

ISG3-G7

PROYECTO LA RASA



Delgado Serrano, Jose
Fernández Manzano, Antonio
Hu, Chao
Padilla Molina, Pedro

TUTOR: ALFONSO MÁRQUEZ
<https://repositorio.informatica.us.es/svn/qagjt6zmzq6y7wepg3y>

ÍNDICE

1. INTRODUCCIÓN DEL PROBLEMA.....	2
2. GLOSARIO DE TÉRMINOS.....	3
3. DESCRIPCIÓN DEL PROBLEMA.....	5
4. FOTOS.....	6
5. CATÁLOGO DE REQUISITOS.....	8
5.1. REQUISITOS GENERALES/OBJETIVOS.....	8
5.2. REQUISITOS DE INFORMACIÓN.....	9
5.3. REQUISITOS FUNCIONALES.....	11
5.4. REQUISITOS NO FUNCIONALES.....	13
5.5. REGLAS DE NEGOCIO.....	13
6. PRUEBAS DE ACEPTACIÓN.....	15
7. MODELO CONCEPTUAL.....	18
7.1. DIAGRAMAS DE CLASES UML CON RESTICCIONES.....	18
7.2. ESCENARIOS DE PRUEBA.....	19
8. MATRIZ DE TRAZABILIDAD.....	20
9. MODELO RELACIONAL.....	22
9.1. CAMBIOS DE UN MODELO A OTRO.....	23
10. SQL.....	24
10.1. CREACIÓN Y POBLACIÓN DE TABLAS.....	26
10.2. TRIGGERS	35
10.2.1 TRIGGERS ASOCIADOS A SECUENCIAS.....	36
10.2.2 TRIGGERS ASOCIADOS A SECUENCIAS.....	38
10.3. CONSULTAS.....	42
10.4. PRUEBAS.....	44

1. INTRODUCCIÓN DEL PROBLEMA

Este proyecto para la asignatura Introducción a la Ingeniería del Software y los Sistemas de Información enfocará el desarrollo de un sistema de información para una autoescuela, que permitirá administrar y controlar mejor, tanto a alumnos como profesionales que conviven en el entorno. La autoescuela “La Rasa” nos ofreció su colaboración para el desarrollo de dicho sistema.

Centrándose en la gestión de la autoescuela, se pueden diferenciar varias etapas:

El proceso de matriculación es desarrollado por la gerente de la empresa. Para ello el alumno debe acudir personalmente a la autoescuela, con el consiguiente gasto que implica el desplazamiento. Este hecho es uno de los puntos a desarrollar por el sistema, siendo la matriculación online, el objetivo principal.

Las clases teóricas son impartidas por una profesora. Suelen tener horario fijo, aunque la autoescuela ofrece la posibilidad de flexibilizar dicho horario, con un aviso con antelación.

El examen teórico escapa del dominio del sistema en cuanto a la fecha se refiere. El programa “FacilAutoGestión” ofrece una vista sobre cada alumno junto a la realización de test, así como su índice de aciertos y fallos. Detalla también los problemas de cada alumno, por lo que, en este apartado, la autoescuela no necesita mejora.

Las clases prácticas son desarrolladas por dos profesores, únicamente centrados en este dominio. La duración de cada práctica es asignada por el profesor, aunque siempre en continuo contacto con el alumno escuchando sus preferencias. Sin embargo, el horario requerido por el alumno no siempre está disponible, por lo que una previsualización de las tramas horarias disponibles para la realización de las prácticas es otro de los objetivos primordiales del sistema de información.

El examen práctico, al igual que el teórico, es controlado por la DGT. En este caso, el profesor de prácticas ofrece una visión subjetiva sobre la evolución del alumno, por lo tanto, el sistema de información no entrará en este apartado.

La autoescuela posee un sistema de información bastante completo, cuyo nombre es, “FacilAutoGestión”. Este programa almacena datos de alumnos, trabajadores, clientes, proveedores, personal administrativo, así como las horas y el desarrollo que desempeña cada uno. A pesar de ello, tras la primera reunión mantenida con el cliente, dio a conocer las carencias que poseía el programa.

Al haber analizado la situación y tras las entrevistas con el personal de la autoescuela La Rasa, se llega a la conclusión que es necesario un sistema de información que permita complementar ambos programas para su uso, tanto doméstico como en el entorno laborable.

