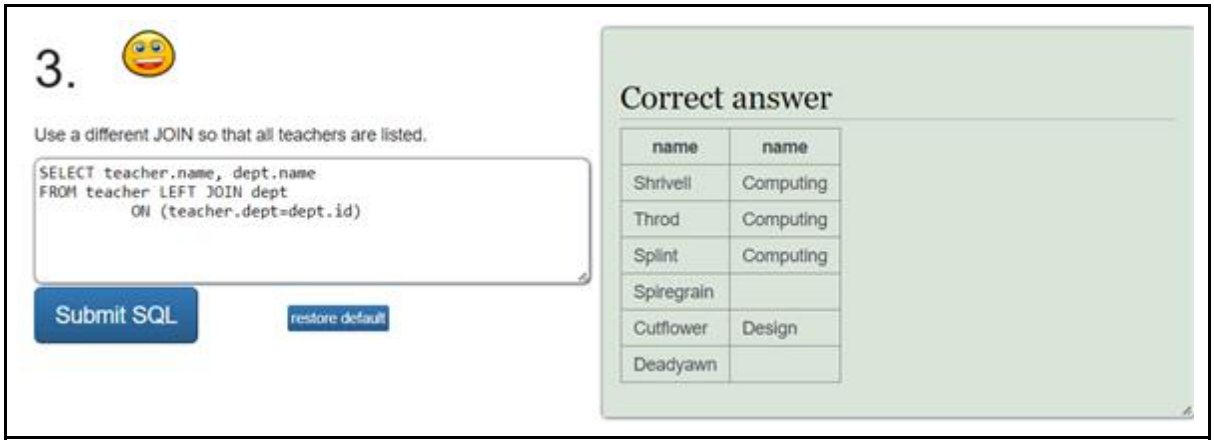
Correct answer

name
Spiregrain
Deadyawn

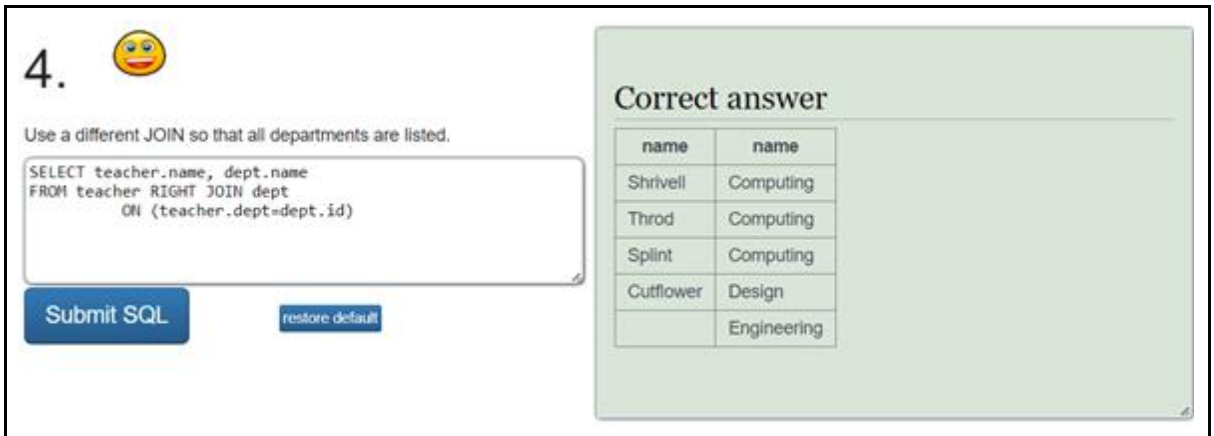
```
SELECT name
FROM teacher
WHERE dept is NULL
```



SELECT teacher.name, dept.name
FROM teacher INNER JOIN dept
ON (teacher.dept=dept.id)



SELECT teacher.name, dept.name
FROM teacher LEFT JOIN dept
ON (teacher.dept=dept.id)



SELECT teacher.name, dept.name
FROM teacher RIGHT JOIN dept
ON (teacher.dept=dept.id)

5.

Use COALESCE to print the mobile number. Use the number '07986 444 2266' if there is no number given. **Show teacher name and mobile number or '07986 444 2266'**

```
SELECT name,
COALESCE(mobile, '07986 444 2266')
FROM teacher
```

Submit SQL restore default

Correct answer

name	COALESCE(mobi..
Shrivell	07986 555 1234
Throd	07122 555 1920
Splint	07986 444 2266
Spiregrain	07986 444 2266
Cutflower	07996 555 6574
Deadyawn	07986 444 2266

SELECT name,
COALESCE(mobile, '07986 444 2266')
FROM teacher

6.

Use the COALESCE function and a LEFT JOIN to print the teacher **name** and department name. Use the string 'None' where there is no department.

```
SELECT COALESCE(teacher.name, 'NONE'), COALESCE(dept.name,
'None')
FROM teacher LEFT JOIN dept ON (teacher.dept=dept.id)
```

Submit SQL restore default

Correct answer

COALESCE(teac..	COALESCE(dept..
Shrivell	Computing
Throd	Computing
Splint	Computing
Spiregrain	None
Cutflower	Design
Deadyawn	None

SELECT COALESCE(teacher.name, 'NONE'), COALESCE(dept.name, 'None')
FROM teacher LEFT JOIN dept ON (teacher.dept=dept.id)

7.

Use COUNT to show the number of teachers and the number of mobile phones.

```
SELECT COUNT(name), COUNT(mobile)
FROM teacher
```

Submit SQL restore default

Correct answer

COUNT(name)	COUNT(mobile)
6	3

SELECT COUNT(name), COUNT(mobile)

FROM teacher

8.

Use COUNT and GROUP BY **dept.name** to show each department and the number of staff. Use a RIGHT JOIN to ensure that the Engineering department is listed.

```
SELECT dept.name, COUNT(teacher.name)
FROM teacher RIGHT JOIN dept ON (teacher.dept=dept.id)
GROUP BY dept.name
```

Correct answer

name	COUNT(teacher..
Computing	3
Design	1
Engineering	0

SELECT dept.name, COUNT(teacher.name)
FROM teacher RIGHT JOIN dept ON (teacher.dept=dept.id)
GROUP BY dept.name

9.

Use CASE to show the **name** of each teacher followed by 'Sci' if the teacher is in **dept** 1 or 2 and 'Art' otherwise.

```
SELECT teacher.name,
CASE WHEN dept.id = 1 THEN 'Sci'
     WHEN dept.id = 2 THEN 'Sci'
     ELSE 'Art' END
FROM teacher LEFT JOIN dept ON (teacher.dept=dept.id)
```

Submit SQL restore default

Correct answer

name	CASE WHEN dep..
Shrivell	Sci
Throd	Sci
Splint	Sci
Spiregrain	Art
Cutflower	Sci
Deadyawn	Art

SELECT teacher.name,
CASE WHEN dept.id = 1 THEN 'Sci'
WHEN dept.id = 2 THEN 'Sci'
ELSE 'Art' END
FROM teacher LEFT JOIN dept ON (teacher.dept=dept.id)

10.

