



	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)	BASES DE DATOS
 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	PROYECTO:Supermercados HIPERMAXI.	Generación de código
	Autor: Laura Cañizares Herrera, Damián Eduardo Domínguez De Barros, Carla Fernanda Flores Gonzales, Bárbara Valentina García Deus.	alu0101129945 alu0101278353 alu0101138183 alu0101108452
Versión: 1.0	Tiempo invertido: 3h	Fecha : 08/02/21

Entregar un enlace a github al producto desarrollado en cada una de las fases del proyecto, y un archivo zip con el proyecto git.

El Readme de Github debe incluir una descripción de la GENERACIÓN DE CÓDIGO DE COMPONENTES Y PROCEDIMIENTOS (CSI-2)

CÓDIGO DE COMPONENTES
<pre>-- MySQL Script generated by MySQL Workbench -- Fri Dec 18 16:04:11 2020 -- Model: New Model Version: 1.0 -- MySQL Workbench Forward Engineering SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0; SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0; SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION'; -- Schema SUPERMERCADO -- Schema SUPERMERCADO CREATE SCHEMA IF NOT EXISTS `SUPERMERCADO` DEFAULT CHARACTER SET utf8 ; USE `SUPERMERCADO` ; -- Table `SUPERMERCADO`.`supermercado` CREATE TABLE IF NOT EXISTS `SUPERMERCADO`.`supermercado` (`nombre` VARCHAR(20) NOT NULL, `ubicacion` VARCHAR(60) NOT NULL, `hora_cierre` TIME NOT NULL, `hora_apertura` TIME NOT NULL, PRIMARY KEY (`nombre`)) ENGINE = InnoDB;</pre>

	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)	BASES DE DATOS
 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	PROYECTO: Supermercados HIPERMAXI.	Generación de código
	Autor: Laura Cañizares Herrera, Damián Eduardo Domínguez De Barros, Carla Fernanda Flores Gonzales, Bárbara Valentina García Deus.	alu0101129945 alu0101278353 alu0101138183 alu0101108452
Versión: 1.0	Tiempo invertido: 3h	Fecha : 08/02/21

```
-- Table `SUPERMERCADO`.`producto`
-----
CREATE TABLE IF NOT EXISTS `SUPERMERCADO`.`producto` (
  `id_producto` INT NOT NULL,
  `precio` INT NOT NULL,
  `tipo` VARCHAR(45) NOT NULL,
  `stock_general` INT NOT NULL,
  PRIMARY KEY (`id_producto`))
ENGINE = InnoDB;

-----


-- Table `SUPERMERCADO`.`cliente_socio`
-----
CREATE TABLE IF NOT EXISTS `SUPERMERCADO`.`cliente_socio` (
  `codigo_cliente` INT NOT NULL,
  `CP` INT NOT NULL,
  `dni` VARCHAR(9) NOT NULL,
  `dirección` VARCHAR(45) NOT NULL,
  `nombre` VARCHAR(45) NOT NULL,
  `cuenta_bancaria` VARCHAR(45) NOT NULL,
  `bonificación` FLOAT NOT NULL,
  PRIMARY KEY (`codigo_cliente`))
ENGINE = InnoDB;

-----

-- Table `SUPERMERCADO`.`cliente_NoSocio`
-----
CREATE TABLE IF NOT EXISTS `SUPERMERCADO`.`cliente_NoSocio` (
  `codigo_cliente` INT NOT NULL,
  `CP` INT NOT NULL,
  PRIMARY KEY (`codigo_cliente`))
ENGINE = InnoDB;

-----

-- Table `SUPERMERCADO`.`compra`
-----
CREATE TABLE IF NOT EXISTS `SUPERMERCADO`.`compra` (
  `fecha` DATE NOT NULL,
  `id_caja` VARCHAR(10) NOT NULL,
```

