Universitat Oberta de Catalunya

uoc.edu

Diseño de Implementación de la Base de Datos para una Aplicación de Control de Cambios

María del Carmen Fernández Moreno
Grado de Ingeniería Informática
Base de Datos

Jordi Ferrer Duran
Josep Cobarsí Morales

Enero 2024

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Diseño de Implementación de la Base de Datos para una Aplicación de Control de Cambios*
Nombre del autor:	*Carmen Fernández Moreno*
Nombre del consultor/a:	*Jordi Ferrer Duran*
Nombre del PRA:	*Josep Cobarsí Morales*
Fecha de entrega (mm/aaaa):	01/2024
Titulación:	*Grado de Ingeniería Informática*
Área del Trabajo Final:	*Bases de Datos*
Idioma del trabajo:	*Castellano*
Palabras clave	*Control de Cambios, Base de Datos, Relacional*

Resumen del Trabajo:

Este proyecto consiste en el diseño e implementación de una base de datos (BD) para una aplicación especializada en la gestión de cambios en aplicaciones informáticas, siguiendo las prácticas de ITIL. Este sistema responde a las necesidades específicas de una empresa de desarrollo de *software*, buscando superar las limitaciones de las herramientas generalistas existentes en el mercado. El enfoque principal es definir y controlar los cambios, considerados como cualquier variación en las funcionalidades, *software* o *hardware* de la aplicación. Un aspecto clave es la creación de un inventario detallado de todas las aplicaciones en producción, incluyendo información vital como contactos, fechas de producción, usuarios, tecnología utilizada y nivel de criticidad.

El proceso de aprobación de cambios involucra en el proyecto a varios actores clave, como el responsable técnico de IT, el responsable del área de negocio y el gestor de cambios de la empresa, apoyados por el Global Change Advisory Board (GCAB). La BD maneja esta estructura de aprobación, registrando cada paso, categoría de cambio y comentarios pertinentes. El seguimiento post-implantación y la auditoría de los cambios son esenciales para garantizar la eficacia y mejora continua. Se presta especial atención a la escalabilidad de la BD, su capacidad para integrarse con otros sistemas y la implementación de mecanismos de prueba y mantenimiento.

En términos generales, este proyecto se enfoca en el diseño de una solución de BD eficiente y especializada para el control de cambios en aplicaciones informáticas, garantizando trazabilidad, integridad y adaptabilidad a las necesidades futuras de la empresa.

Abstract (in English, 250 words or less):

This project consists of the design and implementation of a database (DB) for an application specialized in change management in software applications, following ITIL practices. This system responds to the specific needs of a software development company, seeking to overcome the limitations of the generalist tools existing in the market. The main focus is to define and control changes, considered as any variation in the functionalities, software or hardware of the application. A key aspect is the creation of a detailed inventory of all applications in production, including vital information such as contacts, production dates, users, technology used and criticality level.

The change approval process involves several key players in the project, such as the IT technical manager, the business area manager and the company's change manager, supported by the Global Change Advisory Board (GCAB). The GCAB manages this approval structure, recording each step, change category and relevant comments. Post-implementation monitoring and auditing of changes are essential to ensure effectiveness and continuous improvement. Special attention is given to the scalability of the DB, its ability to integrate with other systems, and the implementation of testing and maintenance mechanisms.

In general terms, this project focuses on the design of an efficient and specialized DB solution for change control in computer applications, guaranteeing traceability, integrity and adaptability to the future needs of the company.

Índice

Lista de figuras

Lista de tablas

1. Introducción

Este proyecto surge de la necesidad de una empresa de desarrollo de software de disponer de una herramienta propia para la gestión de cambios en aplicaciones informáticas, siguiendo las recomendaciones de ITIL (1). La empresa identifica una brecha en las soluciones existentes en el mercado, que considera demasiado generales para sus requerimientos específicos. En respuesta, el proyecto propone el diseño y la implementación de una base de datos personalizada que soporte una aplicación dedicada a la gestión de cambios. Este trabajo se centrará en definir la estructura y las capacidades de esta base de datos, asegurando que cumpla con los requisitos de la empresa y facilite la gestión y análisis eficientes de los datos.

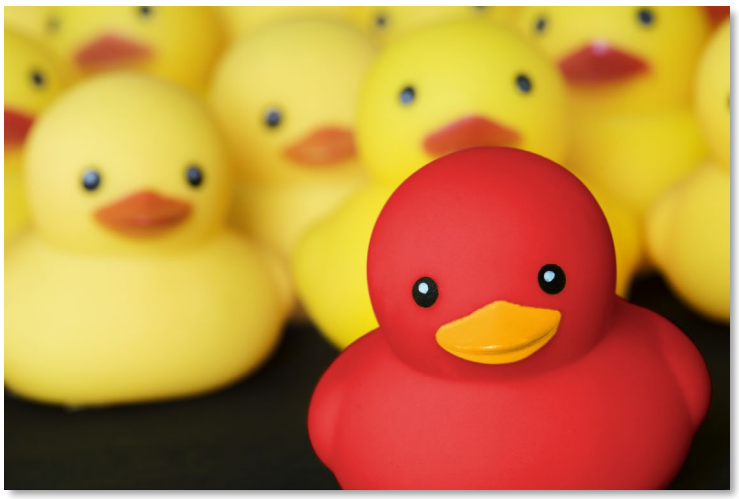


Figura 1: Representación de la gestión de cambios

La contribución principal de este proyecto es el diseño y la implementación de una base de datos robusta y a medida para una aplicación de gestión de cambios en TI. La base de datos incluirá un inventario detallado de las aplicaciones con información clave y seguirá un proceso de aprobación de cambios que involucra a varios roles organizacionales. Se establecerán categorías de cambios y planes de acción para su gestión, considerando el alcance geográfico y la posibilidad de auditorías del proceso. Además, se definirán consultas específicas para evaluar aspectos críticos del proceso de cambio. El diseño será escalable y adaptable, con capacidad para incorporar nuevas funcionalidades y consultas, asegurando así su relevancia y utilidad a largo plazo.

1.1 Contexto y justificación del Trabajo

A continuación se analiza el punto de partida del trabajo abordando cuestiones clave como la necesidad que se busca satisfacer, la relevancia del tema y el estado actual de soluciones para el problema en cuestión. También se describe la aportación realizada que describe el resultado se quiere obtener.

1.1.1 Punto de partida

En este proyecto se plantea el diseño e implementación de una base de datos para dar respuesta a una necesidad planteada por una empresa de desarrollo de *software* (la empresa) que desea desarrollar una aplicación propia para gestionar uno de los procesos ITIL (*Information Technology Infrastructure Library*) más relevantes, la gestión de cambios en aplicaciones informáticas (1).

El objetivo principal de la gestión de cambios de ITIL es garantizar que los cambios se realicen de manera controlada, minimizando el riesgo de interrupción de los servicios de TI. El proceso implica crear solicitudes (RFC), evaluar, planificar, aprobar, implementar, revisar y cerrar cambios. El flujo del proceso de gestión de ITIL consta de cuatro etapas: la primera etapa es la etapa RFC, donde se plantea una solicitud de cambio; la segunda etapa es la de evaluación y planificación del cambio; la tercera etapa es la de aprobación del cambio, donde este se aprueba o rechaza; y la etapa final es la de implementación del cambio, donde se implementa y revisa. La gestión de cambios ITIL proporciona varios beneficios, incluido un menor riesgo de interrupción de los servicios de TI, una mayor eficiencia, una mejor comunicación y colaboración y una mayor satisfacción del cliente. Al seguir el proceso de gestión de cambios de ITIL, las organizaciones pueden garantizar que los cambios se realicen de manera controlada y estructurada, minimizando el riesgo de interrupción de los servicios (2).

Sin embargo, la empresa considera que las aplicaciones existentes en el mercado, como por ejemplo: Bartech (3), Easyvista (4) o Asana (5) son demasiado generales, por lo que se plantea la creación de una aplicación propia centrada en la gestión de cambios, considerando que esta podría ser bien recibida por las empresas del sector. El cometido de este proyecto será el diseño e implementación de la base de datos necesaria para esta aplicación, a medida, de gestión de cambios. En el proyecto se definirá la estructura de la base de datos que dé soporte a la aplicación de gestión de cambios y que permitirá el control de los requisitos exigidos por la empresa y la ejecución de todas las consultas que se consideren necesarias para la correcta gestión y análisis de los datos almacenados en el sistema. El desarrollo de esta futura aplicación queda fuera del alcance de este trabajo.

1.1.2 Aportación realizada

Como se ha indicado anteriormente, el objeto de este proyecto es el diseño e implementación de la base de datos que dé soporte a una aplicación de control de cambios para aplicaciones informáticas, que permitirá el control de los requisitos exigidos por la empresa y la ejecución de todas las consultas que se consideren necesarias para la correcta gestión y análisis de los datos almacenados en el sistema. A grandes rasgos se relacionan a continuación los requisitos con los debe contar la base de datos.

Para formalizar los cambios, la base de datos contará con un inventario de todas las aplicaciones, incluidos los contactos técnicos y de negocio, la fecha de puesta en producción, el promedio de los usuarios, la tecnología utilizada, el tipo de

infraestructura y la criticidad. Todos los cambios deben seguir un flujo de aprobación que involucre a los gerentes técnicos y de negocio, un gestor de cambios de la empresa y el Consejo Asesor de Cambio Global (GCAB). Todos estos aprobadores contarán con un sustituto que puntualmente disponga del rol de aprobador.

Se definirán como mínimo tres diferentes categorías de cambio: 1 (cambio crítico), 2 (cambio standard) y 3 (cambio poco importante); y un plan de acción para su ejecución con incidencias o ejecución reprogramada. La solicitud también debe considerar el alcance geográfico de los cambios y la posibilidad de auditar la ejecución del proceso de gestión de cambios. El modelo de datos debe permitir guardar todos los datos necesarios y se deben implementar procedimientos ABM.

Las consultas mínimas requeridas incluyen número de cambios en aprobación, cambios aprobados no ejecutados correctamente, responsable técnico con más cambios ejecutados correctamente, porcentaje de acciones definidas por ejecuciones incorrectas cerradas en el tiempo inicialmente definido, número total de cambios aprobados en el GCAB, número de cambios no aprobados, porcentaje de cambios cuya ejecución fue replanificada, número total de incumplimientos detectados durante las auditorías, número total de planes de acción sobre ejecuciones fallidas que están abiertos y persona específica con rol de aprobador que ha sido reemplazada con mayor antelación a una indisponibilidad no planificada. El desarrollo de una base de datos para un sistema específico debe ser escalable y capaz de incorporar funcionalidades y consultas adicionales. El informe debe detallar posibles líneas de evolución e incluir la inicialización con un conjunto suficiente de datos y pruebas exhaustivas para garantizar una funcionalidad correcta. También deberían incluirse mecanismos para resolver posibles problemas de integración y acciones de registro.

1.2 Objetivos del Trabajo

Los objetivos técnicos principales y secundarios del proyecto son los siguientes:

1.2.1 Objetivos técnicos principales

1. Diseñar una BD que permita gestionar un inventario de aplicaciones, incluyendo datos como la persona de contacto técnica, fecha de puesta en producción y tecnología utilizada.

2. Definir un modelo de datos que almacene información sobre cambios en aplicaciones, incluyendo cambios técnicos y funcionales.

3. Establecer un flujo de aprobación para cada cambio, involucrando a aprobadores como el responsable técnico, el responsable del área de negocio, el gestor de cambios de la empresa y el Consejo Asesor de Cambio Global (GCAB).

4. Registrar en la BD las aprobaciones de cambios, incluyendo quién las aprobó, su rol y la fecha de aprobación.

5. Permitir a los aprobadores poner comentarios en las aprobaciones, almacenándolos en la BD.

6. Definir categorías de cambio, con importancia de 1 (crítico) a 3 (poco importante), y permitir su gestión dinámica.

7. Registrar la fecha de creación y aprobación de cada categoría de cambio en la BD.

8. Capturar el alcance geográfico de los cambios y considerar el número de usuarios afectados y el impacto económico.

9. Diseñar un sistema de gestión de cambios que incluya la ejecución y calificación post-implementación.

10. Clasificar la ejecución de cambios en categorías como "Ejecución correcta", "Ejecución con incidencias" y "Ejecución replanificada".

11. Establecer planes de acción para mejoras en cambios con ejecución problemática y permitir su seguimiento en la BD.

12. Validar el cierre de planes de acción mediante la aprobación del gestor de cambios.

13. Habilitar auditorías del proceso de gestión de cambios y registrar incumplimientos detectados.

14. Diseñar un modelo de datos que pueda ser escalable para incorporar futuras necesidades.

15. Crear consultas estadísticas eficientes para acceder a los datos almacenados.

16. Garantizar que las consultas estadísticas se ejecuten en tiempo constante (1 SELECT por consulta).

17. Definir consultas mínimas, como el número de cambios en proceso de aprobación, cambios aprobados con ejecución incorrecta y responsable técnico con más cambios ejecutados de manera correcta.

18. Facilitar la identificación de aprobadores substitutos y gestionar su rol de aprobador.

19. Incluir mecanismos de registro de acciones en la BD para facilitar el mantenimiento y solución de problemas de integración.

20. Presentar un conjunto de datos de prueba y realizar pruebas exhaustivas para garantizar el correcto funcionamiento del sistema implementado.

1.2.2 Objetivos técnicos secundarios

1. Establecer un mecanismo de registro de excepciones para manejar errores inesperados durante la ejecución de procedimientos de BD.

2. Diseñar una estructura de control de versiones para la BD, permitiendo la reversión a versiones anteriores si es necesario.

3. Implementar procedimientos de Alta, Baja y Modificación (ABM) para las entidades relevantes en la BD.

4. Desarrollar un sistema de registro de log de acciones, incluyendo las llamadas a procedimientos y los parámetros utilizados.

5. Permitir la ejecución de consultas SQL complejas para extraer información detallada de la BD.

6. Crear procedimientos de BD para la inicialización de datos de prueba con el fin de realizar pruebas exhaustivas.

7. Diseñar una estructura de almacenamiento de datos eficiente para manejar grandes volúmenes de información.

8. Habilitar la generación de informes y análisis de datos a través de herramientas de Business Intelligence (BI).

9. Implementar mecanismos de seguridad para garantizar la protección de los datos almacenados en la BD.

10. Proporcionar documentación detallada que describa la estructura de la BD, los procedimientos de BD y las restricciones de integridad para facilitar futuras actualizaciones y mantenimiento del sistema.

1.3 Enfoque y método seguido

Para el proyecto de diseño e implementación de una base de datos para una aplicación de gestión de cambios en aplicaciones informáticas, existen varias estrategias posibles. Las más destacadas son:

Desarrollar un Producto Nuevo: esta estrategia implica diseñar y construir una base de datos desde cero, asegurando que se adapte específicamente a las necesidades y requisitos únicos de la empresa. Permite una personalización completa y adaptación precisa a los procesos ITIL y especificaciones de control de cambios.

Adaptar un Producto Existente: consiste en tomar una solución de *software* ya existente y modificarla para cumplir con los requisitos del proyecto. Esta estrategia puede ser más rápida y económica en términos de desarrollo, pero puede comprometer la capacidad de personalizar completamente la aplicación para satisfacer las necesidades específicas del negocio.

Integrar Soluciones de Múltiples Proveedores: esta estrategia combina componentes de diferentes aplicaciones existentes para crear una solución. Proporciona la flexibilidad para aprovechar diferentes sistemas, pero puede enfrentar desafíos para integrar y mantener la coherencia en la gestión de cambios.

Uso de Plataformas de Desarrollo de *Low Code*: utilizar plataformas que permitan el desarrollo de aplicaciones con una codificación manual mínima. Esta es una opción rápida y flexible, aunque puede limitar las funciones avanzadas o la personalización (6).

La estrategia seleccionada para este proyecto es **desarrollar un producto nuevo**. Aunque el desarrollo de un producto nuevo requiere más tiempo y recursos iniciales en comparación con otras estrategias, ofrece la mejor combinación de personalización, escalabilidad y alineación con los objetivos específicos del proyecto. Esta elección se debe a las siguientes razones:

Personalización: permite crear una solución que se adapta perfectamente a los requisitos específicos del proyecto y empresa. Esto es fundamental porque el proyecto tiene requisitos detallados y específicos que es posible que los productos existentes no cumplan completamente.

Flexibilidad y Escalabilidad: un desarrollo desde cero brinda la oportunidad de diseñar la aplicación y la base de datos con una arquitectura que pueda evolucionar y adaptarse a las necesidades futuras de la empresa.

Integración con Procesos Específicos: la empresa busca una herramienta que se alinee con procesos ITIL específicos para la gestión de cambios. Un desarrollo a medida facilita esta integración.

Control Total sobre la Solución: al desarrollar nuevos productos, las empresas tienen control total sobre cada aspecto del *software*, desde la funcionalidad hasta la seguridad y el mantenimiento, lo cual es fundamental para las aplicaciones que gestionan procesos comerciales críticos.

1.4 Impacto en sostenibilidad, ético-social y de diversidad

Este proyecto se organiza alrededor de estas tres grandes dimensiones

I. Sostenibilidad
II. Comportamiento ético y responsabilidad social (RS)
III. Diversidad (género, entre otros) y derechos humanos

Que a su vez se encuentran alineadas con los ODS (Objetivos de Desarrollo Sostenible 2030, ONU) [1] (7), con los que la UOC se encuentra públicamente comprometida.

1.4.1 Dimensión sostenibilidad

En el contexto de este proyecto, el concepto de sostenibilidad requiere crear y ejecutar un diseño que reduzca el impacto ambiental de la base de datos. Esto implica utilizar infraestructuras de TI energéticamente eficientes que dependan de fuentes de energía renovables. Además, la sostenibilidad se logra promoviendo prácticas sostenibles dentro de la gestión y utilización de la base de datos. Esto incluye abogar por la eficiencia en el almacenamiento y procesamiento de datos, así como apoyar la sostenibilidad de las comunidades y ecosistemas que dependen de la aplicación.

1.4.2 Dimensión comportamiento ético y de responsabilidad social (RS)

El objetivo principal de este aspecto es garantizar que la base de datos y la aplicación de gestión de cambios se creen y utilicen de manera ética y responsable. Esto implica respaldar la igualdad de accesibilidad, preservar la confidencialidad y la seguridad de los datos y garantizar que la aplicación asegure, en lugar de socavar, el bienestar social y económico. La base de datos debe funcionar como un catalizador para facilitar prácticas laborales equitativas, promover la educación y la atención médica y contribuir activamente a la paz y la justicia dentro de las comunidades a las que sirve.

1.4.3 Dimensión diversidad, género y derechos humanos

En este ámbito se ha de garantizar que tanto la base de datos como la aplicación sean integrales y respeten los derechos humanos fundamentales. Esto requiere promover la paridad de género en todas las facetas de la empresa, garantizar que la aplicación pueda ser utilizada por personas de diferentes capacidades y orígenes, y abogar por la inclusión y la reducción de las disparidades a través de su creación e implementación. En el desarrollo de equipos, el diseño de bases de datos y la recopilación y uso de datos, la diversidad debe ser un factor crucial a considerar.

[1] En 2015, la ONU aprobó la Agenda 2030, sobre el Desarrollo Sostenible, que cuenta con 17 Objetivos de Desarrollo Sostenible (ODS) que se pueden consultar en: https://www.un.org/sustainabledevelopment/es/.[1]

Cada una de estas dimensiones contribuye a un enfoque integral, asegurando que el proyecto no solo sea técnicamente exitoso, sino también social y ambientalmente responsable.

1.5 Planificación del Trabajo

Se describen a continuación las tareas a realizar y una planificación temporal de cada tarea utilizando un diagrama de Gantt. Además se destacan los hitos parciales de cada una de las PEC.

1.5.1 Tareas del Trabajo

Las fases del proyecto son las definidas en la metodología en cascada (*waterfall*), donde las diferentes etapas se realizan una a continuación de la anterior (8). Se adopta este sistema de trabajo, fundamentalmente, porque el equipo de trabajo está compuesto por una sola persona, por lo que será más efectivo centrarse en una única fase hasta finalizarla, que realizar iteraciones, requeridas en otros métodos, como por ejemplo los métodos Agile. La sencillez de esta metodología y que se trate de un proyecto bien definido, en cuanto a requerimientos y herramientas, son dos condiciones más que hacen de la metodología en cascada la más adecuada para este trabajo. Las fases de este método serán las siguientes:

- Investigación: en esta fase se asegura que la base de datos cumpla con los objetivos previamente establecidos, se obtienen y analizan los requisitos para el desarrollo del proyecto.

- Diseño de la base de datos: en esta fase se define la arquitectura del software y la planificación del proyecto.

- Desarrollo: en esta fase se desarrolla la programación del software y la exploración de los errores.

- Implantación: se integra el software en el entorno y se realizan las pruebas de la versión beta.

- Mantenimiento, se entrega y pone en producción la base de datos.

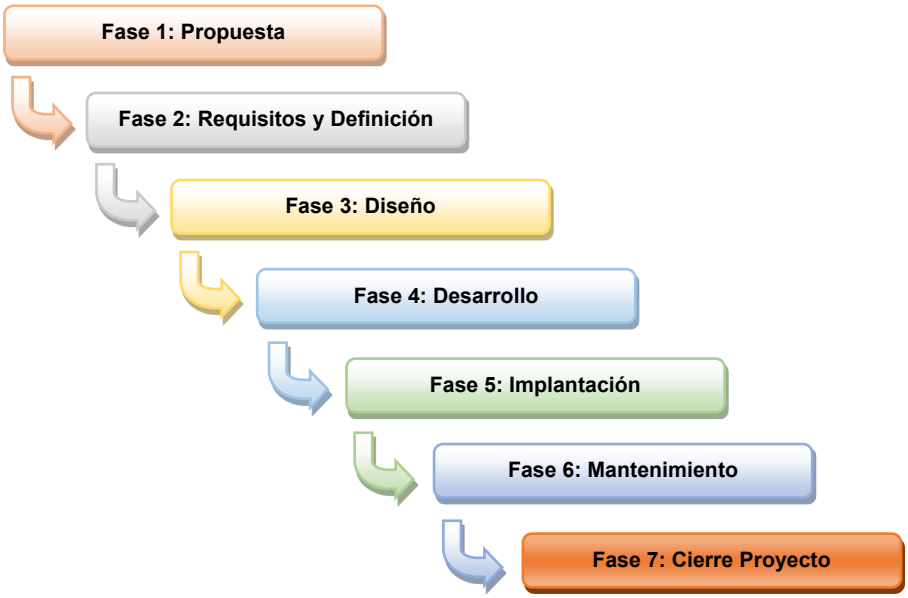


Figura 2: Fases de la metodología en cascada

En esta metodología en cascada se integrarán las cinco fases del diseño de una base de datos que constituirán las tareas propias del proyecto:

a) Requisitos y definición:

1. Recogida y análisis de requisitos

Esta fase implica identificar los actores del sistema de información que interactuarán con la base de datos, incluidos los usuarios y las aplicaciones, y recopilar y analizar sus requisitos. Esto incluye no solo requisitos relacionados con la estructura y forma de la información, sino también cualquier otro requisito que los usuarios esperan de la base de datos, como procesos que deben ejecutarse, restricciones de datos, restricciones de rendimiento, requisitos de seguridad y requisitos de rendimiento.

