Solución Desafío 3

Consultas en múltiples tablas

Descripción

1. Crea y agrega al entregable las consultas para completar el setup de acuerdo a lo pedido. **(1 Punto)**

```
CREATE DATABASE desafio3_consuelo_garrido_789;
```

TABLA USUARIOS

```
CREATE TABLE usuarios (id SERIAL, email VARCHAR, nombre VARCHAR,
apellido VARCHAR, rol VARCHAR);
```

```
INSERT INTO usuarios(id, email, nombre, apellido, rol) VALUES (DEFAULT,
'juan@mail.com', 'juan', 'perez', 'administrador');
INSERT INTO usuarios(id, email, nombre, apellido, rol) VALUES (DEFAULT,'
diego@mail.com', 'diego', 'munoz', 'usuario');
INSERT INTO usuarios(id, email, nombre, apellido, rol) VALUES
(DEFAULT,'maria@mail.com', 'maria', 'meza', 'usuario');
INSERT INTO usuarios(id, email, nombre, apellido, rol) VALUES
(DEFAULT,'roxana@mail.com','roxana', 'diaz', 'usuario');
INSERT INTO usuarios(id, email, nombre, apellido, rol) VALUES
(DEFAULT,'pedro@mail.com', 'pedro', 'diaz', 'usuario');
```

id	email	nombre	apellido	rol
1	juan@mail.com	juan	perez	administrador
2	diego@mail.com	diego	munoz	usuario
3	maria@mail.com	maria	meza	usuario
4	roxana@mail.com	roxana	diaz	usuario
5	pedro@mail.com	pedro	diaz	usuario

(5 filas)

TABLA POSTS

```
CREATE TABLE posts (id SERIAL, titulo VARCHAR, contenido TEXT,
fecha_creacion DATE, fecha_actualizacion DATE, destacado BOOLEAN,
usuario_id BIGINT);
```

```
INSERT INTO posts (id, titulo, contenido, fecha_creacion,
fecha_actualizacion, destacado, usuario_id) VALUES (DEFAULT, 'prueba',
'contenido prueba', '01/01/2021', '01/02/2021', true, 1);
INSERT INTO posts (id, titulo, contenido, fecha_creacion,
fecha_actualizacion, destacado, usuario_id) VALUES (DEFAULT, 'prueba2',
'contenido prueba2', '01/03/2021', '01/03/2021', true, 1);
INSERT INTO posts (id, titulo, contenido, fecha_creacion,
fecha_actualizacion, destacado, usuario_id) VALUES (DEFAULT,
'ejercicios', 'contenido ejercicios', '02/05/2021', '03/04/2021', true,
2);
INSERT INTO posts (id, titulo, contenido, fecha_creacion,
fecha_actualizacion, destacado, usuario_id) VALUES (DEFAULT,
'ejercicios2', 'contenido ejercicios2', '03/05/2021', '04/04/2021',
false, 2);
INSERT INTO posts (id, titulo, contenido, fecha_creacion,
fecha_actualizacion, destacado, usuario_id) VALUES (DEFAULT, 'random',
'contenido random', '03/06/2021', '04/05/2021', false, null);
```

id	titulo	contenido	fecha_creacion	fecha_actualizacion	destacado	usuario_id
1	prueba	contenido prueba	2021-01-01	2021-02-01	t	1
2	prueba2	contenido prueba2	2021-03-01	2021-03-01	t	1
3	ejercicios	contenido ejercicios	2021-05-02	2021-04-03	t	2
4	ejercicios2	contenido ejercicios2	2021-05-03	2021-04-04	f	2
5	random	contenido random	2021-06-03	2021-05-04	f	

(5 filas)

TABLA COMENTARIOS

```
CREATE TABLE comentarios (id SERIAL, contenido VARCHAR, fecha_creacion
DATE, usuario_id BIGINT, post_id BIGINT);
```

```
INSERT INTO comentarios (id, contenido, fecha_creacion, usuario_id,
post_id) VALUES (DEFAULT, 'comentario 1', '03/06/2021', 1, 1);
INSERT INTO comentarios (id, contenido, fecha_creacion, usuario_id,
post_id) VALUES (DEFAULT, 'comentario 2', '03/06/2021', 2, 1);
INSERT INTO comentarios (id, contenido, fecha_creacion, usuario_id,
post_id) VALUES (DEFAULT, 'comentario 3', '04/06/2021', 3, 1);
INSERT INTO comentarios (id, contenido, fecha_creacion, usuario_id,
post_id) VALUES (DEFAULT, 'comentario 4', '04/06/2021', 1, 2);
INSERT INTO comentarios (id, contenido, fecha_creacion, usuario_id,
post_id) VALUES (DEFAULT, 'comentario 5', '04/06/2021', 2, 2);
```

id	contenido	fecha_creacion	usuario_id	post_id
1	comentario 1	2021-06-03	1	1
2	comentario 2	2021-06-03	2	1
3	comentario 3	2021-06-04	3	1
4	comentario 4	2021-06-04	1	2
5	comentario 5	2021-06-04	2	2