	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)	BASES DE DATOS
 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	PROYECTO:Supermercados HIPERMAXI.	Generación de código
	Autor: Laura Cañizares Herrera, Damián Eduardo Domínguez De Barros, Carla Fernanda Flores Gonzales, Bárbara Valentina García Deus.	alu0101129945 alu0101278353 alu0101138183 alu0101108452
Versión: 1.0	Tiempo invertido: 3h	Fecha : 08/02/21


```

`factura` VARCHAR(45) NOT NULL,
`hora` TIME NULL,
`cliente_socio_codigo_cliente` INT NULL,
`cliente_NoSocio_codigo_cliente` INT NULL,
`supermercado_nombre` VARCHAR(20) NOT NULL,
PRIMARY KEY (`factura`),
-- INDEX `fk_compra_cliente_socio1_idx` (`cliente_socio_codigo_cliente` ASC) VISIBLE,
-- INDEX `fk_compra_cliente_NoSocio1_idx` (`cliente_NoSocio_codigo_cliente` ASC) VISIBLE,
-- INDEX `fk_compra_supermercado1_idx` (`supermercado_nombre` ASC) VISIBLE,
CONSTRAINT `fk_compra_cliente_socio1`
  FOREIGN KEY (`cliente_socio_codigo_cliente`)
    REFERENCES `SUPERMERCADO`.`cliente_socio` (`codigo_cliente`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
CONSTRAINT `fk_compra_cliente_NoSocio1`
  FOREIGN KEY (`cliente_NoSocio_codigo_cliente`)
    REFERENCES `SUPERMERCADO`.`cliente_NoSocio` (`codigo_cliente`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
CONSTRAINT `fk_compra_supermercado1`
  FOREIGN KEY (`supermercado_nombre`)
    REFERENCES `SUPERMERCADO`.`supermercado` (`nombre`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

-----
-- Table `SUPERMERCADO`.`proveedor_nacional`
-----
CREATE TABLE IF NOT EXISTS `SUPERMERCADO`.`proveedor_nacional` (
  `nombre` VARCHAR(45) NOT NULL,
  `ubicacion` VARCHAR(80) NOT NULL,
  PRIMARY KEY (`nombre`))
ENGINE = InnoDB;

-----
-- Table `SUPERMERCADO`.`proveedor_extranjero`
-----
CREATE TABLE IF NOT EXISTS `SUPERMERCADO`.`proveedor_extranjero` (

```

	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)	BASES DE DATOS
 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	PROYECTO: Supermercados HIPERMAXI.	Generación de código
	Autor: Laura Cañizares Herrera, Damián Eduardo Domínguez De Barros, Carla Fernanda Flores Gonzales, Bárbara Valentina García Deus.	alu0101129945 alu0101278353 alu0101138183 alu0101108452
Versión: 1.0	Tiempo invertido: 3h	Fecha : 08/02/21


```

`nombre` VARCHAR(45) NOT NULL,
`ubicacion` VARCHAR(80) NOT NULL,
PRIMARY KEY (`nombre`))
ENGINE = InnoDB;

-----
-- Table `SUPERMERCADO`.`supermercado_has_producto`
-----
CREATE TABLE IF NOT EXISTS `SUPERMERCADO`.`supermercado_has_producto` (
  `supermercado_nombre` VARCHAR(20) NOT NULL,
  `producto_id_producto` INT NOT NULL,
  `stock_almacen` INT NOT NULL,
  PRIMARY KEY (`supermercado_nombre`, `producto_id_producto`),
  -- INDEX `fk_supermercado_has_producto_producto1_idx` (`producto_id_producto` ASC)
  VISIBLE,
  -- INDEX `fk_supermercado_has_producto_supermercado1_idx` (`supermercado_nombre`
  ASC) VISIBLE,
  CONSTRAINT `fk_supermercado_has_producto_supermercado1`
    FOREIGN KEY (`supermercado_nombre`)
    REFERENCES `SUPERMERCADO`.`supermercado` (`nombre`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_supermercado_has_producto_producto1`
    FOREIGN KEY (`producto_id_producto`)
    REFERENCES `SUPERMERCADO`.`producto` (`id_producto`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

-----
-- Table `SUPERMERCADO`.`compra_has_producto`
-----
CREATE TABLE IF NOT EXISTS `SUPERMERCADO`.`compra_has_producto` (
  `compra_factura` VARCHAR(45) NOT NULL,
  `producto_id_producto` INT NOT NULL,
  `cantidad_producto` INT NOT NULL,
  PRIMARY KEY (`compra_factura`, `producto_id_producto`),
  -- INDEX `fk_compra_has_producto_producto1_idx` (`producto_id_producto` ASC) VISIBLE,
  -- INDEX `fk_compra_has_producto_compra1_idx` (`compra_factura` ASC) VISIBLE,
  CONSTRAINT `fk_compra_has_producto_compra1`