Por tanto, el objetivo de este proyecto es la creación de un sistema de información que complemente a “FacilAutoGestión”, y capaz de subsanar las necesidades planteadas por el personal entrevistado, entre ellas, el uso doméstico del sistema y una digitalización del sistema de prácticas de alumnos actual.

2. GLOSARIO DE TÉRMINOS

Accidente: Suceso imprevisto que tiene lugar en el desarrollo de las clases prácticas, en el que se ven involucrados uno o más vehículos. Normalmente sucede por un fallo de algún conductor/a o debido a una avería de uno o varios vehículos.

Alumno/a: Cliente que está asistiendo a clases teóricas o clases prácticas en la autoescuela.

Avería: Fallo en el vehículo usado para impartir las clases prácticas, puede ser tanto un fallo mecánico como eléctrico.

Bono de Práctica: Cupón / ticket que le permite al alumno/a dar la clase práctica con su respectivo profesor/a de prácticas.

Clases Prácticas: Clases impartidas por el profesor/a en el vehículo de la autoescuela. Puede variar su duración entre 30', 45', 1h, donde se enseña al alumno a manejar el vehículo por la vía pública cumpliendo todas sus normas para poder superar el examen práctico.

Clases Teóricas: Clases impartidas por el profesor/a en la autoescuela de cara al examen teórico donde se explican distintos temas enfocados hacia el ámbito de las normas de circulación y conducción en la vía pública.

Conductor/a: Persona que circula con un vehículo por las carreteras de la vía pública.

Examen Oficial: Prueba que debe realizar el alumno/a para obtener el permiso correspondiente, realizado en la DGT. Se divide en examen teórico y práctico. La calificación será siempre de APTO o NO APTO.

Examen Práctico: Examen que se realiza en el coche de la autoescuela donde un examinador/a de la DGT valorará si eres apto para la circulación en las carreteras de la vía pública. El profesor/a de prácticas te acompañará y actuará si lo cree necesario. Si el profesor/a tiene que intervenir el alumno/a será calificado automáticamente como no apto.

Examen Teórico: Examen que consta de 20 o 30 preguntas de las cuales solo puedes fallar un 10% como máximo para superar la prueba y al cual te preparas realizando unos test y asistiendo a las clases teóricas impartidas en la autoescuela. Una vez aprobado se podrá proceder a comenzar a preparar el examen práctico.

Expediente de Prácticas: Documento que permite tener un seguimiento de las clases prácticas de cada alumno/a que se archiva junto a los bonos de prácticas correspondientes a cada alumno.

DGT: Las siglas significan Dirección General de Tráfico y es donde se presentará el alumno/a a los exámenes oficiales.

Peatón: Persona que transita por la vía pública.

Permiso de Conducción / Carnet: Documento oficial que se obtiene tras haber aprobado las distintas fases del examen y que te permite circular por las carreteras de la vía pública. Dependiendo del tipo de permiso obtenido se podrá circular con solo determinados vehículos. Los permisos impartidos son: B, AM, A1, A2, A.

Profesor/a: Trabajador de la autoescuela que imparte clases teóricas o prácticas.

Temario: Contenido que se debe aprender de cara al examen oficial.

Test: Simulación del examen teórico que se hace recogiendo anteriores preguntas que han aparecido en exámenes oficiales anteriores. Puede contener, dependiendo del permiso, de 20 o 30 preguntas. Para superar el test debes tener un porcentaje de acierto igual o mayor del 90%.

Vehículo: Herramienta de utilización en las clases prácticas para el desarrollo del aprendizaje del alumno para la circulación en vía pública. Existen diferentes tipos de vehículos entre los que se encuentran: turismos, camiones, ciclomotores, motocicletas, autocaravanas...

Vía Pública: Conjunto de carreteras y vías por donde se puede transitar cumpliendo ciertas normas preestablecidas.

3. VISIÓN GENERAL DEL SISTEMA

El sistema de información que posee la autoescuela actualmente engloba a todos los trabajadores, así como a los alumnos. Nuestro sistema se centrará en los alumnos y en la facilitación de trabajo a los profesores.

El funcionamiento de la herramienta es bastante completo, aunque se pueden destacar varios problemas sobre los que se basará nuestro sistema.

