Comenzado el	jueves, 5 de junio de 2025, 15:57
Estado	Finalizado
Finalizado en	jueves, 5 de junio de 2025, 17:18
Tiempo empleado	1 hora 21 minutos

Pregunta **1**

Finalizado

Se puntúa como 0 sobre 1,00

Utilizando el esquema unc_esq_peliculas. Contar todos los distribuidores nacionales cuyo teléfono empiece con 23.

Tiempo estimado 20 min

a. **Ninguna de las opciones es correcta**

b. **7**

c. **135**

d. **120**

e. **0**

Respuesta correcta

SQL que lo resuelve:

```
SELECT count(*)
FROM unc_esq_peliculas.distribuidor
WHERE tipo = 'N' AND telefono LIKE '23%';
```

La respuesta correcta es:
7

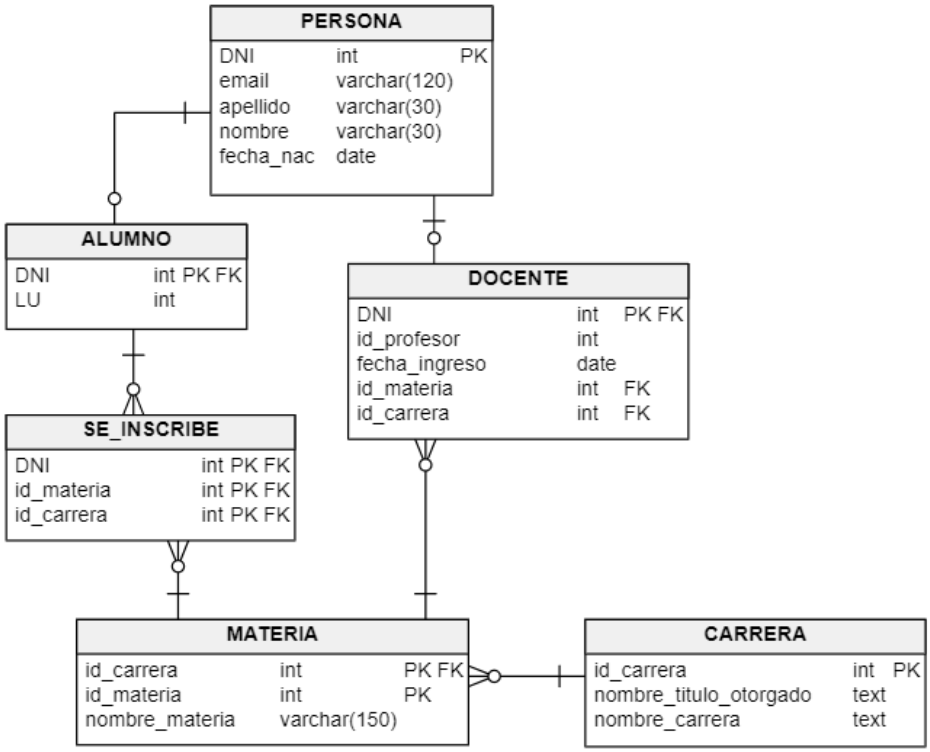
Pregunta 2

Finalizado

Se puntúa como 0 sobre 1,00

Tiempo estimado 15 min

Dado el esquema de la figura:



Si las foreign key fueran las que se indican debajo y se encuentran cargados los siguientes datos:

ALUMNO	
DNI	*LU*
38159753	2563
41597842	2689
39852456	3598
42357852	2941
43591573	3698

MATERIA		
id_carrea	*id_materia*	*nombre_materia*
A	2	materia1
B	3	materia2
A	1	materia3
E	1	materia4
C	4	materia5

SE_INSCRIBE		
DNI	*id_materia*	*id_carrea*
41597842	1	A
39852456	3	B
38159753	1	E
41597842	2	A
43591573	1	E

```
FK_SE_INSCRIBE_ALUMNO
   ON UPDATE CASCADE
   ON DELETE RESTRICT
FK_SE_INSCRIBE_MATERIA
   ON UPDATE RESTRICT
   ON DELETE CASCADE
```

¿Cuál/es de las siguientes operaciones proceden y cuál/es fallan?