```

	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)	BASES DE DATOS
 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	PROYECTO:Supermercados HIPERMAXI.	Generación de código
	Autor: Laura Cañizares Herrera, Damián Eduardo Domínguez De Barros, Carla Fernanda Flores Gonzales, Bárbara Valentina García Deus.	alu0101129945 alu0101278353 alu0101138183 alu0101108452
Versión: 1.0	Tiempo invertido: 3h	Fecha : 08/02/21


```

FOREIGN KEY (`compra_factura`)
REFERENCES `SUPERMERCADO`.`compra` (`factura`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `fk_compra_has_producto_producto1`
FOREIGN KEY (`producto_id_producto`)
REFERENCES `SUPERMERCADO`.`producto` (`id_producto`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

-----
-- Table `SUPERMERCADO`.`producto_has_proveedor_nacional`
-----
CREATE TABLE IF NOT EXISTS `SUPERMERCADO`.`producto_has_proveedor_nacional` (
  `producto_id_producto` INT NOT NULL,
  `proveedor_nacional_nombre` VARCHAR(45) NOT NULL,
  `cantidad` INT NOT NULL,
  PRIMARY KEY (`producto_id_producto`, `proveedor_nacional_nombre`),
  -- INDEX `fk_producto_has_proveedor_nacional_proveedor_nacional1_idx`
  (`proveedor_nacional_nombre` ASC) VISIBLE,
  -- INDEX `fk_producto_has_proveedor_nacional_producto1_idx` (`producto_id_producto` ASC)
  VISIBLE,
  CONSTRAINT `fk_producto_has_proveedor_nacional_producto1`
  FOREIGN KEY (`producto_id_producto`)
  REFERENCES `SUPERMERCADO`.`producto` (`id_producto`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION,
  CONSTRAINT `fk_producto_has_proveedor_nacional_proveedor_nacional1`
  FOREIGN KEY (`proveedor_nacional_nombre`)
  REFERENCES `SUPERMERCADO`.`proveedor_nacional` (`nombre`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION)
ENGINE = InnoDB;

-----
-- Table `SUPERMERCADO`.`producto_has_proveedor_extranjero`
-----
CREATE TABLE IF NOT EXISTS `SUPERMERCADO`.`producto_has_proveedor_extranjero` (
  `producto_id_producto` INT NOT NULL,

```

 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)	BASES DE DATOS
	PROYECTO:Supermercados HIPERMAXI.	Generación de código
	Autor: Laura Cañizares Herrera, Damián Eduardo Domínguez De Barros, Carla Fernanda Flores Gonzales, Bárbara Valentina García Deus.	alu0101129945 alu0101278353 alu0101138183 alu0101108452
Versión: 1.0	Tiempo invertido: 3h	Fecha : 08/02/21

```

`proveedor_extranjero_nombre` VARCHAR(45) NOT NULL,
`cantidad` INT NOT NULL,
PRIMARY KEY (`producto_id_producto`, `proveedor_extranjero_nombre`),
--          INDEX      `fk_producto_has_proveedor_extranjero_proveedor_extranjero1_idx`
(`proveedor_extranjero_nombre` ASC) VISIBLE,
--  INDEX `fk_producto_has_proveedor_extranjero_producto1_idx` (`producto_id_producto`
ASC) VISIBLE,
CONSTRAINT `fk_producto_has_proveedor_extranjero_producto1`
FOREIGN KEY (`producto_id_producto`)
REFERENCES `SUPERMERCADO`.`producto` (`id_producto`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `fk_producto_has_proveedor_extranjero_proveedor_extranjero1`
FOREIGN KEY (`proveedor_extranjero_nombre`)
REFERENCES `SUPERMERCADO`.`proveedor_extranjero` (`nombre`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

CÓDIGO DE PROCEDIMIENTOS DE OPERACIÓN Y SEGURIDAD

- [Trigger de stock:](#)

```


DELIMITER $$
USE `SUPERMERCADO`$$
CREATE DEFINER = CURRENT_USER TRIGGER `SUPERMERCADO`.`stock_BEFORE_INSERT` BEFORE INSERT ON
`compra_has_producto` FOR EACH ROW
BEGIN
    SET @producto = NEW.producto_id_producto;
    SET @cantidad_producto = NEW.cantidad_producto;

    select supermercado_nombre into @supermercado from compra where factura = new.compra_factura;

