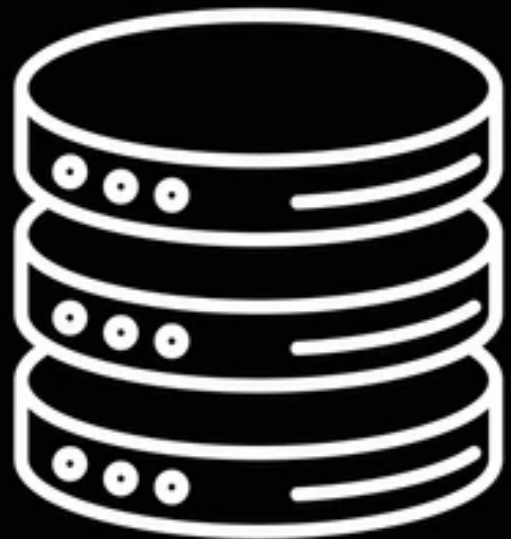



Proyecto

Bases de Datos

Cristina Afonso Otero





Índice


 Introducción.....	2
 Análisis del Enunciado.....	3
 Modelo Conceptual.....	4
 Modelo Relacional.....	5
 Implementación en MySQL.....	6
Creación de la base de datos y tablas.....	7
Inserción de datos.....	7
 Consultas Propuestas.....	8
 Propuesta de Ampliación de Datos.....	8
Modelo Relacional Ampliado.....	10
 Vistas y Triggers.....	10
Vistas:.....	11
Triggers:.....	11
 Conclusión.....	13

Introducción

Este proyecto corresponde al trabajo final de la asignatura de Bases de Datos del primer curso del ciclo formativo de Desarrollo de Aplicaciones Multiplataforma (DAM). Su objetivo principal es el diseño e implementación de una base de datos relacional utilizando MySQL. Para ello, he optado por desarrollar el Caso Práctico 1: **Sistema de Gestión de Empleados para TechSolutions Inc.**, cuyo enunciado es el siguiente:

 **Caso Práctico 1: Sistema de Gestión de Empleados para TechSolutions Inc.**

 **Descripción del Caso Estimado** Equipo de Analistas, En TechSolutions Inc., estamos comprometidos con la eficiencia y la organización en la gestión de nuestros recursos humanos. 🙌 Como parte de nuestros esfuerzos continuos por mejorar, buscamos implementar un sistema de gestión de empleados que nos permita administrar de manera más efectiva la información de nuestros empleados, departamentos y puestos de trabajo. Actualmente, enfrentamos desafíos como: Falta de integración entre la información de empleados, departamentos y puestos. Dificultad para acceder a datos históricos y actualizados sobre la estructura organizacional. Gestión manual de nóminas, evaluaciones y asignaciones de roles, lo que genera errores y retrasos. Este sistema nos permitirá centralizar y automatizar la información relacionada con empleados, departamentos y puestos de trabajo, mejorando así nuestra eficiencia operativa y la toma de decisiones.

 **Objetivo** Diseñar e implementar un sistema de gestión de empleados que administre eficientemente la información sobre: Empleados: Datos personales, roles y evaluaciones. Departamentos: Estructura organizacional y asignación de empleados. Puestos de trabajo: Descripción de roles, salarios y responsabilidades.

 **Instrucciones para los Analistas**

1. Identificación de Entidades y Atributos

Identifiquen las entidades clave que deben formar parte del sistema. Estas entidades deben incluir, como mínimo: Empleados: Información personal, roles y evaluaciones. Departamentos: Estructura organizacional y asignación de empleados. Puestos de trabajo: Descripción de roles, salarios y responsabilidades. Asegúrense de definir los atributos relevantes para cada entidad.

