Sistema de gestión de amonestaciones y sanciones en centros educativos

José Miguel Aparicio Oviedo
ETIG

Consultor: Manel Rella Ruiz

Junio 2011

Índice

1. Introducción

1.1 Justificación y aportación

Este trabajo de fin de carrera (TFC) constituye el broche a muchos años de esfuerzo y sacrificio en la búsqueda de esa meta personal que es convertirse en ingeniero técnico.

Después de la formación adquirida en las distintas etapas de la carrera llega el momento de ponerlas todas en conjunto y hacer un esfuerzo global para poder llegar a una solución íntegra que aporte un conocimiento genérico de los proyectos informáticos y su ciclo de vida.

En este caso, he elegido el área de Bases de datos ya que es un tema que me apasiona especialmente y con el que me encuentro cómodo trabajando. Espero exprimir todo mi conocimiento sobre los proyectos informáticos en general y sobre el entorno de las bases de datos en particular para conseguir un producto satisfactorio.

Por lo tanto, este TFC se justifica en la necesidad por mi parte de realizar un proyecto informático completo que aporte una visión conjunta de todas las disciplinas de la carrera pero centrada en el mundo de las bases de datos.

1.2 Objetivos

El objetivo principal de este trabajo es el aprendizaje. Aprendizaje de la metodología adecuada para realizar un proyecto de considerable envergadura basado en los conocimientos adquiridos durante la carrera.

Pero también es un reto ante el que hay que demostrar que se está preparado para afrontar un caso real de trabajo. Una manera de demostrar que se está preparado para afrontar el mundo laboral que aguarda al finalizar los estudios.

Otro de los grandes objetivos consiste en afianzar los conocimientos de bases de datos adquiridos previamente en las asignaturas de esta área.

Centrándonos en los objetivos específicos de este TFC y dejando un poco de lado los objetivos más abstractos, nuestra meta consiste en entregar un producto completamente funcional e independiente realizado en base a los requisitos aportados por el consultor (que actúa como cliente) en el enunciado del mismo.

Para ello tendremos que tener al final del semestre los siguientes documentos:

- Memoria
- Presentación

- Producto

Se detallan cada uno de ellos en el apartado 1.5 del presente documento.

Todo ello se irá realizando, con el apoyo del cliente, conforme a la planificación especificada en el apartado 1.4 de este mismo documento.

1.3 Enfoque y metodología

Para realizar el trabajo necesitaremos una planificación que abarque el tiempo que tenemos disponible. Esta planificación se dividirá en las distintas fases del proyecto que serán:

- Análisis: Realizando una lectura comprensiva de los requisitos expuestos en el enunciado y resolviendo todas las dudas posibles con el cliente mediante comunicación por el foro de la asignatura o mail se llevará a cabo un análisis detallado de la propuesta para erigir unos sólidos cimientos en los que basar las siguientes fases del proyecto.

- Diseño: Una vez realizado el análisis se procederá a plasmar en el diseño conceptual todos los conceptos adquiridos en la fase anterior. Para ello se utilizarán diagramas UML que nos permitan tener la idea completa del aspecto y las interacciones que tendrá la base de datos que necesitamos desarrollar.

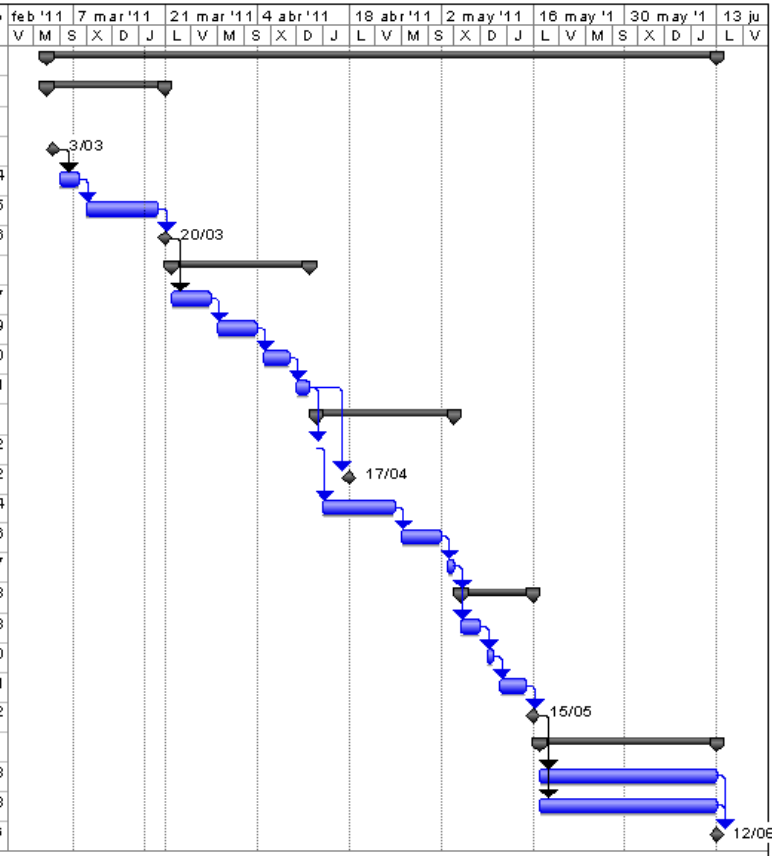
- Implementación: Después se transformará ese diseño en algo tangible mediante la implementación. Realizaremos el desarrollo de los scripts necesarios para la creación de la base de datos y todos sus contenidos asociados reflejados en el diseño. También se crearán los procedimientos almacenados y disparadores necesarios siguiendo la metodología recomendada por el cliente con tratamiento de excepciones y logging. Usaremos el lenguaje PL/SQL y el sistema de gestión de bases de datos ORACLE en esta fase.

- Pruebas: Finalmente prepararemos una batería de datos y con ellos realizaremos una serie de pruebas previamente documentadas que nos permitan encontrar posibles errores en el diseño o codificación de la base de datos y nos confirmen finalmente, una vez subsanados, que la funcionalidad que pretende el cliente se cumple en el producto creado.

1.4 Planificación

Para simplificar, la siguiente planificación se basa en una media diaria de 1.5 horas de trabajo durante todos los días de la semana lo que nos da un total de 153 horas totales en los 102 días previstos.

Id	Nombre de tarea	Duración	Comienzo	Fin	Predeceso
1	TFC BD	103 días	mié 2/03/11	dom 12/06/11	
2	Planificación	18 días	mié 2/03/11	dom 20/03/11	
3	Instalación Entorno	1 día	mié 2/03/11	mié 2/03/11	
4	Recogida Requisitos	0 días	jue 3/03/11	jue 3/03/11	
5	Lectura y Visión Global	4 días	vie 4/03/11	lun 7/03/11	4
6	Elaboración Plan de Trabajo	12 días	mar 8/03/11	sáb 19/03/11	5
7	Entrega PAC1	0 días	dom 20/03/11	dom 20/03/11	6
8	Diseño	22 días	lun 21/03/11	lun 11/04/11	
9	Análisis Requisitos	7 días	lun 21/03/11	dom 27/03/11	7
10	Diseño Conceptual	7 días	lun 28/03/11	dom 3/04/11	9
11	Diseño Lógico	5 días	lun 4/04/11	vie 8/04/11	10
12	Diseño Físico	3 días	sáb 9/04/11	lun 11/04/11	11
13	Desarrollo	22 días	mar 12/04/11	mar 3/05/11	
14	Scripts de creación de BD	1 día	mar 12/04/11	mar 12/04/11	12
15	Entrega PAC2	0 días	dom 17/04/11	dom 17/04/11	12
16	Procedimientos almacenados	12 días	mié 13/04/11	dom 24/04/11	14
17	Módulo Estadístico	7 días	lun 25/04/11	dom 1/05/11	16
18	Documentación	2 días	lun 2/05/11	mar 3/05/11	17
19	Pruebas	11 días	mié 4/05/11	dom 15/05/11	18
20	Diseño Juego Pruebas	4 días	mié 4/05/11	sáb 7/05/11	18
21	Importación de datos	2 días	dom 8/05/11	lun 9/05/11	20
22	Pruebas	5 días	mar 10/05/11	sáb 14/05/11	21
23	Entrega PAC3	0 días	dom 15/05/11	dom 15/05/11	22
24	Memoria y Presentación	28 días	lun 16/05/11	dom 12/06/11	
25	Elaboración Memoria	28 días	lun 16/05/11	dom 12/06/11	23
26	Elaboración Presentación	28 días	lun 16/05/11	dom 12/06/11	23
27	Entrega FINAL	0 días	dom 12/06/11	dom 12/06/11	25;26



Proyecto: TFC
Fecha: vie 18/03/11

Tarea
Progreso
Hito
Resumen
Tarea resumida
Hito resumido
Progreso resumido
División
Tareas externas
Resumen del proyecto
Agrupar por síntesis
Fecha límite

	Nombre de tarea	Duración	Comienzo	Fin	deceso
1	**TFC BD**	**103 días**	**mié 2/03/11**	**dom 12/06/11**	
2	**Planificación**	**18 días**	**mié 2/03/11**	**dom 20/03/11**	
3	Instalación Entorno	1 día	mié 2/03/11	mié 2/03/11	
4	Recogida Requisitos	0 días	jue 3/03/11	jue 3/03/11	
5	Lectura y Visión Global	4 días	vie 4/03/11	lun 7/03/11	4
6	Elaboración Plan de Trabajo	12 días	mar 8/03/11	sáb 19/03/11	5
7	Entrega PAC1	0 días	dom 20/03/11	dom 20/03/11	6
8	**Diseño**	**22 días**	**lun 21/03/11**	**lun 11/04/11**	
9	Análisis Requisitos	7 días	lun 21/03/11	dom 27/03/11	7
10	Diseño Conceptual	7 días	lun 28/03/11	dom 3/04/11	9
11	Diseño Lógico	5 días	lun 4/04/11	vie 8/04/11	10
12	Diseño Físico	3 días	sáb 9/04/11	lun 11/04/11	11
13	**Desarrollo**	**22 días**	**mar 12/04/11**	**mar 3/05/11**	
14	Scripts de creación de BD	1 día	mar 12/04/11	mar 12/04/11	12
15	Entrega PAC2	0 días	dom 17/04/11	dom 17/04/11	12
16	Procedimientos almacenados	12 días	mié 13/04/11	dom 24/04/11	14
17	Módulo Estadístico	7 días	lun 25/04/11	dom 1/05/11	16
18	Documentación	2 días	lun 2/05/11	mar 3/05/11	17
19	**Pruebas**	**11 días**	**mié 4/05/11**	**dom 15/05/11**	**18**
20	Diseño Juego Pruebas	4 días	mié 4/05/11	sáb 7/05/11	18
21	Importación de datos	2 días	dom 8/05/11	lun 9/05/11	20
22	Pruebas	5 días	mar 10/05/11	sáb 14/05/11	21
23	Entrega PAC3	0 días	dom 15/05/11	dom 15/05/11	22
24	**Memoria y Presentación**	**28 días**	**lun 16/05/11**	**dom 12/06/11**	
25	Elaboración Memoria	28 días	lun 16/05/11	dom 12/06/11	23
26	Elaboración Presentación	28 días	lun 16/05/11	dom 12/06/11	23
27	Entrega FINAL	0 días	dom 12/06/11	dom 12/06/11	25;26

Pasamos a describir cada una de las tareas coincidiendo su número con el que aparece en la imagen de arriba:

1. TFC BD: Proyecto Trabajo final de carrera de Bases de datos.

2. Planificación: Se comienza la planificación del trabajo.

3. Instalación del Entorno: Se realiza la instalación del entorno de desarrollo necesario para realizar las tareas. Exactamente, se compone de una máquina virtual VMWare con sistema operativo Windows XP en la que instalaremos el Oracle Express 10 para realizar las tareas necesarias de base de datos y el MagicDraw como aplicación necesaria para los diagramas UML.

4. Recogida de requisitos: Publicación del enunciado en el que aparecen los requisitos funcionales.

5. Lectura y visión global: Consiste en leer varias veces los requisitos y la definición del proyecto para obtener una comprensión suficiente que permita

crear un mapa general el mismo y realizar el plan de trabajo que regirá su desarrollo.

6. Elaboración Plan Trabajo: Elaborar este documento para planificar el ciclo de vida de desarrollo del proyecto y tener una guía durante el mismo de las tareas a realizar y el tiempo que ha de llevar cada una de ellas. El plan de trabajo es básico para la comprensión total del proyecto y su puesta en funcionamiento.

7. Entrega PAC1: Entrega de la versión final del plan de trabajo.

8. Diseño: Comienzo del diseño del proyecto.

9. Análisis de Requisitos: Determinación de las necesidades y condiciones que ha de contemplar la base de datos tomando en cuenta los requisitos aportados en el enunciado. Resolución de dudas mediante preguntas al cliente.

10. Diseño Conceptual: Partiendo del análisis, elaboración del esqueleto de la base de datos que describa el contenido de su información. En esta fase, el diseño se realiza a alto nivel, totalmente independiente del SGBD utilizado y mediante el modelo entidad-relación plasmado en diagramas UML.

11. Diseño Lógico: Convertiremos el diseño conceptual en un esquema lógico que se adapte a nuestro SGBD (ORACLE)

12. Diseño Físico: En esta etapa, se parte del esquema lógico global obtenido y se obtiene una descripción de la implementación de la base de datos. Esta descripción incluye las estructuras de almacenamiento y los métodos de acceso que se utilizarán para conseguir un acceso eficiente a los datos.

13. Desarrollo: Inicio de la fase de desarrollo.

14. Scripts de creación de BD: Se prepararán los scripts con el código PL/SQL necesario para la creación de la base de datos, con sus entidades y relaciones siguiendo las especificaciones del diseño previamente realizado

15. Entrega PAC2: Entrega del diseño completo de la base de datos al cliente.

16. Procedimientos Almacenados: Se prepararán los scripts con el código PL/SQL necesario para la creación de los procedimientos almacenados de la base de datos siguiendo las especificaciones del diseño previamente realizado. Se crearán procedimientos para el alta, baja y modificación de alumnos, profesores, cursos, asignaturas y calendario escolar, así como de las amonestaciones y sanciones con sus tipologías. También realizaremos

procedimientos de consulta. Todos incorporarán el sistema de log y de tratamiento de excepciones explicado en los requerimientos.

17. Módulo estadístico: Se diseñarán las tablas necesarias para almacenar los datos que sirvan para tener una consulta directa de los datos estadísticos requeridos.

18. Documentación: Se realizará la documentación de los procedimientos almacenados para guiar a los futuros desarrolladores que tenga que usarlos y/o interactuar con ellos y así que puedan hacerlo sin necesidad de conocer el código fuente.

19. Pruebas: Inicio de la fase de pruebas.

20. Diseño juego de pruebas: Se diseñará un juego de pruebas que confirme que la base de datos y sus procedimientos funcionan adecuadamente y cumplen todas las reglas de negocio.