En primer lugar, el uso de la herramienta es exclusivo de la autoescuela, es decir, no se puede disponer de su uso doméstico. Esto supone una ralentización del trabajo administrativo, así como de consulta sobre el desarrollo del alumnado.

La herramienta posee un apartado de impresión de bonos de prácticas para validar la asistencia a las clases. Estos bonos tienen un formato demasiado pequeño, por lo que la persona entrevistada nos hizo saber sobre el problema que le plantea, ya que cada bono ocupa una cuarta parte del papel. Esto supone además del alto coste en folios, una pérdida de tiempo que entorpece el trabajo administrativo.

Otro conflicto presentado por el personal es la entrada de datos en el programa. El problema principal surge en la utilización de caracteres especiales, como “*” “/” “_”, entre otros muchos. La omisión de dichos caracteres es uno de los objetivos del proyecto a realizar, ya que complica la escritura de fechas, horas, y nombres de alumnos, ya que el programa posee un estilo de entrada predeterminado.

Por último, la accesibilidad y el rendimiento del programa es limitado, tanto la velocidad de procesamiento como la restricción de acceso a determinado personal administrativo. La utilidad de ampliar dicha restricción fue uno de los puntos clave planteado por la autoescuela.

En conclusión, nuestro objetivo es mejorar cada uno de los puntos débiles que presenta la aplicación que posee la autoescuela, facilitando así su uso, tal y como requirió la persona entrevistada.

4. FOTOS

[illegible]

Programa que usa la autoescuela “FacilAutoGestión”



Interior de la autoescuela



Fotos de la 1ª Reunión que tiene lugar en la autoescuela

5. CATÁLOGO DE REQUISITOS

5.1. REQUISITOS GENERALES/OBJETIVOS

Gestión del profesorado: la base de datos llevará a cabo la distinción de profesores prácticos de los teóricos, las clases impartidas, el sueldo, las pagas extras, la actividad actual (distinción entre profesor que está de alta y de baja) y las vacaciones.

Gestión de clases: el sistema de información permitirá registrar según las distintas clases: el número de prácticas dadas y el profesor que las imparte.

Gestión de matriculación: el sistema permitirá la matriculación de un alumno.

Gestión del cliente: la base de datos mostrará las clases que le imparten al cliente, el importe recibido. También diferenciará entre clases teóricas y prácticas que se le darán al cliente y distinguirá si el usuario ya obtuvo el carné o no.

Gestión del alumno: se conocerá con el sistema de información: el número de matrícula del alumno, el permiso al que está matriculado para su obtención, el estado en que está dicho alumno (parte teórica, práctica, finalizado o baja) y el número de clases prácticas realizadas.

Gestión de bonos: la base de datos dará a conocer si éste está sellado, si el número de referencia es correcto, su importe y la duración de la práctica a la que se puede acudir con dicho bono.

5.2. REQUISITOS DE INFORMACIÓN

RI 001 – Información de los alumnos

Como profesor

Quiero disponer de la información de mis alumnos [(Nº Matrícula, permiso, estado, clases prácticas)(el alumno elige una tarifa, firma un contrato, es enseñado por un profesor, al que entrega un bono comprado por él, con el fin de realizar un examen)].

Para poder apuntarlos al examen.

RI 002 - Información sobre los horarios

Como directora de la autoescuela

Quiero disponer de las horas de trabajo de mis empleados [(Profesor(clasesImpartidas), Bono(Duracion))(El profesor recibe un bono, enseña a un alumno, e imparte clase a un cliente)]

Para llevar un control exhaustivo de las horas trabajadas

RI 003 – Información sobre fechas de exámenes

Como alumno

Quiero conocer la fecha de los exámenes[(Examen(Fecha))(El examen es realizado por un alumno, y supervisado por un examinador)]

Para elegir la que mejor me convenga

RI 004 – Información sobre ingresos

Como director

Quiero disponer de los ingresos generados[(Factura(Importe))(La factura es pagada por un cliente, y abonada por una empresa)]

Para llevar un control de la liquidez de la autoescuela

RI 005 – Información sobre profesores

Como usuario

Quiero conocer los profesores entre los que puedo elegir [(nombre, apellidos, dni, teléfono, fecha de nacimiento y correo)]

