Historia de Usuario

Título: Gestión de Inventario y Registro de Ventas
Autor: Área de Operaciones
Dirigido a: Equipo de Desarrollo – Unube
Fecha: [14-11-2025]

---

1. Descripción

Como administrador del sistema de ventas de Unube, quiero registrar categorías, productos, clientes y ventas, para disponer de un modelo de datos confiable que permita consultar inventarios, validar stock, registrar transacciones y generar reportes que apoyen la operación del negocio.

2. Criterios de Aceptación

#	Criterio
1	Se deben crear **cuatro tablas normalizadas** con los campos sugeridos.
2	La tabla **CATEGORIA** deberá contemplar: ID_CATEGORIA, NOMBRE, DESCRIPCION, FECHA_CREACION, FECHA_MODIFICACION.
3	La tabla **PRODUCTO** deberá contemplar: ID_PRODUCTO, NOMBRE, PRECIO, STOCK, ID_CATEGORIA, DESCRIPCION, ESTATUS, FECHA_CREACION, FECHA_MODIFICACION.
4	La tabla **CLIENTE** deberá contemplar: ID_CLIENTE, NOMBRE, APELLIDOS, CORREO, TELEFONO, FECHA_REGISTRO, FECHA_CREACION, FECHA_MODIFICACION.
5	La tabla **VENTA** deberá contemplar: ID_VENTA, ID_CLIENTE, ID_PRODUCTO, CANTIDAD, FECHA_VENTA, USUARIO_REGISTRO, FECHA_CREACION, FECHA_MODIFICACION.
6	Deben agregarse descripciones utilizando COMMENT ON TABLE y COMMENT ON COLUMN para documentar propósito de cada tabla y cada campo.
7	Se deben generar **dos vistas**: una que muestre productos con su categoría; otra que muestre ventas con datos de cliente y producto.
8	Se debe crear un **Stored Procedure (SP)** que registre una venta validando: cliente, producto, stock disponible y actualización del stock.
9	El SP debe generar el ID de venta automáticamente.
10	Se debe crear una **función** que calcule el total de la venta (cantidad × precio).
11	Se debe crear una **segunda función adicional** cuyo propósito queda a elección del desarrollador (por ejemplo: calcular IVA, validar stock, retornar estatus, etc.).

#	Criterio
12	Se deben realizar **inserciones directas** en las tablas para demostrar su funcionamiento.
13	Se deben ejecutar consultas de prueba que demuestren el funcionamiento de las vistas.
14	Se deben generar pruebas unitarias en SQL que muestren: inserciones, consultas, ejecución del SP y ejecución de ambas funciones.
15	Se deben incluir **imágenes de las pruebas unitarias**.
16	Se debe elaborar un **Diagrama Entidad–Relación (DER)** de las cuatro tablas, indicando sus relaciones y cardinalidades.
17	Todo el desarrollo debe subirse a un **repositorio personal de GitHub**, incluyendo scripts, DER, evidencias e instrucciones.
18	Se debe generar un **archivo comprimido (.zip o .rar)** con scripts, imágenes de evidencia y documentación.
19	El archivo deberá enviarse al correo: **oscar.e.sanchez.g@gmail.com**.
20	La entrega debe estar organizada, explicada y ejecutable directamente en Oracle SQL.

---

3. Observaciones

- Incluir en cada tabla campos de auditoría:
    - FECHA_CREACION
    - FECHA_MODIFICACION
- Los comentarios de tabla/columna deben explicar claramente el propósito, por ejemplo:
    - COMMENT ON TABLE PRODUCTO IS 'Tabla que almacena información de los productos disponibles en el catálogo.';
    - COMMENT ON COLUMN PRODUCTO.PRECIO IS 'Precio unitario del producto en moneda local.';
- El DER debe incluir claves primarias, foráneas y relaciones entre entidades.
- Es obligatorio entregar capturas que demuestren el funcionamiento completo del modelo.
- El repositorio de GitHub debe incluir un archivo README con instrucciones claras.

---

4. Buenas Prácticas Recomendadas

1. **Usar nombres consistentes y en un mismo idioma**, preferentemente español o inglés, pero no mezclado.
2. **Utilizar claves primarias numéricas autogeneradas** cuando sea posible para mejorar rendimiento y simplicidad.
3. **Separar scripts por tipo:**
    - 01_creacion_tablas.sql
    - 02_inserciones.sql
    - 03_vistas.sql
    - 04_funciones_sp.sql
    - 05_pruebas.sql
4. **Evitar nombres ambiguos** como “dato1”, “valor”, “nombre2”.
5. **Seguir estándares de formato SQL**, como mayúsculas para palabras reservadas y minúsculas para nombres de objetos.

---

5. Justificación

Esta historia de usuario es necesaria para:

- Construir un modelo de datos robusto y documentado que permita estandarizar las operaciones del módulo de inventarios y ventas.
- Garantizar trazabilidad y claridad gracias al uso de comentarios, DER y documentación.
- Asegurar calidad mediante pruebas unitarias, imágenes y repositorio controlado en GitHub.
- Proveer una base sólida para futuras integraciones y ampliaciones del sistema Unube.