Use CASE to show the name of each teacher followed by 'Sci' if the teacher is in dept 1 or 2, show 'Art' if the teacher's dept is 3 and 'None' otherwise.

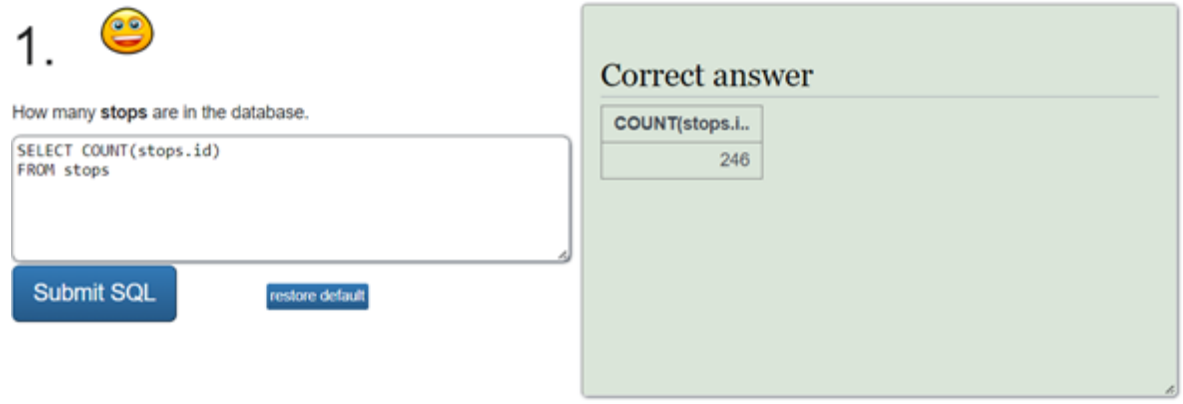
```
SELECT teacher.name,
CASE
WHEN dept.id = 1 THEN 'Sci'
WHEN dept.id = 2 THEN 'Sci'
WHEN dept.id = 3 THEN 'Art'
ELSE 'None' END
FROM teacher LEFT JOIN dept ON (dept.id=teacher.dept)
```

Correct answer

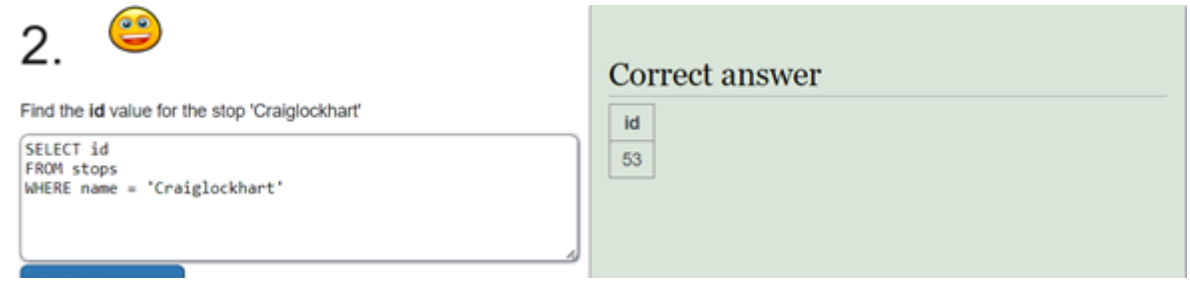
name	CASE WHEN de..
Shrivell	Sci
Throd	Sci
Splint	Sci
Spiregrain	None
Cutflower	Sci
Deadyawn	None

```
SELECT teacher.name,
CASE
WHEN dept.id = 1 THEN 'Sci'
WHEN dept.id = 2 THEN 'Sci'
WHEN dept.id = 3 THEN 'Art'
ELSE 'None' END
FROM teacher LEFT JOIN dept ON (dept.id=teacher.dept)
```

SELF JOIN




```
SELECT COUNT(stops.id)
FROM stops
```




```
SELECT id
FROM stops
WHERE name = 'Craiglockhart'
```

3.

Give the **id** and the **name** for the **stops** on the '4' 'LRT' service.

```
SELECT id, name
FROM stops
JOIN route
ON stops.id = route.stop
WHERE num = 4
AND company = 'LRT'
ORDER BY pos
```

Submit SQL restore default

Correct answer

id	name
19	Bingham
177	Northfield
149	London Road
194	Princes Street
115	Haymarket
53	Craiglockhart
179	Oxgangs

SELECT id, name
FROM stops
JOIN route
ON stops.id = route.stop
WHERE num = 4
AND company = 'LRT'
ORDER BY pos

4.

The query shown gives the number of routes that visit either London Road (149) or Craiglockhart (53). Run the query and notice the two services that link these **stops** have a count of 2. Add a HAVING clause to restrict the output to these two routes.

```
SELECT company, num, COUNT(*)
FROM route WHERE stop=149 OR stop=53
GROUP BY company, num
HAVING COUNT(*) = 2
```

Submit SQL

Correct answer

company	num	COUNT(*)
LRT	4	2
LRT	45	2

SELECT company, num, COUNT(*)
FROM route WHERE stop=149 OR stop=53
GROUP BY company, num
HAVING COUNT(*) = 2

5.

Execute the self join shown and observe that b.stop gives all the places you can get to from Craiglockhart, without changing routes. Change the query so that it shows the services from Craiglockhart to London Road.

```
SELECT a.company, a.num, a.stop, b.stop
 FROM route a
   JOIN route b
     ON (a.company = b.company
       AND a.num = b.num)
 WHERE a.stop = 53
   AND b.stop = 149
```

Submit SQL

Correct answer

company	num	stop	stop
LRT	4	53	149
LRT	45	53	149

```
SELECT a.company, a.num, a.stop, b.stop
 FROM route a
   JOIN route b
     ON (a.company = b.company
       AND a.num = b.num)
 WHERE a.stop = 53
   AND b.stop = 149
```

6.

