Universitat Oberta de Catalunya
uoc.edu

BBDD para aplicación de control de cambios

Juan Ovelar
Grado de Ingeniería Informática
Base de Datos

Jordi Ferrer Duran
Josep Cobarsí Morales

15/01/2024

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Diseño e implementación de la base de datos para una aplicación de control de cambios*
Nombre del autor:	*Juan Ovelar*
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	*TFG, Bases de Datos, Gestión de Cambios.*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados i conclusiones del trabajo.*

Este trabajo se realiza en el marco de la asignatura Trabajo Final de Grado de la UOC, teniendo un doble objetivo. Desde el punto de vista académico, pretende conseguir que el alumno ponga en práctica el conocimiento adquirido a lo largo de los años en el Grado de Ingeniería Informática.

Por otro lado, el objetivo del trabajo consiste en realizar el diseño e implementación de una base de datos para una aplicación de control de cambios. En la actualidad, el software en general está en constante evolución, por ello resulta imprescindible para las empresas desarrolladoras contar con una herramienta que les permita gestionar, de una manera centralizada y ordenada, los cambios.

En cuanto al desarrollo del proyecto, se establece la metodología “en cascada” definiendo una serie de fases que deben completarse: planificación, análisis y requisitos, diseño, implementación y pruebas. De manera transversal, transcurre una sexta fase que se lleva a cabo a lo largo de todo el proyecto: documentación.

Los resultados obtenidos de este proyecto se entregarán en forma de producto final, donde se incluirá: esta memoria, los scripts de generación de la estructura de la base de datos, scripts de inserción de datos de pruebas y el conjunto de pruebas como tal. Además de un video y presentación resumen del trabajo realizado durante el semestre.

El desarrollo del proyecto se ha llevado a cabo casi en su totalidad, a excepción de un apartado que no se ha conseguido desarrollar por fallas en la planificación y consecuente falta de tiempo.

Abstract (in English, 250 words or less):

This work is a result from the Final Degree Project of the UOC, it has a dual objective. From an academic perspective, is a proof to the student apply the knowledge acquired throughout the years in the Computer Engineering Degree.

The goal of this project is design a database for a change control application. Currently, the applications are constantly evolving and it have a very big number of change to apply. The development companies needs a tool that allows them to manage these changes in a centralized and orderly way.

Regarding the project's development, the "waterfall" methodology is established. A series of phases to be completed were defined: planning, analysis and requirements, design, implementation, and testing. A sixth phase, documentation, runs through the entire project.

The results obtained from this project will be delivered like a final product, which will include: this report, scripts for generating the database structure, scripts for inserting test data, and the set of tests. In addition, a video and summary presentation of the work done during the semester.

The project development has been finalized almost entirely, except for a section that could not be developed due to planning failures and consequent lack of time.

Índice

1. Introducción

1.1. Contexto y justificación del Trabajo

La evolución de la informática y la tecnología en los últimos años ha sido asombrosa. En este proceso de cambio constante, resulta imprescindible contar con sólidos procesos para gestionar las modificaciones que se realizan de forma periódica en las numerosas aplicaciones que utilizamos en nuestra vida cotidiana. En este sentido, como usuario sólo experimentamos las mejoras y cambios en estas aplicaciones una vez que ya han sido implementadas, probadas y están listas para su uso. Sin embargo, no vemos ni nos percatamos del extenso y meticuloso trabajo que hay detrás de cada nueva funcionalidad, de cada modificación y de cada actualización en dichas aplicaciones.

Sin entrar en detalle, cada vez que una aplicación se somete a una mejora o a una nueva versión, hay una serie de procesos y tareas que deben llevarse a cabo de manera ordenada. El objetivo de estos procesos es garantizar que, tras lanzar una nueva versión o una modificación, no se comprometa al funcionamiento normal de la aplicación, y que los usuarios puedan disfrutar de una experiencia fluida y sin interrupciones ni problemas.

En el ámbito de la informática y la tecnología, la correcta gestión del cambio se convierte en un componente esencial. En un entorno donde la evolución es constante, y las aplicaciones tecnológicas evolucionan y mejoran continuamente, es fundamental contar con sólidos procesos y prácticas que garanticen la calidad y el éxito de los procesos de desarrollo y despliegue. Un enfoque ampliamente reconocido en esta área es el conjunto de prácticas conocidas como ITIL (Information Technology Infrastructure Library), que se ha convertido en un estándar de referencia en la industria.

Si consideramos el volumen de modificaciones y cambios que se implementan dentro de una aplicación a lo largo del tiempo, se hace evidente la necesidad de contar con herramientas especializadas para ayudar a las empresas de desarrollo de software a gestionar eficazmente esta importante tarea. La gestión del cambio no solo se trata de introducir nuevas características, sino también de asegurarse de que no se interrumpan las operaciones existentes y que los usuarios puedan disfrutar de una experiencia ininterrumpida y de alta calidad.

En el contexto de este Trabajo Final de Grado y en este proyecto en particular, nos enfocaremos en el planteamiento y desarrollo de la base de datos que servirá como pilar fundamental para una aplicación específica de gestión del cambio, ya que creemos que las aplicaciones existentes en el mercado actual son muy generalistas y ninguna aborda de manera directa este aspecto. La aplicación resultante será esencial para garantizar que las empresas de desarrollo de software puedan llevar a cabo modificaciones y actualizaciones de manera eficiente, manteniendo la estabilidad y la calidad de sus aplicaciones. La base de datos que crearemos proporcionará un marco sólido para el seguimiento, la documentación y la coordinación de todos los cambios, contribuyendo también a una gestión más efectiva y a la satisfacción de los usuarios finales.

Por último, y aunque no forma parte del alcance de este TFG, cabe mencionar que esta iniciativa se traducirá en un proceso de desarrollo de software más fluido y confiable en un entorno de rápido cambio tecnológico.

1.2. Objetivos del Trabajo

Objetivos relacionados con el marco de proyecto en la gestión de cambios en aplicaciones:

- **Facilitar la creación de inventarios de cambios:** Con este objetivo buscamos simplificar y agilizar el proceso de enumerar todos los cambios planificados para una aplicación en desarrollo. La idea es proporcionar una herramienta que permita a los equipos de desarrollo llevar un registro detallado de todas las modificaciones previstas.

- **Mejorar la gestión y trazabilidad de cambios:** En este caso, el objetivo es elevar la calidad de la administración de cambios en el desarrollo de software. Se busca establecer un sistema que garantice una trazabilidad completa de cada modificación, desde la idea inicial hasta la implementación final. Esta mejora contribuye a una gestión más eficiente y transparente de los cambios.

- **Identificar importancia e impacto de cambios:** La meta es poder evaluar la importancia y el impacto de cada modificación en el producto final. Esta evaluación ayuda a priorizar los cambios y a tomar decisiones informadas en el proceso de desarrollo.

- **Optimizar la publicación de versiones:** Este objetivo se enfoca en perfeccionar el proceso de lanzamiento de nuevas versiones de una aplicación. El fin es hacer más confiable la publicación de versiones actualizadas, mejorando la experiencia del usuario y la eficiencia operativa.

- **Desarrollar el modelo de datos:** Se busca crear un modelo de datos que sirva como columna vertebral de una aplicación dedicada a la gestión de cambios en el desarrollo de software. Este modelo de datos será esencial para la estructuración y organización de la información relacionada con los cambios.

- **Realizar el diseño conceptual, lógico y físico de la Base de Datos:** Aquí, el objetivo es realizar un análisis exhaustivo del enunciado para realizar un diseño que abarque desde la concepción inicial hasta la creación de la base de datos.

- **Aplicar técnicas de DataWareHouse**: Con este objetivo se busca aplicar técnicas útiles para la gestión de grandes volúmenes de datos, permitiendo la realización eficiente de diversas consultas que den respuestas a necesidades del negocio.

Objetivos relacionados con el desarrollo del TFG:

- **Aplicar conocimientos adquiridos:** El objetivo es poner en práctica todas las habilidades y conocimientos adquiridos durante el Grado en Ingeniería en

Informática. El TFG se convierte en una oportunidad para aplicar la teoría aprendida en una situación real.

- **Analizar el enunciado y crear especificaciones:** Aquí hacemos hincapié en la capacidad de análisis. El objetivo es que sea capaz de comprender a fondo el enunciado del proyecto y extraer las especificaciones necesarias para llevar a cabo el trabajo de manera efectiva.

- **Implementar una estrategia de Gestión de Proyectos:** Además de la programación y la implementación técnica, el objetivo incluye el desarrollo de habilidades de gestión de proyectos. Esto implica la planificación, programación y ejecución efectiva del TFG, que garantice un enfoque organizado y metódico en todo el proceso de desarrollo.

1.3. Enfoque y método seguido

Para abordar este proyecto, es fundamental tener en cuenta que estamos trabajando en un contexto con plazos predefinidos que deben respetarse para cumplir con las entregas de los avances del proyecto (PECs). Dado el carácter de este Trabajo de Fin de Grado (TFG), que establece una serie de especificaciones bien definidas y una duración en concreto, el enfoque "en cascada" o Ciclo de Vida de Desarrollo de Sistemas (SDLC) se presenta como el modelo de gestión más adecuado para esta situación. A través de este enfoque, estableceremos un proceso lineal compuesto por una serie de fases dependientes entre sí, en las que cada fase debe completarse antes de que podamos avanzar a la siguiente. Este método nos permitirá planificar con precisión cada etapa, incluyendo su duración, las tareas involucradas y sus dependencias mutuas, lo que a su vez nos proporcionará una comprensión realista de los costos asociados a la ejecución del proyecto.

Las fases del proyecto se definen de la siguiente manera:

- **Planificación:**
  En esta fase inicial, se establecerá una visión clara del proyecto y se definirán sus objetivos. Se elaborará un plan de trabajo detallado que incluirá un listado exhaustivo de las tareas a realizar, junto con la estimación de las horas necesarias para cada tarea. Asimismo, se identificarán tareas críticas y secundarias, lo que permitirá adaptar el alcance del proyecto según las horas de trabajo disponibles. En esta etapa también se identificarán los riesgos potenciales del proyecto y se propondrán estrategias de mitigación.

- **Análisis y requisitos:**
  Se llevará a cabo un análisis profundo de las necesidades y requerimientos establecidos en el enunciado del trabajo. Durante esta etapa, se prestará especial atención para garantizar una comprensión sólida de lo que se espera del sistema y, de esta manera, evitar posibles fallos en el producto final.

- **Diseño:**
  En esta fase, se traducirán las especificaciones recopiladas durante el análisis y la toma de requisitos en un diseño detallado de la base de datos. Se comenzará con la creación del modelo conceptual, luego se llevará a cabo el diseño lógico de la aplicación y, finalmente, el diseño físico, que incluye la

definición de tablas, relaciones, atributos y restricciones. También se considerará la optimización del rendimiento para adaptarse a las necesidades de DataWareHouse indicadas por el cliente.

- **Implementación:**
  En esta etapa se construirá la base de datos siguiendo las especificaciones y el diseño previamente establecidos. Esto incluirá la creación de tablas, vistas, procedimientos almacenados y otros objetos necesarios. Además, se realizará una carga inicial de un conjunto de datos para llevar a cabo pruebas y validaciones.

- **Pruebas:**
  Es crucial garantizar que el sistema de base de datos funcione conforme a lo previsto. Por tanto, se llevarán a cabo pruebas unitarias para cada componente de la base de datos, así como pruebas generales para verificar que todos los elementos funcionen de la manera esperada.

- **Despliegue**:
  Todo proyecto culmina con el despliegue de la base de datos en un entorno productivo. Esto implicará la instalación y configuración de un servidor que alojará la base de datos final. También será esencial definir procedimientos de copia de seguridad y recuperación tanto para el sistema como para los datos, así como realizar una monitorización del servicio para asegurar el rendimiento y la disponibilidad del mismo.
  Es importante mencionar que esta fase, así como la fase de Mantenimiento, común en muchos proyectos informáticos, no se llevará a cabo en este caso, ya que nuestro proyecto concluye con las fases de implementación y pruebas.

Además, hemos incorporado una adaptación al modelo para incluir la fase de "revisión y documentación", que se llevará a cabo de manera transversal a lo largo de todo el desarrollo del proyecto. Esto significa que se realizará revisiones periódicas y se creará documentación al inicio, durante el desarrollo de cada fase y al finalizarlas, con el propósito de registrar y mostrar la evolución del proyecto de manera efectiva.

1.4. Planificación del Trabajo

Para el desarrollo de este TFG, lo primero que se ha realizado ha sido un inventario del tiempo disponible para el desarrollo del mismo, teniendo en cuenta los días disponibles para cada una de las entregas previstas. Cabe destacar que por problemas personales no se ha podido comenzar con el proyecto en las fechas previstas, lo cual penaliza la cantidad de horas disponibles para las dos primeras entregas (PEC1 y PEC2).

A continuación, se exponen las fechas de entregas teóricas y reales según mi situación personal, junto con la disponibilidad en días y horas.

Para la ejecución de este Trabajo de Fin de Grado (TFG), se ha realizado en primer lugar una evaluación minuciosa del tiempo disponible, tomando en consideración los plazos establecidos para cada una de las entregas programadas.

Sin embargo, es fundamental subrayar que hubieron asuntos personales que incidieron en el tiempo disponible para este proyecto, afectando negativamente al cumplimiento del calendario original del TFG, en especial en la disponibilidad de horas dedicadas a las dos primeras entregas, la PEC1 y PEC2.

En vista de esta situación, resulta necesario documentar tanto las fechas de entrega originalmente planificadas como las fechas reales de acuerdo con mi situación personal. Esta comparativa resulta importante para contextualizar el proyecto y comprender la distribución del tiempo a lo largo del desarrollo del TFG.

En las siguientes secciones, se detallará la disponibilidad de tiempo en términos de días y horas, mostrando cómo se ha adaptado el cronograma a las circunstancias personales, y cómo se han reorganizado los recursos para cumplir con los compromisos del TFG y las exigencias del proyecto

Actividad	F. Inicio teórica	F. entrega teórica	Días Teóricos
PEC1. Plan de trabajo.	28/09/2023	09/10/2023	11
PEC2. Seguimiento.	10/10/2023	13/11/2023	34
PEC3. Seguimiento.	21/11/2023	21/12/2023	30
Entrega final TFG.	19/12/2023	15/01/2024	27

						Días Disponibles		Horas Disponibles			
Actividad	F. inicio Real	F. entrega Real	Días Reales	Días L-V	Días S-D	Horas L-V	Horas S-D	Festivos	Total	Con Margen	Reque-ridas
PEC1. Plan de trabajo.	11/10/2023	18/10/2023	7	5	2	10	10		20	16	20
PEC2. Seguimiento.	18/10/2023	13/11/2023	26	18	8	36	40		76	61	65
PEC3. Seguimiento.	21/11/2023	21/12/2023	30	22	8	44	40		84	67	65
Entrega final TFG.	19/12/2023	15/01/2024	27	20	8	40	40	18	98	78	125

Los criterios para determinar los recursos disponibles fueron:

Horas disponibles de Lunes a Viernes	2
Horas disponibles Sábados y Domingos	5
Margen de horas para desfases	20%
(*) Para la entrega final se añaden horas de los festivos del calendario ya que será necesario contar con algunas horas más para llegar a tiempo con el proyecto.	

Una vez que se ha identificado el tiempo disponible, se procedió a abordar la planificación de las tareas necesarias para el proyecto. Cada tarea fue analizada para obtener una estimación lo más realista posible de las horas requeridas, con el propósito de permitir su posterior programación y considerando las fechas de entrega establecidas para cada una de las Pruebas de Evaluación Continua (PEC).

1.4.1. Entregas

Las entregas se han estructurado de la siguiente manera:

- **PEC 1:**
  Esta fase se enfoca principalmente en la etapa de planificación, y su entrega comprenderá un listado de tareas debidamente programado y presentado en formato de Diagrama de Gantt.

- **PEC 2:**
  Para la segunda entrega, se abordará de manera integral la fase de Análisis y Requisitos, con una extensión que incluirá también una parte de la fase de Diseño. Se entregará el diseño conceptual en este punto.

- **PEC 3:**
  La tercera entrega se dedicará por completo a la fase de Diseño, culminando con los diseños lógicos y físicos del proyecto.

- **Entrega Final:**
  En la entrega final, se contemplarán las fases de Implementación y Pruebas. Se presentará el producto final en su totalidad, acompañado de la memoria del TFG y toda la documentación generada a lo largo del proyecto.

1.4.2. Tareas identificadas

Acceso al listado de Tareas On Line

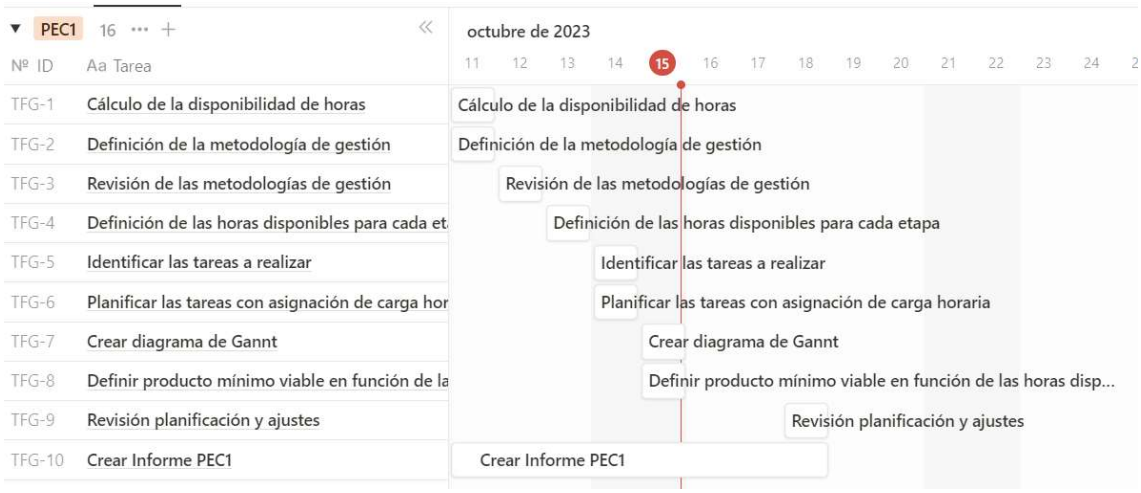
ID	Tarea	Fase	Horas	Tipo	Entrega	Fechas
TFG-1	Cálculo de la disponibilidad de horas	Planificación	1	Crítica	PEC1	11/10/2023 → 11/10/2023
TFG-2	Definición de la metodología de gestión	Planificación	1	Crítica	PEC1	11/10/2023 → 11/10/2023
TFG-3	Revisión de las metodologías de gestión	Planificación	2	Crítica	PEC1	12/10/2023 → 12/10/2023
TFG-4	Definición de las horas disponibles para cada etapa	Planificación	2	Crítica	PEC1	13/10/2023 → 13/10/2023
TFG-5	Identificar las tareas a realizar	Planificación	2	Crítica	PEC1	14/10/2023 → 14/10/2023
TFG-6	Planificar las tareas con asignación de carga horaria	Planificación	2	Crítica	PEC1	14/10/2023 → 14/10/2023
TFG-7	Crear diagrama de Gannt	Planificación	1	Crítica	PEC1	15/10/2023 → 15/10/2023
TFG-8	Definir producto mínimo viable en función de las horas disponibles	Planificación	2	Crítica	PEC1	15/10/2023 → 15/10/2023

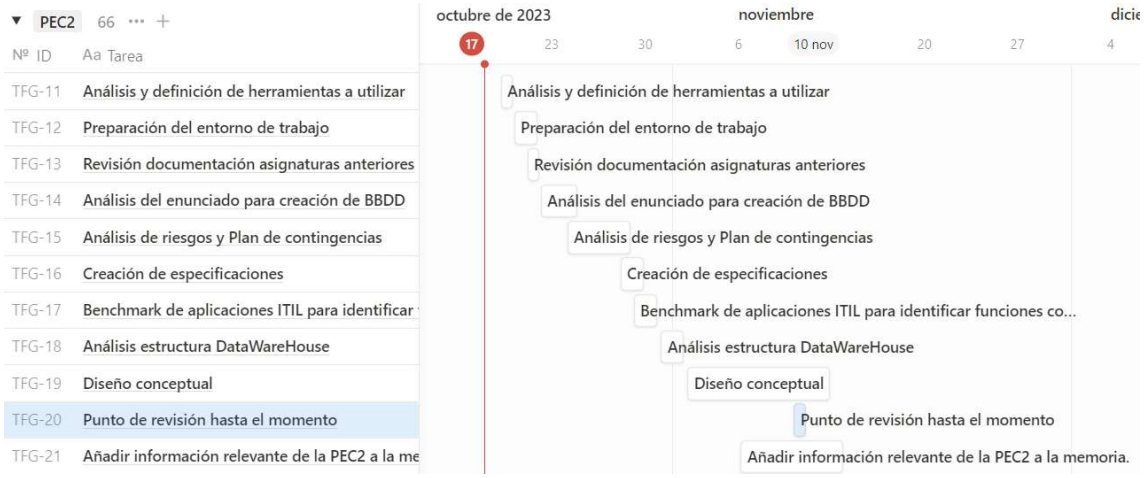
TFG-9	Revisión planificación y ajustes	Revisión	1	Crítica	PEC1	18/10/2023 → 18/10/2023
TFG-10	Crear la memoria con la información necesaria para la PEC1	Documentación	2	Crítica	PEC1	11/10/2023 → 18/10/2023
TFG-11	Análisis y definición de herramientas a utilizar	Análisis y Requisitos	2	Crítica	PEC2	19/10/2023 → 19/10/2023
TFG-12	Preparación del entorno de trabajo	Análisis y Requisitos	4	Crítica	PEC2	20/10/2023 → 21/10/2023
TFG-13	Revisión documentación asignaturas anteriores	Análisis y Requisitos	3	Crítica	PEC2	21/10/2023 → 21/10/2023
TFG-14	Análisis del enunciado para creación de BBDD	Análisis y Requisitos	8	Crítica	PEC2	22/10/2023 → 24/10/2023
TFG-15	Análisis de riesgos y Plan de contingencias	Análisis y Requisitos	10	Crítica	PEC2	24/10/2023 → 28/10/2023
TFG-16	Creación de especificaciones	Análisis y Requisitos	5	Crítica	PEC2	28/10/2023 → 29/10/2023
TFG-17	Benchmark de aplicaciones ITIL para identificar funciones complementarias	Análisis y Requisitos	4	Secundaria	PEC2	29/10/2023 → 30/10/2023
TFG-18	Análisis estructura DataWareHouse	Análisis y Requisitos	5	Crítica	PEC2	31/10/2023 → 01/11/2023
TFG-19	Diseño conceptual	Diseño	20	Crítica	PEC2	02/11/2023 → 12/11/2023
TFG-20	Punto de revisión hasta el momento	Revisión	2	Crítica	PEC2	15/10/2023 → 13/11/2023
TFG-21	Añadir información relevante de la PEC2 a la memoria.	Documentación	3	Crítica	PEC2	06/11/2023 → 13/11/2023
TFG-22	Diseño lógico	Diseño	20	Crítica	PEC3	21/11/2023 → 27/11/2023
TFG-23	Comprobación Formas Normales	Diseño	10	Crítica	PEC3	28/11/2023 → 02/12/2023
TFG-24	Diseño físico	Diseño	20	Crítica	PEC3	02/12/2023 → 09/12/2023
TFG-25	Diagrama Entidad Relación	Diseño	10	Secundaria	PEC3	09/12/2023 → 10/12/2023
TFG-26	Punto de revisión hasta el momento	Revisión	2	Crítica	PEC3	11/12/2023 → 11/12/2023
TFG-27	Añadir información relevante de la PEC3 a la memoria.	Documentación	3	Crítica	PEC3	11/12/2023 → 19/12/2023
TFG-28	Creación de tablas	Implementación	5	Crítica	Final	20/12/2023 → 22/12/2023
TFG-29	Creación de Procedimientos Almacenados	Implementación	5	Crítica	Final	22/12/2023 → 24/12/2023
TFG-30	Creación de Juego de Datos para pruebas	Implementación	5	Crítica	Final	25/12/2023 → 25/12/2023
TFG-31	Consultas DWH	Implementación	35	Crítica	Final	26/12/2023 → 07/01/2024

