**MODELOS Y BASES DE DATOS**

**SQL Básico**

**2019-2**

**Autoestudio 02**

**Autores: Ricardo Amaya Rivera & Javier Esteban López Peña MBDA – 1**

**INVESTIGACIÓN**

**A. NULL**

**1. ¿Qué significa?**

Un campo con un valor NULL es un campo sin valor. Un valor NULL es diferente de un valor cero o un campo que contiene espacios. Un campo con un valor NULL es uno que se ha dejado vacío durante la creación del registro

**2. ¿Resultado de operarlo con los diferentes tipos de operadores: aritméticos,**

**lógicos y de comparación?**

No es posible probar valores NULL con operadores de comparación, como =, <o <>. Tendremos que usar los operadores IS NULL y IS NOT NULL en su lugar.

**Operadores aritméticos**

Como Null no es un dato, sino un marcador para un valor ausente, usando operadores matemáticos Null da un resultado desconocido, que está representado por Null.

En el ejemplo siguiente, multiplicando 10 por resultados Null en Null:

10 \* NULL -- Result is NULL

**Operadores lógicos y de comparación**

Ya que Null no es un miembro de un dominio de datos, no se considera un «valor», sino más bien un marcador (o marcador de posición) indicando la ausencia de valor. Debido a esto, las comparaciones con Null nunca resulta verdadero o falso, pero siempre en un tercer resultado lógico, desconocido. El resultado lógico de la expresión siguiente, que compara el valor 10 a Null, es desconocido:

SELECT 10 = NULL -- Results in Unknown

Sin embargo, ciertas operaciones en Null pueden devolver valores si el valor ausente no es relevante para el resultado de la operación. Considere el siguiente ejemplo:

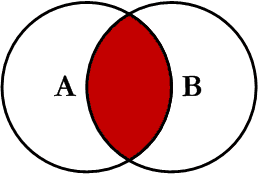
SELECT NULL OR TRUE -- Results in True

En este caso, el hecho de que el valor de la izquierda o es incognoscible es irrelevante, porque el resultado de la operación sería verdadero independientemente del valor de la izquierda.

**B. JUNTA**

**1. ¿Cuáles son las diferencias entre junta interna y externa?**

Junta interna



Esta es la Unión más simple, más entendida y es la más común. Esta consulta devolverá todos los registros coincidentes entre dos tablas A Y B. Esta Unión se escribe como sigue:

SQL

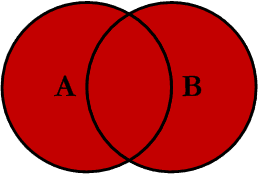
SELECT <select\_list>

FROM Table\_A A

INNER JOIN Table\_B B

ON A.Key = B.Key

Junta externa



Esta Unión también se puede referir como una Unión exterior completa o una Unión completa. Esta consulta devolverá todos los registros de ambas tablas, uniendo los registros de la tabla izquierda (tabla A) que coinciden con los registros de la tabla derecha (tabla B). Esta Unión se escribe como sigue:

SQL

SELECT <select\_list>

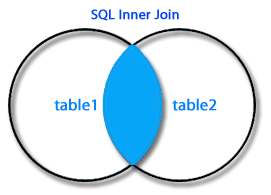
FROM Table\_A A

FULL OUTER JOIN Table\_B B

ON A.Key = B.Key

**2. ¿Qué opciones se tienen para la junta interna?**

**INNER JOIN O JOIN:** Devuelve todas las filas cuando hay al menos una coincidencia en ambas tablas.

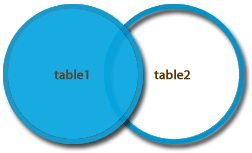


**JOIN NATURAL:** Es un tipo de JOIN estructurado de tal manera que, columnas con el mismo nombre de tablas asociadas aparecen sólo una vez. Se diferencia del INNER JOIN por el número de columnas que devuelve.

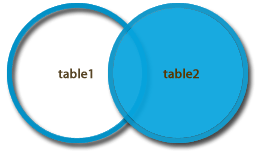
**CROSS JOIN:** devuelve el *producto cartesiano* de filas de tablas en el join. En otras palabras, va a producir filas que combinan cada fila de la primera tabla con cada fila de la segunda tabla.

**3. ¿Qué opciones se tienen para la junta externa?**

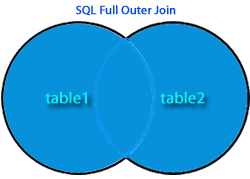
LEFT JOIN: Devuelve todas las filas de la tabla de la izquierda, y las filas coincidentes de la tabla de la derecha.