La fase de recopilación y análisis de requisitos se puede dividir en tres sub-fases: recopilación de requisitos, estructuración y refinamiento de requisitos y formalización de requisitos. Es importante involucrar a los usuarios de la base de datos durante el proceso de desarrollo para aumentar su grado de implicación y satisfacción. El siguiente paso es convertir los requisitos a un formato estructurado utilizando técnicas de especificación de requisitos como el análisis orientado a objetos, diagramas de flujo de datos o la notación Z.

Esta fase es muy importante ya que es necesario detectar y corregir errores o problemas en las fases iniciales del proyecto para evitar correcciones costosas en las fases finales. La capacidad de recopilar y capturar las necesidades de los usuarios e implementarlas correctamente en la solución final es crucial para la satisfacción del usuario final. (9).

b) Diseño:

2. Diseño conceptual

La fase de diseño conceptual del diseño de una base de datos tiene como objetivo crear un marco conceptual de alto nivel basado en los requisitos, especificaciones y restricciones recopilados en la fase anterior. El objetivo es diseñar un esquema conceptual de la base de datos que sea consistente con los requisitos y sirva como referencia para verificar que se cumplan todos los requisitos sin considerar el tipo de base de datos o el lenguaje de implementación específico.

El modelo entidad-relación (ER) es uno de los modelos de datos de alto nivel más utilizados en el diseño conceptual debido a su simplicidad y facilidad de uso. Incluye tipos de entidades, atributos y tipos de relaciones entre entidades para reflejar los requisitos del mundo real. El Lenguaje Unificado de Modelado (UML) es un lenguaje gráfico que incluye un diagrama de clases para representar información del dominio del discurso. Los diagramas de clases son diagramas estáticos que describen la estructura de un sistema en función de las clases o tipos de entidades del sistema, sus atributos y las asociaciones o tipos de relaciones que se establecen entre ellos. La fase de diseño conceptual se centra en la estructura de la información sin resolver cuestiones relacionadas con la tecnología. (9).

3. Diseño lógico

La fase de diseño lógico del diseño de una base de datos implica elegir un tipo de base de datos y transformar el modelo conceptual en un modelo lógico que depende del tipo de base de datos elegido. El tipo de base de datos elegida determina el diseño del esquema lógico. En esta fase, la atención se centra en cuestiones tecnológicas relacionadas con el modelo de base de datos. El resultado de esta fase es un modelo lógico de la estructura de la información, que generalmente es un modelo relacional si se elige un sistema de gestión de bases de datos relacionales.

La fase de diseño lógico tiene tres sub-fases, que se aplican secuencialmente: reconsideraciones del modelo conceptual, transformación del modelo conceptual en modelo lógico y estandarización. En la primera sub-fase se realiza un análisis en profundidad del modelo conceptual para detectar y corregir trampas de diseño. En la segunda sub-fase, el diagrama de clases UML obtenido de la etapa de diseño conceptual se transforma utilizando una estructura de datos de modelo relacional. En la tercera sub-fase, se aplica la teoría de la normalización para garantizar que se cumplan buenos criterios de diseño, utilizando formas normales. El modelo relacional es el modelo lógico específico para bases de datos relacionales, con funciones específicas propias de cada forma normal. (9).

c)Desarrollo:

4. Diseño físico

La fase de diseño físico del diseño de una base de datos implica elegir un SGBD específico y adaptar el esquema lógico obtenido en la fase anterior al sistema elegido. Es importante considerar aspectos de implementación física y eficiencia que dependen específicamente del DBMS elegido. Se deben estudiar los niveles físicos y virtuales de las bases de datos para ver aspectos como las estructuras de almacenamiento y las rutas de acceso a través de los archivos de las bases de datos.

Cada SGBD ofrece diferentes opciones para organizar archivos y rutas de acceso, y es necesario conocer la implementación concreta para implementar de manera más eficiente el diseño físico del sistema de información. La transformación del modelo lógico al modelo físico implica transformar el modelo lógico de una base de datos en un modelo físico, de acuerdo con el SGBD elegido para implementar el sistema de información, las características específicas del hardware utilizado, el sistema operativo y el software básico.

Cada SGBD ha desarrollado su propio lenguaje para implementar el diseño físico de la base de datos, según las características del entorno, y obtener el máximo rendimiento del hardware, del sistema operativo y del propio gestor. El estándar SQL incorpora la definición de todos los componentes lógicos de la base de datos de diseño, pero no contiene ningún elemento de diseño físico. Para transformar el diseño lógico de la base de datos en un SGBD específico, se parte de las definiciones de la tabla para luego relacionar cada elemento con un espacio apropiado en el nivel virtual y finalmente relacionar cada espacio virtual con un archivo físico, que constituye el nivel físico de la base de datos. sistema de información. (9).

d) Implantación:

5. Implementación y optimización

La etapa de implementación y optimización del diseño de la base de datos implica cargar datos y ajustar parámetros para optimizar el rendimiento. El objetivo principal es optimizar el rendimiento de la base de datos determinando el tamaño de las tablas, los tipos de accesos y consultas y la frecuencia. También se especifican roles de usuario y aplicaciones para determinar los permisos de diferentes grupos. El procesamiento y la optimización de consultas es un aspecto crítico del diseño de la base de datos, ya que determina el tiempo que el sistema de gestión necesita para resolver las consultas de los usuarios.

El lenguaje SQL utilizado por las bases de datos relacionales es declarativo, lo que significa que el resultado deseado se especifica a partir de una consulta realizada. El procesamiento de consultas incluye todo el

conjunto de actividades que realiza el SGBD, que tienen como objetivo extraer información de la base de datos para lograr la estrategia más eficiente y proporcionar un mejor rendimiento del sistema de información. El procesamiento de vistas implica la creación de tablas lógicas que permiten el acceso a la información de una o varias tablas mediante una consulta predefinida. Las vistas proporcionan mecanismos de seguridad y permiten al diseñador de la base de datos personalizar la vista que tienen los diferentes usuarios de la base de datos. Finalmente, la gestión de la seguridad implica proteger la base de datos contra accesos no autorizados y asignar y revocar privilegios de diferentes usuarios. El componente de seguridad del DBMS es responsable de estas acciones y se ha vuelto cada vez más importante en la protección de información confidencial. (9).

1.5.2 Descripción de las tareas específicas del Trabajo

Conforme a las fases del diseño de bases de datos expuestas en el apartado anterior se definen las tareas a llevar a cabo durante el desarrollo del proyecto:

TAREAS	**Fase 1 - Propuesta**	
	Descripción del proyecto	Se proporciona una explicación general del proyecto, incluyendo su alcance y objetivos.
	Objetivos técnicos	Se establecen los objetivos técnicos específicos que se deben lograr con el proyecto.
	Estudio de viabilidad	Se realiza una evaluación técnica de la viabilidad del proyecto.
	Fases del proyecto	Se identifican y definen las etapas clave que compondrán el proyecto.
	Listado de tareas	Se enumera y detalla cada una de las actividades y tareas necesarias para llevar a cabo el proyecto.
	Planificación	En esta etapa, se crea un plan general del proyecto, que incluye una visión de alto nivel de la cronología y los recursos necesarios.
	Fechas clave y diagrama de Gantt	Se establecen fechas importantes y se elabora un diagrama de Gantt que visualiza la secuencia y duración de las tareas.
	Fase 2 - Investigación	
	1. RECOGIDA Y ANÁLISIS DE REQUISITOS	Se recopilan y analizan los requisitos del sistema, involucrando a las partes interesadas y documentando sus necesidades.
	Recolección de requisitos	Se lleva a cabo la recolección de información y requisitos necesarios para el diseño de la base de datos, acorde a la documentación aportada.
	Estructuración y refinamiento de requisitos	Los requisitos recopilados se organizan y refinan para garantizar su coherencia y comprensión.
	Formalización de requisitos	Se documentan formalmente los requisitos en un formato estructurado.
	Redacción documento PEC1	Se prepara un documento entregable acorde a los requerimientos y puntualizaciones del consultor.
	Fase 3 - Diseño	
	2. DISEÑO CONCEPTUAL	Se crea un diseño conceptual del sistema, que define la estructura general y las relaciones de los datos.

	Conceptualización estructura del sistema	Se describe la estructura del sistema en función de las clases o tipos de entidades del sistema, sus atributos y las asociaciones o tipos de relaciones que se establecen entre ellos.
	Elaboración del diagrama UML	Se utiliza UML (Lenguaje de Modelado Unificado) para elaborar diagramas que representen visualmente el diseño conceptual.
	***Feedback* Consultor**	Se busca retroalimentación y orientación del consultor para refinar y mejorar el diseño.
	Inclusión PEC1 en memoria	Se incorpora la PEC1 en la documentación general del proyecto.
	Redacción documento memoria	Se incorpora todo lo trabajado hasta el momento a la memoria del proyecto.
	Fase 4 - Desarrollo	
	3. DISEÑO LÓGICO	Se establece un diseño lógico, que incluye la elección del tipo de base de datos y la transformación del modelo conceptual en un modelo lógico.
	Elección tipo base de datos	Se selecciona el tipo de base de datos que mejor se adapte a los requisitos del proyecto, centrándose en cuestiones tecnológicas relacionadas con el modelo de base de datos.
	Revisión modelo conceptual	Se realiza un análisis en profundidad del modelo conceptual para detectar y corregir errores de diseño.
	Transformación del modelo conceptual en modelo lógico	Se realiza la conversión del diseño conceptual en un modelo lógico, que incluye la estructura de tablas y relaciones. El diagrama de clases UML obtenido de la etapa de diseño conceptual se transforma utilizando una estructura de datos de modelo relacional.
	Estandarización	Se aplica la teoría de la normalización para garantizar que se cumplan buenos criterios de diseño, utilizando formas normales.
	***Feedback* Consultor**	Se busca la retroalimentación y asesoramiento del consultor.
	4. DISEÑO FÍSICO	Se realiza el diseño físico de la base de datos, que incluye detalles como la selección del sistema de gestión de bases de datos (SGBD), la instalación del entorno Oracle y la transformación del modelo lógico en un modelo físico.
	Búsqueda información SGBD	Se investiga y recopila información relevante sobre el sistema de gestión de bases de datos seleccionado.
	Selección SGBD	Se elige el SGBD que mejor se ajuste a las necesidades del proyecto.
	Instalación entorno Oracle	Se configura y establece el entorno Oracle como parte del diseño físico.
	Transformación el modelo lógico en modelo físico	Se convierte el modelo lógico en una implementación física de la base de datos, de acuerdo con el SGBD elegido para implementar el sistema de información, las características específicas del hardware utilizado, el sistema operativo y el software básico
	***Feedback* Consultor**	Se obtiene la retroalimentación del consultor sobre el diseño físico.
	Redacción documento memoria	Se incorpora todo lo trabajado hasta el momento a la memoria del proyecto.
	Fase 5 – Implantación	

	5. IMPLEMENTACIÓN Y OPTIMIZACIÓN	Se procede con la implementación del sistema, que incluye la creación y carga de tablas, la ejecución de scripts y la creación de procedimientos. También se optimiza el rendimiento del sistema.
	Creación y carga de tablas	Se generan las tablas necesarias para almacenar los datos y se cargan con información relevante.
	Ejecución de scripts de la BD	Se ejecutan scripts en la base de datos para configurar y ajustar su funcionamiento.
	Creación de procedimientos	Se desarrollan procedimientos que permiten el funcionamiento adecuado del sistema.
	***Feedback* Consultor**	Se obtiene la retroalimentación del consultor sobre la implementación y optimización.
	Fase 6 - Mantenimiento	
	Elaboración plan de pruebas	Se crea un plan detallado de pruebas para evaluar el funcionamiento del sistema.
	Elaboración scripts pruebas	Se desarrollan scripts y casos de prueba para ejecutar las pruebas planificadas.
	Depuración y corrección de errores	Se identifican y corrigen los errores y problemas encontrados durante las pruebas.
	***Feedback* Consultor**	Se obtiene la retroalimentación del consultor sobre el mantenimiento del sistema.
	Redacción documento memoria	Se incorpora todo lo trabajado hasta el momento a la memoria del proyecto.
	Elaboración vídeo presentación	Se crea un vídeo de presentación que resume y comunica de manera efectiva los aspectos clave del proyecto de diseño de bases de datos.
	Elaboración informe autoevaluación	Se realiza un informe de autoevaluación en el que se analiza y evalúa el propio trabajo realizado en el proyecto.
	Fase 7 - CIERRE PROYECTO	
	Preparación defensa virtual	Antes de la defensa virtual, se realiza una preparación exhaustiva que implica repasar la presentación, ensayar respuestas a posibles preguntas y asegurarse de que todos los aspectos del proyecto estén bien entendidos y comunicados.
	Defensa virtual	Se responde a preguntas y se defienden las decisiones tomadas en el proyecto.

Tabla 1: Descripción de tareas del Trabajo

1.5.3 Planificación del Trabajo

Las fases del proyecto son las habitualmente definidas en la metodología en cascada, que se han definido en un apartado anterior, incluyendo alguna más para adaptar la elaboración del trabajo a los entregables del proyecto:

a) Fechas clave:

Como fechas clave se consideran las fechas de inicio y fin de los entregables del proyecto:

	FECHAS CLAVE					
	DESCRIPCIÓN		INICIO	FIN	DÍAS	HORAS
TAREAS	PEC1	Plan de trabajo	28/09/2023	09/10/2023	12	30,0
	PEC2	Seguimiento	10/10/2023	13/11/2023	35	87,5
	PEC3	Seguimiento	14/11/2023	18/12/2023	35	87,5
	PEC4	Entrega final TFG	19/12/2023	15/01/2024	28	70,0
	Defensa		16/01/2024	26/01/2024	11	27,5
					121	**302,5**

Tabla 2: Fechas clave

Se contemplan 121 días de trabajo en total, en función de dos horas y media al día de dedicación al proyecto, lo que supone un total de 302,5 horas empleadas en el mismo.

b) Hitos (Milestones)

A continuación se adjunta una tabla de los hitos previstos para este proyecto. Estos hitos reflejan todas las fases, tareas y subtareas del trabajo:

		HITOS		
		INICIO	FIN	DÍAS
TAREAS	**Fase 1 - Propuesta**	**28/09/2023**	**03/10/2023**	**6**
	Inicio Entregable1	**28/09/2023**	**28/09/2023**	**0**
	Descripción del proyecto	28/09/2023	28/09/2023	1
	Objetivos técnicos	29/09/2023	29/09/2023	1
	Estudio de viabilidad	30/09/2023	30/09/2023	1
	Fases del proyecto	01/10/2023	01/10/2023	1
	Listado de tareas	02/10/2023	02/10/2023	1
	Planificación	03/10/2023	03/10/2023	1
	Fechas clave y diagrama de Gantt	03/10/2023	03/10/2023	1
	Fase 2 - Investigación	**04/10/2023**	**09/10/2023**	**6**
	1. Recogida y análisis de requisitos	04/10/2023	06/10/2023	3
	Recopilación de requisitos	04/10/2023	04/10/2023	1
	Estructuración y refinamiento de requisitos	05/10/2023	05/10/2023	1
	Formalización de requisitos	06/10/2023	06/10/2023	1

	Redacción documento PEC1	07/10/2023	09/10/2023	3
	Entregable1	**09/10/2023**	**09/10/2023**	**0**
	Fase 3 - Diseño	**10/10/2023**	**13/11/2023**	**35**
	Inicio Entregable2	**10/10/2023**	**10/10/2023**	**0**
	2. Diseño conceptual	10/10/2023	09/11/2023	31
	Conceptualización estructura del sistema	10/10/2023	24/10/2023	15
	Elaboración del diagrama UML	25/10/2023	09/11/2023	16
	***Feedback* Consultor**	10/11/2023	10/11/2023	1
	Inclusión PEC1 en memoria	11/11/2023	11/11/2023	1
	Redacción documento memoria	12/11/2023	13/11/2023	2
	Entregable2	**13/11/2023**	**13/11/2023**	**0**
	Fase 4 - Desarrollo	**14/11/2023**	**18/12/2023**	**35**
	Inicio Entregable3	**14/11/2023**	**14/11/2023**	**0**
	3. Diseño lógico	14/11/2023	30/11/2023	17
	Elección tipo base de datos	14/11/2023	14/11/2023	1
	Revisión modelo conceptual	15/11/2023	17/11/2023	3
	Transformación del modelo conceptual en modelo lógico	18/11/2023	28/11/2023	11
	Estandarización	29/11/2023	30/11/2023	2
	***Feedback* Consultor**	01/12/2023	01/12/2023	1
	4. Diseño físico	02/12/2023	15/12/2023	14
	Búsqueda información SGBD	02/12/2023	02/12/2023	1
	Selección SGBD	03/12/2023	03/12/2023	1
	Instalación entorno Oracle	04/12/2023	04/12/2023	1
	Transformación el modelo lógico en modelo físico	05/12/2023	15/12/2023	11
	***Feedback* Consultor**	16/12/2023	16/12/2023	1
	Redacción documento memoria	17/12/2023	18/12/2023	2
	Entregable3	**18/12/2023**	**18/12/2023**	**0**
	Fase 5 - Implantación	**19/12/2023**	**04/01/2024**	**17**
	Inicio Entregable4	**19/12/2023**	**19/12/2023**	**0**
	5. Implementación y optimización	19/12/2023	03/01/2024	16
	Creación y carga de tablas	19/12/2023	24/12/2023	6
	Ejecución de scripts de la BD	25/12/2023	28/12/2023	4
	Creación de procedimientos	29/12/2023	03/01/2024	6
	***Feedback* Consultor**	04/01/2024	04/01/2024	1
	Fase 6 - Mantenimiento	**05/01/2024**	**15/01/2024**	**11**
	Elaboración plan de pruebas	05/01/2024	05/01/2024	1
	Elaboración scripts pruebas	06/01/2024	08/01/2024	3
	Depuración y corrección de errores	09/01/2024	11/01/2024	3
	***Feedback* Consultor**	12/01/2024	12/01/2024	1
	Redacción documento memoria	13/01/2024	13/01/2024	1
	Elaboración vídeo presentación	14/01/2024	14/01/2024	1
	Elaboración informe autoevaluación	15/01/2024	15/01/2024	1
	Entregable4	**15/01/2024**	**15/01/2024**	**0**

	Fase 7 - CIERRE PROYECTO	**16/01/2024**	**26/01/2024**	**11**
	Inicio Defensa virtual	**16/01/2024**	**16/01/2024**	**0**
	Preparación defensa virtual	**16/01/2024**	**21/01/2024**	**6**
	Defensa virtual	**22/01/2024**	**26/01/2024**	**5**
	Cierre Proyecto	**26/01/2024**	**26/01/2024**	**0**

Tabla 3: Hitos

c) Diagrama de Gantt

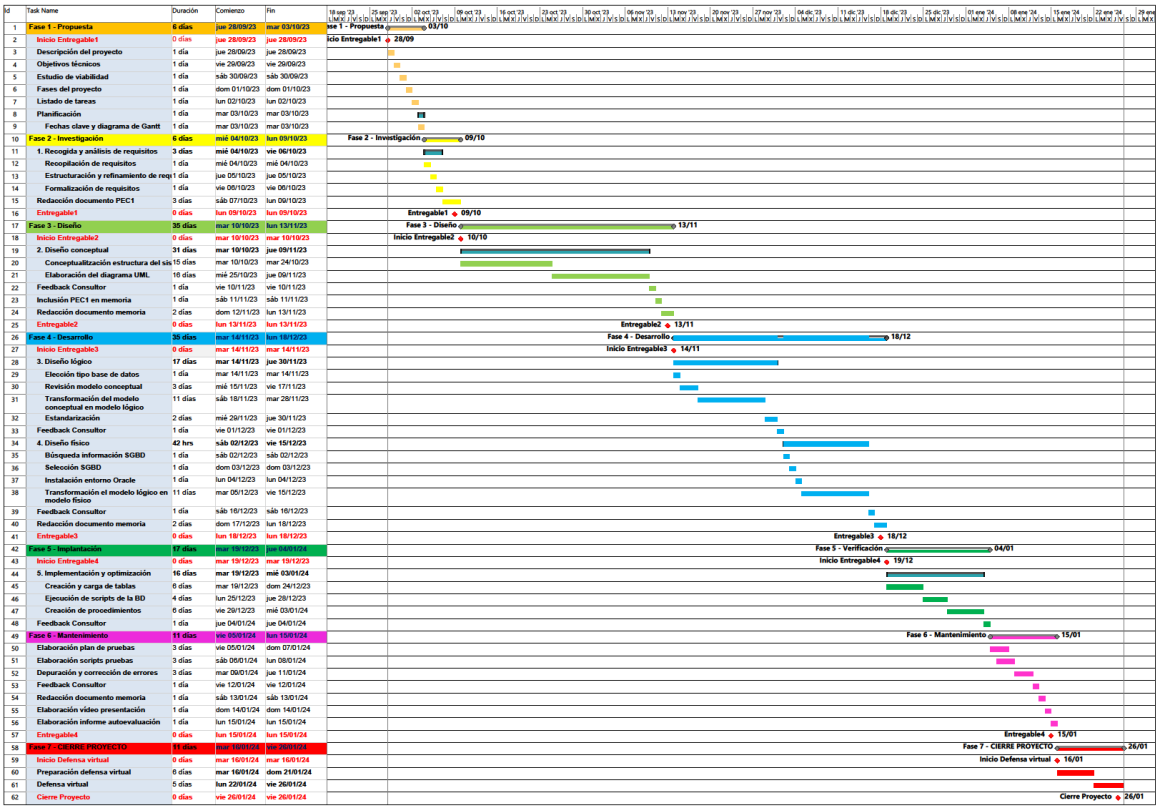


Tabla 4: Diagrama de Gantt

Se anexa a este proyecto un fichero Microsoft Project para poder visualizar mejor este diagrama de Gantt.

d) Desviaciones conforme a planificación

El consultor amplía el plazo de entrega del entregable correspondiente a la PEC3 en tres días, del 18 al 21 de diciembre. Aunque en la planificación del proyecto se había contemplado la entrega de dicho entregable el 18 de diciembre, por motivos laborales, debido a una mayor carga de trabajo en el último mes del año, se produce un retraso en el desarrollo del diseño físico del proyecto. Por tanto, se aprovecha esta prórroga, retrasando la entrega en tres días. Esto supondrá tener que acortar la fase de optimización e implementación también en tres días, por lo que será necesario realizarla en 13 días en lugar de 16. A priori, no se

trata de una desviación crítica ya que se dispondrán de más horas de trabajo al poder aprovechar el periodo vacacional.