    SELECT stock_almacen INTO @stock_almacen FROM supermercado_has_producto WHERE producto_id_producto =
    @producto AND supermercado_nombre = @supermercado;
    IF @stock_almacen >= @cantidad_producto THEN
        UPDATE supermercado_has_producto SET stock_almacen = @stock_almacen - @cantidad_producto WHERE
        producto_id_producto = @producto AND supermercado_nombre = @supermercado;
    ELSE
        SIGNAL SQLSTATE 'ERROR' SET MESSAGE_TEXT = 'No hay stock suficiente del producto';
    END IF;
END$$
DELIMITER ;

```

En este trigger lo primero que realizamos es obtener el ID del producto y la cantidad que se quiere comprar. Luego procedemos a seleccionar el nombre del supermercado para poder ver el stock disponible de los productos.

 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI) PROYECTO:Supermercados HIPERMAXI.	BASES DE DATOS Generación de código
	Autor: Laura Cañizares Herrera, Damián Eduardo Domínguez De Barros, Carla Fernanda Flores Gonzales, Bárbara Valentina García Deus.	alu0101129945 alu0101278353 alu0101138183 alu0101108452
Versión: 1.0	Tiempo invertido: 3h	Fecha : 08/02/21

En el stock del almacén comprobamos que el ID del producto de supermercado_has_producto sea igual al ID del producto comprado para poder comprobar las cantidades restantes del producto.

A continuación, verificamos que la cantidad de producto del almacén sea mayor o igual que la cantidad del producto que se quiere comprar. Si es así, se procede a la actualización del stock del almacén haciendo la diferencia entre el total de productos comprado y lo que resta en el almacén. En otro caso, se emitirá el mensaje de que no hay stock suficiente del producto.

- [Trigger bonificación:](#)


```
USE `SUPERMERCADO`$$
CREATE DEFINER = CURRENT_USER TRIGGER `SUPERMERCADO`.`compra_bonificacion_AFTER_INSERT` AFTER INSERT ON `compra_has_producto` FOR
EACH ROW
BEGIN
    DECLARE i INT DEFAULT 0;
    select factura into @factura from compra where factura = new.compra_factura;
    select cliente_socio_codigo_cliente into @cliente from compra where factura = new.compra_factura;
    set @total = 0;
    if @cliente is not NULL then
        select bonificación into @bonificacion from cliente_socio where codigo_cliente = @cliente;

        select count(compra_factura) into @n from compra_has_producto where compra_factura = @factura;
        set i = 0;
        while i < @n DO
            select producto_id_producto into @id_producto from compra_has_producto where compra_factura = @factura limit i,1;
            select precio into @precio from producto where id_producto = @id_producto;
            select cantidad_producto into @cantidad from compra_has_producto where compra_factura = @factura limit i,1;
            set @total := @total + (@cantidad * @precio);
            set i := i + 1;
        end while;

        if @bonificacion > 1.0 then
            SET @total_con_descuento = @total / @bonificacion;
        end if;
        if @total > 100 then
            set @bonificacion = @bonificacion + 0.01;
        end if;
        UPDATE cliente_socio set bonificación = @bonificacion where codigo_cliente = @cliente;
    end if;
END$$
```

En este trigger lo primero que hacemos es seleccionar la factura y el código del cliente de la nueva compra. Comprobamos que la compra la ha hecho un cliente y procedemos a calcular el valor de su compra.

Primero obtenemos su bonificación actual. Después contamos cuántos productos ha comprado. Esto lo sabemos porque para la misma factura van a existir distintas entradas con distintos productos.

	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI)	BASES DE DATOS
 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	PROYECTO: Supermercados HIPERMAXI.	Generación de código
	Autor: Laura Cañizares Herrera, Damián Eduardo Domínguez De Barros, Carla Fernanda Flores Gonzales, Bárbara Valentina García Deus.	alu0101129945 alu0101278353 alu0101138183 alu0101108452
Versión: 1.0	Tiempo invertido: 3h	Fecha : 08/02/21

Al saber cuántos productos compró, entramos en un bucle en el que seleccionamos fila a fila el ID del producto para poder saber su precio y multiplicarlo por la cantidad. El resultado lo vamos guardando en la variable @total.


Al final comprobamos si su bonificación es mayor a 1 (ha realizado anteriormente una compra superior a 100€). Si lo es, le aplicamos un descuento. Posteriormente, comprobamos el precio de su compra para aumentar su bonificación o no.

- [Trigger proveedores:](#)
 - Trigger proveedores nacionales:

```

DELIMITER $$
USE `SUPERMERCADO`$$
CREATE DEFINER = CURRENT_USER TRIGGER `SUPERMERCADO`.`proveedor_AFTER_INSERT` AFTER INSERT ON `producto_has_proveedor_nacional` FOR EACH ROW
BEGIN
  DECLARE i INT DEFAULT 0;
  select count(stock_almacen) into @numero_almacenes from supermercado_has_producto where producto_id_producto = NEW.producto_id_producto;
  set i = 0;
  while i < @numero_almacenes DO
    SELECT stock_general INTO @stock_general FROM producto WHERE id_producto = NEW.producto_id_producto;
    SELECT stock_almacen INTO @stock_almacen FROM supermercado_has_producto WHERE producto_id_producto = NEW.producto_id_producto limit 1,1;
    IF @stock_general >= 20 THEN
      IF @stock_almacen < 10 THEN
        UPDATE supermercado_has_producto SET stock_almacen = @stock_almacen + 20 where producto_id_producto = NEW.producto_id_producto;
        UPDATE producto SET stock_general = @stock_general - 20 WHERE id_producto = NEW.producto_id_producto;
      END IF;
    END IF;
    set i := i + 1;
  end while;
  SELECT cantidad INTO @cantidad_suministrada_nacional FROM producto_has_proveedor_nacional WHERE producto_id_producto = NEW.producto_id_producto;
  UPDATE producto SET stock_general = @stock_general + @cantidad_suministrada_nacional WHERE id_producto = NEW.producto_id_producto;
END$$
DELIMITER ;

```


 Escuela Superior de Ingeniería y Tecnología Universidad de La Laguna	CONSTRUCCIÓN DEL SISTEMA DE INFORMACIÓN (CSI) PROYECTO:Supermercados HIPERMAXI.	BASES DE DATOS Generación de código
	Autor: Laura Cañizares Herrera, Damián Eduardo Domínguez De Barros, Carla Fernanda Flores Gonzales, Bárbara Valentina García Deus.	alu0101129945 alu0101278353 alu0101138183 alu0101108452
Versión: 1.0	Tiempo invertido: 3h	Fecha : 08/02/21

- Trigger proveedores extranjeros:

```

DELIMITER $$
USE `SUPERMERCADO`$$
CREATE DEFINER = CURRENT_USER TRIGGER `SUPERMERCADO`.`proveedor_extranjero_AFTER_INSERT` AFTER INSERT ON
`producto_has_proveedor_extranjero` FOR EACH ROW
BEGIN
    DECLARE i INT DEFAULT 0;
    select count(stock_almacen) into @numero_almacenes from supermercado_has_producto where producto_id_producto = NEW.
    producto_id_producto;
    set i = 0;
    while i < @numero_almacenes DO
        SELECT stock_general INTO @stock_general FROM producto WHERE id_producto = NEW.producto_id_producto;
        SELECT stock_almacen INTO @stock_almacen FROM supermercado_has_producto WHERE producto_id_producto = NEW.producto_id_producto
        limit i,1;
        IF @stock_general >= 20 THEN
            IF @stock_almacen < 10 THEN
                UPDATE supermercado_has_producto SET stock_almacen = @stock_almacen + 20 where producto_id_producto = NEW.
                producto_id_producto;
                UPDATE producto SET stock_general = @stock_general - 20 WHERE id_producto = NEW.producto_id_producto;
            END IF;
        END IF;
        set i := i + 1;
    end while;
    SELECT cantidad INTO @cantidad_suministrada_extranjero FROM producto_has_proveedor_extranjero WHERE producto_id_producto = NEW.
    producto_id_producto;
    UPDATE producto SET stock_general = @stock_general + @cantidad_suministrada_extranjero WHERE id_producto = NEW.
    producto_id_producto;
END$$
DELIMITER ;

```

En estos triggers seleccionamos el stock del almacén como número de almacenes comprobando que el código de cada producto ingresado sea igual al que tenemos en almacén.

Entramos en un bucle teniendo ya el número de almacenes, seleccionamos el stock general del producto, el stock del almacén y la cantidad de la tabla de proveedor nacional o extranjero como `@cantidad_suministrada_XX` , para luego actualizar el stock general del producto sumando la cantidad actual del mismo más la cantidad suministrada por “X” proveedor.

Finalmente verificamos si el stock general del producto es mayor a 20 y el stock del almacén menor a 10, si es así se reparte 20 productos a dicho almacén y actualizamos el stock general restando los 20 productos distribuidos a ese almacén. Esta tarea la realiza siempre la cadena cuando queda poco stock en algún supermercado y por eso se ha automatizado en la base de datos. De esta forma siempre habrá productos disponibles en todos los supermercados.