RIGHT JOIN: Devuelve todas las filas de la tabla de la derecha, y las filas coincidentes de la tabla de la izquierda.



FULL JOIN: Devuelve todas las filas de las dos tablas, la izquierda y la derecha. También se llama FULL OUTER JOIN.



**BIBLIOGRAFÍA**

Anónimo. Null(SQL). Obtenido de: https://en.wikipedia.org/wiki/Null\_(SQL)

Anónimo. Three-valued logic. Obtenido de: <https://en.wikipedia.org/wiki/Three-valued_logic>

[C.L. Moffatt](https://www.codeproject.com/script/Membership/View.aspx?mid=5909363),2009.Visual Representation of SQL Joins. Obtenido de: <https://www.codeproject.com/Articles/33052/Visual-Representation-of-SQL-Joins>

Lázaro,2018. Principales tipos de JOINS en SQL. Obtenido de: https://diego.com.es/principales-tipos-de-joins-en-sql

Stackoverflow. CRUZADAS VS INNER JOIN. Obtenido de: <https://stackoverflow.com/questions/17759687/cross-join-vs-inner-join-in-sql-server-2008>

**PRACTICA**

1. **REALICEN LOS EJERCICIOS PROPUESTOS EN LOS SIGUIENTES TUTORIALES. (56 PUNTOS)**

**JOIN**

**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 = 'GER'

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

SELECT id, stadium, team1, team2 FROM game

WHERE id = '1012'

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

SELECT player, teamid, stadium, mdate FROM goal

JOIN game ON game.id=goal.matchid

WHERE teamid = 'GER'

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

Top of Form

SELECT team1, team2, player FROM goal

JOIN game ON game.id=goal.matchid

WHERE player LIKE 'Mario%'

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

Top of Form

SELECT player, teamid, coach, gtime FROM goal

JOIN eteam ON goal.teamid=eteam.id

WHERE gtime<=10

**List the 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 game.team1=eteam.id

WHERE coach ='Fernando Santos'Top of Form

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

SELECT player FROM goal

JOIN game ON goal.matchid=game.id

WHERE stadium ='National Stadium, Warsaw'

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

SELECT DISTINCT player FROM game

JOIN goal ON game.id = goal.matchid

WHERE (teamid<>'GER' AND (team2='GER' OR team1='GER'))Top of Form

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

SELECT teamname, COUNT(teamid) FROM goal

JOIN eteam ON goal.teamid=eteam.id

GROUP BY teamname

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

SELECT stadium, COUNT(teamid) FROM goal

JOIN game ON goal.matchid=game.id

GROUP BY stadiumTop of Form

**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 goal.matchid = game.id

WHERE (team1 = 'POL' OR team2 = 'POL')

GROUP BY matchid, mdateTop of Form

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

Top of Form

SELECT matchid,mdate, COUNT(teamid) FROM game

JOIN goal ON goal.matchid = game.id

WHERE (teamid = 'GER')

GROUP BY matchid, mdateTop of Form

**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.**

NO SE PUDO REALIZAR

**More JOIN operations**

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

SELECT id, title FROM movie

WHERE yr=1962Top of Form

**Give year of 'Citizen Kane'.**

SELECT yr FROM movie

WHERE title = 'Citizen Kane'

**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

**What id number does the actor 'Glenn Close' have?**

SELECT id FROM actor

WHERE name='Glenn Close'

**What is the id of the film 'Casablanca'**

SELECT id FROM movie

WHERE title='Casablanca'Top of Form

**Obtain the cast list for 'Casablanca'.**

SELECT name FROM casting

JOIN actor ON casting.actorid=actor.id

WHERE movieid=11768

**Obtain the cast list for the film 'Alien'**

SELECT name FROM casting

JOIN actor ON casting.actorid=actor.id

JOIN movie ON casting.movieid=movie.id

WHERE title='Alien'Top of Form

**List the films in which 'Harrison Ford' has appeared**

SELECT title FROM casting

JOIN actor ON casting.actorid=actor.id

JOIN movie ON casting.movieid=movie.id

WHERE name='Harrison Ford'Top of Form

**List the films where 'Harrison Ford' has appeared - but not in the starring role. [Note: the ord field of casting gives the position of the actor. If ord=1 then this actor is in the starring role]**