The query shown is similar to the previous one, however by joining two copies of the **stops** table we can refer to **stops** by **name** rather than by number. Change the query so that the services between 'Craiglockhart' and 'London Road' are shown. If you are tired of these places try 'Fairmilehead' against 'Tollcross'

```
SELECT a.company, a.num, stopa.name, stopb.name
  FROM route a
          JOIN route b
                  ON (a.company = b.company
                      AND a.num = b.num)
          JOIN stops stopa
                  ON (a.stop = stopa.id)
          JOIN stops stopb
                  ON (b.stop = stopb.id)
  WHERE stopa.name = 'Craiglockhart'
          AND stopb.name = 'London Road'
```

Correct answer

company	num	name	name
LRT	4	Craiglockhart	London Road
LRT	45	Craiglockhart	London Road

```
SELECT a.company, a.num, stopa.name, stopb.name
  FROM route a
          JOIN route b
                  ON (a.company = b.company
                      AND a.num = b.num)
          JOIN stops stopa
                  ON (a.stop = stopa.id)
          JOIN stops stopb
                  ON (b.stop = stopb.id)
  WHERE stopa.name = 'Craiglockhart'
          AND stopb.name = 'London Road'
```

7.

Give a list of all the services which connect stops 115 and 137 ('Haymarket' and 'Leith')

```
SELECT DISTINCT a.company, a.num
  FROM route a
    JOIN route b
      ON (a.company = b.company
        AND a.num = b.num)
  WHERE (a.stop = 115
    AND b.stop = 137)
```

Submit SQL restore default

Correct answer

company	num
LRT	12
LRT	2
LRT	22
LRT	25
LRT	2A
SMT	C5

```
SELECT DISTINCT a.company, a.num
  FROM route a
    JOIN route b
      ON (a.company = b.company
        AND a.num = b.num)
  WHERE (a.stop = 115
    AND b.stop = 137)
```

8.

Give a list of the services which connect the **stops** 'Craiglockhart' and 'Tollcross'

```
SELECT DISTINCT a.company, a.num
  FROM route a
    JOIN route b
      ON (a.company = b.company
        AND a.num = b.num)
    JOIN stops stopa
      ON (a.stop = stopa.id)
    JOIN stops stopb
      ON (b.stop = stopb.id)
  WHERE stopa.name = 'Craiglockhart'
    AND stopb.name = 'Tollcross'
```

Correct answer

company	num
LRT	10
LRT	27
LRT	45
LRT	47

```
SELECT DISTINCT a.company, a.num
  FROM route a
    JOIN route b
      ON (a.company = b.company
        AND a.num = b.num)
    JOIN stops stopa
      ON (a.stop = stopa.id)
    JOIN stops stopb
      ON (b.stop = stopb.id)
  WHERE stopa.name = 'Craiglockhart'
    AND stopb.name = 'Tollcross'
```

9.

Give a distinct list of the **stops** which may be reached from 'Craiglockhart' by taking one bus, including 'Craiglockhart' itself, offered by the LRT company. Include the company and bus no. of the relevant services.

```
SELECT DISTINCT stopb.name, b.company, b.num
  FROM route a
    JOIN route b
      ON (a.company = b.company
        AND a.num = b.num)
    JOIN stops stopa
      ON (a.stop = stopa.id)
    JOIN stops stopb
      ON (b.stop = stopb.id)
  WHERE stopa.name = 'Craiglockhart'
```

Correct answer

name	company	num
Silverknowes	LRT	10
Muirhouse	LRT	10
Newhaven	LRT	10
Leith	LRT	10
Leith Walk	LRT	10
Princes Street	LRT	10
Tollcross	LRT	10

```
SELECT DISTINCT stopb.name, b.company, b.num
  FROM route a
    JOIN route b
      ON (a.company = b.company
        AND a.num = b.num)
    JOIN stops stopa
      ON (a.stop = stopa.id)
    JOIN stops stopb
      ON (b.stop = stopb.id)
  WHERE stopa.name = 'Craiglockhart'
```

10.

Find the routes involving two buses that can go from **Craiglockhart** to **Lochend**.
Show the bus no. and company for the first bus, the name of the stop for the transfer,
and the bus no. and company for the second bus.

Hint

```
y.company
  FROM
    (SELECT DISTINCT stopb.name, b.company,
b.num
    FROM route a
      JOIN route b
        ON (a.company = b.company
          AND a.num = b.num)
      JOIN stops stopa
        ON (a.stop = stopa.id)
      JOIN stops stopb
        ON (b.stop = stopb.id)
    WHERE stopa.name = 'Craiglockhart') x
```

Correct answer

num	company	name	num	company
10	LRT	Leith	34	LRT
10	LRT	Leith	35	LRT
10	LRT	Leith	87	LRT
10	LRT	Leith	C5	SMT
10	LRT	Princes Street	65	LRT
10	LRT	Princes Street	C5	SMT
27	LRT	Canonmills	34	LRT

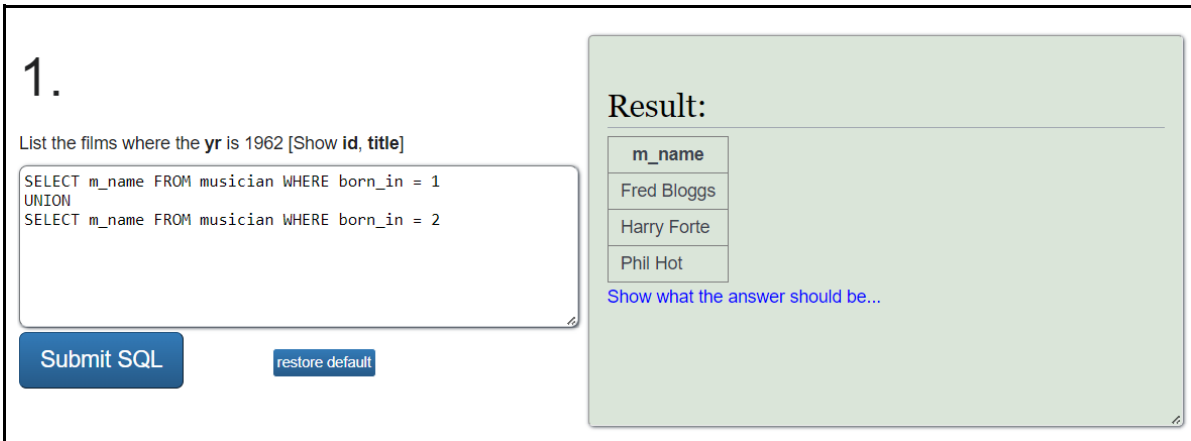
```
SELECT x.num, x.company, x.name, y.num, y.company
  FROM
    (SELECT DISTINCT stopb.name, b.company, b.num
    FROM route a
      JOIN route b
        ON (a.company = b.company
          AND a.num = b.num)
```

```
  JOIN stops stopa
    ON (a.stop = stopa.id)
  JOIN stops stopb
    ON (b.stop = stopb.id)
WHERE stopa.name = 'Craiglockhart') x

JOIN

(SELECT DISTINCT stopc.name, c.company, c.num
FROM route c
JOIN route d
  ON (c.company = d.company
    AND c.num=d.num)
  JOIN stops stopc ON (c.stop = stopc.id)
  JOIN stops stopd ON (d.stop = stopd.id)
WHERE stopd.name = 'Lochend') y

  ON (y.name = x.name)

ORDER BY x.num, name, y.num
```

