MODELOS Y BASES DE DATOS

SQL Básico

2019-02

Guía autoestudio 2/6

INVESTIGACIÓN

NULL

¿Qué significa?

En SQL, NULL no es un valor. Es un *estado* que indica que el valor de ese item es desconocido o no existente. No es cero o blanco o una "cadena vacía" y no se comporta como ninguno de esos valores.

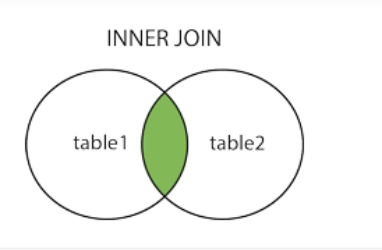
¿Resultado de operarlo con los diferentes tipos de operadores: aritméticos, lógicos y de comparación?

El valor NULL es un valor especial, y por tanto, no se puede comparar con los operadores aritméticos normales (=, >, <, <>), y en su lugar debemos utilizar los operadores IS y IS NOT.

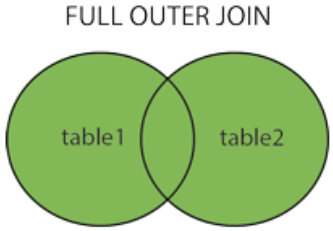
JUNTA

¿ Cuáles son las diferencias entre junta interna y externa?

La palabra clave INNER JOIN selecciona registros que tienen valores coincidentes en ambas tablas.



The FULL OUTER JOIN keyword return all records when there is a match in left (table1) or right (table2) table records.



¿ Qué opciones se tienen para la junta interna ?

¿ Qué opciones se tienen para la junta externa?

Opciones para la Junta interna:

- JOIN

- NATURAL JOIN

- CROSS JOIN

-INNER JOIN

Opciones para la Junta externa:

-LEFT JOIN

- RIGHT JOIN

- FULL JOIN

PRACTICA

Realicen los ejercicios propuestos en los siguientes tutoriales.

6.JOIN

1.Modify it to show the matchid and player name for all goals scored by Germany. To identify German players, check for: teamid = 'GER'

```
SELECT matchid,player

FROM goal
WHERE TEAMID LIKE 'GER'
```

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

```
SELECT id,stadium,team1,team2
FROM game JOIN goal
WHERE id=matchid AND player LIKE '%Bender
```

3.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='GER'
```

4.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%'
```

5.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
```

6.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'
```

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(id=matchid)
WHERE stadium='National Stadium, Warsaw'
```

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

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

9.Show teamname and the total number of goals scored.

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

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

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

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

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

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

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

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.

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

7.More Join Operations

1.List the films where the yr is 1962 [Show id, title]

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

2.Give year of 'Citizen Kane'.

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

3.List all of the Star Trek movies, include the id, title and yr (all of these movies include the words Star Trek in the title). Order results by year.

```
SELECT id,title,yr
FROM movie
WHERE title LIKE '%Star Trek%'
ORDER BY yr
```

4.What id number does the actor 'Glenn Close' have?

```
SELECT id
FROM actor
WHERE name='Glenn Close'
```

5.What is the id of the film 'Casablanca'

```
SELECT id
FROM movie
WHERE title='Casablanca'
```

6.Obtain the cast list for 'Casablanca'.

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

7.Obtain the cast list for the film 'Alien'

```
SELECT name
FROM casting JOIN actor ON id=actorid
WHERE movieid=(SELECT id
FROM movie
WHERE title='Alien')
```

8.List the films in which 'Harrison Ford' has appeared

```
SELECT title
FROM movie JOIN (SELECT movieid,actorid
FROM actor JOIN casting ON actorid=id
WHERE NAME='Harrison Ford') AS TABLA1 ON movieid=id
```

9.SELECT title

```
FROM movie JOIN (SELECT movieid,actorid
FROM actor JOIN casting ON actorid=id
WHERE NAME='Harrison Ford'AND ord!=1)as TABLA1 ON movieid=id
```

10.List the films together with the leading star for all 1962 films.

```
SELECT title,name
FROM movie JOIN (SELECT movieid,actorid,name
FROM actor JOIN casting ON actorid=id
WHERE ord=1)as TABLA1 ON movieid=id
WHERE yr=1962
```

11.Which were the busiest years for 'Rock Hudson', show the year and the number of movies he made each year for any year in which he made more than 2 movies.