DELETE FROM MATERIA WHERE id_carrera ='C';

Procede

DELETE FROM ALUMNO WHERE DNI = '41597842';

Falla

DELETE FROM MATERIA WHERE id_materia = 3;

Procede

UPDATE SE_INSCRIBE SET id_materia = 2 WHERE DNI = '41597842' AND id_materia = 1;

Falla

UPDATE ALUMNO SET DNI = '39852458' WHERE DNI = '39852456'';

Procede

UPDATE MATERIA SET id_materia = '3' WHERE id_carrera ='E';

Falla

DELETE FROM SE_INSCRIBE WHERE DNI = '38159753';

Procede

Respuesta correcta

La respuesta correcta es:
DELETE FROM MATERIA WHERE id_carrera ='C'; → Procede,

DELETE FROM ALUMNO WHERE DNI = '41597842';

→ Falla,

DELETE FROM MATERIA WHERE id_materia = 3; → Procede,

UPDATE SE_INSCRIBE SET id_materia = 2 WHERE DNI = '41597842' AND id_materia = 1;

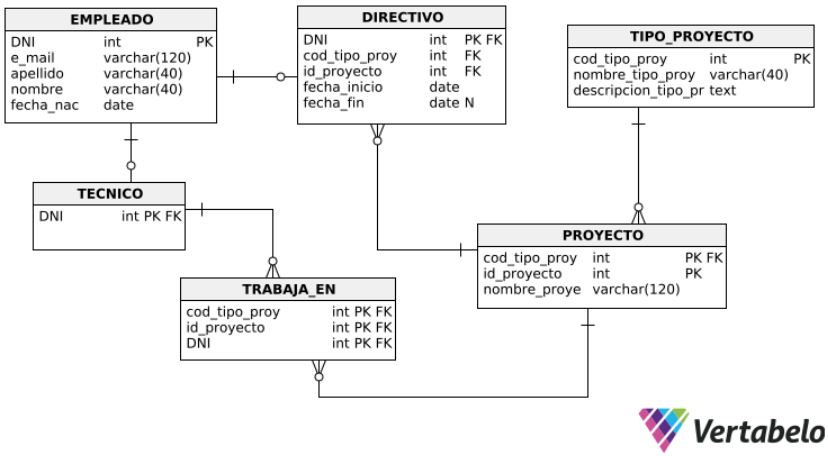
→ Falla,

UPDATE ALUMNO SET DNI = '39852458' WHERE DNI = '39852456''; → Procede,
UPDATE MATERIA SET id_materia = '3' WHERE id_carrera ='E'; → Falla,
DELETE FROM SE_INSCRIBE WHERE DNI = '38159753'; → Procede

Pregunta 3

Finalizado

Se puntúa como 0 sobre 1,00



Tiempo estimado 15 min

Para el esquema de la figura y dadas las siguientes definiciones de vistas:

```
CREATE OR REPLACE VIEW v_emp_mail
AS
SELECT DNI, e_mail, apellido, nombre, fecha_nac
FROM empleado
WHERE e_mail not like '%hotmail%';

CREATE OR REPLACE VIEW v_mail_menor
AS
SELECT *
FROM v_emp_mail
WHERE DNI < 23456789
WITH LOCAL CHECK OPTION;

CREATE OR REPLACE VIEW v_menor_parcial
AS
SELECT *
FROM v_mail_menor
WHERE apellido like '%ado'
WITH LOCAL CHECK OPTION;
```

Para las siguientes sentencias ejecutadas de manera independiente señalar las opciones que son VERDADERAS. Nota: Las respuestas incorrectas restan del puntaje total. Tenga cuidado al cortar y pegar las sentencias con las comillas simples ' '.

a. INSERT INTO v_mail_menor (DNI, e_mail, apellido, nombre, fecha_nac)
VALUES (22345678, 'cc@hotmail.com', 'Alvarado', 'J', to_date('20170103','YYYYMMDD'))
NO Procede, da error.

b. INSERT INTO v_mail_menor (DNI, e_mail, apellido, nombre, fecha_nac)
VALUES (22345678, 'cc@hotmail.com', 'Alvarado', 'J', to_date('20170103','YYYYMMDD'))
NO Procede, porque no cumple con la condición de la vista v_emp_mail

c. INSERT INTO v_mail_menor (DNI, e_mail, apellido, nombre, fecha_nac)
VALUES (22345678, 'cc@hotmail.com', 'Alvarado', 'J', to_date('20170103','YYYYMMDD'))
Procede, inserta los datos en la tabla empleado pero no se muestran en la vista v_emp_mail