2. 🔗 Relaciones entre Entidades

Analicen y definan las relaciones entre las entidades: Empleado → Departamento: Un empleado pertenece a un departamento (relación muchos a uno). Empleado → Puesto: Un empleado tiene un puesto de trabajo (relación muchos a uno). Departamento → Puesto: Un departamento puede tener varios puestos de trabajo (relación uno a muchos).

3. 📊 Diseño del Modelo Conceptual

Utilicen herramientas como Draw.io, Lucidchart o MySQL Workbench para crear un diagrama Entidad-Relación (ER) que represente las entidades, atributos y relaciones identificadas.

4. 🗄️ Implementación en MySQL

Una vez definido el modelo conceptual, implementen el modelo relacional en MySQL. Incluyan: Scripts de creación de tablas. Inserción de datos de prueba. Consultas SQL para obtener información relevante (por ejemplo, listado de empleados por departamento, puestos vacantes, etc.).

5. ✨ Ampliación del Sistema

Propongan y desarrollen al menos dos funcionalidades adicionales que mejoren el sistema. Algunas ideas: Implementación de vistas para simplificar consultas frecuentes. Creación de triggers para actualizar automáticamente el salario promedio por departamento. Generación de informes de evaluación de desempeño por empleado.

6. 📝 Documentación

Deberán Elaborar un informe técnico, tienen las instrucciones en el readme de la carpeta.

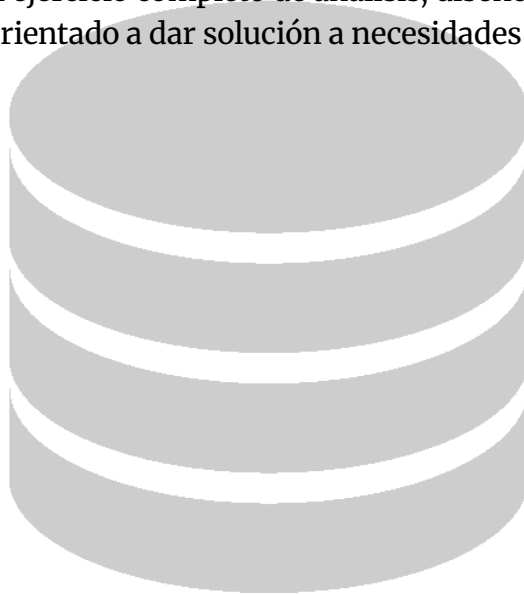
🔍 Análisis del Enunciado

El caso práctico trata de desarrollar un Sistema de Gestión de Empleados para *TechSolutions Inc.*, que ayude a automatizar y centralizar la información de los empleados, departamentos y puestos de trabajo. De ahí que la empresa identifique una serie de problemas importantes como la falta de una integración de datos, poco acceso a la información de una manera actualizada, así como aquellos procesos manuales que son propensos a dar errores.

El proyecto plantea realizar un trabajo estructurado que engloba:

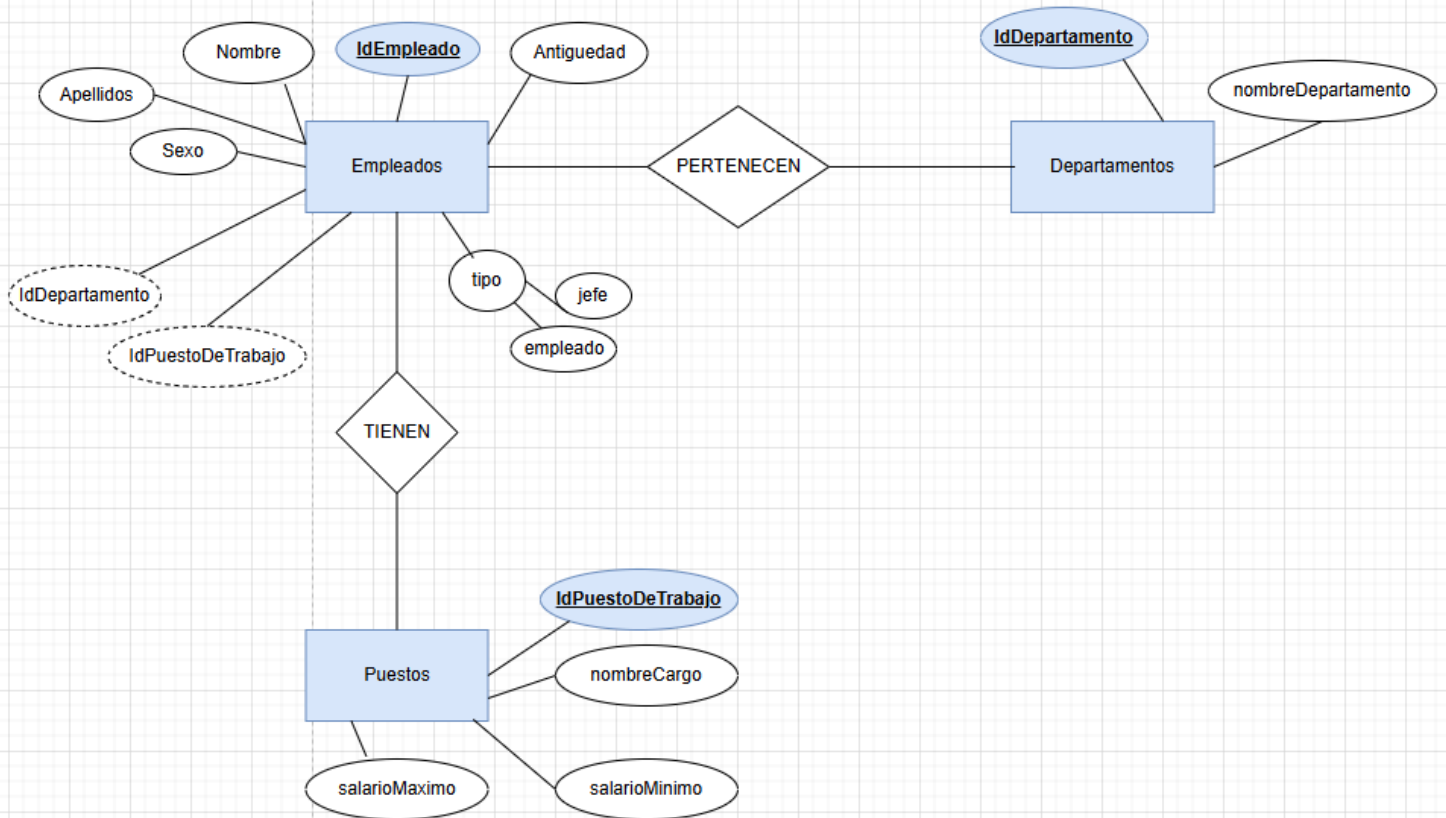
- Identificación de entidades y las correspondientes relaciones (empleados, departamentos y puestos).
- El diseño conceptual a partir de un diagrama ER.
- La implementación técnica a partir de la herramienta MySQL (tablas, datos, consultas).
- Las ampliaciones funcionales opcionales como valor añadido.
- La documentación técnica final.

En resumidas cuentas, un ejercicio completo de análisis, diseño e implementación de una base de datos relacional orientado a dar solución a necesidades reales de gestión de empresa.