21. Importación de datos: Creación de un script que rellene las tablas con los datos necesarios para realizar las pruebas.

22. Pruebas: Creación de los scripts que se ejecutarán para probar la base de datos. Prueba de la base de datos y corrección de la misma hasta que el juego de pruebas sea ejecutado satisfactoriamente en su totalidad.

23. Entrega PAC3: Entrega de la base de datos como producto.

24. Memoria y presentación final: Comienzo de la elaboración de la memoria y la presentación.

25. Elaboración memoria: Adaptación y revisión de la memoria, así como agregación de los apartados que falten para la entrega final.

26. Elaboración presentación final: Creación de una presentación para el tribunal de evaluación que sintetice la razón de ser de este trabajo y como ha sido realizado.

27. Entrega FINAL: Entrega del proyecto completo y fin del trabajo.

Esta planificación general es flexible y se apoya en la evolución, por lo que puede ser modificada durante la marcha dependiendo del ritmo de trabajo, siempre con la aprobación previa del cliente.

2. Análisis y diseño

2.1 Descripción del trabajo

El TFC consiste en diseñar e implementar una base de datos para ser utilizada por la Generalitat de Cataluña y que permita a esta, mediante una aplicación que se conecte a ella y sirva de capa de negocio e interfaz con el usuario, gestionar las amonestaciones y sanciones en los centros educativos de la comunidad. El desarrollo de la aplicación en sí queda fuera del ámbito de este trabajo por lo que solo se entregará la base de datos pero con los procedimientos almacenados, disparadores y otras herramientas necesarias para su futura integración con la aplicación.

Para realizarlo el consultor simula ser el cliente que hace la petición a nuestra empresa informática. Partiendo de los requisitos que este proporciona se deberá trabajar para obtener el mejor producto posible de cara a la evaluación final realizada por el tribunal correspondiente.

A grandes rasgos, los requisitos generales consisten en desarrollar una base de datos que nos permita almacenar e interactuar con la información que se genere en el marco que se describe a continuación: La Generalitat dispone de distintos centros educativos. En cada uno de estos centros se impartirán distintos cursos en los que se matriculan alumnos y en los que se enseñan una serie de asignaturas. Estas, son impartidas por profesores. Mediante un calendario escolar se organizan los horarios de los cursos para cada asignatura. Cuando un alumno comete un acto de indisciplina, un profesor puede aplicarle una amonestación de un tipo determinado. Los tipos están gestionados por los profesores que los agregan y eliminan según la necesidad. Existirán una serie de reglas por las que se comprobará si la amonestación conlleva algún tipo de sanción. Las sanciones y las reglas para aplicarlas también serán gestionadas por los profesores.

Aparte de almacenar y consultar toda la información anterior también será necesario desarrollar un módulo de la base de datos que permita realizar consultas complejas (detalladas más adelante) de una forma optimizada.

2.2 Requisitos Funcionales

- R1: Se almacenarán los datos de los alumnos y los profesores de cada centro.

- R2: Se almacenarán los cursos y las asignaturas que se impartan.

- R3: El calendario escolar debe almacenar:
    - Curso
    - Asignatura
    - Horario de la asignatura (Día de la semana y hora)
    - Días festivos

- R4: Una amonestación debe almacenar:
    - Alumno amonestado
    - Fecha y hora
    - Curso
    - Asignatura
    - Profesor responsable
    - Tipo
    - Gravedad
    - Si se ha comunicado a los padres

- R5-R6: Los tipos de amonestaciones serán definidos por los profesores.

- R7: Se debe permitir introducir una sanción.

- R8: Cada sanción debe ir ligada a un tipo de amonestación definido.

- R9: Los tipos de sanciones serán definidos por los profesores.

- R10: Se han de almacenar las reglas que activan las sanciones automáticamente. Siempre se tendrá en cuenta para esto la forma sintáctica siguiente:

  Si ([Amonestación] [<|>|=] [Valor numérico definido]) => "descripción de sanción"

  Se ha de guardar la fecha de la sanción.

- R11: Se tendrá que saber a que curso pertenece cada alumno, que profesores imparten cada asignatura y que profesores son responsables de cada curso.

- R12: Se almacenarán los días y las horas de atención a los padres y a los alumnos por parte de los profesores.

- R13: Se debe permitir conocer las amonestaciones y sanciones que se le han asignado a cada alumno.

- R14: Se deben permitir las siguientes modificaciones en los datos:

    - Alta, baja y modificación de alumnos.
    - Alta, baja y modificación de profesores.
    - Alta, baja y modificación de cursos.
    - Alta, baja y modificación de asignaturas.
    - Alta, baja y modificación del calendario escolar.
    - Alta, baja y modificación de amonestaciones.
    - Alta, baja y modificación de sanciones.
    - Alta, baja y modificación de tipos de amonestaciones.
    - Alta, baja y modificación de tipos de sanciones.

Se deben permitir también las siguientes consultas:

- Todas las amonestaciones impuestas indicando su información básica.
- Todos los alumnos de un curso indicando su información básica.
- Todos los tipos de amonestaciones y sanciones.
- Todas las amonestaciones y sanciones de un alumno.

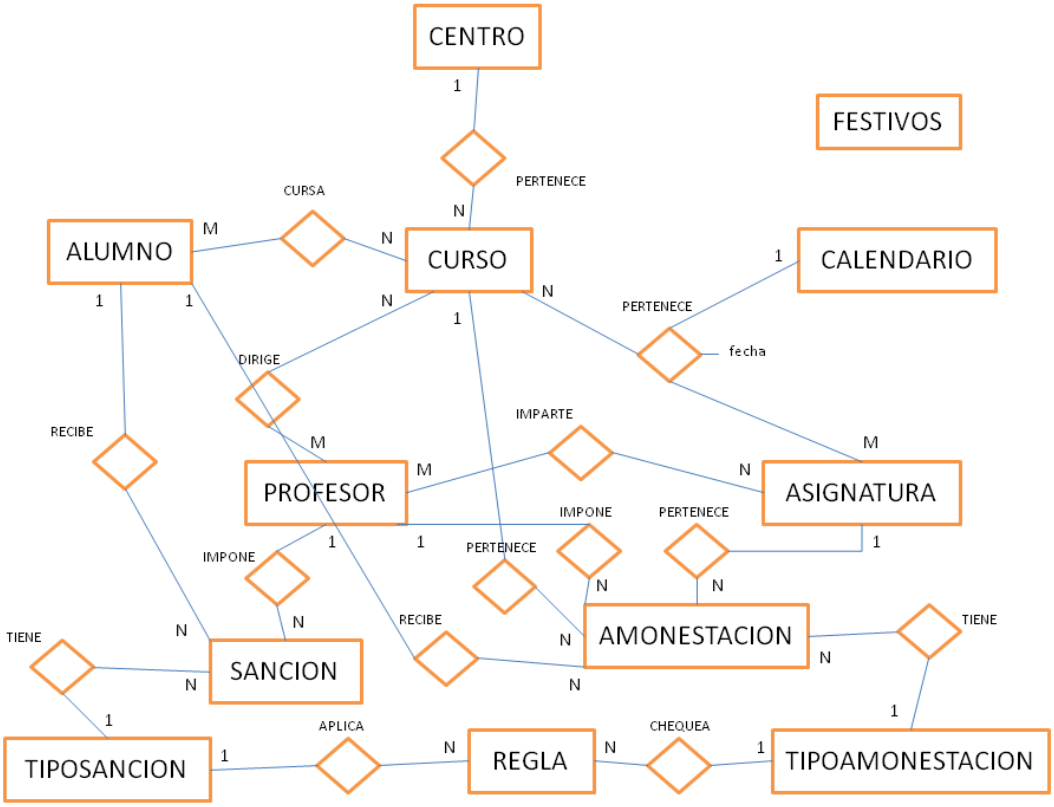
- R15: Se debe crear un módulo estadístico que permita la consulta de una serie de informes en tiempo constante 1. Las consultas necesarias son las siguientes:

  - Número de amonestaciones por alumno.
  - Número de sanciones por alumno y año.
  - Media de amonestaciones por profesor y año.
  - Número de sanciones por curso y año.
  - Nombre del alumno más sancionado en un año dado.
  - Nombre del profesor más amonestador por curso.
  - Media de sanciones que tienen los alumnos por curso.
  - Número de alumnos que no tienen ninguna amonestación.

2.3 Diseño conceptual

El diseño conceptual de la base de datos nos va a permitir obtener la estructura de la información que conformará la misma, independientemente de la tecnología que se vaya a utilizar. Para ello usamos el diagrama Entidad-Relación que construimos con los datos recabados en el análisis.

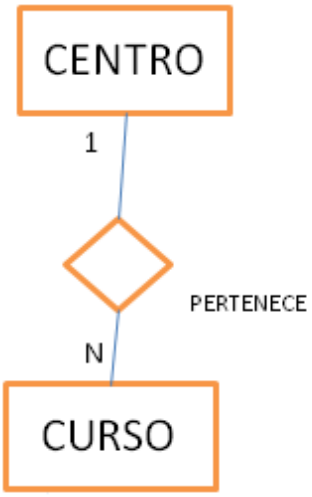
2.3.1 Diagrama Entidad-Relación



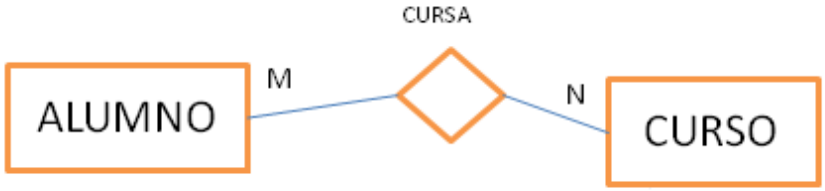
2.3.2 Entidades

- Centro
- Curso
- Alumno
- Profesor
- Asignatura
- Sanción
- Amonestación
- Calendario
- Festivo
- Tipo de Sanción
- Tipo de Amonestación
- Regla

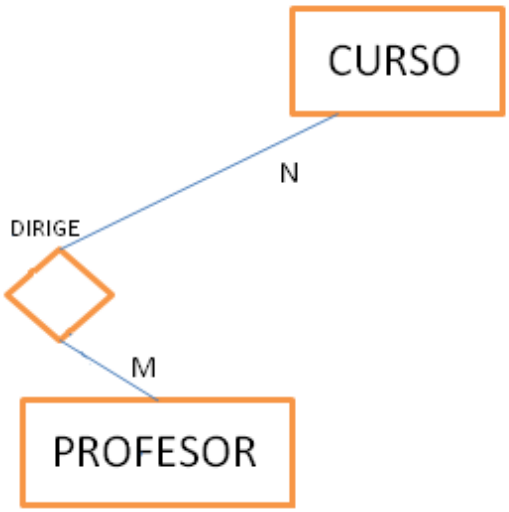
2.3.3 Relaciones



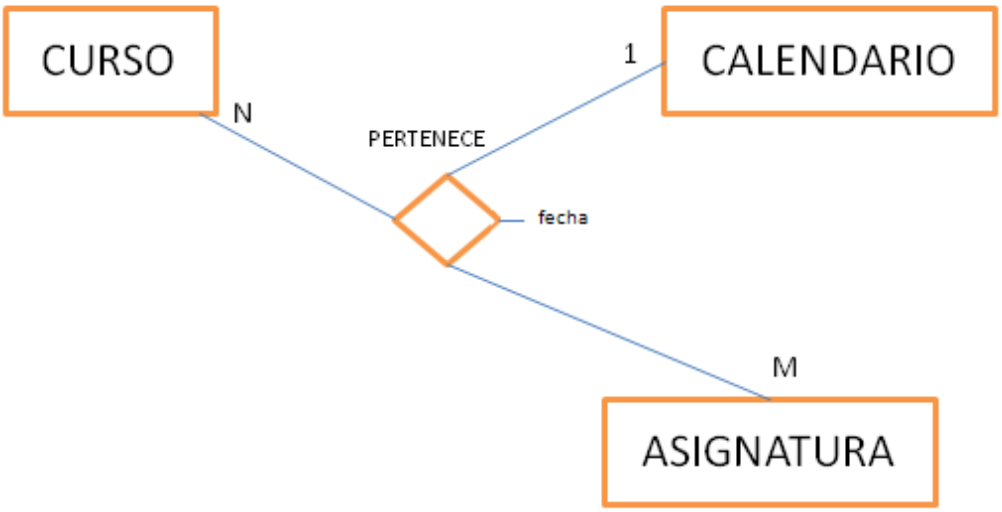
- Un centro imparte N cursos
- Un curso es impartido en 1 centro



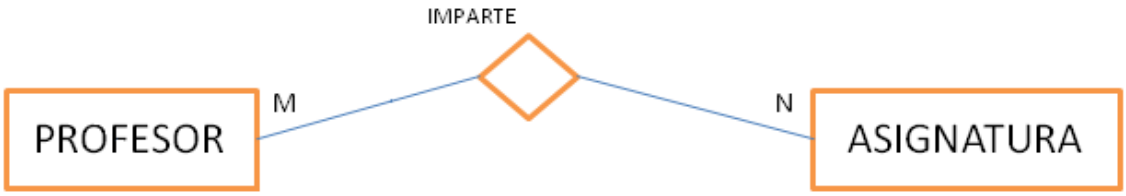
- Un alumno cursa N cursos
- Un curso es cursado por M alumnos



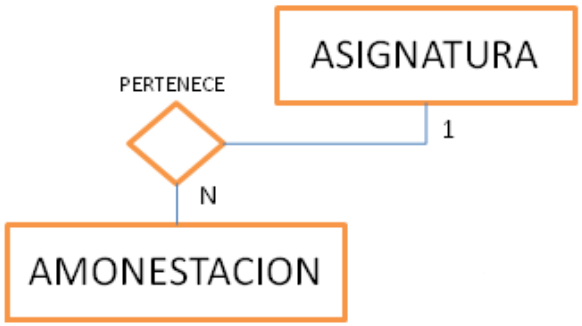
- Un profesor es responsable/dirige N cursos.
- Un curso puede ser dirigido por M profesores.



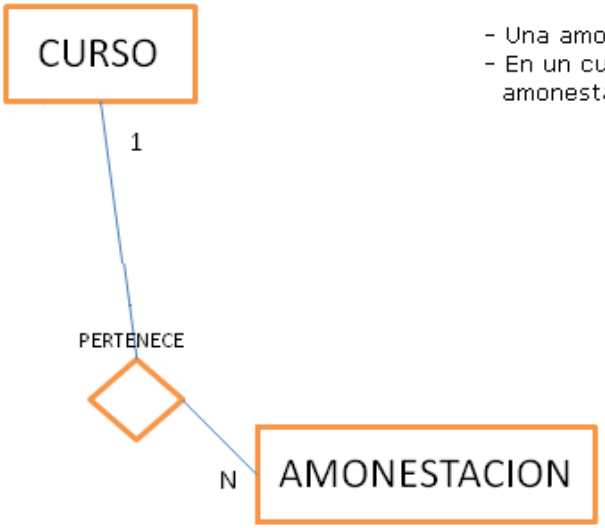
- Un curso se compone de M asignaturas.
- Una asignatura puede pertenecer a N cursos.
- Una asignatura de un curso tiene 1 calendario con un atributo fecha que nos permite reflejar todas las fechas en las que se imparte una asignatura dentro de un curso en cada calendario.