C. Propongan preguntas que cumplan los siguientes requerimientos.
Usen la base de datos musician.

-5 consultas: una para cada operador de conjuntos

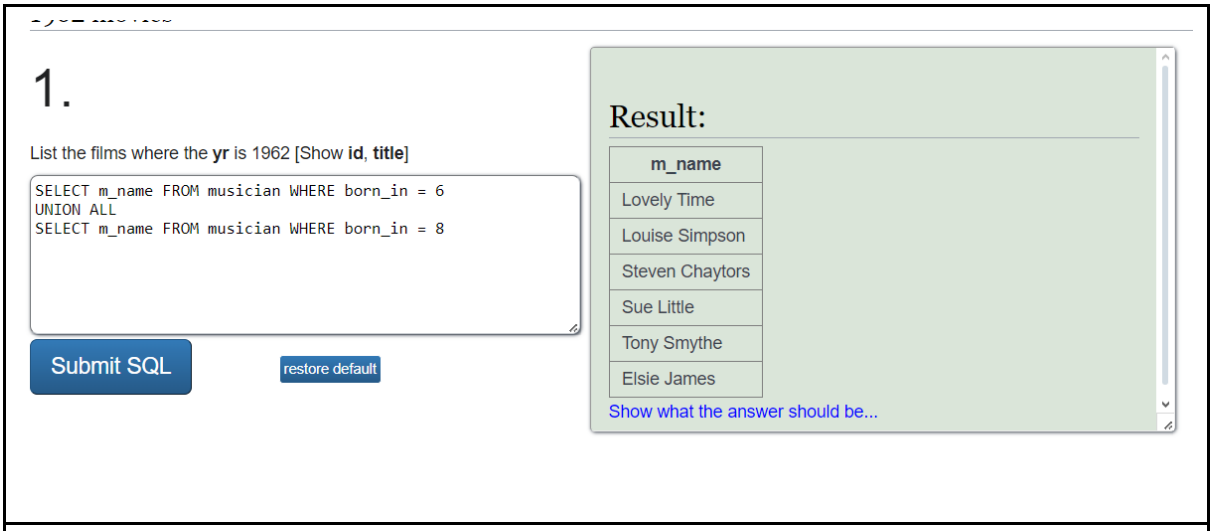
-¿Qué músicos nacieron en Manchester y Edinburg?
(UNION para mostrar los músicos que nacieron en un lugar y en otro)




```
SELECT m_name FROM musician WHERE born_in = 1
UNION
SELECT m_name FROM musician WHERE born_in = 2
```

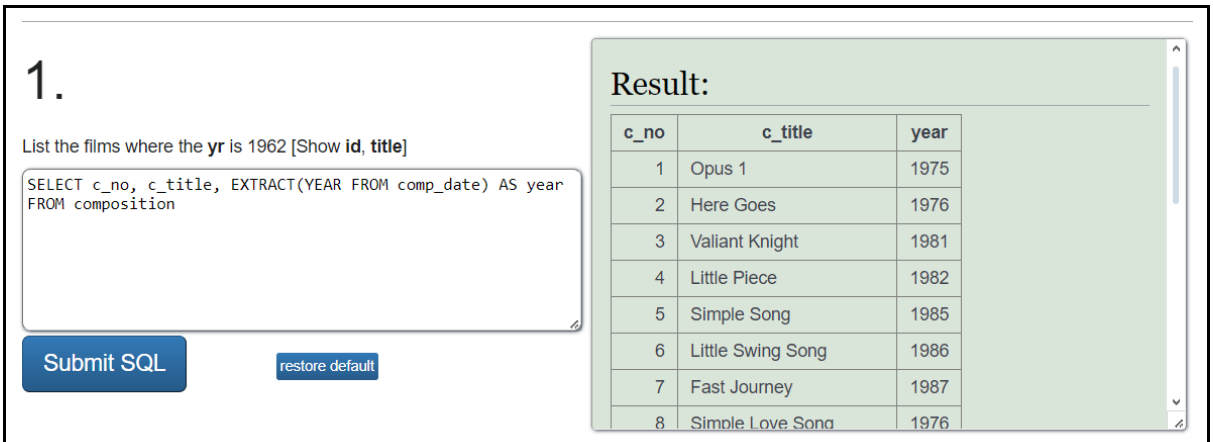
- ¿Qué músicos nacieron en Glasgow y Chicago?

(UNION ALL para unir los músicos que nacieron en un lugar y en otro, no importa si se repiten)



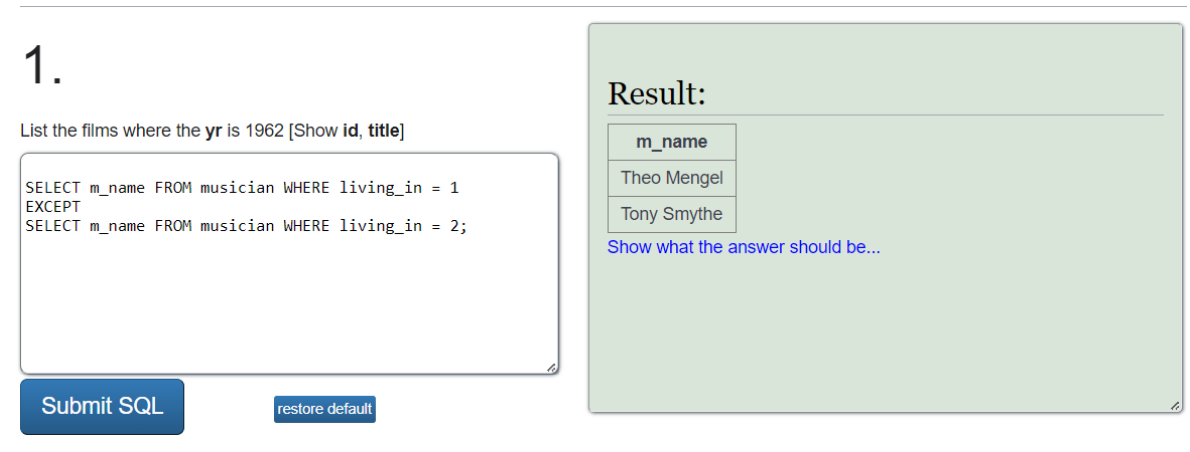
SELECT m_name FROM musician WHERE born_in = 6
UNION ALL
SELECT m_name FROM musician WHERE born_in = 8

-¿En qué año salieron las canciones?
(EXTRACT para sacar solo el valor del año de la fecha en la que fue lanzada cada una de las canciones)