1.6 Viabilidad del proyecto

Los objetivos establecidos en el proyecto, son bastante realistas, en base a que el conocimiento acerca de las tecnologías de bases de datos de Oracle no es otro que el adquirido en el estudio de las asignaturas del Grado de Ingeniería Informática. No obstante, en dichos objetivos se establece un orden de prioridad, y se considera que la consecución de los objetivos secundarios no es tan importante como la de los principales, por lo que en el caso de encontrarse con algún imprevisto que afecte negativamente a la planificación estos pueden dejar de ejecutarse sin comprometer el objeto global del proyecto.

Como se ha expuesto en los apartados anteriores se ha establecido una planificación de 121 jornadas, considerando una media de dedicación horaria de 2 horas y media al día, lo cual resulta un total de 302,5 horas de dedicación total al proyecto, esta previsión supera la equivalencia en horas de los 12 créditos de la asignatura. En base a la distribución de las tareas en el tiempo del que se dispone, la planificación está bastante ajustada, por lo que cualquier desviación requerirá de una replanificación de tareas para lograr cumplir con las fechas clave (entregas) del proyecto.

1.6.1 Análisis de riesgos

En este proyecto de diseño e implementación de una base de datos para una aplicación de control de cambios, se pueden identificar varios riesgos potenciales, junto con sus respectivos planes de contingencia:

1. Riesgo de requisitos incompletos o cambiantes
Plan de contingencia: implementar revisiones periódicas de requisitos y estar preparados para ajustes ágiles en el diseño y desarrollo.

2. Riesgo tecnológico en la implementación
Plan de contingencia: realizar una investigación exhaustiva de las tecnologías y herramientas antes de su implementación. Probar prototipos y versiones preliminares para asegurarse de que las tecnologías elegidas sean adecuadas para los objetivos del proyecto.

3. Riesgo de sobrecarga de datos y problemas de rendimiento
Plan de contingencia: diseñar la base de datos con escalabilidad y eficiencia en mente desde el principio. Realizar pruebas de carga para garantizar que el sistema pueda manejar volúmenes de datos grandes y picos inesperados de actividad.

4. Riesgo de seguridad de datos
Plan de contingencia: implementar medidas de seguridad robustas, incluyendo encriptación de datos, controles de acceso y auditorías de seguridad regulares. Preparar planes de respuesta ante incidentes de seguridad.

5. Riesgo de retrasos en el cronograma
Plan de contingencia: establecer un cronograma realista con hitos claros y tiempo asignado para revisiones y pruebas. Monitorizar el progreso regularmente y ajustar los recursos o estrategias según sea necesario.

6. Riesgo de problemas mantenimiento
Plan de contingencia: desarrollar una documentación exhaustiva y mantener mecanismos para pruebas continuas y resolución de problemas..

1.6.2 Recursos necesarios

Para la realización de este proyecto, se disponen de las siguientes herramientas y conocimientos básicos de las mismas:

a) *Software:*

- Oracle Database Expres Edition (10): https://www.oracle.com/es/database/technologies/xe-downloads.html
- Documentación Oracle Database Expres Edition (11): https://www.oracle.com/database/technologies/appdev/xe.html
- Oracle SQL Developer (12): https://www.oracle.com/database/technologies/appdev/sql-developer.html
- Documentación de Oracle Developer (12): https://www.oracle.com/database/technologies/appdev/sql-developer.html
- Draw.io: Elaboración de diagrama UML (13)
- Microsoft Project: Elaboración de diagramas Gantt

b) *Hardware:*

- Ordenador Portátil: MSI Creator (Procesador: 12th Gen Intel(R) Core(TM) i7-12700H 2.70 GHz / RAM: 32,0 GB)
- Sistema operativo: Windows 11

1.7 Breve sumario de productos obtenidos

Este Trabajo se compone de varios componentes clave, cada uno aborda diferentes aspectos del diseño y ejecución del sistema de bases de datos:

1. **Memoria de Trabajo Fin de Grado:** este es el documento principal que detalla de forma exhaustiva el proceso de planificación y ejecución del sistema de bases de datos. Incluye el diseño conceptual, lógico y físico de la BD, así como funcionalidades futuras consideradas para su inclusión.

2. **Scripts del Sistema de BD:** conjunto de scripts SQL para Oracle Express que incluye la creación de la BD, usuarios, tablas, secuencias y procedimientos almacenados. Este componente es esencial para establecer la estructura de la base de datos y asegura que esta responda a todas las necesidades identificadas, incluyendo el repositorio estadístico y medidas para resolver problemas de integración, como un log para los procedimientos.

3. **Anexos y Pruebas del Repositorio Estadístico:** documento anexo con imágenes de las pruebas realizadas para las consultas del repositorio estadístico. Este anexo sirve para validar el correcto funcionamiento de la BD a través de datos de prueba y un script SQL específico.

4. **Presentación del Proyecto:** incluye una presentación en formato de video, que sintetiza y difunde el trabajo realizado, facilitando la comprensión y el alcance del proyecto.

5. **Autoinforme del Estudiante:** documento de autoevaluación donde el alumno reflexiona sobre su desempeño y las decisiones tomadas a lo largo del proyecto. Este componente es un requisito de la asignatura y ofrece una visión introspectiva del proceso de aprendizaje y desarrollo del proyecto.

1.8 Breve descripción de los otros capítulos de la memoria

En este apartado se incluye una breve explicación de los contenidos de cada capítulo y su relación con el proyecto global:

Capítulo 2. Recogida y análisis de requisitos: se exponen los requisitos recopilados. Se organizan y refinan para garantizar su coherencia y comprensión.

Capítulo 3. Diseño de la base de datos: este capítulo recoge el diseño conceptual de la base de datos

> **Capítulo 3.1. Diseño conceptual:** se crea un diseño conceptual del sistema, que define la estructura general y las relaciones de los datos.

Capítulo 4. Desarrollo de la base de datos: este capítulo recoge el diseño lógico y físico de la base de datos

> **Capítulo 4.1. Diseño lógico:** se establece un diseño lógico, que incluye la elección del tipo de base de datos y la transformación del modelo conceptual en un modelo lógico.
>
> **Capítulo 4.2. Diseño físico:** se realiza el diseño físico de la base de datos, que incluye detalles como la selección del sistema de gestión de bases de datos (SGBD), la instalación del entorno Oracle y la transformación del modelo lógico en un modelo físico.

Capítulo 5. Implementación: se procede con la implementación del sistema, que incluye la creación y carga de tablas, la ejecución de scripts y la creación de procedimientos. También se optimiza el rendimiento del sistema.

Capítulo 6. Pruebas: se expone un plan detallado de pruebas para evaluar el funcionamiento del sistema.

Capítulo 7. Conclusiones: este capítulo incluye las conclusiones del trabajo; una reflexión crítica sobre el logro de los objetivos planteados inicialmente; un análisis crítico del seguimiento de la planificación y metodología a lo largo del producto; y las líneas de trabajo futuro que no se han podido explorar en este trabajo y han quedado pendientes.

Capítulo 8. Glosario: definición de los términos y acrónimos más relevantes utilizados en esta Memoria.

Capítulo 9. Bibliografía: listado de las referencias empleadas en la elaboración de la memoria.

Capítulo 10. Anexos: listado de los apartados complementarios o que son demasiado extensos para incluir en esta Memoria.

2. Recogida y análisis de requisitos:

2.1 Recopilación de requisitos

Los requisitos para el diseño de la base de datos para una aplicación de control de cambios recopilados de “la empresa” son los siguientes:

- Gestión de Cambios ITIL: la BD debe ser capaz de gestionar eficientemente los cambios en aplicaciones informáticas siguiendo los lineamientos de ITIL.

- Inventario de Aplicaciones: debe existir un inventario en la BD que contenga todas las aplicaciones sujetas a control de cambios, incluyendo detalles como contacto técnico, contacto del área de negocio, fecha de puesta en producción, número medio de usuarios, tecnología usada, tipo de infraestructura, criticidad de la aplicación y otros datos relevantes.

- Flujo de Aprobación de Cambios: la BD debe soportar un flujo de aprobación para los cambios realizados, involucrando a diversos roles como el responsable técnico, el responsable del área de negocio, el gestor de cambios de la empresa y el Global Change Advisory Board (GCAB).

- Registro de Aprobaciones y Comentarios: cada aprobación y los comentarios relacionados deben ser registrados en la BD, indicando quién aprobó el cambio, su rol en el proceso y la fecha de aprobación.

- Categorización de Cambios: se deben poder definir y almacenar diferentes categorías de cambios, cada una con su nivel de importancia, y la posibilidad de añadir o eliminar categorías.

- Alcance Geográfico de Cambios: la BD debe considerar el alcance geográfico de los cambios, incluyendo el impacto económico y el número de usuarios afectados.

- Seguimiento Post-Aprobación y Planes de Acción: debe existir una funcionalidad para el seguimiento de cambios después de su aprobación, incluyendo la categorización de la ejecución del cambio y la posibilidad de almacenar planes de acción para mejoras.

- Auditoría de Proceso de Gestión de Cambios: La BD debe permitir el registro de auditorías y las no conformidades detectadas.

- Escalabilidad: La BD debe ser escalable para adaptarse a necesidades futuras.

- Inicialización y Pruebas: Deben incluirse scripts para la creación de la BD, inicialización con un conjunto adecuado de datos y un conjunto de pruebas para garantizar el correcto funcionamiento.

- Mantenimiento y Funcionalidad: Se valorará la inclusión de mecanismos para facilitar el mantenimiento y la integración del sistema, como un log de acciones y herramientas para probar la funcionalidad de la BD.

- Consultas Específicas: Deben poder realizarse consultas detalladas para evaluar aspectos como el número de cambios en proceso de aprobación, cambios no ejecutados correctamente, responsables técnicos con mayor número de cambios exitosos, entre otros.

2.2 Estructuración, refinamiento y formalización de requisitos

Los requisitos recopilados se han refinado para garantizar su coherencia y comprensión. También se han clasificado como funcionales y no funcionales; y se han numerado para obtener un formato estructurado y formal.

Los **requisitos funcionales** son declaraciones que describen las características y funciones específicas que debe tener un sistema para satisfacer los requisitos del usuario (14), es decir, definen cómo debe comportarse un sistema para satisfacer dichas necesidades o expectativas. Los requisitos funcionales son esenciales para garantizar que un sistema informático esté diseñado para realizar las funciones previstas de forma eficaz. Estos requisitos se capturan en casos de uso y definen el comportamiento básico de un sistema (15).

Los **requisitos no funcionales**, por otro lado, son declaraciones que definen cómo debe funcionar un sistema para satisfacer las expectativas del usuario. Describen los atributos de calidad del sistema, como rendimiento, escalabilidad, portabilidad, confiabilidad y seguridad (16). Los requisitos no funcionales son tan importantes como los funcionales y se utilizan para definir el alcance y la calidad del sistema.

La principal diferencia entre requisitos funcionales y no funcionales es que los requisitos funcionales describen lo que debe hacer un sistema, mientras que los requisitos no funcionales describen cómo debe funcionar un sistema.

Requisitos Funcionales	
RF-01	La BD debe soportar la gestión completa del proceso de cambios ITIL para aplicaciones informáticas, desde la identificación del cambio hasta su implementación y revisión.
	“... desarrollar una aplicación propia para gestionar uno de los procesos ITIL (Information Technology Infrastructure Library) más importantes, la gestión de cambios en aplicaciones informáticas."
RF-02	La BD debe mantener un inventario detallado de todas las aplicaciones sujetas a cambios, incluyendo información como contacto técnico, fecha de puesta en producción, y datos de criticidad.
	“...se deberá definir un inventario con todas las aplicaciones a considerar... En este inventario de aplicaciones debe constar, como mínimo, la persona de contacto técnica, la persona de contacto por del área de negocio implicada, la fecha de puesta en producción...”

RF-03	Debe gestionar un flujo de aprobación de cambios implicando roles como responsable técnico, responsable del área de negocio, gestor de cambios y GCAB.
	“Cada cambio que se haga en una aplicación, tanto técnico com funcional, debe seguir un flujo de aprobación que, como mínimo, ha de considerar a los siguientes aprobadores: Responsable técnico... Responsable del área de negocio implicada... Gestor de cambios de la empresa... GCAB (Global Change Advisory Board)...”
RF-04	La BD debe registrar cada aprobación de cambio, incluyendo el aprobador, su rol, fecha de aprobación y comentarios asociados.
	“...la aplicación deberá poder registrar todas las aprobaciones que se realicen indicando, para cada una de ellas, quién la ha aprobado, con qué rol dentro del proceso y cuándo lo ha hecho. Cada aprobador también tendrá la posibilidad de poner comentarios a les aprobaciones...”
RF-05	Capacidad para clasificar los cambios en distintas categorías de importancia y almacenar esta clasificación.
	“Dentro del proceso de gestión de cambios, se han definido diferentes categorías de cambio... El sistema de BD ha de permitir almacenar tantas categorías como se quieran...”
RF-06	Gestión de sustitutos para cada rol de aprobación en ausencias planificadas o no planificadas.
	“Todos los aprobadores listados en el apartado anterior deben de tener un substituto para actuar de manera inmediata en caso de ausencia planificada (vacaciones, permisos,...) o no planificada (enfermedad, despido,..).”
RF-07	Registro y seguimiento de todos los cambios realizados, incluyendo historial detallado.
	Este requisito se infiere de la necesidad general de trazabilidad en los sistemas de gestión de cambios ITIL.
RF-08	La BD debe considerar y registrar el alcance geográfico y el impacto económico de los cambios.
	“Un tema importante a considerar también es el alcance geográfico de los cambios... además del número de usuarios afectados en caso de problemas en la ejecución del cambio, también se debe considerar el impacto económico en el negocio de cualquier no disponibilidad de la aplicación.”
RF-09	Capacidad para registrar el resultado de la implementación de cada cambio, incluyendo la clasificación de su ejecución.
	“El responsable técnico del cambio debe informar al gestor de cambios de la empresa de cómo ha ido el paso a producción, calificándolo en una de les siguientes categorías: Ejecución correcta... Ejecución con incidencias... Ejecución replanificada...”
RF-10	Almacenamiento y gestión de planes de acción para cambios con ejecuciones no satisfactorias.
	“Para las 2 últimas categorías de la ejecución, se deberá definir un plan de acción para mejorar los diferentes aspectos que no hayan ido como se esperaba.”

RF-11	La BD debe permitir la auditoría del proceso de gestión de cambios y almacenar las no conformidades detectadas.
	“La aplicación también debe contemplar la posibilidad de que la ejecución del proceso de gestión de cambios sea auditado para validar su correcta ejecución. Cada incumplimiento que se detecte en las auditorías se debe poder guardar en la BD.”
RF-12	Facilitar la realización de consultas específicas sobre aspectos del proceso de gestión de cambios.
	"…y la ejecución de las consultas que se consideren necesarias para la correcta gestión y análisis de los datos almacenados en el sistema."
RF-13	La BD debe permitir la gestión de roles y accesos según diferentes perfiles de usuario.
	Este requisito se infiere de la necesidad general de integración y compatibilidad en sistemas empresariales.
RF-14	Integración con herramientas externas de gestión de software y seguimiento de errores.
	Implícito en la naturaleza del proyecto de gestionar cambios en aplicaciones informáticas y la posible necesidad de integración con otras herramientas.
RF-15	Almacenar el historial completo de los cambios realizados en las aplicaciones.
	Implícito en la gestión eficiente de cambios y la trazabilidad requerida.
RF-16	La BD debe implementar un sistema de logs para registrar todas las operaciones realizadas. Esto incluye almacenar el nombre de cada procedimiento ejecutado, los parámetros de entrada y salida, y los resultados de la operación, facilitando así la trazabilidad y el control de errores.
	"Para el tema del log de las acciones realizadas, se recomienda almacenar todas las llamadas a procedimientos que se hagan en una tabla de log, almacenando el nombre del procedimiento ejecutado y los parámetros de entrada y de salida."

Tabla 5: Requisitos funcionales

Requisitos No Funcionales:	
RNF-01	Escalabilidad para adaptarse a futuras necesidades y expansiones.
	"La BD deberá de ser escalable para poder ir incorporando progresivamente todas aquellas necesidades que surjan durante su vigencia."
RNF-02	Alto rendimiento en el manejo de grandes volúmenes de datos y consultas eficientes.
	"Se deberá tener en cuenta que la aplicación ha de funcionar para cualquier volumen de datos, y por este motivo es muy importante que la gestión de los datos almacenados se haga siguiendo las técnicas que se aplican a grandes volúmenes de datos..."
RNF-03	Medidas de seguridad robustas para protección de datos.
	La seguridad de los datos es un requisito implícito y fundamental en cualquier sistema de base de datos, especialmente en uno que maneja procesos ITIL y datos de negocio sensibles.
RNF-04	Alta disponibilidad para garantizar el acceso continuo a la BD.
	La alta disponibilidad es crucial para garantizar que el sistema de base de datos esté siempre accesible, especialmente en un entorno empresarial donde los procesos de cambio son constantes.
RNF-05	Estrategias de recuperación ante desastres y copias de seguridad regulares.
	Al igual que la seguridad de los datos, la recuperación ante desastres y las copias de seguridad son requisitos implícitos en cualquier sistema de base de datos empresarial.
RNF-06	Documentación completa y soporte para facilitar el mantenimiento y la solución de problemas.
	"Finalmente, y con tal de facilitar el mantenimiento del sistema, se valorará muy positivamente el disponer de mecanismos que permitan resolver potenciales problemas de integración con el resto del sistema..."
RNF-07	Interoperabilidad con otros sistemas y formatos de datos.
	La interoperabilidad es un requisito implícito para sistemas que forman parte de un ecosistema IT más amplio.
RNF-08	Cumplimiento de normativas y estándares de seguridad y privacidad de datos.
	El cumplimiento de normativas y estándares de seguridad y privacidad de datos es esencial en sistemas de base de datos empresariales, especialmente en el contexto de ITIL y la gestión de cambios.

Tabla 6: Requisitos no funcionales

3. Diseño de la base de datos:

3.1 Diseño Conceptual

Este capítulo recoge el diseño conceptual de la base de datos. En la fase de diseño conceptual el objetivo es desarrollar un esquema conceptual de alto nivel basado en requisitos y restricciones previamente recopilados. Este esquema actúa como un marco de referencia para garantizar la consistencia con los requisitos, independientemente del tipo de base de datos o lenguaje de programación específico. Para representar la estructura de información del sistema, se ha empleado el Lenguaje Unificado de Modelado (UML) incluyendo entidades, atributos y relaciones, centrándose en la organización de la información más que en aspectos tecnológicos (9).

La notación empleada para el lenguaje UML es la siguiente:

- Nombre entidad: Se utilizan nombres en singular y grafía Pascal. Ejemplo: EstadoCambio.
- Atributos entidades: nombres en singular y grafía Camel. Ej.: idAprobador
- Atributos booleanos: nombre precedido de es, y grafía Camel. Ejemplo: esAprobador.
- Relaciones entre entidades: Se utilizan verbos en tercera persona del singular y grafía Camel. Ejemplo: pertenece.
- Relaciones entre entidades de tipo rol: Se utiliza un rol y grafía Camel. Ejemplo: responsable

Se adjunta el diagrama UML con la solución propuesta para la Base de Datos:

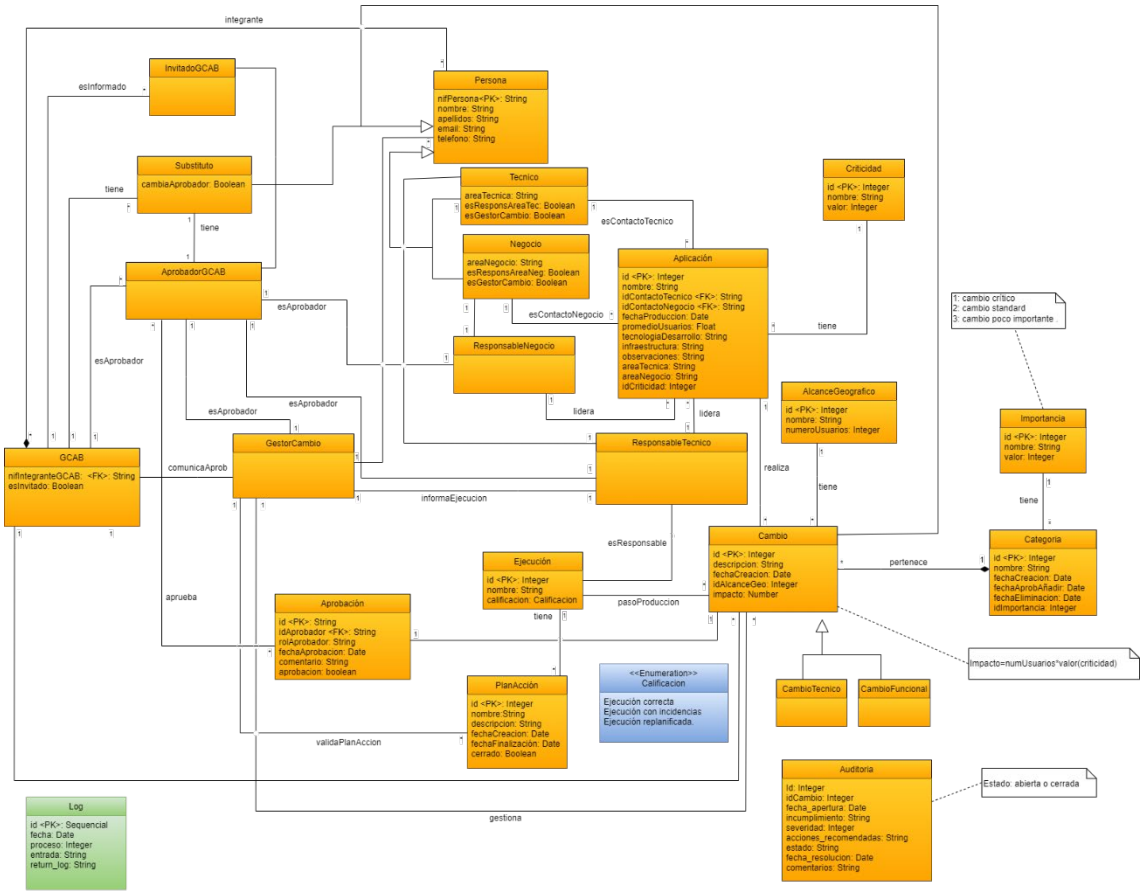


Tabla 7: Diagrama UML

Las entidades y atributos reflejados en este diagrama son los siguientes:

Entidad: **Persona** (recoge los datos delos actores en la gestión de datos)
Atributos:

- nifPersona: PK - String
- nombre: String
- apellido: String
- email: String
- telefono: String

Entidad: **Tecnico** (Hereda de Persona)
Atributos:

- areaTecnica: String
- esResponsAreaTec: Boolean
- esGestorCambio: Boolean

Entidad: **Negocio** (Hereda de Persona)
Atributos:

- areaNegocio: String

esResponsAreaNeg: Boolean
esGestorCambio: Boolean

Entidad: **Aplicación** (recoge los datos de la aplicación)
Atributos:
id: PK - Integer
nombre: String
idContactoTecnico: FK - String
idContactoNegocio: FK - String
fechaProduccion: Date
promedioUsuarios: Float
tecnologiaDesarrollo: String
infraestructura: String
observaciones: String
areaTecnica: String
areaNegocio: String
idCriticidad: FK - Integer

Entidad: **Criticidad** (recoge la criticidad de la aplicación con un valor del 1 al 5 de menor a mayor criticidad)
Atributos:
id: PK-> Integer
nombre: String
valor: Integer

Entidad: **AlcanceGeografico** (recoge el alcance geográfico medido por el número de usuarios)
Atributos:
id: PK - Integer
nombre: String
numeroUsuarios: Integer

Entidad: **Cambio** (entidad central que recoge todos los datos del cambio y el impacto que se calcula en función de la criticidad y el alcance de usuarios)
Atributos:
id: PK - Integer
descripcion: String
fechaCreacion: Date
idAlcanceGeo: FK - Integer
idAplicacion: FK - Integer
idCategoria: FK - Integer

Entidad: **Importancia** (recoge la importancia de la categoría del del cambio con un valor del 1 al 3 de menor a mayor importancia)
Atributos:
id: PK-> Integer
nombre: String
valor: Integer

Entidad: **Categoria** (recoge los datos de la categoría del cambio)

Atributos:
- id: PK - Integer
- nombre: String
- fechaCreacion: Date
- fechaAprobAnadir: Date
- fechaEliminacion: Date

Entidad: **Ejecucion** (recoge el estado de ejecución en tres estadios: 1: Ejecución correcta, 2: Ejecución con incidencias y 3: Ejecución replanificada)
Atributos:
- id: PK - Integer
- nombre: String
- idCambio: FK - Integer
- calificacion: Enum('Ejecución correcta', 'Ejecución con incidencias', 'Ejecución replanificada')

Entidad: **PlanAccion** (datos del plan de acción que se adopta si se dan los dos últimos estados del estado de ejecución)
Atributos:
- id: PK - Integer
- nombre: String
- descripcion: String
- fechaCreacion: Date
- fechaEjecucion: Date
- cerrado: Boolean
- idEjecucion: FK - Integer

Entidad: **GCAB** (recoge los datos de los integrantes y de su aprobación del cambio)
Atributos:
- nifIntegrante: PK-FK - String
- esInvitado: Boolean

Entidad: **Aprobacion** (recoge los datos de la aprobación de los cambios)
Atributos:
- id: PK - String
- idAprobador: FK - String
- fechaAprobacion: Date
- comentario: String
- aprobacion: Boolean

Entidad: **CambioTecnico** (hereda de cambio, hace referencia a un tipo de cambio técnico)

Entidad: **CambioFuncional** (hereda de cambio, hace referencia a un tipo de cambio funcional)

Entidad: **AprobadorGCAB** (recoge los datos de los aprobadores)
Atributos:
- Asume herencia de atributos de Persona

Entidad: **InvitadoGCAB** (datos de la persona invitada en la GCAB)
Atributos:
Asume herencia de atributos de Persona

Entidad: **Substituto** (datos de la persona substituta de los aprobadores)
Atributos:
Asume herencia de atributos de Persona

Entidad: **GestorCambio** (recoge los datos del Gestor de Cambios y de su aprobación del cambio)

Entidad: **ResponsableTecnico** (recoge los datos del Responsable Técnico, de su aprobación del cambio y del estado de ejecución)

Entidad: **ResponsableNegocio** (recoge los datos del Responsable de Negocio y de su aprobación del cambio)

Entidad: **Log** (recoge los datos para la auditoría del sistema)
Atributos:
id: PK Secuencial
fecha: Date
proceso: String
entrada: String
return_log: String

Entidad: **Auditoria** (recoge los datos de las auditorías realizadas en el proceso de gestión de cambios)
Atributos:
Id: Integer
idCambio: Integer
fechaApertura: Date
incumplimiento: String
severidad: Integer
accionesRecomendadas: String
estado: String
fechaResolucion: Date
comentarios: String

Algunas de las relaciones entre estas entidades son las siguientes:

Relaciones Directas (Asociaciones):

Persona - Invitado: Una asociación que indica que una Persona puede ser un Invitado. Con multiplicidad 1 a *.

Persona - Aprobador: Indica que una Persona puede tener el rol de Aprobador. Con multiplicidad 1 a *.

Persona - Tecnico/Negocio: Similar a Invitado/Aprobador, estas relaciones indican que una Persona puede especializarse en un Técnico o en un Negocio.

Aprobador - Substituto: Indica que un Aprobador puede tener un Substituto. La multiplicidad es 1 a 1 ya que la sustitución es obligatoria.

Aprobador - GestorCambio/ResponsableTecnico/ResponsableNegocio/GCAB: Cada una de estas clases es una especialización de Aprobador, indicando diferentes roles que un Aprobador puede tener.

Aplicacion - AlcanceGeografico/Criticidad: Estas asociaciones significan que cada Aplicacion tiene un AlcanceGeografico y una Criticidad asociados. La multiplicidad es 1 para AlcanceGeografico y Criticidad, lo que significa que cada aplicación tiene exactamente un alcance geográfico y una criticidad.

Cambio - Aplicacion: Un Cambio está asociado con una Aplicacion. La multiplicidad será 1 para Aplicacion, lo que significa que cada cambio está asociado con una sola aplicación.

Cambio - EstadoEjecucion/Categoria/PlanAccion: Indica que cada Cambio tiene un EstadoEjecucion y puede estar asociado con una Categoria y tener un PlanAccion. La multiplicidad de EstadoEjecucion y Categoria será 1, y PlanAccion *.

Cambio - Aprobacion: Significa que cada Cambio debe pasar por un proceso de Aprobacion. La multiplicidad será para Aprobacion, lo que significa que cada cambio puede tener una aprobación.

Aprobacion - Aprobador: Una Aprobacion está vinculada a un Aprobador. La multiplicidad será 1 para Aprobador, ya que cada aprobación debe estar vinculada a un solo aprobador.

Relaciones de Herencia (Generalización):
Tecnico/Negocio hereda de Persona.
GestorCambio/ResponsableTecnico/ResponsableNegocio hereda de Aprobador.

Relaciones Implícitas (por Atributos Compartidos):
Aprobacion podría estar vinculada implícitamente a Log si las aprobaciones se registran en un log.

3.1.1. Repositorios estadísticos

Dado el contexto de un sistema de gestión de cambios ITIL, los repositorios estadísticos podrían incluir datos sobre la frecuencia y tipos de cambios, la efectividad de los cambios y las tendencias en las aprobaciones:

Repositorio de Cambios y Categorías:
Un repositorio que recopile datos sobre los cambios realizados, incluyendo su categoría, criticidad e impacto. Esto puede ser útil para analizar qué tipos de cambios son más comunes y cuál es su impacto en el sistema.

Repositorio de Aprobaciones:
Este repositorio rastreará todas las aprobaciones, quién las realizó, y cuándo. Podría ser empleado para medir la eficiencia del proceso de aprobación y para identificar cuellos de botella.

Repositorio de Ejecuciones de Cambios:
Contendrá datos sobre cómo se ejecutan los cambios, incluyendo si las ejecuciones fueron exitosas y si se siguieron los planes de acción correspondientes.

Repositorio de Planes de Acción:
Agrupará información sobre los planes de acción, su duración y éxito. Será útil para evaluar la calidad y la efectividad de las respuestas a los problemas surgidos durante los cambios.

Repositorio de Auditorías:
Recopilará las instancias de incumplimiento detectadas en las auditorías y permitirá el análisis para mejorar los procesos y la conformidad.

4. Desarrollo de la base de datos:

Este capítulo recoge el diseño lógico y físico de la base de datos.

4.1 Diseño Lógico

El diseño lógico en una base de datos relacional se refiere a la fase en la que se define la estructura interna de la base de datos, basándose en el diseño conceptual. En esta etapa, se transforman los requisitos del diseño conceptual en un modelo lógico que puede ser implementado utilizando un sistema de gestión de bases de datos relacional (DBMS) (17). A continuación se relacionan los aspectos clave de esta etapa del diseño:

- Definición de Tablas y Relaciones: Se definen tablas para cada entidad identificada en el diseño conceptual. Cada tabla tiene un conjunto de atributos (columnas) que corresponden a las propiedades de la entidad.

- Claves Primarias y Foráneas: Se identifican las claves primarias para cada tabla, que son atributos únicos que identifican cada registro (fila) de la tabla. Las claves foráneas se utilizan para establecer relaciones entre tablas, haciendo referencia a las claves primarias de otras tablas.

- Normalización: El proceso de normalización implica la organización de datos en las tablas para minimizar la redundancia y mejorar la integridad

de los datos. Esto se logra a través de varias formas normales, cada una de las cuales reduce ciertos tipos de problemas de diseño.

- Definición de Restricciones de Integridad: Se establecen restricciones de integridad para asegurar la precisión y consistencia de los datos. Estas incluyen restricciones de clave primaria, clave foránea, restricciones únicas, y restricciones de chequeo (18).

- Vistas y Índices: Las vistas son consultas almacenadas que permiten a los usuarios interactuar con los datos de formas más complejas sin modificar la estructura de la base de datos. Los índices se utilizan para mejorar el rendimiento de las consultas.

- Mapeo del Modelo Entidad-Relación al Modelo Relacional: Este proceso implica convertir el modelo entidad-relación (ER) del diseño conceptual en un modelo relacional lógico, respetando las reglas y limitaciones del modelo relacional.

El diseño lógico es esencial para el rendimiento, la escalabilidad, la seguridad y la eficiencia de la base de datos. Debe ser realizado meticulosamente para asegurar que la base de datos final pueda manejar los requisitos de los usuarios y las aplicaciones que la utilizarán (19). En los siguientes apartados se incluyen los esquemas lógicos de la base de datos, se subrayan las claves primarias y se marcan en negrita los atributos obligatorios, es decir, que no pueden tomar el valor NULL. Los atributos que son únicos (claves alternativas) están subrayados en discontinuo:

4.1.1. Control de cambios

1. Persona (nifPersona, **nombre, apellidos, email, telefono)**

2. Rol (id)

3. Invitado (id)
   {id} is foreign key to Rol

4. EstadoRolAprobador (id, nombre)

5. Tecnico (**areaTecnica**, nifPersona)
   {nifPersona} is foreign key to Persona

6. Negocio (**areaNegocio**, nifPersona)
   {nifPersona} is foreign key to Persona

7. Substituto (id, **rolAprobador**, **idEstadoRolAprobador**)
   {id} is foreign key to Rol
   {idEstadoRolAprobador} is foreign key to EstadoRolAprobador

8. Aprobador (id, **idSubstituto**)
   {id} is foreign key to Rol

{idSubstituto} is foreign key to Substituto

9. Criticidad (id, **nombre, valor**)

10. Importancia (id, **nombre, valor**)

11. Categoria (id, **nombre**, fechaCreacion, fechaAccion, fechaEliminacion)
Check1(fechaAccion IS NULL OR fechaAccion >= fechaCreacion)
Check1(fechaEliminacion IS NULL OR fechaEliminacion >= fechaAccion)

12. Cambio (id, **descripcion, fechaCreacion**, **impacto**)

13. AlcanceGeografico (id, **nombre**, numeroUsuarios)

14. Aplicacion (id, nombre, **idContactoTecnico, idContactoNegocio, fechaProduccion**, promedioUsuarios, tecnologiaDesarrollo, infraestructura, observaciones)
{idContactoTecnico} is foreign key to Tecnico
{idContactoNegocio} is foreign key to Negocio

15. EstadoEjecucion (id, nombre, valor)

16. Aprobacion (id, idResponsable, fechaAprobador, comentario, aprobacion)
{idResponsable} is foreign key to Persona

17. PlanAccion (id, **nombre**, **accion**, **fechaCreacion**, fechaRealizacion)

18. CambioTecnico (id)
{id} is foreign key to Cambio

19. CambioFuncional (id)
{id} is foreign key to Cambio

20. GestorCambio (idGestorCambio, aprobacionGestor, idAprobacion)
{idAprobacion} is foreign key to Aprobacion

21. ResponsableTecnico (idResponsableTec, aprobTecnica, idEstadoEjecucion)
{idAprobador} is foreign key to Persona

22. ResponsableNegocio (idResponsableNeg, aprobNegocio)
{idResponsableNeg} is foreign key to Persona

23. CCAB (idCCAB, idGestor, idIntegrante, idInvitado, idSubstituto, aprobacionCCAB)
{idGestor} is foreign key to GestorCambio
{idInvitado } is foreign key to Invitado
{idIntegrante} is foreign key to Integrante
{idSubstituto} is foreign key to Substituto

4.1.2. Log

1. Log (id, **fecha**, **proceso, entrada, RSP,** observaciones)

4.1.3. Repositorios Estadísticos

Repositorio de Cambios y Categorías:

CambioCategoria (idCambio, idCategoria, idCriticidad, impacto)
{idCambio} is foreign key to Cambio
{idCategoria} is foreign key to Categoria
{idCriticidad} is foreign key to Criticidad

Repositorio de Aprobaciones:

AprobacionDetalle (idAprobacion, idResponsable, fechaAprobador)
{idAprobacion} is foreign key to Aprobacion
{idResponsable} is foreign key to Persona

Repositorio de Ejecuciones de Cambios:

EjecucionCambio (idEjecucion, idCambio, exitoso: Boolean, idPlanAccion)
{idEjecucion} is foreign key to EstadoEjecucion
{idCambio} is foreign key to Cambio
{idPlanAccion} is foreign key to PlanAccion

Repositorio de Planes de Acción:

PlanAccionDetalle (idPlanAccion, duracion, exitoso: Boolean)
{idPlanAccion} is foreign key to PlanAccion

Repositorio de Auditorías:

AuditoriaIncidencia (idAuditoria, idIncidencia, idCambio)
{idAuditoria} is foreign key to Auditoria
{idIncidencia} is foreign key to Incumplimiento
{idCambio} is foreign key to Cambio

4.2 Diseño Físico

El modelo físico de una base de datos relacional describe cómo se almacenan y organizan físicamente los datos definidos en el modelo lógico. A diferencia del modelo lógico, que se centra en la estructura de los datos y sus relaciones sin considerar cómo se almacenan físicamente los datos, el modelo físico se centra en el almacenamiento, la eficiencia y el rendimiento de las operaciones de la base de datos. A continuación se detallan los componentes clave del modelo físico en sistemas de bases de datos relacionales (20):

- Estructura de almacenamiento de tablas y registros: define cómo se almacenan físicamente las tablas y los registros. Esto incluye la disposición de filas y columnas en el almacenamiento, el uso de espacios de tabla y particiones, y la forma en que se organizan los registros en páginas o bloques de datos.

- Índices: son estructuras clave en bases de datos relacionales que ayudan a acelerar la recuperación de datos. Se pueden crear índices en una o más columnas de una tabla para permitir búsquedas más rápidas. Los tipos de índices comunes incluyen índices de árbol B e índices hash.

- Restricciones de clave e integridad: incluidas claves primarias y externas, restricciones únicas y otras restricciones que garantizan la integridad de los datos. El modelo físico detalla cómo se implementan y mantienen estas restricciones a nivel de almacenamiento. (21) (22)

- Métodos de optimización y acceso a consultas: Implica el uso de planes y algoritmos de ejecución de consultas para acceder a los datos de manera eficiente. Esto incluye seleccionar índices apropiados y optimizar uniones.

- Estrategias de particionamiento y fragmentación: una base de datos se puede dividir en partes (particiones) según ciertos criterios (como rangos, listas o hashes) para mejorar el rendimiento y la gestión. (23)

- Gestión de transacciones y control de concurrencia: detalles sobre cómo se manejan las transacciones para garantizar la coherencia e integridad de los datos, incluidos bloqueos, registros de transacciones y mecanismos de aislamiento.

- Seguridad y control de acceso: Incluye cómo se implementan los permisos y controles de acceso a nivel físico y cómo se protegen los datos del acceso no autorizado. (24)

- Estrategias de copia de seguridad y recuperación: información detallada sobre cómo realizar copias de seguridad y recuperación de datos para evitar la pérdida de datos y fallas del sistema.

El diseño del modelo físico es específico de cada DBMS relacional y puede variar significativamente según el producto (por ejemplo, Oracle, MySQL, SQL Server,

etc.). La eficiencia de este diseño es fundamental para el rendimiento general del sistema de base de datos, especialmente en bases de datos de grandes empresas.

4.2.1. SGBD seleccionado

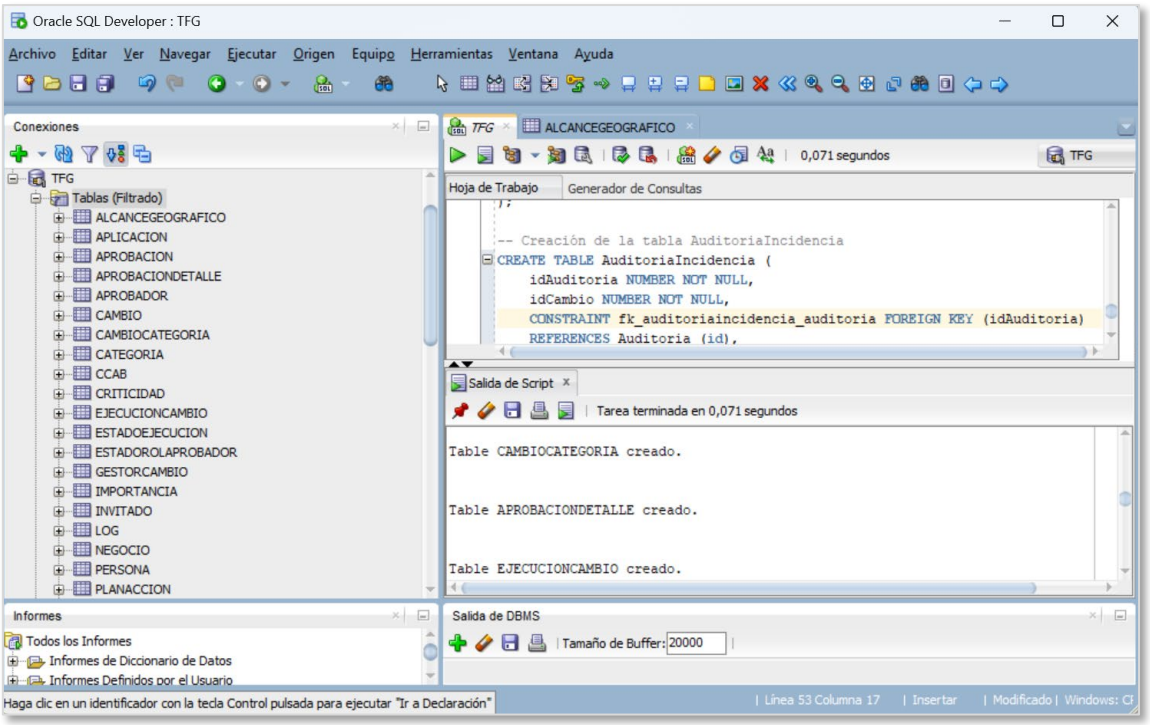
Para el desarrollo de este proyecto se ha seleccionado el sistema de gestión de bases de datos (SGBD) Oracle. Este sistema está diseñado para ser escalable, lo que lo convierte en una opción ideal para proyectos de cualquier tamaño, se pueden agregar o eliminar recursos fácilmente para satisfacer demandas variables. El rendimiento de la base de datos es otro factor a tener en cuenta, a medida que crecen los datos, el rendimiento de la base de datos puede verse afectado, Oracle está diseñado para manejar grandes cantidades de datos sin sacrificar el rendimiento. (25)

Este sistema proporciona una amplia gama de funciones de seguridad para administrar cuentas de usuario, autenticación, privilegios, seguridad de aplicaciones, cifrado, seguridad de red y más. Las medidas de confidencialidad están diseñadas para evitar intentos de acceso no autorizados a información confidencial. La disponibilidad es otro factor crítico cuando se trata de administrar bases de datos, Oracle proporciona Data Guard, que se puede utilizar con tecnología tradicional de respaldo, restauración y clúster para proporcionar un alto nivel de protección y disponibilidad de datos (26).

Oracle DBMS proporciona una amplia gama de servicios en soporte de productos, incluidos controles de estado, parches y asesores de mantenimiento (27). Además, Oracle tiene una comunidad grande y diversa de usuarios y desarrolladores que colaboran para compartir conocimientos y abordar problemas de seguridad. Es sencillo encontrar y descargar versiones de mantenimiento, *fixpacks* o arreglos rápidos con My Oracle Support (28).

En conclusión, Oracle ofrece varias ventajas sobre otros sistemas de gestión de bases de datos, incluida escalabilidad, rendimiento, seguridad, disponibilidad, compatibilidad e integración. Con amplios servicios de soporte y mantenimiento, además de rentabilidad, Oracle es la opción ideal para proyectos de cualquier tamaño. Al seleccionar Oracle los datos estarán seguros, disponibles y administrados de manera eficiente.

Se adjunta imagen de la generación de las tablas en sqldeveloper:

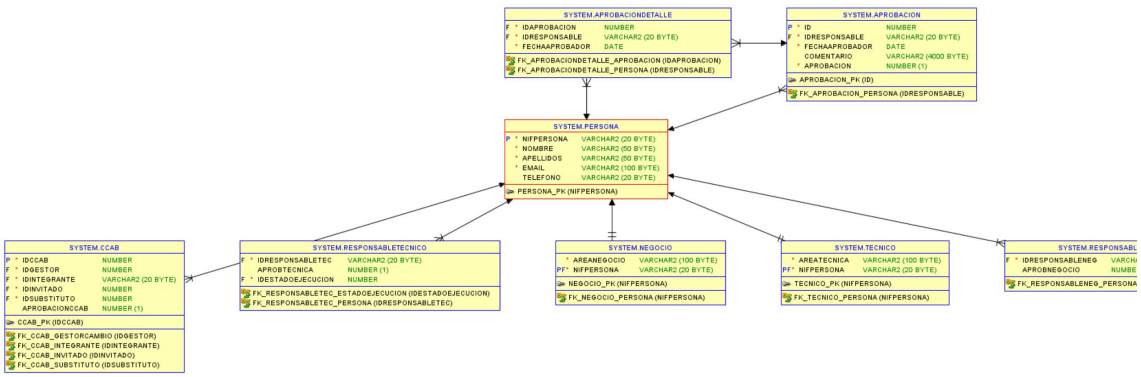


4.2.2. Tablas Control de Cambios

Persona

Esta tabla contiene información personal de los individuos relacionados con los proyectos, como desarrolladores, gestores de proyectos o partes interesadas. Incluye datos de identificación y contacto.