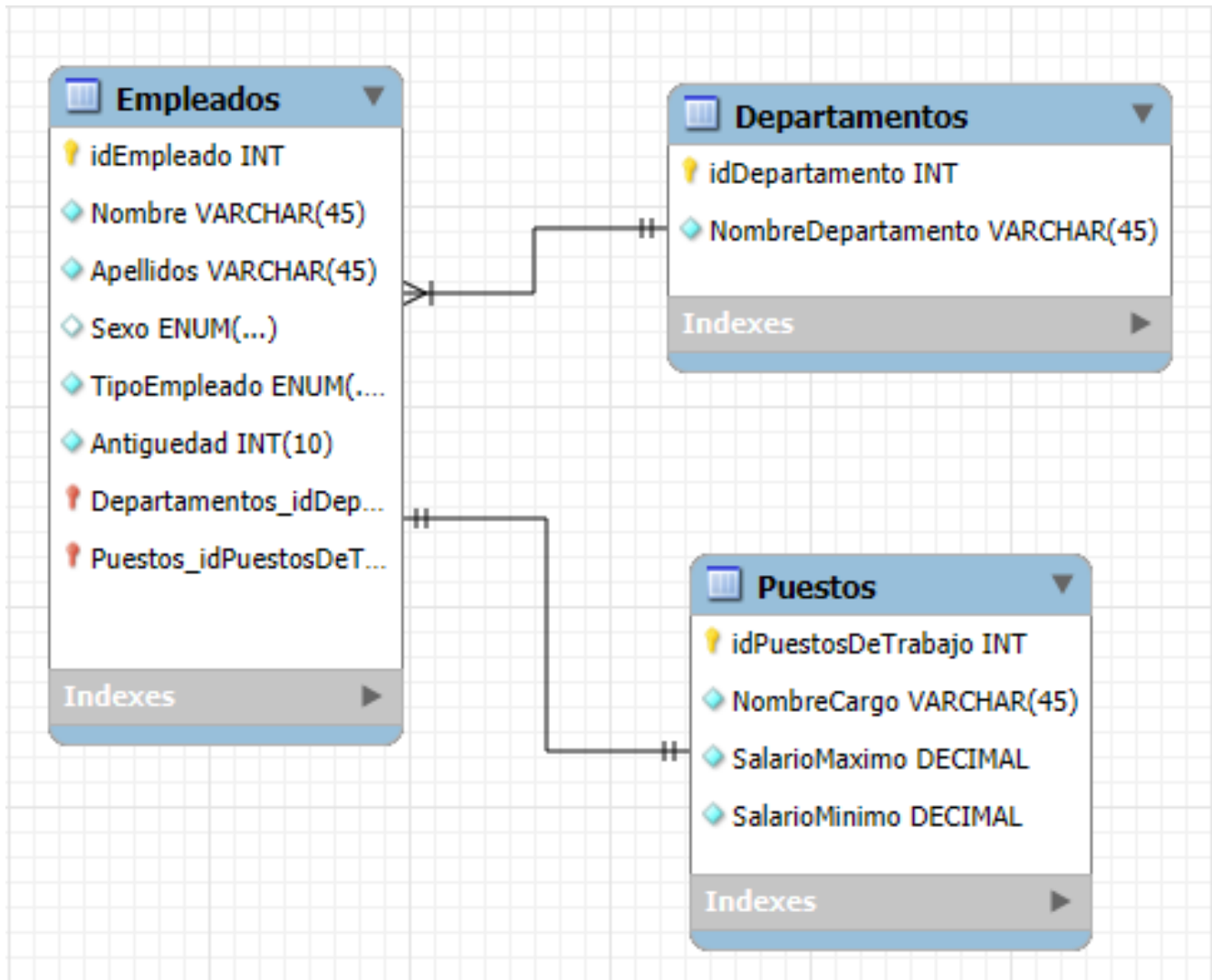


Modelo Conceptual

Sistema de Gestión de Empleados para Tech Solutions Inc.



Modelo Relacional



Implementación en MySQL

Para llevar a cabo la implementación del modelo relacional, utilicé MySQL como sistema gestor de base de datos. Creé la base de datos con el nombre **techsolutions** y dentro de ella he definido las tablas necesarias para representar los datos de la empresa.