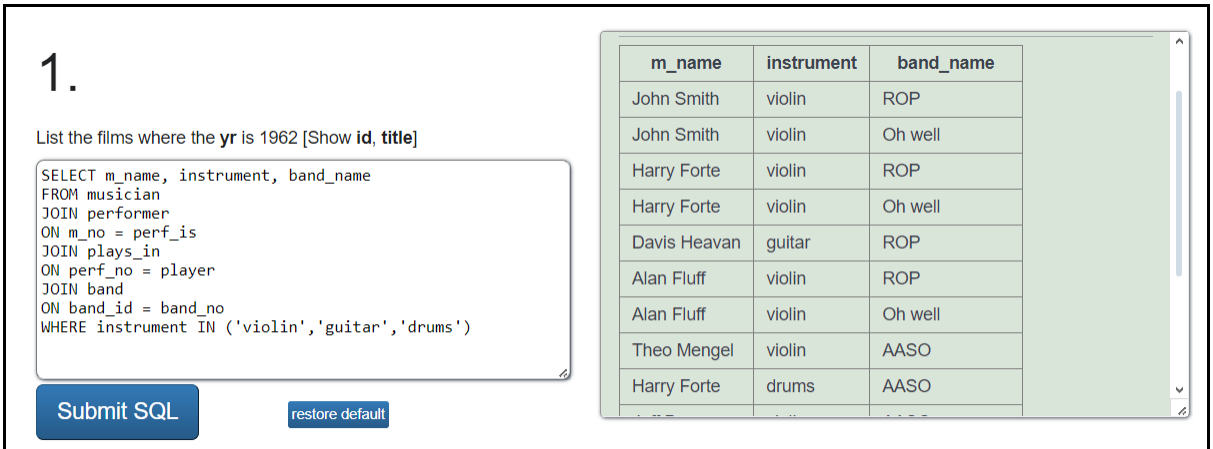
SELECT c_no, c_title, EXTRACT(YEAR FROM comp_date) AS year
FROM composition

-¿Qué músicos viven en Manchester pero no viven en Edinburg?
(EXCEPT para mostrar lo que hay en un conjunto pero no en el otro)




```
SELECT m_name FROM musician WHERE living_in = 1
EXCEPT
SELECT m_name FROM musician WHERE living_in = 2
```

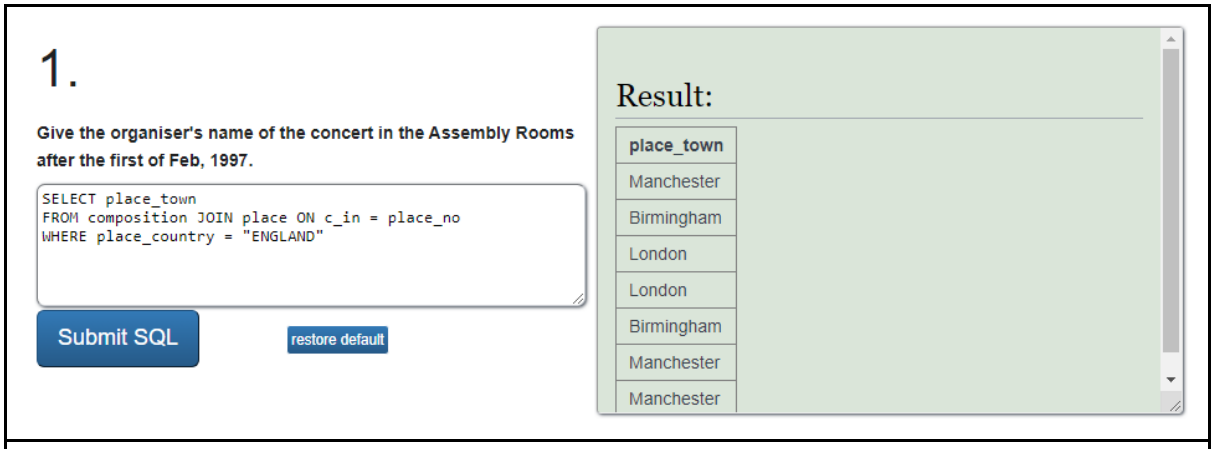
-¿Cuáles son los músicos que pertenecen al menos a una banda en donde tocan violín, guitarra o batería?
(IN para que en la columna 'instrument' solo me regrese los datos en donde el músico toque violín, guitarra o batería)




```
SELECT m_name, instrument, band_name
FROM musician
JOIN performer
ON m_no = perf_is
JOIN plays_in
ON perf_no = player
JOIN band
ON band_id = band_no
WHERE instrument IN ('violin','guitar','drums')
```

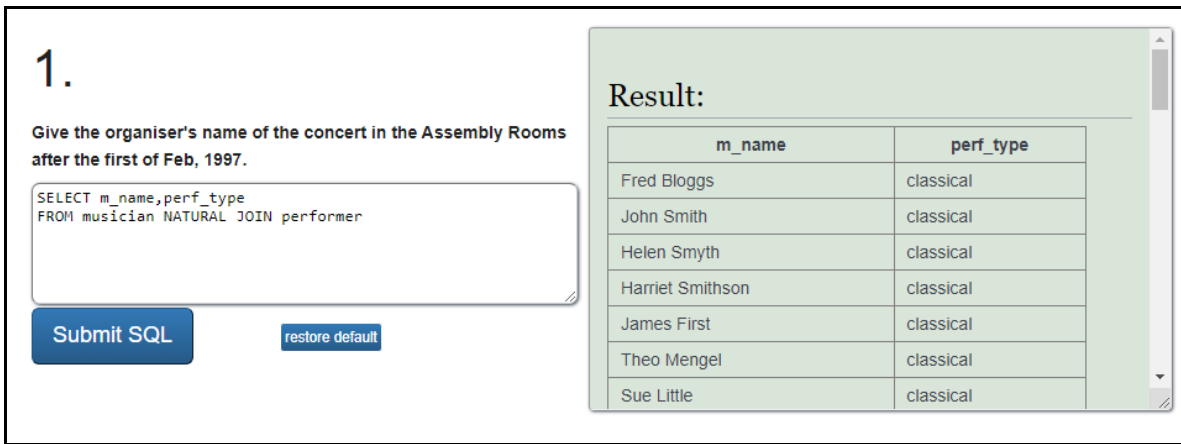
-6 consultas: una para cada operador de junta

¿Cómo puedo obtener los nombres de las ciudades donde se han interpretado composiciones, pero solo si estas ciudades están en Inglaterra?
JOIN ON (Une dos tablas a través de una columna que tengan en común**)**



