CENTRE ESPECÍFIC D´EDUCACIÓ A DISTÀNCIA DE LA COMUNITAT VALENCIANA

EXAMEN MODELO B

2ª EVALUACIÓN

BASES DE DATOS 22/23

CFGS DAW

CONSULTAS, EXTENSIONES, GESTIÓN Y NOSQL

SOLUCIONES

Autores:

Abelardo Martínez y Pau Miñana

Licencia Creative Commons

Reconocimiento – NoComercial – CompartirIgual (by-nc-sa): No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

BASES DE DATOS RELACIONALES

[4 PUNTOS] PARTE 1: CONSULTAS

1. Imprime el código, el nombre completo y la fecha de antigüedad de los oficiales imperiales con antigüedad anterior a 1/01/2000 cuyo nombre comienza por "A" o "B". Ordena el listado por fecha de antigüedad (de más antigua a más reciente) y nombre y apellidos (alfabético).

   Se admite cualquier variante de nombre y apellidos, como por ejemplo CONCAT.

```
SELECT O.codigo, O.nombre, O.apellidos, O.fecha_ant
FROM OficialImperial O
WHERE O.fecha_ant < '2000-01-01'
AND (O.nombre LIKE 'A%' OR O.nombre LIKE 'B%')
ORDER BY O.fecha_ant, O.nombre, O.apellidos;
```

2. Listar nombre y apellidos (concatenados por coma) de los oficiales imperiales que no manden troopers, ordenado por nombre y apellidos.

Opción 1

```
SELECT CONCAT(O.nombre,', ', O.apellidos) AS Oficial
FROM OficialImperial O
WHERE O.codigo NOT IN
 (SELECT M.cod_oficial
    FROM Mandar M)
ORDER BY CONCAT(O.nombre,', ', O.apellidos);
```

Opción 2

```
SELECT CONCAT(O.nombre,', ', O.apellidos) AS Oficial
FROM OficialImperial O
WHERE NOT EXISTS
 (SELECT M.cod_oficial
    FROM Mandar M
   WHERE O.codigo = M.cod_oficial)
ORDER BY 1;
```

Opción 3

```
SELECT CONCAT(O.nombre, ', ', O.apellidos) AS Oficial
FROM OficialImperial O
 LEFT JOIN Mandar M ON O.codigo = M.cod_oficial
WHERE M.cod_oficial IS NULL
ORDER BY 1;
```

3. Imprime sin repeticiones el nombre completo del oficial u oficiales imperiales que hayan usado una llave en la fecha más reciente.

   Se admite cualquier variante de nombre y apellidos, como por ejemplo CONCAT.

Opción 1

```
SELECT DISTINCT(CONCAT(O.nombre,', ', O.apellidos)) AS Oficial
FROM OficialImperial O, Llaves LL1
WHERE O.codigo = LL1.codigo
  AND LL1.fecha_ult =
 (SELECT MAX(LL2.fecha_ult)
    FROM Llaves LL2);
```

Opción 2

```
SELECT DISTINCT(CONCAT(O.nombre,', ', O.apellidos)) AS Oficial
FROM OficialImperial O, Llaves LL1
WHERE O.codigo = LL1.codigo
  AND LL1.fecha_ult >= ALL
 (SELECT LL2.fecha_ult
    FROM Llaves LL2);
```

4. Crea una vista (con nombre "Turnos_oficial_trooper") para obtener el código del oficial imperial, fecha, turno, y código, serie y rango de los troopers que son/han sido mandados por un oficial. Ordena la vista por fecha de más antigua a más reciente. Una vez creada, realiza una consulta sobre esa vista para obtener el nombre completo de los oficiales imperiales y número de los troopers que manda, que han sido mandados en el mes de enero de 2023 y que sean del turno de noche.

Vista

```
DROP VIEW IF EXISTS Turnos_oficial_trooper;
```

```
CREATE VIEW Turnos_oficial_trooper AS
SELECT M.cod_oficial, M.fecha, M.turno, M.cod_trooper, T.serie, T.rango
FROM Mandar M, Trooper T
WHERE M.cod_trooper = T.codigo
ORDER BY M.fecha;

SELECT * FROM Turnos_oficial_trooper;
```

Se admite cualquier variante de nombre y apellidos, como por ejemplo CONCAT.

Consulta

```
SELECT O.nombre, O.apellidos, COUNT(T.cod_trooper) AS Total_troopers_noche
FROM Turnos_oficial_trooper T, OficialImperial O
WHERE T.cod_oficial = O.codigo
AND T.turno = 'Noche'
AND T.fecha BETWEEN '2023-01-01' AND '2023-01-31'
GROUP BY T.cod_oficial,  O.nombre, O.apellidos;
```

5. Obtener el nombre completo y el total de centros del oficial imperial u oficiales imperiales que más centros de control dirigen.

