**CONSULTA 1B: EXISTS VS IN [1 puntos]**

**1A) Sin IN, con EXISTS**

|  |
| --- |
| SELECT d.nombre AS Nombre, d.apellidos AS Apellidos  FROM dawers d  WHERE EXISTS (SELECT \* FROM documentalistas doc, alqdoc a WHERE doc.dni=a.dni AND a.dni=d.dni)  ORDER BY d.nombre, d.apellidos DESC; |

**1B) Con IN, sin EXISTS**

|  |
| --- |
| SELECT d.nombre AS Nombre, d.apellidos AS Apellidos  FROM dawers d, documentalistas doc  WHERE d.dni=doc.dni AND doc.dni IN(SELECT a.dni FROM alqdoc a)  ORDER BY d.nombre DESC , d.apellidos DESC; |

**CONSULTA 2B: CREAR UNA VISTA VS USARLA [1 puntos]**

**2A) Crea la vista y lista todos sus elementos**

|  |
| --- |
| CREATE OR REPLACE VIEW VI\_ALQ\_X\_MUSEOS  AS SELECT a.idalquiler AS ID\_Alquiler, al.fecha AS Fecha, d.dni AS DNI, CONCAT(d.apellidos, "(",d.nombre,")"), m.nombre  FROM alqdoc a, dawers d, museos m, documentalistas doc, alquileres al  WHERE a.dni=d.dni AND a.idmuseo=m.idmuseo AND d.dni=doc.dni AND a.idalquiler=al.idalquiler  ORDER BY m.nombre DESC, d.apellidos DESC, d.nombre DESC; |

**2B) Crea una consulta sobre esa vista**

|  |
| --- |
| SELECT museo, count(museo) AS ALQUILERES  FROM vi\_alq\_x\_museos  HAVING count(museo)>1 AND museo LIKE "%a%"; |

**CONSULTA 3B: CARTESIANO VS JOIN [1 puntos]**

**3A) Sin JOIN**

|  |
| --- |
| SELECT CONCAT(d.apellidos, "(", d.nombre, ")") AS Documentalista, SUM(lad.cantidad) AS "Total Alquileres"  FROM dawers d, lineas\_alqdoc lad, alqdoc ad  WHERE d.dni=ad.dni AND lad.idalquiler=ad.idalquiler  GROUP BY Documentalista  ORDER BY "Total Alquileres" DESC; |

**3B) Con JOIN**

|  |
| --- |
| SELECT CONCAT(d.apellidos, "(", d.nombre, ")") AS Documentalista, SUM(lad.cantidad) AS "Total Alquileres"  FROM dawers d INNER JOIN lineas\_alqdoc lad INNER JOIN alqdoc ad  ON d.dni=ad.dni AND lad.idalquiler=ad.idalquiler  GROUP BY Documentalista  ORDER BY "Total Alquileres" DESC; |

**CONSULTA 4B: MÁXIMO DE UNA SUMA [1 puntos]**

**4A) Muestra todos**

|  |
| --- |
| SELECT m.nombre AS "Película", m.precio AS "Precio unitario", COUNT(laa.numlinea) AS "Unidades Alquiladas"  FROM materiales m, lineas\_alqafi laa, peliculas p  WHERE m.idmaterial=laa.idmaterial AND laa.idmaterial=p.idmaterial; |

**4B) Muestra solo el MÁS VENDIDO (sin usar LIMIT ni TOP)**

|  |
| --- |
| SELECT m.nombre AS "Película", m.precio AS "Precio unitario", COUNT(laa.numlinea) AS "Unidades Alquiladas"  FROM materiales m, lineas\_alqafi laa, peliculas p  WHERE m.idmaterial=laa.idmaterial AND laa.idmaterial=p.idmaterial  HAVING (SELECT "Unidades Alquiladas" as unidades2  FROM materiales m2, lineas\_alqafi laa2, peliculas p2  WHERE m2.idmaterial=laa2.idmaterial AND laa2.idmaterial=p2.idmaterial  HAVING unidades2>="Unidades Alquiladas"); |