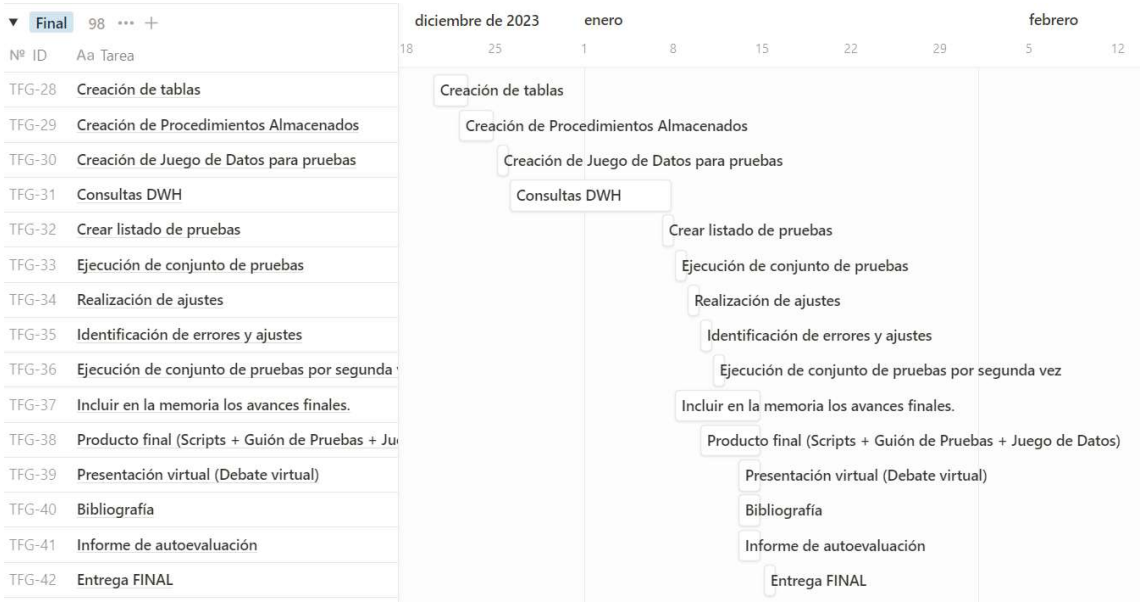
TFG-32	Crear listado de pruebas	Implementación	5	Crítica	Final	07/01/2024 → 07/01/2024
TFG-33	Ejecución de conjunto de pruebas	Pruebas	3	Crítica	Final	08/01/2024 → 08/01/2024
TFG-34	Realización de ajustes	Pruebas	5	Crítica	Final	09/01/2024 → 09/01/2024
TFG-35	Identificación de errores y ajustes	Pruebas	5	Crítica	Final	10/01/2024 → 10/01/2024
TFG-36	Ejecución de conjunto de pruebas por segunda vez	Pruebas	3	Secundaria	Final	11/01/2024 → 11/01/2024
TFG-37	Incluir en la memoria los avances finales.	Documentación	5	Crítica	Final	08/01/2024 → 14/01/2024
TFG-38	Producto final (Scripts + Guión de Pruebas + Juego de Datos)	Documentación	5	Crítica	Final	10/01/2024 → 14/01/2024
TFG-39	Presentación virtual (Debate virtual)	Documentación	10	Crítica	Final	13/01/2024 → 14/01/2024
TFG-40	Bibliografía	Documentación	2	Crítica	Final	13/01/2024 → 14/01/2024
TFG-41	Informe de autoevaluación	Documentación	3	Crítica	Final	13/01/2024 → 14/01/2024
TFG-42	Entrega FINAL	Documentación	2	Crítica	Final	15/01/2024 → 15/01/2024

1.4.3. Diagrama de Gantt

[Acceso al Diagrama de Gantt On Line]







1.4.4. Riesgos identificados

Tras la realización del análisis de riesgos se identifican los listados a continuación, junto con las tareas necesarias para mitigarlos.

ID	Riesgo	Descripción	Impacto	Mitigación
R001	Error en estimación de horas	La ejecución de una tarea en concreto lleva más tiempo del planificado inicialmente,	Alto	- Dejar un margen de tiempo para poder asumir desfases. - Llevar un control constante del tiempo consumido y del estado de las tareas.

		afectando negativamente al cumplimiento de las demás tareas		- Dedicar más horas durante los fines de semana, que son los días disponibles. - En última instancia, solicitar una extensión en la fecha de entrega al profesor, aunque esto penalizaría la entrega siguiente.
R002	Problemas de Hardware	Aunque es poco probable, se debe considerar la posibilidad de destrucción o pérdida del equipo informático donde se desarrolla el proyecto.	Alto	- Almacenar la información en la nube, tanto del proyecto como de los instaladores del software utilizado. - Documentar detalladamente la preparación del entorno para que sea fácil de replicar en otro equipo. - Utilizar ficheros de la nube en modo "Sin conexión". - Aprovechar un segundo equipo (como el portátil del trabajo) para realizar la preparación del entorno de trabajo, especialmente la instalación de Oracle.
R003	Necesidad de mayor dedicación de horas al trabajo	Debido al sector en el que se desarrolla la actividad de la empresa donde trabajo, el periodo invernal podría requerir una mayor dedicación de horas diarias al trabajo, lo que reduciría las horas disponibles para el proyecto.	Medio	- Dejar un margen de horas para poder asumir desfases. - Dedicar más horas durante los fines de semana, que son los días disponibles. - En última instancia, solicitar una extensión en la fecha de entrega al profesor, aunque esto penalizaría la entrega siguiente.
R004	Problemas de conectividad	Poco probable, pero se contempla la posibilidad de perder conexión a internet durante el desarrollo del proyecto.	Medio	- Utilizar ficheros de la nube en modo "Sin conexión"
R005	Problema de acceso a BBDD de desarrollo	Riesgo poco probable, pero se contempla el hecho de que por algún problema del software o por una acción involuntaria la Base de Datos quede inoperativa.	Medio	- Aprovechar un segundo equipo (como el portátil del trabajo) para realizar la preparación del entorno de trabajo, especialmente la instalación de Oracle. - Documentar detalladamente la preparación del entorno para que sea fácil de replicar en otro equipo. - Guardar el desarrollo de los scripts en un directorio en la nube y trabajar con Notepad++ para mantener los documentos en memoria en caso de cierre o apagado inesperado del equipo.
R006	Fallo en especificaciones	Se contempla la posibilidad de haber fallado en la identificación de los requisitos funcionales y no funcionales del proyecto.	Alto	- Dejar un margen de horas para poder asumir desfases. - Dedicar el máximo tiempo posible a la identificación de los requisitos. - En caso de dudas sobre los requisitos identificados, contactar de inmediato con el profesor para aclararlas.
R007	No dedicar horas prevista por motivos personales	Se engloba en este riesgo cualquier tipo de motivo personal imprevisto que impida dedicar menos horas de las previstas (enfermedad, trámites, viajes, etc)	Medio	- Dejar un margen de tiempo para poder asumir desfases. - Dedicar más horas durante los fines de semana, que son los días disponibles. - En caso de que el volumen de horas sea alto, comunicar de manera proactiva al profesor y buscar soluciones antes de que afecte al proyecto. - En última instancia, solicitar una extensión en la fecha de entrega al profesor, aunque esto penalizaría la entrega siguiente.

Del análisis realizado, vemos que el hecho de establecer un margen de tiempo coherente para poder asumir los posibles desfases será clave en el correcto desarrollo del proyecto y en la mitigación de los riesgos identificados.
Se considera correcta pues la decisión de haber establecido un 20% de horas de margen en la planificación inicial de las tareas.

1.5. Breve sumario de productos obtenidos

Como productos finales de este trabajo, se obtendrán los siguientes entregables:

- **Planificación del proyecto:**
  Consistirá en un documento detallado que reflejará la planificación de todas las tareas necesarias para llevar a cabo el proyecto. Con el propósito de lograr este objetivo, se ha optado por utilizar la herramienta *Notion* como gestor de tareas. Esta plataforma nos permite crear bases de datos de tareas y facilita la manipulación de la información de diversas maneras, lo que nos permite obtener diversas visualizaciones. Hemos decidido añadir a los entregables las siguientes "vistas":

    - Listado de tareas: Se trata de una presentación detallada de cada tarea en formato de tabla, incluyendo la información necesaria para su categorización y organización.

    - Diagrama de Gantt: Consiste en una visualización que muestra las tareas en una línea de tiempo, lo que facilita la programación y seguimiento.

    - Cuadro de control Kanbann / Waterfall: Con esta visualización podremos gestionar las tareas de manera clara y eficiente, con tareas organizadas según las diferentes fases definidas en la metodología "en cascada" para este proyecto y clasificadas según su estado actual.

- **Memoria del Trabajo Final de Grado:**
  Este documento, que acompañará la entrega final del TFG, sintetiza todo el trabajo realizado durante el proyecto, abarcando tanto la planificación anteriormente mencionada como las decisiones tomadas en las diferentes etapas del proyecto.

- **Scripts:**
  Este entregable comprende un conjunto de scripts en código PL/SQL utilizados para la creación de tablas, vistas, procedimientos almacenados y demás objetos necesarios para la creación de la base de datos.

- **Guion de Pruebas:**
  Se creará un guion detallado de pruebas a realizar después de la implementación de la base de datos y se adjuntará a la entrega como parte del producto final.

- **Juego de datos para pruebas:**
  Se proporcionará un conjunto de scripts en PL/SQL destinados a la carga de datos inicial de prueba que se utilizará para realizar los testeos necesarios.

- **Presentación virtual:**
  Este documento servirá como la base de la presentación que se realizará durante la defensa virtual ante el tribunal evaluador.

- **Bibliografía:**
  Anexo que se adjuntará a la memoria donde se listará toda la bibliografía consultada para la elaboración del proyecto.

- **Informe de autoevaluación:**
  Corresponde con un documento requerido para el desarrollo de este TFG que contendrá una autoevaluación por parte del estudiante.

1.6. Breve descripción de los otros capítulos de la memoria

A lo largo del documento encontraremos el contenido de este segregado en diferentes apartados:

- **Seguimiento de la Planificación:**
  En este apartado iremos explicando el desarrollo de cada una de las fases y los hitos más relevantes de cada una de ellas. También dejaremos constancia de las tareas que han tenido algún tipo de problema o los percances sucedidos en el desarrollo de estas.

- **Ética, responsabilidad y sostenibilidad:**
  Se añade un apartado para comentar específicamente los aspectos relevantes a la gestión de una base de datos de forma ética, sostenible y socialmente responsable.

- **Análisis y Requisitos del proyecto:**
  Explicaremos aquí en detalle el análisis realizado sobre la documentación entregada por el cliente y obtendremos el listado de requisitos funcionales y no funcionales que se buscan para la aplicación.

- **Diseño:**
  Aquí nos centraremos en explicar detalladamente cada uno de las decisiones tomadas en la fase de Diseño, tanto a nivel de diseño conceptual y lógico como a nivel de diseño físico.

- **Implementación:**
  Haremos una presentación de los pasos realizados en la fase de implementación. Explicaremos cómo ha sido el desarrollo de esta fase y los resultados obtenidos.

- **Pruebas:**
  Comentaremos en detalle el juego de pruebas definidos junto con el conjunto de datos que se ha preparado para dichas pruebas.

- **Conclusiones:**
  Se expondrán las conclusiones generales del proyecto.

- **Glosario:**
  Listaremos aquellos términos de uso poco frecuente con el fin de que cualquier lector sea capaz de seguir el contenido del documento con facilidad.

- **Bibliografía:**
  Listaremos las fuentes consultadas para el desarrollo del proyecto.

2. Seguimiento de la planificación

En este apartado se llevará a cabo un breve resumen de seguimiento de cada una de las fases, también se reflejarán los principales hitos e información importante sobre lo sucedido.

2.1 Fase Planificación

En esta fase, la prioridad fue entender en detalle los objetivos que se plantean para el proyecto. Se ha dedicado tiempo a investigar y evaluar cuál sería la estrategia más adecuada para llevar a cabo este trabajo de la manera más efectiva posible. Tras realizar este análisis, se define que la metodología "Waterfall" (en cascada) sería la elección más idónea para la gestión de este proyecto.

Esta estrategia implica la división del proyecto en fases bien definidas. Para cada una de estas fases se ha definido una lista de tareas que deben llevarse a cabo. Se realiza una minuciosa estimación de la cantidad de horas que requeriría cada una de estas tareas, y se utiliza esta estimación como base para la planificación de todo el proyecto. Con esto, se consigue un calendario detallado que abarca cada tarea, fase y el proyecto en su conjunto.

Cabe destacar que la fase de planificación se vio ligeramente afectada por una circunstancia personal que retrasó el inicio del proyecto algunos días. Sin embargo, gracias a la comprensión del profesorado, se permitió la extensión de la fecha de entrega de la fase inicial (PEC1). Sin embargo, esto ha tenido un impacto negativo en la cantidad de horas disponibles para la siguiente fase.

Por otro lado, es importante mencionar que se ha logrado mantener el desarrollo de la fase de planificación en línea con las horas previstas para la entrega, sin que se registren desviaciones significativas en el desarrollo de las tareas programadas.

Por último, a raíz del *feedback* por parte del profesorado de la primera entrega, surge una tarea a cubrir que no se había previsto inicialmente. La misma hace referencia a plantear una serie de acciones para que el diseño de la base de datos sea ético, sostenible y socialmente responsable, por lo tanto, se añade la tarea *TFG-43 Planteamiento Acciones "Ética, Sostenibilidad, Responsabilidad Social"*

2.2. Fase Análisis

Análisis de Riesgos

Durante el transcurso de la fase de planificación, vimos que existía la necesidad de realizar un análisis de riesgos que, en un principio, no habíamos tenido en cuenta en la fase inicial del proyecto. Tras avanzar, nos encontramos con que esto ha sido un error, ya que la identificación de los riesgos podría haber tenido un impacto importante en la definición de las tareas a llevar a cabo, afectando a la planificación global del proyecto.

Para afrontar esta situación, se modifica la planificación de la segunda fase. Ahora, la tarea de análisis de riesgos (TFG-015) se ha establecido como la

primera actividad a llevar a cabo dentro de dicha fase. Este cambio tiene como objetivo garantizar que los riesgos se aborden y se tengan en cuenta de manera proactiva en todo el desarrollo del proyecto, y no como un elemento posterior.

Es importante destacar que, tras realizar este análisis de riesgos, no surgieron nuevas tareas de mitigación. Las tareas originalmente propuestas resultaron ser suficientes para mitigar correctamente todos los riesgos identificados. Con este ajuste se asegura que la gestión de riesgos esté integrada en el flujo de trabajo antes de continuar.

Herramientas y entorno de trabajo

En línea con lo estipulado en las tareas TFG-11 y TFG-12 se realiza el análisis de las herramientas a utilizar para el desarrollo del proyecto y la preparación del entorno de trabajo para la implementación de la base de datos.

En primer lugar, se realiza una investigación sobre las diferentes versiones de Oracle DataBase y sobre OracleCloud, concluyendo en la decisión de trabajar con un servidor de BBDD Oracle Express instalado en el equipo local. Para la gestión de la BBDD se utilizará SQL DEVELOPER.

- Oracle Database XE 21c
https://www.oracle.com/es/database/technologies/appdev/xe/quickstart.html
- SQL Developer 23.1
https://www.oracle.com/database/sqldeveloper/technologies/download/

Paralelamente, se necesita definir las herramientas más adecuadas para la consecución de las siguientes tareas:

- **Gestión de la planificación y las tareas:**
  Teniendo en cuenta la experiencia personal previa con diferentes herramientas de Gestión de Proyectos, como pueden ser Microsoft Proyect, Asana, Trello o Axosoft, se ha definido la utilización del software Notion para este fin. Esta herramienta resulta muy versátil para este tipo de gestiones, pudiendo adaptar las visualizaciones según las diferentes necesidades, como por ejemplo gestión de tablas, bases de datos, diagramas de Gannt, documentación, entre otros.

- **Elaboración de Scripts:**
  Para la creación de scripts de PL-SQL se utilizará la herramienta Notepad++, ya que éste ofrece una funcionalidad de autoguardado que nos protege ante cierres inesperado del software o del PC. Si bien el proceso de realización puede ser un poco más engorroso, nos aseguramos así de contar con un backup de los diferentes scripts.

- **Realización del diseño conceptual:**
  En este caso, basándonos en el uso previo de la herramienta, se opta por utilizar DrawIO. La funcionalidad que nos brinda es la suficiente para conseguir cumplir con el objetivo solicitado en el proyecto.

- **Creación de entregables:**
  Utilizaremos la suite de Microsoft Office 365 para este fin, en concreto los productos Word, Excel y Powerpoint para el desarrollo de los diferentes entregables.

- **Sistema Operativo:**
  Trabajaremos con Windows 11, la versión que ya tiene instalada mi equipo:
  Edición Windows 11 Home
  Versión 22H2
  Instalado el 09/02/2023
  Versión del sistema operativo 22621.2428
  Experiencia Windows Feature Experience Pack 1000.22674.1000.0

- **Otros recursos necesarios:**
  - Materiales de la asignatura de Gestión de Proyectos
  - Materiales de las asignaturas Uso de BBDD, Diseño de BBDD y Arquitectura de BBDD.
  - PEC's y Prácticas de las asignaturas antes mencionadas.
  - Guías de instalación de los diferentes softwares utilizados y sus manuales.

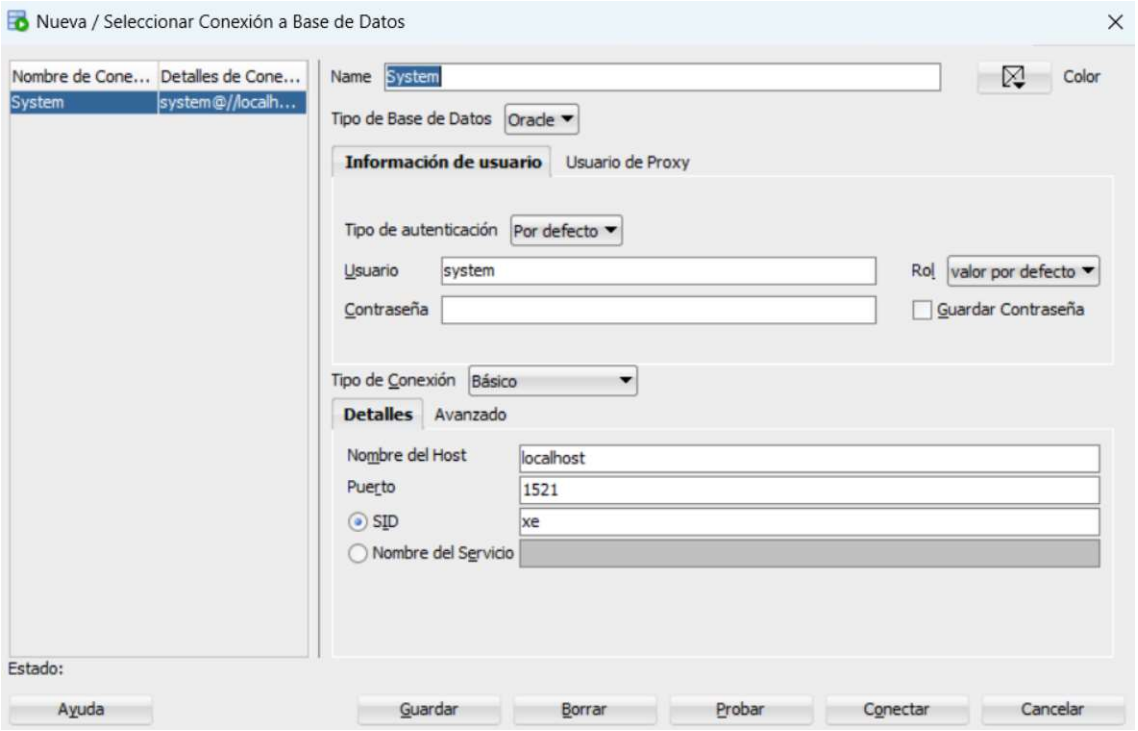
Por último, cabe mencionar que, en cuanto a hardware, realizaremos el proyecto trabajando sobre un equipo con las siguientes características:

Nombre del dispositivo	JPONOTE
Procesador	Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz 1.99 GHz
RAM instalada	16,0 GB (15,9 GB usable)
Identificador de dispositivo	8F2B0318-2D78-439A-8681-74A7E52B7E8A
Id. del producto	00325-96365-18521-AAOEM
Tipo de sistema	Sistema operativo de 64 bits, procesador basado en x64

Respecto a la preparación del entorno de trabajo se realiza la instalación del SGBD Oracle, siguiendo los pasos del documento adjunto "Instalación Oracle", y de SQL Developer para la gestión de la base de datos.

InstalacionOracle.pdf

A continuación, se crea la base de datos que utilizaremos para el proyecto, creando en primer lugar una conexión con el usuario "system" y lanzando a posteriori un conjunto de scripts de prueba para comprobar que todo funciona correctamente.



Tras finalizar con las pruebas, se creará un usuario específico para el desarrollo de la práctica, ya que es importante no trabajar con el usuario SYSTEM por tener todos los privilegios sobre la BBDD.

Se creará un usuario TFG con un *tablespace* por defecto TFG y un *tablespace* temporal TEMP.

Se crea el usuario de la siguiente manera:

```
CREATE USER "TFG"
IDENTIFIED BY Uoc1234
DEFAULT TABLESPACE users
QUOTA UNLIMITED ON users
TEMPORARY TABLESPACE temp;
```

Se otorgan los permisos necesarios al usuario recién creado:

```
GRANT
CREATE SESSION,
CREATE USER,
CREATE TABLE,
CREATE VIEW,
CREATE TRIGGER,
CREATE SEQUENCE
TO "TFG";
```

Finalmente, se crea una nueva conexión con el nuevo usuario y con esto, ya está el entorno listo para comenzar a crear tablas, vistas y demás objetos necesarios sin correr el peligro de tocar información sensible de la BBDD.

Análisis del enunciado

Respecto a la tarea “TFG-16 Creación de especificaciones” surge la duda de si el planteo realizado es el esperado, por lo que se contacta con el profesorado para aclarar esta cuestión.

Lo mismo sucede con la tarea de “TFG-18 Análisis de la estructura DataWareHouse”, han surgido una serie de dudas que han tenido que consultarse con el profesorado, aun así, esto no ha sido un *stopper* y se ha podido continuar con el desarrollo de las siguientes tareas mientras se esperaba la respuesta por parte del profesorado.

Las dudas fueron aclaradas rápidamente lo que ha permitido finalizar ambas tareas sin problemas.

Cabe destacar que se ha decidido dejar la tarea secundaria “TFG17 - Benchmark de aplicaciones ITIL para identificar funciones complementarias” para ejecutarla posteriormente. Se prevé la necesidad de utilizar las horas asignadas a esta tarea para otra de mayor relevancia, asumiendo que de contarse con el tiempo necesario se retomaría esta tarea posteriormente.

Por el resto de las tareas de la fase de análisis se han desarrollado con normalidad y sin registrarse contratiempos.

2.3. Fase Diseño

Diseño Conceptual

El desarrollo de la tarea “TFG-22 Diseño conceptual” se ha visto afectado negativamente en cuanto a las horas inicialmente planificadas. Por un lado, el diseño en sí ha llevado más tiempo del previsto inicialmente porque hubo ciertas dudas respecto a cuál sería la mejor manera de plantearlo. A esto se le ha sumado que, por cuestiones laborales, no he podido dedicar el tiempo planificado, tal y como se había mencionado en el riesgo “R003 - Necesidad de mayor dedicación de horas al trabajo”.
Una buena parte de las horas fue asumida con el margen de tolerancia estipulado para este tipo de situaciones, sin embargo, no ha sido suficiente con lo que no se ha alcanzado a finalizar con el diseño conceptual en su totalidad.

