

MODELOS Y BASES DE DATOS

SQL Básico

2019-02

Guía autoestudio 2/6

OBJETIVOS

Desarrollar competencias básicas escribir consultas en SQL considerando el valor que representa lo desconocido, operaciones entre conjuntos y juntas explícitas.

SQL- Detalle

- Dar nuevos nombres a tablas AS
- El valor NULL (DESCONOCIDO)
- Consultas que implican operaciones de conjuntos
UNION, UNION ALL, INTERSECT, EXTRACT, IN
- Consultas que con junta explícita:
 - Junta interna: de equivalencia, natural, cruzada
JOIN, NATURAL JOIN, CROSS JOIN
 - Junta externa: tabla izquierda, tabla derecha, completa
LEFT JOIN, RIGHT JOIN, FULL JOIN
- Operadores
 - Desconocido : ISNULL, COALESCE
 - Lógicos : EXISTS, Comparación ANY, Comparación ALL,
 - Condicionales: CASE

ENTREGA

Publicar las respuestas en el espacio correspondiente en un archivo .zip , el nombre de este archivo debe ser la concatenación en orden alfabético de los primeros apellidos de cada uno de los miembros.

INVESTIGACIÓN

No olviden incluir la bibliografía.

A. NULL

1. ¿Qué significa?

En SQL, NULL no es un valor. Es un estado que indica que el valor de ese ítem es desconocido o no existente. No es cero o blanco o una "cadena vacía" y no se comporta como ninguno de esos valores. Pocas cosas en SQL ,llevan a tanta confusión como NULL, y será difícil de entender mientras no entiendas la siguiente simple.

El resultado al hacer uso del NULL con operadores aritméticos o lógicos es un error, pero muchas veces es usado con operadores de comparación para revisar si alguna fila de la tabla tiene valor nulo o no.

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

El resultado al hacer uso del NULL con operadores aritméticos o lógicos es un error, pero muchas veces es usado con operadores de comparación para revisar si alguna fila de la tabla tiene valor nulo o no.

En SQL , no es un valor es un estado que indica que el valor de ese ítem

B. JUNTA

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

En la junta interna cada registro de la tabla A es combinado con los correspondientes en la tabla B que satisfagan la condición que se especifique en la junta. Por otro lado, en la junta externa no se requiere que un registro en una tabla tenga un registro relacionado en la otra tabla. El registro es mantenido en la tabla combinada, aunque no exista el correspondiente en la otra tabla.

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

- JOIN
 - NATURAL JOIN
 - CROSS JOIN
3. ¿Qué opciones se tienen para la junta externa?
- LEFT JOIN
 - RIGHT JOIN
 - FULL JOIN

PRACTICA

Usando SQLzoo.net [<http://sqlzoo.net/>]

[En auto02.doc]

A. Realicen los ejercicios propuestos en los siguientes tutoriales. (56puntos) Utilice el motor My SQL 5.

[Escriban la sentencia en SQL en [auto02.doc](#) y ejecuten la sentencia SQL en sqlzoo . Si no lograron escribir alguna sentencia indiquen el punto de problema]

Tutorials: Learn SQL in stages

6 JOIN

1.



The first example shows the goal scored by a player with the last name 'Bender'. The `*` says to list all the columns in the table - a shorter way of saying `matchid, teamid, player, gtime`

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'
```

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
```

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 = 'GER'
```

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 game JOIN goal on (id=matchid)  
where player LIKE 'Mario%'
```

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 (id=teamid)  
WHERE gtime<=10
```

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 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 (goal.matchid = game.id)
where stadium = 'National Stadium, Warsaw'
```

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

9. 😊

Show teamname and the total number of goals scored.

COUNT and GROUP BY

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

10. 😊

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

```
select stadium, COUNT(gtime)
from game JOIN goal ON (id = matchid)
group by stadium
```

11. 😊

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

```
SELECT matchid, mdate, a FROM
(select matchid, count(teamid) as a
FROM game JOIN goal ON id= matchid
WHERE team1= 'POL' or team2= 'POL'
GROUP BY matchid) AS b
JOIN game ON id= matchid
```

12. 😊

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

```
SELECT matchid,mdate, a FROM
(SELECT matchid,count(teamid) AS a
FROM game JOIN goal ON Id= matchid
WHERE (team1= 'GER' or team2= 'GER') and teamid= 'GER'
GROUP BY matchid) AS b JOIN game ON id= matchid
```

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.

| 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,matchid,team1,team2 ORDER BY mdate,matchid, team1
```

7 More JOIN operations

In which we join actors to movies in the Movie Database.

1962 movies

1. 😊

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

```
SELECT ID,TITLE
FROM movie
WHERE yr=1962
```

Submit SQL

Restore default

2.



Give year of 'Citizen Kane'.

```
select yr
from movie
where title ='Citizen Kane'
```

Submit SQL

Restore default

Star Trek movies

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%'
```

Submit SQL

Restore default

id for actor Glenn Close

4.



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

```
select id
from actor
where name ='Glenn Close'
```

Submit SQL

Restore default

id for Casablanca

5.



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

```
select id
from movie
where title= 'Casablanca'
```

Submit SQL

Restore default

Cast list for Casablanca

6.



Obtain the cast list for 'Casablanca'.

what is a cast list?