Es Verdadera porque la operación procede e inserta los datos en la tabla empleado al cumplir con el WHERE de v_mail_menor (LOCAL CHECK OPTION) y no evaluar el WHERE de v_emp_mail por no tener opción de chequeo en su definición. Luego, la tupla no se muestra en la vista v_emp_mail porque e_mail contiene "hotmail" y su WHERE pide que no lo contenga

d.
INSERT INTO v_mail_menor (DNI, e_mail, apellido, nombre, fecha_nac)
VALUES (22345678, 'cc@hotmail.com', 'Alvarado', 'J', to_date('20170103','YYYYMMDD'))
Procede, inserta los datos en la tabla empleado y se muestran en todas las vistas

e. INSERT INTO v_mail_menor (DNI, e_mail, apellido, nombre, fecha_nac)
VALUES (22345678, 'cc@hotmail.com', 'Alvarado', 'J', to_date('20170103','YYYYMMDD'))
Procede, inserta los datos en la tabla empleado y se muestran en la vista v_emp_mail y en la vista v_mail_menor

f. INSERT INTO v_mail_menor (DNI, e_mail, apellido, nombre, fecha_nac)
VALUES (22345678, 'cc@hotmail.com', 'Alvarado', 'J', to_date('20170103','YYYYMMDD'))
Procede, inserta los datos en la tabla empleado pero no se muestran en la vista v_mail_menor

Respuesta parcialmente correcta.
Ha seleccionado correctamente 1.
Las respuestas correctas son:

INSERT INTO v_mail_menor (DNI, e_mail, apellido, nombre, fecha_nac)
VALUES (22345678, 'cc@hotmail.com', 'Alvarado', 'J', to_date('20170103','YYYYMMDD'))
Procede, inserta los datos en la tabla empleado pero no se muestran en la vista v_mail_menor
,

INSERT INTO v_mail_menor (DNI, e_mail, apellido, nombre, fecha_nac)
VALUES (22345678, 'cc@hotmail.com', 'Alvarado', 'J', to_date('20170103','YYYYMMDD'))
Procede, inserta los datos en la tabla empleado pero no se muestran en la vista v_emp_mail

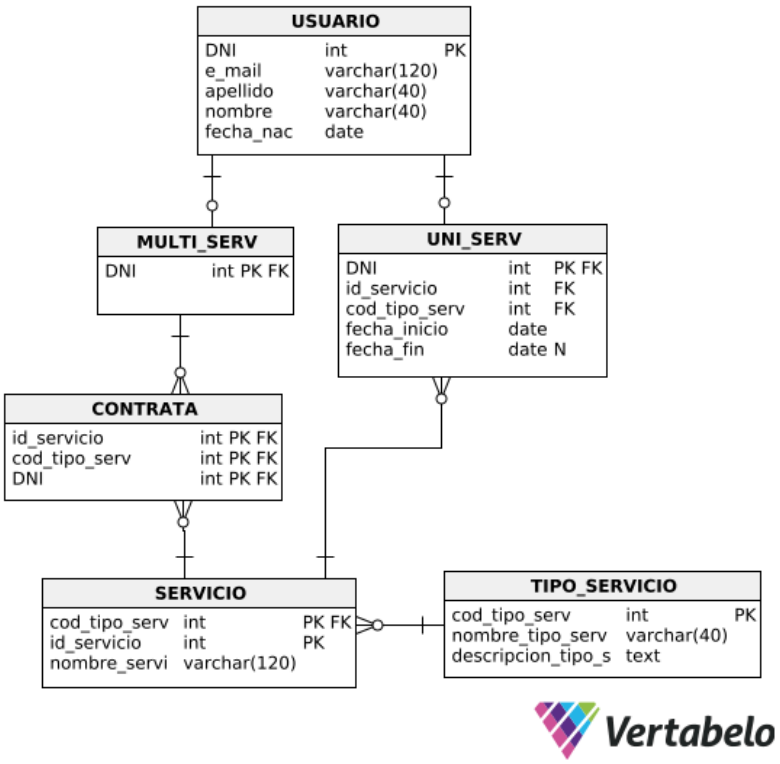
Pregunta 4

Finalizado

Se puntúa como 0 sobre 1,00

Tiempo estimado 45 min

Dado el siguiente esquema - Script de creación de tablas



1) De la sentencia Declarativa mas restrictuva que controle lo siguiente:

- Un Servicio no puede ser contratado en las dos modalidades de Servicio

2) En caso de que no pueda ser implementado en PostgreSQL declarativamente, de la solución procedural mas eficiente.

```
-- 1) CHECK Global
CREATE ASSERTION servicio_modalidad
CHECK (NOT EXISTS (
       SELECT 1
       FROM CONTRATA c
       JOIN UNI_SERV us ON (c.cod_tipo_serv = us.cod_tipo_serv
       AND c.id_servicio = us.id_servicio)
));

-- 2) TRIGGER
-- CONTRATA
CREATE FUNCTION fn_contrata_servicio_no_duplicado()
RETURNS TRIGGER AS $$
BEGIN
    IF EXISTS (
        SELECT 1
        FROM UNI_SERV us
        WHERE us.cod_tipo_serv = NEW.cod_tipo_serv
        AND us.id_servicio = NEW.id_servicio
    ) THEN RAISE EXCEPTION 'ERROR! El Servicio ya existe en UNI_SERV.';
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER tr_contrata_servicio_no_duplicado
BEFORE INSERT OR UPDATE OF cod_tipo_serv, id_servicio ON CONTRATA
FOR EACH ROW EXECUTE FUNCTION fn_contrata_servicio_no_duplicado();

-- UNI_SERV
```

1)

```
CREATE ASSERTION ck_servicio_no_contratado_en_ambas_modalidades
      CHECK ( NOT EXISTS( SELECT 1
                  FROM contrata
                  WHERE (id_servicio, cod_tipo_serv)
                      IN (SELECT id_servicio, cod_tipo_serv
                              FROM uni_serv)));
```

```
CREATE ASSERTION ck_servicio_no_contratado_en_ambas_modalidades
      CHECK ( NOT EXISTS( SELECT 1
                  FROM contrata c
                  JOIN uni_serv u USING (id_servicio, cod_tipo_serv)));
```

2) NOTA: Tener en cuenta lo siguiente : Que exista un DNI en tabla Multi_Serv, y el mismo DNI en Uni_Serv, no implica que en Contrata y en Uni_Serv aparezcan relacionadas con el MISMO servicio.

```
CREATE OR REPLACE FUNCTION verificar_contratacion() RETURNS trigger AS $$
   BEGIN
        IF (TG_TABLE_NAME = 'uni_serv' ) THEN
                IF EXISTS (SELECT 1 FROM contrata
                           WHERE id_servicio = NEW.id_servicio
                           AND cod_tipo_serv = NEW.cod_tipo_serv
                           ) THEN
                          RAISE EXCEPTION 'El servicio % de tipo % ya está contratado en modalidad múltiple',
NEW.id_servicio, NEW.cod_tipo_serv;
                END IF;
        ELSIF (TG_TABLE_NAME = 'contrata') THEN
              IF EXISTS (SELECT 1 FROM uni_serv
                         WHERE id_servicio = NEW.id_servicio
                         AND cod_tipo_serv = NEW.cod_tipo_serv
                         ) THEN
                         RAISE EXCEPTION 'El servicio % de tipo %  ya está contratado en modalidad única',
NEW.id_servicio , NEW.cod_tipo_serv;
               END IF;
        END IF;
        RETURN NEW;
    END $$ LANGUAGE 'plpgsql';
```

```
CREATE TRIGGER TRG_B_INS_UPD_uni_serv
BEFORE INSERT OR UPDATE OF id_servicio, cod_tipo_serv
ON uni_serv
FOR EACH ROW
EXECUTE FUNCTION verificar_contratacion();
```

```
CREATE TRIGGER TRG_B_INS_UPD__contrata
BEFORE INSERT OR UPDATE OF id_servicio, cod_tipo_serv
ON contrata
FOR EACH ROW
EXECUTE FUNCTION verificar_contratacion();
```