Para seleccionar al que prefiera

RI 006 – Información sobre el vehículo

Como alumno

Quiero conocer el estado del vehículo que voy a usar para las prácticas[(marca, matrícula. Seguro, fechaITV, estado)(El vehículo es usado por un usuario, y corresponde a un profesor de prácticas)]

Para asegurarme de dar las clases con un vehículo en buenas condiciones

RI 007 – Información sobre permisos de conducir

Como usuario

Quiero conocer los tipos de permisos que imparte la autoescuela[(AM,A1,A2,A,B)(Permiso es un tipo enumerado)]

Para poder elegir el adecuado

RI 008 – Información sobre el número de alumnos

Como gestora

Quiero conocer el número de alumnos de los distintos permisos[(TipoPermiso)AM,A1,A2,A,B-TipoEnum]

Para saber el número de profesores que voy a necesitar

RI 009 – Información sobre las clases impartidas

Como profesor

Quiero conocer el número de clases que voy a impartir[(Profesor)ClasesImpartidas-Integer]

Para optimizar el tiempo empleado[Profesor enseña a Alumno, Profesor imparte a Cliente]

RI 010 – Información sobre aprobados

Como director

Quiero conocer el porcentaje de aprobados [Examen(NºAlumnosAptos)]

Para controlar el rendimiento de los profesores[Alumno realiza Examen]

RI 011 – Información sobre vacaciones

Como profesor

Quiero conocer el número de días de vacaciones de las que dispongo[Profesor(Vacaciones)]

Para poder organizar los días a lo largo del año

RI 012 – Información sobre tarifas

Como alumno

Quiero disponer de las tarifas que ofrece la autoescuela

Para elegir la que más se adapte a mi situación[Alumno elige Tarifa, Contrato tiene Tarifa]

RI 013 – Información de importes

Como alumno

Quiero conocer el importe total pagado[Contrato(Precio), Tarifa(Descuento), Bono(Importe)]

Para controlar el dinero invertido[Alumno elige Tarifa, Contrato tiene Tarifa, Alumno compra Bono, Profesor recibe Bono]

RI 014 – Información sobre el sueldo y pagas extras

Como profesor

Quiero acceder a la información de mis pagas extras[Profesor(PagaExtra), Profesor(Sueldo)]

Para llevar un control sobre mi sueldo

5.3. REQUISITOS FUNCIONALES

RF 001 – Ordenación de alumnos

Como usuario

Quiero que el sistema permita ordenar de distintas formas a los alumnos

Para poder hacer una búsqueda de manera más sencilla

RF 002 – Acceso desde múltiples plataformas

Como gestora

Quiero que el sistema permita trabajar desde mi domicilio

Para aprovechar el tiempo disponible

RF 003 – Cambios en examen

Como profesor

Quiero que el sistema permita modificar los alumnos que se presentan a examen

Para poder realizar cualquier cambio necesario

RF 004 – Modificación de la fecha ITV

Como profesor de prácticas

Quiero que el sistema permita modificar la fecha de la ITV

Para poder circular con el vehículo legalmente

RF 005 – Impresión de bonos

Como gestora

Quiero que el sistema permita la impresión de bonos

Para poder agilizar la gestión de las clases prácticas

RF 006 – Eliminación de clientes

Como gestora

Quiero que el sistema permita eliminar a los clientes

Para poder llevar un mejor control

RF 007 – Adición de tarifas

Como gestora

Quiero que el sistema permita añadir una tarifa

Para poder ofrecerla a los alumnos

RF 008 – Tarifa sin usar

Como gerente

Quiero que el sistema permita saber aquellas tarifas sin usar

Para poder ofrecerlas a los alumnos

RF 009 – Tipo de alumnos mas frecuentes

Como profesor

Quiero que el sistema permita conocer los alumnos que están estudiando cada parte del permiso

Para que no haya una carga extra de trabajo

RF 010 – Alumnos con más de 15 clases prácticas

Como gerente

Quiero que el sistema permita seleccionar aquellos alumnos con más de 15 clases prácticas

Para saber los alumnos que estarán próximamente preparados para el examen práctico