Creación de la base de datos y tablas

Comencé creando la base de datos y configurando algunos parámetros del servidor (como el conjunto de caracteres y zona horaria) para evitar errores de compatibilidad. Después de eso, creé tres tablas principales:

- **departamentos:** almacena los diferentes departamentos de la empresa, como Recursos Humanos, Informática, etc.
- **puestos:** contiene los distintos cargos disponibles, junto con su salario mínimo y máximo.
- **empleados:** guarda los datos de cada empleado, incluyendo nombre, apellidos, sexo, tipo (jefe o empleado), antigüedad, y su relación con un departamento y un puesto.

Todas las tablas están relacionadas correctamente usando llaves foráneas para mantener la integridad de los datos. También se incluí llaves primarias.

Inserción de datos

Una vez creadas las tablas, inserté registros de prueba:

- 5 departamentos
- 14 puestos de trabajo
- 16 empleados

Los datos insertados simulan una empresa realista con estructura jerárquica. Por ejemplo, hay un CEO, varios jefes de departamento y empleados con distintos cargos. También se cuidó que cada empleado esté vinculado correctamente a su departamento y puesto correspondiente. Esta información servirá más adelante para hacer consultas, vistas o pruebas funcionales.



Consultas Propuestas

A continuación se detallan las consultas SQL que he considerado más útiles para la explotación de la base de datos, orientadas a facilitar la gestión y el análisis de la información del sistema:

1. **Listado completo de los empleados con su puesto de trabajo y departamento correspondiente.** Permite obtener una visión global de la plantilla y de su distribución organizativa.
2. **Número de empleados por departamento.** Permite el análisis de la carga de personal existente por área de la empresa.
3. **Promedio de los salarios por puesto de trabajo.** Facilita el estudio de las compensaciones medias en función del rol que desempeñan.
4. **Identificación de los empleados que están asignados a roles de jefatura.** Permite identificar fácilmente al personal que tiene funciones de liderazgo.
5. **Empleados con una antigüedad mayor a cinco años.** Permite identificar al personal más antiguo.
6. **Cálculo estimado de la suma de los salarios por departamento (promedio por empleado).** Permite obtener una estimación del coste salarial que tiene cada área de la empresa a partir del promedio del coste por trabajador.



Propuesta de Ampliación de Datos

Propongo ampliar la base de datos incorporando nueva información relevante para la gestión de recursos humanos. En concreto, se plantea:

- **Añadir nuevas columnas a la tabla de empleados**, con el objetivo de enriquecer el registro individual de cada trabajador. Estas columnas serían:
 - **SueldoActual**: para reflejar el salario vigente del empleado.
 - **NumeroIdentificacion**: documento identificativo único (como DNI o NIE).
 - **FechaNacimiento**: dato clave para cálculos de edad, beneficios y planificación de recursos.
- **Crear una nueva tabla de asistencia de empleados**, destinada a registrar la presencia diaria del personal. Esta tabla incluiría:
 - **idEmpleado**: clave foránea vinculada a la tabla de empleados.
 - **Fecha**: fecha de la asistencia.
 - **HoraEntrada**: hora en la que el empleado comienza su jornada.
 - **HoraSalida**: hora en la que finaliza su jornada.