Use **movieid=11768**, (or whatever value you got from the previous question)

```
select name
from actor join casting on(actorid=id)
where movieid=11768
```

Submit SQL

Restore default

7.



Obtain the cast list for the film 'Alien'

```
select name
from actor join casting on(actorid = id)
where movieid = (select id
from movie
where title = 'Alien')
```


Harrison Ford movies

8.



List the films in which 'Harrison Ford' has appeared

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

Submit SQL

Restore default

Harrison Ford as a supporting actor

9.



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 movie JOIN casting ON(movie.id=casting.movieid) JOIN actor ON
(casting.actorid = actor.id) WHERE actor.name = 'Harrison Ford' and
ord<>1
```

Submit SQL

Restore default

Lead actors in 1962 movies

10. 😊

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

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

Submit SQL

Restore default

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"?

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

Submit SQL

Restore default

8 Using Null

1.



List the teachers who have NULL for their department.

Why we cannot use =

```
select name  
from teacher  
where dept is NULL
```

3.



Use a different JOIN so that all teachers are listed.

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

Submit SQL

Restore default

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)
```

Using the COALESCE function

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

7.



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

```
SELECT COUNT(name) AS nombres,COUNT(mobile) AS mobiles
FROM teacher
```

Submit SQL

Restore default

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 name,  
CASE  
WHEN dept= 1 OR dept=2  
THEN 'Sci'  
ELSE 'Art' END AS dept FROM teacher
```

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 name, CASE  
WHEN dept= 1 or dept= 2  
THEN 'Sci'  
WHEN dept= 3 THEN 'Art'  
ELSE 'None' END AS dept  
FROM teacher
```

8+ Numeric Examples

In which we look at a survey and deal with some more complex calculations.

Check out one row

1. 😊

The example shows the number who responded for:

- question 1
- at 'Edinburgh Napier University'
- studying '(8) Computer Science'

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'
```

Calculate how many agree or strongly agree

2. 😊

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
```

Unhappy Computer Students

3. 😊

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
```

More Computing or Creative Students?

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'.

HINT

```
SELECT subject,SUM(response)
FROM nss
WHERE question='Q22'AND( subject = '(H) Creative Arts and Design' OR
subject='(8) Computer Science')
GROUP BY subject
```

Strongly Agree Numbers

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'.

HINT

```
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
```

Strongly Agree, Percentage

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'.

Use the **ROUND** function to show the percentage without decimal places.

```
SELECT subject, ROUND( SUM(response* A_STRONGLY_AGREE) /SUM(response))
FROM nss
WHERE question='Q22' AND (subject = '(H) Creative Arts and Design'
OR subject='(8) Computer Science')
GROUP BY subject
```

Submit SQL

Restore default

9 Self join

1. 

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

```
SELECT COUNT(name)
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 route INNER JOIN stops ON id= stop
WHERE num= 4 AND company= 'LRT'
```

4.



The query shown gives the number of routes that visit either London Road and notice the two services that link these **stops** have a count of 2. Add a two routes.

```
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 changes for different routes. Change the query so that it shows the

```
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 **st** by **name** rather than by number. Change the query so that the services between 'Craig 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 ('

```
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 'Tollc

```
SELECT 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' itself, offered by the LRT company. Include the company and bus no. c

```
SELECT name, company, num
FROM (SELECT X.num, stop, company FROM route JOIN
      (SELECT num FROM stops JOIN route ON id= stop
      WHERE company= 'LRT' AND name= 'Craiglockhart')
      AS X ON X.num= route.num ) AS Z JOIN stops
WHERE id= stop AND company= 'LRT'
```

B. Realicen los ejercicios propuestos en el siguiente quiz (59 puntos)

Tutorials: Learn SQL in stages

10 Tutorial Quizzes

1.

1. Select the code which produces this table

| name | population |
|-------------|------------|
| Bahrain | 1234571 |
| Swaziland | 1220000 |
| Timor-Leste | 1066409 |

```
FROM world
SELECT name, population BETWEEN 1000000 AND 1250000
```

```
FROM name, population
WHERE population BETWEEN 1000000 AND 1250000
SELECT world
```

```
SELECT name, population
FROM world
WHERE population BETWEEN 1000000 AND 1250000
```

```
SELECT population BETWEEN 1000000 AND 1250000
FROM world
```

```
WHERE population BETWEEN 1000000 AND 1250000
SELECT name, population FROM world
```

2. Pick the result you would obtain from this code:

```
SELECT name, population
FROM world
WHERE name LIKE "A1%"
```

Table-A

| |
|---------|
| Albania |
| Algeria |

Table-B

| | |
|--------|----------|
| %bania | 3200000 |
| %geria | 32900000 |

Table-C

| | |
|----|---|
| Al | 0 |
|----|---|

Table-D

| | |
|---------|---------|
| Albania | 3200000 |
|---------|---------|

Table-E

| | |
|---------|----------|
| Albania | 3200000 |
| Algeria | 32900000 |

3. Select the code which shows the countries that end in A or L

```
SELECT name FROM world
WHERE name LIKE 'a%' AND name LIKE 'l%'
```

```
SELECT name FROM world
WHERE name LIKE 'a%' OR name LIKE 'l%'
```

```
SELECT name FROM world
WHERE name LIKE '%a' AND name LIKE '%l'
```

```
SELECT name FROM world
WHERE name LIKE '%a' OR 'l%'
```

```
SELECT name FROM world
WHERE name LIKE '%a' OR name LIKE '%l'
```

4. Pick the result from the query

