```
USE [master]

GO

/****** Object:  Database [ParkingSystem]    Script Date: 7/13/2025 8:29:29 PM ******/

CREATE DATABASE [ParkingSystem]

 CONTAINMENT = NONE

 ON  PRIMARY

( NAME = N'ParkingSystem', FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL16.SQLEXPRESS\MSSQL\DATA\ParkingSystem.mdf' , SIZE = 8192KB , MAXSIZE = UNLIMITED, FILEGROWTH = 65536KB )

 LOG ON

( NAME = N'ParkingSystem_log', FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL16.SQLEXPRESS\MSSQL\DATA\ParkingSystem_log.ldf' , SIZE = 8192KB , MAXSIZE = 2048GB , FILEGROWTH = 65536KB )

 WITH CATALOG_COLLATION = DATABASE_DEFAULT, LEDGER = OFF

GO

ALTER DATABASE [ParkingSystem] SET COMPATIBILITY_LEVEL = 160

GO

IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))

begin

EXEC [ParkingSystem].[dbo].[sp_fulltext_database] @action = 'enable'

end

GO

ALTER DATABASE [ParkingSystem] SET ANSI_NULL_DEFAULT OFF

GO

ALTER DATABASE [ParkingSystem] SET ANSI_NULLS OFF

GO

ALTER DATABASE [ParkingSystem] SET ANSI_PADDING OFF

GO

ALTER DATABASE [ParkingSystem] SET ANSI_WARNINGS OFF

GO

ALTER DATABASE [ParkingSystem] SET ARITHABORT OFF

GO

ALTER DATABASE [ParkingSystem] SET AUTO_CLOSE ON
```

```
GO

ALTER DATABASE [ParkingSystem] SET AUTO_SHRINK OFF

GO

ALTER DATABASE [ParkingSystem] SET AUTO_UPDATE_STATISTICS ON

GO

ALTER DATABASE [ParkingSystem] SET CURSOR_CLOSE_ON_COMMIT OFF

GO

ALTER DATABASE [ParkingSystem] SET CURSOR_DEFAULT  GLOBAL

GO

ALTER DATABASE [ParkingSystem] SET CONCAT_NULL_YIELDS_NULL OFF

GO

ALTER DATABASE [ParkingSystem] SET NUMERIC_ROUNDABORT OFF

GO

ALTER DATABASE [ParkingSystem] SET QUOTED_IDENTIFIER OFF

GO

ALTER DATABASE [ParkingSystem] SET RECURSIVE_TRIGGERS OFF

GO

ALTER DATABASE [ParkingSystem] SET  ENABLE_BROKER

GO

ALTER DATABASE [ParkingSystem] SET AUTO_UPDATE_STATISTICS_ASYNC OFF

GO

ALTER DATABASE [ParkingSystem] SET DATE_CORRELATION_OPTIMIZATION OFF

GO

ALTER DATABASE [ParkingSystem] SET TRUSTWORTHY OFF

GO

ALTER DATABASE [ParkingSystem] SET ALLOW_SNAPSHOT_ISOLATION OFF

GO

ALTER DATABASE [ParkingSystem] SET PARAMETERIZATION SIMPLE

GO

ALTER DATABASE [ParkingSystem] SET READ_COMMITTED_SNAPSHOT OFF

GO
```

```
ALTER DATABASE [ParkingSystem] SET HONOR_BROKER_PRIORITY OFF

GO

ALTER DATABASE [ParkingSystem] SET RECOVERY SIMPLE

GO

ALTER DATABASE [ParkingSystem] SET  MULTI_USER

GO

ALTER DATABASE [ParkingSystem] SET PAGE_VERIFY CHECKSUM

GO

ALTER DATABASE [ParkingSystem] SET DB_CHAINING OFF

GO

ALTER DATABASE [ParkingSystem] SET FILESTREAM( NON_TRANSACTED_ACCESS = OFF )

GO

ALTER DATABASE [ParkingSystem] SET TARGET_RECOVERY_TIME = 60 SECONDS

GO

ALTER DATABASE [ParkingSystem] SET DELAYED_DURABILITY = DISABLED

GO

ALTER DATABASE [ParkingSystem] SET ACCELERATED_DATABASE_RECOVERY = OFF

GO

ALTER DATABASE [ParkingSystem] SET QUERY_STORE = ON

GO

ALTER DATABASE [ParkingSystem] SET QUERY_STORE (OPERATION_MODE = READ_WRITE, CLEANUP_POLICY =
(STALE_QUERY_THRESHOLD_DAYS = 30), DATA_FLUSH_INTERVAL_SECONDS = 900, INTERVAL_LENGTH_MINUTES = 60,
MAX_STORAGE_SIZE_MB = 1000, QUERY_CAPTURE_MODE = AUTO, SIZE_BASED_CLEANUP_MODE = AUTO,