RF 011 – Profesor con más alumnos

Como gerente

Quiero que el sistema permita conocer el profesor con más alumnos asignados

Para poder discernir a que profesor debo asignarle más alumnos

RF 012 – Vehículos con la ITV pasada

Como profesor

Quiero que el sistema permita ver los vehículos que tienen la ITV al día

Para conocer los vehículos que están a mi disposición

RF 013 – Sueldo total a profesores

Como directora

Quiero que el sistema permita sumar el sueldo de todos los profesores tanto de prácticas o de teoría

Para poder gestionar las finanzas de la autoescuela con eficacia

RF 014 – Alumnos totales matriculados en la autoescuela

Como gerente

Quiero que el sistema permita saber el número total de alumnos matriculados en la autoescuela

Para poder ver el número total de alumnos que maneja la autoescuela

5.4. REQUISITOS NO FUNCIONALES

RNF 001 – Manejo del sistema

Como gestora

Quiero que el sistema pueda ser manejado fácilmente

Para un cómodo entendimiento con el usuario

RNF 002 – Disponibilidad 24 horas

Como usuario

Quiero que el sistema esté disponible las 24 horas

Para poder utilizarlo cuando lo necesite

RNF 003 – Búsquedas rápidas

Como usuario

Quiero que el sistema realice las búsquedas lo más rápido posible

Para evitar perder tiempo por fallos computacionales

RNF 004 – Requisitos de seguridad

Como usuario

Quiero que el sistema cumpla los requisitos de protección

Para que no haya ningún problema legal

5.5. REGLAS DE NEGOCIO

RN 001 – Uso del vehículo

Como director

Quiero que el vehículo solo pueda pertenecer al profesor de prácticas

Para garantizar un uso correcto

RN 002 - Entrega de bonos

Como director

Quiero que el bono solo pueda ser entregado a un profesor de prácticas

Para que no haya un exceso de bonos

RN 003 – Control de vacaciones

Como gestora

Quiero que los días de vacaciones sean máximo de 30 días

Para que cumpla lo estipulado por el contrato

RN 004 – Días de baja

Como gestora

Quiero que los días de baja sean máximo 60 días, salvo excepciones

Para tener un control sobre los trabajadores

RN 005 – Clases impartidas

Como director

Quiero que se cumpla: las clases máximas impartidas sean 40 clases/semana

Para garantizar que cumple lo estipulado por ley

RN 006 – Tarifa única

Como director

Quiero que a cada contrato se le asigne una sola tarifa

Para administrar los descuentos elegidos por los alumnos

RN 007 – Alumnos examinados

Como gestora

Quiero que el alumno solo pueda presentarse a examen si ha aprobado el teórico

Para llevar un control sobre el alumnado

6. PRUEBAS DE ACEPTACIÓN

RN 001 – Uso del vehículo


Como director

Quiero que el vehículo solo pueda pertenecer al profesor de prácticas

Para garantizar un uso correcto



Se asigna un vehículo a un profesor de prácticas. El sistema no devuelve ningún error.

Se intenta asignar un vehículo a un profesor de teoría, y se recibe un error porque el vehículo es exclusivo del profesor de prácticas. 

RN 002 - Entrega de bonos


Como director

Quiero que el bono solo pueda ser entregado a un profesor de prácticas

Para que no haya un exceso de bonos



Se entrega un bono a un profesor de prácticas. El sistema no devuelve ningún error.

Se intenta asignar un bono a un usuario distinto al profesor de prácticas. El sistema devuelve un error porque el bono es exclusivo para profesores de prácticas 

RN 003 – Control de vacaciones


Como gestora

Quiero que los días de vacaciones sean máximo de 30 días

Para que cumpla lo estipulado por el contrato



Se registra unas vacaciones de un profesor de 28 días. El sistema no devuelve ningún error.

Se intenta registrar un número de vacaciones superior a 30 días, por ejemplo 32 días, y se recibe un error debido a que el máximo de días de vacaciones son 30. 

RN 004 – Días de baja


Como gestora

Quiero que los días de baja sean máximo 60 días, salvo excepciones

Para tener un control sobre los trabajadores



Se registra días de baja de un trabajador, y se le asigna el valor de 50. El sistema no devuelve ningún error.

Se intenta registrar 69 días de baja de un trabajador. El sistema devolverá un error porque supera el máximo. 