**CONSULTA 5B: GROUP BY VS SUBCONSULTAS [1 puntos]**

**5A) Con GROUP BY, sin SUBCONSULTA**

|  |
| --- |
| SELECT d.dni AS "Dawer (DNI)", d.apellidos AS "Dawer (Apellidos)", COUNT(m.idmedio) AS "Número de Medios"  FROM dawers d, medios\_de\_pago m  WHERE d.dni=m.dni AND d.apellidos like "%a" OR d.apellidos LIKE "%e" OR d.apellidos LIKE "%i" OR d.apellidos LIKE "%o" OR d.apellidos LIKE "%u"  GROUP BY d.dni , d.apellidos; |

**5B) Sin GROUP BY, con SUBCONSULTA**

|  |
| --- |
| SELECT d.dni AS "Dawer (DNI)", d.apellidos AS "Dawer (Apellidos)", (SELECT COUNT(m.idmedio)) AS "Numero de medios"  FROM dawers d, medios\_de\_pago m  WHERE d.dni=m.dni AND d.apellidos like "%a" OR d.apellidos LIKE "%e" OR d.apellidos LIKE "%i" OR d.apellidos LIKE "%o" OR d.apellidos LIKE "%u"; |

**SCRIPT 1B: DAWERS POR ALQUILERES ASOCIADOS (1 PUNTOS)**

|  |
| --- |
| USE dawers;  DROP FUNCTION IF EXISTS f\_numventas\_por\_dawer;  DROP FUNCTION IF EXISTS f\_numeuros\_por\_dawer;  DELIMITER $$  CREATE FUNCTION f\_numventas\_por\_dawer (PARAM\_DNI VARCHAR(9)) RETURNS INT  DETERMINISTIC  BEGIN  DECLARE alqafi INT;  DECLARE alqdoc INT;  IF (LENGTH(PARAM\_DNI)<9)  THEN  SIGNAL SQLSTATE "45000" SET MESSAGE\_TEXT = "El DNI contiene menos de 9 caracteres";  ELSE  SET alqafi = (SELECT COUNT(idalquiler)  FROM alqafi a  WHERE a.dni = PARAM\_DNI);  SET alqdoc = (SELECT COUNT(idalquiler)  FROM alqdoc b  WHERE b.dni = PARAM\_DNI);  END IF;  RETURN alqafi + alqdoc;  END$$  CREATE FUNCTION f\_numeuros\_por\_dawer (PARAM\_DNI VARCHAR(9)) RETURNS DECIMAL(10,2)  DETERMINISTIC  BEGIN  DECLARE eurafi DECIMAL(10,2);  DECLARE eurdoc DECIMAL(10,2);  IF (LENGTH(PARAM\_DNI)<9)  THEN  SIGNAL SQLSTATE "45000" SET MESSAGE\_TEXT = "El DNI contiene menos de 9 caracteres";  ELSE  SET eurafi = (SELECT SUM(li.cantidad)  FROM lineas\_alqafi li, alqiafi alq  WHERE li.idalquiler= alq.idalquiler AND alq.dni=PARAM\_DNI);  SET eurdoc = (SELECT SUM(li.cantidad)  FROM lineas\_alqdoc li, alqidoc alq  WHERE li.idalquiler= alq.idalquiler AND alq.dni=PARAM\_DNI);  END IF;  RETURN eurafi + eurdoc;  END $$  DELIMITER ; |

**SCRIPT 2B: LISTAR MATERIALES ALQUILADOS (1 PUNTOS)**

|  |
| --- |
| USE dawers;  DROP FUNCTION IF EXISTS f\_tipo\_material;  DROP PROCEDURE IF EXISTS p\_materiales\_alquilados;  DELIMITER //  CREATE FUNCTION f\_tipo\_material(PARAM\_IDMATERIAL VARCHAR(15)) RETURNS VARCHAR(4)  DETERMINISTIC  BEGIN  IF (PARAM\_IDMATERIAL IN (SELECT idmaterial FROM peliculas))  THEN RETURN "PELI";  ELSEIF (PARAM\_IDMATERIAL IN (SELECT idmaterial FROM documentales))  THEN RETURN "DOC";  ELSE  RETURN "??";  END IF;  END //  CREATE PROCEDURE p\_materiales\_alquilados()  BEGIN  SELECT m.idmaterial AS IDMaterial, m.nombre As Nombre, f\_tipo\_material(m.idmaterial) AS Tipo  FROM materiales m  ORDER BY idmaterial;  END //  DELIMITER ; |