```
SELECT name,length(name)
FROM world
WHERE length(name)=5 and region='Europe'
```

| name | length(name) |
|-------|--------------|
| Benin | 5 |
| Lybia | 5 |
| Egypt | 5 |

| name | length(name) |
|-------|--------------|
| Italy | 5 |
| Egypt | 5 |
| Spain | 5 |

| name | length(name) |
|-------|--------------|
| Italy | 5 |
| Malta | 5 |
| Spain | 5 |

| name | length(name) |
|--------|--------------|
| Italy | 5 |
| France | 6 |
| Spain | 5 |

| name | length(name) |
|--------|--------------|
| Sweden | 6 |
| Norway | 6 |
| Poland | 6 |

5. Here are the first few rows of the world table:

| name | region | area | population | gdp |
|-------------|-------------|---------|------------|-------------|
| Afghanistan | South Asia | 652225 | 26000000 | |
| Albania | Europe | 28728 | 3200000 | 6656000000 |
| Algeria | Middle East | 2400000 | 32900000 | 75012000000 |
| Andorra | Europe | 468 | 64000 | |
| ... | | | | |

Pick the result you would obtain from this code:

```
SELECT name, area*2 FROM world WHERE population = 64000
```

| | |
|---------|-----|
| Andorra | 234 |
|---------|-----|

| | |
|---------|-----|
| Andorra | 468 |
|---------|-----|

| | |
|---------|-----|
| Andorra | 936 |
|---------|-----|

| | |
|---------|------|
| Andorra | 4680 |
|---------|------|

| | |
|---------|-------|
| Andorra | 936 |
| Albania | 57456 |

6. Select the code that would show the countries with an area larger than 50000 and a population smaller than 10000000

```
SELECT name, area, population
FROM world
WHERE area < 50000 AND population < 10000000
```

```
SELECT name, area, population
FROM world
WHERE area < 50000 AND population > 10000000
```

```
SELECT name, area, population
FROM world
WHERE area > 50000 AND population < 10000000
```

```
SELECT name, area, population
FROM world
WHERE area > 50000 AND population > 10000000
```

```
SELECT name, area, population
FROM world
WHERE area = 50000 AND population = 10000000
```

7. Select the code that shows the population density of China, Australia, Nigeria and France

```
SELECT name, area/population
FROM world WHERE name IN ('China', 'Nigeria', 'France', 'Australia')
```

```
SELECT name, area/population
FROM world WHERE name LIKE ('China', 'Nigeria', 'France', 'Australia')
```

```
SELECT name, population/area
FROM world
WHERE name IN ('China', 'Nigeria', 'France', 'Australia')
```

```
SELECT name, population/area
FROM world
WHERE name LIKE ('China', 'Nigeria', 'France', 'Australia')
```

```
SELECT name, population
FROM world
WHERE name IN ('China', 'Nigeria', 'France', 'Australia')
```

1. Select the code which gives the name of countries beginning with U

```
SELECT name
  FROM world
 WHERE name
 BEGIN with U
```

```
SELECT name
  FROM world
 WHERE name LIKE '%U'
```

```
SELECT name
  FROM world
 WHERE name LIKE '%u%'
```

```
SELECT name
  FROM world
 WHERE name LIKE U
```



```
SELECT name
  FROM world
 WHERE name LIKE 'U%'
```

2. Select the code which shows just the population of United Kingdom?

```
SELECT population
FROM 'United Kingdom'
```

```
SELECT name
FROM world
WHERE population = 'United Kingdom'
```

```
SELECT FROM world
WHERE population IN 'United Kingdom'
```



```
SELECT population
FROM world
WHERE name = 'United Kingdom'
```

```
SELECT population
FROM world
WHERE 'United Kingdom' IN name
```

3. Select the answer which shows the problem with this SQL code - the intended result should be the continent of France:

```
SELECT continent
FROM world
WHERE 'name' = 'France'
```

continent should be 'continent'



'name' should be name

'France' should be "France"

'France' should be France

= should be IN

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

```
SELECT name, population / 10
FROM world
WHERE population < 10000
```

| | |
|---------|------|
| Andorra | 6400 |
| Nauru | 990 |

| | |
|---------|-------|
| Andorra | 64000 |
| Nauru | 9900 |

| | |
|-------|----|
| Nauru | 99 |
|-------|----|

| | | |
|-------------------------------------|-------|-----|
| <input checked="" type="checkbox"/> | Nauru | 990 |
|-------------------------------------|-------|-----|

| | |
|-------|------|
| Nauru | 9900 |
|-------|------|

5. Select the code which would reveal the name and population of countries in Europe and Asia

```
SELECT name
FROM world
WHERE continent IN ('Europe', 'Asia')
```



```
SELECT name, population
FROM world
WHERE continent IN ('Europe', 'Asia')
```

```
SELECT name, population
FROM world
WHERE name IN (Europe Asia)
```

```
SELECT name, population
FROM world
WHERE name IS ('Europe', 'Asia')
```

```
SELECT name, population
FROM world
WHERE continent = ('Europe', 'Asia')
```


6. Select the code which would give two rows

