UNIVERSIDAD POLITÉCNICA DE SAN LUIS POTOSÍ

24 DE NOVIEMBRE DE 2025

PROYECTO DE OPTIMIZACIÓN
MANUAL DE PROGRAMADOR

JUAN PABLO GONZÁLEZ NARVAEZ – 179804
RAFAEL LÓPEZ LARA – 182239

BASES DE DATOS
Profesor: Jesús Alberto Revilla Silva
Tercer Parcial

Contenido

1. Estructura de la Base de Datos

1.1 Diagrama Entidad-Relación

roles (1) ←--- (N) usuarios (N) ---→ (0) eventos

|

↓

materia_prima (1) ←--- (N) cortes_optimizados (N) ---→ (1) piezas (N) ---→ (1) productos

| |

↓ ↓

geometrias

1.2 Descripción de Tablas

Tablas de Seguridad:

roles: (id_rol, nombre_rol, descripcion)

usuarios: (id_usuario, nombre_completo, email, password_hash, id_rol, fecha_creacion, activo)

Tablas de Inventario:

materia_prima: (id_materia, numero_parte, ancho, alto, distancia_min_piezas, distancia_min_borde, area_total, stock_disponible)

productos: (id_producto, numero_parte, descripcion, fecha_alta)

piezas: (id_pieza, id_producto, nombre_pieza, cantidad_requerida, area_base)

Tablas de Geometría y Operación:

geometrias: (id_geometria, id_pieza, orden_secuencia, tipo_componente, parametros_geo)

eventos: (id_evento, tipo_evento, payload_json, procesado, fecha_evento)

cortes_optimizados: (id_corte, id_materia, id_pieza, posicion_x, posicion_y, rotacion_grados, orden_colocacion, fecha_corte)

2. Funciones y Procedimientos Almacenados

2.1 Gestión de Usuarios

fn_listar_roles()

- Propósito: Listar todos los roles disponibles
- Retorna: Tabla con (id, nombre, descripción)

sp_crear_usuario(p_nombre, p_email, p_password, p_nombre_rol)

- Propósito: Crear nuevo usuario con hash MD5
- Parámetros:
    - p_nombre: Nombre completo
    - p_email: Email único
    - p_password: Contraseña en texto plano
    - p_nombre_rol: Administrador u Operador
- Retorna: ID del usuario creado

fn_obtener_usuario(p_identificador)

- Propósito: Buscar usuario por email o ID
- Retorna: Datos del usuario con nombre del rol

sp_actualizar_usuario(p_id_usuario, p_nuevo_nombre, p_nuevo_rol)

- Propósito: Actualizar nombre y/o rol de usuario
- Parámetros opcionales: Ambos pueden ser NULL

sp_eliminar_usuario(p_id_usuario)

- Propósito: Desactivar usuario
- Retorna: boolean indicando éxito

2.2 Gestión de Inventario

sp_alta_materia_prima(...)

- Validaciones: Dimensiones > 0, número de parte único
- Campo calculado: area_total (ancho * alto)

sp_alta_producto(p_numero_parte, p_descripcion)

- Propósito: Registrar nuevo producto

sp_agregar_pieza(p_id_producto, p_nombre_pieza, p_cantidad, p_area_base)

- Validación: Verifica que el producto exista

sp_agregar_geometria(p_id_pieza, p_orden, p_tipo, p_json_params)

- Tipos soportados: 'LINEA', 'ARCO', 'CIRCULO'
- Parámetros: JSON con coordenadas específicas por tipo

2.3 Lógica de Negocio

sp_rotar_posicionar_figuras(p_payload JSONB)

- Flujo:
    - Registra evento en tabla `eventos`
    - Extrae parámetros del JSON
    - Valida existencia de pieza y materia prima
    - Inserta en `cortes_optimizados`
- Estructura JSON esperada:

```
{
  "id_pieza": 1,
  "id_materia": 1,
  "x": 10.0,
  "y": 15.5,
  "rotacion": 45.0
}
```

fn_calcular_utilizacion(p_id_materia)

- Fórmula: (SUM(area_base_piezas) / area_total_materia) * 100
- Retorna: Porcentaje con 2 decimales

3. Triggers

trg_validar_posicion

- Tabla: cortes_optimizados
- Evento: BEFORE INSERT
- Propósito: Validar que las coordenadas estén dentro de los límites de la materia prima
- Función: fn_trigger_validar_limites()

4. Seguridad y Permisos (06_seguridad.sql)

Roles de Base de Datos:

- db_admin: Privilegios completos (SELECT, INSERT, UPDATE, DELETE en todas las tablas)
- db_operador:
    - SELECT en todas las tablas
    - INSERT en `eventos` y `cortes_optimizados`
    - USAGE en sequences
    - EXECUTE en procedimientos específicos

5. Estructura de JSON en Geometrías

Para LÍNEA:

```
{
  "x1": 0, "y1": 0,
  "x2": 100, "y2": 0
}
```

Para ARCO:

```
{
```

"centro_x": 50, "centro_y": 50,

"radio": 25,

"angulo_inicio": 0,

"angulo_fin": 180

}

Para CÍRCULO:

{

"centro_x": 50, "centro_y": 50,

"radio": 25

}

6. Consideraciones de Implementación

6.1 Concurrencia

- Las funciones utilizan transacciones implícitas
- El trigger valida condiciones antes de la inserción

6.2 Rendimiento

- Índices automáticos en claves primarias
- Campo area_total como GENERATED ALWAYS para evitar cálculos repetitivos

6.3 Mantenibilidad

- Nomenclatura consistente:

- sp_ para stored procedures
- fn_ para funciones que retornan valores
- trg_ para triggers

7. Ejemplos de Uso Avanzado

Consulta de Utilización por Producto:

```
SELECT p.numero_parte, SUM(pi.area_base) as area_total_piezas
FROM productos p
JOIN piezas pi ON p.id_producto = pi.id_producto
GROUP BY p.id_producto, p.numero_parte;
```

Reporte de Eventos No Procesados:

```
SELECT id_evento, tipo_evento, fecha_evento
FROM eventos
WHERE procesado = FALSE
ORDER BY fecha_evento DESC;
```