```
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
```

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

```
SELECT title,name
FROM movie JOIN casting ON movie.id=movieid
JOIN actor ON actor.id=actorid
WHERE movie.id IN (SELECT movieid
FROM actor JOIN casting ON id=actorid
WHERE name='Julie Andrews') and ord=1
```

8.Null

1.List the teachers who have NULL for their department.

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

2.Note the INNER JOIN misses the teachers with no department and the departments with no teacher.

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

3.Use a different JOIN so that all teachers are listed.

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

4.Use a different JOIN so that all departments are listed.

```
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 teacher.name,(CASE WHEN mobile IS NOT NULL THEN COALESCE(mobile)
ELSE '07986 444 2266' END) num_teacher
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 teacher.name,(CASE WHEN dept.name IS NULL THEN 'None' ELSE dept.name
END)depart
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), SUM(num)
FROM (SELECT name,COUNT(mobile) AS num
FROM teacher
GROUP BY name)AS tabla1
```

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
```

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 OR 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.

8+Numeric Examples

1.Show the percentage who STRONGLY AGREE

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

2.Show the institution and subject where the score is at least 100 for question 15.

```
SELECT
round(sum(A_STRONGLY_AGREE)*100/(sum(A_NA)+sum(A_STRONGLY_DISAGREE)
+sum(A_DISAGREE)+sum(A_NEUTRAL)+sum(A_AGREE)+sum(A_STRONGLY_AGRE))
)
FROM nss  WHERE question='Q01'
AND institution='Edinburgh Napier University'
AND subject='(8) Computer Science'
```

3.Show the institution and score where the score for '(8) Computer Science' is less than 50 for question 'Q15'

```
SELECT institution,subject
FROM nss
WHERE score >= 100 and question = 'Q15'
```

4.Show the subject and total number of students who responded to question 22 for each of the subjects '(8) Computer Science' and '(H) Creative Arts and Design'.

```
SELECT institution,score
FROM nss
WHERE question='Q15'
AND subject='(8) Computer Science'
AND score < 50
```

5.Show the subject and total number of students who A_STRONGLY_AGREE to question 22 for each of the subjects '(8) Computer Science' and '(H) Creative Arts and Design'.

```
(SELECT subject,SUM(response)
FROM nss
WHERE subject = '(8) Computer Science'
AND question = 'q22'
GROUP BY subject)

UNION

(SELECT subject,SUM(response)
FROM nss
WHERE subject = '(H) Creative Arts and Design'
AND question = 'q22'
GROUP BY subject)
```

6.Show the percentage of students who A_STRONGLY_AGREE to question 22 for the subject '(8) Computer Science' show the same figure for the subject '(H) Creative Arts and Design'.

```
(SELECT subject,SUM((A_STRONGLY_AGREE*response)/100)
FROM nss
WHERE subject = '(8) Computer Science'
AND question = 'q22'
GROUP BY subject)

UNION

(SELECT subject,SUM((A_STRONGLY_AGREE*response)/100)
FROM nss
WHERE subject = '(H) Creative Arts and Design'
AND question = 'q22'
GROUP BY subject)
```

7.Show the average scores for question 'Q22' for each institution that include 'Manchester' in the name.

```
(SELECT subject,
round(sum((A_STRONGLY_AGREE*response)/100)*100/(sum((A_NA*response)/100)+su
m((A_STRONGLY_DISAGREE*response)/100)+
sum((A_DISAGREE*response)/100)+sum((A_NEUTRAL*response)/100)+sum((A_AGREE
*response)/100)+sum((A_STRONGLY_AGREE*response)/100)))
FROM nss  WHERE question='Q22'
AND subject='(8) Computer Science'
GROUP BY subject)

UNION

(SELECT
subject,round(sum((A_STRONGLY_AGREE*response)/100)*100/(sum((A_NA*response)/
100)+sum((A_STRONGLY_DISAGREE*response)/100)+
sum((A_DISAGREE*response)/100)+sum((A_NEUTRAL*response)/100)+sum((A_AGREE
*response)/100)+sum((A_STRONGLY_AGREE*response)/100)))
FROM nss  WHERE question='Q22'
AND subject='(H) Creative Arts and Design'
GROUP BY subject)
```

9 Self join

1.How many stops are in the database.

```
SELECT count(id) as suma
FROM stops
```

2.Find the id value for the stop 'Craiglockhart'

```
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,route
WHERE company = 'LRT' and num = 4 and id = stop;
```

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(*) as cont
FROM route WHERE stop=149 OR stop=53
GROUP BY company, num
HAVING cont > 1
```

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
```