RN 005 – Horas trabajadas


Como director

Quiero que se cumpla: las clases máximas impartidas sean 40 clases/semana

Para garantizar que cumple lo estipulado por ley

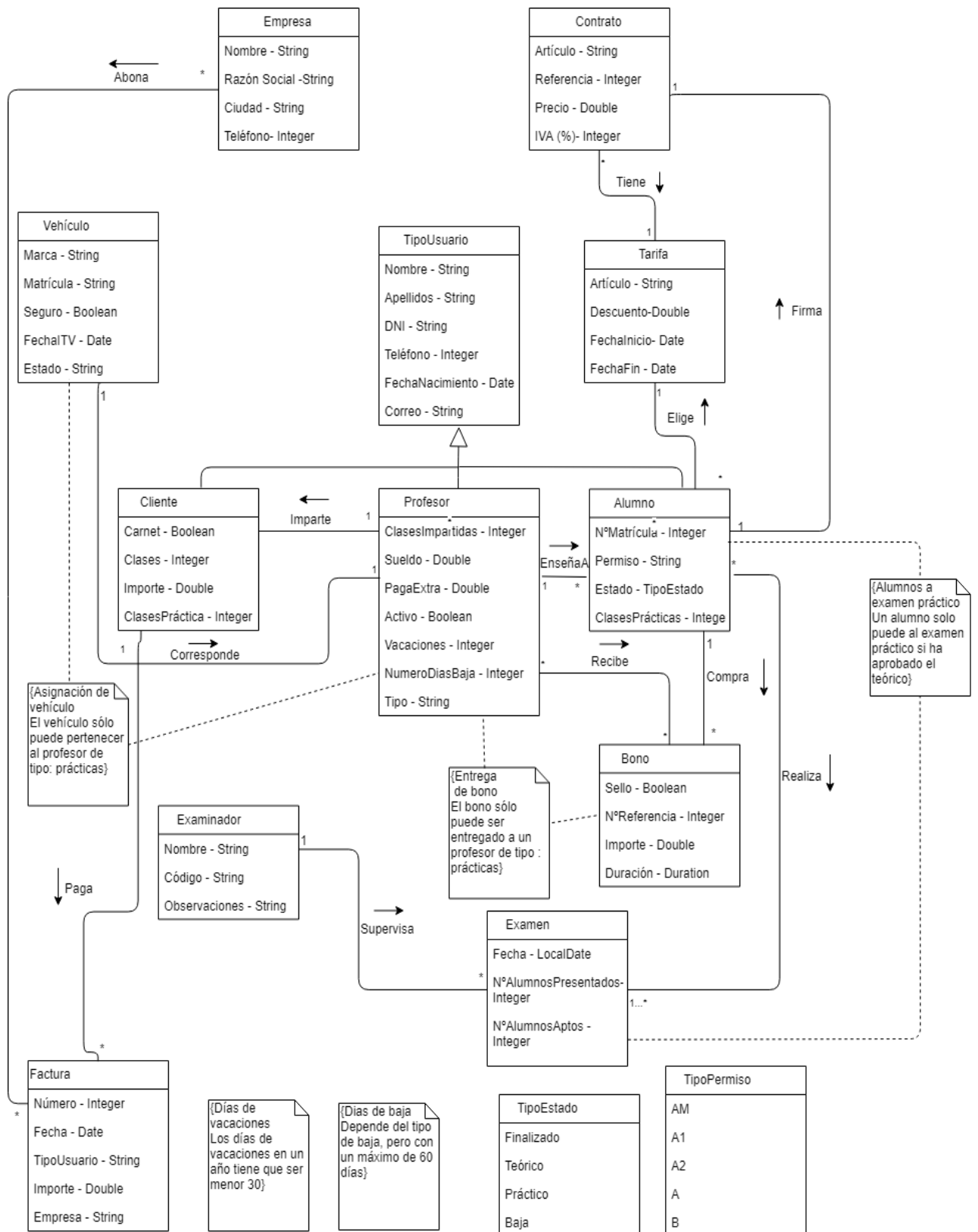


Se intenta registrar clases impartidas por un profesor con valor de 30. El sistema no devuelve ningún error.

Se intenta registrar clases impartidas con valor de 50. El sistema devuelve un error debido a que supera el máximo permitido. 