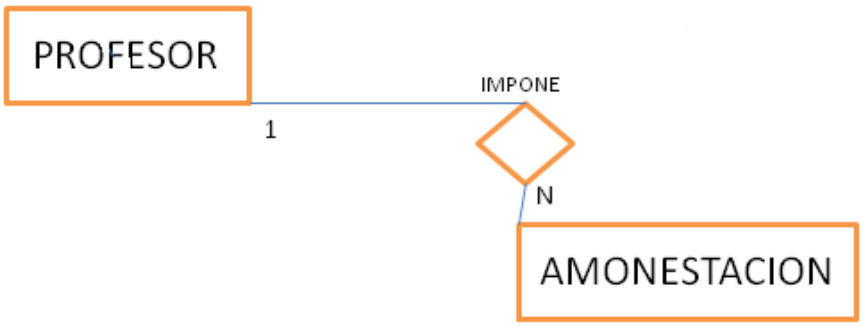
- Un profesor imparte N asignaturas.
- Una asignatura puede ser impartida por M profesores.



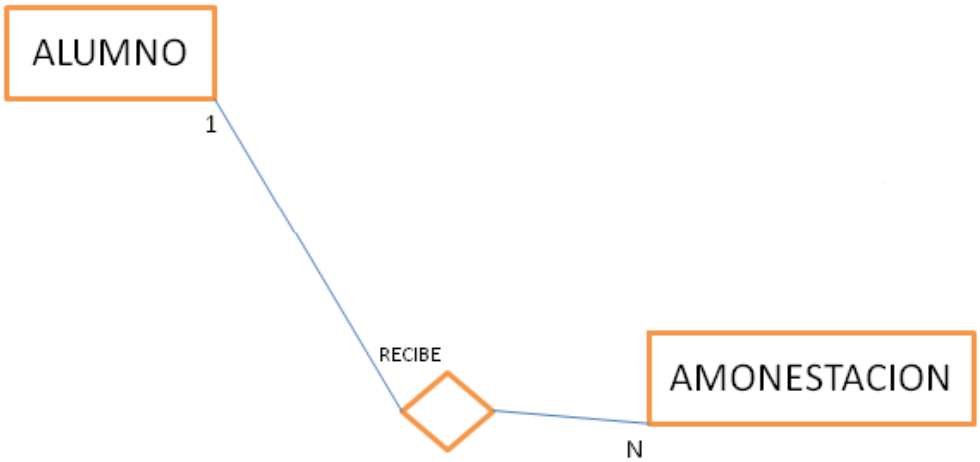
- Una amonestación es impuesta en una asignatura.
- En una asignatura pueden producirse N amonestaciones



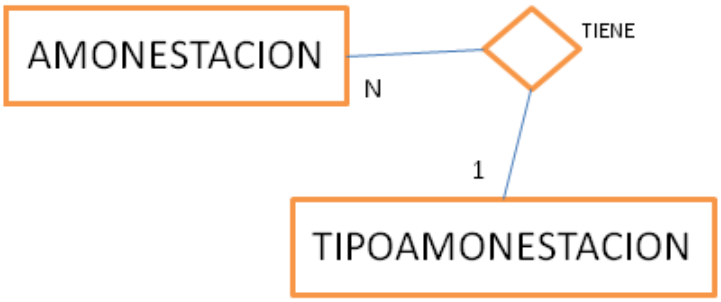
- Una amonestación es impuesta en un curso.
- En un curso pueden producirse N amonestaciones



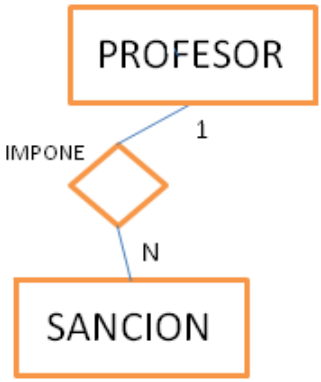
- Una amonestación es impuesta por 1 profesor.
- Un profesor impone N amonestaciones.



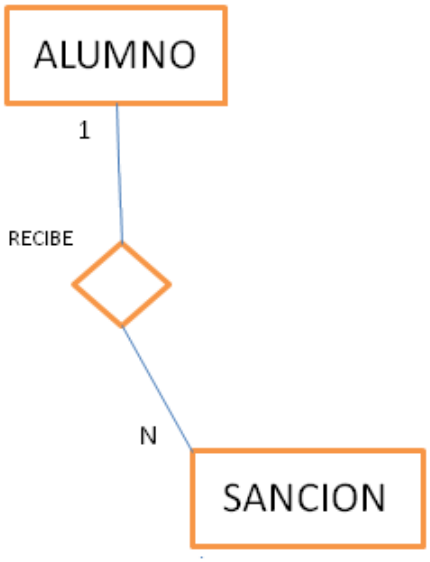
- Una amonestación es aplicada a un alumno.
- Un alumno puede recibir N amonestaciones.



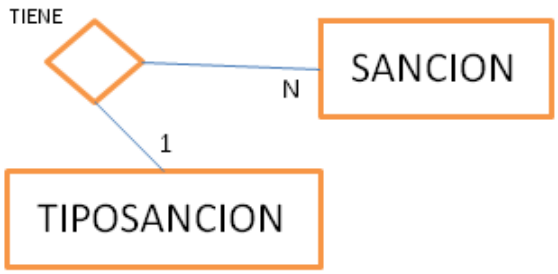
- Una amonestación tiene un tipo de amonestación
- Un tipo de amonestación puede aplicársele a N amonestaciones



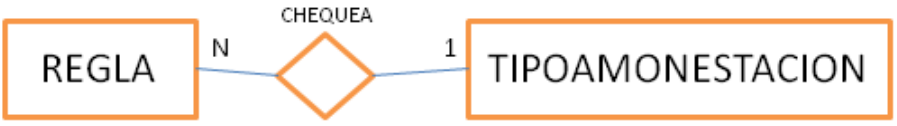
- Una sanción es impuesta por un profesor.
- Un profesor puede imponer N saciones



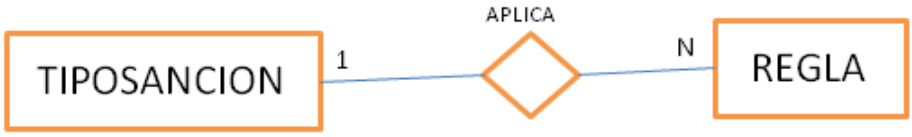
- Una sanción es impuesta a un alumno.
- Un alumno puede recibir N sanciones.



- Una sanción tiene un tipo de sanción.
- Un tipo de sanción puede aplicársele a N sanciones.



- Una regla chequea un tipo de amonestación.
- Un tipo de amonestación puede ser chequeado en N reglas.



- Una regla aplica a un tipo de sanción.
- Un tipo de sanción puede ser aplicada en N reglas.

2.4 Diseño lógico

En esta fase ajustaremos el diseño conceptual que hemos creado al SGBD que vamos a utilizar, ORACLE, que es relacional. Así pues, cada entidad la transformamos en una relación por lo que nos quedan las siguientes:

- CENTROS(idcentro, nombre, dirección)
- CURSOS(idcurso, nombre, fechainicio, fechafin)
- ALUMNOS(idalumno, nombre, apellidos, dirección, dni)
- PROFESORES(idprofesor, nombre, apellidos, dirección, dni)
- ASIGNATURAS(idasignatura, nombre, creditos)
- CALENDARIOESCOLAR(idcalendario, idcurso, idasignatura, fecha, horainicio, horafin)
- FESTIVOS(fecha, descripcion)
- TIPOAMONESTACIONES(idtipo, descripción)
- TIPOSANCIONES(idtipo, descripción)
- AMONESTACIONES(idamonestacion, fecha, gravedad, comunicado)
- SANCIONES(idsancion, fecha, resolucion)
- REGLASSANCIONES(idregla, operando, valornumerico)

Ahora pasamos a resolver la interrelaciones:

- Interrelaciones binarias 1:N

- CENTRO-CURSO: La entidad del lado N, que es CURSO ,contendrá un campo con la clave primaria de CENTRO, que hará de clave foránea para relacionar ambas entidades.
- ASIGNATURA-AMONESTACION: La entidad del lado N, que es AMONESTACION, contendrá un campo con la clave primaria de ASIGNATURA, que hará de clave foránea para relacionar ambas entidades.
- CURSO-AMONESTACION: La entidad del lado N, que es AMONESTACION, contendrá un campo con la clave primaria de CURSO, que hará de clave foránea para relacionar ambas entidades.
- PROFESOR-AMONESTACION: La entidad del lado N, que es AMONESTACION, contendrá un campo con la clave primaria de PROFESOR, que hará de clave foránea para relacionar ambas entidades.
- ALUMNO-AMONESTACION: La entidad del lado N, que es AMONESTACION, contendrá un campo con la clave primaria de ALUMNO, que hará de clave foránea para relacionar ambas entidades.
- TIPOAMONESTACION-AMONESTACION: La entidad del lado N, que es AMONESTACION, contendrá un campo con la clave primaria de TIPOAMONESTACION, que hará de clave foránea para relacionar ambas entidades.
- PROFESOR-SANCION: La entidad del lado N, que es SANCION, contendrá un campo con la clave primaria de PROFESOR, que hará de clave foránea para relacionar ambas entidades.
- ALUMNO- SANCION: La entidad del lado N, que es SANCION, contendrá un campo con la clave primaria de ALUMNO, que hará de clave foránea para relacionar ambas entidades.
- TIPOSANCION - SANCION: La entidad del lado N, que es SANCION, contendrá un campo con la clave primaria de TIPOSANCION, que hará de clave foránea para relacionar ambas entidades.
- TIPOAMONESTACION-REGLA: La entidad del lado N, que es REGLA, contendrá un campo con la clave primaria de TIPOAMONESTACION, que hará de clave foránea para relacionar ambas entidades.
- TIPOSANCION - REGLA: La entidad del lado N, que es REGLA, contendrá un campo con la clave primaria de TIPOSANCION, que hará de clave foránea para relacionar ambas entidades.

- Interrelaciones binarias M:N
  - ALUMNO-CURSO: Dan lugar a una nueva relación ALUMNOSCURSO que tendrá como clave primaria las claves primarias de ambas relaciones.

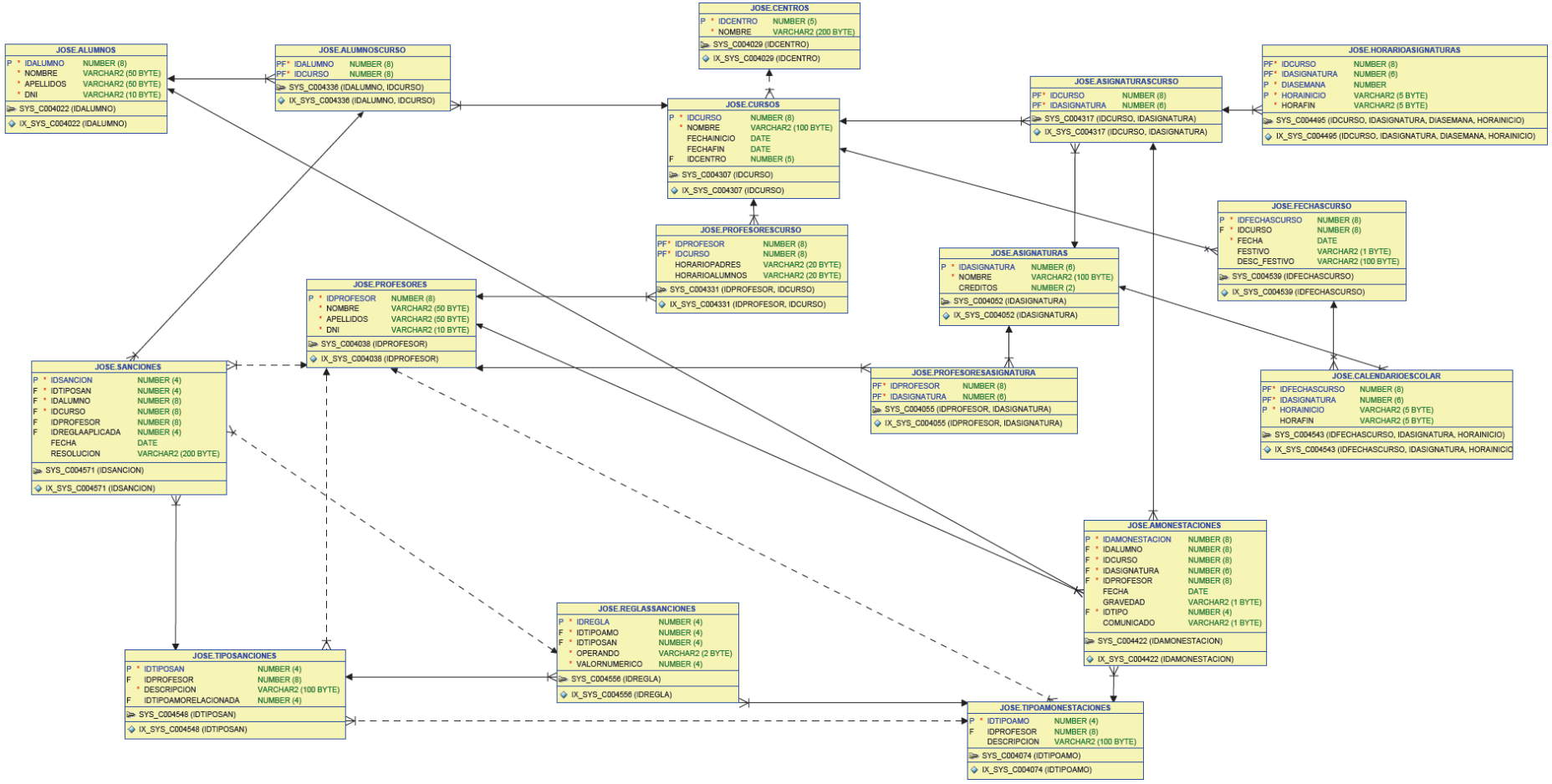
    - PROFESOR-CURSO: Dan lugar a una nueva relación PROFESORESCURSO que tendrá como clave primaria las claves primarias de ambas relaciones.
    - PROFESOR-ASIGNATURA: Dan lugar a una nueva relación PROFESORESASIGNATURA que tendrá como clave primaria las claves primarias de ambas relaciones.