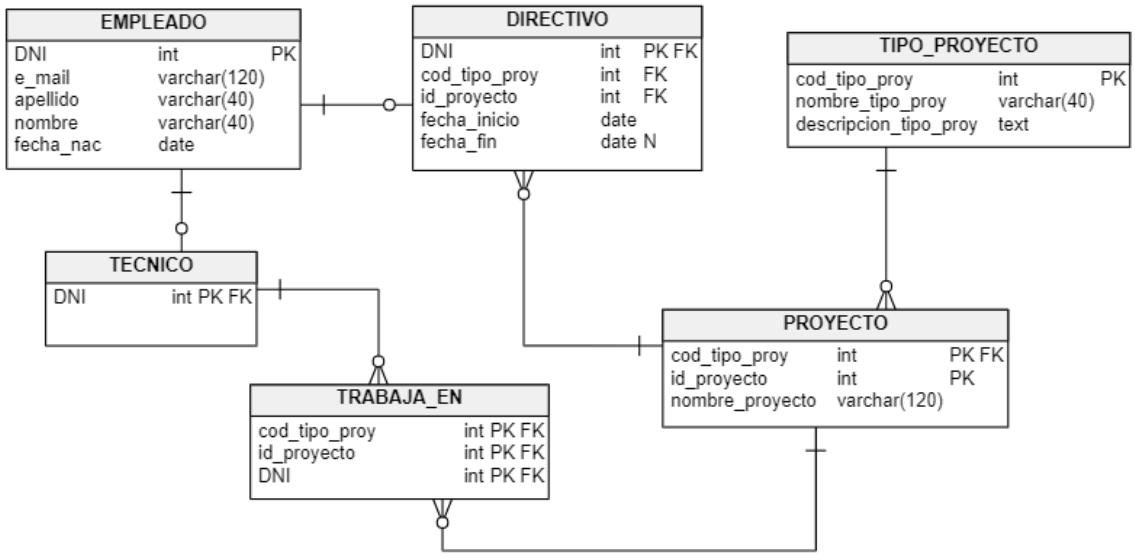
Pregunta 5

Finalizado

Se puntúa como 0 sobre 1,00

Tiempo estimado 25 min

Considere el siguiente esquema (script creación solo de tablas) correspondiente a un sistema de contratación de servicios, cuyo diagrama en Vertabelo es el siguiente:



Considere las siguientes sentencias SQL de declaración de claves primarias (PK) y claves extranjeras (FK) sobre el esquema dado:

```
-- Primary Key
ALTER TABLE DIRECTIVO ADD CONSTRAINT PK_DIRECTIVO PRIMARY KEY (DNI);
ALTER TABLE EMPLEADO ADD CONSTRAINT PK_EMPLEADO PRIMARY KEY (DNI);
ALTER TABLE TECNICO ADD CONSTRAINT PK_TECNICO PRIMARY KEY (DNI);
ALTER TABLE TIPO_PROYECTO ADD  CONSTRAINT PK_TIPO_PROYECTO PRIMARY KEY (cod_tipo_proy);
```

```
-- foreign keys
ALTER TABLE PROYECTO ADD CONSTRAINT FK_PROYECTO_TIPO_PROYECTO
    FOREIGN KEY (cod_tipo_proy)
    REFERENCES TIPO_PROYECTO (cod_tipo_proy);
ALTER TABLE TRABAJA_EN ADD CONSTRAINT FK_TRABAJA_EN_PROYECTO
    FOREIGN KEY (id_proyecto, cod_tipo_proy)
    REFERENCES PROYECTO (id_proyecto, cod_tipo_proy);
ALTER TABLE TRABAJA_EN ADD CONSTRAINT FK_TRABAJA_EN_TECNICO
    FOREIGN KEY (DNI)
    REFERENCES TECNICO (DNI);
```

