Segundo Parcial práctico

Enunciado general: La empresa TechNova desarrolla proyectos tecnológicos para diferentes clientes. Necesita un sistema en MySQL que permita administrar la información de sus departamentos, empleados, proyectos y asignaciones de trabajo.
Cada estudiante deberá implementar las soluciones SQL correspondientes a su reto asignado (1 al 16). Todos trabajarán con las mismas tablas base, pero aplicarán diferentes reglas, funciones, procedimientos o triggers según su número de reto. Cada estudiante deberá crear una base de datos llamada BD_TechNova y allí resolver todo el parcial

- **Estructura base común (para todos)**

Cada estudiante deberá crear estas tablas (DDL):

```
CREATE TABLE Departamento (
 id_departamento INT AUTO_INCREMENT PRIMARY KEY,
 nombre VARCHAR(100) NOT NULL,
 presupuesto DECIMAL(12,2) CHECK (presupuesto > 0)
);

CREATE TABLE Empleado (
 id_empleado INT AUTO_INCREMENT PRIMARY KEY,
 nombre VARCHAR(100),
 cargo VARCHAR(50),
 salario DECIMAL(10,2) CHECK (salario > 0),
 id_departamento INT,
 fecha_ingreso DATE,
 FOREIGN KEY (id_departamento) REFERENCES Departamento(id_departamento)
);

CREATE TABLE Proyecto (
 id_proyecto INT AUTO_INCREMENT PRIMARY KEY,
 nombre VARCHAR(100),
 fecha_inicio DATE,
 presupuesto DECIMAL(12,2),
 id_departamento INT,
 FOREIGN KEY (id_departamento) REFERENCES Departamento(id_departamento)
);

CREATE TABLE Asignacion (
 id_asignacion INT AUTO_INCREMENT PRIMARY KEY,
 id_empleado INT,
```

```
 id_proyecto INT,
 horas_trabajadas INT CHECK (horas_trabajadas >= 0),
 FOREIGN KEY (id_empleado) REFERENCES Empleado(id_empleado),
 FOREIGN KEY (id_proyecto) REFERENCES Proyecto(id_proyecto)
);
```

Luego insertan algunos datos de ejemplo para trabajar (mínimo 3 departamentos, 5 empleados, 3 proyectos y 5 asignaciones).

RETOS:

Cada estudiante debe resolver el reto que le corresponda según la asignación.

Reto 1 – Incremento de salarios

Cuando el presupuesto del departamento supera 10.000.000, aumentar en 10% los salarios del departamento.

- Procedimiento: AumentarSalario(dep_id, porcentaje)
- Función: TotalSalarios(dep_id)
- Trigger: registrar en HistorialCambios los cambios de salario.
- Transacción: si el presupuesto queda < 0 tras el ajuste → ROLLBACK.

Reto 2 – Reducción de presupuesto

Si un proyecto tiene más de 3 empleados asignados, reducir su presupuesto en 5%.

- Procedimiento: ReducirPresupuestoProyecto(id_proyecto)
- Función: TotalEmpleadosProyecto(id_proyecto)
- Trigger: registrar cambios de presupuesto en HistorialPresupuesto.
- Transacción: si presupuesto < 1.000.000 → ROLLBACK.

Reto 3 – Bonificación por horas extras

Si un empleado trabaja más de 120 horas totales, recibe una bonificación del 8% sobre su salario.

- Función: HorasEmpleado(id_empleado)
- Procedimiento: AplicarBonificacion()
- Trigger: registrar en HistorialBonificaciones los aumentos.
- Transacción: revierte si la bonificación supera 500.000.

Reto 4 – Control de nuevos proyectos

Ningún departamento puede tener más de 3 proyectos activos.

• Trigger: BEFORE INSERT en Proyecto que impida registrar el 4.º proyecto.
• Función: ProyectosDepartamento(id_dep)
• Procedimiento: listar departamentos con proyectos límite.
• Transacción: si se inserta un 4.º proyecto → ROLLBACK.

Reto 5 – Reasignación de empleados

Mueve todos los empleados de un departamento a otro, siempre que el destino tenga más presupuesto.

• Procedimiento: ReasignarEmpleados(dep_origen, dep_destino)
• Función: PresupuestoDepartamento(id)
• Trigger: registrar cada cambio de departamento.
• Transacción: si el destino no tiene más presupuesto → ROLLBACK.