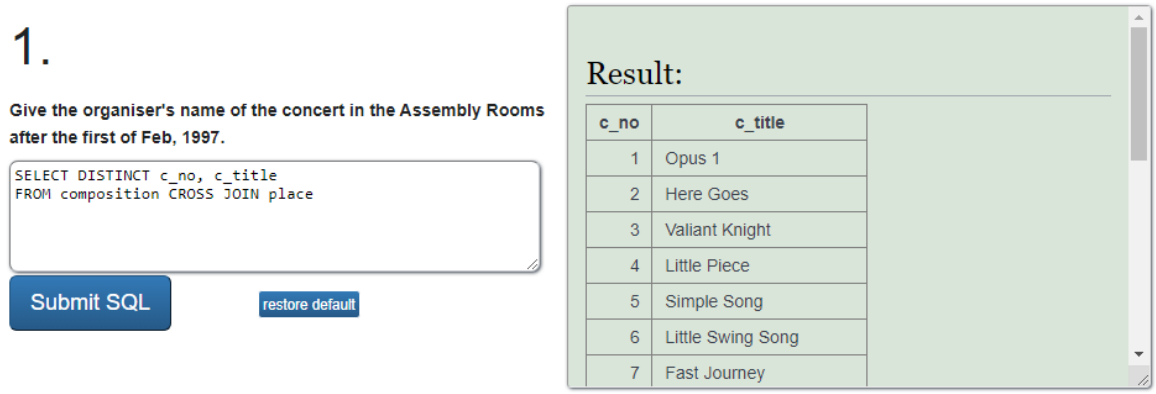

```
SELECT place_town
FROM composition JOIN place ON c_in = place_no
WHERE place_country = "ENGLAND"
```

¿Cómo puedo obtener los nombres de los músicos y el tipo de interpretación que realizan, utilizando la relación entre la tabla de músicos y la tabla de intérpretes?
NATURAL JOIN(UNE DOS TABLAS, LAS PEGA Y YA, A MENOS QUE EN CADA UNA HAYA UN NOMBRE EN COMÚN, SIENDO ASI QUE ELIMINA LAS COLUMNAS REPETIDAS**). NO NECESITA LA PALABRA ON PARA FUNCIONAR**



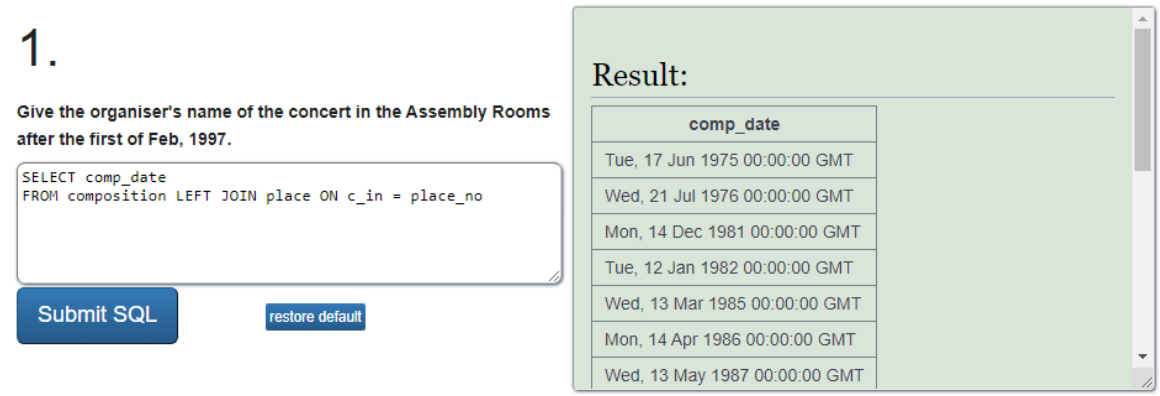

```
SELECT m_name,perf_type
FROM musician NATURAL JOIN performer
```

¿Cómo puedo obtener una lista distinta de las composiciones (número y título) junto con todas las posibles combinaciones de lugares?
CROSS JOIN(Hace un producto cartesiano entre las filas de las tablas**)**



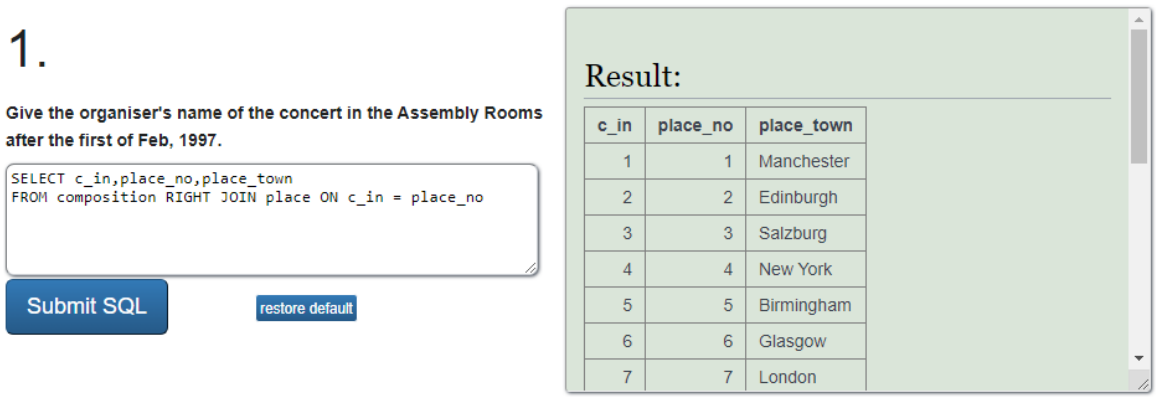

```
SELECT DISTINCT c_no, c_title
FROM composition CROSS JOIN place
```

¿Cómo puedo obtener las fechas de las composiciones, asegurándome de incluir todas las composiciones incluso si no tienen un lugar asociado?
LEFT JOIN(Une la tabla principal con la intersección entre la tabla 1 y la tabla 2.**)**



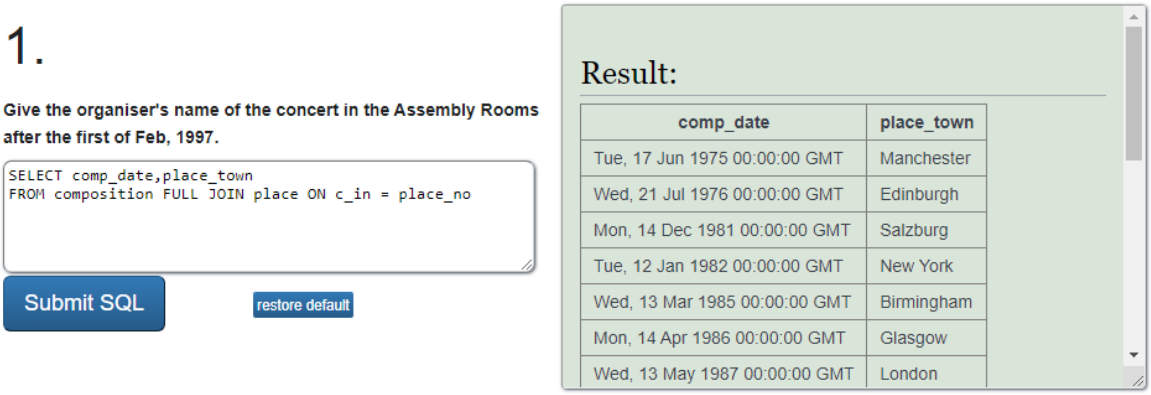

```
SELECT comp_date
FROM composition LEFT JOIN place ON c_in = place_no
```