MAX_PLANS_PER_QUERY = 200, WAIT_STATS_CAPTURE_MODE = ON)

GO

USE [ParkingSystem]

GO

/****** Object:  Table [dbo].[EspaciosFila]    Script Date: 7/13/2025 8:29:29 PM ******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO
```

```sql
CREATE TABLE [dbo].[EspaciosFila](
	[id_espacio_fila] [int] IDENTITY(1,1) NOT NULL,
	[numero_espacio] [varchar](10) NOT NULL,
	[capacidad] [int] NOT NULL,
	[estado] [varchar](20) NULL,
PRIMARY KEY CLUSTERED
(
	[id_espacio_fila] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[ListaEspera]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[ListaEspera](
	[id_espera] [int] IDENTITY(1,1) NOT NULL,
	[id_vehiculo] [int] NOT NULL,
	[fecha_solicitud] [datetime] NULL,
	[estado] [varchar](20) NULL,
PRIMARY KEY CLUSTERED
(
	[id_espera] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[MovimientosTemporales]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[MovimientosTemporales](
	[id_movimiento] [int] IDENTITY(1,1) NOT NULL,
	[id_vehiculo] [int] NULL,
	[id_espacio_fila] [int] NULL,
	[posicion_origen] [int] NULL,
	[fecha_movimiento] [datetime] NULL,
	[fecha_retorno] [datetime] NULL,
PRIMARY KEY CLUSTERED
(
	[id_movimiento] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[PilaVehiculos]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[PilaVehiculos](
	[id_pila] [int] IDENTITY(1,1) NOT NULL,
	[id_espacio_fila] [int] NULL,
	[id_vehiculo] [int] NULL,
	[posicion] [int] NULL,
	[fecha_entrada] [datetime] NULL,
PRIMARY KEY CLUSTERED
(
	[id_pila] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
```

```
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Usuarios]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Usuarios](
	[id_usuario] [int] IDENTITY(1,1) NOT NULL,
	[cedula] [varchar](20) NOT NULL,
	[nombre] [varchar](100) NOT NULL,
	[telefono] [varchar](20) NULL,
	[email] [varchar](100) NULL,
PRIMARY KEY CLUSTERED
(
	[id_usuario] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY],
UNIQUE NONCLUSTERED
(
	[cedula] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Vehiculos]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Vehiculos](
	[id_vehiculo] [int] IDENTITY(1,1) NOT NULL,
	[placa] [varchar](20) NOT NULL,
	[marca] [varchar](50) NOT NULL,
	[modelo] [varchar](50) NOT NULL,
	[id_usuario] [int] NULL,
	[hora_entrada] [datetime] NULL,
	[hora_salida] [datetime] NULL,
PRIMARY KEY CLUSTERED
(
	[id_vehiculo] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY],
UNIQUE NONCLUSTERED
(
	[placa] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[EspaciosFila] ADD  DEFAULT ((3)) FOR [capacidad]
GO
ALTER TABLE [dbo].[EspaciosFila] ADD  DEFAULT ('disponible') FOR [estado]
GO
ALTER TABLE [dbo].[ListaEspera] ADD  DEFAULT (getdate()) FOR [fecha_solicitud]
GO
ALTER TABLE [dbo].[ListaEspera] ADD  DEFAULT ('pendiente') FOR [estado]
GO
ALTER TABLE [dbo].[MovimientosTemporales] ADD  DEFAULT (getdate()) FOR [fecha_movimiento]
GO
ALTER TABLE [dbo].[PilaVehiculos] ADD  DEFAULT (getdate()) FOR [fecha_entrada]
GO