SELECT title FROM casting

JOIN actor ON casting.actorid=actor.id

JOIN movie ON casting.movieid=movie.id

WHERE name='Harrison Ford' AND ord > 1

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

SELECT title, name FROM casting

JOIN actor ON casting.actorid=actor.id

JOIN movie ON casting.movieid=movie.id

WHERE ord=1 AND yr='1962'

**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.Top of Form**

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

**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 (movieid = movie.id AND ord=1)

JOIN actor ON actor.id = actorid

WHERE movie.id IN (SELECT movieid FROM casting

WHERE actorid IN (SELECT id FROM actor

WHERE name='Julie Andrews'))

**Obtain a list, in alphabetical order, of actors who've had at least 30 starring roles.**

SELECT name FROM movie

JOIN casting ON (movieid = movie.id AND ord=1)

JOIN actor ON actor.id = actorid

GROUP BY name

HAVING COUNT(\*) > 29

**List the films released in the year 1978 ordered by the number of actors in the cast, then by title.**

SELECT title, COUNT(actorid) FROM casting

JOIN actor ON casting.actorid=actor.id

JOIN movie ON casting.movieid=movie.id

WHERE yr=1978

GROUP BY title

ORDER BY COUNT(actorid) DESC, titleTop of Form

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

SELECT DISTINCT name FROM movie

JOIN casting ON movieid = movie.id

JOIN actor ON actor.id = actorid

WHERE movie.id IN (SELECT movieid FROM casting

WHERE actorid IN (SELECT id FROM actor

WHERE name='Art Garfunkel')) AND name <> 'Art Garfunkel'Top of Form

**NULL**

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

SELECT name FROM teacher

WHERE dept IS NULL

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

SELECT teacher.name, dept.name

 FROM teacher  JOIN dept

           ON (teacher.dept=dept.id)

**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)

**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)

**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

**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, COALESCE(dept.name,'None')

 FROM teacher  LEFT JOIN dept

           ON (teacher.dept=dept.id)

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

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

**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

**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 name,

(CASE WHEN dept= 1 OR dept = 2 THEN 'Sci' ELSE 'Art' END)

FROM teacher

**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 name,

(CASE WHEN dept= 1 OR dept = 2 THEN 'Sci'

WHEN dept = 3 THEN 'Art' ELSE 'None' END)

FROM teacher

**8+ Numeric Examples**

**Show the the percentage who STRONGLY AGREE**

SELECT A\_STRONGLY\_AGREE

  FROM nss

 WHERE question='Q01'

   AND institution='Edinburgh Napier University'

   AND subject='(8) Computer Science'

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

SELECT institution, subject

  FROM nss

 WHERE question='Q15'

   AND score >= 100

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

SELECT institution,score

  FROM nss

 WHERE question='Q15'

   AND subject='(8) Computer Science'

   AND score < 50

**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 subject, SUM(response)

  FROM nss

 WHERE question='Q22'

   AND (subject='(H) Creative Arts and Design'

   OR subject='(8) Computer Science')

GROUP BY subject

**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\*A\_STRONGLY\_AGREE/100)

  FROM nss

 WHERE question='Q22'

   AND (subject='(H) Creative Arts and Design'

   OR subject='(8) Computer Science')

GROUP BY subject

**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, ROUND(SUM(response \* A\_STRONGLY\_AGREE / 100) / SUM(response) \* 100, 0)

  FROM nss

 WHERE question='Q22'

   AND (subject='(H) Creative Arts and Design'

   OR subject='(8) Computer Science')

GROUP BY subject

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

SELECT institution,ROUND(SUM(response \* score) / SUM(response) )

  FROM nss

 WHERE question='Q22'

   AND (institution LIKE '%Manchester%')

GROUP BY institution

ORDER BY institution

**Show the institution, the total sample size and the number of computing students for institutions in Manchester for 'Q01**

SELECT institution, SUM(sample), (SELECT sample FROM nss as y

                                   WHERE subject='(8) Computer Science'

                                     AND x.institution = y.institution AND question='Q01') AS comp FROM nss  as x

WHERE question='Q01' AND (institution LIKE '%Manchester%')

GROUP BY institution

**Self join**

**How many stops are in the database.**

SELECT COUNT(\*) FROM stops

**Find the id value for the stop 'Craiglockhart'**

SELECT id FROM stops

WHERE name= 'Craiglockhart'

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

SELECT id, name FROM route

JOIN stops ON route.stop = stops.id

WHERE num = 4 AND company = 'LRT'