Se admite cualquier variante de nombre y apellidos, como por ejemplo CONCAT.

Opción 1

```
SELECT CONCAT (O.nombre, ', ', O.apellidos) AS Oficial_mas_centros, COUNT(C1.codigo) AS TotalCentros
FROM OficialImperial O, CentroControl C1
WHERE O.codigo = C1.codigo
GROUP BY C1.codigo, CONCAT (O.nombre, ', ', O.apellidos)
HAVING COUNT(C1.codigo) >= ALL
 (SELECT COUNT(C2.codigo)
   FROM CentroControl C2
  GROUP BY C2.codigo);
```

Opción 2

```
SELECT CONCAT(O.nombre,',  ',O.apellidos) AS Oficial_mas_centros, COUNT(C1.codigo) AS TotalCentros
FROM OficialImperial O
```

```
INNER JOIN CentroControl C1 ON O.codigo = C1.codigo
GROUP BY O.codigo, CONCAT(O.nombre, ', ', O.apellidos)
HAVING COUNT(C1.codigo) =
  (SELECT MAX(TotalCentros)
    FROM (SELECT COUNT(C2.codigo) AS TotalCentros
            FROM CentroControl C2
            GROUP BY C2.codigo) AS TotalCentros);
```

[3 PUNTOS] PARTE 2: EXTENSIONES

1. Crea un procedimiento "pFechaOfiTroopTurno" para dada una fecha y un turno, contar el total de oficiales y troopers que hay/ha habido en ese turno y fecha, y listar los apellidos y nombre (concatenados con coma) de los oficiales imperiales ordenado por apellidos y nombre (alfabético). Muestra un mensaje de error si el turno no es 'Mañana', 'Tarde' o 'Noche'.

```
DROP PROCEDURE IF EXISTS pFechaOfiTroopTurno;
DELIMITER ||
CREATE PROCEDURE pFechaOfiTroopTurno (IN dtFecha DATE, IN vchTurno VARCHAR(10), OUT iTotal INT)
BEGIN
  DECLARE inumOficiales INT;
  DECLARE inumTroopers INT;

  IF vchTurno NOT IN ('Mañana','Tarde','Noche') THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Parámetros inesperados\n*****\n====> Indica Mañana, Tarde o Noche.\n*****\n';
  END IF;

  SELECT COUNT(DISTINCT(cod_oficial)) INTO inumOficiales
    FROM Mandar
   WHERE fecha = dtFecha
     AND turno = vchTurno);
  SELECT COUNT(DISTINCT(cod_trooper)) INTO inumTroopers
    FROM Mandar
   WHERE fecha = dtFecha
     AND turno = vchTurno);
  SET iTotal = inumOficiales+inumTroopers;

  SELECT DISTINCT(CONCAT(O.apellidos, ',', O.nombre))
    FROM Mandar M, OficialImperial O
   WHERE M.cod_oficial = O.codigo
     AND M.fecha = dtFecha
     AND M.turno = vchTurno
```

```
 ORDER BY 1;
END||
DELIMITER ;


CALL pFechaOfiTroopTurno ('2023-01-20', 'Noche', @TOTAL);
SELECT @TOTAL as Total;
```

2. Crea los triggers necesarios para asegurar que un oficial imperial dirija al menos un centro de control (participación mínima de 1), ignorando las inserciones.

```
-- Antes de borrar centros de control
DROP TRIGGER IF EXISTS tAntesBorrarCentroControl;
DELIMITER ||
CREATE TRIGGER tAntesBorrarCentroControl
BEFORE DELETE ON CentroControl
FOR EACH ROW
BEGIN
  IF (SELECT COUNT(*) FROM CentroControl C
      WHERE C.codigo = OLD.codigo
    ) < 2
  THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Operación no permitida\n*****\n====>
Todo Oficial Imperial debe dirigir, como mínimo, un centro de control.\n*****\n';
END IF;
END||
DELIMITER ;


-- Antes de actualizar centros de control
DROP TRIGGER IF EXISTS tAntesActualizarCentroControl;
DELIMITER ||
CREATE TRIGGER tAntesActualizarCentroControl
BEFORE UPDATE ON CentroControl
FOR EACH ROW
BEGIN
  IF (
    (OLD.codigo <> NEW.codigo)
```

```
    AND
    (SELECT COUNT(*) FROM CentroControl C
      WHERE C.codigo = OLD.codigo
    ) < 2
  )
  THEN
   SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Operación no permitida\n*****\n====>
Todo Oficial Imperial debe dirigir, como mínimo, un centro de control.\n*****\n';
 END IF;
END||
DELIMITER ;
```