ALTER TABLE [dbo].[ListaEspera]  WITH CHECK ADD FOREIGN KEY([id_vehiculo])
REFERENCES [dbo].[Vehiculos] ([id_vehiculo])
```

```
GO
ALTER TABLE [dbo].[MovimientosTemporales]  WITH CHECK ADD FOREIGN KEY([id_espacio_fila])
REFERENCES [dbo].[EspaciosFila] ([id_espacio_fila])
GO
ALTER TABLE [dbo].[MovimientosTemporales]  WITH CHECK ADD FOREIGN KEY([id_vehiculo])
REFERENCES [dbo].[Vehiculos] ([id_vehiculo])
GO
ALTER TABLE [dbo].[PilaVehiculos]  WITH CHECK ADD FOREIGN KEY([id_espacio_fila])
REFERENCES [dbo].[EspaciosFila] ([id_espacio_fila])
GO
ALTER TABLE [dbo].[PilaVehiculos]  WITH CHECK ADD FOREIGN KEY([id_vehiculo])
REFERENCES [dbo].[Vehiculos] ([id_vehiculo])
GO
ALTER TABLE [dbo].[Vehiculos]  WITH CHECK ADD FOREIGN KEY([id_usuario])
REFERENCES [dbo].[Usuarios] ([id_usuario])
GO
/****** Object:  StoredProcedure [dbo].[sp_ActualizarEstadoEspacio]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- Actualizar estado de espacio
CREATE PROCEDURE [dbo].[sp_ActualizarEstadoEspacio]
    @id_espacio INT,
    @nuevo_estado VARCHAR(20)
AS
BEGIN
    IF @nuevo_estado IN ('libre', 'ocupado', 'reservado')
    BEGIN
        UPDATE Espacios_Parqueo
        SET estado = @nuevo_estado
        WHERE id_espacio = @id_espacio

        RETURN 1

    END

    RETURN 0

END


GO

/****** Object:  StoredProcedure [dbo].[sp_AgregarListaEspera]    Script Date: 7/13/2025 8:29:29 PM ******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

-- Agregar a lista de espera

CREATE PROCEDURE [dbo].[sp_AgregarListaEspera]

    @id_vehiculo INT

AS

BEGIN

    INSERT INTO Lista_Espera (id_vehiculo)

    VALUES (@id_vehiculo)

    RETURN SCOPE_IDENTITY()

END


GO

/****** Object:  StoredProcedure [dbo].[sp_CancelarReservacion]    Script Date: 7/13/2025 8:29:29 PM ******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp_CancelarReservacion]

    @id_reservacion INT

AS
```

```
BEGIN

   BEGIN TRANSACTION

      UPDATE Reservaciones

      SET estado = 'cancelada'

      WHERE id_reservacion = @id_reservacion



      UPDATE Espacios

      SET estado = 'libre'

      WHERE id_espacio = (SELECT id_espacio FROM Reservaciones WHERE id_reservacion = @id_reservacion)

   COMMIT

END



GO

/****** Object:  StoredProcedure [dbo].[sp_CrearFactura]    Script Date: 7/13/2025 8:29:29 PM ******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp_CrearFactura]

   @id_reservacion INT

AS

BEGIN

   DECLARE @horas INT

   DECLARE @monto_total DECIMAL(10,2)

   DECLARE @tarifa_hora DECIMAL(10,2) = 1000.00



   SELECT @horas = DATEDIFF(HOUR, fecha_inicio, fecha_fin)

   FROM Reservaciones

   WHERE id_reservacion = @id_reservacion



   SET @monto_total = @horas * @tarifa_hora
```

```
    INSERT INTO Facturas (id_reservacion, monto_total)
    VALUES (@id_reservacion, @monto_total)

    RETURN SCOPE_IDENTITY()
END

GO
/****** Object:  StoredProcedure [dbo].[sp_CrearReservacion]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROCEDURE [dbo].[sp_CrearReservacion]
    @id_vehiculo INT,
    @id_espacio INT,
    @fecha_inicio DATETIME,
    @fecha_fin DATETIME