```
SELECT name FROM world
WHERE name = 'Cuba'
```

```
SELECT name FROM world
WHERE name = 'Cuba'
AND name = 'Togo'
```

```
SELECT name FROM world
WHERE name EITHER ('Cuba', 'Togo')
```

```
SELECT name FROM world
WHERE name IN ('Cuba', 'Togo')
```

```
SELECT name FROM WHERE name IS 'Mali'
```

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

```
SELECT name FROM world
WHERE continent = 'South America'
AND population > 40000000
```

| |
|-------------|
| Afghanistan |
|-------------|

| |
|--------|
| Brazil |
|--------|

| |
|----------|
| Colombia |
|----------|

| |
|--------|
| Brazil |
|--------|

| |
|--------|
| Brazil |
|--------|

| |
|----------|
| Colombia |
|----------|

| | |
|--------|---------------|
| Brazil | South America |
|--------|---------------|

| | |
|----------|---------------|
| Colombia | South America |
|----------|---------------|

| | |
|--------|-----------|
| Brazil | 182800000 |
|--------|-----------|

| | |
|----------|----------|
| Colombia | 45600000 |
|----------|----------|

3.

1. Pick the code which shows the name of winner's names beginning with C and ending in n

```
SELECT name FROM nobel
WHERE winner LIKE 'C%' AND winner LIKE '%n'
```

```
SELECT name FROM nobel
WHERE winner LIKE 'C' AND winner LIKE 'n'
```

```
SELECT name FROM nobel
WHERE winner LIKE 'C%' AND winner LIKE '%n'
```

```
SELECT winner FROM nobel
WHERE winner LIKE 'C' AND winner LIKE 'n'
```

```
SELECT winner FROM nobel
WHERE winner LIKE 'C%' AND winner LIKE '%n'
```

2. Select the code that shows how many Chemistry awards were given between 1950 and 1960

```
SELECT COUNT(subject) FROM nobel
WHERE subject = 'Chemistry'
AND BETWEEN 1950 and 1960
```

```
SELECT COUNT(subject) FROM nobel
WHERE subject = 'Chemistry'
AND yr BETWEEN (1950, 1960)
```

```
SELECT COUNT(subject) FROM nobel
WHERE subject = 'Chemistry'
AND yr BETWEEN 1950 and 1960
```

```
SELECT subject FROM nobel
WHERE subject = 'Chemistry'
AND yr BETWEEN 1950 and 1960
```

```
SELECT subject FROM nobel
WHERE subject = 'Chemistry'
AND yr BETWEEN (1950, 1960)
```

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

```
SELECT subject, winner FROM nobel WHERE winner LIKE 'Sir%' AND yr LIKE '196%'
```

| | |
|----------|-------------------------|
| Medicine | John Eccles |
| Medicine | Frank Macfarlane Burnet |

| | |
|-----------|-----------------------|
| Chemistry | Sir Cyril Hinshelwood |
|-----------|-----------------------|

| | |
|----------|-----------------------------|
| Medicine | Sir John Eccles |
| Medicine | Sir Frank Macfarlane Burnet |

| | |
|-----------|-------------------------|
| Medicine | John Eccles |
| Medicine | Frank Macfarlane Burnet |
| Chemistry | Willard F.Libby |

| | |
|-----------------------------|--|
| Sir John Eccles | |
| Sir Frank Macfarlane Burnet | |

5. Select the code which would show the year when neither a Physics or Chemistry award was given

```
SELECT yr FROM nobel
WHERE subject NOT IN(SELECT yr
FROM nobel
WHERE subject IN ('Chemistry','Physics'))
```

```
SELECT yr FROM nobel
WHERE subject NOT IN(SELECT subject
FROM nobel
WHERE subject IN ('Chemistry','Physics'))
```

```
SELECT yr FROM nobel
WHERE yr NOT IN(SELECT yr
FROM nobel
WHERE subject IN ('Chemistry','Physics'))
```

```
SELECT yr FROM nobel
WHERE yr NOT IN(SELECT subject
FROM nobel
WHERE subject IN ('Chemistry','Physics'))
```

6. Select the code which shows the years when a Medicine award was given but no Peace or Literature award was

```
SELECT DISTINCT yr
FROM nobel
WHERE subject='Medicine' AND
subject NOT IN(SELECT yr from nobel
WHERE subject='Literature')
AND yr NOT IN (SELECT yr
FROM nobel
WHERE subject='Peace')
```

```
SELECT DISTINCT yr
FROM nobel WHERE subject='Medicine'
AND yr NOT IN(SELECT yr from nobel
WHERE subject='Literature'
AND subject='Peace')
```

```
SELECT DISTINCT yr
FROM nobel
WHERE subject='Medicine'
AND yr NOT IN(SELECT yr FROM nobel
WHERE subject='Literature')
AND yr NOT IN (SELECT yr FROM nobel
WHERE subject='Peace')
```

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

```
SELECT subject, COUNT(subject)
FROM nobel
WHERE yr = '1960'
GROUP BY subject
```

| |
|---|
| 1 |
| 1 |
| 2 |
| 1 |
| 1 |

| | |
|-----------|---|
| Chemistry | 6 |
|-----------|---|

| | |
|------------|---|
| Chemistry | 3 |
| Literature | 1 |
| Medicine | 2 |
| Peace | 0 |
| Physics | 2 |

| | |
|------------|---|
| Chemistry | 1 |
| Literature | 1 |
| Medicine | 2 |
| Peace | 1 |
| Physics | 1 |

4.

1. Select the code that shows the name, region and population of the smallest country in each region