¿Cómo puedo obtener la información de los lugares (número y ciudad), incluyendo aquellos que no están asociados con ninguna composición?
RIGHT JOIN(Une la tabla principal con la intersección entre la tabla 2 y la tabla 1.**)**




```
SELECT c_in,place_no,place_town
FROM composition RIGHT JOIN place ON c_in = place_no
```

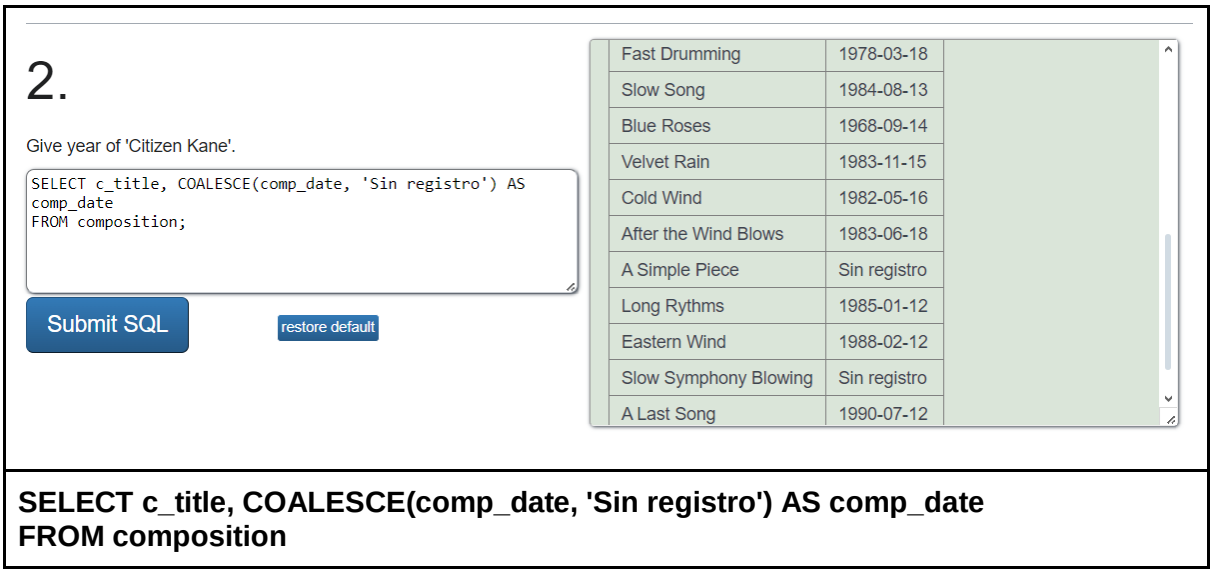
¿Cómo puedo obtener las fechas de las composiciones y las ciudades asociadas, incluyendo tanto las composiciones sin lugar como los lugares sin composiciones?
FULL JOIN (Es la RIGHT JOIN y la LEFT JOIN en una misma función.**)**




```
SELECT comp_date,place_town
FROM composition FULL JOIN place ON c_in = place_no
```

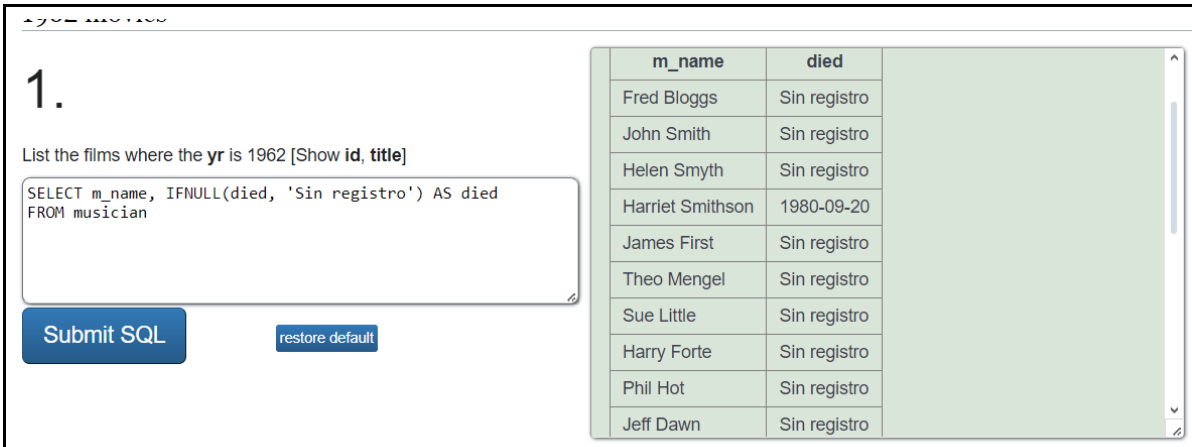
-2 consultas: una para cada operador de desconocido

-¿De cuáles canciones no se tienen conocimiento de su fecha de lanzamiento? (COALESCE para que en la columna 'comp_date' los espacios que sean NULL los reemplace por 'Sin registro'. Para COALESCE se pueden usar varias tablas para intercambiar el valor NULL por algo, pero en la base de datos 'Musician' eran pocas los que tenían valores NULL)



SELECT c_title, COALESCE(comp_date, 'Sin registro') AS comp_date
FROM composition

-¿De cuáles músicos se tienen conocimiento que no han fallecido? (IFNULL para que en la columna 'died' los espacios que sean NULL los reemplace por 'Sin registro'. Para IFNULL (que tiene una función similar al COALESCE) solo se puede tomar una columna que tengan valores NULL para intercambiarlos)