Ante esta situación, se ha decidido centrar los esfuerzos en el diseño conceptual de la base de datos en sí, dejando para una segunda instancia el diseño conceptual del apartado de DataWare House. Para ello, se crea una nueva tarea en la fase de diseño, la “TFG-44 – Diseño Conceptual DWH” la cual se planifica como primera tarea del intervalo de tiempo correspondiente a la PEC3.

Diseño Conceptual DataWareHouse

Como ya se ha comentado, la primera tarea del tiempo correspondiente a la PEC3 ha sido la realización del diseño conceptual del apartado de DataWareHouse. Esta tarea se ha realizado en el tiempo estipulado sin inconvenientes.

Diseño Lógico

El siguiente ítem que llevar a cabo ha sido la tarea “TFG-22 Diseño Lógico”. El objetivo de esta tarea era convertir todo lo diseñado hasta el momento en el diseño conceptual en un esquema lógico relacional, teniendo en cuenta que el tipo de base de datos a utilizar sería una base de datos relacional.

El desarrollo de esta tarea nos ha servido para identificar pequeños errores cometidos durante el diseño conceptual, los cuáles serán expuestos en el apartado correspondiente al "Diseño Lógico".

El tiempo dedicado ha sido el previsto inicialmente en la planificación.

Diseño Físico

Al momento de llevar a cabo la principal tarea de esta fase, la "TFG-24 Diseño Físico" hemos visto algunos puntos importantes:

- A priori, el diseño llevaría más horas de las previstas.
- Habíamos creado la tarea "TFG-28 Creación de tablas" que se llevaría a cabo junto con la tarea TFG-24.
- Sería necesario crear un conjunto de datos de pruebas para comprobar la correcta creación de las tablas y hacer un primer test con el objetivo de identificar posibles fallos del diseño. Estas pruebas han sacado a la luz algunos puntos a corregir.

Con estos factores, el tiempo de diseño físico planificado inicialmente se ha extendido entre en un 40% y un 50%, con lo que hemos absorbido las horas de margen establecidas inicialmente más horas de otras tareas para poder llegar a término.

Por otro lado, la tarea "TFG-25 Diagrama Entidad Relación" no llegamos a abordarla por una cuestión de tiempo disponible. Al tratarse de una tarea categorizada como secundaria se decide no llevarla a cabo por el momento y, de ser posible, realizarla en tiempo de la PEC4.

Finalmente, cabe destacar que durante el diseño surge la necesidad de crear una tarea específica para crear los diferentes disparadores necesarios en nuestra base de datos. La nueva tarea es la "TFG-45 Creación Triggers Campos Calculados" y se ha calendarizado como primera tarea del tiempo dedicado para la PEC4.

2.4. Fase Implementación

Comenzamos la fase implementación desarrollando la tarea "TFG-28 Creación de Tablas", la cual resulta bastante simple, puesto que el script de creación lo obtenemos casi en su totalidad de la fase de diseño físico. Realizamos una revisión, algunos ajustes y ejecutamos el script para tener todas las tablas necesarias en nuestra BBDD. Esta tarea se desarrolla sin problemas, incluso en 2 horas menos del tiempo previsto.

El siguiente punto que abordamos es la creación de los disparadores que se ocuparán de gestionar los campos que hemos definido como campos calculados en el diseño conceptual. Comenzamos la tarea "TFG-45 Creación Triggers Campos Calculados" por identificar cuáles son los campos calculados que tenemos en nuestra base de datos y trabajamos en los scripts para crearlos. Aquí nos encontramos con una problemática, ya que, tras el análisis, habíamos previsto que uno de los trigger consulte información de la propia tabla. Esta acción Oracle no la permite, generando un error "mutating table error", por lo que hemos investigado en la aplicación de Computed Triggers, lo que soluciona el problema.

la tarea "TFG-30 Creación de Juego de Datos para pruebas"

Antes de avanzar con el orden de tareas establecido, hemos visto conveniente hacer una revisión del orden de las tareas a ejecutar. Consideramos que no era lógico ejecutar la tarea “TFG-29 Creación de Procedimientos Almacenados” antes de la definición de las pruebas. Ejecutamos pues, la tarea “TFG-30 Creación de Juego de Datos para pruebas”, la cual ya estaba bastante avanzada porque se hizo una primera introducción de datos mientras se realizaba el diseño físico; para luego listar el conjunto de procedimientos almacenados que se deberían crear junto con las pruebas a realizar en cada uno de ellos. De esta forma, abordamos la tarea “TFG-32 Crear listado de pruebas” y posteriormente procedemos a la creación de los procedimientos almacenados.

Ésta última tarea de la fase de implementación ha generado un desvío importante en cuanto a las horas previstas: de unas 8hs teóricas a a unas aproximadamente 20hs reales. Cabe destacar que durante el desarrollo se han ido haciendo una serie de tests unitarios que han hecho que se tarde más de lo previsto.

Llegados a este punto y de manera previsoria, hemos decidido aplazar la tarea “TFG-31 Consultas DWH” para el final de la fase de pruebas. Si bien ejecutarla a posteriori podría requerir un esfuerzo extra en cuanto a cantidad de horas por el hecho de tener que realizar un segundo conjunto de pruebas, se ha considerado la mejor opción para poder llegar con un producto funcional y bastante avanzado a la fecha de entrega del proyecto.

2.5. Fase Pruebas

Tras trabajar en el desarrollo de los procedimientos almacenados, y por las sinergias identificadas entre sí, se han desarrollado las tareas correspondientes a la fase de pruebas de manera conjunta:

- “TFG-33 Ejecución de conjunto de pruebas”
- “TFG-34 Realización de ajustes”
- “TFG-35 Identificación de errores y ajustes”
- “TFG-36 Ejecución de conjunto de pruebas por segunda vez”

Esta fase en su totalidad ha sido mal valorada en cuanto a las horas necesarias para su ejecución. Vemos que la suma de estas 4 tareas representa un total teórico de 16 horas, mientras que en la realidad se ha extendido a 24 horas aproximadamente. Situación que, sumado a lo ya acumulado, ha generado un desvío importante y han provocado un atraso en el desarrollo de las otras tareas, a pesar de haber utilizado el margen de horas previstos en el riesgo “R001 - Error en estimación de horas”.

3. Ética, responsabilidad y sostenibilidad.

Para el diseño de una Base de Datos, en este caso para la aplicación de control de cambios, no sólo debemos tener en cuenta los aspectos técnicos, sino que también se debe tener en consideración los aspectos éticos, de sostenibilidad y de responsabilidad social. Para ello establecemos una serie de acciones para tener en cuenta que ayudarán a proteger los derechos de los usuarios y a desarrollar software de una manera más responsable y que beneficie a la sociedad en general.

ID	Tipo	Acción
AER001	Aspectos Legales	Garantizar que la base de datos cumple con las leyes y regulaciones en materia de protección de datos.

AER002	Aspectos Legales	Mantener registros y documentación que demuestren el cumplimiento legal.
AER003	Transparencia	Crear documentación sobre cómo se recopilan, almacenan y utilizan los datos.
AER004	Transparencia	Garantizar que los usuarios pueden acceder a información clara sobre sus datos personales y sobre cómo éstos serán utilizados.
AER005	Privacidad y Seguridad	Establecer políticas de privacidad que expliquen cómo se llevará acabo el manejo de los datos del usuario y cómo serán protegidos.
AER006	Privacidad y Seguridad	Aplicar técnicas de encriptación para proteger datos confidenciales del usuario como pueden ser las contraseñas.
AER007	Privacidad y Seguridad	Realización de auditorías de seguridad que garanticen la integridad de los datos y prevengan los accesos no autorizados.
AER008	Responsabilidad Social	Describir cómo tanto la base de datos como la aplicación final puede impactar en los usuarios y en la sociedad.
AER009	Responsabilidad Social	Implementar políticas para evitar el uso de datos con fines no éticos.
AER010	Responsabilidad Social	Considerar cómo cada uno de los cambios en el software puede influir en la experiencia del usuario, en la satisfacción del cliente y en la sociedad en general. Por ejemplo añadiendo un campo de descripción del impacto en el usuario.
AER011	Responsabilidad Social	Formar a los usuarios de la base de datos en prácticas éticas de manejo de datos.
AER012	Sostenibilidad	A la hora de diseñar la base de datos, hacerlo teniendo en cuenta el consumo de recursos, como espacio de almacenamiento y energía.
AER013	Sostenibilidad	Almacenar datos de una manera eficiente y utilizar técnicas de compresión en los casos que sea viable.
AER014	Sostenibilidad	Considerar la eficiencia de las consultas para minimizar la carga en los servidores.

Con el estudio de las acciones planteadas, vemos necesario crear un requisito no funcional para englobar aquellas tareas relacionadas con estos aspectos.

Por último, cabe destacar que el desarrollo de la aplicación está fuera del alcance del proyecto, pero de estarlo también podríamos tener en cuenta aspectos de accesibilidad y usabilidad de la aplicación. De esta manera, se podría establecer políticas para conseguir, por ejemplo, que usuarios con discapacidades visuales puedan hacer uso
del software.

4. Análisis y Requisitos del proyecto

4.1. Estudio del caso presentado

En este apartado describiré cómo he abordado el análisis del enunciado y la metodología aplicada para desarrollar el diseño de nuestra base de datos.

En primer lugar, se ha llevado a cabo una lectura minuciosa del enunciado en su totalidad. El principal objetivo de esta primera lectura fue ponernos en contexto y obtener una visión general del alcance del proyecto. Esta etapa inicial ha permitido identificar aquellos párrafos clave en los que se encontraría la información esencial para la creación de la base de datos.

Posteriormente, a través de repetidas lecturas, se ha ido identificando aquellas oraciones y frases que dejaban ver las primeras entidades necesarias para el diseño. Por ejemplo, se ha conseguido identificar entidades como "Aplicación" y "Cambios" como puntos de partida, como así también se ha podido identificar la relación entre ellas.

Como metodología de trabajo, simplemente se ha optado por resaltar con un color en el texto aquellas partes que sugerían posibles entidades o atributos, y subrayar aquellas partes que ayudaban a identificar las relaciones entre estas entidades.

Por ejemplo:

Con tal de poder formalizar los cambios que se harán en cualquiera de las aplicaciones en producción, es clave tenerlas controladas. Con este objetivo, se deberá definir un inventario con todas las aplicaciones a considerar. No se podrá pasar a producción una aplicación si no se ha registrado en el inventario. 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, el número medio de usuarios, la tecnologia usada para su desarrollo, el tipo de infrastructura usada, la criticidad de la aplicación, y, en general, cualquier dato que se considere importante para indentificar las aplicaciones y su impacto en caso de no disponibilidad.

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:** persona del departamento de IT que lidera la aplicación desde el punto de vista técnico.
- **Responsable del área de negocio implicada:** persona que lidera las decisiones funcionales a implementar en la aplicación. Es quien coordina las diferentes áreas de negocio que usan la aplicación.
- **Gestor de cambios de la empresa:** persona que debe dar la aceptación para mover a producción cualquier cambio sobre les aplicaciones que formen parte del inventario de la empresa. Esta persona transmite el resultado de las discusiones que se hacen sobre los cambios en la GCAB. No puede aceptar un cambio si éste no ha sido aprovado, formalmente, en la GCAB.

Durante este trabajo de análisis nos hemos encontrado con algunos aspectos que han generado dudas de cómo deberían enfocarse, algunos de ellos se han planteado al

profesorado, mientras que otros se han dejado a un lado por el momento, pero dejándolas inventariadas para poder abordarlas y resolverlas posteriormente.

Dudas surgidas del análisis:

- ¿Cómo modelar las aprobaciones de cada cambio o el flujo de aprobaciones que éstos deben seguir?
  Esta pregunta fue trasladada al profesorado porque la he considerado clave y dependiendo del enfoque puede variar considerablemente el diseño final.
- ¿Cómo modelar el bloque relacionado con el impacto en caso de no disponibilidad de la aplicación? Inicialmente se han ido tratando estos aspectos como atributos tanto de la entidad "Aplicación", como de la entidad "Cambio". Sin embargo, se deja pendiente de revisión si este enfoque puede ser mejorado.
- ¿Sería correcto modelar el "rol" de la persona aprobadora como un enumerado? ¿O debería ser una entidad?
- ¿Cómo debería gestionar campos calculados de ciertas entidades que dependen de cambios o modificaciones de otras entidades?
- La entidad "aprobación", ¿debería ser una entidad asociativa o una relación ternaria?
- ¿Cuál sería la mejor manera de abordar todo el bloque de DataWare House?

Como se ha dicho previamente, a excepción de la primera duda, las demás se han dejado para ser revisadas en la fase de modelado del diseño conceptual y, en caso de no poder resolverlas, se optará por contactar con el profesorado.

4.2. Requisitos del Producto Final

Para la definición de los requisitos funcionales y no funcionales se trabaja y extiende lo identificado en el capítulo "Objetivos del Trabajo" de este mismo documento, analizando en profundidad el enunciado del trabajo.

Se parte de la base que los requisitos funcionales describen las acciones y comportamientos específicos que un sistema o software debe realizar, mientras que los requisitos no funcionales establecen características de rendimiento, seguridad, usabilidad y calidad que el sistema debe cumplir.

ID	Tipo	Requisito
REQ001	Funcional	Gestionar cada uno de los cambios que se prevé realizar dentro de una aplicación
REQ002	Funcional	Contar con un inventario de las aplicaciones a gestionar
REQ003	Funcional	Implementar un flujo de aprobación por cada cambio que se desee implementar
REQ004	Funcional	Contar con un listado de aprobadores y sus posibles sustitutos en caso de ausencias de los aprobadores titulares
REQ005	Funcional	Registrar cada una de las aprobaciones
REQ006	Funcional	Establecer categorías de impacto de los cambios y poder gestionarlas (ABM)
REQ007	Funcional	Poder establecer el alcance geográfico de cada uno de los cambios
REQ008	Funcional	Establecer el impacto económico y a nivel de usuarios en el caso de un cambio mal ejecutado
REQ009	Funcional	Controlar el paso a producción de las aplicaciones
REQ010	Funcional	Almacenar el estado de cada paso a producción y registrar cómo ha sido el proceso.
REQ011	Funcional	Registrar planes de acción para cada paso a producción no correcto.

REQ012	Funcional	Registrar las auditorías que se realizan sobre la aplicación y los incumplimientos detectados.
REQ013	Funcional	Inicializar la BBDD con un conjunto de datos
REQ014	Funcional	Ejecutar un exhaustivo juego de pruebas
REQ015	Funcional	Realizar un log de acciones realizadas en la BBDD
REQ016	Funcional	Implementar un mecanismo para testear la funcionalidad de la BBDD
REQ017	NO Funcional	Gestión y acceso a la información (ABM) mediante procedimientos almacenados.
REQ018	NO Funcional	La aplicación ha de ser escalable y permitir gestionar cualquier volumen de datos
REQ019	NO Funcional	Los resultados deben ser en tiempo constante 1
REQ020	NO Funcional	Estandarizar el registro de logs
REQ021	NO Funcional	Implementar una correcta gestión de errores y excepciones.
REQ022	NO Funcional	Plantear diferentes líneas de evolución de la BBDD
REQ023	NO Funcional	Añadir los campos, tablas o funciones necesarias para conseguir una base de datos ética, sostenible y socialmente responsable.

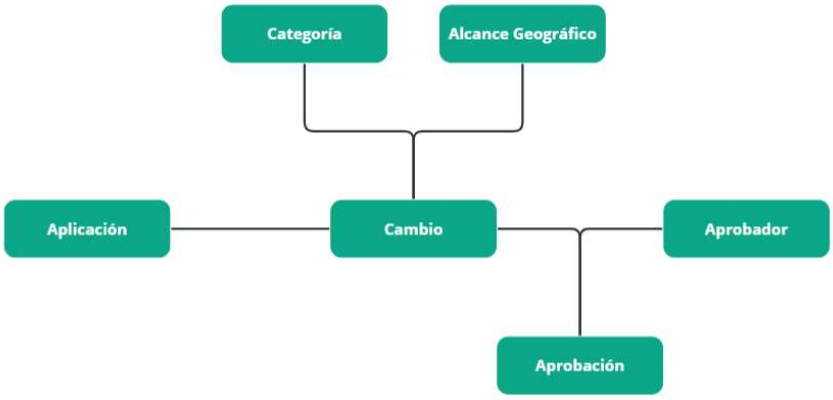
5. Diseño

5.1. Diseño Conceptual

Como ya se ha comentado anteriormente, de las primeras lecturas del enunciado se han dejado ver las principales entidades necesarias para nuestro diseño. Sin embargo, hemos tenido que realizar sucesivas y exhaustivas lecturas con el fin de identificar las diferentes entidades, sus atributos y las relaciones entre todas ellas.

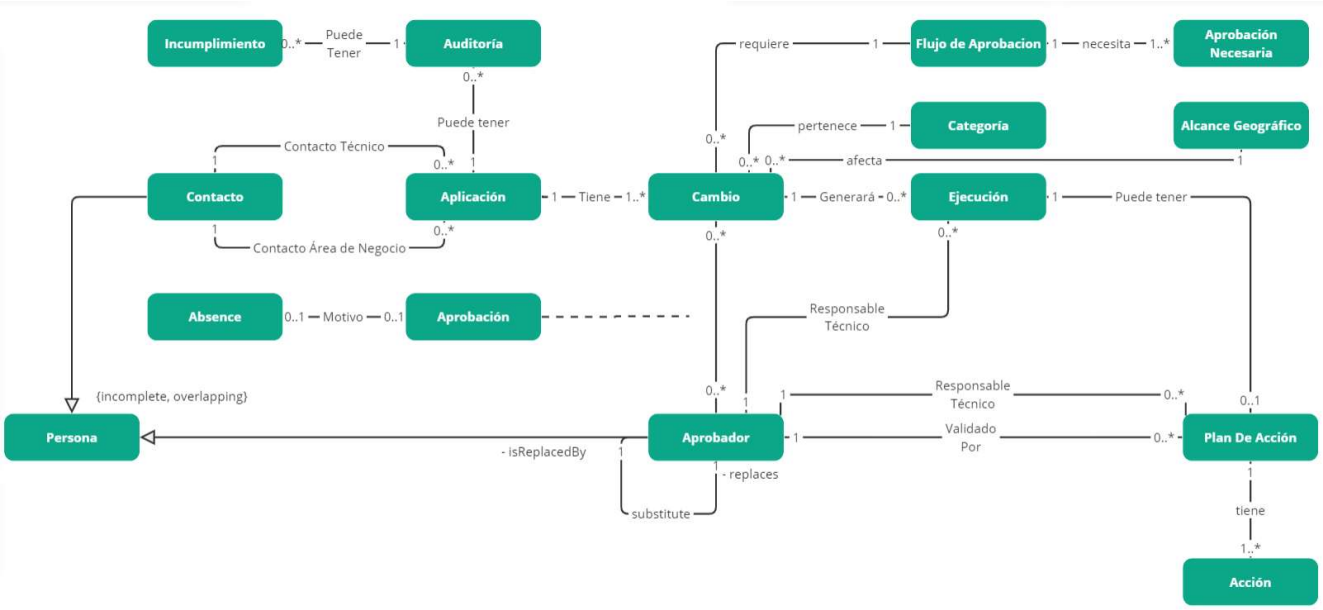
El enfoque que se ha adoptado para llevar adelante el diseño conceptual de la base de datos ha sido el de ir desglosando los problemas para poder resolver siempre los problemas más pequeños, además de ir abordando aquellos que, a priori, resultan más simples de resolver y dejando los complejos para el final.

Así, en primera instancia, se han podido identificar las principales entidades y sus relaciones: Aplicación, Contacto, Cambio, Categoria, Alcance Geográfico, Aprobador y Aprobación. Considerando esta última, junto con el flujo que se debe realizar, como la parte más compleja de modelar dada la naturaleza del enunciado.



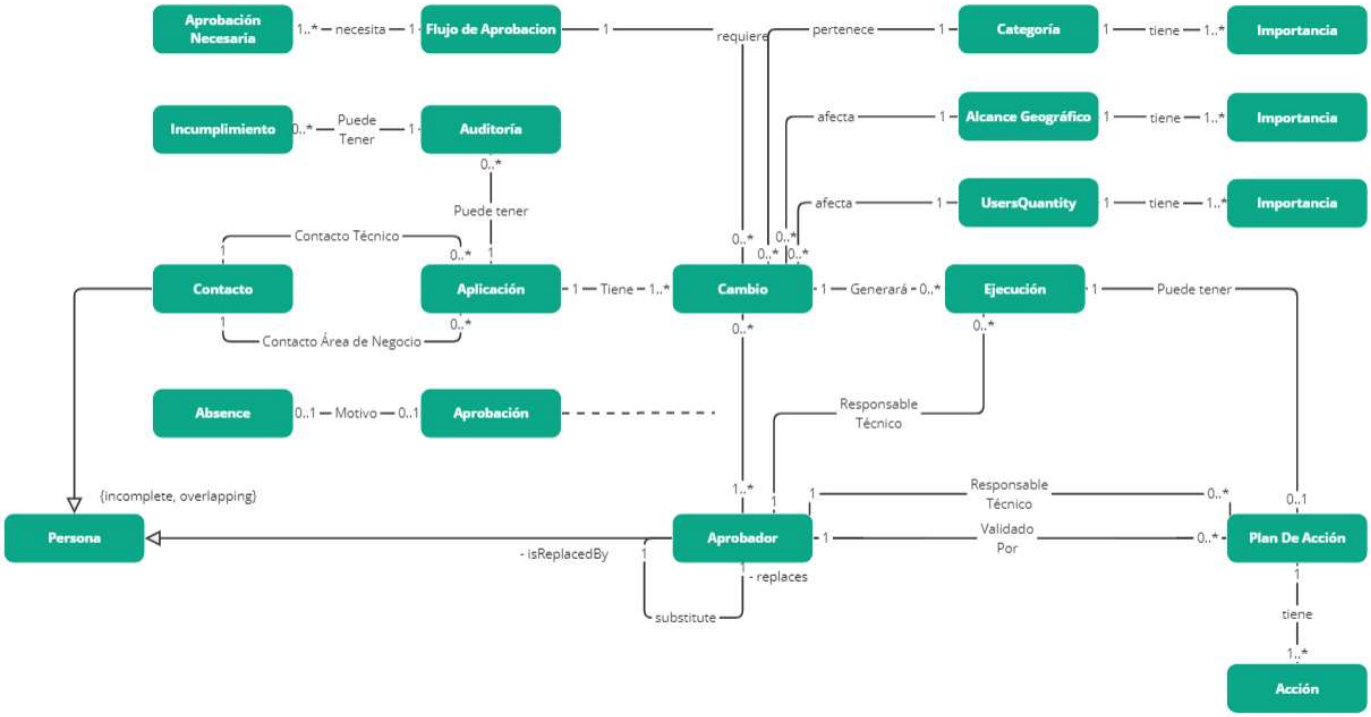
Posteriormente, al trasladar las entidades al entorno gráfico en un documento realizado con el software Miró, hemos ido viendo necesarias nuevas entidades como han sido la entidad Persona y Ausencia. A la vez que empezamos a modelar las partes del enunciado que habíamos dejado para segunda instancia, como ser las entidades que ayudarán a representar la gestión de cambios (entidad Ejecución, PlanAcción y Acción) y la representación de las auditorías mediante la entidad Auditoría e Incumplimiento.