```
SELECT region, name, FROM bbc x WHERE population <= ALL (SELECT population FROM bbc y WHERE y.region=x.region AND population>0)
```

```
SELECT region, name, population FROM bbc WHERE population <= ALL (SELECT population FROM bbc WHERE population>0)
```

```
SELECT region, name, population FROM bbc x WHERE population <= ALL (SELECT population FROM bbc y WHERE y.region=x.region AND population>0)
```

```
SELECT region, name, population FROM bbc x WHERE population = ALL (SELECT population FROM bbc y WHERE y.region=x.region AND population>0)
```

```
SELECT region, name, population FROM bbc x WHERE population <= ALL (SELECT population FROM bbc y WHERE y.region=x.region AND population>0)
```

1. Select the code that shows the name, region and population of the smallest country in each region

```
SELECT region, name, FROM bbc x WHERE population <= ALL (SELECT population FROM bbc y WHERE y.region=x.region AND population>0)
```

```
SELECT region, name, population FROM bbc WHERE population <= ALL (SELECT population FROM bbc WHERE population>0)
```

```
SELECT region, name, population FROM bbc x WHERE population <= ALL (SELECT population FROM bbc y WHERE y.region=x.region AND population>0)
```

```
SELECT region, name, population FROM bbc x WHERE population = ALL (SELECT population FROM bbc y WHERE y.region=x.region AND population>0)
```

```
SELECT region, name, population FROM bbc x WHERE population <= ALL (SELECT population FROM bbc y WHERE y.region=x.region AND population<0)
```

2. Select the code that shows the countries belonging to regions with all populations over 50000

```
SELECT name,region,population FROM bbc x WHERE 50000 < ALL (SELECT population FROM bbc y WHERE population>0)
```

```
SELECT name,region,population FROM bbc x WHERE 50000 < ALL (SELECT population FROM bbc y WHERE x.region=y.region AND y.population>0)
```

```
SELECT name,region,population FROM bbc x WHERE 50000 = ALL (SELECT population FROM bbc y WHERE x.region=y.region AND y.population>0)
```

```
SELECT name,region,population FROM bbc x WHERE 50000 > ALL (SELECT population FROM bbc y WHERE x.region=y.region AND y.population>0)
```

```
SELECT name,region,population FROM bbc x WHERE 500000 < ALL (SELECT population FROM bbc y WHERE x.region=y.region AND y.population>0)
```

2. Select the code that shows the countries belonging to regions with all populations over 50000

```
SELECT name,region,population FROM bbc x WHERE 50000 < ALL (SELECT population FROM bbc y WHERE population>0)
```

```
SELECT name,region,population FROM bbc x WHERE 50000 < ALL (SELECT population FROM bbc y WHERE x.region=y.region AND y.population>0)
```

```
SELECT name,region,population FROM bbc x WHERE 50000 = ALL (SELECT population FROM bbc y WHERE x.region=y.region AND y.population>0)
```

```
SELECT name,region,population FROM bbc x WHERE 50000 > ALL (SELECT population FROM bbc y WHERE x.region=y.region AND y.population>0)
```

```
SELECT name,region,population FROM bbc x WHERE 500000 < ALL (SELECT population FROM bbc y WHERE x.region=y.region AND y.population>0)
```

3. Select the code that shows the countries with a less than a third of the population of the countries around it

```
SELECT name, region FROM bbc x
WHERE population < ALL (SELECT population/3 FROM bbc y WHERE y.region = x.region AND y.name != x.name)
```

```
SELECT name, region FROM bbc x
WHERE population = ALL (SELECT population/3 FROM bbc y WHERE y.region = x.region AND y.name != x.name)
```

```
SELECT name, region FROM bbc x
WHERE population > ALL (SELECT population/3 FROM bbc y WHERE y.region = x.region AND y.name != x.name)
```

```
SELECT name, region FROM bbc x WHERE population < ALL (SELECT population*3 FROM bbc y WHERE y.region = x.region AND y.name != x.name)
```

```
SELECT name, region FROM bbc x WHERE population < ALL (SELECT population/3 FROM bbc y WHERE y.name != x.name)
```

3. Select the code that shows the countries with a less than a third of the population of the countries around it

```
SELECT name, region FROM bbc x
WHERE population < ALL (SELECT population/3 FROM bbc y WHERE y.region = x.region AND y.name != x.name)
```

```
SELECT name, region FROM bbc x
WHERE population = ALL (SELECT population/3 FROM bbc y WHERE y.region = x.region AND y.name != x.name)
```

```
SELECT name, region FROM bbc x
WHERE population > ALL (SELECT population/3 FROM bbc y WHERE y.region = x.region AND y.name != x.name)
```

```
SELECT name, region FROM bbc x WHERE population < ALL (SELECT population*3 FROM bbc y WHERE y.region = x.region AND y.name != x.name)
```

```
SELECT name, region FROM bbc x WHERE population < ALL (SELECT population/3 FROM bbc y WHERE y.name != x.name)
```

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