- Interrelaciones ternarias 1:M:N
    - CALENDARIO-ASIGNATURA-CURSO: En este caso vamos a resolverla con una nueva relación que una asignatura y curso ASIGNATURASCURSO que contenga como clave primaria las claves primarias de ambas relaciones y después vamos a crear una relación CALENDARIOESCOLAR que contenga en su clave primaria esta clave primaria de la nueva relación creada y también el atributo fecha.

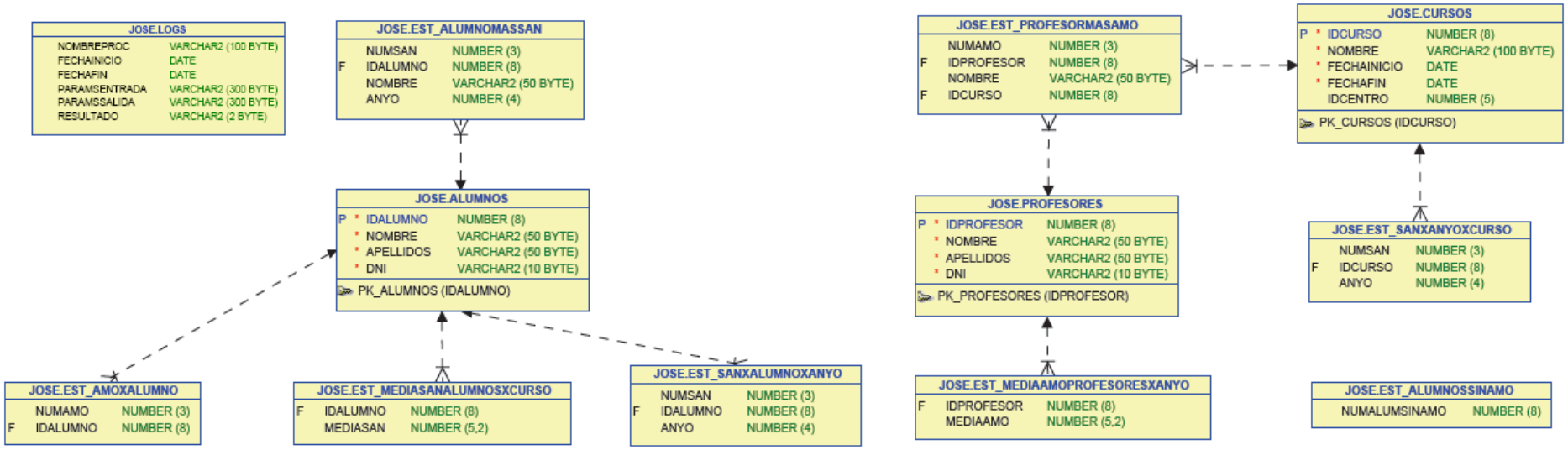
Finalmente el diseño lógico queda con las siguientes relaciones y atributos:

- CENTROS(idcentro, nombre, dirección)
- CURSOS(idcurso, nombre, fechainicio, fechafin, idcentro)
- ALUMNOS(idalumno, nombre, apellidos, dirección, dni)
- PROFESORES(idprofesor, nombre, apellidos, dirección, dni)
- ASIGNATURAS(idasignatura, nombre, creditos)
- CALENDARIOESCOLAR(idcalendario, idcurso, idasignatura, fecha, horainicio, horafin)
- FESTIVOS(fecha, descripcion)
- TIPOAMONESTACIONES(idtipo, idprofesor, descripción)
- TIPOSANCIONES(idtipo, idprofesor, descripción, idamonestacionrelacionada)
- AMONESTACIONES(idamonestacion, idcurso, idalumno, idasignatura, idprofesor, fecha, gravedad, idtipo, comunicado)
- SANCIONES(idsancion, idtiposan, idalumno, idcurso, idreglaaplicada, fecha, resolucion)
- REGLASSANCIONES(idregla, idtipoamo, idtiposan, operando, valornumerico)
- ASIGNATURASCURSO(idcurso, idasignatura)
- PROFESORESASIGNATURA(idprofesor, idasignatura)
- PROFESORESCURSO(idprofesor, idcurso, horariopadres, horarioalumnos)
- ALUMNOSCURSO(idalumno, idcurso)

2.4.1 Modelo Entidad Relación tablas principales



2.4.2 Modelo Entidad Relación tablas módulo estadístico y log



2.5 Diseño físico

Al trabajar con la versión de ORACLE Express la base de datos EX ya está creada por defecto y no nos permite crear más, así que utilizaremos esta, pero en el entorno de producción habría que seguir varios pasos para crear la base de datos una vez instalada la instancia ORACLE.

- Asignar a la variable de entorno ORACLE_SID como valor, el nombre de nuestra nueva base de datos.
- Crear un fichero de inicialización llamado initNOMBREDELABASEDEDATOS.ora que contendrá todas las características de la base de datos. Se puede obtener más información acerca de este fichero en la documentación de ORACLE.
- Conectar a la instancia de la base de datos que tenemos instalada como SYSDBA.
- Levantar la instancia sin montar ninguna base de datos con el comando STARTUP NOMOUNT
- Ejecutar la sentencia CREATE DATABASE con los parámetros correspondientes.
- Crear los diferentes TABLESPACES que necesitaremos mediante la sentencia CREATE TABLESPACE.

Una vez hecho esto ya estaríamos en las mismas condiciones así que ahora pasamos a crear un tablespace.

```
CREATE TABLESPACE tfc DATAFILE 'c:\oraclexe\oradata\xe\tfc.dbf' SIZE
100M EXTENT MANAGEMENT LOCAL AUTOALLOCATE
```

Creamos un usuario para la base de datos.

```
CREATE USER Jose IDENTIFIED BY 'jose' DEFAULT TABLESPACE tfc
```

Aunque habría que crear distintos roles para cada tipo de usuario (administradores, profesores, etc.) para simplificar, no los usaremos así que directamente le daremos privilegios de SYSDBA al usuario creado para que no tenga restricciones que nos afecten.

```
GRANT SYSDBA TO Jose;
```

Ya tenemos la base de datos preparada para crear las tablas y triggers así que utilizando el diseño lógico, construimos las sentencias de creación y las ejecutamos. Para ello utilizaremos este script preparado para ser ejecutado desde sql*plus o cualquier aplicación de gestión de base de datos.


Tablas.sql

El siguiente paso es ejecutar las sentencias de creación de procedimientos almacenados. Para ello utilizaremos este script preparado para ser ejecutado desde sql*plus o cualquier aplicación de gestión de base de datos.

3. Implementación

Una vez terminado el diseño y con las tablas preparadas para recibir datos comenzamos la fase de implementación en la cual las rellenaremos. Vamos a explicar para que sirve y como funciona cada elemento de la base de datos que hemos creado en el paso anterior.

3.1 Tablas

3.1.1 Tablas de explotación

ALUMNOS: Nos permite almacenar los alumnos con sus datos personales como nos pide el requisito R1.
ALUMNOSCURSO: Nos permite relacionar los alumnos con el curso que van a cursar como nos pide el requisito R11.
AMONESTACIONES: Nos permite guardar todas las amonestaciones impuestas a los alumnos como nos pide el requisito R4.
ASIGNATURAS: Nos permite almacenar las asignaturas con sus datos como nos pide el requisito R2.
ASIGNATURASCURSO: Nos permite almacenar las asignaturas que le corresponden a cada curso como nos pide el requisito R2.
CALENDARIOESCOLAR: Nos permite almacenar el calendario escolar de cada curso como nos pide el requisito R3.
CENTROS: Nos permite almacenar los centros educativos con sus datos como nos pide el requisito R1.
CURSOS: Nos permite almacenar los cursos con sus fechas de impartición como nos pide el requisito R2.
FECHASCURSO: Nos permite almacenar los datos de las fechas de cada curso y sus festivos como nos pide el requisito R3.
HORARIOASIGNATURAS: Nos permite almacenar los horarios de las asignaturas como nos pide el requisito R3.
HORARIOS: Nos permite almacenar los horarios y usarlos en el calendario escolar y en los horarios de atención de los profesores como nos piden los requisitos R3 y R12.
LOGS: Nos permite almacenar los logs de toda la aplicación.
PROFESORES: Nos permite almacenar los profesores y sus datos como nos pide el requisito R1.
PROFESORESASIGNATURA: Nos permite almacenar los profesores que imparten docencia en cada asignatura como nos pide el requisito R11.
PROFESORESCURSO: Nos permite saber de qué curso es responsable cada profesor como nos pide el requisito R11.
REGLASSANCIONES: Nos permite almacenar las reglas para aplicar sanciones como nos pide el requisito R10.
SANCIONES: Nos permite almacenar las sanciones impuestas a los alumnos como nos pide el requisito R8.
TIPOAMONESTACIONES: Nos permite almacenar los tipos de amonestaciones que se pueden aplicar como nos piden los requisitos R5 y R6.
TIPOSANCIONES: Nos permite almacenar los tipos de sanciones que se pueden

aplicar como nos piden el requisito R9.

3.1.2 Tablas del módulo estadístico

EST_ALUMNOMASSAN: Nos permite almacenar los alumnos más sancionados en un año dado. Esta tabla resuelve el requisito R15.5
EST_ALUMNOSSINAMO: Nos permite almacenar los alumnos que no tienen amonestaciones. Esta tabla resuelve el requisito R15.8
EST_AMOXALUMNO: Nos permite almacenar las amonestaciones por alumno. Esta tabla resuelve el requisito R15.1
EST_MEDIAAMOPROFESORESXANYO: Nos permite almacenar la media de amonestaciones que ponen los profesores por año. Esta tabla resuelve el requisito R15.3
EST_MEDIASANALUMNOSXCURSO: Nos permite almacenar la media de sanciones que reciben los alumnos por curso. Esta tabla resuelve el requisito R15.7
EST_PROFESORMASAMO: Nos permite almacenar el profesor más amonestador. Esta tabla resuelve el requisito R15.6
EST_SANXALUMNOXANYO: Nos permite almacenar las sanciones recibidas por los alumnos por año. Esta tabla resuelve el requisito R15.2
EST_SANXANYOXCURSO: Nos permite almacenar las sanciones por año y curso. Esta tabla resuelve el requisito R15.4

3.2 Vistas

Usaremos las vistas para simplificar el método de consulta. Tan solo realizando una consulta sobre cada una de ellas tendremos los datos requeridos. Cada una de estas vistas resuelve cada una de las consultas del requisito R14.

ALUMNOSXCURSO: La usamos para consultar los alumnos por curso.
AMOYSANXALUMNO: La usamos para consultar las amonestaciones y sanciones por alumno.
LISTADOAMONESTACIONES: La usamos para consultar el listado de amonestaciones.
TIPOSAMONESTACIONESYSANCIONES: La usamos para consultar el listado de todos los tipos de sanciones y amonestaciones

3.3 Triggers

3.3.1 Asignar ID a asignatura

3.3.1.1 Funcionalidad

El trigger NUEVAASIGNATURA permite asignar un nuevo ID cuando se inserta una asignatura.

3.3.1.2 Ejecución

El trigger se lanza cuando se inserta una nueva fila en la tabla ASIGNATURAS justo antes de confirmar su inserción.

3.3.1.3 Descripción

1. Si no se proporciona el ID de asignatura en el INSERT se le asigna como ID el siguiente número de la secuencia ASIGNATURAS_SEQ.

3.3.2 Asignar ID a alumno

3.3.2.1 Funcionalidad

El trigger se lanza cuando se inserta una nueva fila en la tabla ALUMNOS justo antes de confirmar su inserción

3.3.2.2 Ejecución

El trigger se lanza después de insertar una nueva fila en la tabla ALUMNOS.

3.3.2.3 Descripción

1. Si no se proporciona el ID del alumno en el INSERT se le asigna como ID el siguiente número de la secuencia ALUMNOS_SEQ.

3.3.3 Asignar ID a profesor

3.3.3.1 Funcionalidad

El trigger se lanza cuando se inserta una nueva fila en la tabla PROFESORES justo antes de confirmar su inserción

3.3.3.2 Ejecución

El trigger se lanza después de insertar una nueva fila en la tabla PROFESORES.

3.3.3.3 Descripción

1. Si no se proporciona el ID del profesor en el INSERT se le asigna como ID el siguiente número de la secuencia PROFESORES_SEQ.

3.3.4 Asignar ID a festivo

3.3.4.1 Funcionalidad

El trigger NUEVOFESTIVO permite asignar un nuevo ID cuando se inserta un festivo.

3.3.4.2 Ejecución

El trigger se lanza cuando se inserta una nueva fila en la tabla FESTIVOS justo antes de confirmar su inserción.

3.3.4.3 Descripción

1. Si no se proporciona el ID del festivo en el INSERT se le asigna como ID el siguiente número de la secuencia FESTIVOS_SEQ.

3.3.5 Asignar ID a centro

3.3.5.1 Funcionalidad

El trigger NUEVOCENTRO permite asignar un nuevo ID cuando se inserta un centro.

3.3.5.2 Ejecución

El trigger se lanza cuando se inserta una nueva fila en la tabla CENTROS justo antes de confirmar su inserción.

3.3.5.3 Descripción

1. Si no se proporciona el ID del centro en el INSERT se le asigna como ID el siguiente número de la secuencia CENTROS_SEQ.

3.3.6 Asignar ID a curso

3.3.6.1 Funcionalidad

El trigger NUEVOCURSO permite asignar un nuevo ID cuando se inserta un curso.

3.3.6.2 Ejecución

El trigger se lanza cuando se inserta una nueva fila en la tabla CURSOS justo antes de confirmar su inserción.

3.3.6.3 Descripción

1. Si no se proporciona el ID del curso en el INSERT se le asigna como ID el siguiente número de la secuencia CURSOS_SEQ.

3.3.7 Asignar ID a amonestación

3.3.7.1 Funcionalidad

El trigger NUEVAAMONESTACION permite asignar un nuevo ID cuando se inserta una amonestación.

3.3.7.2 Ejecución

El trigger se lanza cuando se inserta una nueva fila en la tabla AMONESTACIONES justo antes de confirmar su inserción.

3.3.7.3 Descripción

1. Si no se proporciona el ID de la amonestación en el INSERT se le asigna como ID el siguiente número de la secuencia AMONESTACIONES_SEQ.

3.3.8 Asignar ID a sanción

3.3.8.1 Funcionalidad

El trigger NUEVASANCION permite asignar un nuevo ID cuando se inserta una sanción.

3.3.8.2 Ejecución

El trigger se lanza cuando se inserta una nueva fila en la tabla AMONESTACIONES justo antes de confirmar su inserción.

3.3.8.3 Descripción

1. Si no se proporciona el ID de la amonestación en el INSERT se le asigna como ID el siguiente número de la secuencia AMONESTACIONES_SEQ.

3.3.9 Asignar ID a regla

3.3.9.1 Funcionalidad

El trigger NUEVAREGLA permite asignar un nuevo ID cuando se inserta una regla.