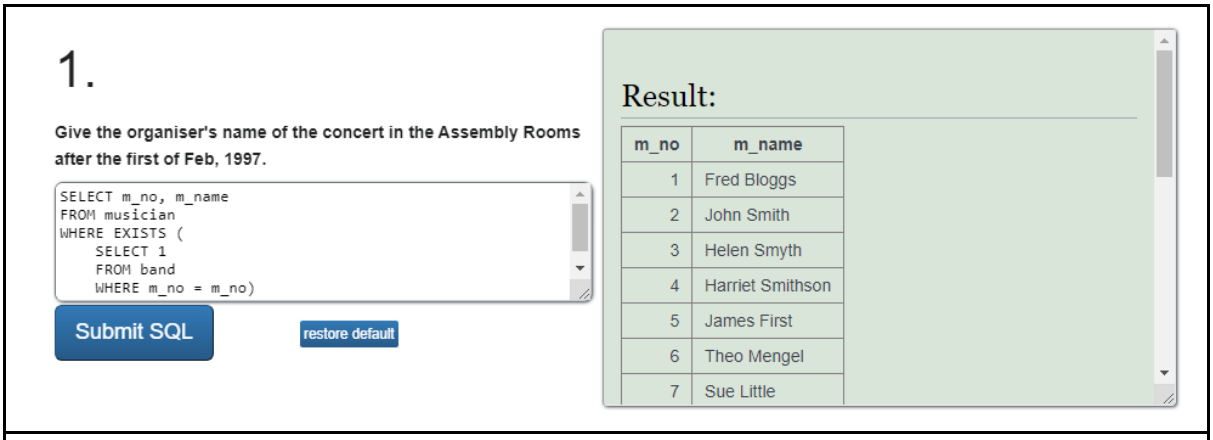


SELECT m_name, IFNULL(died, 'Sin registro') AS died
FROM musician

-3 consultas: una para cada uno de los tipos de operadores lógicos

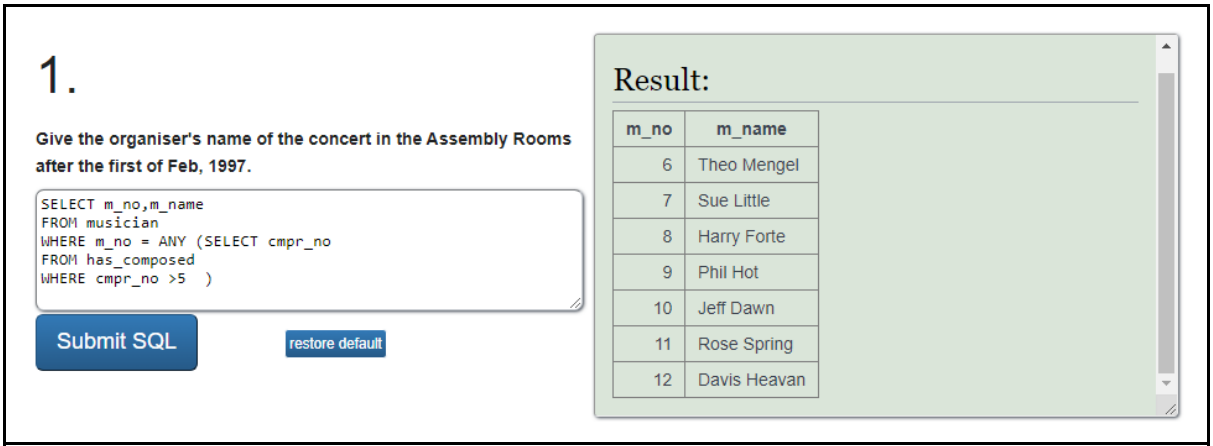
¿Cómo puedo obtener los números y nombres de los músicos que están asociados a alguna banda, utilizando una subconsulta con EXISTS?

EXISTS(Si por el menos se recibe una fila, o columna entonces se cumple lo anterior)



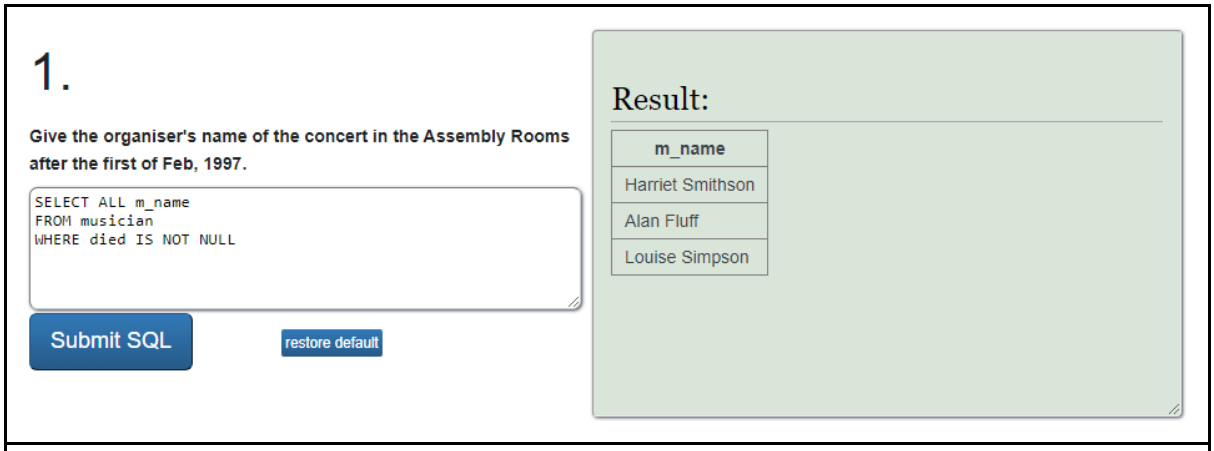

```
SELECT m_no, m_name
FROM musician
WHERE EXISTS (
   SELECT 1
   FROM band
   WHERE m_no = m_no)
```

¿Cómo puedo obtener los números y nombres de los músicos que han compuesto algo y cuyo número de compositor es mayor que 5?
ANY(De acuerdo a la subquery, se comparan veces con la parte de WHERE principal**).**




```
SELECT m_no,m_name
FROM musician
WHERE m_no = ANY (SELECT cmpr_no
FROM has_composed
WHERE cmpr_no >5)
```

¿Cómo puedo obtener los nombres de los músicos que han fallecido, asegurándome de incluir todos los nombres en caso de coincidencia?
ALL(Tiene una función parecida a ANY, pero en este caso una condición por defecto es que debe ser para todas las filas de la tabla**)**




```
SELECT ALL m_name
FROM musician
WHERE died IS NOT NULL
```

-1 consulta: para el operador CASE

-¿Cuál es la nacionalidad de los músicos?

(CASE para crear condicionales dentro del query. En este caso, cuando en la columna 'place_country' el país pertenece al continente europeo, se le asigna en una columna adicional a cada músico el 'European'. Para cuando 'place_country' no está en el continente europeo (en este caso solo hay un país más que es Estados Unidos), queda como 'American')


```
SELECT m_name, place_country,
      CASE
         WHEN place_country IN ('England', 'Scotland', 'Austria', 'Netherlands') THEN 'European'
         ELSE 'American'
      END AS region
FROM musician
LEFT JOIN place ON musician.born_in = place.place_no
```