```
SELECT name FROM bbc
WHERE population >
  (SELECT population
   FROM bbc
   WHERE name='United Kingdom')
AND region IN
  (SELECT region
   FROM bbc
   WHERE name = 'United Kingdom')
```

Table-A

| |
|----------|
| Andorra |
| Albania |
| Austria |
| Bulgaria |

Table-B

| | |
|---------|--------|
| France | Europe |
| Germany | Europe |
| Russia | Europe |
| Turkey | Europe |

Table-C

| |
|---------|
| France |
| Germany |
| Andorra |
| Albania |

Table-D

| |
|---------|
| France |
| Germany |
| Russia |
| Turkey |

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

```
SELECT name FROM bbc
WHERE population >
  (SELECT population
   FROM bbc
   WHERE name='United Kingdom')
AND region IN
  (SELECT region
   FROM bbc
   WHERE name = 'United Kingdom')
```

Table-A

| |
|----------|
| Andorra |
| Albania |
| Austria |
| Bulgaria |

Table-B

| | |
|---------|--------|
| France | Europe |
| Germany | Europe |
| Russia | Europe |
| Turkey | Europe |

Table-C

| |
|---------|
| France |
| Germany |
| Andorra |
| Albania |

Table-D

| |
|---------|
| France |
| Germany |
| Russia |
| Turkey |

5. Select the code that would show the countries with a greater GDP than any country in Africa (some countries may have NULL gdp values).

```
SELECT name FROM bbc
WHERE gdp > ALL (SELECT MAX(gdp) FROM bbc WHERE region = 'Africa' AND gdp=0)
```

```
SELECT name FROM bbc
WHERE gdp > (SELECT MAX(gdp) FROM bbc WHERE region = 'Africa')
```

```
SELECT name FROM bbc
WHERE gdp > ALL (SELECT MIN(gdp) FROM bbc WHERE region = 'Africa')
```

```
SELECT name FROM bbc
WHERE gdp > ALL (SELECT gdp FROM bbc WHERE region = 'Africa')
```

```
SELECT name FROM bbc
WHERE gdp > ALL (SELECT gdp FROM bbc WHERE region = 'Africa' AND gdp<>NULL)
```

5. Select the code that would show the countries with a greater GDP than any country in Africa (some countries may have NULL gdp values).

```
SELECT name FROM bbc
WHERE gdp > ALL (SELECT MAX(gdp) FROM bbc WHERE region = 'Africa' AND gdp<>0)
```

```
SELECT name FROM bbc
WHERE gdp > (SELECT MAX(gdp) FROM bbc WHERE region = 'Africa')
```

```
SELECT name FROM bbc
WHERE gdp > ALL (SELECT MIN(gdp) FROM bbc WHERE region = 'Africa')
```

```
SELECT name FROM bbc
WHERE gdp > ALL (SELECT gdp FROM bbc WHERE region = 'Africa')
```

```
SELECT name FROM bbc
WHERE gdp > ALL (SELECT gdp FROM bbc WHERE region = 'Africa' AND gdp<>NULL)
```

6. Select the code that shows the countries with population smaller than Russia but bigger than Denmark

```
SELECT name FROM bbc
WHERE population < (SELECT population FROM bbc WHERE name='Denmark')
AND population > (SELECT population FROM bbc WHERE name='Russia')
```

```
SELECT name FROM bbc
WHERE population < (SELECT population FROM bbc WHERE name='Russia')
AND population > (SELECT population FROM bbc WHERE name='Denmark')
```

```
SELECT name FROM bbc
WHERE population = (SELECT population FROM bbc WHERE name='Russia')
AND population > (SELECT population FROM bbc WHERE name='Denmark')
```

```
SELECT name FROM bbc
WHERE population > (SELECT population FROM bbc WHERE name='Russia')
AND population > (SELECT population FROM bbc WHERE name='Denmark')
```

```
SELECT name FROM bbc
WHERE population < (SELECT population FROM bbc WHERE name='Russia')
AND population > (SELECT population FROM bbc WHERE name='Denmark')
```


6. Select the code that shows the countries with population smaller than Russia but bigger than Denmark

```
SELECT name FROM bbc
WHERE population < (SELECT population FROM bbc WHERE name='Denmark')
AND population > (SELECT population FROM bbc WHERE name='Russia')
```

```
SELECT name FROM bbc
WHERE population < (SELECT population FROM bbc WHERE name='Russia')
AND population > (SELECT population FROM bbc WHERE name='Denmark')
```

```
SELECT name FROM bbc
WHERE population = (SELECT population FROM bbc WHERE name='Russia')
AND population > (SELECT population FROM bbc WHERE name='Denmark')
```

```
SELECT name FROM bbc
WHERE population > (SELECT population FROM bbc WHERE name='Russia')
AND population > (SELECT population FROM bbc WHERE name='Denmark')
```

```
SELECT name FROM bbc
WHERE population < (SELECT population FROM bbc WHERE name='Russia')
AND population > (SELECT population FROM bbc WHERE name='Denmark')
```

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