**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(\*)>1

**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

**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'

**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

**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'

**Proporcione una lista distinta de las paradas a las que se puede llegar desde 'Craiglockhart' tomando un autobús, incluido 'Craiglockhart', ofrecido por la compañía LRT. Incluya la compañía y el autobús no. de los servicios relevantes.**

SELECT DISTINCT stopb.name, 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 a.company = 'LRT'

**Encuentre las rutas que involucran dos autobuses que pueden ir desde Craiglockhart a Lochend .  
Mostrar el bus no. y compañía para el primer autobús, el nombre de la parada para el traslado  
y el autobús no. y compañía para el segundo autobús.**

NO SE PUDO REALIZAR.

1. **Tutorials: Learn SQL in stages - QUICES**

**10** **TUTORIAL QUIZZES**

Se encuentran en la carpeta llamada Tutorial Quizzes dentro del archivo adjunto

**C.**

**• 5 consultas: una para cada operador de conjuntos**

**1. Las avenidas donde se realizaron los conciertos en 15/06/97 y 95/02/21**

SELECT concert\_venue FROM concert

WHERE con\_date = '97/06/15'

UNION

SELECT concert\_venue FROM concert

WHERE con\_date = '95/02/21'

**2. Los nombres de los músicos que murieron en 97/05/15 y 80/09/20**

SELECT m\_name FROM musician

WHERE died = '97/05/15'

UNION ALL

SELECT m\_name FROM musician

WHERE died = '80/09/20'

**3. Los nombres de los músicos que nacieron en USA**

SELECT m\_name

FROM musician JOIN place ON (born\_in=place\_no)

WHERE place\_country='USA'

INTERSECT

SELECT m\_name

FROM musician JOIN place ON (born\_in=place\_no)

WHERE place\_town='New York' OR place\_town = 'Chicago'

**4. El año de nacimiento de Fred Bloggs**

SELECT EXTRACT (YEAR FROM (SELECT born FROM musician

WHERE m\_name = 'Fred Bloggs'))

**5. La fecha de Nacimiento junto al nombre de John Smith y James First**

SELECT m\_name, born

FROM musician

WHERE m\_name IN ('John Smith', 'James First')

**• 4 consultas: dos para junta interna y dos para junta externa**

1. **El lugar de Nacimiento de cada musico**

SELECT DISTINCT m\_name, place\_country FROM musician

NATURAL JOIN place

WHERE place\_no = born\_in

1. **El lugar donde vive cada musico**

SELECT DISTINCT m\_name, place\_country FROM musician

JOIN place ON place\_no = living\_in

1. **Los musicos que viven actualmente en Escocia**

SELECT DISTINCT m\_name, place\_country FROM musician

FULL JOIN place ON (place\_no = living\_in)

WHERE place\_country = 'Scotland'

1. **Los músicos que nacieron en Austria**

SELECT DISTINCT m\_name FROM musician

LEFT JOIN place ON (place\_no = born\_in)

WHERE place\_country = 'Austria'

**• 2 consultas: una para cada operador de desconocido**

1. **Muestre todos músicos vivos y las fecha de muerte de los fallecidos (Colocando ‘Alive’ cuando sea null)**

SELECT m\_name, (COALESCE(died,'Alive')) AS died FROM musician

1. **Los nombres y fechas de cumpleaños de las bandas (Colar ‘Unknown’ si se desconoce)**

SELECT band\_name, (ISNULL(b\_date,'Unknown')) AS b\_date FROM band

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

1. **Seleccione el nombre del musico que tenga z en el nombre**

SELECT m\_name

FROM musician

WHERE EXISTS (SELECT m\_no FROM musician WHERE m\_name like '%z%')

1. **Los musicos nacidos de 1948 en adelante ordenados por fechas**

SELECT m\_name, born FROM musician

WHERE born>= ALL(SELECT born FROM musician)

ORDER BY born

1. **Los músicos nacidos de 1965 hacia atrás ordenados por fechas**

SELECT m\_name, born FROM musician

WHERE born< ANY(SELECT born FROM musician)

ORDER BY born

**• 1 consulta: para el operador CASE**

1. **Muestre el nombre de los intérpretes distintos y si tocan o no jazz**

SELECT musician.m\_name, (case WHEN perf\_type = 'jazz' THEN 'yes!' ELSE 'no!' END) as PLAY\_JAZZ  
FROM musician JOIN performer ON (perf\_is=m\_no)