En cuanto al flujo de autorización, se ha decidido modelarlo mediante entidades que permitan definir N flujos y así en cada cambio definir para cada uno de ellos el o los aprobadores necesarios. Se tiene en cuenta también la posibilidad de definir para cada flujo de aprobación si la misma es "en orden" o "independiente". Refiriendo el primer caso al hecho de que el aprobador 2 no pueda aprobar hasta que el aprobador 1 lo haya hecho; o, en la otra casuística, daría igual el orden de los aprobadores siempre que lo hayan hecho. Para modelar estos flujos nos apoyaremos en las entidades FlujoAprobación y AprobaciónNecesaria.



Por último, tras revisar el modelo del impacto que puede tener un cambio que deje la aplicación inoperativa, se ha decidido añadir unas entidades auxiliares: *GeoScopeImportance, CategoryImportance, NUser y NUserImportance.*

A continuación, entraremos en detalle de cada una de estas entidades, no sin antes



mencionar que tanto el diseño conceptual de la base de datos, como el futuro diseño lógico y físico lo haremos teniendo en cuenta:

- La nomenclatura de los objetos será en inglés: así se dará legibilidad a la base de datos a cualquier persona que posteriormente pueda necesitar realizar acciones sobre ellas.
- Cada entidad contará con un atributo de tipo entero autonumérico que servirá como clave primaria de la entidad.
- Las entidades no están "cerradas" a futuros cambios, permitiendo así que la base de datos evolucione con el tiempo.

Entidades

Entidad Aplicación (Application)

Esta entidad ha sido la más evidente pero no así sus atributos, si bien algunos de ellos están claros, otros podrían tratarse de diferentes maneras. Este es el caso de los atributos "persona de contacto técnica" y "persona de contacto por área de negocio". Inicialmente se había pensado en modelar estos atributos como simples *strings* pero se ha considerado que este planteo no era lo suficientemente escalable ni versátil, por lo que se sustituye esta idea por la necesidad de crear una nueva entidad para almacenar los contactos.

Esta nueva entidad tendría una doble relación con la entidad aplicación para poder referenciar por un lado al contacto "técnico" y por otro al del "área de negocio"

Por otro lado, los atributos que son necesarios para medir el impacto económico en caso de no disponibilidad de la aplicación se quedan pendientes de definición final si son necesarios modelarlos como atributos de esta entidad o si se deben de extraer de aquí. Estos atributos son "número medio de usuarios" y "criticidad".

Modificación tras revisión de cómo modelar el impacto:

Se decide finalmente contemplar los atributos "número medio de usuarios" y "criticidad" de la aplicación a modo informativo en esta entidad. Se tendrá trazabilidad del impacto real en caso de no disponibilidad en cada uno de los cambios, ya que cada uno de ellos es el que, individualmente, puede penalizar en el uso de la aplicación. El atributo *avgUsers* será un entero mientras que el atributo *criticality* será un tipo enumerado *criticalLevel* que tendrá los siguientes valores:

- Low
- Medium
- High

Entidad Contacto (Contact)

Esta entidad, que no forma parte del enunciado inicial, surge de la necesidad de poder representar personas que puedan relacionarse con cada aplicación para así poder tener una base de datos con información personal y de contacto de cada aplicación.
Se establecen los atributos típicos de una entidad de este tipo: nombre, apellidos, correo electrónico, número de teléfono y empresa que representa.

Modificación tras análisis de la entidad *UserApprover:*
Como se verá más adelante, al crear la entidad para los aprobadores, surge la necesidad de crear una entidad *Person* que será una generalización que almacenará la mayor parte de los atributos definidos para *Contact.* Ésta última se convierte entonces en una especialización de *Person* y sólo se almacenará en ella el atributo *enterprise,* que representa la empresa para la que cada contacto trabaja.
(*) En una futura evolución del sistema podría convertirse esta empresa en una entidad específica.

Entidad Cambio (Change)

Al identificar esta entidad se puede ver fácilmente la estrecha relación que tiene con la entidad *Application* y también su cardinalidad. El enunciado detalla cómo "El primer cambio en una aplicación se produce cuando se pone en producción…", por lo que se entiende que toda aplicación tendrá mínimo un cambio. Lo que lleva a determinar que la entidad *Application* tendrá una cardinalidad de 1, mientras que la de *Change* tendrá

una cardinalidad 1..*. Esta relación, convierte a la entidad *Change* en una entidad débil respecto a *Application.*

Se establecen los atributos *description, orderDate*, *isApproved* y *approvalDate*, siendo estos últimos atributos calculados que se informarán en base a las aprobaciones que dicho cambio ha recibido.

Por otro lado, al igual que sucede con la entidad *Application,* los atributos que son necesarios para medir el impacto económico en caso de no disponibilidad de la aplicación se quedan pendientes de definición final si son necesarios modelarlos como atributos de esta entidad o si se deben de extraer de aquí. Estos atributos son *affectedUsers* y *economicImpact*.

Modificación tras revisión de cómo modelar el impacto:

Finalmente, se decide modificar el planteamiento inicial de esta entidad para poder modelar el impacto de cada cambio en caso de no disponibilidad. Para ello, se creará un atributo *impact* que será un campo calculado resultante entre la multiplicación de los órdenes de importancia de las entidades *Category, GeoScope y UserQuantity.*

Entidad Categoría (Category)

En el enunciado se ve claramente la necesidad de contar con una entidad para gestionar las categorías, la cual tiene una relación con la entidad *Change,* de cardinalidad 1 a 0..*, ya que todo cambio debe tener una categoría asignada pero podría existir una categoría que no tenga ningún cambio asociado a ella.

En cuanto a sus atributos se establece un *categoryName* para poder identificarla, un *createdDate* para almacenar la fecha de creación en el sistema, un *approvalDate* que guardará la fecha en la que dicha categoría fue aprobada para su uso, un *isDeleted* que será un campo *booleano* con un valor por defecto de *false* y un *deletedDate*, para que, en caso de haber activado el booleano *isDeleted* a *true*, se almacene la fecha en la que se hizo esta acción.

Por último, he dejado aparte la mención del atributo *importance*, dado que si bien este valor será un numérico entero se corresponde con la importancia que se le quiere asignar a cada categoría y puede ser un valor que mute en el tiempo. → Pendiente definir si es necesaria una entidad que sólo almacena la importancia de cada categoría junto con una fecha de *validez hasta*, así podríamos modificar la importancia de las categorías en el tiempo sabiendo siempre a partir de qué momento es válida cada importancia.

Entidad Importancia de Categoría (CategoryImportance)

Modificación tras revisión de cómo modelar el impacto

Finalmente se define crear una nueva entidad que almacene la importancia de cada categoría (*importanceOrder)* y una fecha de validez hasta (*validUntil)*. Las importancias activas no tendrán informado el valor fecha de validez hasta.

Esta entidad tendrá una relación con cardinalidad 1 hacia *Category,* la cual participa con cardinalidad 1..* en la relación. Esto es así ya que una categoría debe tener mínimamente una importancia asignada y puede tener más de una a lo largo del tiempo, pero siempre sólo una vigente. A su vez, la importancia no puede pertenecer a más de una Categoría.

Entidad Alcance Geográfico (GeoScope)

Se determina crear la entidad *GeoScope* para poder gestionar los diferentes tipos de alcances geográficos existentes (local, nacional, global, etc)

Esta entidad almacenará inicialmente dos atributos de tipo string: *scopeName* y *description.*

Se entiende del enunciado que cada cambio puede tener un alcance geográfico diferente y por lo tanto es necesaria una relación entre *GeoScope* y *Change*, con cardinalidad 1 a 1 entre ambas entidades.

Modificación tras revisión de cómo modelar el impacto:

Se añaden a esta entidad una serie de atributos para tratarlos de la misma forma que se tratan a las categorías, ya que con estos elementos se calculará el impacto de cada cambio. Los atributos que se añaden son *createdDate* para almacenar la fecha de creación en el sistema, un *approvalDate* que guardará la fecha en la que dicho alcance geográfico fue aprobado para su uso, un *isDeleted* que será un campo *booleano* con un valor por defecto de *false* y un *deletedDate*, para que, en caso de haber activado el booleano *isDeleted* a *true*, se almacene la fecha en la que se hizo esta acción.

Entidad Importancia del Alcance Geográfico (GeoScopeImportance)

Modificación tras revisión de cómo modelar el impacto

Se define crear una nueva entidad que almacene la importancia de cada alcance geográfico (*importanceOrder)* y una fecha de validez hasta (*validUntil)*. Las importancias activas no tendrán informado el valor fecha de validez hasta.

Esta entidad tendrá una relación con cardinalidad 1 hacia *GeoScope,* la cual participa con cardinalidad 1..* en la relación. Esto es así ya que un alcance geográfico debe tener mínimamente una importancia asignada y puede tener más de una a lo largo del tiempo, pero siempre sólo una vigente. A su vez, la importancia no puede pertenecer a más de un Alcance Geográfico.

Entidad Número de Usuarios (NUsers)

Modificación tras revisión de cómo modelar el impacto

Se define crear esta entidad para almacenar los rangos de números de usuarios afectados por un cambio.
Para esta entidad tendremos los atributos *rangeName, from y to*, siendo estos últimos, dos campos de valor enteros que representarán el valor desde y hasta de cada rango de usuario.

Dado que estos elementos serán clave para determinar el impacto de un cambio, se trata esta entidad de la misma forma que las categorías para lo cual se añaden una serie de atributos: *createdDate* para almacenar la fecha de creación en el sistema, un *approvalDate* que guardará la fecha en la que dicho rango de usuarios fue aprobado para su uso, un *isDeleted* que será un campo *booleano* con un valor por defecto de *false* y un *deletedDate*, para que, en caso de haber activado el booleano *isDeleted* a *true*, se almacene la fecha en la que se hizo esta acción.

Esta entidad se relacionará con la entidad *Change* con una cardinalidad de 0..*, mientras que *Change* participa en la relación con una cardinalidad de 1. Esto es así ya que un cambio debe tener obligatoriamente un rango de usuarios afectados en caso de no disponibilidad, pero un rango puede que no tenga relación con ningún cambio o puede que tenga muchos cambios asociados.

Entidad Importancia del Número de Usuarios (NUsersImportance)

Modificación tras revisión de cómo modelar el impacto

Se define crear una nueva entidad que almacene la importancia de cada rango de usuarios afectados (*importanceOrder)* y una fecha de validez hasta (*validUntil)*. Las importancias activas no tendrán informado el valor fecha de validez hasta.

Esta entidad tendrá una relación con cardinalidad 1 hacia *NUsers,* la cual participa con cardinalidad 1..* en la relación. Esto es así ya que un rango de usuarios debe tener mínimamente una importancia asignada y puede tener más de una a lo largo del tiempo, pero siempre sólo una vigente. A su vez, la importancia no puede pertenecer a más de un rango de usuarios.

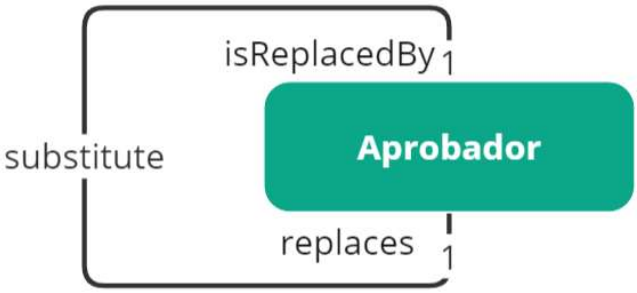
Entidad Aprobador (UserApprover)

Al analizar esta entidad vemos que necesita atributos similares a los de la entidad *Contact*, por lo que replanteamos lo que habíamos definido hasta el momento, dando origen a una nueva entidad *Person,* que será una generalización. Esta almacenará los datos típicos de una entidad de este tipo, mientras que la entidad *UserApprover* se convertirá en una especialización de *Person* y aquí almacenaremos lo necesario para las personas que serán aprobadoras.

Los atributos pues serán *user, password y role.* Este último atributo almacenará el rol de desempeña una persona en el momento de la aprobación. Teniendo en cuenta que los posibles roles están previamente determinados vemos oportuno crear un tipo enumerado *roleType* para almacenar estos valores:

- Technical Manager
- Business Area Manager
- Change Manager
- GCAB Member

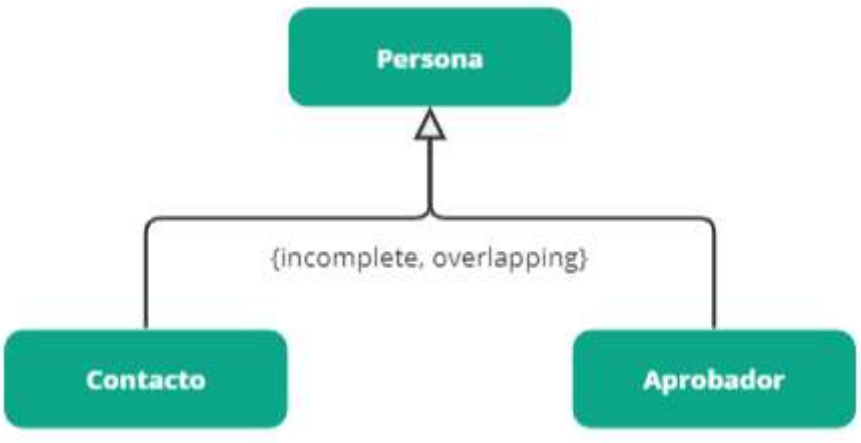
Respecto a las relaciones de esta entidad, se puede identificar claramente en el enunciado la necesidad de realizar una relación recursiva con la propia entidad *UserApprover* para almacenar el aprobador sustituto en caso de ausencia de un aprobador. La cardinalidad de esta relación será 1 a 1 ya que obligatoriamente cada aprobador debe tener un sustituto. Los roles dentro de esta relación serán *isReplacedBy* y *replaces.*



Por último, y quizás un poco más complejo de modelar, surge la necesidad de relacionar cada cambio con un aprobador, pero el hecho de que cada aprobación necesite almacenar datos propios de la aprobación da origen a una nueva entidad *Approval*.
Dado que todo cambio debe tener un aprobador como mínimo, la cardinalidad de la entidad *Change* en esta relación será de 1..*, mientras que en la entidad *UserApproval* será de *0..*.* Pues pueden existir aprobadores que no tengan ningún cambio aprobado o que aprueben más de uno.

Entidad Persona (Person)

Como hemos dicho antes, esta entidad será una generalización que nos permitirá almacenar a las diferentes personas que intervienen en el proceso y a partir de aquí, en función del "papel" que jueguen dentro de la aplicación podrán formar parte de una especialización u otra. Es importante destacar que una misma persona podría llegar hacer el contacto técnico de una aplicación, pero a su vez también podría ser un aprobador, con lo que el tipo de generalización para esta casuística sería solapado u *overlapping*. También se debe tener en cuenta que, aunque en el modelo inicial no se contempla, dejamos el modelo preparado para una futura evolución donde una persona puede no formar parte de ninguna especialización, dando el carácter a esta relación de tipo parcial o *incomplete.*

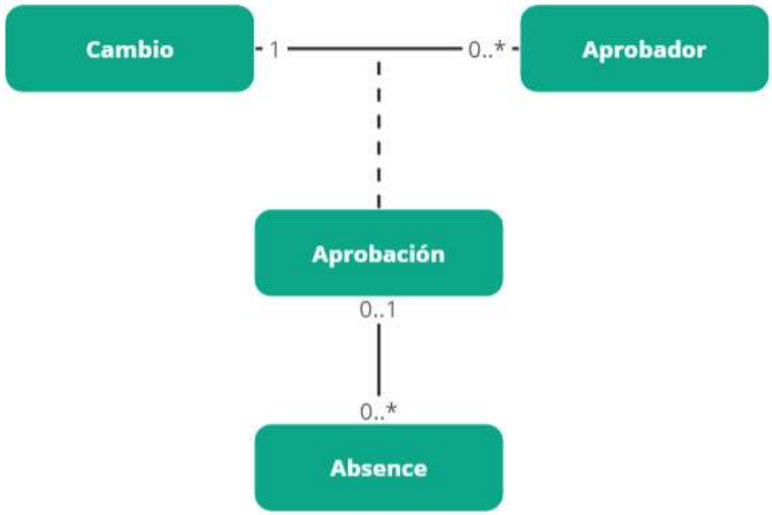


En cuanto a atributos, se establecen los típicos de una entidad como ésta: nombre, apellidos, correo electrónico y número de teléfono.

Entidad Aprobación (Approval)

Esta entidad, producto de una relación asociativa entre *Change* y *UserApprover*, almacenará el rol (atributo *role)* que ocupaba el aprobador en el momento de la aprobación, la fecha de la aprobación (approvalDate), comentarios que serán opcionales y un atributo calculado *bySubsitute* que marcaremos a *true* en caso de que la aprobación haya sido por un sustituto.

Se extrae del enunciado la necesidad de almacenar el motivo de la sustitución en caso de que la aprobación sea realizada por un sustituto, lo que da origen a una nueva entidad *Absence* que tendrá estrecha relación con la entidad *Approval.* Teniendo en cuenta que no todas las aprobaciones tendrán una razón de ausencia se establece una cardinalidad de 0..1 para la entidad *Approvals* y una cardinalidad 0..* para la entidad *Absence.*



Entidad Ausencia (Absence)

Aquí se almacenarán los diferentes motivos de ausencia de un aprobador, por lo que a nivel de atributos contaremos con uno de tipo *string* para almacenar el nombre de la ausencia (*absence)* y otro para almacenar el tipo de ausencia (*type)*. Siendo éste un tipo enumerado ya que los valores que puede contener son:

- Planned
- Not Planned

Esta entidad estará relacionada con la entidad *Approval,* dicha relación ya ha sido explicada anteriormente.

Entidad Ejecución (Execution)

Se pretende recoger aquí cada una de las ejecuciones que se realiza de cada cambio junto con su estado, para lo cual existirán los atributos *executionDate, comments* y *status.* Éste último será un tipo enumerado para poder representar los diferentes estados de una ejecución:

- Correct
- With Incidents
- Rescheduled

Esta entidad estará relacionada con *Change*, y dado que un cambio puede que se plantee pero que por algún motivo nunca llegue a pasarse a producción, la cardinalidad será de 0..*. Por su parte, la entidad *Execution* tendrá una cardinalidad de 1 ya que cada elemento pertenecerá a un cambio en concreto.

Entidad Plan de Acción (ActionPlan)

Según recoge el enunciado, en los casos en los que la ejecución (o el paso a producción) no sea correcto es necesario establecer un plan de acción para solucionar los problemas identificados. Esta entidad almacenará dichos planes de acción y tendrá relación con *Execution*, con una cardinalidad de 0..1 ya que cada ejecución puede tener como máximo 1 plan de acción o puede no tener en caso de que haya sido correcta. Por otro lado, la cardinalidad para *ActionPlan* será de 1 ya que un plan no puede estar asociado a más de una ejecución.

En cuanto a atributos, almacenaremos una *description,* un campo *isValidated* que señalará si dicho plan de acción ha sido validado o no, por defecto su valor será *false*; y sólo si está validado se almacenará también la fecha de validación (*validationDate)*.

Entidad Acción (Action)

En esta entidad almacenaremos cada una de las acciones que deberá llevarse a cabo en cada plan de acción, mediante el atributo *action.* Teniendo en cuenta que no puede existir un plan sin acciones y que cada acción no puede pertenecer a más de un plan, la cardinalidad de la relación respecto a la entidad *ActionPlan* es de 1, mientras que respecto a *Action* es de 1..*

Entidad Auditoría (Audit)

Según recoge el enunciado, existe la posibilidad de que una aplicación sea auditada, por ello se crea la entidad Auditoría en la que crearemos los atributos *detail y auditor* de tipo *string*; y *auditDate* de tipo *date.* En cuanto al atributo *auditor,* inicialmente lo modelamos como un atributo de texto, pero podríamos evolucionar la base de datos para almacenar los auditores o las personas que cumplen con este rol.

Esta entidad se relaciona con *Application,* con una cardinalidad de 0..* ya que pueden existir aplicaciones que no hayan pasado por ninguna auditoría y otras que tengan más de una. Por contrapartida la relación desde el punto de vista de *Audit* tiene una cardinalidad de 1, ya que una auditoría puede corresponder sólo a una aplicación

Entidad Incumplimiento (Non-Compliance)

Con esta entidad se busca almacenar los incumplimientos detectados en una auditoría, de cada uno de ellos se almacenarán los detalles en un atributo de tipo *string*. A nivel de relaciones, cada incumplimiento sólo podrá pertenecer a una auditoría, siendo su cardinalidad de 1; y cada auditoría puede no tener o tener más de un incumplimiento, por lo que su cardinalidad será de 0..*.

Entidad Flujo de Aprobación (ApprovalFlow)

Esta entidad no está especificada en el enunciado como tal, sino que surge a raíz de ver la necesidad de modelar flujos de aprobaciones. Si bien, para la casuística que se nos plantea el flujo está bastante definido y claro, se considera una buena solución plantear los flujos como algo que se pueda definir según necesidades. Lo que buscamos

es poder definir N flujos donde se podrá establecer si para poder realizar una aprobación será necesario una autorización o más de una, y si ellas deberán ser realizadas en orden de aprobación o sin él. También este modelo da pie a que se pueda evolucionar la aplicación y los flujos de aprobación hacia otros escenarios.
Dicho esto, los atributos que consideramos imprescindibles para esta entidad es el nombre del flujo, *flow,* y un tipo de validación, *validation.* Este último atributo será un tipo enumerado con los siguientes valores:

- By Order
- Without Order

A nivel de relaciones, esta entidad tendrá una relación directa con la entidad *Change*, con una cardinalidad de 0..* ya que un flujo existente en la base de datos puede que no haya sido utilizado o puede estarlo en más de un cambio. Por parte de la entidad *Change*, la cardinalidad será de 1, ya que cada cambio obligatoriamente tendrá un flujo de aprobación asignado.

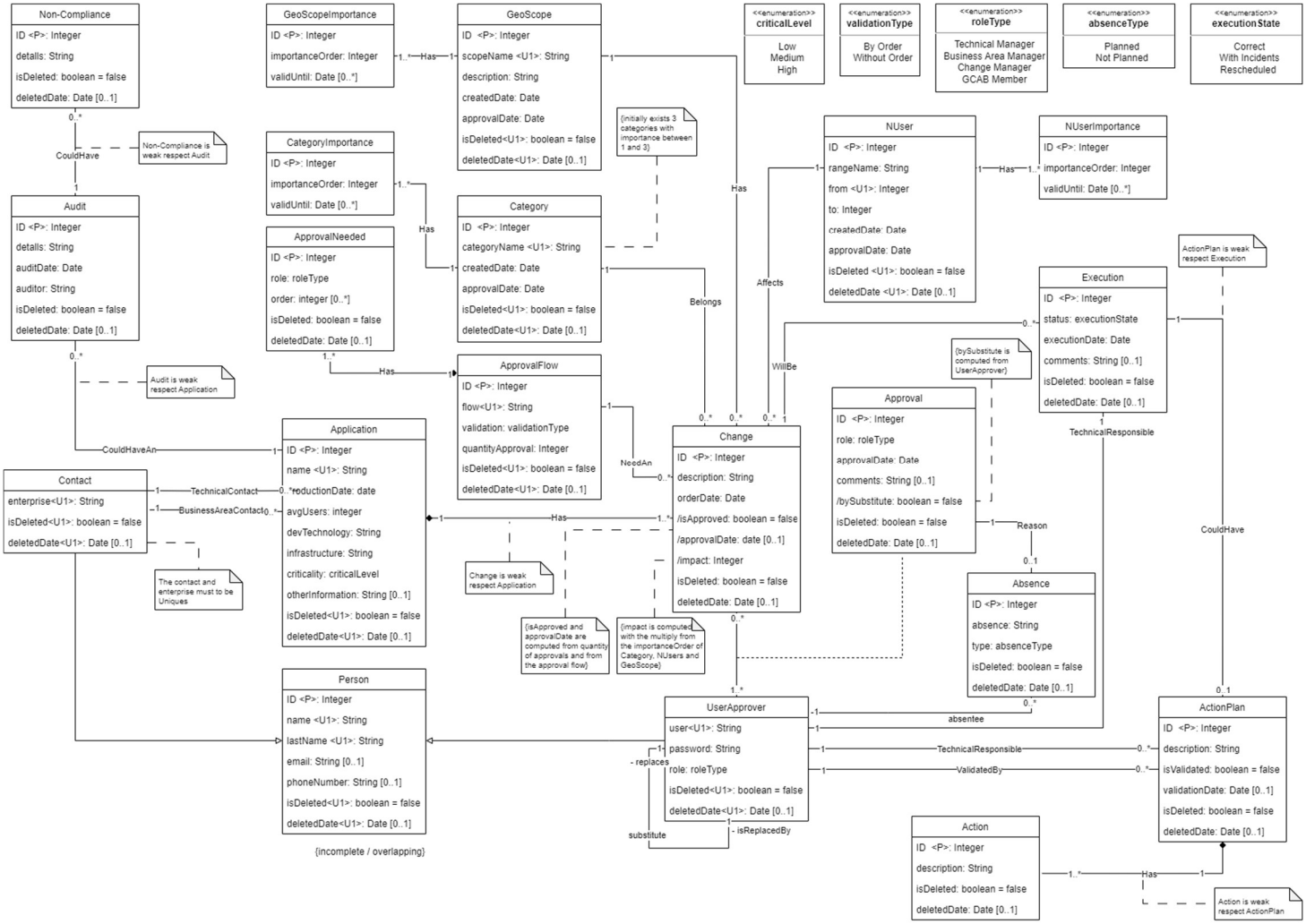
Entidad Aprobación Necesaria (ApprovalNeeded)

Esta entidad surge como complemento de la entidad anterior y servirá para modelar cada una de las aprobaciones que serán necesarias para dar por aprobado un cambio. Para el caso que nos ocupa tendremos un registro por cada rol que es necesario que apruebe cada cambio. Por lo tanto, tendremos el atributo *role* de tipo enumerado *roleType* (ya detallado con anterioridad) y el atributo *order,* que será opcional en función del tipo de validación determinado en el flujo de aprobación.
Esta entidad estará relacionada con *ApprovalFlow* con una cardinalidad de 1 ya que una aprobación puede pertenecer sólo a un flujo de aprobación; mientras que en esta relación la entidad *ApprovalFlow* tendrá una cardinalidad de 1..*, ya que un flujo debe tener como mínimo una aprobación.