[1 PUNTO] PARTE 3: GESTIÓN DE USUARIOS

1. Crea un usuario *administrador* (con contraseña "centro") que tenga permisos completos en la BD desde el servidor y permiso de consulta y actualización en todas las tablas desde la IP de su casa 192.0.68.107.

```
CREATE USER IF NOT EXISTS administrador@localhost IDENTIFIED BY 'centro';
GRANT ALL PRIVILEGES ON baseimperialDB.* TO administrador@localhost;
FLUSH PRIVILEGES;
CREATE USER IF NOT EXISTS administrador@192.0.68.107 IDENTIFIED BY 'centro';
REVOKE ALL PRIVILEGES ON baseimperialDB.* FROM administrador@192.0.68.107;
GRANT SELECT, UPDATE ON baseimperialDB.* TO administrador@192.0.68.107;
FLUSH PRIVILEGES;
```

2. Crea un usuario *personal* (con la misma contraseña anterior) que solo pueda usar el procedimiento creado en el apartado anterior, pero pueda otorgar permiso para usarlo a otros. Desde cualquier IP. *Para dar permisos de uso de funciones/procedimientos el permiso se llama EXECUTE y para estos casos hay que añadir FUNCTION o PROCEDURE después del ON.*

```
CREATE USER IF NOT EXISTS personal@'%' IDENTIFIED BY 'centro';
REVOKE ALL PRIVILEGES ON baseimperialDB.* FROM personal@'%';
GRANT EXECUTE ON PROCEDURE baseimperialDB.pFechaOfiTroopTurno TO personal@'%'
WITH GRANT OPTION;
FLUSH PRIVILEGES;
```

NOSQL

[2 PUNTOS] PARTE 4: MONGODB

Vamos a adaptar para MongoDB solo un extracto de parte de los oficiales imperiales de la Base de datos. Deja que se asignen los _id automáticamente y usa los siguientes datos:

Codigo	Nombre	Apellidos	Fecha_ant	Llaves
"OFI001"	"Almirante"	"Tarkin"	1977-05-04	ARRAY
"OFI002"	"Almirante"	"Thrawn"	1977-05-04	ARRAY

El ARRAY en Llaves contendrá documentos con la llave y la fecha de último uso de la misma:

Llave	Fecha_ult
OFI001	
"Sala mando"	1977-05-04
"Almacenes"	1977-05-04
OFI002	
"Sala mando"	
"Hangares"	

1. Borrar la base de datos si existiese, crearla de nuevo, crear la colección indicada e insertar los documentos indicados. Debes tener en cuenta el tipo de datos más adecuado para cada campo. Una vez insertados debes listar todos los documentos con todos sus campos ordenados por Nombre y Apellidos.

   Entrega el script para realizar los pasos anteriores y, en caso de que importes los datos en lugar de insertarlos, el documento JSON usado para esa importación.

```
// limpiar pantalla (buena praxis)
cls
// conectar con la BD y borrarla (la crea si no existe)
use centromandoMDB
db.dropDatabase()
use centromandoMDB
```

```
// crear una nueva colección
db.createCollection("oficiales")
// crea varios documentos
var j_ofi1 = {codigo: "OFI001", nombre: "Almirante", apellidos: "Tarkin",
        llaves:[
        {llave:"Sala mando", fecha_ult: ISODate("1977-05-04T00:00:00.000Z"},
        {llave:"Almacenes", fecha_ult: ISODate("1977-05-04T00:00:00.000Z"}
        ]}
var j_ofi2 = {codigo: "OFI001", nombre: "Almirante", apellidos: "Thrawn",
        llaves:[
        {llave:"Sala mando"},
        {llave:"Almacenes"}
        ]}
// inserta esos documentos en la colección
// Alternativa: insertOne
db.oficiales.insertMany([j_ofi1, j_ofi2])
//listar todos los documentos
var j_proy       = {nombre:1, apellidos:1, _id:0}
var j_orden     = {nombre:1, apellidos:1}
// consulta
db.oficiales.find({}, j_proy).sort(j_orden)
```

2. Realiza una consulta que muestre el nombre y apellidos del oficial imperial que tenga fecha de uso (Fecha_ult) en cualquiera de sus Llaves o su nombre sea "Almirante", ordenados por Nombre y Apellidos de forma alfabética inversa. Oculta el campo _id.

```
var j_valorA     = {$exists:true}
var j_filtro1    = {"llaves.fecha_ult":j_valorA}
var j_valorB     = {$eq:"Almirante"}
var j_filtro2    = {nombre:j_valorB}
var j_filtroFinal = {$or:[j_filtro1, j_filtro2]}
var j_proy       = {nombre:1, apellidos:1, _id:0}
var j_orden      = {nombre:-1, apellidos:-1}
// consulta
db.oficiales.find(j_filtroFinal, j_proy).sort(j_orden)
```