Reto 6 – Control de salarios mínimos

Impedir que se inserte o actualice un empleado con salario inferior a 1.200.000.

• Trigger: BEFORE INSERT y BEFORE UPDATE.
• Procedimiento: ActualizarSalariosMinimos()
• Función: EmpleadosPorDebajoSalarioMin()
• Transacción: revertir inserciones inválidas.

Reto 7 – Eliminación controlada de empleados

No permitir eliminar empleados con horas registradas en proyectos.

• Trigger: BEFORE DELETE en Empleado.
• Función: TotalHorasEmpleado(id)
• Procedimiento: IntentarEliminarEmpleado(id)
• Transacción: si tiene horas asignadas → ROLLBACK.

Reto 8 – Revisión de productividad

Mostrar los empleados cuyo total de horas sea superior al promedio general.

• Subconsulta: comparar horas por empleado vs promedio general.
• Función: PromedioHorasGlobal()
• Procedimiento: ListarEmpleadosDestacados()
• Trigger: registrar cambios de productividad.
• Transacción: revertir si horas < 0.

Reto 9 – Control de costos de proyecto

Si el total de salarios de los empleados de un proyecto supera su presupuesto, reducir presupuesto en 10%.

• Función: TotalSalariosProyecto(id_proyecto)
• Procedimiento: AjustarPresupuestoProyecto()
• Trigger: registrar cambios en HistorialProyectos.
• Transacción: si presupuesto < 0 → ROLLBACK.

Reto 10 – Evaluación de desempeño

Aumentar salario de empleados que trabajen en más de 2 proyectos.

• Función: ProyectosPorEmpleado(id_empleado)
• Procedimiento: IncrementarPorDesempeño()
• Trigger: registrar incrementos en HistorialAumentos.
• Transacción: revierte si aumento total > 2.000.000.

Reto 11 – Control de fechas

Ningún proyecto puede tener fecha de inicio anterior al 2020.

• Trigger: BEFORE INSERT en Proyecto.
• Procedimiento: ActualizarFechasInvalidas()
• Función: ProyectosInvalidos()
• Transacción: revertir inserciones con fechas no válidas.

Reto 12 – Departamentos con bajo rendimiento

Identificar departamentos cuyo promedio de horas trabajadas < 50.

• Subconsulta: promedio de horas por departamento.
• Función: PromedioHorasDep(id_dep)
• Procedimiento: ListarDepartamentosBajos()
• Trigger: registrar cuando un departamento baja de ese umbral.
• Transacción: revertir reducciones mal calculadas.

Reto 13 – Auditoría de altas

Registrar automáticamente en una tabla de auditoría cada vez que se inserta un empleado nuevo.

• Trigger: AFTER INSERT en Empleado.
• Función: TotalEmpleadosDepartamento(id_dep)

• Procedimiento: MostrarAltasRecientes()
• Transacción: revertir si la auditoría falla.

Reto 14 – Control de proyectos sin asignaciones

Eliminar proyectos sin empleados asignados y registrar en auditoría.

• Subconsulta: detectar proyectos sin asignaciones.
• Procedimiento: EliminarProyectosInactivos()
• Trigger: AFTER DELETE registrar acción.
• Transacción: si se borra más de 2 → ROLLBACK.

Reto 15 – Ajuste de presupuesto anual

Aumentar presupuesto de todos los departamentos en 7% si la suma total de horas trabajadas supera 1000.

• Función: TotalHorasEmpresa()
• Procedimiento: ActualizarPresupuestos()
• Trigger: registrar cambios presupuestales.
• Transacción: revertir si total de horas < 1000.

Reto 16 – Control de asignaciones duplicadas

Impedir que un mismo empleado esté asignado dos veces al mismo proyecto.

• Trigger: BEFORE INSERT en Asignacion.
• Función: VerificarAsignacion(id_empleado, id_proyecto)
• Procedimiento: CorregirDuplicadas()
• Transacción: revertir inserciones duplicadas.

Entregables:

Crear un paquete en el repositorio de trabajos en clase llamado Segundo Parcial Práctico, y en el cargar:

- Pantallazos de los resultados de ejecución
- Script con las sentencias.

Entregar en un PDF el enlace del repositorio donde esta cargado el Parcial