3.3.9.2 Ejecución

El trigger se lanza cuando se inserta una nueva fila en la tabla REGLAS justo antes de confirmar su inserción.

3.3.9.3 Descripción

1. Si no se proporciona el ID de la regla en el INSERT se le asigna como ID el siguiente número de la secuencia REGLAS_SEQ.

3.3.10 Asignar ID a festivo

3.3.10.1 Funcionalidad

El trigger NUEVATIPOAMO permite asignar un nuevo ID cuando se inserta un nuevo tipo de amonestación.

3.3.10.2 Ejecución

El trigger se lanza cuando se inserta una nueva fila en la tabla TIPOAMONESTACIONES justo antes de confirmar su inserción.

3.3.10.3 Descripción

1. Si no se proporciona el ID del tipo de amonestación en el INSERT se le asigna como ID el siguiente número de la secuencia TIPOAMO_SEQ.

3.3.11 Asignar ID a tipo de sanción

3.3.11.1 Funcionalidad

El trigger NUEVATIPOSAN permite asignar un nuevo ID cuando se inserta un nuevo tipo de sanción.

3.3.11.2 Ejecución

El trigger se lanza cuando se inserta una nueva fila en la tabla TIPOSANCIONES justo antes de confirmar su inserción.

3.3.11.3 Descripción

1. Si no se proporciona el ID del tipo de sanción en el INSERT se le asigna como ID el siguiente número de la secuencia TIPOSAN_SEQ.

3.3.12 Asignar ID a una fecha del curso

3.3.12.1 Funcionalidad

El trigger NUEVAFECHACURSO permite asignar un nuevo ID cuando se inserta un festivo.

3.3.12.2 Ejecución

El trigger se lanza cuando se inserta una nueva fila en la tabla FECHASCURSO justo antes de confirmar su inserción.

3.3.12.3 Descripción

1. Si no se proporciona el ID de la fecha del curso en el INSERT se le asigna como ID el siguiente número de la secuencia FECHASCURSO_SEQ.

3.3.13 Asignar ID a horario

3.3.13.1 Funcionalidad

El trigger NUEVOHORARIO permite asignar un nuevo ID cuando se inserta un festivo.

3.3.13.2 Ejecución

El trigger se lanza cuando se inserta una nueva fila en la tabla HORARIOS justo antes de confirmar su inserción.

3.3.13.3 Descripción

1. Si no se proporciona el ID del horario en el INSERT se le asigna como ID el siguiente número de la secuencia HORARIOS_SEQ.

3.3.14 Rellenar las fechas del curso

3.3.14.1 Funcionalidad

El trigger FECHASCURSO rellena la tabla del mismo nombre cuando se inserta un nuevo curso.

3.3.14.2 Ejecución

El trigger se lanza cuando se inserta una nueva fila en la tabla CURSOS justo después de confirmar su inserción.

3.3.14.3 Descripción

1. Guardamos la fecha de inicio en una variable auxiliar.
2. Insertamos una fila en FECHASCURSO con el ID del curso y la fecha de la variable auxiliar.
3. Añadimos un día a la fecha de la variable auxiliar.
4. Comprobamos si la nueva fecha obtenida es mayor que la fecha final del curso.
5. Si se cumple la condición terminamos la ejecución, en caso contrario volvemos al paso 2.

3.3.15 Modificar las fechas del curso al cambiar la fecha de inicio.

3.3.15.1 Funcionalidad

El trigger MODFECHASCURSOINICIO ajusta las fechas del curso en la tabla FECHASCURSO cuando se modifica la fecha de inicio de un curso.

3.3.15.2 Ejecución

El trigger se lanza cuando se modifica la columna FECHAINICIO de una fila existente en la tabla CURSOS justo después de confirmar la modificación.

3.3.15.3 Descripción

1. Si la nueva fecha de inicio es mayor que la anterior, borramos todas las filas de la tabla FECHASCURSO entre la antigua fecha de inicio y la nueva fecha de inicio.
2. Si, por el contrario, la nueva fecha de inicio es menor que la anterior asignamos a una variable auxiliar la nueva fecha de inicio.
3. Insertamos una fila en FECHASCURSO con el ID del curso y la fecha de la variable auxiliar.
4. Añadimos un día a la fecha de la variable auxiliar.
5. Comprobamos si la nueva fecha obtenida es igual que la antigua fecha inicial del curso.
6. Si se cumple la condición terminamos la ejecución, en caso contrario volvemos al paso 3.

3.3.16 Modificar las fechas del curso al cambiar la fecha de fin

3.3.16.1 Funcionalidad

El trigger MODFECHASCURSOFIN ajusta las fechas del curso en la tabla FECHASCURSO cuando se modifica la fecha de fin de un curso.

3.3.16.2 Ejecución

El trigger se lanza cuando se modifica la columna FECHAFIN de una fila existente en la tabla CURSOS justo después de confirmar la modificación.

3.3.16.3 Descripción

1. Si la nueva fecha de fin es menor que la anterior, borramos todas las filas de la tabla FECHASCURSO entre la nueva fecha de fin y la antigua fecha de fin.
2. Si, por el contrario, la nueva fecha de fin es mayor que la anterior asignamos a una variable auxiliar la nueva fecha de fin.
3. Insertamos una fila en FECHASCURSO con el ID del curso y la fecha de la variable auxiliar.
4. Añadimos un día a la fecha de la variable auxiliar.
5. Comprobamos si la nueva fecha obtenida es mayor que la nueva fecha final del curso.
6. Si se cumple la condición terminamos la ejecución, en caso contrario volvemos al paso 3.

3.3.17 Actualizar el calendario escolar

3.3.17.1 Funcionalidad

El trigger ACTUALIZACALENDARIO rellena el calendario escolar en la tabla CALENDARIOESCOLAR cuando se insertan o modifican los horarios de las asignaturas.

3.3.17.2 Ejecución

El trigger se lanza cuando se inserta o modifica una fila en la tabla HORARIOASIGNATURAS justo después de confirmar la modificación.

3.3.17.3 Descripción

1. Seleccionamos el día de la semana, la hora de inicio y la hora de fin de la tabla HORARIOS para el horario correspondiente.
2. Hacemos un INSERT en la tabla CALENDARIOESCOLAR de todas las fechas en las que se va a impartir esa asignatura en ese curso y su hora de inicio y fin en cada día.
3. Si la fecha ya existe capturamos el error de pk duplicada y hacemos un UPDATE con las nuevas horas.

3.3.18 Borrar el calendario escolar

3.3.18.1 Funcionalidad

El trigger BORRACALENDARIO borra las fechas del calendario escolar en la tabla CALENDARIOESCOLAR cuando se eliminan los horarios de las asignaturas.

3.3.18.2 Ejecución

El trigger se lanza cuando se elimina una fila en la tabla HORARIOASIGNATURAS justo después de confirmar la modificación.

3.3.18.3 Descripción

1. Seleccionamos el día de la semana, la hora de inicio y la hora de fin de la tabla HORARIOS para el horario correspondiente.
2. Hacemos un DELETE en la tabla CALENDARIOESCOLAR de todas las fechas en las que se impartía esa asignatura en ese curso.

3.3.19 Comprobar las reglas

3.3.19.1 Funcionalidad

El trigger COMPROBARREGLAS llama al procedimiento almacenado p_comprobarReglas cuando se inserta o se modifica una amonestación. Para más información ver la sección 3.47.

3.3.19.2 Ejecución

El trigger se lanza cuando se inserta o se modifica una fila en la tabla AMONESTACIONES justo después de confirmar la modificación.

3.3.19.3 Descripción

1. Llamamos al procedimiento almacenado p_comprobarReglas

3.4 Procedimientos almacenados

Para explotar la base de datos utilizaremos llamadas a procedimientos almacenados en la misma que mediante unos parámetros de entrada previamente definidos se encargan de colocar la información en cada una de las tablas. Estos procedimientos almacenados pueden ser llamados también desde lenguajes de alto nivel para modificar o leer los datos.

Veamos ahora en profundidad cada uno de ellos. Primero explicaremos la funcionalidad que nos permite realizar, después los parámetros que admite y que devuelve y finalmente una breve descripción de su funcionamiento. Para mostrar los parámetros utilizaremos una tabla como esta:

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
V_NOM	VARCHAR2	E	N	Nombre del centro
RSP	VARCHAR2	S	S	OK/ERROR

Dónde utilizamos la siguiente sintaxis:

- Nombre: Indica el nombre del parámetro.

- Tipo: Indica el tipo de dato del parámetro.

- Dir: E si el parámetro es de entrada, S si es de salida. ES si es de Entrada/Salida.

- Obli: S si el parámetro es obligatorio. N si es opcional.

- Descripción: Nos indica la información que debemos pasar en el parámetro.

Se puede consultar el código fuente de todos los procedimientos en el fichero procedimientos.sql del apartado 2.5.

3.4.1 Insertar Log

3.4.1.1 Funcionalidad

El procedimiento p_insertaLog permite insertar un nuevo registro en la tabla de log con información sobre el procedimiento que se está ejecutando.

3.4.1.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
NOMPROC	VARCHAR2	E	S	Nombre del procedimiento que se ejecuta
F_INI	DATE	E	S	Fecha de inicio del procedimiento
F_FIN	DATE	E	S	Fecha de finalización del procedimiento
PARAMSENT	VARCHAR2	E	S	Parámetros de entrada del procedimiento
PARAMSSAL	VARCHAR2	E	S	Parámetros de salida del procedimiento

3.4.1.3 Descripción

1. Se realiza el INSERT en la tabla LOGS con los datos que recibimos.

2. Se confirman los cambios con COMMIT.

En caso de error en cualquier paso se muestra por pantalla un mensaje con la descripción del error.

3.4.2 Alta de centro

3.4.2.1 Funcionalidad

El procedimiento p_altaCentro permite dar de alta un nuevo centro.

Un trigger se encarga de asignar un id al centro automáticamente.

3.4.2.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
V_NOM	VARCHAR2	E	S	Nombre del centro
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.2.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

    - V_Nom tiene que estar relleno.

3. Se realiza el INSERT en la tabla CENTROS

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.3 Alta de curso

3.4.3.1 Funcionalidad

El procedimiento p_altaCurso permite dar de alta un nuevo curso.

Un trigger se encarga de asignar un id al curso automáticamente.

Cuando se da de alta un curso, el trigger FECHASCURSO inserta automáticamente en la tabla FECHASCURSO todas las fechas desde la fecha de inicio hasta la fecha

de fin. Para más información consultar la documentación del trigger en la sección 3.3.17.

3.4.3.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
V_NOM	VARCHAR2	E	S	Nombre del curso
F_INICIO	DATE	E	S	Fecha de inicio del curso
F_FIN	DATE	E	S	Fecha de fin del curso
ID_CENTRO	NUMBER	E	S	ID del centro al que pertenece el curso
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.3.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- V_Nom tiene que estar relleno.

- F_inicio no puede ser mayor que f_fin. Esto está controlado con una restricción en los campos de la tabla, por lo que el procedimiento devolvería un error si no se cumple.

3. Se realiza el INSERT en la tabla CURSOS

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.4 Baja de curso

3.4.4.1 Funcionalidad

El procedimiento p_bajaCurso permite dar de baja un curso existente.

Al dar de baja un curso se eliminarán sus registros relacionados en las tablas FECHASCURSO, ASIGNATURASCURSO, HORARIOASIGNATURAS, PROFESORESCURSO y ALUMNOSCURSO.

3.4.4.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID	NUMBER	E	S	ID del curso que se quiere eliminar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.4.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id tiene que estar relleno.

3. Se realiza el DELETE en la tabla CURSOS

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.5 Modificación de curso

3.4.5.1 Funcionalidad

El procedimiento p_modCurso permite modificar un curso existente. Para ello hay que enviar la ID del curso y cualquiera de los cuatro campos que se permiten modificar: Nombre, fecha de inicio, fecha de fin o id_centro. No es necesario que todos los campos opcionales estén rellenos pero sí deben ir en el orden preestablecido. Si no se quiere modificar un campo se puede enviar vacío si es una cadena o con un cero si es un número.

Si se modifica la fecha de inicio del curso, el trigger MODFECHASCURSOINICIO se encarga de corregir automáticamente en la tabla FECHASCURSO todas las fechas desde la nueva fecha de inicio hasta la fecha de fin. Así mismo el trigger MODFECHASCURSOFIN se encarga del mismo trabajo cuando se modifica la fecha de fin. Para más información consultar la documentación de estos triggers en las secciones 3.3.14 y 3.3.15.

3.4.5.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID	NUMBER	E	S	ID del curso que se quiere modificar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error
V_NOM	VARCHAR2	E	N	Nombre del curso
F_INICIO	DATE	E	N	Fecha de inicio del curso
F_FIN	DATE	E	N	Fecha de fin del curso
ID_CENTRO	NUMBER	E	N	ID del centro al que pertenece el curso

3.4.5.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id tiene que estar relleno.

- F_inicio no puede ser mayor que f_fin. Esto está controlado con una restricción en los campos de la tabla, por lo que el procedimiento devolvería un error si no se cumple.

3. Se realiza el UPDATE en la tabla CURSOS. Solo se actualizan los campos que se han pasado como parámetro.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.6 Alta de alumno

3.4.6.1 Funcionalidad

El procedimiento p_altaAlumno permite dar de alta un nuevo alumno.

Un trigger se encarga de asignar un id al alumno automáticamente.

3.4.6.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
V_NOM	VARCHAR2	E	S	Nombre del alumno
V_APE	VARCHAR2	E	S	Apellidos del alumno
V_DNI	VARCHAR2	E	S	Documento de identidad del alumno
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.6.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- V_Nom tiene que estar relleno.

- V_Ape tiene que estar relleno.

- V_Dni tiene que estar relleno.

3. Se realiza el INSERT en la tabla ALUMNOS

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.7 Baja de alumno

3.4.7.1 Funcionalidad

El procedimiento p_bajaAlumno permite dar de baja un curso existente.

Al dar de baja un alumno se eliminarán sus registros relacionados en la tabla ALUMNOSCURSO.

3.4.7.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID	NUMBER	E	S	ID del alumno que se quiere eliminar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.4.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id tiene que estar relleno.

3. Se realiza el DELETE en la tabla CURSOS

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.8 Modificación de alumno

3.4.8.1 Funcionalidad

El procedimiento p_modAlumno permite modificar un alumno existente. Para ello hay que enviar la ID del alumno y cualquiera de los tres campos que se permiten modificar: Nombre, apellidos o dni. No es necesario que todos los campos opcionales estén rellenos pero sí deben ir en el orden preestablecido. Si no se quiere modificar un campo se puede enviar vacío si es una cadena o con un cero si es un número.

3.4.8.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID	NUMBER	E	S	ID del alumno que se quiere modificar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error
V_NOM	VARCHAR2	E	N	Nombre del alumno
V_APE	VARCHAR2	E	N	Apellidos del alumno
V_DNI	VARCHAR2	E	N	DNI del alumno

3.4.5.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id tiene que estar relleno.