Diseño Conceptual Final

Como se comentó en el inicio del proyecto, el diseño conceptual de esta base de datos se ha realizado con la herramienta Draw IO. A continuación, se añade el diseño trabajado, pero para una mejor visualización también se añade un link para su consulta:

*** **NOTA:** *El diseño conceptual detallado a continuación es el resultado final tras haber ido identificando y realizando pequeños cambios en la estructura, producto de la detección y resolución de errores en las diferentes fases del diseño.*



Acceso al diseño online versión inicial:

TFG.drawio

Acceso al diseño online versión final:

TFGv2.drawio

Resolución de dudas antes del modelado

Como se explicó anteriormente, en la fase de análisis surgieron una serie de dudas que necesitaban de resolución para poder realizar el diseño conceptual. Se trasladan a continuación las dudas y se añaden las respuestas obtenidas tras el estudio de cada casuística.

- ¿Cómo modelar las aprobaciones de cada cambio o el flujo de aprobaciones que éstos deben seguir?
  Resolución: *Esta pregunta fue trasladada al profesorado, quien nos ha*

confirmado que el enfoque para modelar un flujo de aprobación sería el más acertado.

- ¿Cómo modelar el bloque relacionado con el impacto en caso de no disponibilidad de la aplicación? Inicialmente se han ido tratando estos aspectos como atributos tanto de la entidad "Aplicación", como de la entidad "Cambio". Sin embargo, se deja pendiente de revisión si este enfoque puede ser mejorado.
  Resolución: *Finalmente, se decide extraer los elementos que influyen en el impacto en caso de no disponibilidad y con cada uno de ellos crear una entidad, de esta manera, podemos establecer un orden de importancia para cada elemento. Por su parte, en la entidad Change, se añade un atribute calculado que hará la multiplicación de los ordenes de importancia de cada elemento, así, los valores más altos tendrán un mayor impacto que los demás.*

- ¿Sería correcto modelar el "rol" de la persona aprobadora como un enumerado? ¿O debería ser una entidad?
  Resolución: *Tras revisarlo, se considera que el rol se puede modelar como un tipo enumerado ya que se trata de elementos que no deberían de variar, son bastante limitados y estáticos. Consideramos esta la mejor decisión ya que afectará positivamente en términos de simplicidad de la base de datos y de eficiencia de esta.*

- ¿Cómo debería gestionar campos calculados de ciertas entidades que dependen de cambios o modificaciones de otras entidades?
  Resolución: *Tras el análisis de estos aspectos se ha considerado la opción de establecer estos controles en la capa de aplicación. Sin embargo, creemos que dada la naturaleza del proyecto sería interesante intentar tener el máximo de posibilidades controladas desde la capa de base de datos, a pesar de que puede tener una pequeña afectación negativa en términos de rendimiento y eficiencia. Esta definición no implica que alguna casuística en concreto decida trasladarse a la capa de aplicación a la hora de realizar el diseño lógico y físico de la base de datos.*
  Para los casos que se controlaran desde la base de datos iremos pivotando entre disparadores y procedimientos almacenados, la decisión se tomará para cada casuística buscando la mejor opción para cada una de ellas en cuanto a eficiencia.

- La entidad "aprobación", ¿debería ser una entidad asociativa o una relación ternaria?
  Resolución: *Tras un segundo análisis, se define la entidad aprobación como una entidad asociativa entre Change y UserApprover. Se descarta totalmente la creación de una entidad ternaria.*

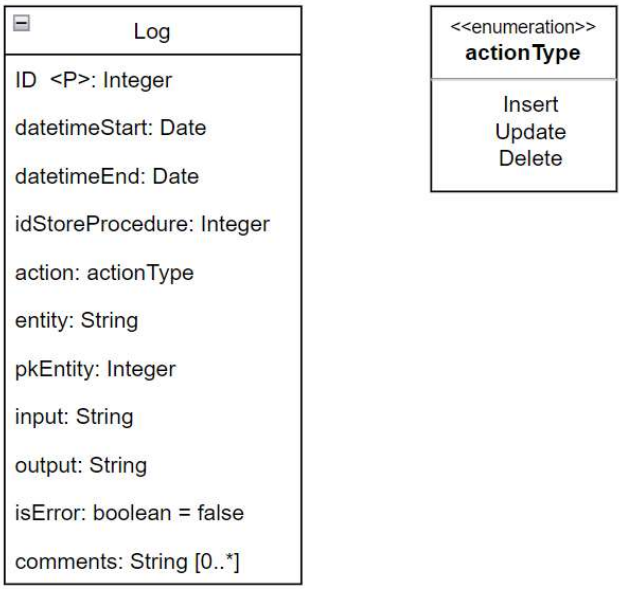
- ¿Cuál sería la mejor manera de abordar todo el bloque de *DataWare House*?
  Resolución: *Tras el análisis para la ejecución de la tarea "TFG-44 Diseño Conceptual DWH" se da respuesta a este apartado.*

Registro de Log

Para el registro de las acciones realizadas en la base de datos se creará una entidad LOG que almacenará cada ejecución de los procedimientos almacenados utilizados para el Alta, Baja y Modificación de elementos.

En esta entidad se almacenará fecha y hora de inicio y fin de cada ejecución (*dateTimeStart y dateTimeEnd),* el identificador del procedimiento almacenado que se está ejecutando (*idStoredProcedure)*, la acción que está realizando (*action*, que será un tipo enumerado con valores *Insert, Update y Delete)*, la entidad sobre la que actúa (*entity*) y el identificador del elemento de la entidad sobre la que se actúa (*pkEntity)*.

Para tener el máximo de detalle se almacenarán también los parámetros de entrada y salida de cada ejecución (*input y output)*, así como un atributo que nos pueda dar visión fácilmente si la ejecución ha generado un error (*isError*) y una opción de comentarios para poder añadir cualquier información relevante (*comments*).



DataWare House

Para establecer el diseño conceptual necesario hemos partido por analizar cada una de las consultas que nos requiere el enunciado, enumerándolas.

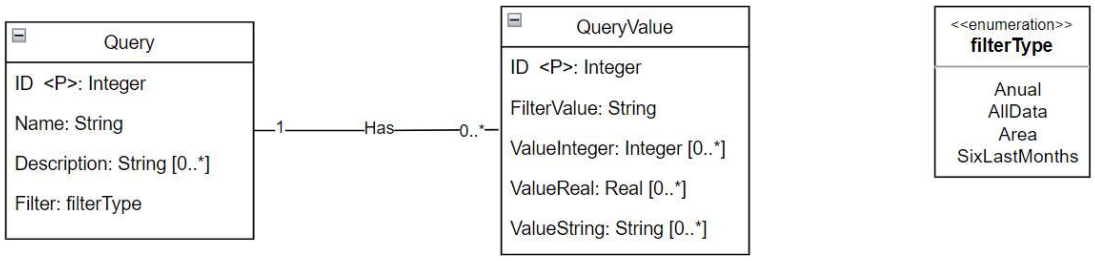
1. En el momento de ejecutar la consulta, número de cambios registrados en el sistema que están en proceso de aprobación.
2. En el último año, número de cambios aprobados que su ejecución no ha sido correcta
3. Teniendo en cuenta toda la información existente en la BD, responsable técnico con más cambios ejecutados de manera correcta
4. Dado un año concreto, porcentaje de acciones definidas para ejecuciones no correctas cerradas en el tiempo inicialmente definido en el plan de acción correspondiente
5. En el año en curso, número total de cambios aprobados en la GCAB
6. En un año concreto, número de cambios que no fueron aprobados y, consecuentemente, no se ejecutaron
7. Dada una región geográfica concreta, porcentaje de cambios que su ejecución se replanificó
8. En los últimos 6 meses, número total de incumplimientos detectados durante las auditorías realizadas
9. En el momento de ejecutar la consulta, número total de planes de acción sobre ejecuciones no correctas que están abiertos

10. En un momento dado, responsable técnico con más cambios en curso
11. 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

Hemos trabajado cada una de ellas para situarlas en un esquema de tabla con ejemplificaciones y hemos obtenido el siguiente resultado:

ID	Metrica	Explicación	Filtro	ValorFiltro	ValorEntero	ValorReal	ValorString
1	CambiosEnAprobación		Actual		5		
2	CambiosNoOK		Anual	2023	2		
3	ReponsableConMasCambiosOK		Actual				Pepe
4	AccionesEjecNoOK		Anual	2023		4,5	
5	AccionesEjecNoOK		Anual	2022		2,3	
6	AccionesEjecNoOK		Anual	2021		1,5	
7	AccionesEjecNoOK		Anual	2020		6,3	
8	CambiosAprobadosGCAB		Anual	2023	5		
9	CambiosNOAprobados		Anual	2023	4		
10	CambiosNOAprobados		Anual	2022	3		
11	CambiosNOAprobados		Anual	2021	2		
12	CambiosEjecReplanificada		Region	Centro		4,2	
13	CambiosEjecReplanificada		Region	Norte		3,4	
14	CambiosEjecReplanificada		Region	Mundial		9,5	
15	Incumplimientos		6Meses		15		
16	PlanesAbiertosEjecNoOK		Actual		20		
17	ReponsableConCambiosEnCurso		Actual		5		
18	AprobadorMasSustituidoNoPlanif		Actual				Joan

Con este punto de partida, hemos analizado cuáles serían las tablas necesarias y sus atributos, quedando de la siguiente manera:



5.2. Diseño Lógico

En este apartado generaremos un esquema lógico a partir del diseño conceptual trabajado hasta este momento. Cabe destacar que, al estar trabajando sobre una base de datos relacional, la nomenclatura que recibe nuestro diseño es de modelo lógico relacional o modelo relacional.

A partir de este proceso, podremos identificar posibles fallos cometidos en el diseño conceptual, además de identificar todas las tablas que necesitaremos a la hora de realizar el diseño físico como también aquellas columnas que formarán parte de las relaciones con otras tablas de la base de datos. En otras situaciones, también durante esta fase podríamos identificar tablas que no existen en primera instancia y que son producto de relaciones *varios a varios,* para el caso en cuestión no aplica ya que no tenemos relaciones de este tipo.

**** **NOTA:** El diseño lógico detallado a continuación es el resultado final tras haber ido identificando y realizando pequeños cambios en la estructura, producto de la detección y resolución de errores.*

Explicaremos a continuación el modelo lógico por partes para intentar simplificar la lectura:

Cambio e impacto de cada cambio

```
Change      (id, description, orderDate, isApproved,
            approvalDate, impact, idCategory, idGeoScope,
            idNUser, idApprovalFlow, idApplication, isdeleted,
            deletedDate)
            {idApplication} is foreing key to Application
            {idCategory} is foreing key to Category
            {idGeoScope} is foreing key to GeoScope
            {idNUser} is foreing key to NUser
            {idApprovalFlow} is foreing key to ApprovalFlow

Category    (id, categoryName, createdDate, approvalDate,
            isDeleted, deletedDate)

CategoryImportance (id, importanceOrder, validUntil, idCategory)
            {idCategory} is foreing key to Category

GeoScope    (id, scopeName, description, createdDate,
            approvalDate, isDeleted, deletedDate)

GeoScopeImportance (id, importanceOrder, validUntil, idGeoScope)
            {idGeoScope} is foreing key to GeoScope

NUser       (id, rangeName, from, to, createdDate, approvalDate,
            isDeleted, deletedDate)

NUserImportance (id, importanceOrder, validUntil, idNUser)
            {idNUser} is foreing key to NUser
```

Flujos de aprobaciones

```
ApprovalFlow (id, flow, validation, quantityApproval, isdeleted,
            deletedDate)

ApprovalNeeded (id, role, order, idApprovalFlow, isdeleted,
deletedDate)
            {idApprovalFlow} is foreing key to Approval Flow
```

Aplicación

Application (id, **name**, productionDate, avgUsers, devTechnology, infraestructure, criticality, otherInformation, idTechnicalContact, idBusinessAreaContact, **isdeleted**, deletedDate)
{idTechnicalContact} is foreing key to Contact
{idBusinessAreaContact} is foreing key to Contact

Auditorías

Audit (id, **detalls**, **auditDate**, **auditor**, **idApplication**, **isdeleted**, deletedDate)
{idApplication} is foreing key to Application

Non-Compliance (id, **detalls**, **idAudit**, **isdeleted**, deletedDate)
{idAudit} is foreing key to Audit

Personas

Person (id, **name**, **lastName**, **email**, **phoneNumber**, **isdeleted**, deletedDate)

Contact (id, **enterprise**, **idPerson**, **isdeleted**, deletedDate)
{idPerson} is foreing key to Person

UserApprover (id, **user**, **password**, **role**, **idPerson**, **idSubstitute**, **isdeleted**, deletedDate)
{idPerson} is foreing key to Person
{idSubstitute} is foreing key to UserApprover

Ejecuciones

Execution (id, **status**, **executionDate**, comments, **idChange**, idTechnicalResponsible, **isdeleted**, deletedDate)
{idChange} is foreing key to Change
{idTechnicalResponsible} is foreing key to UserApproval

ActionPlan (id, **description**, **isValidated**, validationDate, **idExecution**, **idTechnicalResponsible**, **idValidatedBy**, **isdeleted**, deletedDate)
{idExecution} is foreing key to Execution
{idTechnicalResponsible} is foreing key to UserApproval
{idValidatedBy} is foreing key to UserApproval

Action (id, **description**, **idActionPlan**, **isdeleted**, deletedDate)
{idActionPlan} is foreing key to ActionPlan

Aprobaciones

```
Approval    (id, role, approvalDate, comments, bySubstitute,
            idChange, idUserApproval, isdeleted, deletedDate)
            {idChange} is foreing key to Change
            {idUserApproval} is foreing key to UserApproval

Absence     (id, absence, type, idApproval, idabsentee,
            isdeleted, deletedDate)
            {idApproval} is foreing key to Approval
            {idAbsentee} is foreing key to UserApprover
```

Logs

```
Log   (id, dateTimeStart, dateTimeEnd, idStoredProcedure,
      action, entity, pkEntity, input, output, isError, comments)
```

DataWareHouse

```
Query       (id, name, description, filter)

QueryValue  (id, filterValue, valueInteger, valueReal,
            valueString, idQuery)
            {idQuery} is foreing key to Query
```

Errores identificados del diseño conceptual

Al realizar el diseño lógico hemos encontrado los siguientes errores en el diseño conceptual, que ya han sido corregidos.

- La cardinalidad entre las entidades Approval y Absence era incorrecta. Lo correcto es que la entidad Approval tenga una cardinalidad 1 ya que una ausencia debe estar asociada obligatoriamente con una aprobación, mientras que la entidad Absence tiene una cardinalidad 0..1 ya que una aprobación puede que tenga 1 o ninguna ausencia.

5.3. Diseño Físico

En esta fase nos centraremos en la transformación de nuestro diseño lógico en lo que será el diseño final aplicado en nuestra base de datos Oracle SQL. Para ello, el primer problema con el que nos encontramos a la hora de realizar el diseño físico es definir los tipos de datos más adecuados a utilizar.

En nuestro diseño conceptual hemos definido los tipos:

- Integer
- Integer Autoincremental para los identificadores de nuestras tablas.
- String
- Date

- Boolean
- Real
- Tipos Enumerados

Teniendo en cuenta que estamos trabajando con la versión Oracle Database 21c Express mencionaremos a continuación la manera en la que abordamos la implementación de cada tipo de dato:

1. **Integer:** serán tratados como campos de tipo NUMBER con precisión 8 y escala 0, para evitar el uso de valores decimales.

2. **Integer autonumérico:** Nos apoyaremos en una funcionalidad que nos permite crear campos identificadores autonuméricos: NUMBER GENERATED ALWAYS AS IDENTITY. Mediante esta instrucción crearemos campos de tipo numéricos cuyos valores no son necesario de informar a la hora de crear un registro, ya que el propio gestor de base de datos irá asignando valores incrementales de manera automática.

3. **String:** los trataremos como campos de tipo CHAR o VARCHAR con un tamaño variable, en función del contenido que se espera almacenar.

4. **Date:** Los gestionaremos de dos formas, por un lado usaremos el tipo DATE para los casos donde sólo nos interesa almacenar fechas. Por otro lado, utilizaremos el tipo de datos TIMESTAMP para las casuística en las que necesitamos almacenar fecha y hora completa.

5. **Boolean:** Al no existir como un tipo nativo de Oracle SQL, lo sustituiremos mediante dos acciones. La primera consiste en crear estos campos como valores NUMBER de precisión 1. La segunda consiste en añadir por cada campo de este tipo, una restricción CHECK que sólo permita la inserción de valores 0 y 1.

6. **Real:** Una vez más, estos campos serán de tipo NUMBER con la precisión necesaria para cada caso, pero con un valor de ESCALA de 1, 2 o 3 decimales, según corresponda.

7. **Tipos enumerados:** Dada una restricción que Oracle DataBase tiene en cuanto a la gestión de tipos de datos enumerados de forma nativa, nos vemos obligados a recurrir a otras alternativas de diseño que no suelen ser tan eficientes para modelar este tipo de datos, algunas de ellas son:
   - Utilizar CONSTRAINT de tipo CHECK
   - Crear una TABLA por cada TYPE ENUM
   - Utilizar funcionalidad de PAQUETES de Oracle

   Tras analizarlas, quizás la más lógica y simple podría ser utilizar CONSTRAINT de tipo CHECK pero esto nos lleva a que si tuviésemos que utilizar el mismo ENUM en otras tablas, ante una actualización, tendríamos que acordarnos de modificar todas las tablas que hacen referencia a este ENUM. También en caso de crear nuevos valores, implica modificar todas las tablas que lo utilicen.

   Visto esta problemática, y pensando en futuras evoluciones del sistema, como también en el hecho de tener la información centralizada, preferimos optar por crear una TABLA por cada TYPE ENUM definido inicialmente en el diseño conceptual.

Otro tema importante para resolver en el diseño físico es la implementación de los **campos calculados**. Estos, al tener dependencias de otros campos de la misma tabla o incluso de otras, será necesario gestionarlo mediante disparadores.
Se detalla a continuación el diseño físico de las tablas, siguiendo el mismo criterio de agrupamiento planteado para el diseño lógico.

A su vez, en el fichero "01. CreacionTablas.sql" se deja constancia del código PL/SQL que se utiliza para crear las tablas en el gestor de BBDD.