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'
```

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)
JOIN stops stopa ON (a.stop=stopa.id)
JOIN stops stopb ON (b.stop=stopb.id)
WHERE stopa.name='Haymarket' and stopb.name = 'Leith'
```

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'
```

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 bstop.name, a.company, a.num FROM
route AS a JOIN route AS b ON (a.company = b.company AND a.num = b.num)
JOIN stops AS astop ON (a.stop = astop.id)
JOIN stops AS bstop ON (b.stop = bstop.id)
WHERE astop.name = 'Craiglockhart'
```

10. Tutorial Quizzes

Join Quiz

1. You want to find the stadium where player 'Dimitris Salpingidis' scored. Select the JOIN condition to use:

```
eteam JOIN game ON (id=team1)
```

```
eteam JOIN game ON (id=team2)
```

```
eteam JOIN goal ON (teamid=id)
```

```
game  JOIN goal ON (id=matchid)
```

```
game  JOIN goal ON (team1=teamid OR team2=teamid)
```

2. You JOIN the tables **goal** and **eteam** in an SQL statement. Indicate the list of column names that may be used in the SELECT line:

```
gtime, mdate, stadium, matchid
```

```
mdate, stadium, id
```

```
matchid, teamid, player, gtime, id, teamname, coach
```

```
matchid, teamid, player, gtime, mdate, stadium, team1
```

```
stadium, team1, team2
```

3. Select the code which shows players, their team and the amount of goals they scored against Greece(GRE).

```
SELECT player, teamid, COUNT(*)
  FROM game JOIN goal ON matchid = id
 WHERE (team1 = "GRE" OR team2 = "GRE")
   AND teamid != 'GRE'
 GROUP BY player, teamid
```

```
SELECT player, teamid, COUNT(*)
  FROM game JOIN goal ON matchid = id
 WHERE (team1 = "GRE") AND teamid != 'GRE'
 GROUP BY player, teamid
```

```
SELECT player, teamid, COUNT(*)
  FROM game JOIN goal ON matchid = id
 WHERE (team1 = "POL" OR team2 = "POL")
   AND teamid != 'POL'
 GROUP BY player, teamid
```

```
SELECT player, teamid, COUNT(*)
  FROM game JOIN goal WITH matchid = id
 WHERE (team1 = "GRE" OR team2 = "GRE")
   AND teamid != 'GRE'
```

4. Select the result that would be obtained from this code:

```
SELECT DISTINCT teamid, mdate
  FROM goal JOIN game on (matchid=id)
 WHERE mdate = '9 June 2012'
```

DEN	9 June 2012
GER	9 June 2012

DEN
GER

DEN	9 June 2012
DEN	9 June 2012
POL	9 June 2012
RUS	9 June 2012

GRE
CZE
POL

5. Select the code which would show the player and their team for those who have scored against Poland(POL) in National Stadium, Warsaw.

```
SELECT DISTINCT player, teamid
FROM game JOIN goal ON matchid = id
WHERE stadium = 'National Stadium, Warsaw'
AND (team1 = 'GER' OR team2 = 'GER')
  AND teamid != 'GER'
```

```
SELECT DISTINCT player, teamid
 FROM game JOIN goal ON matchid = id
WHERE stadium = 'National Stadium, Warsaw'
AND (team1 = 'POL' OR team2 = 'POL')
  AND teamid != 'POL'
```

```
SELECT DISTINCT player, teamid
  FROM game JOIN goal ON matchid = id
 WHERE stadium = 'National Stadium, Warsaw' AND teamid != 'POL'
```

6. Select the code which shows the player, their team and the time they scored, for players who have played in Stadion Miejski (Wroclaw) but not against Italy(ITA).

```
SELECT DISTINCT player, teamid, gtime
  FROM game JOIN goal ON matchid = id
 WHERE stadium = 'National Stadium, Warsaw'
   AND (( teamid = team2 AND team1 != 'ITA') OR ( teamid = team1 AND team2 != 'ITA'))
```

```
SELECT DISTINCT player, teamid, gtime
  FROM game JOIN goal ON matchid = id
 WHERE stadium = 'Stadion Miejski (Wroclaw)'
   AND (( teamid = team2 AND team1 != 'ESP') OR ( teamid = team1 AND team2 != 'ESP'))
```

```
SELECT DISTINCT player, teamid, gtime
  FROM game JOIN goal ON matchid = id
 WHERE stadium = 'Stadion Miejski (Wroclaw)'
   AND (( teamid = team2 AND team1 != 'ITA') OR ( teamid = team1 AND team2 != 'ITA'))
```