3. Se realiza el UPDATE en la tabla ALUMNOS. Solo se actualizan los campos que se han pasado como parámetro.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.9 Alta de profesor

3.4.9.1 Funcionalidad

El procedimiento p_altaProfesor permite dar de alta un nuevo profesor.

Un trigger se encarga de asignar un id al profesor automáticamente.

3.4.9.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
V_NOM	VARCHAR2	E	S	Nombre del profesor
V_APE	VARCHAR2	E	S	Apellidos del profesor
V_DNI	VARCHAR2	E	S	Documento de identidad del profesor

RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.9.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- V_Nom tiene que estar relleno.

- V_Ape tiene que estar relleno.

- V_Dni tiene que estar relleno.

3. Se realiza el INSERT en la tabla PROFESORES

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.10 Baja de profesor

3.4.10.1 Funcionalidad

El procedimiento p_bajaProfesor permite dar de baja un profesor existente.

Al dar de baja un profesor se eliminarán sus registros relacionados en las tablas PROFESORESASIGNATURA y PROFESORESCURSO.

3.4.10.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID	NUMBER	E	S	ID del profesor que se quiere eliminar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.10.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id tiene que estar relleno.

3. Se realiza el DELETE en la tabla PROFESORES

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.11 Modificación de profesor

3.4.11.1 Funcionalidad

El procedimiento p_modProfesor permite modificar un profesor existente. Para ello hay que enviar la ID del mismo y cualquiera de los tres campos que se permiten modificar: Nombre, apellidos o dni. No es necesario que todos los campos opcionales estén rellenos pero sí deben ir en el orden preestablecido. Si no se quiere modificar un campo se puede enviar vacío si es una cadena o con un cero si es un número.

3.4.11.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID	NUMBER	E	S	ID del profesor que se quiere modificar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error
V_NOM	VARCHAR2	E	N	Nombre del profesor
V_APE	VARCHAR2	E	N	Apellidos del profesor
V_DNI	VARCHAR2	E	N	DNI del profesor

3.4.11.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id tiene que estar relleno.

3. Se realiza el UPDATE en la tabla PROFESORES. Solo se actualizan los campos que se han pasado como parámetro.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.12 Alta de asignatura

3.4.12.1 Funcionalidad

El procedimiento p_altaAsignatura permite dar de alta una nueva asignatura.

Un trigger se encarga de asignar un id a la asignatura automáticamente.

3.4.12.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
V_NOM	VARCHAR2	E	S	Nombre de la asignatura
CRED	NUMBER	E	S	Número de créditos de la asignatura
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.12.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

    - V_Nom tiene que estar relleno.

    - Cred tiene que estar relleno.

3. Se realiza el INSERT en la tabla ASIGNATURAS

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.13 Baja de asignatura

3.4.13.1 Funcionalidad

El procedimiento p_bajaAsignatura permite dar de baja una asignatura existente.

Al dar de baja una asignatura se eliminarán sus registros relacionados en las tablas ASIGNATURASCURSO, HORARIOASIGNATURAS, CALENDARIOESCOLAR y PROFESORESASIGNATURA.

3.4.13.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID	NUMBER	E	S	ID de la asignatura que se quiere eliminar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.13.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id tiene que estar relleno.

3. Se realiza el DELETE en la tabla ASIGNATURAS

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.14 Modificación de asignatura

3.4.14.1 Funcionalidad

El procedimiento p_modAsignatura permite modificar una asignatura existente. Para ello hay que enviar la ID de la misma y cualquiera de los dos campos que se permiten modificar: Nombre y número de créditos. No es necesario que todos los campos opcionales estén rellenos pero sí deben ir en el orden preestablecido. Si no se quiere modificar un campo se puede enviar vacío si es una cadena o con un cero si es un número.

3.4.14.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID	NUMBER	E	S	ID de la asignatura que se quiere modificar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error
V_NOM	VARCHAR2	E	N	Nombre de la asignatura
CRED	NUMBER	E	N	Número de créditos de la asignatura

3.4.14.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id tiene que estar relleno.

3. Se realiza el UPDATE en la tabla ASIGNATURAS. Solo se actualizan los campos que se han pasado como parámetro.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.15 Alta de profesor responsable en un curso

3.4.15.1 Funcionalidad

El procedimiento p_altaProfesorCurso permite dar de alta un profesor como responsable de un curso.

3.4.15.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_PROFESOR	NUMBER	E	S	ID del profesor
ID_CURSO	NUMBER	E	S	ID del curso
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.15.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_profesor tiene que estar relleno.

- Id_curso tiene que estar relleno.

3. Se realiza el INSERT en la tabla PROFESORESCURSO

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.16 Baja de profesor responsable en un curso

3.4.16.1 Funcionalidad

El procedimiento p_bajaProfesorCurso permite dar de baja un profesor como responsable de un curso.

3.4.16.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_CURSO	NUMBER	E	S	ID del curso cuyo profesor responsable se quiere eliminar
ID_PROFESOR	NUMBER	E	S	ID del profesor responsable que se quiere eliminar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.16.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_curso tiene que estar relleno.

- Id_profesor tiene que estar relleno.

3. Se realiza el DELETE en la tabla PROFESORESCURSO

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.17 Alta de horario de atención a alumnos de profesor responsable

3.4.17.1 Funcionalidad

El procedimiento p_altaHorarioAlumnoProf permite dar de alta un nuevo horario de atención del profesor responsable del curso a los alumnos de éste.

3.4.17.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_CURSO	NUMBER	E	S	ID del curso del que es responsable el profesor
ID_PROFESOR	NUMBER	E	S	ID del profesor responsable
DIA_SEMANA	NUMBER	E	S	Número correspondiente al día de la semana (1-7)
HORA_INICIO	VARCHAR2	E	S	Hora de inicio del horario de atención
HORA_FIN	CARCHAR2	E	S	Hora de fin del horario de atención
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.17.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_curso tiene que estar relleno.

- Id_profesor tiene que estar relleno.

- Dia_semana tiene que estar relleno.

- Dia_semana tiene que ser un número del 1 al 7 (correspondiente a los siete días de la semana: 1=Lunes, 2=Martes, etc.). Esto está controlado con una restricción en el campo de la tabla, por lo que el procedimiento devolvería un error si no se cumple.

- Hora_inicio tiene que estar relleno.

- Hora_fin tiene que estar relleno.

- Hora_inicio y hora_fin deben ser una hora con el formato 'HH24:MI'. Si no, se devolverá un error de formato de hora incorrecto.

- Hora_inicio tiene que ser menor que hora_fin. Esto está controlado con una restricción en los campos de la tabla, por lo que el procedimiento devolvería un error si no se cumple.

3. Se comprueba que el profesor es responsable de ese curso. En caso de que no lo sea se devuelve un error.

4. Se busca el horario en la tabla HORARIOS. Si no se encuentra se inserta. Se guarda su ID para el posterior INSERT.

5. Se realiza el INSERT en la tabla PROFESORESCURSO. Se especifica 'A' en el campo TIPOHORARIO.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.18 Baja de horario de atención a alumnos de profesor responsable

3.4.18.1 Funcionalidad

El procedimiento p_bajaHorarioAlumnosProf permite dar de baja un horario de atención del profesor responsable del curso a los alumnos de éste.

3.4.18.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN

ID_CURSO	NUMBER	E	S	ID del curso del que es responsable el profesor
ID_PROFESOR	NUMBER	E	S	ID del profesor responsable del curso
ID_HORARIO	NUMBER	E	S	ID del horario que se quiere eliminar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.18.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_curso tiene que estar relleno.

- Id_profesor tiene que estar relleno.

- Id_horario tiene que estar relleno.

3. Se comprueba que el profesor es responsable de ese curso. En caso de que no lo sea se devuelve un error.

4. Se realiza el DELETE en la tabla PROFESORESCURSO

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.19 Alta de horario de atención a padres de profesor responsable

3.4.19.1 Funcionalidad

El procedimiento p_altaHorarioPadresProf permite dar de alta un nuevo horario de atención del profesor responsable del curso a los padres de los alumnos de éste.

3.4.19.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_CURSO	NUMBER	E	S	ID del curso del que es responsable el profesor
ID_PROFESOR	NUMBER	E	S	ID del profesor responsable
DIA_SEMANA	NUMBER	E	S	Número correspondiente al día de la semana (1-7)
HORA_INICIO	VARCHAR2	E	S	Hora de inicio del horario de atención
HORA_FIN	CARCHAR2	E	S	Hora de fin del horario de atención
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.19.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_curso tiene que estar relleno.

- Id_profesor tiene que estar relleno.

- Dia_semana tiene que estar relleno.

- Dia_semana tiene que ser un número del 1 al 7 (correspondiente a los siete días de la semana: 1=Lunes, 2=Martes, etc.). Esto está controlado con una restricción en el campo de la tabla, por lo que el procedimiento devolvería un error si no se cumple.

- Hora_inicio tiene que estar relleno.

- Hora_fin tiene que estar relleno.

- Hora_inicio y hora_fin deben ser una hora con el formato 'HH24:MI'. Si no, se devolverá un error de formato de hora incorrecto.

- Hora_inicio tiene que ser menor que hora_fin. Esto está controlado con una restricción en los campos de la tabla, por lo que el procedimiento devolvería un error si no se cumple.

3. Se comprueba que el profesor es responsable de ese curso. En caso de que no lo sea se devuelve un error.

4. Se busca el horario en la tabla HORARIOS. Si no se encuentra se inserta. Se guarda su ID para el posterior INSERT.

5. Se realiza el INSERT en la tabla PROFESORESCURSO. Se especifica 'P' en el campo TIPOHORARIO.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.20 Baja de horario de atención a padres de profesor responsable

3.4.20.1 Funcionalidad

El procedimiento p_bajaHorarioPadresProf permite dar de baja un horario de atención del profesor responsable del curso a los padres de los alumnos de éste.

3.4.20.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_CURSO	NUMBER	E	S	ID del curso del que es responsable el profesor
ID_PROFESOR	NUMBER	E	S	ID del profesor responsable del curso
ID_HORARIO	NUMBER	E	S	ID del horario que se quiere eliminar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.20.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_curso tiene que estar relleno.

- Id_profesor tiene que estar relleno.

- Id_horario tiene que estar relleno.

3. Se comprueba que el profesor es responsable de ese curso. En caso de que no lo sea se devuelve un error.

4. Se realiza el DELETE en la tabla PROFESORESCURSO

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.21 Alta de profesor en una asignatura

3.4.21.1 Funcionalidad

El procedimiento p_altaProfesorAsignatura permite dar de alta un profesor como docente en una asignatura.

3.4.21.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_PROFESOR	NUMBER	E	S	ID del profesor
ID_ASIGNATURA	NUMBER	E	S	ID de la asignatura
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.21.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_profesor tiene que estar relleno.

- Id_asignatura tiene que estar relleno.

3. Se realiza el INSERT en la tabla PROFESORESASIGNATURA

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.22 Baja de profesor en una asignatura

3.4.22.1 Funcionalidad

El procedimiento p_bajaProfesorAsignatura permite dar de baja un profesor como docente de una asignatura.

3.4.22.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_ASIGNATURA	NUMBER	E	S	ID de la asignatura cuyo profesor docente se quiere eliminar
ID_PROFESOR	NUMBER	E	S	ID del profesor que se quiere eliminar de la docencia de la asignatura
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.22.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_asignatura tiene que estar relleno.

- Id_profesor tiene que estar relleno.

3. Se realiza el DELETE en la tabla PROFESORESASIGNATURA

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.23 Alta de alumno en un curso

3.4.23.1 Funcionalidad

El procedimiento p_altaAlumnoCurso permite dar de alta un alumno en un curso.

3.4.23.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_CURSO	NUMBER	E	S	ID del curso
ID_ALUMNO	NUMBER	E	S	ID de la alumno
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.23.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_curso tiene que estar relleno.

- Id_alumno tiene que estar relleno.

3. Se realiza el INSERT en la tabla ALUMNOSCURSO

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.24 Baja de alumno en un curso

3.4.24.1 Funcionalidad

El procedimiento p_bajaAlumnoCurso permite dar de baja un alumno inscrito en un curso.

3.4.24.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_CURSO	NUMBER	E	S	ID del curso cuyo alumno se quiere

				eliminar
ID_ALUMNO	NUMBER	E	S	ID del alumno que se quiere eliminar del curso.
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.24.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_curso tiene que estar relleno.

- Id_alumno tiene que estar relleno.

3. Se realiza el DELETE en la tabla ALUMNOSCURSO

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.25 Alta de asignatura en un curso

3.4.25.1 Funcionalidad

El procedimiento p_altaAsignaturaCurso permite dar de alta una asignatura en un curso.

3.4.25.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_CURSO	NUMBER	E	S	ID del curso
ID_ASIGNATURA	NUMBER	E	S	ID de la asignatura
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.25.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_curso tiene que estar relleno.

- Id_asignatura tiene que estar relleno.

3. Se realiza el INSERT en la tabla ASIGNATURASCURSO

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.26 Baja de asignatura en un curso

3.4.26.1 Funcionalidad

El procedimiento p_bajaAsignaturaCurso permite dar de baja una asignatura perteneciente a un curso.

3.4.26.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_CURSO	NUMBER	E	S	ID del curso cuyo alumno se quiere eliminar
ID_ASIGNATURA	NUMBER	E	S	ID de la asignatura que se quiere eliminar del curso.
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.26.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_curso tiene que estar relleno.

- Id_asignatura tiene que estar relleno.

3. Se realiza el DELETE en la tabla ASIGNATURASCURSO

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.27 Alta de horario de asignatura

3.4.27.1 Funcionalidad

El procedimiento p_altaHorarioAsignatura permite dar de alta un nuevo horario para una asignatura en un curso concreto.

Cuando se da de alta un horario para una asignatura, el trigger ACTUALIZACALENDARIO inserta automáticamente en la tabla CALENDARIOESCOLAR el horario para todos los días del curso en los que se va a impartir esta asignatura. Para más información consultar la documentación del trigger en la sección 3.3.16.

3.4.27.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_CURSO	NUMBER	E	S	ID del curso
ID_ASIGNATURA	NUMBER	E	S	ID de la asignatura
DIA_SEMANA	NUMBER	E	S	Número correspondiente al día de la semana (1-7)
HORA_INICIO	VARCHAR2	E	S	Hora de inicio del horario de atención
HORA_FIN	CARCHAR2	E	S	Hora de fin del horario de atención
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.27.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_curso tiene que estar relleno.

- Id_asignatura tiene que estar relleno.

- Dia_semana tiene que estar relleno.

- Dia_semana tiene que ser un número del 1 al 7 (correspondiente a los siete días de la semana: 1=Lunes, 2=Martes, etc.). Esto está controlado con una restricción en el campo de la tabla, por lo que el procedimiento devolvería un error si no se cumple.

