**Documentación del Proyecto: Evaluación de Módulo – Módulo 5**

Este sistema tiene como propósito gestionar productos, proveedores y transacciones de compras y ventas, manteniendo actualizado el **inventario en tiempo real** mediante triggers y procedimientos almacenados.

Permite:

* Registrar **productos** y **proveedores**.
* Administrar **compras** y **ventas** a través de transacciones SQL, generadas en procedimientos almacenados.
* Controlar el **stock automáticamente** con un desencadenador, o trigger.
* Evitar ventas con stock insuficiente usando procedimientos almacenados con ROLLBACK.
* Consultar el **inventario actual** y el historial de movimientos.

**Modelo de Datos**

**Tablas principales**

1. **productos** → Catálogo de productos disponibles.
2. **cantidad\_productos** → Stock disponible por producto, relacionada con la tabla productos.
3. **proveedores** → Información de los proveedores.
4. **tipo\_transaccion** → Define los tipos de movimiento (Compra, Venta).
5. **transacciones** → Registra las operaciones de compra y venta.

**Vistas**

* **inventario** → Vista consolidada del stock de cada producto, uniendo a través de un JOIN las tablas *producto* y *cantidad\_producto*.
* **tr\_todas** → Historial de transacciones con datos descriptivos. Se unen las tablas *transacciones*, *proveedores* y *productos*.
* **tr\_venta** → Historial de transacciones realizadas como venta, con sus respectivos datos descriptivos. Se unen las tablas *transacciones*, *proveedores* y *productos*.
* **tr\_compra** → Historial de transacciones realizadas como compra, con sus respectivos datos descriptivos. Se unen las tablas *transacciones*, *proveedores* y *productos*.

**Triggers**

* **productos\_AFTER\_INSERT** → Inserta, y relaciona, automáticamente el producto, a través del id del registro ingresado, en la tabla *cantidad\_productos*, cual su stock inicial va a ser siempre 0. Desencadenador programado en la tabla *productos*.
* **transacciones\_AFTER\_INSERT** → Actualiza stock al insertar registros en la tabla *transacciones*. Desencadenador programado en la tabla *transacciones*.

**Procedimientos almacenados**

* **registrar\_compra** → Registra una compra y aumenta stock.
* **registrar\_venta** → Registra una venta con validación de stock. Si no hay suficiente, hace ROLLBACK, arrojando el error 1644, junto con un mensaje personalizado.

## **Diccionario de Datos**

A continuación, se detalla cada una de las tablas que componen la base de datos.

### **Tabla**: *productos*

**Descripción:** Almacena el catálogo principal de todos los productos disponibles.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nombre del Campo** | **Tipo de Dato** | **¿Nulable?** | **Clave (PK)** | **Valor por Defecto** | **Descripción** |
| id | INT | No | **PK** | AUTO\_INCREMENT | Identificador único autoincremental para cada producto. |
| nombre | VARCHAR(100) | No |  |  | Nombre comercial del producto. |
| descripcion | MEDIUMTEXT | No |  |  | Descripción detallada de las características del producto. |
| precio | INT UNSIGNED | No |  |  | Precio unitario de venta del producto. |
| created\_at | DATETIME | No |  | CURRENT\_TIMESTAMP | Fecha y hora de creación del registro. |
| updated\_at | DATETIME | Sí |  | ON UPDATE CURRENT\_TIMESTAMP | Fecha y hora de la última actualización del registro. |

### **Tabla**: *cantidad\_productos*

**Descripción:** Controla el stock o la cantidad disponible de cada producto. Se actualiza automáticamente mediante un trigger.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nombre del Campo** | **Tipo de Dato** | **¿Nulable?** | **Clave (PK/FK/UNIQUE)** | **Valor por Defecto** | **Descripción** |
| id | INT | No | **PK** | AUTO\_INCREMENT | Identificador único del registro de stock. |
| producto\_id | INT | No | **FK** -> productos(id) / UNIQUE |  | Referencia al producto al que pertenece este stock. |
| cantidad | INT UNSIGNED | No |  | 0 | Cantidad de unidades disponibles en inventario. |
| created\_at | DATETIME | No |  | CURRENT\_TIMESTAMP | Fecha y hora de creación del registro. |
| updated\_at | DATETIME | Sí |  | ON UPDATE CURRENT\_TIMESTAMP | Fecha y hora de la última actualización del registro. |

### **Tabla**: *proveedores*

**Descripción:** Almacena la información de contacto de las empresas o personas que suministran los productos.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nombre del Campo** | **Tipo de Dato** | **¿Nulable?** | **Clave (PK)** | **Valor por Defecto** | **Descripción** |
| id | INT | No | **PK** | AUTO\_INCREMENT | Identificador único para cada proveedor. |
| nombre | VARCHAR(100) | No |  |  | Nombre del proveedor. |
| direccion | VARCHAR(255) | No |  |  | Dirección física del proveedor. |
| telefono | VARCHAR(20) | Sí |  |  | Teléfono de contacto. |
| email | VARCHAR(255) | Sí |  |  | Correo electrónico de contacto. |
| created\_at | DATETIME | No |  | CURRENT\_TIMESTAMP | Fecha y hora de creación del registro. |
| updated\_at | DATETIME | Sí |  | ON UPDATE CURRENT\_TIMESTAMP | Fecha y hora de la última actualización del registro. |