7. Select the result that would be obtained from this code:

```
SELECT teamname, COUNT(*)
  FROM eteam JOIN goal ON teamid = id
 GROUP BY teamname
HAVING COUNT(*) < 3
```

2
2
1
2

Netherlands	2
Poland	2
Republic of Ireland	1
Ukraine	2

Netherlands
Poland
Republic of Ireland
Ukraine

JOIN Quiz - part 2

1. Select the statement which lists the unfortunate directors of the movies which have caused financial loses (gross < budget)

```
SELECT JOIN(name FROM actor, movie
       ON actor.id:director WHERE gross < budget)
 GROUP BY name
```

```
SELECT name
 FROM actor INNER JOIN movie BY actor.id = director
 HAVING gross < budget
```

```
SELECT name
  FROM actor INNER JOIN movie ON actor.id = director
 WHERE gross < budget
```

2. Select the correct example of JOINing three tables

```
SELECT *
  FROM actor JOIN casting BY actor.id = actorid
 JOIN movie BY movie.id = movieid
```

```
SELECT *
  FROM actor JOIN casting ON actor.id = actorid
   AND JOIN movie ON movie.id = movieid
```

```
SELECT *
  FROM actor JOIN casting
  JOIN movie ON actor.id = actorid
   AND movie.id = movieid
```

```
SELECT *
  FROM actor JOIN casting ON actor.id = actorid
   AND movie ON movie.id = movieid
```

```
SELECT *
  FROM actor JOIN casting ON actor.id = actorid
  JOIN movie ON movie.id = movieid
```

3. Select the statement that shows the list of actors called 'John' by order of number of movies in which they acted

```
SELECT name, COUNT(movieid)
  FROM actor JOIN casting ON actorid=actor.id
 WHERE name IN 'John %'
 GROUP BY name ORDER BY 2
```

```
SELECT name, COUNT(movieid)
  FROM actor JOIN casting ON actorid=actor.id
 WHERE name LIKE 'J%'
 GROUP BY name ORDER BY 2 DESC
```

```
SELECT name, COUNT(movieid)
  FROM casting JOIN actor ON actorid=actor.id
 WHERE name LIKE 'John %'
 GROUP BY name ORDER BY 2 DESC
```

```
SELECT name, COUNT(movieid)
  FROM casting JOIN actor
 WHERE (actorid ON actor.id)
```

4. Select the result that would be obtained from the following code:

```
SELECT title
  FROM movie JOIN casting ON (movieid=movie.id)
             JOIN actor   ON (actorid=actor.id)
 WHERE name='Paul Hogan' AND ord = 1
```

Table-A

"Crocodile" Dundee	1
Crocodile Dundee in Los Angeles	1
Flipper	1
Lightning Jack	1

Table-B

"Crocodile" Dundee
Crocodile Dundee in Los Angeles
Flipper
Lightning Jack

Table-C

"Crocodile" Dundee
Paul Hogan
1

5. Select the statement that lists all the actors that starred in movies directed by Ridley Scott who has id 351

```
SELECT name
  FROM movie JOIN casting
    AND actor ON movie.id = movieid
    AND actor.id = actorid
 WHERE ord = 1
  AND actor = 351
```

```
SELECT name
  FROM movie JOIN casting
  JOIN actor ON movie.id = movieid
    OR actor.id = actorid
 WHERE ord = 1 AND director = 351
```

```
SELECT name
  FROM movie JOIN casting ON movie.id = movieid
  JOIN actor ON actor.id = actorid
 WHERE ord = 1 AND actorid = 351
```

```
SELECT name
  FROM movie JOIN casting ON movie.id = movieid
  JOIN actor ON actor.id = actorid
WHERE ord = 1 AND director = 351
```

6. There are two sensible ways to connect movie and actor. They are:

- link the director column in movies with the id column in actor
- join casting to itself

- link the actor column in movies with the primary key in actor
- connect the primary keys of movie and actor via the casting table

✓
- link the director column in movies with the primary key in actor
- connect the primary keys of movie and actor via the casting table

- link the director column in movies with the primary key in actor
- connect the primary keys of movie and casting via the actor table

- link the movie column in actor with the director column in actor
- connect movie and actor via the casting table

7. Select the result that would be obtained from the following code:

```
SELECT title, yr
  FROM movie, casting, actor
 WHERE name='Robert De Niro' AND movieid=movie.id AND actorid=actor.id AND ord = 3
```