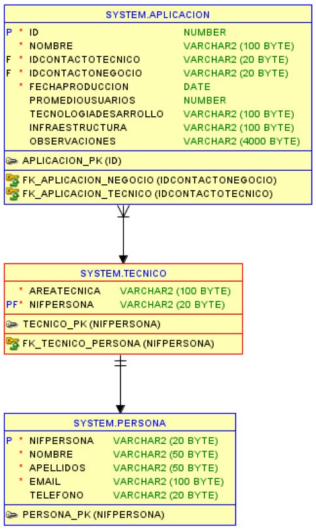
Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	nifPersona	VARCHAR2(20)	No	Identificador único de la persona
	nombre	VARCHAR2(50)	No	Nombre de la persona
	apellidos	VARCHAR2(50)	No	Apellidos de la persona
	email	VARCHAR2(100)	No	Correo electrónico de la persona
	telefono	VARCHAR2(20)	Sí	Teléfono de contacto de la persona



Tecnico

Representa a los especialistas técnicos involucrados en el proyecto, almacenando su área técnica específica y asociándolos con sus datos personales.

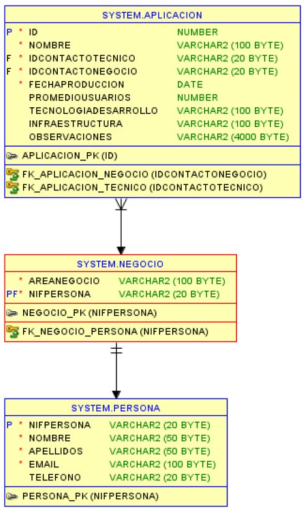
Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	nifNegocio	VARCHAR2(20)	No	Identificador único del negocio
FK	nifPersona	VARCHAR2(20)	No	Clave foránea a la tabla Persona
	areaNegocio	VARCHAR2(100)	No	Área de negocio del negocio
	esResponsAreaNeg	BOOLEAN	No	Indica si es responsable de área de negocio
	esGestorCambio	BOOLEAN	No	Indica si es gestor de cambio



Negocio

Similar a la tabla Técnico, pero enfocado en personal relacionado con el aspecto empresarial del proyecto, como analistas de negocios o gestores de producto.

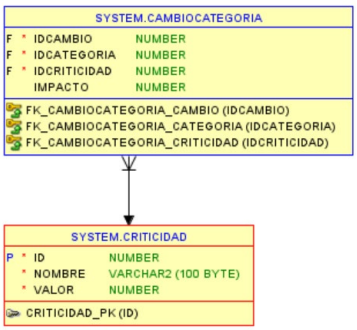
Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	nifNegocio	VARCHAR2(20)	No	Identificador único del negocio
FK	nifPersona	VARCHAR2(20)	No	Clave foránea a la tabla Persona
	areaNegocio	VARCHAR2(100)	No	Área de negocio del negocio
	esResponsAreaNeg	BOOLEAN	No	Indica si es responsable de área de negocio
	esGestorCambio	BOOLEAN	No	Indica si es gestor de cambio



Criticidad

Registra los niveles de criticidad que pueden asignarse a los cambios, lo cual ayuda a priorizar y evaluar el impacto de los cambios propuestos.

Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	id	NUMBER(5)	No	Identificador único de la criticidad
	nombre	VARCHAR2(100)	No	Nombre de la criticidad
	valor	NUMBER(1)	No	Valor de la criticidad

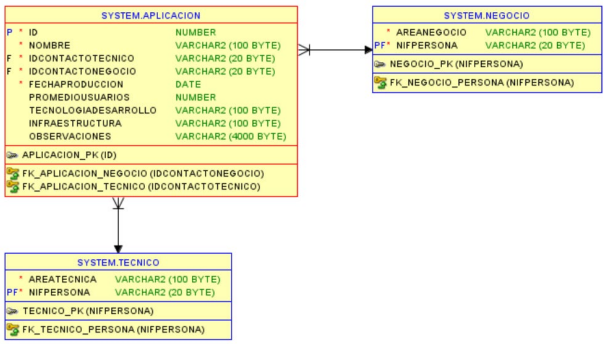


Aplicación

Esta tabla se refiere a las aplicaciones de software que están siendo desarrolladas o mantenidas, con información detallada sobre cada una.

Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	id	NUMBER(5)	No	Identificador único de la aplicación
FK	idContactoTecnico	VARCHAR2(20)	No	Clave foránea a la tabla Tecnico
FK	idContactoNegocio	VARCHAR2(20)	No	Clave foránea a la tabla Negocio
	nombre	VARCHAR2(100)	No	Nombre de la aplicación
	fechaProduccion	DATE	No	Fecha de producción de la aplicación

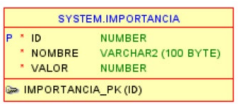
	promedioUsuarios	NUMBER(5)	No	Promedio de usuarios de la aplicación
	tecnologiaDesarrollo	VARCHAR2(100)	Sí	Tecnología de desarrollo de la aplicación
	infraestructura	VARCHAR2(100)	Sí	Infraestructura de la aplicación
	observaciones	VARCHAR2(4000)	Sí	Observaciones sobre la aplicación
	areaTecnica	VARCHAR2(100)	No	Área técnica de la aplicación
	areaNegocio	VARCHAR2(100)	No	Área de negocio de la aplicación
FK	idCriticidad	NUMBER(5)	No	Clave foránea a la tabla Criticidad



Importancia

Similar a la tabla de Criticidad, pero usada para clasificar los cambios desde una perspectiva de importancia empresarial o técnica.

Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	id	NUMBER(5)	No	Identificador único de la importancia
	nombre	VARCHAR2(100)	No	Nombre de la importancia
	valor	NUMBER(1)	No	Valor de la importancia

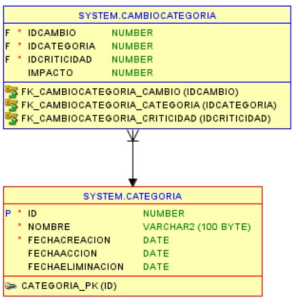


Categoria

Agrupa los cambios en categorías, lo cual será útil para la organización, filtrado y reportes de cambios.

Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	id	NUMBER(5)	No	Identificador único de la categoría
	nombre	VARCHAR2(100)	No	Nombre de la categoría
	fechaCreacion	DATE	No	Fecha de creación de la categoría

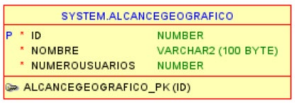
	fechaAprobAñadir	DATE	No	Fecha de aprobación de añadir categoría
	fechaEliminacion	DATE	Sí	Fecha de eliminación de la categoría
FK	idImportancia	NUMBER(5)	No	Clave foránea a la tabla Importancia



AlcanceGeografico

Se emplea para determinar el alcance de los cambios en términos de la ubicación geográfica de los usuarios afectados o de los sistemas.

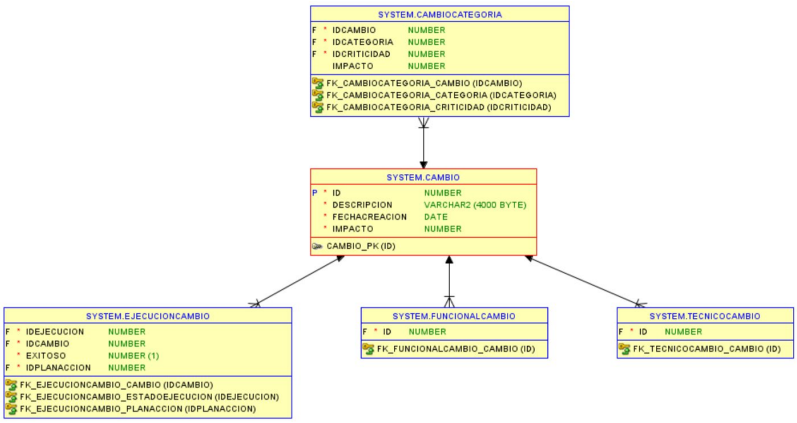
Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	id	NUMBER(5)	No	Identificador único del alcance geográfico
	nombre	VARCHAR2(100)	No	Nombre del alcance geográfico
	numeroUsuarios	NUMBER(10)	No	Número de usuarios del alcance geográfico



Cambio

Es el núcleo del sistema, donde se registran todos los cambios propuestos, su descripción, impacto y otra información relevante.

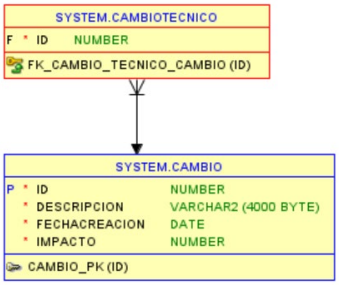
Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	id	NUMBER(5)	No	Identificador único del cambio
	descripcion	VARCHAR2(4000)	No	Descripción del cambio
	fechaCreacion	DATE	No	Fecha de creación del cambio
FK	idAlcanceGeo	NUMBER(5)	No	Clave foránea al alcance geográfico
	impacto	NUMBER(5)	No	Impacto del cambio
FK	idAplicacion	NUMBER(5)	No	Clave foránea a la tabla Aplicación
FK	idCategoria	NUMBER(5)	No	Clave foránea a la tabla Categoría



CambioTecnico

Esta tabla está diseñada para catalogar los cambios técnicos dentro de un proyecto informático. Los cambios técnicos generalmente se refieren a modificaciones en el hardware, infraestructura de red, configuraciones del sistema operativo, o ajustes de bajo nivel que no afectan directamente las funcionalidades que el usuario final experimenta.

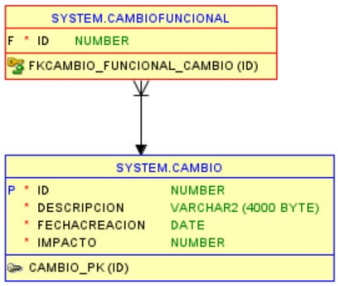
Clave	Nombre	Tipo de Dato	Nullable	Descripción
FK	idCambio	NUMBER(5)	No	Clave foránea al cambio



CambioFuncional

Se centra en los cambios funcionales. Un cambio funcional afecta a las características y comportamientos del software desde la perspectiva del usuario final. Esto podría incluir cambios en la interfaz de usuario, la adición de nuevas características, o la modificación de la lógica empresarial.

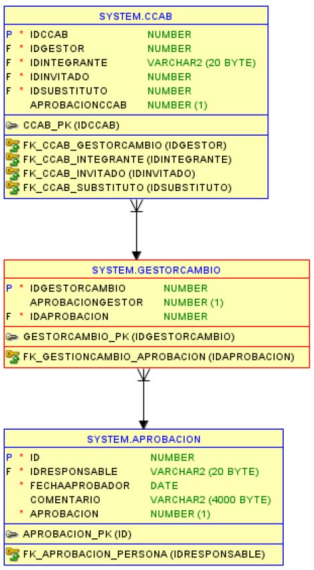
Clave	Nombre	Tipo de Dato	Nullable	Descripción
FK	idCambio	NUMBER(5)	No	Clave foránea al cambio



GestorCambio

Identifica a los individuos responsables de la gestión de los cambios, asegurando que los cambios se manejen de manera ordenada y efectiva.

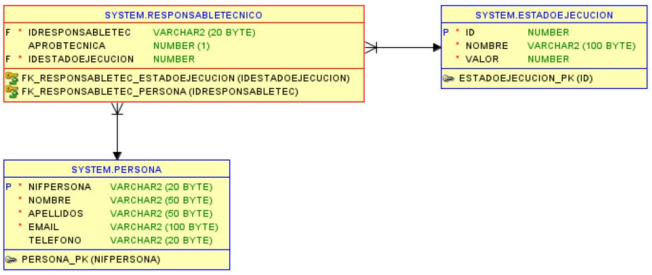
Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	nifGestorCambio	VARCHAR2(20)	No	Identificador único del gestor de cambio
FK	nifPersona	VARCHAR2(20)	No	Clave foránea a la tabla Persona



ResponsableTecnico

Esta tabla contiene información sobre los individuos responsables de los aspectos técnicos de los cambios.

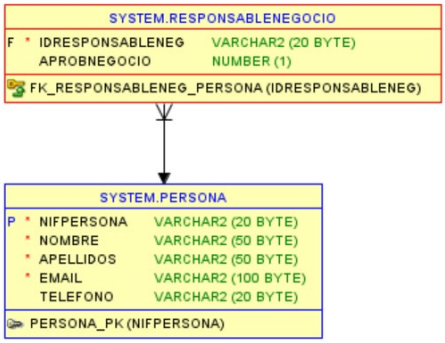
Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	nifResponsableTec	VARCHAR2(20)	No	Identificador único del responsable técnico
FK	nifTecnico	VARCHAR2(20)	No	Clave foránea a la tabla Técnico
FK	idAppResponsableTec	NUMBER(5)	No	Clave foránea a la tabla Aplicación



ResponsableNegocio

Esta tabla contiene información sobre los individuos responsables de los aspectos de negocio de los cambios.

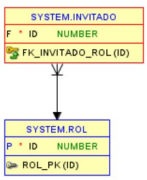
Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	nifResponsableNeg	VARCHAR2(20)	No	Identificador único del responsable de negocio
FK	nifNegocio	VARCHAR2(20)	No	Clave foránea a la tabla Negocio
FK	idAppResponsableNeg	NUMBER(5)	No	Clave foránea a la tabla Aplicación

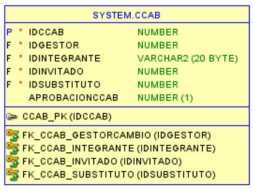


InvitadoGCAB

Relaciona a las personas con los roles que pueden ser invitados a participar en un proceso de revisión o aprobación de cambios.

Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	nifInvitado	VARCHAR2(20)	No	Identificador único del invitado GCAB
FK	nifPersona	VARCHAR2(20)	No	Clave foránea a la tabla Persona
FK	idCambio	NUMBER(5)	No	Clave foránea al cambio

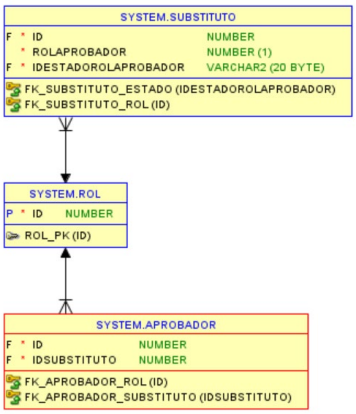




AprobadorGCAB

Contiene datos de los responsables de la aprobación de los cambios. Esto es esencial para la gestión de control de cambios y la toma de decisiones.

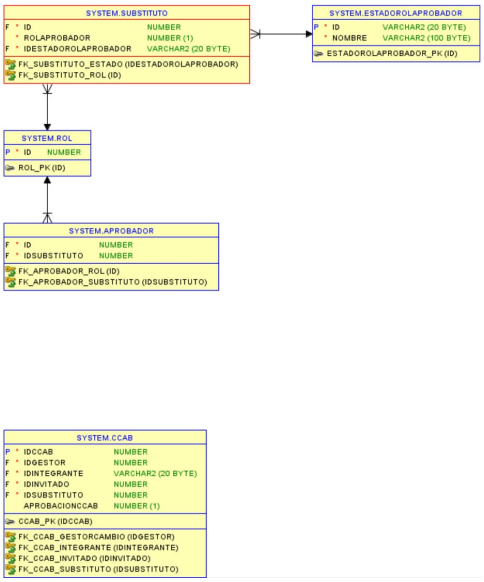
Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	nifAprobador	VARCHAR2(20)	No	Identificador único del aprobador GCAB
FK	idCambio	NUMBER(5)	No	Clave foránea al cambio
FK	AprobadorGCAB	VARCHAR2(20)	No	Clave foránea a la tabla Persona



Substituto

Mantiene un registro de aquellos individuos que pueden actuar como sustitutos en el rol de aprobación, asegurando la continuidad del proceso de gestión de cambios.

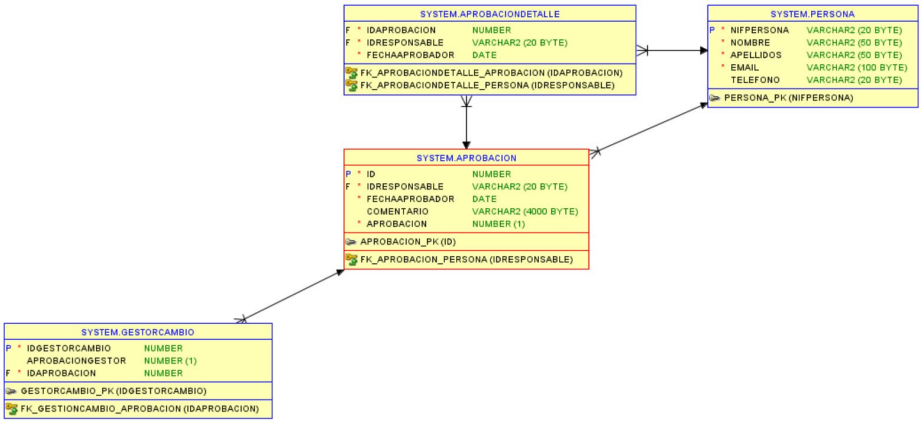
Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	nifSubstituto	VARCHAR2(20)	No	Identificador único del substituto
	cambiaAprobador	BOOLEAN	No	Indica si el substituto cambia al aprobador
FK	nifAprobador	VARCHAR2(20)	No	Clave foránea al aprobador
FK	idCambio	NUMBER(5)	No	Clave foránea al cambio
FK	nifSubstituto	VARCHAR2(20)	No	Clave foránea a la tabla Persona



Aprobacion

Captura los detalles de las aprobaciones de cambios, incluyendo quién aprobó el cambio, cuándo y cualquier comentario relevante.

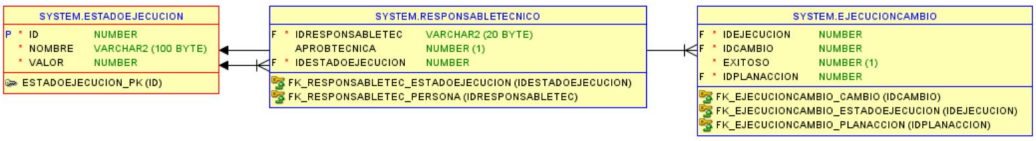
Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	id	NUMBER(5)	No	Identificador único de la aprobación
FK	idAprobador	VARCHAR2(20)	No	Clave foránea a la tabla Aprobador
	rolAprobador	VARCHAR2(100)	No	Rol del aprobador
	fechaAprobacion	DATE	No	Fecha de aprobación
	comentario	VARCHAR2(4000)	Sí	Comentario sobre la aprobación
	aprobacion	BOOLEAN	No	Indica si la aprobación es válida
FK	idCambio	NUMBER(5)	No	Clave foránea al cambio



Ejecucion

Almacena los estados posibles para la ejecución de los cambios, como "En progreso", "Completado" o "Fallido".

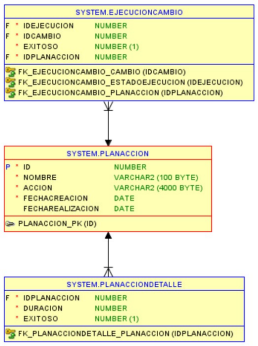
Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	id	NUMBER(5)	No	Identificador único de la ejecución
	nombre	VARCHAR2(100)	No	Nombre de la ejecución
FK	idCambio	NUMBER(5)	No	Clave foránea al cambio
	calificacion	VARCHAR2(100)	Sí	Calificación de la ejecución



PlanAccion

Describe los planes de acción establecidos en respuesta a los cambios, lo cual es esencial para la planificación y ejecución efectiva de los cambios.

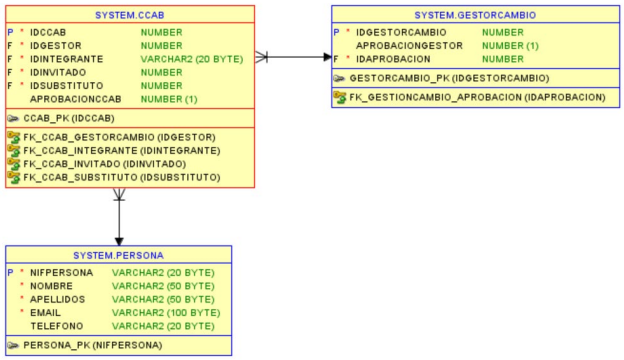
Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	id	NUMBER(5)	No	Identificador único del plan de acción
	nombre	VARCHAR2(100)	No	Nombre del plan de acción
	descripcion	VARCHAR2(4000)	No	Descripción del plan de acción
	fechaCreacion	DATE	No	Fecha de creación del plan de acción
	fechaEjecucion	DATE	Sí	Fecha de ejecución del plan de acción
	cerrado	BOOLEAN	No	Indica si el plan de acción está cerrado
FK	idEjecucion	NUMBER(5)	No	Clave foránea a la tabla Ejecución

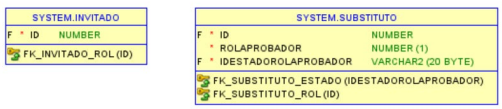


GCAB

Se refiere al Comité de Control de Cambios y Aprobaciones, que es el grupo responsable de revisar y aprobar los cambios propuestos.

Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	nifIntegranteGCAB	VARCHAR2(20)	No	Identificador único del integrante GCAB
	esInvitado	BOOLEAN	No	Indica si el integrante es invitado GCAB
FK	nifPersona	VARCHAR2(20)	No	Clave foránea a la tabla Persona
FK	idCambio	NUMBER(5)	No	Clave foránea al cambio





Auditoria

En esta tabla se recogen los datos de las auditorías realizadas en el proceso de gestión de cambios.

Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	id	INT	No	Identificador único del registro de auditoría
FK	idCambio	NUMBER(5)	No	Clave foránea del cambio asociado
	fechaApertura	DATE	No	Fecha de apertura de la auditoría
	incumplimiento	VARCHAR(500)	No	Descripción del incumplimiento
	severidad	NUMBER(1)	No	Severidad del incumplimiento
	accionesRecomendadas	VARCHAR(500)	Sí	Acciones recomendadas tras la auditoría

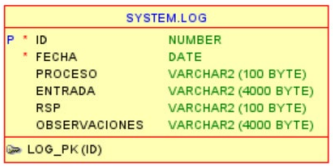
	estado	VARCHAR(50)	No	Estado de la auditoría, por defecto 'abierta'
	fechaResolucion	DATE	Sí	Fecha en que se resolvió la auditoría
	comentarios	VARCHAR(1000)	Sí	Comentarios adicionales sobre la auditoría

4.2.3. Tablas Log

Log

Registra eventos o actividades importantes dentro del sistema, proporcionando un historial detallado que puede ser utilizado para auditorías, seguimiento y análisis.

Clave	Nombre	Tipo de Dato	Nullable	Descripción
PK	id	NUMBER(5)	No	Identificador único del registro de log
	fecha	DATE	No	Fecha de registro en el log
	proceso	VARCHAR2(100)	No	Nombre del proceso registrado
	entrada	VARCHAR2(4000)	No	Descripción de la entrada en el log
	return_log	VARCHAR2(100)	No	Retorno del registro en el log

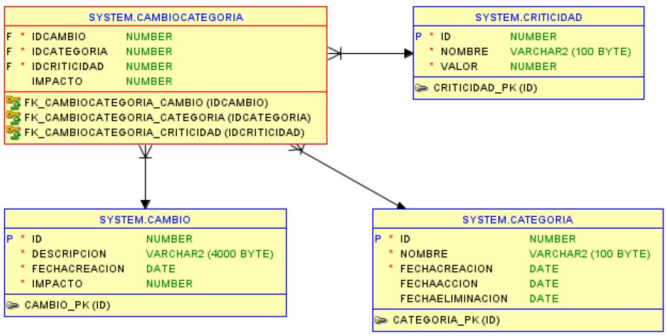


4.2.4. Tablas Repositorio Estadístico

CambioCategoria

Relaciona los cambios con categorías y niveles de criticidad, proporcionando una forma de ver rápidamente la naturaleza y urgencia de los cambios.

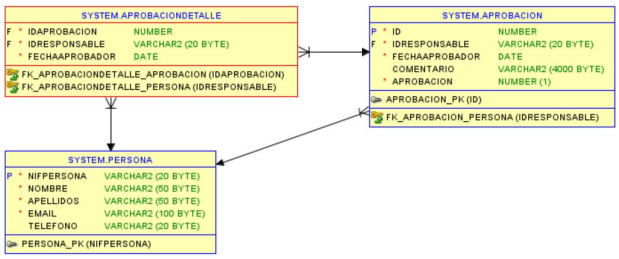
	Nombre	Tipo Dato	Nullable	Descripción
FK	idCambio	NUMBER	No	Clave foránea a la tabla Cambio
FK	idCategoria	NUMBER	No	Clave foránea a la tabla Categoria
FK	idCriticidad	NUMBER	No	Clave foránea a la tabla Criticidad
	impacto	NUMBER	Sí	Impacto asociado al cambio



AprobacionDetalle

Proporciona un seguimiento detallado de las aprobaciones, permitiendo una revisión histórica de las decisiones tomadas en el proceso de cambio.

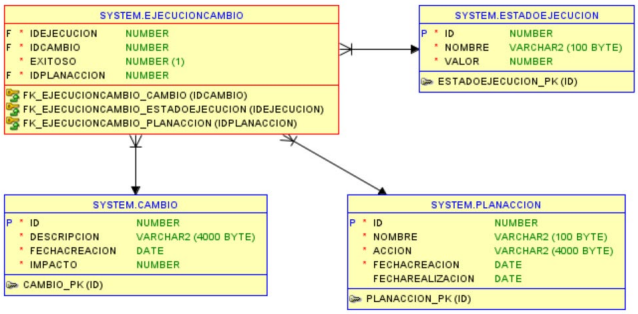
	Nombre	Tipo Dato	Nullable	Descripción
FK	idAprobacion	NUMBER	No	Clave foránea a la tabla Aprobacion
FK	idResponsable	VARCHAR2(20)	No	Clave foránea a la tabla Persona
	fechaAprobador	DATE	No	Fecha en la que se registró la aprobación



EjecucionCambio

Captura información sobre cómo se implementan los cambios, incluyendo si se completaron con éxito y si se siguieron los planes de acción.

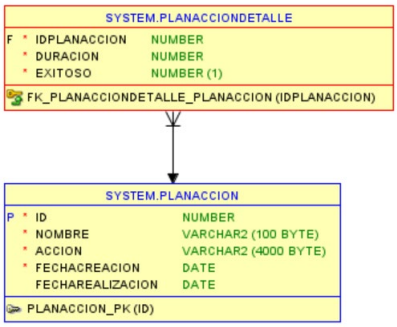
	Nombre	Tipo Dato	Nullable	Descripción
FK	idEjecucion	NUMBER	No	Clave foránea a la tabla EstadoEjecucion
FK	idCambio	NUMBER	No	Clave foránea a la tabla Cambio
	exitoso	NUMBER(1)	No	Indicador de si la ejecución fue exitosa (1) o no (0)
FK	idPlanAccion	NUMBER	No	Clave foránea a la tabla PlanAccion



PlanAccionDetalle

Ofrece detalles adicionales sobre los planes de acción, como la duración y si se lograron los resultados deseados.

	Nombre	Tipo Dato	Nullable	Descripción
FK	idPlanAccion	NUMBER	No	Clave foránea a la tabla PlanAccion
	duracion	NUMBER	No	Duración del plan de acción
	exitoso	NUMBER(1)	No	Indicador de si el plan fue exitoso (1) o no (0)



5. Implementación

La fase de implementación de una base de datos es el proceso de convertir el diseño físico de la base de datos, creado durante la fase de diseño, en una base de datos funcional dentro de un Sistema de Gestión de Bases de Datos (SGBD). Esta fase implica varios actuaciones que se indican a continuación.

Las restricciones de integridad son reglas que ayudan a mantener la precisión y coherencia de los datos en una base de datos. Se pueden utilizar para hacer cumplir las reglas comerciales y evitar la entrada de información no válida en la base de datos. Un ejemplo de restricción de integridad es la restricción de integridad referencial, que establece que el ID del cliente (CustID) en la tabla Pedido debe coincidir con un CustID válido en la tabla Cliente. La mayoría de los sistemas de gestión de bases de datos relacionales (RDBMS) admiten la definición de restricciones de integridad, incluidas claves primarias, restricciones únicas, claves externas y restricciones de verificación. También se pueden utilizar restricciones de integridad para garantizar que los datos se ingresen en un formato específico. Por ejemplo, un número de tarjeta de crédito debe

ingresarse en un formato específico para que sea válido. Al hacer cumplir estas restricciones, el sistema de base de datos puede mantener la integridad y precisión de los datos, reduciendo el riesgo de corrupción de datos.

Los procedimientos y funciones almacenados son c conjuntos precompilados de sentencias SQL que se pueden ejecutar repetidamente, lo que ahorra tiempo y reduce el riesgo de errores. Para crear una rutina almacenada, se utilizan declaraciones SQL para definir el procedimiento o función, y el servidor lo conoce. De forma predeterminada, una rutina almacenada se guarda en la base de datos en la que se crea, pero se puede mover a otra base de datos o servidor si es necesario. La seguridad del acceso de los usuarios es un aspecto crítico del desarrollo de las bases de datos.

La fase inicial de carga de datos implica el proceso de transferir datos de diversas fuentes a la base de datos. Extraer, transformar y cargar (ETL) es un proceso que se utiliza para mover datos de múltiples fuentes a un almacén de datos u otro repositorio de datos unificado (29). Antes de comenzar el proceso de carga de datos, es importante definir los requisitos de calidad de los datos para cada elemento, fuente y destino de los datos para garantizar la integridad y precisión de los datos (30). Esto garantizará que se preste la atención necesaria a la migración de datos y que el proceso esté bien definido, lo que reducirá el riesgo de corrupción y pérdida de datos.

Una vez creada la estructura de la base de datos, definidas las restricciones de integridad y completada la carga inicial de datos, los siguientes pasos en la fase de implementación son la configuración, las pruebas, la optimización, la documentación, la capacitación de los usuarios y la puesta en marcha del entorno de base de datos. La configuración del entorno de base de datos implica configurar el entorno de la base de datos, incluida la configuración de permisos de acceso, la optimización del rendimiento de la base de datos y la configuración de procesos de copia de seguridad y recuperación de la base de datos. Las pruebas son un aspecto esencial de la fase de implementación, ya que ayudan a identificar y solucionar cualquier problema antes de que la base de datos entre en funcionamiento. La optimización implica mejorar el rendimiento de la base de datos, como reducir el tiempo necesario para la recuperación de datos o mejorar los tiempos de ejecución de consultas. La documentación es otro componente importante de la fase de implementación, ya que proporciona una referencia para desarrolladores, administradores y usuarios finales. Incluye información sobre la estructura de la base de datos, diccionarios de datos, políticas y procedimientos de seguridad. La capacitación de los usuarios también es esencial para garantizar que los usuarios finales puedan utilizar eficazmente la base de datos y cumplir con las políticas de seguridad. Finalmente, la puesta en marcha de producción implica implementar la base de datos en el entorno de producción, asegurando que todos los aspectos de la fase de implementación se hayan completado con éxito.

5.1 Procedimientos ABM

Nombre del procedimiento	insert_or_update_persona
Descripción	Inserta o actualiza un registro de persona en la base de datos.
Entrada	p_nifPersona IN VARCHAR2, p_nombre IN VARCHAR2, p_apellidos IN VARCHAR2, p_email IN VARCHAR2, p_telefono IN VARCHAR2, RSP OUT VARCHAR2
Funcionamiento	1. Concatena los valores de entrada para el log. 2. Verifica si el NIF ya está almacenado en la tabla Persona. 3. Realiza verificaciones de formato y longitud del NIF, teléfono y email. 4. Inserta o actualiza en la tabla Persona según existencia del NIF. 5. Guarda el registro en el log.
Log	KO: El NIF debe tener exactamente 9 caracteres. KO: El teléfono debe tener exactamente 9 caracteres. KO: El email no tiene un formato válido. OK: Registro insertado. OK: Registro actualizado.
Salida	RSP OUT VARCHAR2, con detalles de la operación y la fecha-hora.

Nombre del procedimiento	insert_or_update_tecnico
Descripción	Inserta o actualiza un registro de técnico en la base de datos, asegurando que el técnico esté registrado como persona.
Entrada	p_nifTecnico IN VARCHAR2, p_areaTecnica IN VARCHAR2, p_esResponsAreaTec IN BOOLEAN, p_esGestorCambio IN BOOLEAN, RSP OUT VARCHAR2
Funcionamiento	1. Verifica si el NIF del técnico existe en la tabla Persona. 2. Si el NIF no existe en Persona, retorna error. 3. Si existe, verifica si ya está en la tabla Tecnico y procede a insertar o actualizar. 4. Registra la operación en el log.
Log	KO: El NIF no existe en la tabla Persona. OK: Nuevo técnico creado. OK: Datos del técnico actualizados.
Salida	RSP OUT VARCHAR2, con detalles de la operación y la fecha-hora.

Nombre del procedimiento	insert_or_update_negocio
Descripción	Inserta o actualiza un registro de negocio en la base de datos, asegurando que el negocio esté registrado como persona.

Entrada	p_nifNegocio IN VARCHAR2, p_areaNegocio IN VARCHAR2, p_esResponsAreaNeg IN BOOLEAN, p_esGestorCambio IN BOOLEAN, RSP OUT VARCHAR2
Funcionamiento	1. Verifica si el NIF del negocio existe en la tabla Persona. 2. Si el NIF no existe en Persona, retorna error. 3. Si existe, verifica si ya está en la tabla Negocio y procede a insertar o actualizar. 4. Registra la operación en el log.
Log	KO: El NIF no existe en la tabla Persona. OK: Nuevo negocio creado. OK: Datos del negocio actualizados. |
Salida	RSP OUT VARCHAR2, con detalles de la operación y la fecha-hora.

Nombre del procedimiento	**insert_or_update_criticidad**
Descripción	Inserta o actualiza un nivel de criticidad en la base de datos, con validaciones para el rango de valores.
Entrada	p_id IN NUMBER, p_nombre IN VARCHAR2, p_valor IN NUMBER, RSP OUT VARCHAR2
Funcionamiento	1. Verifica si el valor está entre 1 y 5. 2. Si el valor está fuera de rango, retorna error. 3. Si el valor es válido, inserta o actualiza en la tabla Criticidad. 4. Registra la operación en el log.
Log	KO: El valor debe estar entre 1 y 5. OK: Registro insertado o actualizado correctamente.
Salida	RSP OUT VARCHAR2, con detalles de la operación y la fecha-hora.

Nombre del procedimiento	**insert_or_update_aplicacion**
Descripción	Inserta o actualiza un registro de aplicación en la base de datos con varias validaciones.
Entrada	p_id IN NUMBER, p_nombre IN VARCHAR2, p_idContactoTecnico IN VARCHAR2, p_idContactoNegocio IN VARCHAR2, p_fechaProduccion IN DATE, p_promedioUsuarios IN NUMBER, p_tecnologiaDesarrollo IN VARCHAR2, p_infraestructura IN VARCHAR2, p_observaciones IN VARCHAR2, p_areaTecnica IN VARCHAR2, p_areaNegocio IN VARCHAR2, p_idCriticidad IN NUMBER, RSP OUT VARCHAR2

Funcionamiento	1. Determina si se necesita nuevo ID. 2. Construye cadena de valores para log. 3. Verifica existencia en Aplicacion. 4. Realiza validaciones de campos. 5. Inserta o actualiza en Aplicacion. 6. Registra en el log.
Log	KO: La fecha de producción no puede ser superior a la fecha actual. KO: idContactoTecnico e idContactoNegocio deben tener 9 caracteres. KO: idContactoTecnico o idContactoNegocio no existen. KO: idCriticidad no existe.OK: Registro insertado. OK: Registro actualizado.
Salida	RSP OUT VARCHAR2, con detalles de operación y fecha-hora.

Nombre del procedimiento	**insert_or_update_importancia**
Descripción	Inserta o actualiza un registro en la tabla Importancia.
Entrada	p_id IN NUMBER, p_nombre IN VARCHAR2, p_valor IN NUMBER, RSP OUT VARCHAR2
Funcionamiento	1. Verifica existencia en Importancia. 2. Valida rango del valor (1 a 3). 3. Inserta o actualiza en Importancia. 4. Registra en el log.
Log	KO: El valor debe estar entre 1 y 3. OK: Registro insertado o actualizado.
Salida	RSP OUT VARCHAR2, con detalles de operación y fecha-hora.

Nombre del procedimiento	**insert_or_update_categoria**
Descripción	Inserta o actualiza un registro en la tabla Categoria.
Entrada	p_id IN NUMBER, p_nombre IN VARCHAR2, p_fechaCreacion IN DATE, p_fechaAprobAnadir IN DATE, p_fechaEliminacion IN DATE, p_idImportancia IN NUMBER, RSP OUT VARCHAR2
Funcionamiento	1. Verifica existencia en Categoria. 2. Valida orden y fechas futuras. 3. Verifica existencia de idImportancia en Importancia. 4. Inserta o actualiza en Categoria. 5. Registra en el log.
Log	KO: Fechas no pueden ser futuras. KO: Problemas con el orden de fechas. KO: idImportancia no existe. OK: Registro insertado o actualizado.
Salida	RSP OUT VARCHAR2, con detalles de operación y fecha-hora.

Nombre del procedimiento	insert_or_update_alcancegeografico
Descripción	Inserta o actualiza un registro en la tabla AlcanceGeografico.
Entrada	p_id IN NUMBER, p_nombre IN VARCHAR2, p_numeroUsuarios IN NUMBER, RSP OUT VARCHAR2
Funcionamiento	1. Verifica existencia en AlcanceGeografico. 2. Valida que el número de usuarios sea positivo. 3. Inserta o actualiza en AlcanceGeografico. 4. Registra en el log.
Log	KO: El número de usuarios no puede ser negativo. OK: Registro insertado o actualizado.
Salida	RSP OUT VARCHAR2, con detalles de operación y fecha-hora.

Nombre del procedimiento	insert_or_update_alcancegeografico
Descripción	Inserta o actualiza un registro en la tabla AlcanceGeografico.
Entrada	p_id IN NUMBER, p_nombre IN VARCHAR2, p_numeroUsuarios IN NUMBER, RSP OUT VARCHAR2
Funcionamiento	1. Verifica existencia en AlcanceGeografico. 2. Valida que el número de usuarios sea positivo. 3. Inserta o actualiza en AlcanceGeografico. 4. Registra en el log.
Log	KO: El número de usuarios no puede ser negativo. OK: Registro insertado o actualizado.
Salida	RSP OUT VARCHAR2, con detalles de operación y fecha-hora.

Nombre del procedimiento	insert_or_update_cambio
Descripción	Inserta o actualiza un registro en la tabla Cambio.
Entrada	p_id IN NUMBER, p_descripcion IN VARCHAR2, p_fechaCreacion IN DATE, p_idAlcanceGeo IN NUMBER, p_idAplicacion IN NUMBER, p_idCategoria IN NUMBER, RSP OUT VARCHAR2
Funcionamiento	1. Asigna un nuevo ID si es necesario. 2. Verifica existencia en Cambio. 3. Valida la fecha de creación y la existencia de IDs en otras tablas. 4. Calcula el impacto. 5. Inserta o actualiza en Cambio. 6. Registra en el log.
Log	KO: Fecha de creación futura KO: IDs no existentes. OK: Registro insertado o actualizado.
Salida	RSP OUT VARCHAR2, con detalles de operación y fecha-hora.

Nombre del procedimiento	insert_or_update_cambiotecnico
Descripción	Inserta un nuevo registro o valida la existencia en la tabla CambioTecnico.
Entrada	p_idCambio IN NUMBER, RSP OUT VARCHAR2
Funcionamiento	1. Verifica existencia en Cambio. 2. Verifica existencia en CambioTecnico. 3. Inserta en CambioTecnico si es necesario. 4. Registra en el log.
Log	KO: ID de Cambio no existe. OK: Nuevo registro creado o ya existente.
Salida	RSP OUT VARCHAR2, con detalles de operación y fecha-hora.

Nombre del procedimiento	insert_or_update_cambiofuncional
Descripción	Inserta un nuevo registro o valida la existencia en la tabla CambioFuncional.
Entrada	p_idCambio IN NUMBER, RSP OUT VARCHAR2
Funcionamiento	1. Verifica existencia en Cambio. 2. Verifica existencia en CambioFuncional. 3. Inserta en CambioFuncional si es necesario. 4. Registra en el log.
Log	KO: ID de Cambio no existe. OK: Nuevo registro creado o ya existente.
Salida	RSP OUT VARCHAR2, con detalles de operación y fecha-hora.

Nombre del procedimiento	insert_invitado
Descripción	Inserta un nuevo registro o valida la existencia en la tabla Invitado.
Entrada	p_nifInvitado IN VARCHAR2, p_idCambio IN NUMBER, RSP OUT VARCHAR2
Funcionamiento	1. Verifica existencia de NIF en Persona y ID en Cambio. 2. Verifica existencia en Invitado. 3. Inserta en Invitado si es necesario. 4. Registra en el log.
Log	KO: NIF de invitado no existe. KO: ID de Cambio no existe. OK: Nuevo invitado creado o ya existente.
Salida	RSP OUT VARCHAR2, con detalles de operación y fecha-hora.

Nombre del procedimiento	insert_aprobadorGCAB
Descripción	Inserta un nuevo aprobador GCAB en la base de datos.
Entrada	p_nifAprobador IN VARCHAR2, p_idCambio IN NUMBER, RSP OUT VARCHAR2

Funcionamiento	1. Verifica existencia del NIF del aprobador en Persona. 2. Verifica existencia del ID de Cambio en Cambio. 3. Inserta en AprobadorGCAB si es necesario. 4. Registra en el log.
Log	KO: NIF del aprobador no existe. KO: ID de Cambio no existe. OK: Nuevo aprobador GCAB creado.
Salida	RSP OUT VARCHAR2, con detalles de operación y fecha-hora.

Nombre del procedimiento	**insert_or_update_substituto**
Descripción	Inserta o actualiza un registro de substituto en la base de datos.
Entrada	p_nifSubstituto IN VARCHAR2, p_cambiaAprobador IN BOOLEAN, p_nifAprobador IN VARCHAR2, p_idCambio IN NUMBER, RSP OUT VARCHAR2
Funcionamiento	1. Verifica existencia del substituto y relacionados en sus respectivas tablas. 2. Inserta o actualiza en Substituto. 3. Registra en el log.
Log	KO: NIF del substituto no existe. KO: NIF aprobador no existe. KO: ID de Cambio no existe. OK: Registro insertado o actualizado.
Salida	RSP OUT VARCHAR2, con detalles de operación y fecha-hora.

Nombre del procedimiento	**insert_or_update_aprobacion**
Descripción	Inserta o actualiza una aprobación en la base de datos.
Entrada	p_id NUMBER, p_idAprobador VARCHAR2, p_fechaAprobacion DATE, p_comentario VARCHAR2, p_aprobacion BOOLEAN, RSP OUT VARCHAR2
Funcionamiento	1. Verifica existencia del aprobador y del cambio en sus respectivas tablas. 2. Determina el rol del aprobador. 3. Inserta o actualiza en Aprobacion. 4. Registra en el log.
Log	KO: NIF aprobador no existe. KO: ID de Cambio no existe. OK: Registro insertado o actualizado.
Salida	RSP OUT VARCHAR2, con detalles de operación y fecha-hora.

Nombre del procedimiento	**insert_or_update_ejecucion**
Descripción	Inserta o actualiza un registro en la tabla Ejecucion.

Entrada	p_id NUMBER, p_nombre VARCHAR2, p_idCambio NUMBER, p_calificacion VARCHAR2, RSP OUT VARCHAR2
Funcionamiento	1. Verifica existencia del ID de Cambio en Cambio.. 2. Valida la calificación con opciones permitidas. 3. Inserta o actualiza en Ejecucion. 4. Registra en el log. |
Log	KO: Calificación no válida. KO: ID de Cambio no existe. OK: Registro insertado o actualizado. |
Salida	RSP OUT VARCHAR2, con detalles de operación y fecha-hora.

