**Primer Recuperatorio – 28/11/2018 – PARTE PRÁCTICA**

**1.c SQL (60 pts)**

**Seleccionar código de fabricante, nombre fabricante, cantidad de órdenes del fabricante, cantidad total vendida del fabricante,  promedio de las cantidades vendidas de todos los Fabricantes de todos aquellos fabricantes cuyas ventas totales sean mayores al PROMEDIO de las ventas de TODOS los fabricantes.**

**Mostrar el resultado ordenado por cantidad total vendida en forma descendente.**

**IMPORTANTE: No se pueden usar procedures, ni Funciones de usuario.**

**manu\_code manu\_name CantOrdenes Total vendido promedioDeTodoslosFabricantes**

ANZ Anza 11 11081.80 3972.85

SHM Shimara 4 5677.91 3972.85

**2.a Stored Procedure (45 pts)**

**Crear un procedimiento procBorraOC que reciba un número de orden de compra por parámetro y realice la eliminación de la misma y sus ítems.**

**Deberá manejar una transacción y deberá manejar excepciones ante algún error que ocurra.**

**El procedimiento deberá guardar en una tabla de auditoria  auditOC  los siguientes datos  order\_num, order\_date, customer\_num, cantidad\_items, total\_orden (SUM(total\_price)), cant\_productos\_comprados (SUM(quantity)), cantidad de ítems.**

**Ante un error deberá almacenar en una tablas erroresOC, order\_num, order\_date, customer\_num, error\_ocurrido VARCHAR(50).**

**2.e Triggers (55 pts)**

**En el  caso que el cliente (customer\_n…**

**Dada la siguiente tabla CURRENT\_STOCK**

create table CURRENT\_STOCK (

stock\_num smallint not null,

manu\_code char(3) not null,

CURRENT\_AMOUNT integer default 0,

created\_date datetime not null, *-- fecha de creación del registro*

updated\_date datetime not null, *-- última fecha de actualización del registro*

PRIMARY KEY (stock\_num, manu\_code)

);

**Crear un trigger que ante un insert, update o delete en la tabla ITEMS actualice la cantidad CURRENT\_AMOUNT de la tabla CURRENT\_STOCK de forma tal que siempre contenga el stock actual del par (stock\_num, manu\_code).**

**Si la operación es un INSERT se restará la cantidad QUANTITY al CURRENT\_AMOUNT.**

**Si la operación es un DELETE se sumará la cantidad QUANTITY al CURRENT\_AMOUNT.**

**Si la operación es un UPDATE se sumará la cantidad QUANTITY nueva y se restará la anterior al CURRENT\_AMOUNT.**

**Si no existe el par (stock\_num, manu\_code) en la tabla CURRENT\_STOCK debe insertarlo en la tablA CURRENT\_STOCK con el valor inicial de 0 (cero) mas/menos la operación a realizar.**

**Tener en cuenta que las operaciones (INSERTs, DELETEs, UPDATEs) pueden ser masivas.**

**1.c SQL (60 pts)**

* ***Solución 1***

SELECT m.manu\_code,

m.manu\_name,

COUNT(distinct i.order\_num) CantOrdenes,

SUM(quantity \* total\_price) total,

(SELECT SUM(quantity \* total\_price) / COUNT(distinct manu\_code)

FROM items i) promedioDeTodoslosFabricantes

FROM manufact m JOIN items i ON (m.manu\_code = i.manu\_code)

GROUP BY m.manu\_code, m.manu\_name

HAVING SUM(quantity \* total\_price) > (SELECT SUM(quantity \* total\_price) /

COUNT(distinct manu\_code)

FROM items i)

* ***Solución 2***

SELECT m.manu\_code,

m.manu\_name,

COUNT(distinct i.order\_num) CantOrdenes,

SUM(quantity \* total\_price) total,

(SELECT AVG(A.total) total

FROM (SELECT manu\_code, SUM(quantity \* total\_price) total

FROM items i

GROUP BY manu\_code)a) promedioDeTodoslosFabricantes

FROM manufact m JOIN items i ON (m.manu\_code = i.manu\_code)

GROUP BY m.manu\_code, m.manu\_name

HAVING SUM(quantity \* total\_price) > (SELECT avg(A.total) total

FROM (SELECT manu\_code,

SUM(quantity \* total\_price) total

FROM items i

GROUP BY manu\_code)

**2.e Triggers (55 pts)**

CREATE TRIGGER itemsTR ON items

AFTER INSERT, UPDATE, DELETE

AS

BEGIN

--

DECLARE @stock\_num smallint, @manu\_code char(3), @quantityI smallint, @quantityD smallint

DECLARE Actualizados CURSOR FOR

SELECT COALESCE(i.stock\_num, d.stock\_num) stock\_num,

COALESCE(i.manu\_code, d.manu\_code) manu\_code,

i.quantity, d.quantity

FROM inserted i FULL JOIN deleted d ON (i.order\_num = d.order\_num AND i.item\_num = d.item\_num)

OPEN Actualizados

FETCH Actualizados

INTO @stock\_num, @manu\_code, @quantityI, @quantityD

WHILE @@FETCH\_STATUS=0

BEGIN

IF NOT EXISTS (SELECT 1 FROM CURRENT\_STOCK p WHERE p.manu\_code = @manu\_code AND p.stock\_num = @stock\_num)

INSERT INTO CURRENT\_STOCK (stock\_num, manu\_code, Current\_Amount, created\_date, updated\_date)

VALUES (@stock\_num, @manu\_code, 0, GETDATE(), GETDATE());

--

UPDATE CURRENT\_STOCK

SET Current\_Amount = Current\_Amount - COALESCE(@quantityI, 0) + COALESCE(@quantityD, 0),

updated\_date = getdate()

WHERE stock\_num = @stock\_num AND manu\_code = @manu\_code;

FETCH Actualizados

INTO @stock\_num, @manu\_code, @quantityI, @quantityD;

END

CLOSE Actualizados

DEALLOCATE Actualizados

END