Escriba las sentencias sentencias SQL de **declaración de claves primarias (PK), claves alterativas (AK) y claves extranjeras (FK)** que faltan teniendo en cuenta los siguiente:
a) Deben definirse las acciones referenciales para que cada vez que se elimine o modifique (PK) de un empleado debe hacerse también de las tablas subtipo de la jerarquía.
b) Los e-mails de los empleados deben ser únicos.

```
-- Primary Key
ALTER TABLE DIRECTIVO ADD CONSTRAINT PK_DIRECTIVO PRIMARY KEY (DNI);
ALTER TABLE EMPLEADO ADD CONSTRAINT PK_EMPLEADO PRIMARY KEY (DNI);
ALTER TABLE TECNICO ADD CONSTRAINT PK_TECNICO PRIMARY KEY (DNI);
ALTER TABLE TIPO_PROYECTO ADD  CONSTRAINT PK_TIPO_PROYECTO PRIMARY KEY (cod_tipo_proy);

-- Resuelto
ALTER TABLE TRABAJA_EN ADD CONSTRAINT PK_TRABAJA_EN PRIMARY KEY (cod_tipo_proy, id_proyecto, DNI);
ALTER TABLE PROYECTO ADD CONSTRAINT PK_PROYECTO PRIMARY KEY (id_proyecto, cod_tipo_proy);

-- Foreign Keys
ALTER TABLE PROYECTO ADD CONSTRAINT FK_PROYECTO_TIPO_PROYECTO
    FOREIGN KEY (cod_tipo_proy)
    REFERENCES TIPO_PROYECTO (cod_tipo_proy);
ALTER TABLE TRABAJA_EN ADD CONSTRAINT FK_TRABAJA_EN_PROYECTO
    FOREIGN KEY (id_proyecto, cod_tipo_proy)
    REFERENCES PROYECTO (id_proyecto, cod_tipo_proy);
ALTER TABLE TRABAJA_EN ADD CONSTRAINT FK_TRABAJA_EN_TECNICO
    FOREIGN KEY (DNI)
    REFERENCES TECNICO (DNI)
    -- Resuelto
    ON DELETE CASCADE
    ON UPDATE CASCADE;

-- Resuelto
ALTER TABLE TECNICO ADD CONSTRAINT FK_TECNICO_EMPLEADO
    FOREIGN KEY (DNI)
    REFERENCES EMPLEADO (DNI)
    ON DELETE CASCADE
    ON UPDATE CASCADE;
ALTER TABLE DIRECTIVO ADD CONSTRAINT FK_DIRECTIVO_EMPLEADO
    FOREIGN KEY (DNI)
    REFERENCES EMPLEADO (DNI)
    ON DELETE CASCADE
    ON UPDATE CASCADE;
ALTER TABLE DIRECTIVO ADD CONSTRAINT FK_DIRECTIVO_PROYECTO
    FOREIGN KEY (id_proyecto, cod_tipo_proy)
    REFERENCES PROYECTO (id_proyecto, cod_tipo_proy);
```

```
-- En EMPLEADO esto ya cumple que el e_mail sea único
-- CONSTRAINT AK_EMPLEADO UNIQUE (e_mail) NOT DEFERRABLE  INITIALLY IMMEDIATE
```

```
ALTER TABLE PROYECTO ADD CONSTRAINT PK_PROYECTO PRIMARY KEY (id_proyecto,cod_tipo_proy);
```

```
ALTER TABLE TRABAJA_EN ADD CONSTRAINT PK_TRABAJA_EN PRIMARY KEY (id_proyecto,cod_tipo_proy,DNI);
```

```
ALTER TABLE DIRECTIVO ADD CONSTRAINT FK_DIRECTIVO_EMPLEADO
    FOREIGN KEY (DNI)
    REFERENCES EMPLEADO (DNI)
    ON DELETE  CASCADE
    ON UPDATE  CASCADE;
```

```
ALTER TABLE DIRECTIVO ADD CONSTRAINT FK_DIRECTIVO_PROYECTO
    FOREIGN KEY (id_proyecto, cod_tipo_proy)
    REFERENCES PROYECTO (id_proyecto, cod_tipo_proy);
```

```
ALTER TABLE TECNICO ADD CONSTRAINT FK_TECNICO_EMPLEADO
    FOREIGN KEY (DNI)
    REFERENCES EMPLEADO (DNI)
    ON DELETE  CASCADE
    ON UPDATE  CASCADE;
```

```
ALTER TABLE EMPLEADO ADD CONSTRAINT AK_EMPLEADO UNIQUE (e_mail);
```

Pregunta 6

Finalizado

Se puntúa como 0 sobre 1,00

Utilizando el esquema unc_esq_voluntario. De qué país(nombre) hay mayor cantidad de voluntarios que realizan una tarea terminada en REP.

Tiempo estimado 20 min

- a. Ninguna de las opciones es correcta
- b. Canadá
- c. Estados Unidos
- d. Alemania
- e. Reino Unido

Respuesta correcta

SQL que lo resuelve:

```
select p.id_pais, p.nombre_pais  as "País",count(V.nro_voluntario) from pais p
join direccion d on d.id_pais =p.id_pais
join institucion i on i.id_direccion = d.id_direccion
join voluntario v on v.id_institucion = i.id_institucion
join tarea t on v.id_tarea = t.id_tarea
where t.id_tarea  like '%REP'
group by p.id_pais, p.nombre_pais
order by 3 desc
```

La respuesta correcta es:

Reino Unido

Actividad previa

◀ Práctico 9 - SQL Avanzado

Ir a...

Mantente en contacto

Facultad, Pabellón Central Paraje Arroyo Seco. Campus Universitario. (B7001BBO) Tandil. Buenos Aires, Argentina

https://exa.unicen.edu.ar/

(+54) (0249) 438-5650 Conmutador: int. 2000

moodle@exa.unicen.edu.ar

Descargar la app para dispositivos móviles