AS
BEGIN
    IF EXISTS (SELECT 1 FROM Espacios WHERE id_espacio = @id_espacio AND estado = 'libre')
    BEGIN
        BEGIN TRANSACTION
            INSERT INTO Reservaciones (id_vehiculo, id_espacio, fecha_inicio, fecha_fin)
            VALUES (@id_vehiculo, @id_espacio, @fecha_inicio, @fecha_fin)

            UPDATE Espacios
            SET estado = 'reservado'
            WHERE id_espacio = @id_espacio
        COMMIT
        RETURN SCOPE_IDENTITY()
    END
    RETURN 0
END

GO
/****** Object:  StoredProcedure [dbo].[sp_InicializarMapaParqueo]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

CREATE PROCEDURE [dbo].[sp_InicializarMapaParqueo]
AS
BEGIN
    -- Limpiar mapa existente
    DELETE FROM MapaParqueo

    DECLARE @fila INT = 1
    DECLARE @columna INT = 1

    WHILE @fila <= 5  -- 5 filas
    BEGIN
        SET @columna = 1
        WHILE @columna <= 3  -- 3 columnas
        BEGIN
            DECLARE @id_espacio INT
            SELECT TOP 1 @id_espacio = id_espacio
            FROM Espacios
            WHERE numero = CHAR(64 + @fila) + CAST(@columna AS VARCHAR)

            IF @id_espacio IS NOT NULL
```

```sql
      BEGIN
        INSERT INTO MapaParqueo (fila, columna, id_espacio)
        VALUES (@fila, @columna, @id_espacio)
      END
      ELSE
      BEGIN
        PRINT 'No se encontró espacio para: ' + CHAR(64 + @fila) + CAST(@columna AS VARCHAR)
      END

      SET @columna = @columna + 1
    END
    SET @fila = @fila + 1
  END
END

GO
/****** Object:  StoredProcedure [dbo].[sp_LiberarEspacio]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

-- Liberar espacio y finalizar reservación
CREATE PROCEDURE [dbo].[sp_LiberarEspacio]
  @id_reservacion INT
AS
BEGIN
  BEGIN TRANSACTION
    -- Obtener id_espacio
    DECLARE @id_espacio INT
    SELECT @id_espacio = id_espacio
```

```
        FROM Reservaciones
        WHERE id_reservacion = @id_reservacion

        -- Actualizar reservación
        UPDATE Reservaciones
        SET estado = 'finalizada',
            fecha_hora_salida_real = GETDATE()
        WHERE id_reservacion = @id_reservacion

        -- Liberar espacio
        UPDATE Espacios_Parqueo
        SET estado = 'libre'
        WHERE id_espacio = @id_espacio

        -- Intentar procesar lista de espera
        EXEC sp_ProcesarListaEspera
    COMMIT
END

GO
/****** Object:  StoredProcedure [dbo].[sp_LiberarYProcesarEspera]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

CREATE PROCEDURE [dbo].[sp_LiberarYProcesarEspera]
    @id_espacio INT
AS
BEGIN
    BEGIN TRY
        BEGIN TRANSACTION
            -- 1. Liberar el espacio
            UPDATE Espacios
            SET estado = 'libre'
            WHERE id_espacio = @id_espacio

            -- 2. Buscar el primer vehículo en lista de espera
            DECLARE @id_vehiculo INT
            SELECT TOP 1 @id_vehiculo = id_vehiculo
            FROM ListaEspera
            WHERE estado = 'pendiente'
            ORDER BY fecha

            -- 3. Si hay vehículos en espera, asignar el espacio
            IF @id_vehiculo IS NOT NULL