**** **NOTA:** El diseño físico detallado a continuación es el resultado final tras haber ido identificando y realizando pequeños cambios en la estructura, producto de la detección y resolución de errores. Se puede encontrar el script de creación de las tablas con los cambios actualizados en el fichero "01. CreacionTablas_v2.sql".*

Cambio e impacto de cada cambio

Tabla	Change			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
Description	Varchar2 (200)	NO		
OrderDate	Date	NO		
IsApproved	Number(38,0)	NO		CHECK Boolean
ApprovalDate	Date	SI		
Impact	Number(38,0)	NO		Trigger
IDCategory	Number(38,0)	NO		FK Category
IDGeoScope	Number(38,0)	NO		FK GeoScope
IDNUser	Number(38,0)	NO		FK Nuser
IDApprovalFlow	Number(38,0)	NO		FK ApprovalFlow
IDApplication	Number(38,0)	NO		FK Application
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

Tabla	Category			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
CategoryName	Varchar2 (200)	NO		
CreatedDate	Timestamp	NO	Fecha Actual	
ApprovalDate	Date	NO		
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

Tabla	CategoryImportance			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
ImportanceOrder	Number(38,0)	NO		
ValidUntil	Date	SI		
IDCategory	Number(38,0)	NO		FK Category

Tabla	GeoScope			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
ScopeName	Varchar2 (200)	NO		
Description	Varchar2 (200)	SI		
CreatedDate	Timestamp	NO	Fecha Actual	
ApprovalDate	Date	NO		
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

Tabla	GeoScopeImportance			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
ImportanceOrder	Number(38,0)	NO		
ValidUntil	Date	SI		
IDGeoScope	Number(38,0)	NO		FK GeoScope

Tabla	Nuser			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
ScopeName	Varchar2 (200)	NO		
From	Number(38,0)	NO		UK 1
To	Number(38,0)	NO		UK 1
CreatedDate	Timestamp	NO	Fecha Actual	
ApprovalDate	Date	NO		
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

Tabla	NUserImportance			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
ImportanceOrder	Number(38,0)	NO		
ValidUntil	Date	SI		
IDNUser	Number(38,0)	NO		FK Nuser

Flujos de Aprobaciones

Tabla	ApprovalFlow			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico

Flow	Varchar2 (200)	NO		
IDValidationType	Number(38,0)	NO		FK RoleType
QuantityApproval	Number(38,0)	NO	0	
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

Tabla	ApprovalNeeded			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
Order	Varchar2 (200)	SI		
IDRoleType	Number(38,0)	NO		FK RoleType
IDApprovalFlow	Number(38,0)	NO		FK ApprovalFlow
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

Aplicación

Tabla	Application			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
Name	Varchar2 (100)	NO		
ProductionDate	Number(38,0)	NO		
AVGUsers	Number(38,0)	SI		
DevTechnology	Varchar2 (200)	SI		
Infrastructure	Varchar2 (200)	SI		
IDCriticalLevel	Number(38,0)	NO		FK CriticalLevel
OtherInformation	Varchar2 (200)	SI		
IDTechnicalContact	Number(38,0)	NO		FK UserApprover
IDBusinessAreaContact	Number(38,0)	NO		FK UserApprover
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

Auditorías

Tabla	Audit			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
Detalls	Varchar2 (200)	NO		
AuditDate	Date	NO		
Auditor	Varchar2 (100)	NO		
IDApplication	Number(38,0)	NO		FK Application
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

Tabla	NonCompliance			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
Detalls	Varchar2 (200)	NO		
IDAudit	Number(38,0)	NO		FK Audit
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

Personas

Tabla	Person			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
Name	Varchar2 (200)	NO		
LastName	Varchar2 (200)	NO		
Email	Varchar2 (200)	NO		
PhoneNumber	Varchar2 (200)	NO		
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

Tabla	Contact			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
Enterprise	Varchar2 (200)	NO		
IDPerson	Number(38,0)	NO		ID Person
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

Tabla	UserApprover			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
User	Varchar2 (200)	NO		
Password	Varchar2 (200)	NO		
IDRoleType	Number(38,0)	NO		ID RoleType
IDPerson	Number(38,0)	NO		ID Person
IDSubstitute	Number(38,0)	NO		ID UserApprover
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

Ejecuciones

Tabla	Execution			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
IDExecutionState	Number(38,0)	NO		FK ExecutionState
ExecutionDate	Date	NO		
Comments	Varchar2 (200)	SI		
IDChange	Number(38,0)	NO		FK Change
IDTechnicalResponsible	Number(38,0)	NO		FK UserApprover
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

Tabla	ActionPlan			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
Description	Varchar2 (200)	NO		
IsValidated	Number(38,0)	NO		CHECK Boolean
ValidationDate	Date	SI		
IDExecution	Number(38,0)	NO		FK Execution
IDTechnicalResponsible	Number(38,0)	NO		FK UserApprover
IDValidatedBy	Number(38,0)	NO		FK UserApprover
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

Tabla	Action			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
Description	Varchar2 (200)	NO		
IDActionPlan	Number(38,0)	NO		FK ActionPlan.ID
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

Aprobaciones

Tabla	Approval			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
IDRoleType	Number(38,0)	NO		FK RoleType
ApprovalDate	Date	NO		
Comments	Varchar2 (200)	NO		
BySubstitute	Number(38,0)	NO		CHECK Boolean
IDChange	Number(38,0)	NO		FK Change
IDUserApproval	Number(38,0)	NO		FK UserApprover
IsDeleted	Number(38,0)	NO		CHECK Boolean

DeletedDate	Date	SI		

Tabla	Absence			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
Absence	Varchar2 (100)	NO		
IDAbsenseType	Number(38,0)	NO		FK AbsenseType.ID
IDApproval	Number(38,0)	NO		FK Approval.ID
IDAbsentee	Number(38,0)	NO		FK UserApprover.ID
IsDeleted	Number(38,0)	NO		CHECK Boolean
DeletedDate	Date	SI		

TiposEnumerados

Tabla	EnumAbsenceType			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
AbsenceType	Varchar2 (200)	NO		

Tabla	EnumCriticalLevel			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
CriticalLevel	Varchar2 (200)	NO		

Tabla	EnumExecutionState			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
ExecutionState	Varchar2 (200)	NO		

Tabla	EnumRoleType			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
RoleType	Varchar2 (200)	NO		

Tabla	EnumValidationType			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
ValidationType	Varchar2 (200)	NO		

Logs

Tabla	Logs			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
DateTimeStart	DateTime	NO		FK RoleType
DateTimeEnd	DateTime	NO		
IDStoredProcedure	Varchar2 (200)	NO		
Action	Varchar2 (100)	NO		CHECK I, U, D
Entity	Varchar2 (200)	NO		FK Change
PKEntity	Varchar2 (200)	NO		FK UserApprover
Input	Varchar2 (200)	NO		
Output	Varchar2 (200)	NO		
IsError	Varchar2 (200)	NO	0	CHECK Boolean
Comments	Varchar2 (200)	SI		

DataWareHouse

Tabla	EnumFilterType			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
FilterType	Varchar2 (200)	NO		

Tabla	Query			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
Name	Varchar2 (200)	NO		
Description	Varchar2 (200)	NO		
IDFilterType	Varchar2 (200)	NO		FK FilterType

Tabla	QueryValue			
Nombre Columna	**TipoDato**	**Permite Null**	**Valor por defecto**	**Observaciones**
ID	Number(38,0)	NO		PK Autonumérico
FilterValue	Varchar2 (200)	NO		
ValueInteger	Number(38,0)	SI		
ValueReal	Number(38,0)	SI		
ValueString	Varchar2 (200)	SI		
IDQuery	Number(38,0)	NO		FK Query

Correcciones realizadas al realizar el diseño físico:

Como ocurrió a la hora de realizar el diseño lógico, en esta fase también hemos identificado algunos errores cometidos en las fases anteriores del diseño. Los enumeramos a continuación:

- Los campos ScopeName (tabla GeoScope), CategoryName (tabla Category), From y To (tabla NUser) deben establecerse como campos únicos para evitar que hayan dos registros que se utilicen con el mismo objetivo.
- La tabla NUser debería almacenar valores FROM y TO como valores enteros pero que representen porcentajes de usuarios, de esta manera podríamos clasificar correctamente las aplicaciones, independientemente de si tienen pocos o muchos usuarios. Estos valores serán calculados manualmente a la hora de insertar un registro utilizando como referencia el campo AVGUSER de la tabla APPLICATION.

Tras finalizar con el diseño físico, hacer una primera inserción de datos y una revisión detectamos que:

- Es necesario añadir una columna que nos hemos olvidado en la tabla APPROVALNEEDED. Se corrige también la propiedad ORDER ya que se plantea crear como un campo NOT NULL y realmente debería permitir NULL.

- Se debe corregir la propiedad IDSUBSTITUTE de la tabla USERAPPROVER para que permita valores NULL ya que se había planteado incorrectamente.

- Se identifica que el campo USER de la tabla USERAPPROVER debe ser tipo UNIQUE.

- Para insertar registros en la tabla CHANGE vemos la necesidad de trabajar correctamente el disparador que calculará el valor del campo IMPACT. Realizaremos unas pruebas de inserción de datos obviando este campo para lo cual lo estableceremos provisoriamente como NULL.

  ALTER TABLE CHANGE
  MODIFY IMPACT NULL

  Ver Código del Trigger empleado en Anexo "11.4 Trigger Calcular Impacto"
  De esta casuística, nos vemos obligados a crear una nueva tarea para la PEC4 que consistirá en crear los disparadores necesarios para todos los campos calculados de la base de datos. La nueva tarea es la "TFG-45 Creación Triggers Campos Calculados"

5.4. Comprobación Formas Normales

Para verificar que nuestro diseño está *normalizado* según los criterios de las formas normales, hemos hecho análisis de cada una de ellas:

- **Primera Forma Normal:** Todos los valores introducidos en cada una de las celdas de nuestras tablas son atómicos, indivisibles y del mismo tipo de dato entre columnas.

- **Segunda Forma Normal:** Podemos afirmar que, en nuestro diseño, todas las columnas no clave de una tabla dependen de la clave primaria y no tenemos en ningún caso una columna que no tenga este comportamiento.

- **Tercera Forma Normal:** No existen dependencias transitivas en nuestras tablas, toda aquella que tabla susceptible de tenerla, como puede ser el caso de la tabla UserApprover y su campo Subsitute, se separan en tablas diferentes haciendo una relación hacia ellas, o una relación recursiva en el ejemplo mencionado.

- **Forma Normal Boyce-Codd:** Teniendo en cuenta que cumplimos la tercera forma normal y que cada valor depende de una clave, podemos afirmar que cumplimos con esta forma normal.

- **Cuarta Forma Normal:** Teniendo en cuenta que no tenemos columnas con multivalores en nuestro diseño, podemos afirmar que cumplimos con la cuarta forma normal.

6. Implementación

La fase de implementación la hemos separado en dos apartados, el primero consiste en la ejecución de los scripts de creación de la estructura de la base de datos: tablas, restricciones y triggers. Para ello ejecutamos el script "01. CreaciónTablas.sql" y comprobamos la ausencia de errores.

01. CreacionTablas.sql

Tras tener la estructura base, se insertan valores de prueba en las tablas de configuración del sistema y, adicionalmente, se insertan algunos registros de ejemplos para poder comprobar que la estructura definida para las tablas es la esperada. En este caso, ejecutamos el script "02. InserciónDatos.sql" y comprobamos la ausencia de errores.

02. InsercionDatos.sql

El segundo apartado de la implementación consistirá en la preparación de los procedimientos almacenados para gestionar las Altas, Bajas y Modificaciones de los diferentes elementos.

Para el desarrollo de los procedimientos, primero hemos hecho una revisión de nuestras tablas y así poder definir los procedimientos necesarios. Contamos con un total de 29 tablas segregados entre ENUMERADOS, CONFIGURACION, DATOS, LOG y DWH. Los procedimientos se crearán inicialmente sólo para las tablas de CONFIGURACION y DATOS, por lo que nos centraremos en un total de 20 tablas.

TABLAS	TIPO
CATEGORY	CONFIGURACION
CATEGORYIMPORTANCE	CONFIGURACION
GEOSCOPE	CONFIGURACION
GEOSCOPEIMPORTANCE	CONFIGURACION
NUSER	CONFIGURACION
NUSERIMPORTANCE	CONFIGURACION
APPROVALFLOW	CONFIGURACION
APPROVALNEEDED	CONFIGURACION
EXECUTION	DATOS
ACTIONPLAN	DATOS
ACTION	DATOS
APPROVAL	DATOS
ABSENCE	DATOS
PERSON	DATOS

CONTACT	DATOS
USERAPPROVER	DATOS
APPLICATION	DATOS
CHANGE	DATOS
AUDIT	DATOS
NONCOMPLIANCE	DATOS
DWH_QUERY	DWH
DWH_QUERYVALUE	DWH
ENUM_FILTERTYPE	DWH
ENUM_VALIDATIONTYPE	ENUMERADOS
ENUM_CRITICALLEVEL	ENUMERADOS
ENUM_ROLETYPE	ENUMERADOS
ENUM_ABSENCETYPE	ENUMERADOS
ENUM_EXECUTIONSTATE	ENUMERADOS
LOG	LOG

Por norma general, se crearán 3 procedimientos por cada tabla, uno para el Alta, otro para la Baja y otro para la MODIFICACIÓN de elementos. Sin embargo, hay algunas tablas que, dada la particularidad de sus relaciones puede que no tengan alguno de ellos o que se gestione de manera conjunta con el procedimiento de otra tabla.

Es el caso de las tablas CATEGORIA, GEOSCOPE y NUSER que con sus procedimientos almacenados gestionamos también los datos de CATEGORIAIMPORTANCE, GESCOPEIMPORTANCE y NUSERIMPORTANCE; o el caso de la tabla ACTIONPLAN en el cual gestionamos altas y bajas de su tabla relacionada EXECUTION y sólo necesitamos el procedimiento para actualizaciones.
Tras esta revisión nos surge un listado de 46 procedimientos a crear.

6.1. Procedimientos Almacenados. Estructura.

Cabe mencionar que previamente a la realización de los procedimientos almacenados, se ha definido una “plantilla” para mantener un criterio uniforme entre todos ellos. El detalle definido es el siguiente:

1. **Nomenclatura:** Todos los procedimientos llevarán el nombre compuesto por “SP_” más el nombre de la tabla en cuestión más la acción que realizará (ADD, DELETE o UPDATE). Un ejemplo sería: SP_APPLICATION_ADD.

2. **Criterio de variables:** Se ha definido que todas las variables que se utilizarán en el procedimiento seguirán el mismo criterio:
    a. Comenzarán por “p_” todas las variables que son parámetros de entrada del procedimiento.
    b. Comenzarán por “v_” todas aquellas que son necesarias internamente para la transacción
    c. Comenzarán por “v_” y serán escritas en mayúsculas en su totalidad aquellas que se utilizan para el registro de LOGs.
    d. Se define la variable “RSP” como variable de salida de los procedimientos almacenados.

3. **Orden dentro del procedimiento:** Para intentar mantener un criterio unificado y simplificar el entendimiento del código, se ha definido una serie de bloques identificados por comentarios.
    a. Definición de variables para la transacción
    b. Definición de variables para el registro del log
    c. Gestión de información para el registro del log
    d. Validaciones que controlan la lógica de negocio.
    e. Transacción.
    f. Registro de LOG OK
    g. Rollback y registro de LOG KO.

6.2. Procedimientos Almacenados. ABM.

A continuación, se detallará el listado de Procedimientos Almacenados necesarios junto con sus respectivas explicaciones y definición de pruebas.

El script que ejecuta todos estos procedimientos los podéis encontrar en el siguiente link:

ProcedimientosAlmacenados

Listado Resumen de Procedimientos Almacenados:

Tabla	Tipo	Acción	StoredProcedure
APPROVALFLOW	CONFIGURACION	ADD	SP_APPROVALFLOW_ADD
APPROVALFLOW	CONFIGURACION	DELETE	SP_APPROVALFLOW_DELETE
APPROVALFLOW	CONFIGURACION	UPDATE	SP_APPROVALFLOW_UPDATE
APPROVALNEEDED	CONFIGURACION	ADD	SP_APPROVALNEEDED_ADD
APPROVALNEEDED	CONFIGURACION	DELETE	SP_APPROVALNEEDED_DELETE
APPROVALNEEDED	CONFIGURACION	UPDATE	SP_APPROVALNEEDED_UPDATE
CATEGORY	CONFIGURACION	ADD	SP_CATEGORY_ADD
CATEGORY	CONFIGURACION	DELETE	SP_CATEGORY_DELETE
CATEGORY	CONFIGURACION	UPDATE	SP_CATEGORY_UPDATE
GEOSCOPE	CONFIGURACION	ADD	SP_GEOSCOPE_ADD
GEOSCOPE	CONFIGURACION	DELETE	SP_GEOSCOPE_DELETE
GEOSCOPE	CONFIGURACION	UPDATE	SP_GEOSCOPE_UPDATE
NUSER	CONFIGURACION	ADD	SP_NUSER_ADD
NUSER	CONFIGURACION	DELETE	SP_NUSER_DELETE
NUSER	CONFIGURACION	UPDATE	SP_NUSER_UPDATE
ABSENCE	DATOS	UPDATE	SP_ABSENCE_UPDATE
ACTION	DATOS	ADD	SP_ACTION_ADD
ACTION	DATOS	DELETE	SP_ACTION_DELETE
ACTION	DATOS	UPDATE	SP_ACTION_UPDATE
ACTIONPLAN	DATOS	UPDATE	SP_ACTIONPLAN_UPDATE
APPLICATION	DATOS	ADD	SP_APPLICATION_ADD
APPLICATION	DATOS	DELETE	SP_APPLICATION_DELETE
APPLICATION	DATOS	UPDATE	SP_APPLICATION_UPDATE
APPROVAL	DATOS	ADD	SP_APPROVAL_ADD

APPROVAL	DATOS	DELETE	SP_APPROVAL_DELETE
AUDIT	DATOS	ADD	SP_AUDIT_ADD
AUDIT	DATOS	DELETE	SP_AUDIT_DELETE
AUDIT	DATOS	UPDATE	SP_AUDIT_UPDATE
CHANGE	DATOS	ADD	SP_CHANGE_ADD
CHANGE	DATOS	DELETE	SP_CHANGE_DELETE
CHANGE	DATOS	UPDATE	SP_CHANGE_UPDATE
CONTACT	DATOS	ADD	SP_CONTACT_ADD
CONTACT	DATOS	DELETE	SP_CONTACT_DELETE
CONTACT	DATOS	UPDATE	SP_CONTACT_UPDATE
EXECUTION	DATOS	ADD	SP_EXECUTION_ADD
EXECUTION	DATOS	DELETE	SP_EXECUTION_DELETE
EXECUTION	DATOS	UPDATE	SP_EXECUTION_UPDATE
NONCOMPLIANCE	DATOS	ADD	SP_NONCOMPLIANCE_ADD
NONCOMPLIANCE	DATOS	DELETE	SP_NONCOMPLIANCE_DELETE
NONCOMPLIANCE	DATOS	UPDATE	SP_NONCOMPLIANCE_UPDATE
PERSON	DATOS	ADD	SP_PERSON_ADD
PERSON	DATOS	DELETE	SP_PERSON_DELETE
PERSON	DATOS	UPDATE	SP_PERSON_UPDATE
USERAPPROVER	DATOS	ADD	SP_USERAPPROVER_ADD
USERAPPROVER	DATOS	DELETE	SP_USERAPPROVER_DELETE
USERAPPROVER	DATOS	UPDATE	SP_USERAPPROVER_UPDATE

Detalle de Procedimientos Almacenados:

Tabla	CATEGORY
Procedimiento	**Código: SP0001 - SP_CATEGORY_ADD**
Descripción	Procedimiento que se ocupa de dar de Alta nuevas categorías y gestionar su importancia
Parámetros de entrada	p_CategoryName IN VARCHAR2, p_ApprovalDate IN DATE, p_ImportanceOrder IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0001	No informar un campo obligatorio o informarlo como NULL
PR0002	Informar un Order ya existente
PR0003	Añadir un elemento con un nombre ya existente
PR0004	Informar todos los datos requeridos

Tabla	CATEGORY
Procedimiento	**Código: SP0002 - SP_CATEGORY_DELETE**
Descripción	Procedimiento que da de baja lógica una categoría y su importancia
Parámetros de entrada	p_CategoryID IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**

PR0005	No informar un campo obligatorio o informarlo como NULL
PR0006	Informar todos los datos requeridos
PR0007	Informar un ID que ya ha sido eliminado
PR0008	Informar un ID inexistente en BBDD

Tabla	**CATEGORY**
Procedimiento	**Código: SP0003 - SP_CATEGORY_UPDATE**
Descripción	Procedimiento que modifica la configuración de una categoría.
Parámetros de entrada	p_CategoryID IN NUMBER, p_CategoryName IN VARCHAR2, p_ApprovalDate IN DATE, p_ImportanceOrder IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0009	No informar un campo obligatorio o informarlo como NULL
PR0010	Informar un ID que ya ha sido eliminado
PR0011	Informar un ID inexistente en BBDD
PR0012	Informar un Order ya existente
PR0013	Informar un nombre ya existente
PR0014	Informar todos los datos requeridos

Tabla	**GEOSCOPE**
Procedimiento	**Código: SP0004 - SP_GEOSCOPE_ADD**
Descripción	Procedimiento que se ocupa de dar de alta nuevo alcance geográfico y gestionar su importancia.
Parámetros de entrada	p_GeoScopeName IN VARCHAR2, p_Description IN VARCHAR2, p_ApprovalDate IN DATE, p_ImportanceOrder IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0015	No informar un campo obligatorio o informarlo como NULL
PR0016	Informar un Order ya existente
PR0017	Añadir un elemento con un nombre ya existente
PR0018	Informar todos los datos requeridos

Tabla	**GEOSCOPE**
Procedimiento	**Código: SP0005 - SP_GEOSCOPE_DELETE**
Descripción	Procedimiento que da de baja lógica un alcance geográfico.
Parámetros de entrada	p_GeoScopeID IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0019	No informar un campo obligatorio o informarlo como NULL
PR0020	Informar un ID que ya ha sido eliminado

PR0021	Informar un ID inexistente en BBDD
PR0022	Informar todos los datos requeridos

Tabla	**GEOSCOPE**
Procedimiento	**Código: SP0006 - SP_GEOSCOPE_UPDATE**
Descripción	Procedimiento que modifica la configuración de un alcance geográfico.
Parámetros de entrada	p_GeoScopeID IN NUMBER, p_GeoScopeName IN VARCHAR2, p_Description IN VARCHAR2, p_ApprovalDate IN DATE, p_ImportanceOrder IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0023	No informar un campo obligatorio o informarlo como NULL
PR0024	Informar un ID que ya ha sido eliminado
PR0025	Informar un ID inexistente en BBDD
PR0026	Informar un Order ya existente
PR0027	Informar un nombre ya existente
PR0028	Informar todos los datos requeridos

Tabla	**NUSER**
Procedimiento	**Código: SP0007 - SP_NUSER_ADD**
Descripción	Procedimiento que se ocupa de dar de alta un nuevo rango de usuarios y gestionar su importancia.
Parámetros de entrada	p_RangeName IN VARCHAR2, p_from IN NUMBER, p_to IN NUMBER, p_ApprovalDate IN DATE, p_ImportanceOrder IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0029	No informar un campo obligatorio o informarlo como NULL
PR0030	Informar un Order ya existente
PR0031	Añadir un elemento con un rango ya existente
PR0032	Informar todos los datos requeridos

Tabla	**NUSER**
Procedimiento	**Código: SP0008 - SP_NUSER_DELETE**
Descripción	Procedimiento que da de baja lógica un rango de usuarios.
Parámetros de entrada	p_NUserID IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0033	No informar un campo obligatorio o informarlo como NULL
PR0034	Informar todos los datos requeridos

PR0035	Informar un ID que ya ha sido eliminado
PR0036	Informar un ID inexistente en BBDD

Tabla	NUSER
Procedimiento	**Código: SP0009 - SP_NUSER_UPDATE**
Descripción	Procedimiento que modifica la configuración de un rango de usuarios
Parámetros de entrada	p_NUserID IN NUMBER, p_RangeName IN VARCHAR2, p_from IN NUMBER, p_to IN NUMBER, p_ApprovalDate IN DATE, p_ImportanceOrder IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0037	No informar un campo obligatorio o informarlo como NULL
PR0038	Informar un ID que ya ha sido eliminado
PR0039	Informar un ID inexistente en BBDD
PR0040	Informar un Order ya existente
PR0041	Informar un rango ya existente
PR0042	Informar todos los datos requeridos

Tabla	APPROVALFLOW
Procedimiento	**Código: SP0010 - SP_APPROVALFLOW_ADD**
Descripción	Procedimiento que se ocupa de dar de alta un nuevo flujo de aprobación
Parámetros de entrada	p_Flow IN VARCHAR2, p_validationType IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0043	No informar un campo obligatorio o informarlo como NULL
PR0044	Añadir un elemento con un nombre ya existente
PR0045	Informar un ValidationType Inexistente
PR0046	Informar todos los datos requeridos

Tabla	APPROVALFLOW
Procedimiento	**Código: SP0011 - SP_APPROVALFLOW_DELETE**
Descripción	Procedimiento que da de baja lógica un flujo de aprobación
Parámetros de entrada	p_ID IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0047	No informar un campo obligatorio o informarlo como NULL
PR0048	Informar todos los datos requeridos
PR0049	Informar un ID que ya ha sido eliminado
PR0050	Informar un ID inexistente en BBDD

Tabla	APPROVALFLOW
Procedimiento	**Código: SP0012 - SP_APPROVALFLOW_UPDATE**
Descripción	Procedimiento que modifica la configuración de un flujo de aprobación
Parámetros de entrada	p_ID IN NUMBER, p_Flow IN VARCHAR2, p_validationType IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0051	No informar un campo obligatorio o informarlo como NULL
PR0052	Informar un ID que ya ha sido eliminado
PR0053	Informar un ID inexistente en BBDD
PR0054	Informar un ValidationType Inexistente
PR0055	Informar un nombre ya existente
PR0056	Informar todos los datos requeridos

Tabla	APPROVALNEEDED
Procedimiento	**Código: SP0013 - SP_APPROVALNEEDED_ADD**
Descripción	Procedimiento que se ocupa de dar de alta los pasos necesarios para un flujo de aprobación.
Parámetros de entrada	p_order IN VARCHAR2, p_IDRoleType IN NUMBER, p_IDApprovalFlow IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0057	No informar un campo obligatorio o informarlo como NULL
PR0058	Informar un ID de RoleType inexistente.
PR0059	Informar un ID de ApprovalFlow que ya ha sido eliminado.
PR0060	Informar un valor Order ya existente para un flujo de aprobación que valida por orden.
PR0061	Informar un RoleType ya existente para un flujo de aprobación.
PR0062	Informar todos los datos requeridos

Tabla	APPROVALNEEDED
Procedimiento	**Código: SP0014 - SP_APPROVALNEEDED_DELETE**
Descripción	Procedimiento que da de baja lógica los pasos necesarios para un flujo de aprobación.
Parámetros de entrada	p_ID IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0063	No informar un campo obligatorio o informarlo como NULL
PR0064	Informar todos los datos requeridos
PR0065	Informar un ID que ya ha sido eliminado
PR0066	Informar un ID inexistente en BBDD