5.2 Triggers indicadores

Para dar respuesta a las consultas siguientes:

- En el momento de ejecutar la consulta, número de cambios registrados en el sistema que están en proceso de aprobación.
- En el último año, número de cambios aprobados que su ejecución no ha sido correcta.
- Teniendo en cuenta toda la información existente en la BD, responsable técnico con más cambios ejecutados de manera correcta.
- Dado una año concreto, porcentaje de acciones definidas para ejecuciones no correctas cerradas en el tiempo inicialmente definido en el plan de acción correspondiente.
- En el año en curso, número total de cambios aprobados en la GCAB
- En un año concreto, número de cambios que no fueron aprobados y, consecuentemente, no se ejecutaron.
- Dada una región geográfica concreta, porcentaje de cambios que su ejecución se replanificó.
- En los últimos 6 meses, número total de incumplimientos detectados durante las auditorías realizadas.
- En el momento de ejecutar la consulta, número total de planes de acción sobre ejecuciones no correctas que están abiertos.
- En un momento dado, responsable técnico con más cambios en curso.
- Teniendo en cuenta todos los datos de la BD, persona concreta con rol de aprobador que ha estado substituida más veces por indisponibilidad no planificada.

Se han creado 11 *triggers* y sus correspondientes contadores en forma de tabla auxiliar, que irán actualizando todos estos datos estadísticos en otra tabla auxiliar llamada Indicadores. El procedimiento que sigue cada *trigger* es el siguiente:

Trigger 'actualiza_cambios_pendientes_trigger':
En el momento de la consulta hay que verificar que en la tabla Aprobacion para un determinado idCambio se encuentra el idAprobador con rol 'RT' y el idAprobador con rol 'RN' y el idAprobador con rol 'GC', y además verificar que se encuentran en la tabla Aprobacion los idAprobador de todos los nifAprobador de la tabla AprobadorGCAB. Se contabilizará como cambio no aprobado aquellos que no cumplan todas y cada una de las condiciones anteriores.

Trigger 'actualiza_ejecucion_incorrecta_trigger':
Realiza la comprobación anterior para cada idCambio, consulta la tabla Ejecucion y contabiliza aquellos idCambio cuyo atributo calificacion es 'Ejecución con incidencias' o 'Ejecución replanificada'.

Trigger 'actualiza_cambios_correctos_trigger':
Para los idCambios de la tabla Ejecución busca los idCambio con la calificacion = 'Ejecución correcta', y consulta este idCambio en la tabla Aprobacion para devolver el idAprobador con rol ='RT' que figure más veces.

Trigger 'actualiza_acciones_ejecucion_trigger':
Para un año concreto consulta en la tabla PlanAccion, contabiliza aquellos id que tienen su atributo fechaCreacion en ese año y el atributo cerrado = TRUE. Calcula el % en relación al número total de ejecuciones.

Trigger 'actualiza_cambios_aprobados_gcab_trigger':
Para el año en curso comprueba en la tabla Aprobacion aquellos idCambio que tienen su atributo fechaCreacion en el año en curso y todos los idAprobador que para ese mismo idCambio forman parte de la tabla AprobadorGCAB.

Trigger 'actualiza_cambios_no_aprobados_trigger':
En el momento de la consulta verifica que en la tabla Aprobacion para un determinado idCambio se encuentra el idAprobador con rol 'RT' y el idAprobador con rol 'RN' y el idAprobador con rol 'GC' y también que se encuentran en la tabla Aprobacion todos los idAprobador de todos los nifAprobador de la tabla AprobadorGCAB. Se contabilizará como cambio no aprobado aquellos que no cumplan todas y cada una de las condiciones anteriores.

Trigger 'actualiza_porcentaje_cambios_replanificados_trigger':
Para un id indicado de la tabla de alcanceGeografico, hay que buscar las aplicaciones asociadas en tabla Aplicaciones (id) y a su vez los cambios (id) asociados a esta aplicación con fechaCreacion dentro de esos 6 meses. Se calcula el % en función de los cambios totales registrados para esos 6 meses (fechaCreacion dentro de esos 6 meses) en la tabla de Cambios.

Trigger 'actualiza_incumplimientos_auditorias_trigger':
En la tabla Auditorias hay se contabilizan los id cuyo atributo incumplimiento no es nulo.

Trigger 'actualiza_planes_accion_abiertos_trigger':
En el momento de ejecutar la consulta, se contabilizan el número total de planes de acción sobre ejecuciones no correctas que están abiertos, en este caso en la tabla PlanAccion hay consultar los id con el atributo cerrado =FALSE.

Trigger 'actualiza_indicador_10_trigger':
En la tabla Tecnico se consulta los nifTecnico con atributo esResponsAreaTec=TRUE; de estos nifTecnico se compara su atributo areaTecnica con el atributo areaTecnica de la Tabla Aplicacion, si coincide, en la tabla Cambios se ontabiliza el número de cambios (id) que tiene cada

aplicación. Una vez obtenido el id de la aplicación se devuelve el nifTecnico con atributo esResponsAreaTec=TRUE y que coincidan los atributos areaTecnica de aplicacion y Tecnico y con mayor número de cambios para esa aplicación. Los datos deben ser anteriores a la fecha introducida por tanto se contabilizaran solo los cambios cuya fechaCreacion sea igual o inferior a la dada.

Trigger 'actualiza_indicador_aprobador_trigger':
En un momento dado, hay que buscar en la tabla Substituto el idAprobador con el mayor número de atributoscambiaAprobador en TRUE.

6. Pruebas

6.1 Pruebas procedimientos ABM

Esta tabla resume el resultado de los casos de prueba ejecutados con el procedimiento insert_or_update_persona, asegurándose de que los casos correctos e incorrectos se manejan como se espera y que los resultados se registran adecuadamente en el sistema.

Nombre de Procedimiento	Descripción Prueba	Resultado
insert_or_update_persona	Casos Correctos: Realizamos 10 inserciones correctas con distintos NIFs. Se verificaron: la correcta inserción en la tabla **Persona**, la creación de registros de log con resultado 'OK' en la tabla **Log**, que el contador **people_counter** refleje 10 y la inserción correcta de los NIFs en tablas adicionales. Ejemplo de inserción: **insert_or_update_persona('123456789', 'Juan', 'Perez', 'juan@example.com', '987654321', temp_log);**	Satisfactorio
insert_or_update_persona	Casos Incorrectos: Intentamos inserciones con datos inválidos como NIFs y teléfonos de longitud incorrecta, emails mal formateados y datos duplicados. Se esperaba que no se insertaran en la tabla **Persona**, que el log registrara 'KO' indicando el motivo del error, y que el contador **people_counter** no se incrementara. Ejemplo de inserción errónea: **insert_or_update_persona('12345678', 'Juan', 'Perez', 'juan@example.com', '987654321', temp_log);**	Satisfactorio

Esta tabla detalla cómo el procedimiento insert_or_update_tecnico maneja y valida la entrada de datos, asegurando la coherencia y la integridad de la base de datos al tratar con la inserción y actualización de registros técnicos. Cada prueba refleja la lógica del procedimiento para manejar correctamente los escenarios válidos e inválidos.

Nombre de Procedimiento	Descripción Prueba	Resultado
insert_or_update_tecnico	**Casos Correctos:** Se realizaron inserciones y actualizaciones de técnicos con NIFs válidos que existen en la tabla **Persona**. Se verificaron inserciones en la tabla **Tecnico**, la actualización de atributos como **areaTecnica**, **esResponsAreaTec**, y **esGestorCambio**, y la creación de registros de log con el resultado 'OK'. Ejemplos incluyen la asignación de nuevas áreas técnicas y la actualización de responsabilidades y gestión de cambios.	Satisfactorio
insert_or_update_tecnico	**Casos Incorrectos:** Se intentaron inserciones con NIFs que no existen en **Persona**, campos de área técnica vacíos, valores booleanos incorrectos, y actualizaciones de técnicos no existentes. En todos los casos, el sistema no realizó la inserción o actualización y generó un registro de log con el resultado 'KO', especificando el error correspondiente. Esto asegura que solo los técnicos válidos se creen o actualicen, y que cualquier intento de insertar datos inválidos se registre adecuadamente para su revisión.	Satisfactorio

Esta tabla refleja que los datos en la tabla Negocio solo se añaden o actualizan si el NIF correspondiente existe en la tabla Persona, y que cualquier error se registre adecuadamente en el sistema.

Nombre de Procedimiento	Descripción Prueba	Resultado
insert_or_update_negocio	**Casos Correctos**: Se verificaron inserciones y actualizaciones en la tabla **Negocio** para NIFs existentes en **Persona**. Incluye pruebas de inserción de nuevos perfiles de negocio con detalles de área y roles, así como la actualización de perfiles de negocio existentes. Ejemplo: La ejecución de **insert_or_update_negocio('111122223N', 'Ventas', TRUE, FALSE, temp_log);** insertaría un usuario de negocio en el área de 'Ventas' con responsabilidad de área, pero sin gestión de cambio. Estas pruebas aseguran que el procedimiento maneja correctamente tanto la creación como la actualización de registros, manteniendo la integridad de los datos.	Satisfactorio

insert_or_update_negocio	**Casos Incorrectos**: Incluye intentos de inserción con NIFs no registrados en **Persona**, falta de especificación de áreas de negocio, o uso incorrecto de valores booleanos. Por ejemplo, **insert_or_update_negocio('123456789X', 'Compras', TRUE, FALSE, temp_log);** intentaría crear un perfil de negocio con un NIF no existente, resultando en un registro 'KO' en el log. Estos casos prueban la robustez del sistema en la gestión de errores y la prevención de inserciones inválidas.	Satisfactorio

Esta tabla detalla cómo el procedimiento insert_or_update_criticidad gestiona diferentes escenarios, asegurando que los datos introducidos sean válidos y estén dentro del rango esperado, y que cualquier error se registre adecuadamente.

Nombre de Procedimiento	Descripción Prueba	Resultado
insert_or_update_criticidad	**Casos Correctos**: Ejecutamos inserciones y actualizaciones en **Criticidad** con valores válidos. Comprobamos que cada inserción actualiza la tabla con el nombre y valor de criticidad adecuados. Verificamos que el log registre 'OK' para cada transacción exitosa. Además, revisamos el número total de registros en **Criticidad** para asegurarnos de que coincida con el número de operaciones realizadas. Ejemplo: **insert_or_update_criticidad(1, 'Baja', 2, temp_log)**; crea una criticidad 'Baja' con valor 2. Comprobamos que se añade correctamente y que el log muestra 'OK'. Verificamos que el número de registros en Criticidad es correcto.	Satisfactorio
insert_or_update_criticidad	**Casos Incorrectos**: Intentamos inserciones y actualizaciones con valores de criticidad fuera del rango permitido o con IDs que no existen. Aseguramos que estos intentos no cambien los datos en **Criticidad**. Verificamos que el log registre 'KO' para cada operación fallida, y revisamos que el número total de registros en **Criticidad** se mantenga constante, indicando que no se añadieron datos erróneos. **Ejemplo: insert_or_update_criticidad(6, 'Incorrecta', 0, temp_log);** intenta crear una criticidad con valor 0, resultando en un fallo y registro 'KO' en el log. Aseguramos que no se añaden datos incorrectos a Criticidad.	Satisfactorio

Esta tabla refleja cómo el procedimiento insert_or_update_aplicacion maneja tanto las inserciones y actualizaciones correctas como las incorrectas, asegurando que los datos se añaden solo cuando son válidos y que cualquier error se registra adecuadamente en el sistema.

Nombre de Procedimiento	Descripción Prueba y Ejemplo de Inserción	Resultado
insert_or_update_aplicacion	**Casos Correctos**: Realizamos inserciones y actualizaciones en **Aplicacion** con datos válidos. Ejemplos: **insert_or_update_aplicacion(NULL, 'Aplicación 1', '123456789', '987654321', TO_DATE('2023-01-14', 'YYYY-MM-DD'), 1000, 'Tecnología 1', 'Infraestructura 1', 'Observaciones 1', 'Área Técnica 1', 'Área Negocio 1', 1, temp_log);** crea una nueva aplicación. Verificamos que se añaden correctamente a la base de datos y que el log muestra 'OK'. Revisamos que los IDs de contacto técnico y negocio existen y que la criticidad está dentro del rango permitido.	Satisfactorio
insert_or_update_aplicacion	**Casos Incorrectos**: Intentamos inserciones con datos inválidos, como fechas de producción futuras, IDs de contacto técnico o negocio incorrectos, o criticidades inexistentes. Ejemplo: **insert_or_update_aplicacion(NULL, 'Aplicación Futura 3', '555555555', '666666666', TO_DATE('2025-01-01', 'YYYY-MM-DD'), 500, 'Tecnología 3', 'Infraestructura 3', 'Observaciones 3', 'Área Técnica 3', 'Área Negocio 3', 3, temp_log);** intenta crear una aplicación con una fecha de producción futura, lo que resulta en un fallo y registro 'KO' en el log. Comprobamos que estos intentos no cambian la tabla **Aplicacion** y que el log refleja adecuadamente los errores.	Satisfactorio

Esta tabla ilustra cómo el sistema maneja tanto los casos correctos como los incorrectos, asegurando la integridad de los datos en la tabla Importancia y una adecuada respuesta del sistema ante entradas erróneas.

Nombre de Procedimiento	Descripción Prueba y Ejemplo de Inserción	Resultado
insert_or_update_importancia	**Casos Correctos**: Realizamos inserciones y actualizaciones válidas en **Importancia**. Ejemplo: **insert_or_update_importancia(1, 'Importancia Alta', 3, temp_log);** crea una nueva importancia con un valor de 3. Verificamos que se añade correctamente y que el log muestra 'OK'.	Satisfactorio
insert_or_update_importancia	**Casos Incorrectos**: Intentamos inserciones con valores fuera del rango permitido (1-3) o IDs no válidos. Ejemplo: **insert_or_update_importancia(4, 'Importancia Invalida', 5, temp_log);** intenta crear una importancia con un valor fuera de rango, resultando en un fallo y registro 'KO' en el log.	Satisfactorio

Esta tabla muestra cómo el procedimiento insert_or_update_categoria gestiona tanto los casos válidos como los inválidos, asegurando que solo se añadan o actualicen datos válidos en la base de datos y que cualquier error se registre adecuadamente en el sistema.

Nombre de Procedimiento	Descripción Prueba y Ejemplo de Inserción	Resultado
insert_or_update_categoria	**Casos Correctos**: Inserciones y actualizaciones en **Categoria** con fechas y valores válidos. Ejemplo: **insert_or_update_categoria(1, 'Categoría A', TO_DATE('01-01-2023', 'DD-MM-YYYY'), TO_DATE('02-01-2023', 'DD-MM-YYYY'), NULL, 1, temp_log);** crea una nueva categoría cumpliendo todas las condiciones de fecha y referencia de importancia.	Satisfactorio
insert_or_update_categoria	**Casos Incorrectos**: Intentos de inserción con fechas futuras, incorrectas, o referencias de importancia inexistentes. Ejemplo: **insert_or_update_categoria(7, 'Categoría G', TO_DATE('08-01-2023', 'DD-MM-YYYY'), TO_DATE('07-01-2023', 'DD-MM-YYYY'), NULL, 1, temp_log);** intenta crear una categoría con una fecha de creación futura, resultando en un fallo y registro 'KO' en el log.	Satisfactorio

Esta tabla proporciona una visión completa y detallada de cómo el procedimiento insert_or_update_alcancegeografico gestiona las inserciones y actualizaciones, garantizando la validez de los datos y la coherencia del sistema.

Nombre de Procedimiento	Descripción Prueba y Ejemplo de Inserción	Resultado
insert_or_update_**alcancegeografico**	**Casos Correctos**: Ejecutamos inserciones y actualizaciones en **AlcanceGeografico** con datos válidos. Ejemplos incluyen: **insert_or_update_alcancegeografico(1, 'Alcance 1', 500, temp_log);** para la creación de un nuevo alcance geográfico o **insert_or_update_alcancegeografico(1, 'Nuevo Nombre', 800, temp_log);** para la actualización de un registro existente. Aseguramos que cada inserción o actualización se refleje correctamente en la base de datos y que el log indique 'OK'. Revisamos que el número de usuarios sea positivo y coherente con los criterios del sistema.	Satisfactorio

insert_or_update_alcancegeografico	**Casos Incorrectos**: Intentamos inserciones con números de usuarios negativos o IDs no válidos. Ejemplo: **insert_or_update_alcancegeografico(6, 'Alcance Negativo', -100, temp_log);** trata de establecer un número negativo de usuarios, lo cual no es permitido, y el sistema lo rechaza, reflejando 'KO' en el log. Estos casos prueban la capacidad del sistema para validar los datos y manejar errores, asegurando que solo se añadan datos coherentes y lógicos.	Satisfactorio

Esta tabla muestra cómo el procedimiento insert_or_update_cambio maneja inserciones y actualizaciones, asegurando que los datos sean coherentes y válidos, y que cualquier error se registre adecuadamente en el sistema.

Procedimiento	Descripción Prueba y Ejemplo de Inserción	Resultado
insert_or_update_cambio	**Casos Correctos**: Se realizan inserciones y actualizaciones en **Cambio**. Ejemplos incluyen la creación de un nuevo cambio con datos válidos, como **insert_or_update_cambio(NULL, 'Cambio 1', TO_DATE('2024-01-15', 'YYYY-MM-DD'), 1, 1, 1, temp_log);** y la actualización de cambios existentes. Se verifica que las fechas sean coherentes y que los IDs de alcance geográfico, aplicación y categoría existan.	Satisfactorio
insert_or_update_cambio	**Casos Incorrectos**: Se prueban inserciones con datos inválidos como fechas futuras o IDs inexistentes. Por ejemplo, **insert_or_update_cambio(NULL, 'Cambio Futuro', TO_DATE('2025-01-15', 'YYYY-MM-DD'), 1, 1, 1, temp_log);** intenta crear un cambio con una fecha de creación futura, lo cual es rechazado por el sistema. Estos casos validan la integridad de los datos y las reglas de negocio.	Satisfactorio

Esta tabla muestra cómo el sistema gestiona las inserciones en CambioTecnico, asegurando que los datos sean coherentes y válidos, y que cualquier error se registre adecuadamente en el sistema.

Nombre de Procedimiento	Descripción Prueba y Ejemplo de Inserción	Resultado
insert_or_update_cambiotecnico	**Casos Correctos**: Realizamos inserciones en **CambioTecnico** basadas en IDs válidos de **Cambio**. Ejemplo: **insert_or_update_cambiotecnico(1, temp_log);** verifica la existencia de un ID de cambio y, si no existe en **CambioTecnico**, lo inserta. Se verifica que la inserción sea correcta y el log muestre 'OK'.	Satisfactorio

insert_or_update_cambiotecnico	**Casos Incorrectos**: Intentamos inserciones con IDs de cambio que no existen en la tabla **Cambio**. Ejemplo: **insert_or_update_cambiotecnico(1000, temp_log);** intenta insertar un registro con un ID de cambio inexistente, lo que resulta en un fallo y registro 'KO' en el log. Esto verifica la integridad referencial entre las tablas.	Satisfactorio

Esta tabla muestra cómo el sistema maneja las inserciones y actualizaciones en CambioFuncional, asegurando que los datos sean coherentes y válidos, y que cualquier error se registre adecuadamente en el sistema.

Nombre de Procedimiento	Descripción Prueba y Ejemplo de Inserción	Resultado
insert_or_update_cambiofuncional	**Casos Correctos**: Inserciones y actualizaciones en **CambioFuncional** basadas en los IDs válidos de **Cambio**. Ejemplos: **insert_or_update_cambiofuncional(1, temp_log);** para crear un nuevo registro o actualizar uno existente. Validamos que los IDs de cambio existan y, de ser necesario, se inserta un nuevo registro o se confirma que ya existe.	Satisfactorio
insert_or_update_cambiofuncional	**Casos Incorrectos**: Intentamos inserciones con IDs de cambio que no existen o son inválidos. Ejemplo: **insert_or_update_cambiofuncional(11, temp_log);** intenta insertar un registro con un ID de cambio inexistente, resultando en un error y registro 'KO' en el log. Estos casos prueban la integridad de datos y las reglas de negocio al no permitir referencias a cambios inexistentes.	Satisfactorio

Esta tabla muestra cómo el sistema gestiona las inserciones en Invitado, asegurando la validez de los datos y la coherencia del sistema.

Nombre de Procedimiento	Descripción Prueba y Ejemplo de Inserción	Resultado
insert_invitado	**Casos Correctos**: Inserciones en **Invitado** basadas en NIFs válidos de **Persona** y IDs válidos de **Cambio**. Ejemplos incluyen **insert_invitado('123456789', 1, temp_log);** para agregar un nuevo invitado o confirmar que ya existe. Se verifica la existencia del NIF en **Persona** y el ID de cambio en **Cambio**, y se registra la operación correctamente en el log.	Satisfactorio

insert_invitado	**Casos Incorrectos**: Se prueban inserciones con NIFs que no existen en **Persona** o IDs de cambio inexistentes en **Cambio**. Ejemplo: **insert_invitado('999999999', 1, temp_log);** intenta agregar un invitado con un NIF inexistente, lo cual resulta en un error y registro 'KO' en el log. Estos casos validan la integridad de datos y las reglas de negocio.	Satisfactorio

Esta tabla muestra cómo el sistema gestiona las inserciones en AprobadorGCAB, asegurando la validez de los datos y la coherencia del sistema.

Nombre de Procedimiento	Descripción Prueba y Ejemplo de Inserción	Resultado
insert_aprobadorGCAB	**Casos Correctos**: Se realiza la inserción de aprobadores en **AprobadorGCAB** con NIFs válidos de **Persona** y IDs válidos de **Cambio**. Por ejemplo, **insert_aprobadorGCAB('123456789', 1, temp_log);** añade un nuevo aprobador si el NIF y el ID de cambio existen. Se verifica la inserción correcta y el log muestra 'OK'.	Satisfactorio
insert_aprobadorGCAB	**Casos Incorrectos**: Se prueban inserciones con NIFs que no existen en **Persona** o IDs de cambio inexistentes en **Cambio**. Por ejemplo, **insert_aprobadorGCAB('999999999', 1, temp_log);** intenta agregar un aprobador con un NIF inexistente, lo que resulta en un fallo y registro 'KO' en el log. Estos casos verifican la integridad de datos y las reglas de negocio.	Satisfactorio

Esta tabla muestra cómo el procedimiento insert_or_update_substituto gestiona las inserciones y actualizaciones en Substituto, asegurando la validez de los datos y la coherencia del sistema.