Table-A

A Bronx Tale	1993	3
Bang the Drum Slowly	1973	3
Limitless	2011	3

Table-B

✓

A Bronx Tale	1993
Bang the Drum Slowly	1973
Limitless	2011

Table-C

A Bronx Tale	3
Bang the Drum Slowly	3
Limitless	3

Table-D

A Bronx Tale
Bang the Drum Slowly

Using Null Quiz

1. Select the code which uses an outer join correctly.

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

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

```
SELECT teacher.name, dept.name FROM teacher, dept JOIN WHERE(teacher.dept = dept.id)
```

```
SELECT teacher.name, dept.name FROM teacher OUTER JOIN dept ON dept.id
```

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

2. Select the correct statement that shows the name of department which employs Cutflower -

```
SELECT dept.name FROM teacher JOIN dept ON (dept.id = (SELECT dept FROM teacher WHERE name = 'Cutflower'))
```

```
SELECT dept.name FROM teacher JOIN dept ON (dept.id = teacher.dept) WHERE dept.id = (SELECT dept FROM teacher HAVING name = 'Cutflower')
```

```
SELECT dept.name FROM teacher JOIN dept ON (dept.id = teacher.dept) WHERE teacher.name = 'Cutflower'
```

```
SELECT dept.name FROM teacher JOIN dept WHERE dept.id = (SELECT dept FROM teacher WHERE name = 'Cutflower')
```

```
SELECT name FROM teacher JOIN dept ON (id = dept) WHERE id = (SELECT dept FROM teacher WHERE name = 'Cutflower')
```

3. Select out of following the code which uses a JOIN to show a list of all the departments and number of employed teachers

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

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

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

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

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

4. Using `SELECT name, dept, COALESCE(dept, 0) AS result FROM teacher` on `teacher` table will:

- display 0 in result column for all teachers
- display 0 in result column for all teachers without department
- do nothing - the statement is incorrect
- set dept value of all teachers to 0
- set dept value of all teachers without department to 0

5. Query:

```
SELECT name,
       CASE WHEN phone = 2752 THEN 'two'
            WHEN phone = 2753 THEN 'three'
            WHEN phone = 2754 THEN 'four'
            END AS digit
  FROM teacher
```

shows following 'digit':

- 'four' for Throd
- NULL for all teachers
- NULL for Shrivell
- 'two' for Cutflower
- 'two' for Deadyawn

6. Select the result that would be obtained from the following code:

```
SELECT name,
     CASE
      WHEN dept
       IN (1)
       THEN 'Computing'
      ELSE 'Other'
     END
 FROM teacher
```

Table-A

Shrivell	Computing
Throd	Computing
Splint	Computing
Spiregrain	Other
Cutflower	Other
Deadyawn	Other

Table-B

Shrivell	Computing
Throd	Computing
Splint	Computing
Spiregrain	Computing
Cutflower	Computing
Deadyawn	Computing

Self-Join Quiz

1. Select the code that would show it is possible to get from Craiglockhart to Haymarket

```
SELECT DISTINCT a.name, b.name
  FROM stops a JOIN route z IN a.id=z.stop
  JOIN route y ON y.num = z.num
  JOIN stops b IN y.stop=b.id
 WHERE a.name='Craiglockhart' AND b.name ='Haymarket'
```

```
SELECT DISTINCT a.name, b.name
  FROM stops a JOIN route z ON a.id=z.stop
  JOIN route y JOIN stops b ON y.stop=b.id
 WHERE a.name='Craiglockhart' AND b.name ='Haymarket'
```

```
SELECT DISTINCT a.name, b.name
  FROM stops a JOIN route z ON a.id=z.stop
  JOIN route y ON y.num = z.num
  JOIN stops b ON y.stop=b.id
 WHERE a.name='Craiglockhart' AND b.name ='Haymarket'
```