Tabla	APPROVALNEEDED
Procedimiento	**Código: SP0015 - SP_APPROVALNEEDED_UPDATE**
Descripción	Procedimiento que modifica la configuración de los pasos necesarios para un flujo de aprobación.
Parámetros de entrada	p_ID IN NUMBER, p_order IN VARCHAR2, p_IDRoleType IN NUMBER, p_IDApprovalFlow IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0067	No informar un campo obligatorio o informarlo como NULL
PR0068	Informar un ID que ya ha sido eliminado
PR0069	Informar un ID inexistente en BBDD
PR0070	Informar un ID de RoleType inexistente.
PR0071	Informar un ID de ApprovalFlow que ya ha sido eliminado.
PR0072	Informar un valor Order ya existente para un flujo de aprobación que valida por orden.
PR0073	Informar un RoleType ya existente para un flujo de aprobación.
PR0074	Informar todos los datos requeridos

Tabla	PERSON
Procedimiento	**Código: SP0016 - SP_PERSON_ADD**
Descripción	Procedimiento que se ocupa de dar de alta personas.
Parámetros de entrada	p_name IN VARCHAR2, p_lastName IN VARCHAR2, p_email IN VARCHAR2, p_phonenumber IN VARCHAR2, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0075	No informar un campo obligatorio o informarlo como NULL
PR0076	Añadir una persona con nombre y apellido ya existente
PR0077	Informar todos los datos requeridos

Tabla	PERSON
Procedimiento	**Código: SP0017 - SP_PERSON_DELETE**
Descripción	Procedimiento que da de baja lógica personas.
Parámetros de entrada	p_ID IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0078	No informar un campo obligatorio o informarlo como NULL
PR0079	Informar todos los datos requeridos
PR0080	Informar un ID que ya ha sido eliminado
PR0081	Informar un ID inexistente en BBDD

Tabla	PERSON
Procedimiento	**Código: SP0018 - SP_PERSON_UPDATE**
Descripción	Procedimiento que modifica los datos de personas.
Parámetros de entrada	p_ID IN NUMBER, p_name IN VARCHAR2, p_lastName IN VARCHAR2, p_email IN VARCHAR2, p_phonenumber IN VARCHAR2, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0082	No informar un campo obligatorio o informarlo como NULL
PR0083	Informar un ID que ya ha sido eliminado
PR0084	Informar un ID inexistente en BBDD
PR0085	Modificar nombre y apellido al de una persona ya existente
PR0086	Informar todos los datos requeridos

Tabla	CONTACT
Procedimiento	**Código: SP0019 - SP_CONTACT_ADD**
Descripción	Procedimiento que se ocupa de dar de alta contactos.
Parámetros de entrada	p_enterprise IN VARCHAR2, p_IDPerson IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0087	No informar un campo obligatorio o informarlo como NULL
PR0088	Añadir una persona y empresa ya existente
PR0089	Añadir un ID de persona eliminado
PR0090	Añadir un ID de persona inexistente
PR0091	Informar todos los datos requeridos

Tabla	CONTACT
Procedimiento	**Código: SP0020 - SP_CONTACT_DELETE**
Descripción	Procedimiento que da de baja lógica contactos.
Parámetros de entrada	p_ID IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0092	No informar un campo obligatorio o informarlo como NULL
PR0093	Informar un ID que ya ha sido eliminado
PR0094	Informar un ID inexistente en BBDD
PR0095	Informar todos los datos requeridos

Tabla	CONTACT
Procedimiento	**Código: SP0021 - SP_CONTACT_UPDATE**
Descripción	Procedimiento que modifica los datos de contactos.

Parámetros de entrada	p_ID IN NUMBER, p_enterprise IN VARCHAR2, p_IDPerson IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0096	No informar un campo obligatorio o informarlo como NULL
PR0097	Informar un ID que ya ha sido eliminado
PR0098	Informar un ID inexistente en BBDD
PR0099	Añadir una persona y empresa ya existente
PR0100	Añadir un ID de persona eliminado
PR0101	Añadir un ID de persona inexistente
PR0102	Informar todos los datos requeridos

Tabla	**USERAPPROVER**
Procedimiento	**Código: SP0022 - SP_USERAPPROVER_ADD**
Descripción	Procedimiento que se ocupa de dar de alta usuarios aprobadores de cambios.
Parámetros de entrada	p_user IN VARCHAR2, p_password IN VARCHAR2, p_IDRole IN NUMBER, p_IDPerson IN NUMBER, p_IDSubstitute IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0103	No informar un campo obligatorio o informarlo como NULL
PR0104	Añadir un usuario ya existente
PR0105	Añadir un ID de persona eliminado
PR0106	Añadir un ID de persona inexistente
PR0107	Añadir un ID de rol inexistente
PR0108	Añadir un ID de substituto eliminado
PR0109	Añadir un ID de substituto inexistente
PR0110	Añadir un ID substituto que ya es substituto de otra persona
PR0111	Informar todos los datos requeridos

Tabla	**USERAPPROVER**
Procedimiento	**Código: SP0023 - SP_USERAPPROVER_DELETE**
Descripción	Procedimiento que da de baja lógica usuarios aprobadores de cambios
Parámetros de entrada	p_ID IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0112	No informar un campo obligatorio o informarlo como NULL
PR0113	Informar un ID que ya ha sido eliminado
PR0114	Informar un ID inexistente en BBDD
PR0115	Informar todos los datos requeridos

Tabla	USERAPPROVER
Procedimiento	**Código: SP0024 - SP_USERAPPROVER_UPDATE**
Descripción	Procedimiento que modifica los datos de usuarios aprobadores de cambios
Parámetros de entrada	p_ID IN NUMBER, p_user IN VARCHAR2, p_password IN VARCHAR2, p_IDRole IN NUMBER, p_IDPerson IN NUMBER, p_IDSubstitute IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0116	No informar un campo obligatorio o informarlo como NULL
PR0117	Informar un ID que ya ha sido eliminado
PR0118	Informar un ID inexistente en BBDD
PR0119	Añadir un usuario ya existente
PR0120	Añadir un ID de persona eliminado
PR0121	Añadir un ID de persona inexistente
PR0122	Añadir un ID de rol inexistente
PR0123	Añadir un ID de substituto eliminado
PR0124	Añadir un ID de substituto inexistente
PR0125	Añadir un ID substituto que ya es substituto de otra persona
PR0126	Informar todos los datos requeridos

Tabla	APPLICATION
Procedimiento	**Código: SP0025 - SP_APPLICATION_ADD**
Descripción	Procedimiento que se ocupa de dar de alta aplicaciones.
Parámetros de entrada	p_name IN VARCHAR2, p_productionDate IN DATE, p_avgUsers IN NUMBER, p_devTechnology IN VARCHAR2, p_infraestructure IN VARCHAR2, p_idcritically IN NUMBER, p_information IN VARCHAR2, p_idTechContact IN NUMBER, p_idBusinessContact NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0127	No informar un campo obligatorio o informarlo como NULL
PR0128	Añadir una aplicación ya existente
PR0129	Añadir un IDCritically inexistente
PR0130	Añadir un ID de contacto técnico eliminado
PR0131	Añadir un ID de contacto técnico inexistente
PR0132	Añadir un ID de contacto comercial eliminado
PR0133	Añadir un ID de contacto comercial inexistente
PR0134	Informar todos los datos requeridos

Tabla	APPLICATION
Procedimiento	**Código: SP0026 - SP_APPLICATION_DELETE**
Descripción	Procedimiento que da de baja lógica aplicaciones
Parámetros de entrada	p_ID IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0135	No informar un campo obligatorio o informarlo como NULL
PR0136	Informar un ID que ya ha sido eliminado
PR0137	Informar un ID inexistente en BBDD
PR0138	Informar todos los datos requeridos

Tabla	APPLICATION
Procedimiento	**Código: SP0027 - SP_APPLICATION_UPDATE**
Descripción	Procedimiento que modifica los datos de aplicaciones
Parámetros de entrada	p_ID IN NUMBER, p_name IN VARCHAR2, p_productionDate IN DATE, p_avgUsers IN NUMBER, p_devTechnology IN VARCHAR2, p_infraestructure IN VARCHAR2, p_idcritically IN NUMBER, p_information IN VARCHAR2, p_idTechContact IN NUMBER, p_idBusinessContact NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0139	No informar un campo obligatorio o informarlo como NULL
PR0140	Informar un ID que ya ha sido eliminado
PR0141	Informar un ID inexistente en BBDD
PR0142	Modificar el nombre al de una aplicación ya existente
PR0143	Añadir un IDCritically inexistente
PR0144	Añadir un ID de contacto técnico eliminado
PR0145	Añadir un ID de contacto técnico inexistente
PR0146	Añadir un ID de contacto comercial eliminado
PR0147	Añadir un ID de contacto comercial inexistente
PR0148	Informar todos los datos requeridos

Tabla	AUDIT
Procedimiento	**Código: SP0028 - SP_AUDIT_ADD**
Descripción	Procedimiento que se ocupa de dar de alta auditorías.
Parámetros de entrada	p_detalls IN VARCHAR2, p_auditDate IN DATE, p_auditor IN VARCHAR2, p_idApp IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**

PR0149	No informar un campo obligatorio o informarlo como NULL
PR0150	Añadir un ID de aplicación eliminado
PR0151	Añadir un ID de aplicación inexistente
PR0152	Informar todos los datos requeridos

Tabla	**AUDIT**
Procedimiento	**Código: SP0029 - SP_AUDIT_DELETE**
Descripción	Procedimiento que da de baja lógica auditorías.
Parámetros de entrada	p_ID IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0153	No informar un campo obligatorio o informarlo como NULL
PR0154	Informar todos los datos requeridos
PR0155	Informar un ID que ya ha sido eliminado
PR0156	Informar un ID inexistente en BBDD

Tabla	**AUDIT**
Procedimiento	**Código: SP0030 - SP_AUDIT_UPDATE**
Descripción	Procedimiento que modifica los datos de auditorías.
Parámetros de entrada	p_ID IN NUMBER, p_detalls IN VARCHAR2, p_auditDate IN DATE, p_auditor IN VARCHAR2, p_idApp IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0157	No informar un campo obligatorio o informarlo como NULL
PR0158	Informar un ID que ya ha sido eliminado
PR0159	Informar un ID inexistente en BBDD
PR0160	Informar un ID de aplicación eliminado
PR0161	Informar un ID de aplicación inexistente
PR0162	Informar todos los datos requeridos

Tabla	**NONCOMPLIANCE**
Procedimiento	**Código: SP0031 - SP_NONCOMPLIANCE_ADD**
Descripción	Procedimiento que se ocupa de dar de alta incumplimientos detectados en las auditorías.
Parámetros de entrada	p_detalls IN VARCHAR2, p_IDAudit IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0163	No informar un campo obligatorio o informarlo como NULL
PR0164	Informar un ID de auditoría eliminado
PR0165	Informar un ID de auditoría inexistente

PR0166	Informar todos los datos requeridos

Tabla	**NONCOMPLIANCE**
Procedimiento	**Código: SP0032 - SP_NONCOMPLIANCE_DELETE**
Descripción	Procedimiento que da de baja lógica incumplimientos detectados en las auditorías.
Parámetros de entrada	p_ID IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0167	No informar un campo obligatorio o informarlo como NULL
PR0168	Informar todos los datos requeridos
PR0169	Informar un ID que ya ha sido eliminado
PR0170	Informar un ID inexistente en BBDD

Tabla	**NONCOMPLIANCE**
Procedimiento	**Código: SP0033 - SP_NONCOMPLIANCE_UPDATE**
Descripción	Procedimiento que modifica los datos de incumplimientos detectados en las auditorías.
Parámetros de entrada	p_ID IN NUMBER, p_detalls IN VARCHAR2, p_IDAudit IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0171	No informar un campo obligatorio o informarlo como NULL
PR0172	Informar un ID que ya ha sido eliminado
PR0173	Informar un ID inexistente en BBDD
PR0174	Informar un ID de auditoría eliminado
PR0175	Informar un ID de auditoría inexistente
PR0176	Informar todos los datos requeridos

Tabla	**CHANGE**
Procedimiento	**Código: SP0034 - SP_CHANGE_ADD**
Descripción	Procedimiento que se ocupa de dar de alta cambios.
Parámetros de entrada	p_description IN VARCHAR2, p_orderDate IN DATE, p_IDCategory IN NUMBER, p_IDGeoScope IN NUMBER, p_IDNUser IN NUMBER, p_IDApprovalFlow IN NUMBER, p_IDApp IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0177	No informar un campo obligatorio o informarlo como NULL
PR0178	Informar un ID de categoría eliminado

PR0179	Informar un ID de categoría inexistente
PR0180	Informar un ID de alcance geográfico eliminado
PR0181	Informar un ID de alcance geográfico inexistente
PR0182	Informar un ID de rango de usuarios eliminado
PR0183	Informar un ID de rango de usuarios inexistente
PR0184	Informar un ID de flujo de aprobación eliminado
PR0185	Informar un ID de flujo de aprobación inexistente
PR0186	Informar un ID de aplicación eliminado
PR0187	Informar un ID de aplicación inexistente
PR0188	Informar todos los datos requeridos

Tabla	**CHANGE**
Procedimiento	**Código: SP0035 - SP_CHANGE_DELETE**
Descripción	Procedimiento que da de baja lógica cambios.
Parámetros de entrada	p_ID IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0189	No informar un campo obligatorio o informarlo como NULL
PR0190	Informar todos los datos requeridos
PR0191	Informar un ID que ya ha sido eliminado
PR0192	Informar un ID inexistente en BBDD

Tabla	**CHANGE**
Procedimiento	**Código: SP0036 - SP_CHANGE_UPDATE**
Descripción	Procedimiento que modifica los datos de cambios.
Parámetros de entrada	p_ID IN NUMBER, p_description IN VARCHAR2, p_orderDate IN DATE, p_IDCategory IN NUMBER, p_IDGeoScope IN NUMBER, p_IDNUser IN NUMBER, p_IDApprovalFlow IN NUMBER, p_IDApp IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0193	No informar un campo obligatorio o informarlo como NULL
PR0194	Informar un ID que ya ha sido eliminado
PR0195	Informar un ID inexistente en BBDD
PR0196	Informar un ID de categoría eliminado
PR0197	Informar un ID de categoría inexistente
PR0198	Informar un ID de alcance geográfico eliminado
PR0199	Informar un ID de alcance geográfico inexistente
PR0200	Informar un ID de rango de usuarios eliminado
PR0201	Informar un ID de rango de usuarios inexistente
PR0202	Informar un ID de flujo de aprobación eliminado
PR0203	Informar un ID de flujo de aprobación inexistente

PR0204	Informar un ID de aplicación eliminado
PR0205	Informar un ID de aplicación inexistente
PR0206	Informar todos los datos requeridos

Tabla	EXECUTION
Procedimiento	**Código: SP0037 - SP_EXECUTION_ADD**
Descripción	Procedimiento que se ocupa de dar de alta ejecuciones.
Parámetros de entrada	p_IDState IN NUMBER, p_executionDate IN DATE, p_comments IN VARCHAR2, p_IDChange IN NUMBER, p_IDTech IN NUMBER, p_actionPlan IN VARCHAR2, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0207	No informar un campo obligatorio o informarlo como NULL
PR0208	Informar un ID de cambio eliminado
PR0209	Informar un ID de cambio inexistente
PR0210	Informar un ID de responsable técnico eliminado
PR0211	Informar un ID de responsable técnico inexistente
PR0212	Informar un IDState inexistente
PR0213	Informar todos los datos requeridos con IDState 1 (correcto)
PR0214	Añadir un registro con IDState 2 o 3 sin informar el plan de acción.
PR0215	Informar todos los datos requeridos con IDState 2 o 3 (no correcto)
PR0216	Añadir un registro para un cambio que ya tiene una ejecución.

Tabla	EXECUTION
Procedimiento	**Código: SP0038 - SP_EXECUTION_DELETE**
Descripción	Procedimiento que da de baja lógica ejecuciones.
Parámetros de entrada	p_ID IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0217	No informar un campo obligatorio o informarlo como NULL
PR0218	Informar un ID de ejecución de una ejecución en estado correcto.
PR0219	Informar un ID que ya ha sido eliminado
PR0220	Informar un ID de ejecución de una ejecución en estado no correcto.
PR0221	Informar un ID inexistente en BBDD

Tabla	EXECUTION
Procedimiento	**Código: SP0039 - SP_EXECUTION_UPDATE**
Descripción	Procedimiento que modifica los datos de ejecuciones.

Parámetros de entrada	p_ID IN NUMBER, p_IDState IN NUMBER, p_executionDate IN DATE, p_comments IN VARCHAR2, p_IDChange IN NUMBER, p_IDTech IN NUMBER, p_actionPlan IN VARCHAR2, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0222	No informar un campo obligatorio o informarlo como NULL
PR0223	Informar un ID que ya ha sido eliminado
PR0224	Informar un ID inexistente en BBDD
PR0225	Informar un ID de cambio eliminado
PR0226	Informar un ID de cambio inexistente
PR0227	Informar un ID de responsable técnico eliminado
PR0228	Informar un ID de responsable técnico inexistente
PR0229	Informar un IDState inexistente
PR0230	Informar todos los datos requeridos con IDState 1 (correcto)
PR0231	Añadir un registro con IDState 2 o 3 sin informar el plan de acción.
PR0232	Informar todos los datos requeridos con IDState 2 o 3 (no correcto)
PR0233	Ejecutar dos veces un update con la misma información con IDState 1 (correcto)
PR0234	Ejecutar dos veces un update con la misma información con IDState 2 o 3 (no correcto)
PR0235	Modificar la persona responsable de una ejecución.

Tabla	ACTIONPLAN
Procedimiento	**Código: SP0040 - SP_ACTIONPLAN_UPDATE**
Descripción	Procedimiento que modifica los datos de un plan de acción. NOTAS: - No existen procedimientos para ALTAS y BAJAS ya que se gestionan desde la EJECUCIÓN. - No se permite la modificación la ejecución a la que pertenece dicho plan de acción. - No se permite modificar la persona responsable del plan de acción.
Parámetros de entrada	p_ID IN NUMBER, p_description IN VARCHAR2, p_isValidated IN NUMBER, p_idValidatedBy IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0236	No informar un campo obligatorio o informarlo como NULL
PR0237	Informar un ID que ya ha sido eliminado
PR0238	Informar un ID inexistente en BBDD
PR0239	Informar un ID de validador eliminado
PR0240	Informar un ID de validador inexistente
PR0241	Informar un ID de validador de una persona que no es gestor de cambios.
PR0242	Informar que un plan ha sido validado sin informar quién lo valida.

PR0243	Informar todos los datos requeridos

Tabla	ACTION
Procedimiento	**Código: SP0041 - SP_ACTION_ADD**
Descripción	Procedimiento que se ocupa de dar de alta acciones de un plan de acción.
Parámetros de entrada	p_description IN VARCHAR2, p_IDActionPlan IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0244	No informar un campo obligatorio o informarlo como NULL
PR0245	Informar un ID de plan de acción eliminado
PR0246	Informar un ID de plan de acción inexistente
PR0247	Informar una acción para un plan de acción ya validado
PR0248	Informar todos los datos requeridos

Tabla	ACTION
Procedimiento	**Código: SP0042 - SP_ACTION_DELETE**
Descripción	Procedimiento que da de baja lógica acciones de un plan de acción.
Parámetros de entrada	p_ID IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0249	No informar un campo obligatorio o informarlo como NULL
PR0250	Informar todos los datos requeridos
PR0251	Informar un ID que ya ha sido eliminado
PR0252	Informar un ID inexistente en BBDD

Tabla	ACTION
Procedimiento	**Código: SP0043 - SP_ACTION_UPDATE**
Descripción	Procedimiento que modifica los datos de acciones de un plan de acción.
Parámetros de entrada	p_ID IN NUMBER, p_description IN VARCHAR2, p_IDActionPlan IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0253	No informar un campo obligatorio o informarlo como NULL
PR0254	Informar un ID que ya ha sido eliminado
PR0255	Informar un ID inexistente en BBDD
PR0256	Informar un ID de plan de acción eliminado
PR0257	Informar un ID de plan de acción inexistente
PR0258	Informar un ID de plan de acción ya validado
PR0259	Informar todos los datos requeridos

Tabla	APPROVAL

Procedimiento	Código: SP0044 - SP_APPROVAL_ADD
Descripción	Procedimiento que se ocupa de dar de alta aprobaciones. NOTA: - No se permitirá la modificación de aprobaciones por lo que todo cambio requerirá que la persona que ha aprobado elimine su aprobación para así dar lugar a que otro la realice.
Parámetros de entrada	p_comments IN VARCHAR2, p_IDChange IN NUMBER, p_IDUserApproval IN NUMBER, RSP OUT VARCHAR2, p_absence IN VARCHAR2 DEFAULT NULL, p_IDAbsenceType IN NUMBER DEFAULT NULL
Código Prueba	**PRUEBAS**
PR0260	No informar un campo obligatorio o informarlo como NULL
PR0261	Informar un ID de cambio eliminado
PR0262	Informar un ID de cambio inexistente
PR0263	Informar un ID de cambio ya aprobado
PR0264	Informar un ID de aprobador eliminado
PR0265	Informar un ID de aprobador inexistente
PR0266	Informar un ID de aprobador con rol diferente al requerido por el flujo de aprobación (por orden)
PR0267	Informar un ID de aprobador con rol diferente al requerido por el flujo de aprobación (sin orden)
PR0268	Informar un ID de aprobador sustituto sin informar el motivo o el tipo de ausencia
PR0269	Informar un ID de aprobador sustituto sin informar el motivo y el tipo de ausencia
PR0270	Informar todos los datos requeridos

Tabla	APPROVAL
Procedimiento	Código: SP0045 - SP_APPROVAL_DELETE
Descripción	Procedimiento que da de baja lógica acciones de aprobaciones. NOTA: - No se permitirá la modificación de aprobaciones por lo que todo cambio requerirá que la persona que ha aprobado elimine su aprobación para así dar lugar a que otro la realice.
Parámetros de entrada	p_ID IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0271	No informar un campo obligatorio o informarlo como NULL.
PR0272	Informar el ID de aprobación de un cambio no aprobado aún.
PR0273	Informar un ID que ya ha sido eliminado.
PR0274	Informar un ID inexistente en BBDD.
PR0275	Informar el ID de aprobación de un cambio ya aprobado.
PR0276	Informar el ID de una aprobación realizada por un sustituto.

Tabla	ABSENCE
Procedimiento	**Código: SP0046 - SP_ABSENCE_UPDATE**
Descripción	Procedimiento que modifica el motivo y tipo de una ausencia. NOTAS: - No existen procedimientos para ALTAS y BAJAS ya que se gestionan desde la APROBACIÓN. - No se permite la modificación de la aprobación a la que pertenece la ausencia. - No se permite modificar la persona que se ha sustituido en la ausencia.
Parámetros de entrada	p_ID IN NUMBER, p_absence IN VARCHAR2, p_idAbsenceType IN NUMBER, RSP OUT VARCHAR2
Código Prueba	**PRUEBAS**
PR0277	No informar un campo obligatorio o informarlo como NULL
PR0278	Informar un ID que ya ha sido eliminado
PR0279	Informar un ID inexistente en BBDD
PR0280	Informar un ID de tipo de ausencia inexistente.
PR0281	Informar todos los datos requeridos

6.3. Triggers

Tras la creación de las tablas necesarias, debemos implementar una serie de disparadores para almacenar los campos que hemos identificado como campos calculados en el diseño conceptual.

El código para generar los disparadores se encuentra en el fichero “05. Triggers.sql”.

05. Triggers.sql

Los *triggers* identificados se detallan a continuación:

CalculateImpact

Este disparador se ejecutará cada vez que se realice un cambio en la tabla CHANGE.