            BEGIN
                -- Actualizar el espacio
                   UPDATE Espacios
                SET estado = 'ocupado'
                WHERE id_espacio = @id_espacio

                -- Actualizar la lista de espera
                UPDATE ListaEspera
                SET estado = 'procesado'
                WHERE id_vehiculo = @id_vehiculo AND estado = 'pendiente'

                -- Registrar la asignación
                PRINT 'Espacio asignado al vehículo en espera: ' + CAST(@id_vehiculo AS VARCHAR)
            END

        COMMIT TRANSACTION
```

```
    END TRY

    BEGIN CATCH

        IF @@TRANCOUNT > 0

            ROLLBACK TRANSACTION

        PRINT 'Error: ' + ERROR_MESSAGE()

        RAISERROR ('Error en sp_LiberarYProcesarEspera', 16, 1)

    END CATCH

END

GO

/****** Object:  StoredProcedure [dbo].[sp_ManejarSalidaVehiculoBloqueado]    Script Date: 7/13/2025 8:29:29 PM ******/

SET ANSI_NULLS ON

GO

SET QUOTED_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[sp_ManejarSalidaVehiculoBloqueado]

    @id_vehiculo INT

AS

BEGIN

    BEGIN TRY

        BEGIN TRANSACTION

            -- Verificar si el vehículo está bloqueado

            DECLARE @vehiculos_bloqueantes TABLE (id_vehiculo INT)



            INSERT INTO @vehiculos_bloqueantes

            SELECT id_vehiculo_bloqueante

            FROM VehiculosBloqueados

            WHERE id_vehiculo_bloqueado = @id_vehiculo

            AND estado = 'activo'



            -- Si está bloqueado, mover a salidas temporales los vehículos que lo bloquean

            IF EXISTS (SELECT 1 FROM @vehiculos_bloqueantes)
            BEGIN
                -- Registrar salidas temporales de vehículos bloqueantes
                INSERT INTO SalidasTemporales (id_vehiculo)
                SELECT id_vehiculo FROM @vehiculos_bloqueantes

                -- Actualizar estado de bloqueos
                UPDATE VehiculosBloqueados
                SET estado = 'procesado'
                WHERE id_vehiculo_bloqueado = @id_vehiculo
                AND estado = 'activo'

                -- Registrar salida del vehículo bloqueado
                EXEC sp_RegistrarSalidaVehiculo @id_vehiculo

                PRINT 'Vehículo y bloqueantes movidos exitosamente'
            END
            ELSE
            BEGIN
                PRINT 'El vehículo no está bloqueado'
                EXEC sp_RegistrarSalidaVehiculo @id_vehiculo
            END

        COMMIT TRANSACTION
    END TRY
    BEGIN CATCH
        IF @@TRANCOUNT > 0
            ROLLBACK TRANSACTION
        PRINT 'Error: ' + ERROR_MESSAGE()
        RAISERROR ('Error al manejar salida de vehículo bloqueado', 16, 1)
    END CATCH
END
```

```
GO
/****** Object:  StoredProcedure [dbo].[sp_ProcesarListaEspera]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- Procesar lista de espera
CREATE PROCEDURE [dbo].[sp_ProcesarListaEspera]
AS
BEGIN
    -- Obtener el primer vehículo en espera
    DECLARE @id_espera INT, @id_vehiculo INT
    SELECT TOP 1
        @id_espera = id_espera,