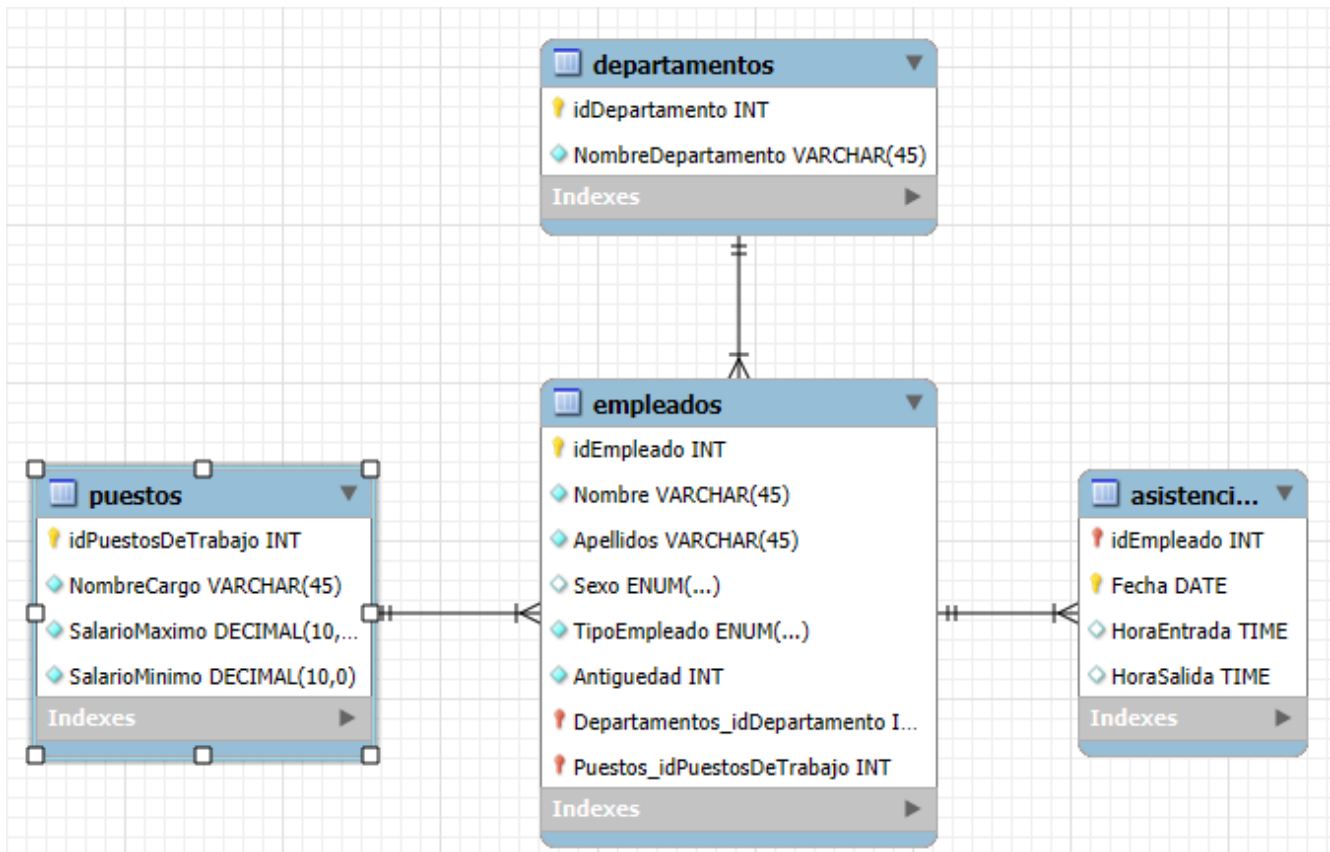
Para asegurar la integridad y unicidad de los registros, he establecido una **clave primaria compuesta** utilizando los campos **idEmpleado** y **Fecha**. Esta decisión se justifica porque:

- Un mismo empleado puede tener múltiples registros de asistencia, pero **solo uno por día**.
- La combinación de **idEmpleado** y **Fecha** garantiza que no se dupliquen entradas para el mismo trabajador en una misma fecha.

- Ni **idEmpleado** ni **Fecha** por sí solos serían suficientes para identificar un registro de forma única.

Esta ampliación permitirá un control más detallado de la información del personal y facilitará la generación de informes sobre jornada laboral, puntualidad y presencia en la empresa, contribuyendo así a una gestión más eficiente de los recursos humanos.

Modelo Relacional Ampliado



Vistas y Triggers

Para facilitar la gestión y obtener información de manera más eficiente, he implementado varias vistas y triggers en la base de datos, con el objetivo de automatizar procesos y mejorar la accesibilidad a los datos más importantes.