2. Select the code that shows the stops that are on route num '2A' which can be reached with one bus from Haymarket?

```
SELECT S2.id, S2.name, R2.company, R2.num
  FROM stops S1, stops S2, route R1, route R2
 WHERE S1.name='Haymarket' AND S1.id=R1.stop
   AND R1.company=R2.company AND R1.num=R2.num
   AND R2.stop=S2.id AND R1.num='2A'
```

```
SELECT S2.id, S2.name, R2.company, R2.num
  FROM stops S1, stops S2, route R1, route R2
 WHERE S1.name='Craiglockhart' AND S1.id=R1.stop
   AND R1.company=R2.company AND R1.num=R2.num
   AND R2.stop=S2.id AND R2.num='2A'
```

```
SELECT S2.id, S2.name, R2.company, R2.num
  FROM stops S1, stops S2, route R1, route R2
 WHERE S1.name='Haymarket' AND S1.id=R1.stop
   AND R1.company=R2.company AND R1.num=R2.num
   AND R2.stop=S2.id
```

```
SELECT S2.id, S2.name, R2.company, R2.num
  FROM stops S1, stops S2, route R1, route R2
 WHERE S1.name='Haymarket' AND S1.id=R1.stop
   AND R1.company=R2.company AND R1.num=R2.num
   AND R2.stop=S2.id AND R2.num='2'
```

```
SELECT S2.id, S2.name, R2.company, R2.num
  FROM stops S1, stops S2, route R1, route R2
 WHERE S1.name='Haymarket' AND S1.id=R1.stop
   AND R1.company=R2.company AND R1.num=R2.num
   AND R2.stop=S2.id AND R2.num='2A'
```

3. Select the code that shows the services available from 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)
```

```
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='Sighthill'
```

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

C. Propongan preguntas que cumplan los siguientes requerimientos.

1) **Consultas con operadores de conjuntos**
   Muestre la cantidad de tipos de musica que hay

```
SELECT COUNT(perf_type) AS TIPOS_DE_MUSICA
FROM(SELECT perf_type
FROM performer

UNION

SELECT comp_type
FROM composer
UNION

SELECT band_type
FROM band) AS tabla1
```

   Liste los nombres de los interpretes y de las bandas con su respectivo tipo de musica

```
SELECT m_name,perf_type
FROM(SELECT m_name,perf_type
FROM musician JOIN performer ON perf_is=m_no) AS tabla1
UNION ALL

SELECT band_name,band_type
FROM band
```

   Liste las ciudades donde habra conciertos y la semana del año en la cual se hara este

```
SELECT place_town,EXTRACT(WEEK FROM con_date)AS concert_week
FROM place JOIN concert ON concert_in=place_no
```

   Liste el nombre de los musicos y la ciudad en donde viven

```
SELECT m_name,place_town
FROM musician JOIN  place ON m_no=place_no
WHERE living_in IN (SELECT place_no FROM place) SELECT m_name,place_town
FROM musician JOIN  place ON m_no=place_no
WHERE living_in IN (SELECT place_no FROM place)
```

2) **Muestre el nombre de las bandas y la cantidad de musicos que han estado en dicha banda**

```
SELECT band_name, COUNT(M_NAME) AS cant_musicos
FROM musician JOIN plays_in ON m_no=player JOIN band ON band_id=band_no
GROUP BY band_name
```

Muestre el nombre de la banda y la ciudad donde se situa dicha banda

```
SELECT band_name,place_town
FROM place JOIN band ON place_no=band_home
```

Muestre la ciudad y el numero de bandas que son de esa ciudad

```
SELECT place_town,COUNT(band_home)
FROM place LEFT JOIN band ON place_no=band_home
GROUP BY place_town
```

Muestre el nombre y la cantidad de instrumentos que toca cada musico

```
SELECT m_name,COUNT(instrument) AS cant_intrument
FROM musician LEFT JOIN performer ON m_no=perf_is
GROUP BY m_name
```

3) **Muestre el nombre y la fecha de los musicos que ya murieron**

```
SELECT m_name,COALESCE(died)
FROM musician
WHERE died IS NOT NULL
```

Muestre el nombre de las bandas y la fecha en que se construyo la banda, en caso de no haber fecha ponga ’no se sabe ’.

```
SELECT band_name,(CASE WHEN b_date IS NOT NULL THEN b_date ELSE 'NO
SE SABE'END)F_CREA_BAND
FROM band
```

4) **Muestre todos los nombres de los musicos si al menos una persona toque el violin**

```
SELECT m_name
FROM musician
WHERE EXISTS(SELECT m_name,instrument FROM musician JOIN performer ON
m_no=perf_is WHERE instrument='violin')
```

Muestre todos los nombres de los musicos si y solo si todos ellos tienen una fecha de fallecimiento

```
SELECT m_name
FROM musician
WHERE died=ALL(SELECT died FROM musician WHERE died IS NULL )
```

5) **Liste a todos los musicos muestre la fecha de su muerte, si no ha muerto ponga ’no ha muerto’**

```
SELECT m_name,(CASE WHEN died IS NOT NULL THEN COALESCE(died) ELSE
'NO HA MUERTO' END)fecha_muerte
FROM musician
```