**SCRIPT 3B: PÉRDIDA SEMÁNTICA EN ALQUILERES (1 PUNTOS)**

|  |
| --- |
| USE dawers;  DROP TRIGGER IF EXISTS t\_restriccion\_Afi;  DROP TRIGGER IF EXISTS t\_restriccion\_Doc;  DELIMITER //  CREATE TRIGGER t\_restriccion\_Afi  BEFORE INSERT ON alqafi  FOR EACH ROW  BEGIN  IF NEW.idalquiler IN (SELECT idalquiler FROM alqdoc)  THEN  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = "No es posible añadir un IDAlquiler como Aficionado si ya existe en Documentalista";  END IF;  END //  CREATE TRIGGER t\_restriccion\_Doc  BEFORE INSERT ON alqdoc  FOR EACH ROW  BEGIN  IF NEW.idalquiler IN (SELECT idalquiler FROM alqafi)  THEN  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = “Operación no permitida\n\*\*\*\*\*\n====> Mensaje de error.\n\*\*\*\*\*\n”;  END IF;  END //  DELIMITER ; |

**SCRIPT 4B: TRIGGERS PARA LAS PARTICIPACIONES 1:N (1 PUNTOS)**

|  |
| --- |
| USE dawers;  DROP TRIGGER IF EXISTS t\_restriccion\_medios;  DROP TRIGGER IF EXISTS t\_actu\_medios;  DELIMITER //  CREATE TRIGGER t\_restriccion\_medios  BEFORE DELETE ON medios\_de\_pago  FOR EACH ROW  BEGIN  IF ((SELECT count(\*) FROM dawers  WHERE dni NOT IN (SELECT dni FROM medios\_de\_pago WHERE idmedio<>OLD.idmedio))>0)  THEN  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = "Operación no permitida\n\*\*\*\*\*\n====> Mensaje de error.\n\*\*\*\*\*\n";  END IF ;  END //  CREATE TRIGGER t\_actu\_medios  BEFORE UPDATE ON medios\_de\_pago  FOR EACH ROW  BEGIN  IF (NEW.dni<>OLD.dni)  AND  (SELECT count(\*)  FROM dawers  WHERE dni NOT IN (SELECT dni FROM medios\_de\_pago WHERE idmedio<>OLD.idmedio))>0  THEN  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = "Operación no permitida\n\*\*\*\*\*\n====> Mensaje de error.\n\*\*\*\*\*\n";  END IF ;  END //  DELIMITER ; |

**SCRIPT 5B LINEAS DE PEDIDO CONSECUTIVAS (1 PUNTOS)**

|  |
| --- |
| USE dawers;  DROP TRIGGER IF EXISTS t\_before\_update\_lineas\_alqafi;  DROP TRIGGER IF EXISTS t\_before\_insert\_lineas\_alqafi;  DELIMITER //  CREATE TRIGGER t\_before\_update\_lineas\_alqafi  BEFORE UPDATE ON lineas\_alqafi  FOR EACH ROW  BEGIN  IF (NEW.numlinea<>OLD.numlinea)  THEN  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = "Operación no permitida\n\*\*\*\*\*\n====> Mensaje de error.\n\*\*\*\*\*\n";  END IF;  END //  CREATE TRIGGER t\_before\_insert\_lineas\_alqafi  BEFORE INSERT ON lineas\_alqafi  FOR EACH ROW  BEGIN  IF (NEW.numlinea<=0) OR  ((NOT EXISTS (SELECT \* FROM lineas\_alqafi WHERE idalquiler=NEW.idalquiler AND numlinea=NEW.numlinea-1) AND  NEW.numlinea!=1))  THEN  SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = "Operación no permitida\n\*\*\*\*\*\n====> Mensaje de error.\n\*\*\*\*\*\n";  END IF;  END //  DELIMITER ; |