Su objetivo es calcular el campo IMPACT en base a la categoría, el alcance geográfico y el rango de usuarios que afectan al cambio.

La consulta recupera el orden de importancia de cada uno de los elementos y multiplica los tres valores para obtener el valor del impacto.

CalculateInfoApproval

Este disparador se ejecutará cada vez que se realice un cambio en la tabla APPROVAL.

Se ha de destacar que este disparador requiere realizar consultas sobre la misma tabla que genera su ejecución. Esto no es una acción que esté permitida en Oracle y genera el error *“mutating table error”.*

Para solventar esta situación se investiga sobre dicho error y se resuelve que el mecanismo que Oracle pone a disposición para este tipo de situaciones es el uso de

COMPOUND TRIGGERS (disparadores compuestos), que permite definición de acciones en 4 tiempos diferentes:

- Antes de una consulta DML
- Antes de cada fila
- Después de cada fila
- Después de una consulta DML

Volviendo al disparador en cuestión, su objetivo es calcular el campo ISAPPROVED de la tabla CHANGE en base al flujo de aprobación que tiene asignado el cambio y al número de aprobaciones que se hayan registrado sobre él.

La consulta recupera la cantidad de aprobaciones necesarias para el cambio y la compara con las existentes. En el momento que ambos números de aprobaciones son iguales se establece el cambio como aprobado, caso contrario se mantiene como no aprobado.

CalculateQuantityApproval

Este disparador se ejecutará cada vez que se realice un cambio en la tabla APPROVALNEEDED.

Su objetivo es calcular el campo QUANTITYAPPROVAL de la tabla APPROVALFLOW en base al número de aprobaciones necesarias que se hayan configurado. Este campo se crea a modo auxiliar para facilitar la gestión de las aprobaciones.

La manera de calcular el valor es obteniendo el valor inicial del flujo de aprobación y si existe un nuevo registro asociado a dicho flujo, el campo es incrementado en uno. Por el contrario, ante una modificación o borrado asociado a dicho flujo de aprobación, genera un decremento en el valor del campo.

CalculateSubstitute

Este disparador se ejecutará cada vez que se realice un cambio en la tabla APPROVAL.

Su objetivo es calcular el campo BYSUBSTITUTE de la tabla APPROVAL en base a la configuración del usuario.

La forma de calcular su valor dependerá de si el usuario aprobador es un sustituto de otro usuario, si lo es, el valor será TRUE sino FALSE.

7. Pruebas

Si bien la definición de pruebas se ha listado en el apartado anterior por una cuestión de sinergias al momento de realizar el detalle de los procedimientos almacenados, consideramos de suficiente importancia detallar los aspectos que se han tenido en cuenta de cara a la definición del listado de pruebas.

Pruebas de Casos Estándar:

- Verificar que el procedimiento se ejecute correctamente con datos válidos.
- Probar con valores mínimos y máximos permitidos para cada parámetro.
- Probar con valores con tipos de datos diferentes.
- Probar el correcto funcionamiento de los disparadores.
- Verificar el comportamiento del procedimiento en todos los casos.

Pruebas de Casos Incorrectos:

- Introducir datos incorrectos y verificar que el procedimiento gestione correctamente los errores.
- Confirmar que se generen mensajes de error comprensibles.

Pruebas de Transacciones:

- Confirmar que el procedimiento gestione correctamente las transacciones (commit y rollback).

7.1. Gestión de Errores

Parte de la definición de pruebas también pasa por tener en cuenta los diferentes escenarios que pueden generar errores y catalogarlos de una manera clara para que sea de fácil interpretación para terceros.

Para esta gestión nos apoyamos en algunos errores gestionados por el sistema, pero también generamos una codificación personalizada para el resto de los errores más relacionados con la lógica de negocio. En este sentido, Oracle pone a disposición de los desarrolladores, un rango de números para errores personalizados que va del -20000 hasta el -20999.

Se define pues una serie de mensajes de error y se categorizan de la siguiente manera:

TIPO	Explicación	CódigoError	EjemploError
ORACLE	Control de campo NOT NULL	ORA-01400	no se puede realizar una inserción NULL en ……..
ORACLE	Check para controlar cuando la consulta NO devuelve datos	ORA-01403	No se ha encontrado ningún dato
ORACLE	Reestricción de integridad referencial	ORA-02291	restricción de integridad ……
CUSTOM	Control de parámetros de entrada obligatorios	ORA-20000	All parameters are mandatory
CUSTOM	Controla si un ID a borrar ya ha sido borrado previamente	ORA-20001	The element already was deleted
CUSTOM	El ID informado para actualizar no existe	ORA-20002	The ID informed does not exist
CUSTOM	Se controla el hecho de asignar más de un elemento a otro, cuando sólo permite uno.	ORA-20003	Already exist a Execution for this Change
CUSTOM	Se intenta asignar a un elemento, otro que ha sido borrado.	ORA-20004	The ActionPlan informed was deleted or does not exist
CUSTOM	El Rol del usuario no es el requerido para continuar con la aprobación del cambio.	ORA-20005	The User has a role required to aproove the change (by order)
CUSTOM	El Rol del usuario no es uno de los necesarios para aprobar el cambio.	ORA-20006	The User has a role not needed to aproove the change (without order)
CUSTOM	El Flujo de Aprobación está mal configurado	ORA-20007	The ApprovalFlow for this change is not correctly configured
CUSTOM	Evita añadir acciones a un plan de acción ya validado	ORA-20008	Is not possible add actions to an ActionPlan already validated
CUSTOM	Controla que una ejecución no correcta tenga un plan de acción asociado.	ORA-20009	The Action Plan is mandatory if the execution state is not correct

CUSTOM	Controla específicamente la validación de un plan de acción.	ORA-20010	To validate an ActionPlan is mandatory an IDValidatedBy
CUSTOM	Las validaciones deben ser realizadas por un Gestor de Cambios	ORA-20011	The Validator must be a Change Manager
CUSTOM	Evita que se apruebe un cambio que ya está previamente aprobado.	ORA-20012	The Change informed is already approved
CUSTOM	Se ha creado un valor en el enumerado ValidationType que no ha sido desarrollado.	ORA-20013	The ValidationType is not developped, contact with the administrator.
CUSTOM	Controla que una aprobación hecha por un sustituto se justifique correctamente.	ORA-20014	The absence reason and type are mandatory if the change is approve by a substitute
CUSTOM	Impide que un cambio ya aprobado se "desapruebe".	ORA-20015	Is not possible delete an approval in a change already approved
CUSTOM	Controla que no haya más de una categoría, alcance geográfico o rango de usuarios con el mismo orden	ORA-20016	Already exist a category with the same order
CUSTOM	Controla que no exista un paso de aprobación con el mismo orden.	ORA-20017	Already exist an approval needed with the same order
CUSTOM	Controla que no exista un paso de aprobación con el mismo rol.	ORA-20018	Already exist an approval needed with the same role type.

7.2. Ejecución de las pruebas

El listado de pruebas ha contemplado 281 escenarios repartidos entre los 46 procedimientos almacenados que se han elaborado. Para llevar a cabo este listado de pruebas, se ha trabajado sobre los datos iniciales y con los que se han ido creando durante las mismas pruebas.

Se ha creado un archivo de TEST para cada entidad sobre la que se han hecho las pruebas, los ficheros se pueden consultar en el siguiente link:

Pruebas

Para tener un control de las pruebas ejecutadas y comprobar su correcto funcionamiento se ha creado una tabla TESTRESULT donde se ha ido almacenando la ejecución de cada una de las pruebas realizadas mediante estos ficheros. Como se ha visto anteriormente, cada prueba tiene un código único para identificarla y con esta codificación podemos analizar si los resultados han sido los esperados o, si no la han sido, realizar las acciones correspondientes.

La tabla resultante se visualiza de la siguiente manera:

TESTDATE	TESTCODE	INPUT	OUTPUT
15/01/24	PR0281	SP_ABSENCE_UPDATE(3, 'Enfermedad MODIF', 2, ...	OK - Element updated succesfully w..
15/01/24	PR0280	SP_ABSENCE_UPDATE(3, 'Estaba planeado', 5, v...	KO - Error to update the element. ..
15/01/24	PR0279	SP_ABSENCE_UPDATE(100, 'Estaba planeado', 1,...	KO - Error to update the element. ..
15/01/24	PR0278	SP_ABSENCE_UPDATE(5, 'Estaba planeado', 1, v...	KO - Error to update the element. ..
15/01/24	PR0277	SP_ABSENCE_UPDATE(NULL, 'Estaba planeado', 1...	KO - Error to update the element. ..
15/01/24	PR0276	SP_APPROVAL_DELETE(40, v_Result);	OK - Element deleted succesfully w..
15/01/24	PR0275	SP_APPROVAL_DELETE(3, v_Result);	KO - Error to delete the element. ..
15/01/24	PR0274	SP_APPROVAL_DELETE(100, v_Result);	KO - Error to delete the element. ..
15/01/24	PR0273	SP_APPROVAL_DELETE(41, v_Result);	KO - Error to delete the element. ..
15/01/24	PR0272	SP_APPROVAL_DELETE(41, v_Result);	OK - Element deleted succesfully w..
15/01/24	PR0271	SP_APPROVAL_DELETE(NULL, v_Result);	KO - Error to delete the element. ..
15/01/24	PR0270	SP_APPROVAL_ADD('Aprobacion 3', 15, 13, v_Re...	OK - Element created succesfully w..
15/01/24	PR0269	SP_APPROVAL_ADD('Aprobacion 3', 15, 13, v_Re...	KO - Error to create the element. ..
15/01/24	PR0268	SP_APPROVAL_ADD('Aprobacion 3', 15, 13, v_Re...	KO - Error to create the element. ..
15/01/24	PR0267	SP_APPROVAL_ADD('Aprobacion 3', 9, 7, v_Resu...	KO - Error to create the element. ..
15/01/24	PR0266	SP_APPROVAL_ADD('Aprobacion 3', 2, 7, v_Resu...	KO - Error to create the element. ..
15/01/24	PR0265	SP_APPROVAL_ADD('Aprobacion 3', 15, 100, v_R...	KO - Error to create the element. ..
15/01/24	PR0264	SP_APPROVAL_ADD('Aprobacion 3', 15, 2, v_Res...	KO - Error to create the element. ..

Los resultados completos de las pruebas ejecutadas se pueden encontrar en el siguiente fichero:

Tabla TESTRESULT.csv

7.3. Errores detectados

A pesar de haber hecho un estricto seguimiento de las tareas a realizar en todas las fases del diseño, en cada una de las etapas se han identificado correcciones a realizar en la estructura de la base de datos.

Tras la ejecución de las pruebas hemos identificado y corregido los siguientes puntos:

1. La ausencia de la clave foránea de la tabla USERAPPROVER en el campo IDROLETYPE.
2. Tras haber añadido el campo ISDELETED a todas las tablas, no habíamos creado la restricción CHECK para convertir este campo de tipo BOOLEANO.
3. La tabla ABSENSE necesita almacenar el ID de la persona ausente por una cuestión de persistencia de la información.
4. Se añadieron algunos campos como claves únicas ya que se ha considerado necesario para una mejor gestión de la información.
5. Se identifica que el campo IDVALIDATEDBY y VALIDATIONDATE de la tabla ACTIONPLAN deben aceptar valores NULL, ya que al crear un plan de acción, este no está validado y por tanto los campos estarán vacíos.

8. Conclusiones

Tras haber experimentado tanto el desarrollo del diseño de una base de datos de principio a fin, como la simulación del encargo por parte de un cliente, puedo decir que he finalizado este proyecto obteniendo grandes aprendizajes que intentaré poner en práctica durante el desarrollo de mis tareas profesionales.

Por un lado, considero que en mi caso ha tenido un valor añadido toda la parte de gestión de proyecto, orden de tareas, toma de requisitos, planificación y seguimiento.
La metodología en cascada que he escogido ha sido clave para el desarrollo del proyecto, aunque en ciertos momentos una metodología más ágil podría haber sido útil, sobre todo en los momentos en los que, por motivos personales, no he podido dedicar el volumen de horas previsto. Sin embargo, creo que he sabido reaccionar ante la necesidad de adaptación y de reajustar la planificación en este tipo de situaciones.
Quisiera destacar que, si bien no he conseguido el logro completo de los objetivos, la

experiencia que he adquirido en gestionar este proyecto bajo circunstancias de presión es totalmente enriquecedora a nivel personal.

Reconozco que la imposibilidad de completar el apartado de *data warehouse* es un aspecto negativo que influye en todo el desarrollo de mi trabajo y en el tiempo que le he dedicado. Sin embargo, me satisface saber que soy capaz de aprender a adaptarme y de priorizar las tareas para llegar con un resultado lo más próximo a lo solicitado, si bien no cumple el 100% de los requisitos.

Por otra parte, en cuanto a la parte práctica del diseño en sí, pienso que he conseguido enfocar el proyecto de una manera correcta, planteando un producto final con muchas áreas de evolución, sólido, estable y escalable. Por supuesto, como todo proyecto de desarrollo de software, siempre hay correcciones que hacer o mejoras que implementar, pero el trabajo realizado representa una buena base para desarrollar el software.

A nivel de aprendizaje, tanto el diseño conceptual como el lógico ha consistido en poner en práctica lo estudiado en asignaturas anteriores. Sin embargo, el diseño físico y la implementación ha representado todo un reto para mí. Si bien estoy acostumbrado a trabajar con bases de datos, normalmente lo hago con motores SQL de Microsoft, con lo que el cambio a Oracle ha sido muy importante para mí y he aprendido muchos conceptos que desconocía, como por supuesto también sintaxis.

Como ya se ha podido ver, no se ha conseguido el 100% de los objetivos planteados, en concreto con el apartado de Data Warehouse, siendo motivo de este incumplimiento dos aspectos importantes. En primer lugar, he cometido ciertos fallos de estimación de tiempos, el más flagrante relacionado con la creación de los procedimientos almacenados y ejecución de las pruebas, por lo que de cara a futuros proyectos seré más cuidadoso y repartiré de otra manera los pesos en las diferentes tareas.
En segundo lugar, tuvieron hecho algunas situaciones de índole personal de urgencia, que me han obligado a ausentarme tanto del trabajo como de los estudios durante una semana. He intentado recuperar este tiempo, pero junto con el desvío de las horas planificadas, ha resultado imposible.

En cuanto al resto de los objetivos, considero que han tenido un alto grado de cumplimiento. Incluso en ciertos aspectos puedo decir que se ha sobrepasado lo pedido por el cliente inicialmente, añadiéndose pequeños detalles que pueden dejan la base de datos preparada para que el sistema pueda evolucionar en diferentes líneas.

Existen varios aspectos sobre los que puede evolucionar este sistema (Data Warehouse es uno de ellos) y que no hemos abordado durante la realización del proyecto. Me gustaría destacar uno que considero puede resultar muy interesante para las empresas desarrolladoras de software: se trataría de añadir una funcionalidad o apartado para poder asignar los cambios y las tareas que se deben realizar a los diferentes técnicos desarrolladores. Permitiendo, de esta manera, tener un control calendarizado de los desarrollos y facilitando la gestión de quien pudiera cumplir el rol de responsable de Desarrollo.

Por supuesto, hay otras muchas líneas de evolución que se podrían explorar. Como mención anteriormente, un proyecto de desarrollo de software siempre ofrece oportunidades para mejoras o adición de nuevas funcionalidades.

9. Glosario

- **ITIL:** acrónimo de *Information Technology Infrastructure Library.* Consiste básicamente en un marco de trabajo y de buenas practicas en la gestión de los servicios de informática. Busca un enfoque orientado al servicio y la alineación entre la TI con los objetivos del negocio, la mejora continua, la gestión del cambio, entre otros; con el objetivo de ayudar a las empresas a ganar calidad y eficiencia en sus procesos:
- **Gestión del cambio:** implica la planificación y ejecución de estrategias para abordar transiciones tecnológicas en las empresas, buscando minimizar el impacto negativo que se pueda dar en los implicados.
- **BD / BBDD:** acrónimo de base de datos.
- **Data WareHouse:** es una base de datos que almacena de manera centralizada y ordenada grandes volúmenes de información.
- **Despliegue:** acción de poner en producción un determinado desarrollo de software en un entorno específico.
- **Diagrama de Gantt:** es una representación visual y calendarizada de las tareas a realizar en un proyecto. Da una visión clara del cronograma y el desarrollo de las tareas a lo largo del tiempo.
- **Kanbann:** se trata de un método visual para organizar, gestionar y controlar el flujo de trabajo y las tareas a realizar en un determinado proyecto.
- **Scripts:** es un conjunto de instrucciones escritas en un lenguaje determinado, en este caso en PL/SQL, para realizar determinadas acciones.
- **PL/SQL:** es un lenguaje de programación utilizado para gestionar bases de datos de Oracle.
- **Benchmark:** estudio y análisis de la competencia con el objetivo de identificar en qué situación se está dentro del mercado.
- **ABM:** siglas de Alta, Baja y Modificación. Se llama así al hecho de gestionar determinadas entidades de un sistema, haciendo referencia a las operaciones básicas de crear, eliminar y modificar registros en las tablas de una base de datos.
- **Entidad:** en el contexto de las bases de datos, una entidad hace referencia a un objeto o concepto del mundo real que se quiere modelar.
- **Log:** es un registro donde se almacenan las acciones relevantes de ser monitorizadas que ayudan a diagnosticar y solucionar un problema.
- **Disparadores:** es un conjunto de instrucciones que se ejecutan tras realizarse una determinada acción, como puede ser antes o después de realizar una inserción, una modificación o un borrado en una tabla de nuestra base de datos.
- **Procedimientos Almacenados:** es un conjunto de instrucciones que está guardado en una base de datos y que puede ejecutarse a discreción del usuario. Se utiliza para normalizar y reutilizar el código.
- **Formas Normales:** representan una serie de estándares que las bases de datos deben cumplir para garantizar su correcto funcionamiento.

10. Bibliografía

Estrategia y enfoque

- Las 12 metodologías más populares para la gestión de proyectos. Asana. (2023, mayo 14). [En línea] - https://asana.com/es/resources/project-management-methodologies - Consultado entre el 10/10/2023 y el 20/10/2023.
- ITIL - Biblioteca de infraestructura de TI. (s. f.). Ibm.com. [En línea] - https://www.ibm.com/es-es/topics/it-infrastructure-library - Consultado entre el Consultado entre el 10/10/2023 y el 20/10/2023.
- Desarrollo en cascada. Wikipedia, The Free Encyclopedia. Wikipedia contributors. (s. f.-a). [En línea] -]https://es.wikipedia.org/w/index.php?title=Desarrollo_en_cascada&oldid=157007523 - Consultado entre el 10/10/2023 y el 20/10/2023.
- Information Technology Infrastructure Library. Wikipedia, The Free Encyclopedia. Wikipedia contributors. (s. f.-b). [En línea] - https://es.wikipedia.org/w/index.php?title=Information_Technology_Infrastructure_Library&oldid=152825182 - Consultado entre el 10/10/2023 y el 20/10/2023.
- Metodologías de gestión de proyectos: 5 tipos para adoptar. Zendesk. (2023, febrero 14). [En línea] - https://www.zendesk.com.mx/blog/metodologias-gestion-proyectos/ - Consultado entre el 10/10/2023 y el 20/10/2023.

Diseño conceptual, lógico y físico.

- **Jordi Casas Roma; Josep Cuartero Olivera** (2020) "Diseño conceptual de bases de datos" – PID_00270598 – UOC – Consultado entre 01/11/2023 y el 10/01/2024.
- **Xavier Burgués Illa; Josep Cuartero Olivera** (2020) "Diseño lógico de bases de datos" – PID_00270596 – UOC – Consultado entre el 20/11/2023 y el 10/01/2024.
- **Blai Cabré i Segarra; Jordi Casas Roma; Dolors Costal Costa; Pere Juanola Juanola: Ivo Plana Vallvé; Àngels Rius Gavidia; Ramon Segret i Sala** (2020) "Diseño físico de bases de datos" - PID_00278244 – UOC - Consultado entre el 01/12/2023 y el 10/01/2024.

PL/SQL

- The ENUM type. (n.d.). Oracle.com. [En línea] - https://docs.oracle.com/cd/E17952_01/mysql-5.7-en/enum.html - Consultado entre el 01/12/2023 y el 20/12/2023
- FOREIGN KEY constraints. (n.d.). Oracle.com. [En línea] https://docs.oracle.com/cd/E17952_01/mysql-5.7-en/create-table-foreign-keys.html - Consultado entre el 01/12/2023 y el 20/12/2023
- **Moore, S., Jayapalan, L., Huey, P., Belden, E., Rich, K., & Moore, V**. PL/SQL data types. [En línea] - https://docs.oracle.com/en/database/oracle/oracle-database/19/lnpls/plsql-data-types.html - Consultado entre el 01/12/2023 y el 20/12/2023
- **Moore, S., Jayapalan, L., Huey, P., Belden, E., Rich, K., & Moore, V.** PL/SQL triggers. [En línea] - https://docs.oracle.com/en/database/oracle/oracle-database/19/lnpls/plsql-triggers.html - Consultado entre el 01/12/2023 y el 20/12/2023

- Oracle / PLSQL: ALTER TABLE statement. Techonthenet.com. [En línea] - https://www.techonthenet.com/oracle/tables/alter_table.php - Consultado entre el 15/12/2023 y el 10/01/2024
- **Alcalde, A.** (2016, January 1). El Baúl del programador. PL/SQL. Excepciones. [En línea] - https://elbauldelprogramador.com/plsql-excepciones/ - Consultado entre el 15/12/2023 y el 10/01/2024
- **García, J.** (2013, December 12). Evitar el error Oracle ORA-04091 (tabla mutante). [En línea] - https://mundodb.es/evitar-el-error-oracle-ora-04091-tabla-mutante - Consultado entre el 15/12/2023 y el 10/01/202
- **Herrera, G., & Perfil, V. T.** Compartiendo Tips de Oracle. Blogspot.com. [En línea] - https://profesionghh.blogspot.com/2015/01/disparadores-compuestos-compound.html - Consultado entre el 15/12/2023 y el 10/01/2024
- **IBM Documentation.** (2023, November 15). Ibm.com. [En línea] - https://www.ibm.com/docs/es/db2/11.5?topic=plsql-if-statement - Consultado entre el 15/12/2023 y el 10/01/2024
- **Jain, A.** (2014, January 22). Compound Triggers in Oracle 11g Tutorial with example. [En línea] - https://www.viralpatel.net/compound-triggers-in-oracle-11g-tutorial-example/ - Consultado entre el 15/12/2023 y el 10/01/2024
- Mutating table error in Oracle. (2019, August 14). [En línea] - https://www.oracletutorial.com/plsql-tutorial/mutating-table-error-in-oracle/ - Consultado entre el 15/12/2023 y el 10/01/2024
- Oracle / PLSQL: BEFORE DELETE trigger. [En línea] - https://www.techonthenet.com/oracle/triggers/before_delete.php - Consultado entre el 15/12/2023 y el 10/01/202
- Oracle 11gR2 PLS-00679 trigger binds not allowed in before/after statement section. [En línea] - https://www.oraexcel.com/oracle-11gR2-PLS-00679/lang-es - Consultado entre el 15/12/2023 y el 10/01/2024
- Oracle ALTER TABLE ADD column by examples. (2017, October 2). [En línea] - https://www.oracletutorial.com/oracle-basics/oracle-alter-table-add-column/ - Consultado entre el 15/12/2023 y el 10/01/2024

Formas Normales

- Normalización de bases de datos. Wikipedia, The Free Encyclopedia. [En línea] https://es.wikipedia.org/w/index.php?title=Normalizaci%C3%B3n_de_bases_de_datos&oldid=153863184 - Consultado entre el 15/12/2023 y el 10/01/2024
- Descripción de la normalización de la base de datos. Microsoft.com. [En línea] - https://learn.microsoft.com/es-es/office/troubleshoot/access/database-normalization-description - Consultado entre el 15/12/2023 y el 10/01/2024