        @id_vehiculo = id_vehiculo
    FROM Lista_Espera
    WHERE estado = 'pendiente'
    ORDER BY fecha_solicitud

    -- Buscar espacio disponible
    DECLARE @id_espacio INT
    SELECT TOP 1 @id_espacio = id_espacio
    FROM Espacios_Parqueo
    WHERE estado = 'libre'

    IF @id_espacio IS NOT NULL
    BEGIN
        BEGIN TRANSACTION
            -- Actualizar estado en lista de espera
            UPDATE Lista_Espera
```

```
      SET estado = 'atendida'
      WHERE id_espera = @id_espera

      -- Crear reservación
      DECLARE @fecha_futura DATETIME = DATEADD(hh, 24, GETDATE())
      EXEC sp_CrearReservacion @id_vehiculo, @id_espacio, @fecha_futura
    COMMIT
    RETURN 1
  END
  RETURN 0
END

GO
/****** Object:  StoredProcedure [dbo].[sp_RegistrarBloqueoVehiculo]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROCEDURE [dbo].[sp_RegistrarBloqueoVehiculo]
  @id_vehiculo_bloqueante INT,
  @id_vehiculo_bloqueado INT,
  @id_espacio INT
AS
BEGIN
  INSERT INTO VehiculosBloqueados
    (id_vehiculo_bloqueado, id_vehiculo_bloqueante, id_espacio)
  VALUES
    (@id_vehiculo_bloqueado, @id_vehiculo_bloqueante, @id_espacio)
END

GO
```

```
/****** Object:  StoredProcedure [dbo].[sp_RegistrarEntradaVehiculo]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROCEDURE [dbo].[sp_RegistrarEntradaVehiculo]
    @id_vehiculo INT,
    @id_espacio INT
AS
BEGIN
    DECLARE @fecha_actual DATETIME = GETDATE()

    UPDATE Espacios_Parqueo
    SET estado = 'ocupado',
        id_vehiculo_actual = @id_vehiculo,
        fecha_hora_entrada = @fecha_actual
    WHERE id_espacio = @id_espacio
        AND estado = 'libre'
END

GO
/****** Object:  StoredProcedure [dbo].[sp_RegistrarPersona]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROCEDURE [dbo].[sp_RegistrarPersona]
    @cedula VARCHAR(20),
    @nombre VARCHAR(100),
    @telefono VARCHAR(20),
    @email VARCHAR(100) = NULL
```

```
AS
BEGIN
    BEGIN TRANSACTION
        INSERT INTO Personas (cedula, nombre, telefono)
        VALUES (@cedula, @nombre, @telefono)

        DECLARE @id_persona INT = SCOPE_IDENTITY()

        INSERT INTO Usuarios (id_persona, email)
        VALUES (@id_persona, @email)
    COMMIT
END

GO
/****** Object:  StoredProcedure [dbo].[sp_RegistrarSalidaVehiculo]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

-- Luego creamos el nuevo procedimiento
CREATE PROCEDURE [dbo].[sp_RegistrarSalidaVehiculo]
    @id_vehiculo INT
AS
BEGIN
    -- Registrar en SalidasTemporales
    INSERT INTO SalidasTemporales (id_vehiculo)
    VALUES (@id_vehiculo)

    -- Actualizar estado del vehículo
    UPDATE Vehiculos
    SET hora_salida = GETDATE()
    WHERE id_vehiculo = @id_vehiculo

    -- Liberar espacio
    UPDATE Espacios
    SET estado = 'libre'
    WHERE id_espacio = (SELECT TOP 1 id_espacio FROM Reservaciones WHERE id_vehiculo = @id_vehiculo AND estado = 'activa')
END

GO
/****** Object:  StoredProcedure [dbo].[sp_RegistrarUsuario]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

-- Procedimiento para registrar usuario
CREATE PROCEDURE [dbo].[sp_RegistrarUsuario]