Nombre de Procedimiento	Descripción Prueba y Ejemplo de Inserción	Resultado
insert_or_update_substituto	**Casos Correctos**: Se realizan inserciones y actualizaciones en **Substituto** verificando que el NIF del substituto exista en **Persona**, el ID de cambio en **Cambio**, y el NIF del aprobador en **Persona**. Por ejemplo, **insert_or_update_substituto('123456789', TRUE, '987654321', 1, temp_log);** agrega o actualiza un substituto, confirmando la existencia de los NIFs y el ID de cambio.	Satisfactorio
insert_or_update_substituto	**Casos Incorrectos**: Se prueban inserciones con NIFs que no existen en **Persona**, IDs de cambio inexistentes en **Cambio**, o con NIFs de substituto y aprobador iguales. Ejemplo: **insert_or_update_substituto('999999999', TRUE, '987654321', 1, temp_log);** intenta agregar un substituto con un NIF inexistente en **Persona**, resultando en un fallo y registro 'KO' en el log.	Satisfactorio

Esta tabla muestra cómo el sistema gestiona las inserciones y actualizaciones en Aprobacion, asegurando la validez de los datos y la coherencia del sistema. Se verifica la existencia de los NIFs y el ID de cambio, se asignan roles apropiados a los aprobadores, y se registra correctamente cada operación en el log. Además, se manejan situaciones en las que los datos no cumplen con los requisitos establecidos, como NIFs inexistente o formatos de datos incorrectos, garantizando así la integridad y la precisión de la información en la base de datos.

Nombre de Procedimiento	Descripción Prueba y Ejemplo de Inserción	Resultado
insert_or_update_aprobacion	**Casos Correctos**: Inserciones y actualizaciones en **Aprobacion** se basan en la existencia del NIF del aprobador en **Persona** y del ID de cambio en **Cambio**. Ejemplos como **insert_or_update_aprobacion(NULL, '123456789', TO_DATE('2024-01-15', 'YYYY-MM-DD'), 'RT', TRUE, temp_log);** crean o actualizan registros, asegurando la validez de los NIFs y el ID de cambio, y asignando roles apropiados.	Satisfactorio
insert_or_update_aprobacion	**Casos Incorrectos**: Se prueban inserciones con NIFs que no existen en **Persona**, IDs de cambio inexistentes en **Cambio**, o datos en formatos incorrectos. Por ejemplo, **insert_or_update_aprobacion(NULL, '999999999', TO_DATE('2024-01-15', 'YYYY-MM-DD'), 'RT', TRUE, temp_log);** intenta agregar una aprobación con un NIF inexistente, lo que resulta en un fallo y registro 'KO' en el log.	Satisfactorio

Esta tabla muestra cómo el sistema gestiona las inserciones y actualizaciones en Ejecucion, asegurando la validez de los datos y la coherencia del sistema. Las calificaciones se limitan a opciones predefinidas para mantener la consistencia de los datos, y se verifica la existencia del ID de cambio para asegurar la coherencia de la información en la base de datos.

Nombre de Procedimiento	Descripción Prueba y Ejemplo de Inserción	Resultado
insert_or_update_ejecucion	**Casos Correctos**: Inserciones y actualizaciones en **Ejecucion** se basan en la existencia del ID de cambio en **Cambio** y una calificación válida. Ejemplos incluyen **insert_or_update_ejecucion(NULL, 'Ejecución 1', 1, 'Ejecución correcta', temp_log);**, donde se añade o actualiza una ejecución, verificando la validez del ID de cambio y la calificación. Se registra 'OK' en el log si los datos son correctos.	Satisfactorio

insert_or_update_ejecucion	**Casos Incorrectos**: Se prueban inserciones con calificaciones no válidas o IDs de cambio inexistentes en **Cambio**. Por ejemplo, **insert_or_update_ejecucion(NULL, 'Ejecución 11', 11, 'Calificación Incorrecta', temp_log);** intenta agregar una ejecución con una calificación no permitida, resultando en un error y registro 'KO' en el log. Estos casos validan la integridad de datos y las reglas de negocio.	Satisfactorio

Esta tabla resume el proceso y los casos de prueba para el procedimiento almacenado insert_or_update_planaccion.

Nombre de Procedimiento	Descripción Prueba y Ejemplo de Inserción	Resultado
insert_or_update_planaccion	Casos Correctos: Se prueban inserciones y actualizaciones en 'PlanAccion' que cumplen con las validaciones de fechas y calificaciones de ejecución existentes. Ejemplos incluyen **insert_or_update_planaccion(NULL, 'Plan Acción 1', ...)** donde se añade o actualiza un plan de acción verificando la validez de la fecha y la calificación de ejecución relacionada. Se registra 'OK' en el log si los datos son correctos.	Satisfactorio
insert_or_update_planaccion	Casos Incorrectos: Se prueban inserciones con fechas de creación futuras o fechas de ejecución anteriores a la creación, o con calificaciones de ejecuciones no permitidas según las reglas del negocio. Por ejemplo, **insert_or_update_planaccion(NULL, 'Plan Acción Incorrecto 1', ...)** intenta agregar un plan con una fecha de creación futura, resultando en un error y registro 'KO' en el log. Estos casos verifican la integridad de datos y las reglas de negocio.	Satisfactorio

Esta tabla resume los escenarios de prueba para el procedimiento almacenado insert_or_update_auditoria, asegurando que la lógica de negocio se cumpla y que los datos insertados o actualizados en la tabla Auditoria sean consistentes y válidos.

Nombre de Procedimiento	Descripción Prueba y Ejemplo de Inserción	Resultado
insert_or_update_auditoria	Casos Correctos: Se validan las inserciones y actualizaciones en la tabla 'Auditoria' que cumplen con los criterios de existencia de **idCambio**, las fechas de apertura y resolución, la severidad dentro del rango permitido, y los estados válidos. Por ejemplo, **insert_or_update_auditoria(NULL, 1, ...)** añade una nueva auditoría con datos válidos. Se registra 'OK' en el log si los datos son correctos.	Satisfactorio

insert_or_update_auditoria	Casos Incorrectos: Se prueban inserciones y actualizaciones con **idCambio** inexistente, fechas no válidas (apertura futura o resolución anterior a la apertura), severidad fuera del rango, o estados no válidos. Por ejemplo, **insert_or_update_auditoria(NULL, 100, ...)** intenta agregar una auditoría con **idCambio** que no existe, resultando en un error y registro 'KO' en el log. Estos casos verifican la integridad de datos y las reglas de negocio.	Satisfactorio

6.2 Pruebas triggers indicadores:

Para los diferentes *triggers* que actualizan las consultas se han realizado las siguientes pruebas:

Trigger 'actualiza_cambios_pendientes_trigger':
Escenario: Se han insertado o actualizado registros en la tabla Aprobacion, lo cual puede incluir cambios en el estado de aprobación de diferentes roles o ajustes en las entradas relacionadas con el grupo AprobadorGCAB, afectando así el conteo de cambios pendientes.
Verificación: Se ha verificado si el Indicador 1 en la tabla Indicadores refleja correctamente la cantidad actualizada de cambios pendientes, examinando el valor actualizado en dicha tabla.
Resultados: Satisfactorios.

Trigger 'actualiza_ejecucion_incorrecta_trigger':
Escenario: Se han insertado o actualizado registros en la tabla Ejecucion dentro del año actual, específicamente marcando ejecuciones como 'Ejecución con incidencias' o 'Ejecución replanificada'.
Verificación: Se ha verificado si el Indicador 2 en la tabla Indicadores refleja correctamente el incremento en la cantidad de ejecuciones incorrectas para el año actual, revisando el valor actualizado en la tabla.
Resultados: Satisfactorios.

Trigger 'actualiza_cambios_correctos_trigger':
Escenario: Se han insertado o actualizado registros en la tabla Ejecucion que han sido calificados como 'Ejecución correcta', lo cual implica registrar y contabilizar cambios correctos realizados por técnicos responsables.
Verificación: Se ha verificado si el Indicador 3 en la tabla Indicadores refleja correctamente el técnico con el mayor número de cambios correctos, revisando el valor actualizado en la tabla para asegurarse de que refleja al técnico con el máximo número de cambios correctos registrados.
Resultados: Satisfactorios.

Trigger 'actualiza_acciones_ejecucion_trigger':
Escenario: Se han insertado o actualizado registros en la tabla PlanAccion dentro del año actual, incluyendo tanto la creación de nuevas acciones como la actualización del estado de acciones existentes (especialmente marcando algunas como cerradas).

Verificación: Se ha verificado si el Indicador 4 en la tabla Indicadores refleja correctamente el porcentaje de acciones cerradas en relación con el total de acciones del año actual, revisando el valor actualizado en la tabla.
Resultados: Satisfactorios.

Trigger 'actualiza_cambios_aprobados_gcab_trigger':
Escenario: Se han insertado o actualizado registros en la tabla Aprobacion que reflejan la aprobación de cambios por todos los miembros del GCAB para el año actual.
Verificación: Se ha verificado si el Indicador 5 en la tabla Indicadores muestra adecuadamente el número total de cambios aprobados por el GCAB, revisando el valor actualizado en la tabla ContadorCambiosAprobadosGCAB y luego en Indicadores.
Resultados: Satisfactorios.

Trigger 'actualiza_cambios_no_aprobados_trigger':
Escenario: Se han insertado registros en la tabla Aprobacion que reflejan la no aprobación de cambios por parte de uno o varios roles de aprobadores (RT, RN, GC) o por el grupo GCAB, dentro del año actual.
Verificación: Se ha verificado si el Indicador 6 en la tabla Indicadores refleja de manera precisa el número total de cambios no aprobados, revisando el contador actualizado en la tabla ContadorCambiosNoAprobados y luego el valor correspondiente en Indicadores.
Resultados: Satisfactorios.

Trigger 'actualiza_porcentaje_cambios_replanificados_trigger':
Escenario: Se han insertado o actualizado registros en la tabla Cambio, incluyendo aquellos marcados como replanificados, lo que impacta en el cálculo del porcentaje de cambios replanificados para un alcance geográfico específico.
Verificación: Se ha verificado si el valor en la tabla del Indicador 7 refleja correctamente el porcentaje actualizado de cambios replanificados, revisando los valores en la tabla PorcentajeCambiosReplanificados y luego comparando con el valor en Indicadores.
Resultados: Satisfactorios.

Trigger 'actualiza_incumplimientos_auditorias_trigger':
Escenario: Se han insertado o actualizado registros en la tabla Auditoria, incluyendo aquellos que reflejan incumplimientos, particularmente dentro del periodo de los últimos 6 meses.
Verificación: Se ha verificado si el Indicador 8 en la tabla Indicadores refleja correctamente el número total de incumplimientos en auditorías durante los últimos 6 meses, revisando el contador actualizado en la tabla ContadorIncumplimientosAuditorias y luego comparando con el valor en Indicadores.
Resultados: Satisfactorios.

Trigger 'actualiza_planes_accion_abiertos_trigger':
Escenario: Se han insertado o actualizado registros en la tabla Ejecucion, lo que puede influir en el estado de los planes de acción, específicamente en la cantidad de planes de acción que permanecen abiertos.

Verificación: Se ha verificado si el Indicador 9 en la tabla Indicadores muestra correctamente el número total de planes de acción abiertos para el año actual, revisando el contador actualizado en la tabla ContadorPlanesAccionAbiertos y luego comparando con el valor en Indicadores.
Resultados: Satisfactorios.

Trigger 'actualiza_indicador_10_trigger':
Escenario: Se han insertado o actualizado registros en la tabla Cambios, lo que podría afectar la cantidad de cambios asociados a cada técnico responsable en su área técnica correspondiente.
Verificación: Se ha verificado si el Indicador 10 en la tabla Indicadores refleja correctamente el técnico responsable con el mayor número de cambios en curso, revisando el valor actualizado en la tabla Indicadores para asegurarse de que corresponde al técnico con el mayor número de cambios registrados hasta la fecha especificada.
Resultados: Satisfactorios.

Trigger 'actualiza_indicador_aprobador_trigger':
Escenario: Se han insertado o actualizado registros en la tabla Substituto que indican cambios en el aprobador, afectando el conteo de cuántas veces cada aprobador ha sido sustituido.
Verificación: Se ha verificado si el Indicador 11 en la tabla Indicadores refleja correctamente el ID del aprobador que ha sido sustituido más veces, revisando el valor actualizado en la tabla Indicadores para asegurarse de que corresponde al aprobador con el mayor número de sustituciones registradas.
Resultados: Satisfactorios.

7. Conclusiones

En este Trabajo de Fin de Grado, se ha diseñado e implementado una base de datos para una Aplicación de Control de Cambios, considerando las complejidades y desafíos que tal tarea implica. A lo largo de este proyecto, se ha seguido una metodología estructurada que abarcó desde la planificación detallada hasta la implementación efectiva, destacando la importancia de un diseño lógico y físico cohesivo.

La ejecución del proyecto reveló la crítica necesidad de una estimación realista del tiempo y recursos, especialmente al enfrentar imprevistos personales y profesionales que impactaron la agenda. No obstante, se lograron los objetivos primordiales, demostrando la viabilidad del sistema desarrollado y su potencial para adaptarse y expandirse con futuras funcionalidades.

La experiencia adquirida en el proceso ha sido invaluable, especialmente en áreas menos exploradas en la práctica profesional diaria, como el diseño conceptual y el uso de disparadores en bases de datos. Las lecciones aprendidas durante este proyecto no solo son aplicables al ámbito académico sino que también proporcionan una sólida base para proyectos futuros en entornos profesionales reales.

Para continuar el desarrollo del proyecto, se recomienda la creación de interfaces de usuario que faciliten la interacción con la base de datos, y la expansión de la capacidad analítica del sistema para ofrecer evaluaciones internas más completas dentro de las organizaciones que lo adopten. Además, se subraya la importancia de abordar aspectos de seguridad, rendimiento y escalabilidad en etapas subsecuentes.

En conclusión, el proyecto no solo cumplió con los requisitos establecidos sino que también abrió caminos para el crecimiento y la mejora continua del sistema de control de cambios, enfatizando el valor del diseño y la planificación en el desarrollo de software.

8. Glosario

Definición de los términos y acrónimos más relevantes utilizados dentro de la Memoria:

ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad): conjunto de propiedades que garantizan transacciones confiables en sistemas de bases de datos.

***Backup* y Recuperación de Datos**: procesos para crear copias de seguridad de los datos y restaurarlos en caso de pérdida o daño.

Clave Foránea: un campo en una tabla que enlaza a la clave primaria de otra tabla, estableciendo una relación entre ellas.

Clave Primaria: un campo o conjunto de campos que identifica de manera única cada registro en una tabla de base de datos.

***Data Mining* (Minería de Datos)**: proceso de descubrir patrones y conocimientos a partir de grandes conjuntos de datos.

Data Warehousing: técnicas y herramientas para el almacenamiento y análisis de grandes cantidades de datos.

Esquema de Base de Datos: estructura que define la organización de los datos en una base de datos.

ETL (Extract, Transform, Load): proceso utilizado en bases de datos para extraer datos de fuentes heterogéneas, transformarlos en un formato adecuado y cargarlos en un destino.

Índice: estructura de datos utilizada para mejorar la velocidad de las operaciones de recuperación en una base de datos.

Modelo Entidad-Relación (ER): modelo conceptual para representar datos y sus relaciones, comúnmente utilizado en el diseño de bases de datos.

Normalización: proceso de diseño de una base de datos para reducir la redundancia y mejorar la integridad de los datos.

Procedimiento Almacenado: un conjunto de instrucciones SQL que se almacenan en la base de datos y pueden ser ejecutadas por el SGBD.

SQL (Structured Query Language): lenguaje estándar para gestionar y manipular bases de datos relacionales.

SGBD (Sistema de Gestión de Bases de Datos): *software* que facilita la creación, manipulación, y gestión de bases de datos.
Integridad de Datos: la calidad y precisión de los datos almacenados en una base de datos.

Transacción: conjunto de operaciones en una base de datos que se ejecutan como una sola unidad.

Trigger **(Disparador)**: SQL que se ejecuta automáticamente en respuesta a ciertos eventos en una base de datos.

8. Bibliografía

1. Malik, Priyanka. ITIL Change Management Process: Overview, Benefits, Limitations (2023). [En línea] 10 de marzo de 2023. [Citado el: 12 de octubre de 2023.] https://whatfix.com/blog/itil-change-management/.
2. Belandria, Joisbel. Guía para implementar la gestión de cambios. [En línea] Tech-Blog, 10 de febrero de 2023. [Citado el: 11 de octubre de 2023.] https://www.gb-advisors.com/es/guia-para-implementar-la-gestion-de-cambios-itil/.
3. Bartech. Implemente cambios en archivos físicos de datos del IBM i en minutos. [En línea] Bartech, 2023. [Citado el: 10 de octubre de 2023.] https://www.bartech.es/productos/mdrapid.
4. EasyVista. Gestión de cambios y versiones. [En línea] EasyVista, 2023. [Citado el: 10 de octubre de 2023.] https://www.easyvista.com/es/solucion/gestion-versiones-y-cambios.
5. Asana. Plantilla para órdenes de cambio: Cómo modificar el alcance de un proyecto (incluye ejemplos). [En línea] Asana, 2023. [Citado el: 10 de octubre de 2023.] https://asana.com/es/resources/change-order-template.
6. Pratt, Mary. Low-code and no-code development platforms. [En línea] TechTarget, 1 de marzo de 2023. [Citado el: 12 de octubre de 2023.] https://www.techtarget.com/searchsoftwarequality/definition/low-code-no-code-development-platform#:~:text=Low%2Dcode%2Fno%2Dcode,create%20mobile%20or%20web%20apps..
7. ONU. Objetivos de desarrollo sostenible. [En línea] ONU, 2023. [Citado el: 11 de octubre de 2023.] https://www.un.org/sustainabledevelopment/es/.
8. Wikipedia. Waterfall model. [En línea] Wikipedia, 2023. [Citado el: 10 de octubre de 2023.] https://en.wikipedia.org/wiki/Waterfall_model.
9. Casas, Jordi. *Introducción al diseño de bases de datos.* Barcelona : FUOC, 2021.
10. Oracle. Oracle Database XE Downloads. [En línea] Oracle, 2023. [Citado el: 20 de octubre de 2023.] https://www.oracle.com/es/database/technologies/xe-downloads.html.
11. —. Free Oracle Database for Everyone. [En línea] Oracle, 2023. [Citado el: 20 de octubre de 2023.] https://www.oracle.com/database/technologies/appdev/xe.html.
12. OCI. SQL Developer. [En línea] OCI, 2023. [Citado el: 20 de octubre de 2023.] https://www.oracle.com/database/sqldeveloper/.
13. KeepCoding Team. ¿Qué es Draw.io? [En línea] KeepCoding Team, 4 de enero de 2023. [Citado el: 20 de octubre de 2023.] https://keepcoding.io/blog/que-es-drawio/.
14. Visure. What are Functional Requirements: Examples, Definition, Complete Guide. [En línea] Visure, 2023. [Citado el: 30 de octubre de 2023.] https://visuresolutions.com/blog/functional-requirements/.
15. Nuclino. A Guide to Functional Requirements (with Examples). [En línea] Nuclino, 2023. [Citado el: 30 de Octubre de 2023.] https://www.nuclino.com/articles/functional-requirements.
16. Gibb, Tom. *Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage.* s.l. : Butterworth-Heinemann, 2005.

17. Rivera, Sonny. Conceptual vs logical vs physical data models. [En línea] Thoughtspot, 1 de junio de 2023. [Citado el: 25 de diciembre de 2023.] https://www.thoughtspot.com/data-trends/data-modeling/conceptual-vs-logical-vs-physical-data-models.
18. MariaDB. Database Design Phase 2: Conceptual Design. [En línea] MariaDB, 2023. [Citado el: 25 de noviembre de 2023.] https://mariadb.com/kb/en/database-design-phase-2-conceptual-design/.
19. Logical Data Model. [En línea] ScienceDirect, 2023. [Citado el: 26 de noviembre de 2023.] https://www.sciencedirect.com/topics/computer-science/logical-data-model.
20. Suszterova, Sandra. Physical Data Model vs. Logical Data Model. [En línea] Gooddata, 7 de marzo de 2023. [Citado el: 1 de diciembre de 2023.] https://www.gooddata.com/blog/physical-vs-logical-data-model/.
21. javaTpoint. Integrity Constraints. [En línea] javaTpoint, 2023. [Citado el: 1 de diciembre de 2023.] https://www.javatpoint.com/dbms-integrity-constraints.
22. IBM. Primary key, referential integrity, check, and unique constraints. [En línea] IBM, 12 de mayo de 2023. [Citado el: 1 de diciembre de 2023.] https://www.ibm.com/docs/en/db2/11.5?topic=concepts-primary-key-referential-integrity-check-unique-constraints.
23. Fragmentation: Storage distribution strategies. [En línea] IBM, 7 de septiembre de 2021. [Citado el: 1 de diciembre de 2023.] https://www.ibm.com/docs/en/informix-servers/14.10?topic=model-fragmentation-storage-distribution-strategies.
24. IBM. Mandatory Access Control. [En línea] IBM, 24 de marzo de 2023. [Citado el: 22 de diciembre de 2023.] https://www.ibm.com/docs/en/aix/7.2?topic=security-mandatory-access-control.
25. Oracle. Design for Scalability. [En línea] Oracle, 30 de agosto de 2023. [Citado el: 2 de diciembre de 2023.] https://docs.oracle.com/en/solutions/oci-best-practices/design-scalability1.html.
26. —. Oracle Database High Availability Features. [En línea] Oracle, 2023. [Citado el: 2 de diciembre de 2023.] https://docs.oracle.com/cd/B13789_01/server.101/b10726/hafeatures.htm.
27. —. Oracle Premier Support. [En línea] Oracle, 2023. [Citado el: 2 de diciembre de 2023.] https://www.oracle.com/es/support/premier/.
28. My Oracle Support Help. Patches and Updates. [En línea] Oracle, 2023. [Citado el: 2 de diciembre de 2023.] https://docs.oracle.com/cd/E25290_01/doc.60/e25224/patchesupdates.htm.
29. IBM. What is ETL? [En línea] IBM, 2023. [Citado el: 4 de diciembre de 2023.] https://www.ibm.com/topics/etl#:~:text=ETL%2C%20which%20stands%20for%20extract,warehouse%20or%20other%20target%20system..
30. Linkedln. ¿Cuáles son las mejores formas de garantizar la integridad y precisión de los datos en la arquitectura de datos? [En línea] Linkedln, 27 de septiembre de 2023. [Citado el: 4 de diciembre de 2023.] https://www.linkedin.com/advice/0/what-best-ways-ensure-data-completeness-accuracy.

9. Anexos

Se incluyen en una carpeta aparte (Fermorcar_anexos), para su correcta visualización ya que en esta memoria se han incluido con unas dimensiones reducidas, los ficheros:

- *Fermorcar_UML.jpg*
- *Fermorcar_Gantt.mpp*

También se adjunta en esta carpeta (Fermorcar_anexos) en formato pdf la presentación de diapositivas realizada para la presentación en vídeo del proyecto:

- Fermorcar_Presentacion.pdf

Por otro lado se presenta el vídeo explicativo del proyecto:

- Fermorcar_Presentacion.mp4