Vistas:

Vista de Empleados y Jefes de Departamento

Esta vista muestra a los empleados junto con el nombre de su jefe de departamento. Así, se puede ver fácilmente qué empleados están a cargo de qué jefes, lo que resulta útil para tener claro quién supervisa a quién dentro de la organización.

Vista de Empleados con Antigüedad Mayor a 5 Años

Muestra los empleados que tienen más de cinco años trabajando en la empresa, calculando la antigüedad a partir de la fecha de contratación. Esto es útil para ver qué empleados tienen más tiempo en la compañía y para tomar decisiones sobre posibles promociones o beneficios.

Vista de Asistencia Diaria por Empleado

Con esta vista se puede ver el registro de asistencia de cada empleado, incluyendo las horas de entrada y salida, ordenadas por fecha. Esto ayuda a tener un control más detallado sobre la presencia de los empleados en la empresa.

Vista de Promedio de Salario por Departamento

Calcula el salario promedio por departamento, proporcionando una visión rápida sobre cómo se distribuyen los salarios en cada área de la empresa. Esta vista es útil para hacer análisis comparativos entre los diferentes departamentos.

Triggers:

Trigger para Asignar Salario Mínimo si no se Especifica Sueldo

Este trigger asegura que, si no se especifica un sueldo para un nuevo empleado, se le asignará automáticamente el salario mínimo de su puesto de trabajo.

Objetivo: Garantizar que todos los empleados tengan un sueldo asignado, incluso si no se proporciona explícitamente.

Trigger para Convertir Nombres y Apellidos a Mayúsculas

Se encarga de convertir los nombres y apellidos de los empleados a mayúsculas antes de ser insertados en la base de datos, para mantener un formato consistente y estandarizado.

Objetivo: Mejorar la calidad de los datos y asegurar uniformidad en la base de datos.

Trigger para Insertar Automáticamente la Hora de Entrada

Si no se especifica la hora de entrada al registrar la asistencia de un empleado, este trigger automáticamente inserta la hora actual. Esto hace que el proceso de registrar las asistencias sea más rápido y sin errores.

Objetivo: Evitar la omisión de la hora de entrada, registrándola automáticamente.

Trigger para Validar la Fecha de Asistencia

Este trigger evita que se registren asistencias con una fecha futura. Si se intenta insertar una fecha que no sea del día actual o anterior, el sistema generará un error y no permitirá el registro.

Objetivo: Prevenir la introducción de datos incorrectos en el sistema.

Con estas vistas y triggers, he optimizado la gestión de datos y mejorado la eficiencia del sistema. Las vistas proporcionan acceso rápido a información relevante, como la asistencia, la antigüedad de los empleados y los sueldos promedio por departamento. Por otro lado, los triggers automatizan tareas como la asignación de sueldos, la estandarización de nombres y el registro de horas, asegurando que la información siempre esté actualizada y consistente.

Conclusión

Este proyecto ha sido una gran oportunidad para aplicar los conocimientos adquiridos en la asignatura de Bases de Datos. A lo largo del desarrollo, he podido trabajar en todas las fases del diseño e implementación de un sistema realista, orientado a mejorar la gestión de empleados en una empresa.

Gracias al uso de MySQL, he podido crear una base de datos bien estructurada, con relaciones claras entre empleados, departamentos y puestos de trabajo. Además, las vistas y triggers añadidos han permitido automatizar tareas y facilitar el acceso a la información, mejorando así la eficiencia del sistema.

También he propuesto ampliaciones útiles, como el control de asistencia y nuevos campos en la tabla de empleados, que aportan más valor al sistema y lo acercan a una solución más completa para el área de recursos humanos.

En resumen, ha sido un proyecto muy enriquecedor, que me ha permitido desarrollar habilidades técnicas y entender mejor cómo una base de datos puede dar respuesta a las necesidades reales de una empresa.