### **Tabla**: *tipo\_transaccion*

**Descripción:** Tabla catálogo que define los tipos de movimiento de inventario (Compra o Venta).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nombre del Campo** | **Tipo de Dato** | **¿Nulable?** | **Clave (PK)** | **Descripción** |
| id\_tipo | INT | No | **PK** | Identificador único del tipo de transacción (1=Compra, 2=Venta). |
| transaccion | VARCHAR(50) | No |  | Nombre descriptivo del tipo de transacción. |

### **Tabla**: *transacciones*

**Descripción:** Registra cada operación de compra o venta, sirviendo como historial de todos los movimientos de inventario.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nombre del Campo** | **Tipo de Dato** | **¿Nulable?** | **Clave (PK/FK)** | **Valor por Defecto** | **Descripción** |
| id | INT | No | **PK** | AUTO\_INCREMENT | Identificador único de la transacción. |
| tipo | INT | No | **FK** -> tipo\_transaccion(id\_tipo) |  | Referencia al tipo de transacción (Compra o Venta). |
| fecha\_transaccion | DATETIME | No |  | CURRENT\_TIMESTAMP | Fecha y hora exactas en que se realizó la operación. |
| producto\_id | INT | No | **FK** -> productos(id) |  | Referencia al producto involucrado en la transacción. |
| cantidad | INT UNSIGNED | No |  |  | Número de unidades del producto que se movieron. |
| proveedor\_id | INT | No | **FK** -> proveedores(id) |  | Referencia al proveedor (o cliente) de la transacción. |
| created\_at | DATETIME | No |  | CURRENT\_TIMESTAMP | Fecha y hora de creación del registro. |
| updated\_at | DATETIME | Sí |  | ON UPDATE CURRENT\_TIMESTAMP | Fecha y hora de la última actualización del registro. |

### Triggers

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre** | **Tabla Asociada** | **Evento** | **Descripción** |
| productos\_AFTER\_INSERT | productos | AFTER INSERT | Cuando se inserta un nuevo producto, este trigger crea automáticamente su registro correspondiente en la tabla cantidad\_productos con un stock inicial de 0. |
| transacciones\_AFTER\_INSERT | transacciones | AFTER INSERT | Tras registrar una transacción, este trigger verifica si es una compra (suma al stock) o una venta (resta al stock) y actualiza la tabla cantidad\_productos. |

### Procedimientos Almacenados

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Parámetros** | **Descripción** |
| registrar\_compra | p\_producto\_id (INT), p\_cantidad (INT), p\_proveedor\_id (INT) | Registra una nueva compra. Inserta un registro en la tabla transacciones con tipo = 1 (Compra). La actualización del stock se delega al trigger transacciones\_AFTER\_INSERT. |
| registrar\_venta | p\_producto\_id (INT), p\_cantidad (INT), p\_proveedor\_id (INT) | Registra una nueva venta. **Primero, verifica si hay stock suficiente.** Si lo hay, inserta la transacción con tipo = 2 (Venta). Si no, cancela toda la operación (ROLLBACK) y devuelve un error. |

## Vistas (Consultas Predefinidas)

|  |  |
| --- | --- |
| **Nombre** | **Descripción** |
| inventario | Muestra una vista consolidada y fácil de leer del stock actual de cada producto, uniendo productos y cantidad\_productos. |
| tr\_todas | Proporciona un historial completo y descriptivo de todas las transacciones (compras y ventas), mostrando nombres en lugar de IDs. |
| tr\_compra | Vista filtrada de tr\_todas que muestra únicamente las transacciones de compra. |
| tr\_venta | Vista filtrada de tr\_todas que muestra únicamente las transacciones de venta. |

**Flujo de procesos**

1. **Alta de producto**
   * Se inserta en la tabla *productos*.
   * El trigger *productos\_AFTER\_INSERT* realiza la inserción automáticamente del registro, a través de su id, en *cantidad\_productos* con stock 0.
2. **Compra**
   * Procedimiento *registrar\_compra* inserta en *transacciones* con id\_tipo = 1, que es de “Compra”.
   * El trigger *transacciones\_AFTER\_INSERT* realiza la suma al stock.
3. **Venta**
   * Procedimiento *registrar\_venta* valida stock.
   * Si hay suficiente → inserta en *transacciones* y el trigger *transacciones\_AFTER\_INSERT* descuenta del stock existente.
   * Si no hay suficiente → ROLLBACK y se genera error, lanzando un mensaje avisando de la insuficiencia del stock.
4. **Consulta**
   * Vista *inventario* muestra el stock actual por producto.
   * Vista *tr\_todas* muestra historial de transacciones con nombres descriptivos.
   * Vista *tr\_compra* muestra historial de compras con sus campos descriptivos.
   * Vista *tr\_venta* muestra historial de ventas con sus campos descriptivos.

**Conclusiones:**

He decidido realizar de esta forma el análisis y el desarrollo de este proyecto, para afrontar la problemática planteada de una forma más automatizada, y que controle errores de stock. Decidí mirar la relación de las tablas *productos* y *transacciones* como 1:N, ya que no especifica que una transacción sea una boleta en sí, y porque cada transacción registra el movimiento de un producto. Esto da pie a una futura integración de una nueva tabla, que pueda contener el dato de un documento de transacción (puede ser una boleta, una factura, una guía de despacho, un ajuste de inventario, etc). Todo dependerá del enfoque que se pueda dar al proyecto o a las diferentes problemáticas que se puedan presentar en un futuro.

Junto con ello, hice la separación de la tabla *transacciones* con *tipo\_transaccion* para normalizar el dato y no caer en incongruencias.

Puede, además, ingresarse una tabla futura, llamada “Categoría”, con la normalización de los productos y su stock, y así mantener su consistencia e integridad en los datos.