```
SELECT name FROM bbc
WHERE population > ALL
    (SELECT MAX(population)
     FROM bbc
     WHERE region = 'Europe')
AND region = 'South Asia'
```

Table-A

| |
|--------------|
| Afghanistan |
| Bhutan |
| Nepal |
| Sri Lanka |
| The Maldives |

Table-B

| |
|------------|
| Bangladesh |
| India |
| Pakistan |

Table-C

| |
|-------|
| China |
| India |

Table-D

| |
|------------|
| Brazil |
| Bangladesh |
| China |
| India |

Table-E

| |
|---------|
| France |
| Germany |
| Russia |
| Turkey |

5.

1. Select the statement that shows the sum of population of all countries in 'Europe'

`SELECT name, population FROM bbc WHERE region = 'Europe'`

`SELECT population FROM bbc WHERE region = 'Europe' SUM BY region`

`SELECT SUM(population) FROM bbc WHERE region = 'Europe'`

`SELECT SUM(population FROM bbc WHERE region = 'Europe')`

`SUM population FROM bbc WHERE region = 'Europe'`

2. Select the statement that shows the number of countries with population smaller than 150000

`SELECT COUNT(name) FROM bbc WHERE population < 150000`

`SELECT COUNT(population < 150000) FROM bbc`

`SELECT name FROM bbc WHERE population < 150000`

`SELECT population AS COUNT FROM bbc WHERE population < 150000`

`SELECT SUM() FROM bbc WHERE population < 150000`

3. Select the list of core SQL aggregate functions

AVG(), COUNT(), FIRST(), LAST(), SUM()

AVG(), COUNT(), MAX(), MEDIAN(), MIN(), ROUND(), SUM()

AVG(), COUNT(), CONCAT(), FIRST(), LAST(), MAX(), MIN(), SUM()

☒ AVG(), COUNT(), MAX(), MIN(), SUM()

COUNT(), SUM()

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

```
SELECT region, SUM(area)
FROM bbc
WHERE SUM(area) > 15000000
GROUP BY region
```

Table-A

| | |
|--------|----------|
| Europe | 17000000 |
|--------|----------|

Table-B

| | |
|---------------|----------|
| Europe | 17000000 |
| Asia-Pacific | 23460000 |
| North America | 21660000 |

Table-C

| |
|---------------|
| Europe |
| Asia-Pacific |
| North America |

No result due to invalid use of the GROUP BY function

☒ No result due to invalid use of the WHERE function

5. Select the statement that shows the average population of 'Poland', 'Germany' and 'Denmark'

```
SELECT AVG(population) FROM bbc WHERE name = ('Poland', 'Germany', 'Denmark')
```

```
SELECT AVG(population) FROM bbc WHERE name IN ('Poland', 'Germany', 'Denmark')
```

```
SELECT AVG(population) FROM bbc WHERE name LIKE ('Poland', 'Germany', 'Denmark')
```

```
SELECT AVG(population) FROM bbc WHERE name LIKE (Poland, Germany, Denmark)
```

```
SELECT population FROM bbc WHERE name IN ('Poland', 'Germany', 'Denmark')
```

6. Select the statement that shows the medium population density of each region

```
SELECT region, AVG(population/area) AS density FROM bbc
```

```
SELECT region, COUNT(population)/COUNT(area) AS density FROM bbc GROUP BY region
```

```
SELECT region, SUM(population)/COUNT(area) AS density FROM bbc GROUP BY region
```

```
SELECT region, SUM(population)/SUM(area) AS density FROM bbc HAVING region
```

```
SELECT region, SUM(population)/SUM(area) AS density FROM bbc GROUP BY region
```

7. Select the statement that shows the name and population density of the country with the largest population

`SELECT name, density AS population/area FROM bbc WHERE population = MAX(population)`

`SELECT name, density AS population/area FROM bbc WHERE population = (SELECT MAX(population) FROM bbc)`

`SELECT name, MAX (population) FROM bbc WHERE population / (SELECT area FROM bbc)`

`SELECT name, population/area AS density FROM bbc WHERE population = (SELECT MAX(population) FROM bbc)`

`SELECT name, population/area AS density FROM bbc WHERE population > (SELECT MAX(population) FROM bbc)`

`SELECT region, SUM(area)
FROM bbc
GROUP BY region
HAVING SUM(area)<= 20000000`

Table-A

| |
|----------|
| 732240 |
| 13403102 |
| 17740392 |
| 4943771 |

Table-B

| | |
|---------------|----------|
| Africa | 22550927 |
| Asia-Pacific | 28759578 |
| Europe | 23866987 |
| North America | 21660000 |

Table-C

| |
|---------------|
| Africa |
| Asia-Pacific |
| Europe |
| North America |

Table-D

| | |
|---------------|----------|
| Americas | 732240 |
| Middle East | 13403102 |
| South America | 17740392 |
| South Asia | 9437710 |

1. Select the statement which lists the unfortunate directors of the movies which have caused financial losses (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
```

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

```
SELECT name
FROM director INNER JOIN movie ON movie.id = director.id
```

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)
AND name LIKE 'John %'
GROUP BY name ORDER BY 2 DESC
```

```
SELECT name, COUNT(movieid)
FROM casting JOIN actor
```

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

```
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 |
| Limitless |

8.

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 |

9.