- Hora_inicio tiene que estar relleno.

- Hora_fin tiene que estar relleno.

- Hora_inicio y hora_fin deben ser una hora con el formato 'HH24:MI'. Si no, se devolverá un error de formato de hora incorrecto.

- Hora_inicio tiene que ser menor que hora_fin. Esto está controlado con una restricción en los campos de la tabla, por lo que el procedimiento devolvería un error si no se cumple.

3. Se comprueba que la asignatura se imparte en ese curso. En caso de que no sea así se devuelve un error.

4. Se busca el horario en la tabla HORARIOS. Si no se encuentra se inserta. Se guarda su ID para el posterior INSERT.

5. Se realiza el INSERT en la tabla HORARIOASIGNATURAS.

6. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.28 Baja de horario de asignatura

3.4.28.1 Funcionalidad

El procedimiento p_bajaHorarioAsignatura permite dar de baja un horario para la asignatura en el curso.

Cuando se da de baja un horario para una asignatura, el trigger BORRACALENDARIO elimina automáticamente en la tabla CALENDARIOESCOLAR el horario para todos los días del curso en los que ya no se va a impartir esta asignatura. Para más información consultar la documentación del trigger en la sección 3.3.17.

3.4.28.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_CURSO	NUMBER	E	S	ID del curso
ID_ASIGNATURA	NUMBER	E	S	ID de la asignatura
ID_HORARIO	NUMBER	E	S	ID del horario que se quiere eliminar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.28.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_curso tiene que estar relleno.

- Id_asignatura tiene que estar relleno.

- Id_horario tiene que estar relleno.

3. Se comprueba que la asignatura se imparte en ese curso. En caso de que no lo sea se devuelve un error.

4. Se realiza el DELETE en la tabla HORARIOASIGNATURAS

5. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.29 Alta de festivo

3.4.29.1 Funcionalidad

El procedimiento p_altaFestivo permite dar de alta un festivo en el calendario escolar.

3.4.29.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
F_FESTIVO	DATE	E	S	Fecha del festivo
DESCRIPCION	VARCHAR2	E	S	Descripción del festivo
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.29.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- F_Festivo tiene que estar relleno.

- Descripcion tiene que estar relleno.

3. Se realiza un UPDATE en la tabla FECHASCURSO que actualiza el campo FESTIVO a 'S' y la descripción para la fecha dada.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.30 Baja de festivo

3.4.30.1 Funcionalidad

El procedimiento p_bajaFestivo permite dar de baja un festivo en el calendario escolar.

3.4.30.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
F_FESTIVO	DATE	E	S	Fecha del festivo
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.30.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- F_Festivo tiene que estar relleno.

3. Se realiza un UPDATE en la tabla FECHASCURSO que actualiza el campo FESTIVO a 'N' para la fecha dada.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.31 Modificación de festivo

3.4.31.1 Funcionalidad

El procedimiento p_modFestivo permite modificar un festivo en el calendario escolar.

3.4.31.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
F_FESTIVO	DATE	E	S	Fecha del festivo
DESCRIPCION	VARCHAR2	E	S	Descripción del festivo
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.31.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- F_Festivo tiene que estar relleno.

- Descripcion tiene que estar relleno.

3. Se realiza un UPDATE en la tabla FECHASCURSO que actualiza el campo FESTIVO a 'S' y la descripción para la fecha dada.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.32 Alta de tipo de amonestación

3.4.32.1 Funcionalidad

El procedimiento p_altaTipoAmonestacion permite dar de alta un nuevo tipo de amonestación.

Un trigger se encarga de asignar un id al tipo de amonestación automáticamente.

3.4.32.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_PROFESOR	NUMBER	E	S	ID del profesor que da de alta la amonestación
DESCRIPCION	NUMBER	E	S	Descripción del tipo de amonestación
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.32.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_profesor tiene que estar relleno.

- Descripcion tiene que estar relleno.

3. Se realiza el INSERT en la tabla TIPOAMONESTACIONES.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.33 Baja de tipo de amonestación

3.4.33.1 Funcionalidad

El procedimiento p_bajaTipoAmonestación permite dar de baja un tipo de amonestación existente.

3.4.33.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_TIPO	NUMBER	E	S	ID del tipo de amonestación que se quiere eliminar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.33.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_tipo tiene que estar relleno.

3. Se realiza el DELETE en la tabla TIPOAMONESTACIONES.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.34 Modificación de tipo de amonestación

3.4.34.1 Funcionalidad

El procedimiento p_modTipoAmonestación permite modificar un tipo de amonestación existente. Para ello hay que enviar la ID de la amonestación, el ID del profesor que la modifica y la descripción.

3.4.34.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_TIPO	NUMBER	E	S	ID del tipo de amonestación que se quiere modificar
ID_PROFESOR	NUMBER	E	S	ID del profesor que modifica el tipo
DESCRIPCION	VARCHAR2	E	S	Descripción del tipo de amonestación
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.34.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_tipo tiene que estar relleno.

- Id_profesor tiene que estar relleno.

- Descripción tiene que estar relleno.

3. Se realiza el UPDATE en la tabla TIPOAMONESTACIONES. Solo se actualizan los campos que se han pasado como parámetro.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.35 Alta de tipo sanción

3.4.35.1 Funcionalidad

El procedimiento p_altaTipoSancion permite dar de alta un nuevo tipo de sanción.

Un trigger se encarga de asignar un id al tipo de sanción automáticamente.

3.4.35.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_PROFESOR	NUMBER	E	S	ID del profesor que da de alta la sanción
DESCRIPCION	NUMBER	E	S	Descripción del tipo de sanción
ID_TIPOAMO	NUMBER	E	S	ID del tipo de amonestación relacionada con esta sanción
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.35.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_profesor tiene que estar relleno.

- Descripcion tiene que estar relleno.

- Id_tipoamo tiene que estar relleno.

3. Se realiza el INSERT en la tabla TIPOSANCIONES.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.36 Baja de tipo de sanción

3.4.36.1 Funcionalidad

El procedimiento p_bajaTipoSanción permite dar de baja un tipo de sanción existente.

3.4.36.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_TIPO	NUMBER	E	S	ID del tipo de sanción que se quiere eliminar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.36.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_tipo tiene que estar relleno.

3. Se realiza el DELETE en la tabla TIPOSANCION.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.37 Modificación de tipo de sanción

3.4.37.1 Funcionalidad

El procedimiento p_modTipoSanción permite modificar un tipo de sanción existente. Para ello hay que enviar la ID de la sanción, el ID del profesor que la modifica, la descripción y el tipo de amonestación relacionada.

3.4.37.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_TIPO	NUMBER	E	S	ID del tipo de sanción que se quiere modificar
ID_PROFESOR	NUMBER	E	S	ID del profesor que modifica el tipo
DESCRIPCION	VARCHAR2	E	S	Descripción del tipo de sanción
ID_TIPOAMO	NUMBER	E	S	ID del tipo de amonestación relacionada con la sanción
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.37.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_tipo tiene que estar relleno.

- Id_profesor tiene que estar relleno.

- Descripción tiene que estar relleno.

- Id_tipoamo tiene que estar relleno.

3. Se realiza el UPDATE en la tabla TIPOSANCIONES. Solo se actualizan los campos que se han pasado como parámetro.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.38 Alta de amonestación

3.4.38.1 Funcionalidad

El procedimiento p_altaAmonestacion permite dar de alta una nueva amonestación.

Un trigger se encarga de asignar un id a la amonestación automáticamente.

Cuando se da de alta una amonestación, el trigger COMPROBARREGLAS se encarga de comprobar si se ha cumplido alguna de las reglas correspondientes a este tipo de amonestación que se está dando de alta y por lo tanto, insertar la sanción correspondiente automáticamente, si procede. Para más información consultar la documentación del trigger en la sección.

Cuando se da de alta una amonestación se actualizan las tablas de estadísticas correspondientes a amonestaciones llamando al procedimiento p_rellenaEstadisticasAmo.

3.4.38.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_ALUMNO	NUMBER	E	S	ID del alumno amonestado
ID_CURSO	NUMBER	E	S	ID del curso
ID_ASIGNATURA	NUMBER	E	S	ID de la asignatura
ID_PROFESOR	NUMBER	E	S	ID del profesor
ID_TIPO	NUMBER	E	S	ID del tipo de amonestación
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error
FECHA_AMO	DATE	E	N	Fecha de la amonestación
GRAVEDAD_AMO	VARCHAR2	E	N	Gravedad de la amonestación
COMUNICADO_AMO	VARCHAR2	E	N	Indica si se ha comunicado o no

3.4.38.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_alumno tiene que estar relleno.

- Id_curso tiene que estar relleno.

- Id_asignatura tiene que estar relleno.

- Id_profesor tiene que estar relleno.

- Id_tipo tiene que estar relleno.

- Gravedad_amo es opcional pero si se envía tiene que ser 'A', 'B' o 'M' que corresponde a Alta, Media y Baja.

- Comunicado_amo es opcional pero si se envía tiene que ser 'S' o 'N' que corresponde a Sí o No. El valor por defecto es 'N'.

3. Se realiza el INSERT en la tabla AMONESTACIONES.

4. Se llama al procedimiento p_rellenaEstadisticasAmo.

5. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.39 Baja de amonestación

3.4.39.1 Funcionalidad

El procedimiento p_bajaAmonestación permite dar de baja una amonestación existente.

Cuando se da de baja una amonestación se actualizan las tablas de estadísticas correspondientes a amonestaciones llamando al procedimiento p_rellenaEstadisticasAmo.

3.4.39.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID	NUMBER	E	S	ID de la amonestación que se quiere eliminar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.39.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

    - Id tiene que estar relleno.

3. Se realiza el DELETE en la tabla AMONESTACIONES.

4. Se llama al procedimiento p_rellenaEstadisticasAmo.

5. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.40 Modificación de amonestación

3.4.40.1 Funcionalidad

El procedimiento p_modAmonestación permite modificar una amonestación existente. Para ello hay que enviar la ID de la amonestación y cualquiera de los ocho campos que se permiten modificar: Id de alumno, id de curso, id de asignatura, id de profesor, id de tipo de amonestación, fecha de la amonestación, gravedad de la amonestación o comunicado de la amonestación. No es necesario que todos los campos opcionales estén rellenos pero sí deben ir en el orden preestablecido. Si no se quiere modificar un campo se puede enviar vacío si es una cadena o con un cero si es un número.

Cuando se modifica una amonestación, el trigger COMPROBARREGLAS se encarga de comprobar si se ha cumplido alguna de las reglas correspondientes a este tipo de amonestación al modificarse y por lo tanto, insertar la sanción correspondiente automáticamente, si procede.

Cuando modifica una amonestación se actualizan las tablas de estadísticas correspondientes a amonestaciones llamando al procedimiento p_rellenaEstadisticasAmo.

3.4.40.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID	NUMBER	E	S	ID de la amonestación que se quiere modificar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error
ID_ALUMNO	NUMBER	E	N	ID del alumno amonestado
ID_CURSO	NUMBER	E	N	ID del curso
ID_ASIGNATURA	NUMBER	E	N	ID de la asignatura
ID_PROFESOR	NUMBER	E	N	ID del profesor
ID_TIPO	NUMBER	E	N	ID del tipo de amonestación
FECHA_AMO	DATE	E	N	Fecha de la amonestación
GRAVEDAD_AMO	VARCHAR2	E	N	Gravedad de la amonestación
COMUNICADO_AMO	VARCHAR2	E	N	Indica si se ha comunicado o no

3.4.40.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id tiene que estar relleno.

- Gravedad_amo es opcional pero si se envía tiene que ser 'A', 'B' o 'M' que corresponde a Alta, Media y Baja.

- Comunicado_amo es opcional pero si se envía tiene que ser 'S' o 'N' que corresponde a Sí o No. El valor por defecto es 'N'.

3. Se realiza el UPDATE en la tabla AMONESTACIONES. Solo se actualizan los campos que se han pasado como parámetro.

4. Se llama al procedimiento p_rellenaEstadisticasAmo.

5. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.41 Alta de sanción

3.4.41.1 Funcionalidad

El procedimiento p_altaSancion permite dar de alta una nueva sanción. Todos los campos son obligatorios excepto Fecha_san. Si no se envía una fecha de sanción se pone la fecha actual del sistema por defecto.

Un trigger se encarga de asignar un id a la sanción automáticamente.

Cuando se da de alta una sanción se actualizan las tablas de estadísticas correspondientes a amonestaciones llamando al procedimiento p_rellenaEstadisticasSan.

3.4.41.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_ALUMNO	NUMBER	E	S	ID del alumno sancionado
ID_CURSO	NUMBER	E	S	ID del curso
ID_PROFESOR	NUMBER	E	S	ID del profesor
ID_TIPO	NUMBER	E	S	ID del tipo de sanción
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error
FECHA_SAN	DATE	E	S	Fecha de la sanción

3.4.41.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

   - Id_alumno tiene que estar relleno.

   - Id_curso tiene que estar relleno.

   - Id_asignatura tiene que estar relleno.

   - Id_tipo tiene que estar relleno.

3. Se realiza el INSERT en la tabla SANCIONES.

4. Se llama al procedimiento p_rellenaEstadisticasSan.

5. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.42 Baja de sanción

3.4.42.1 Funcionalidad

El procedimiento p_bajaSanción permite dar de baja una sanción existente.

Cuando se da de baja una sanción se actualizan las tablas de estadísticas correspondientes a amonestaciones llamando al procedimiento p_rellenaEstadisticasSan.

3.4.42.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID	NUMBER	E	S	ID de la sanción que se quiere eliminar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.42.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

    - Id tiene que estar relleno.

3. Se realiza el DELETE en la tabla SANCIONES.

4. Se llama al procedimiento p_rellenaEstadisticasSan.

5. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.43 Modificación de sanción

3.4.43.1 Funcionalidad

El procedimiento p_modSanción permite modificar una sanción existente. Para ello hay que enviar la ID de la sanción y cualquiera de los siete campos que se permiten modificar: Id de alumno, id de curso, id de asignatura, id de profesor, id de tipo de sanción, fecha de la sanción y resolución de la sanción. No es necesario que todos

los campos opcionales estén rellenos pero sí deben ir en el orden preestablecido. Si no se quiere modificar un campo se puede enviar vacío si es una cadena o con un cero si es un número.

Cuando se modifica una sanción se actualizan las tablas de estadísticas correspondientes a amonestaciones llamando al procedimiento p_rellenaEstadisticasSan.

3.4.43.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID	NUMBER	E	S	ID de la sanción que se quiere modificar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error
ID_ALUMNO	NUMBER	E	N	ID del alumno sancionado
ID_CURSO	NUMBER	E	N	ID del curso
ID_PROFESOR	NUMBER	E	N	ID del profesor
ID_TIPO	NUMBER	E	N	ID del tipo de amonestación
ID_REGLA	NUMBER	E	N	ID de la asignatura
FECHA_SAN	DATE	E	N	Fecha de la amonestación
RESOLUCION_SAN	VARCHAR2	E	N	Indica si se ha comunicado o no

3.4.43.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id tiene que estar relleno.