(5 filas)

2. Cruza los datos de la tabla usuarios y posts mostrando las siguientes columnas. nombre y email del usuario junto al título y contenido del post. **(1 Punto)**

```
SELECT usuarios.nombre, usuarios.email, posts.titulo, posts.contenido
FROM usuarios JOIN posts ON usuarios.id = posts.usuario_id;
```

nombre	email	titulo	contenido
juan	juan@mail.com	prueba	contenido prueba
juan	juan@mail.com	prueba2	contenido prueba2
diego	diego@mail.com	ejercicios	contenido ejercicios
diego	diego@mail.com	ejercicios2	contenido ejercicios2

3. Muestra el id, título y contenido de los posts de los administradores. El administrador puede ser cualquier id y debe ser seleccionado dinámicamente. **(1 Punto)**

```
SELECT posts.id, posts.titulo, posts.contenido FROM usuarios JOIN posts
ON usuarios.id = posts.usuario_id WHERE usuarios.rol =
'administrador';
```

id	titulo	contenido
1	prueba	contenido prueba
2	prueba2	contenido prueba2

4. Cuenta la cantidad de posts de cada usuario. La tabla resultante debe mostrar el id y email del usuario junto con la cantidad de posts de cada usuario.
    a. Hint importante: Aquí hay diferencia entre utilizar inner join, left join o right join, prueba con ambas y con eso determina cuál es la correcta. No da lo mismo desde cuál tabla partes. **(1 Punto)**

```
SELECT usuarios.id, usuarios.email, COUNT(posts.id) FROM usuarios LEFT
JOIN posts ON usuarios.id = posts.usuario_id GROUP BY usuarios.id,
usuarios.email;
```

id	email	count
3	maria@mail.com	0
5	pedro@mail.com	0
4	roxana@mail.com	0
1	juan@mail.com	2
2	diego@mail.com	2

5. Muestra el email del usuario que ha creado más posts. Aquí la tabla resultante tiene un único registro y muestra solo el email. **(1 Punto)**

```
SELECT usuarios.email FROM posts JOIN usuarios ON posts.usuario_id =
usuarios.id GROUP BY usuarios.id, usuarios.email ORDER BY
COUNT(posts.id) DESC;
```

email
juan@mail.com
diego@mail.com

6. Muestra la fecha del último post de cada usuario. **(1 Punto)**
    a. Hint: Utiliza la función de agregado MAX sobre la fecha de creación

```
SELECT nombre, MAX(fecha_creacion)
FROM (SELECT posts.contenido,posts.fecha_creacion, usuarios.nombre FROM
usuarios JOIN posts ON usuarios.id = posts.usuario_id) AS p GROUP BY
p.nombre;
```

nombre	max
diego	2021-05-03
juan	2021-03-01

7. Muestra el título y contenido del post (artículo) con más comentarios. **(1 Punto)**

```
SELECT titulo, contenido FROM posts JOIN (SELECT post_id, COUNT(post_id)
FROM comentarios GROUP BY post_id ORDER BY count DESC LIMIT 1) AS c ON
posts.id = c.post_id;
```

titulo	contenido
prueba	contenido prueba

8. Muestra en una tabla el título de cada post, el contenido de cada post y el contenido de cada comentario asociado a los post mostrados, junto con el email del usuario que lo escribió. (**1 Punto**)

```
SELECT posts.titulo as titulo_post, posts.contenido as contenido_post,
comentarios.contenido as contenido_comentario, usuarios.email
      FROM posts
JOIN comentarios ON posts.id = comentarios.post_id
JOIN usuarios ON comentarios.usuario_id = usuarios.id;
```

titulo_post	contenido_post	contenido_comentario	email
prueba	contenido prueba	comentario 1	juan@mail.com
prueba	contenido prueba	comentario 2	diego@mail.com
prueba	contenido prueba	comentario 3	maria@mail.com
prueba2	contenido prueba2	comentario 4	juan@mail.com
prueba2	contenido prueba2	comentario 5	diego@mail.com

9. Muestra el contenido del último comentario de cada usuario. (**1 Punto**)

```
SELECT fecha_creacion, contenido, usuario_id FROM comentarios as c JOIN
usuarios as u ON c.usuario_id = u.id WHERE c.fecha_creacion = (SELECT
MAX(fecha_creacion) FROM comentarios WHERE usuario_id = u.id);
```

fecha_creacion	contenido	usuario_id
2021-06-04	comentario 4	1
2021-06-04	comentario 5	2
2021-06-04	comentario 3	3

10. Muestra los emails de los usuarios que no han escrito ningún comentario. (**1 Punto**)
    a. Hint: Recuerda el Having

```
SELECT usuarios.email FROM usuarios LEFT JOIN comentarios ON usuarios.id
= comentarios.usuario_id GROUP BY usuarios.email HAVING
COUNT(comentarios.id) = 0;
```

email
roxana@mail.com
pedro@mail.com