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'
```

```
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 = 'Sighthill'
```

```
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 y.name='Craiglockhart' AND z.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'
```

```
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 stopz.name='Tollcross'
```

C. Propongan preguntas que cumplan los siguientes requerimientos. (15 puntos) Usen la base de datos **musicians**

Escoja el motor que prefiera. Justifique la elección.

[Escriban la consulta en lenguaje natural y la sentencia en SQL en **auto02..doc** y ejecuten la sentencia SQL en sqlzoo . Si no lograron escribir alguna sentencia indiquen el punto de problema]

- 5 consultas: una para cada operador de conjuntos
- 4 consultas: dos para junta interna y dos para junta externa
- 2 consultas: una para cada operador de desconocido
- 3 consultas: una para cada uno de los tipos de operadores lógicos
- 1 consulta: para el operador CASE

1)

UNION ALL

SELECT 'BORN IN', m_name FROM musician,place WHERE (born_in = place_no) AND place_town= 'Glasgow' UNION ALL

SELECT 'LIVES IN',m_name FROM musician,place WHERE (living_in= place_no) AND place_town= 'Glasgow'

UNION ALL

SELECT 'IN_BAND_IN',m_name FROM band,place,musician,performer,plays_in WHERE band_home= place_no AND place_town= 'Glasgow'

AND band_id= band_no AND player= perf_no AND perf_is=m_no ORDER BY m_name

UNION

(SELECT place_town AS ciudad_pais

FROM musician JOIN place ON place_no= born_in)

UNION

(SELECT place_country AS Country

FROM musician JOIN place ON place_no= living_in)

INTERSECT

No lo pudimos realizar, ya que se tuvo un error de sintaxis que no se pudo solucionar

EXTRACT

SELECT EXTRACT(YEAR_MONTH

FROM (SELECT died FROM musician

WHERE m_name= 'Harriet Smithson'))

2)

JOIN

SELECT concert_venue AS Venue, place_country AS Country

FROM place JOIN concert ON concert_in= place_no

CROSS JOIN

SELECT m_name AS Name, place_country AS Country

FROM place CROSS JOIN musician ON born_in = place_no

WHERE place_country= 'USA'

LEFT JOIN

SELECT m_name AS name, compositions, COALESCE(instrument, 0) AS instrument

FROM (SELECT m_name, COALESCE(compositions, 0) AS compositions

FROM (SELECT DISTINCT m_name from musician, place WHERE born_in= place_no AND
(place_country= 'England' OR place_country= 'Scotland')) AS PRINCIPAL

LEFT JOIN

(SELECT COUNT(cmp_r_no) AS compositions, m_name AS name FROM

musician, composer, has_composed WHERE m_no= comp_is AND comp_no= cmp_r_no

GROUP BY (name)) AS COMPOSICIONES ON name= m_name) AS UNO

LEFT JOIN

(SELECT COUNT(perf_is) AS instrument , m_name AS name2 FROM musician, performer

WHERE perf_is= m_no GROUP BY (name2)) AS DOS

ON m_name= name2

RIGHT JOIN

SELECT re AS Name, band_name AS 'Jeff''s band' , n AS 'Sue''s band'

FROM (SELECT DISTINCT m_name AS re, band_name FROM

band, place, musician, performer, plays_in WHERE perf_is= m_no AND player= perf_no AND

band_id= band_no AND band_name IN (SELECT band_name FROM

band, place, musician, performer, plays_in WHERE perf_is= m_no AND player= perf_no AND

band_id= band_no AND m_name = 'Jeff Dawn')) AS X

RIGHT OUTER JOIN

(SELECT DISTINCT m_name, band_name AS n FROM

band, place, musician, performer, plays_in WHERE perf_is= m_no AND player= perf_no AND

band_id= band_no AND band_name IN (SELECT band_name FROM

band, place, musician, performer, plays_in WHERE perf_is= m_no AND player= perf_no AND

band_id= band_no AND m_name = 'Sue Little')) AS Y

ON re= m_name

WHERE re IS NOT NULL

3)

ISNULL:

SELECT m_name, ISNULL(died) FROM musician

COALESCE:

SELECT m_name AS name, compositions, COALESCE(instrument, 0) AS instrument FROM

(SELECT m_name, COALESCE(compositions, 0) AS compositions FROM (SELECT

DISTINCT m_name FROM musician, place WHERE born_in = place_no AND

(place_country= 'England' OR place_country= 'Scotland')) AS PRINCIPAL

LEFT JOIN

(SELECT COUNT(cmp_r_no) AS compositions, m_name AS name FROM

```

musician,composer,has_composed WHERE m_no= comp_is AND comp_no= cmpr_no
GROUP BY (name)) AS COMPOSICIONES ON name= m_name)AS UNO
LEFT JOIN
(SELECT COUNT(perf_is) AS instrument , m_name AS name2 FROM musician,performer

WHERE perf_is=m_no GROUP BY (name2)) AS DOS
ON m_name= name2

```

4)

```

ALL:
SELECT instrument
FROM(SELECT count(perf_is) as n,instrument FROM performer GROUP BY instrument) AS NN
WHERE n <= ALL(SELECT COUNT(perf_no) FROM performer GROUP BY instrument)
ANY
SELECT m_name
FROM musician
WHERE m_name <> ANY(SELECT m_name FROM musician WHERE m_name='Fred Bloggs')
EXISTS
SELECT m_name FROM musician
WHERE EXISTS (SELECT m_name FROM musician WHERE m_name= 'Ana')
SELECT m_name FROM musician
WHERE EXISTS (SELECT m_name FROM musician WHERE died IS NULL)

```

5)

```

CASE
SELECT m_name AS name,
CASE
WHEN died IS NULL THEN 'Vive'
ELSE 'Muerto'
END AS X
FROM musician

```