3. Se realiza el UPDATE en la tabla SANCIONES. Solo se actualizan los campos que se han pasado como parámetro.

4. Se llama al procedimiento p_rellenaEstadisticasSan.

5. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.44 Alta de regla

3.4.44.1 Funcionalidad

El procedimiento p_altaRegla permite dar de alta una nueva regla para aplicar una sanción.

3.4.44.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
TIPO_AMO	NUMBER	E	S	Tipo de amonestación
TIPO_SAN	NUMBER	E	S	Tipo de sanción a aplicar para este tipo de amonestación.
OPER	VARCHAR2	E	S	Operador a aplicar en la comparación
VALOR	NUMBER	E	S	Valor con el que se comparará el número de amonestaciones de este tipo.
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.44.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Tipo_amo tiene que estar relleno.

- Tipo_san tiene que estar relleno.

- Oper tiene que estar relleno. El operador tiene que ser uno de los siguientes: >, <, =, >=, <=, =<, =>

- Valor tiene que estar relleno.

3. Se realiza el INSERT en la tabla REGLASSANCIONES.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.45 Baja de regla

3.4.45.1 Funcionalidad

El procedimiento p_bajaRegla permite dar de baja una regla existente.

3.4.45.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID	NUMBER	E	S	ID de la regla que se quiere eliminar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.45.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id tiene que estar relleno.

3. Se realiza el DELETE en la tabla REGLASSANCIONES.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.46 Modificación de regla

3.4.46.1 Funcionalidad

El procedimiento p_modRegla permite modificar una regla existente. Para ello hay que enviar la ID de la regla y cualquiera de los cuatro campos que se permiten modificar: Tipo de amonestación, tipo de sanción, operador y valor. No es necesario que todos los campos opcionales estén rellenos pero sí deben ir en el orden preestablecido. Si no se quiere modificar un campo se puede enviar vacío si es una cadena o con un cero si es un número.

3.4.46.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID	NUMBER	E	S	ID de la sanción que se quiere modificar
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error
TIPO_AMO	NUMBER	E	N	Tipo de amonestación
TIPO_SAN	NUMBER	E	N	Tipo de sanción
OPER	VARCHAR2	E	N	Operador a aplicar en la comparación
VALOR	NUMBER	E	N	Valor con el que se comparará el número de amonestaciones de este tipo.

3.4.46.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id tiene que estar relleno.

3. Se realiza el UPDATE en la tabla REGLASSANCIONES. Solo se actualizan los campos que se han pasado como parámetro.

4. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.47 Comprobar regla

3.4.47.1 Funcionalidad

El procedimiento p_comprobarReglas permite comprobar si se cumple una regla para aplicar una sanción.

3.4.47.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
ID_TIPO	NUMBER	E	S	ID de tipo de amonestación sobre la que se quiere hacer la comprobación
ID_ALUMNO	NUMBER	E	S	ID de alumno
ID_CURSO	NUMBER	E	S	ID de curso
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.47.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Comprobación de parámetros obligatorios:

- Id_tipo tiene que estar relleno.

- Id_alumno tiene que estar relleno.

- Id_curso tiene que estar relleno.

3. Buscamos el número de amonestaciones de este tipo que tiene el alumno desde la fecha en que le fue impuesta la última sanción automática relacionada con este tipo.

4. Recorremos todas las reglas para este tipo de amonestación y comprobamos si se cumple la condición guardada en el operador y el valor.

5. Si la regla se cumple, se realiza el INSERT en la tabla SANCIONES con la fecha actual.

6. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.48 Rellena estadísticas Amonestaciones

3.4.48.1 Funcionalidad

El procedimiento p_rellenaEstadísticasAmo permite rellenar las tablas del módulo de estadísticas relacionadas con amonestaciones con los datos de la base de datos. Cada vez que se inserta una amonestación se ejecuta automáticamente este procedimiento.

3.4.48.2 Parámetros

Ninguno.

3.4.48.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Se borran los datos de las tablas de estadísticas relacionados con amonestaciones.

3. Se realiza el INSERT en la tabla EST_AMOXALUMNO de las amonestaciones por alumno.

4. Se realiza el INSERT en la tabla EST_MEDIAAMOPROFESORESXANYO de las media de amonestaciones por profesor y año.

5. Se realiza el INSERT en la tabla EST_ALUMNOSSINAMO del número de alumnos sin amonestaciones.

6. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.49 Rellena estadísticas Sanciones

3.4.49.1 Funcionalidad

El procedimiento p_rellenaEstadísticasSan permite rellenar las tablas del módulo de estadísticas relacionadas con sanciones con los datos de la base de datos. Cada vez que se inserta una sanción se ejecuta automáticamente este procedimiento.

3.4.49.2 Parámetros

Ninguno.

3.4.49.3 Descripción

1. Se guardan la fecha de inicio y los parámetros de entrada para insertarlos en el log.

2. Se borran los datos de las tablas de estadísticas relacionados con sanciones.

3. Se realiza el INSERT en la tabla EST_SANXALUMNOXANYO de las sanciones por alumno y año.

4. Se realiza el INSERT en la tabla EST_SANXANYOXCURSO de las sanciones por curso y año.

5. Se realiza el INSERT en la tabla EST_ALUMNOMASSAN del alumno más sancionado por año.

6. Se realiza el INSERT en la tabla EST_PROFESORMASAMO del profesor más sancionador por curso.

7. Se realiza el INSERT en la tabla EST_MEDIASANALUMNOSXCURSO de la media de sanciones de alumnos por curso.

8. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.50 Listado de amonestaciones

3.4.50.1 Funcionalidad

El procedimiento p_listadoamonestaciones, incluido en el package LISTADOS, permite consultar todas las amonestaciones impuestas que están almacenadas en el sistema.

3.4.50.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
LIST_AMO	AMO_TABLA	S	S	Listado de todas las amonestaciones impuestas.

RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.50.3 Descripción

1. Se guarda la fecha de inicio para insertarla en el log.

2. Se imprime por pantalla la cabecera del informe.

3. Se realiza la SELECT a la vista LISTADOAMONESTACIONES.

4. Se imprime por pantalla las cabeceras de cada columna de la vista.

5. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.51 Listado de alumnos por curso

3.4.51.1 Funcionalidad

El procedimiento p_listadoalumnosxcurso, incluido en el package LISTADOS, permite consultar todas las amonestaciones impuestas que están almacenadas en el sistema.

3.4.51.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
LIST_ALUMN	ALUMXCUR_TABLA	S	S	Listado de todos los alumnos por curso.
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error
ID_CURSO	NUMBER	E	N	Id del curso sobre el que se quieren consultar sus alumnos.

3.4.51.3 Descripción

1. Se guarda la fecha de inicio para insertarla en el log.

2. Se imprime por pantalla la cabecera del informe.

3. Se realiza la SELECT a la vista ALUMNOSXCURSO para todos los cursos o para el curso pasado por parámetro.

4. Se imprime por pantalla las cabeceras de cada columna de la vista.

5. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.52 Listado de tipos de amonestaciones y sanciones

3.4.52.1 Funcionalidad

El procedimiento p_listadotiposamoysan, incluido en el package LISTADOS, permite consultar todos los tipos de amonestaciones y sanciones que están almacenadas en el sistema.

3.4.52.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
LIST_AMOSAN	AMOYSAN_TABLA	S	S	Listado de todas los tipos de amonestaciones y sanciones.
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error

3.4.52.3 Descripción

1. Se guarda la fecha de inicio para insertarla en el log.

2. Se imprime por pantalla la cabecera del informe.

3. Se realiza la SELECT a la vista TIPOSAMONESTACIONESYSANCIONES.

4. Se imprime por pantalla las cabeceras de cada columna de la vista.

5. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

3.4.53 Listado de amonestaciones y sanciones por alumno

3.4.53.1 Funcionalidad

El procedimiento p_listadoamoysanxalumno, incluido en el package LISTADOS, permite consultar todas las amonestaciones y sanciones impuestas por alumno.

3.4.53.2 Parámetros

NOMBRE	TIPO	DIR	OBLI	DESCRIPCIÓN
LIST_AMOSANXALUM	AMOYSANXALUM_TABLA	S	S	Listado de todas las amonestaciones y sanciones impuestas por alumno.
RSP	VARCHAR2	S	S	OK/ERROR + Descripción del error
ID_CURSO	NUMBER	E	N	Id del curso sobre el que se quieren

				consultar sus alumnos.

3.4.53.3 Descripción

1. Se guarda la fecha de inicio para insertarla en el log.

2. Se imprime por pantalla la cabecera del informe.

3. Se realiza la SELECT a la vista AMOYSANXALUMNO para todos los alumnos o para el alumno pasado por parámetro.

4. Se imprime por pantalla las cabeceras de cada columna de la vista.

5. En caso de éxito se devuelve OK y se inserta en la tabla de logs un registro.

En caso de error en cualquier paso se cancelan todos los cambios hechos en la base de datos, se inserta un registro en la tabla de LOGS y se devuelve una descripción del error en el parámetro de salida.

4. Plan de contingencias

Durante el desarrollo del proyecto existen riesgos que pueden afectar al mismo. Para minimizar el negativo efecto que estos pueden tener sobre nuestro trabajo vamos a realizar un pequeño plan de contingencias de los más probables. Este plan consiste en tres subplanes:

- Plan de respaldo: Contempla las contramedidas preventivas antes de que se materialice una amenaza. Su finalidad es evitar dicha materialización.
- Plan de emergencia: Contempla las contramedidas necesarias durante la materialización de una amenaza, o inmediatamente después. Su finalidad es paliar los efectos adversos de la amenaza.
- Plan de recuperación: Contempla las medidas necesarias después de materializada y controlada la amenaza. Su finalidad es restaurar el estado de las cosas tal y como se encontraban antes de la materialización de la amenaza.

Los riesgos que tendremos en cuenta son:

Pérdida de datos

El portátil en el que se va a trabajar puede perder los datos debido a un fallo de hardware del disco duro o general, un virus e incluso puede producirse algún tipo de error en la copia o borrado de los datos por parte del desarrollador. Al trabajar en una máquina virtual también se corre el riesgo de que el software que nos la proporciona corrompa los datos donde la almacena y sea inaccesible para nosotros.

Plan de respaldo: Se realizarán dos backups al finalizar la jornada de trabajo. Uno en el disco duro del propio portátil, fuera de la máquina virtual y otro en un dispositivo externo (memoria USB o similar). También se dispone de un programa antivirus.

Plan de emergencia: Intentar recuperar el software o hardware que ha fallado y los datos que se encuentra en él.

Plan de recuperación: Volver a instalar los programas necesarios o arreglar el ordenador y recuperar el trabajo de uno de los backups.

Administración deficiente del tiempo

Al no tener dedicación exclusiva al proyecto, es posible que los factores externos, tales como el trabajo, la familia, etc. no nos permitan dedicar el tiempo previamente planificado a la consecución del proyecto y por lo tanto nos veamos en la situación de no poder acabarlo a tiempo.

Plan de respaldo: Avanzar lo máximo posible cuando se tenga tiempo para hacerlo incluso adelantando las fechas de la planificación.

Plan de emergencia: Organizar todas las responsabilidades y replanificar si es necesario.

Plan de recuperación: Sacar tiempo de otras tareas o perder horas de sueño para llegar a tiempo en las entregas.

Planificación errónea

Muy relacionado con el punto anterior, también podemos tener problemas para entregar el proyecto a tiempo si realizamos una planificación errónea ya que trabajaremos menos de lo que necesitaríamos.

Plan de respaldo: Ajustar al máximo la planificación. Tener previstas planificaciones alternativas.

Plan de emergencia: Replanificar con un más adecuado sentido del tiempo y teniendo en cuenta las nuevas circunstancias y lo que resta para la entrega final.

Plan de recuperación: Cumplir la nueva planificación fehacientemente para llegar a tiempo a las entregas.

Diseño erróneo

Un erróneo diseño de la base de datos nos puede llevar a perder mucho tiempo en la fase de desarrollo ya que tendríamos que volver atrás y rediseñar las partes que no fuesen suficientemente eficientes.

Plan de respaldo: Estudiar concienzudamente los requisitos y la metodología de diseño para intentar ajustarlo a las peticiones lo máximo posible.

Plan de emergencia: Identificar los posibles errores y subsanarlos.

Plan de recuperación: Rediseñar lo antes posible el sistema y replanificar para volver a realizar las demás tareas a tiempo.

5. Plan de pruebas

Para realizar las pruebas hay que lanzar el script siguiente desde una consola de SQL*Plus o desde un programa de explotación de base de datos (TOAD, SQL*Developer, etc.) y se nos irá mostrando automáticamente por pantalla los procedimientos que se van a ejecutar y como quedan las tablas después de la ejecución de cada uno. Tenemos dos versiones, en una va parando la ejecución y que espera a que el usuario pulse intro para poder ir observando los resultados y la otra que ejecuta todo sin pausas. También se adjunta un archivo de texto con los resultados de una ejecución del script de pruebas en el entorno de desarrollo.

6. Valoración económica

Contaremos para este proyecto con dos recursos que se encargarán de realizar todo el trabajo. Uno será el jefe de proyecto que se encargará de la planificación inicial y de la realización de la documentación final.

Y el otro será un administrador de base de datos que realice el diseño, la implementación y las pruebas.

Tomando en cuenta la planificación acordada y aplicando las siguientes tarifas basadas en la tarifa media aproximada de la web que se adjunta:

Jefe de proyecto: 40 €/hora
http://www.infolancer.net/freelancers/jefe_de_proyecto

Administrador de base de datos: 30 €/hora
http://www.infolancer.net/freelancers/administrador_de_bases_de_datos

El coste total será:

Jefe de proyecto: 46 días * 1.5 horas * 40 € = 2760 €

Administrador de base de datos: 55 días * 1.5 horas * 30 € = 2475 €

Total proyecto: 5235 €

7. Bibliografía

- Materiales didácticos de las asignaturas Bases de Datos I, Bases de Datos II y Estructura de la información.

- Manual de documentación técnica sobre PL/SQL: Scott Urman. "Oracle9i: Programación PL/SQL", Oracle Press Osborne, McGraw Hill, 2002. (ISBN: 978-84-481-3707-6).

- Artículos sobre planes de contingencia:
  http://forodeseguridad.com/artic/segcorp/7209.htm
  http://es.wikipedia.org/wiki/Plan_de_Contingencias

- Artículo sobre creación de bases de datos:
  http://download.oracle.com/docs/cd/B10500_01/server.920/a96521/create.htm#997674

- Referencia de comandos de SQL*Plus:
  http://download.oracle.com/docs/cd/B10501_01/server.920/a90842/ch13.htm

- Preguntas y respuestas sobre ORACLE: http://www.dba-oracle.com/