    @cedula VARCHAR(20),
    @nombre VARCHAR(100),
    @telefono VARCHAR(20),
    @direccion VARCHAR(200),
    @email VARCHAR(100) = NULL
AS
BEGIN
    INSERT INTO Usuarios (cedula, nombre, telefono, direccion, email)
    VALUES (@cedula, @nombre, @telefono, @direccion, @email)
    RETURN SCOPE_IDENTITY()
END
GO
/****** Object:  StoredProcedure [dbo].[sp_RegistrarVehiculo]    Script Date: 7/13/2025 8:29:29 PM ******/
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROCEDURE [dbo].[sp_RegistrarVehiculo]
    @placa VARCHAR(20),
    @marca VARCHAR(50),
    @modelo VARCHAR(50),
    @id_usuario INT
AS
BEGIN
    INSERT INTO Vehiculos (placa, marca, modelo, id_usuario)
    VALUES (@placa, @marca, @modelo, @id_usuario)
    RETURN SCOPE_IDENTITY()
END
GO
/****** Object:  StoredProcedure [dbo].[sp_RetornoVehiculo]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

CREATE PROCEDURE [dbo].[sp_RetornoVehiculo]
    @id_vehiculo INT
AS
BEGIN
    -- Imprimir para debug
    PRINT 'Procesando retorno del vehículo: ' + CAST(@id_vehiculo AS VARCHAR)

    BEGIN TRY
        BEGIN TRANSACTION
    -- Actualizar SalidasTemporales
    UPDATE SalidasTemporales
    SET hora_retorno = GETDATE()
    WHERE id_vehiculo = @id_vehiculo AND hora_retorno IS NULL

    -- Actualizar Vehiculos
    UPDATE Vehiculos
    SET hora_salida = NULL
    WHERE id_vehiculo = @id_vehiculo

    -- Buscar espacio disponible
    DECLARE @id_espacio INT
    SELECT TOP 1 @id_espacio = id_espacio
    FROM Espacios
    WHERE estado = 'libre'

    IF @id_espacio IS NOT NULL
    BEGIN
        PRINT 'Espacio disponible encontrado: ' + CAST(@id_espacio AS VARCHAR)
        UPDATE Espacios
        SET estado = 'ocupado'
        WHERE id_espacio = @id_espacio
    END
    ELSE
    BEGIN
        PRINT 'No hay espacios disponibles, agregando a lista de espera'
        INSERT INTO ListaEspera (id_vehiculo, fecha, estado)
        VALUES (@id_vehiculo, GETDATE(), 'pendiente')
    END

COMMIT TRANSACTION
```

```
        PRINT 'Retorno procesado exitosamente'
    END TRY
    BEGIN CATCH
        IF @@TRANCOUNT > 0
            ROLLBACK TRANSACTION
        PRINT 'Error: ' + ERROR_MESSAGE()
        RAISERROR ('Error en sp_RetornoVehiculo', 16, 1)
    END CATCH
END

GO
/****** Object:  StoredProcedure [dbo].[sp_VerificarDisponibilidad]    Script Date: 7/13/2025 8:29:29 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROCEDURE [dbo].[sp_VerificarDisponibilidad]
    @fecha_inicio DATETIME,
    @fecha_fin DATETIME
AS
BEGIN
    SELECT e.id_espacio, e.numero, e.estado, e.tipo
    FROM Espacios e
    WHERE e.id_espacio NOT IN (
        SELECT r.id_espacio
        FROM Reservaciones r
        WHERE r.estado = 'activa'
        AND (
            (@fecha_inicio BETWEEN r.fecha_inicio AND r.fecha_fin)
            OR (@fecha_fin BETWEEN r.fecha_inicio AND r.fecha_fin)
            OR (r.fecha_inicio BETWEEN @fecha_inicio AND @fecha_fin)
        )

    )

    AND e.estado = 'libre'

END


GO

USE [master]

GO

ALTER DATABASE [ParkingSystem] SET  READ_WRITE

GO
```