7. MODELO CONCEPTUAL

7.1. DIAGRAMA DE CLASES UML CON RESTRICCIONES



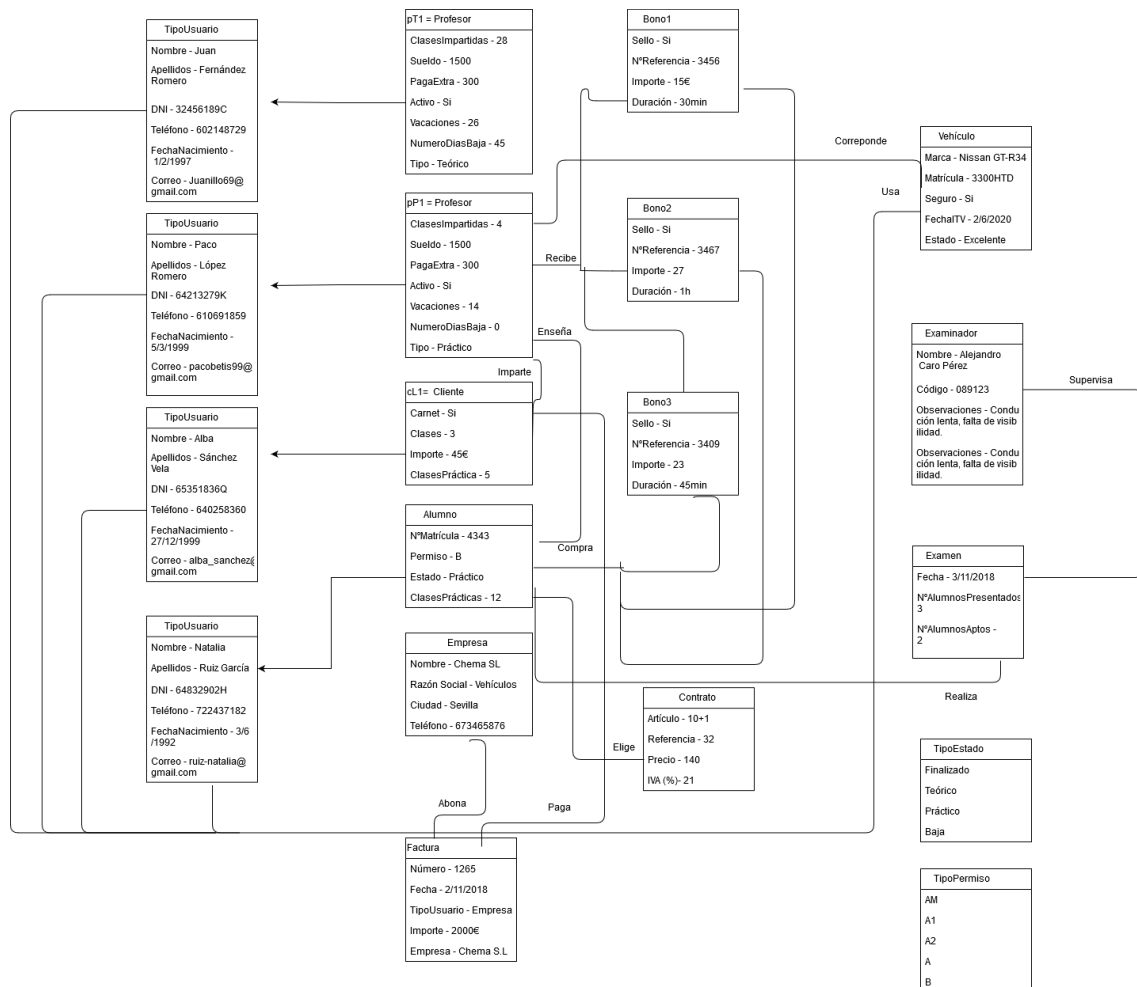
7.2. ESCENARIOS DE PRUEBA

En este escenario de prueba tenemos una empresa y 4 personas, de las cuales 2 son profesores (1 práctico y 1 teórico), 1 cliente 1 alumno. Llamaremos a los usuarios: **Empresa**, **pP1**, **pT1**, **cL1** y **Alumno**, respectivamente.

El profesor **pP1** tiene asignado un vehículo **Vehículo** y recibe tres bonos **Bono1**, **Bono2**, **Bono3** del mismo alumno para dar 3 prácticas ese día.

El cliente **cL1** paga una factura **f1** por dar clases ese día.

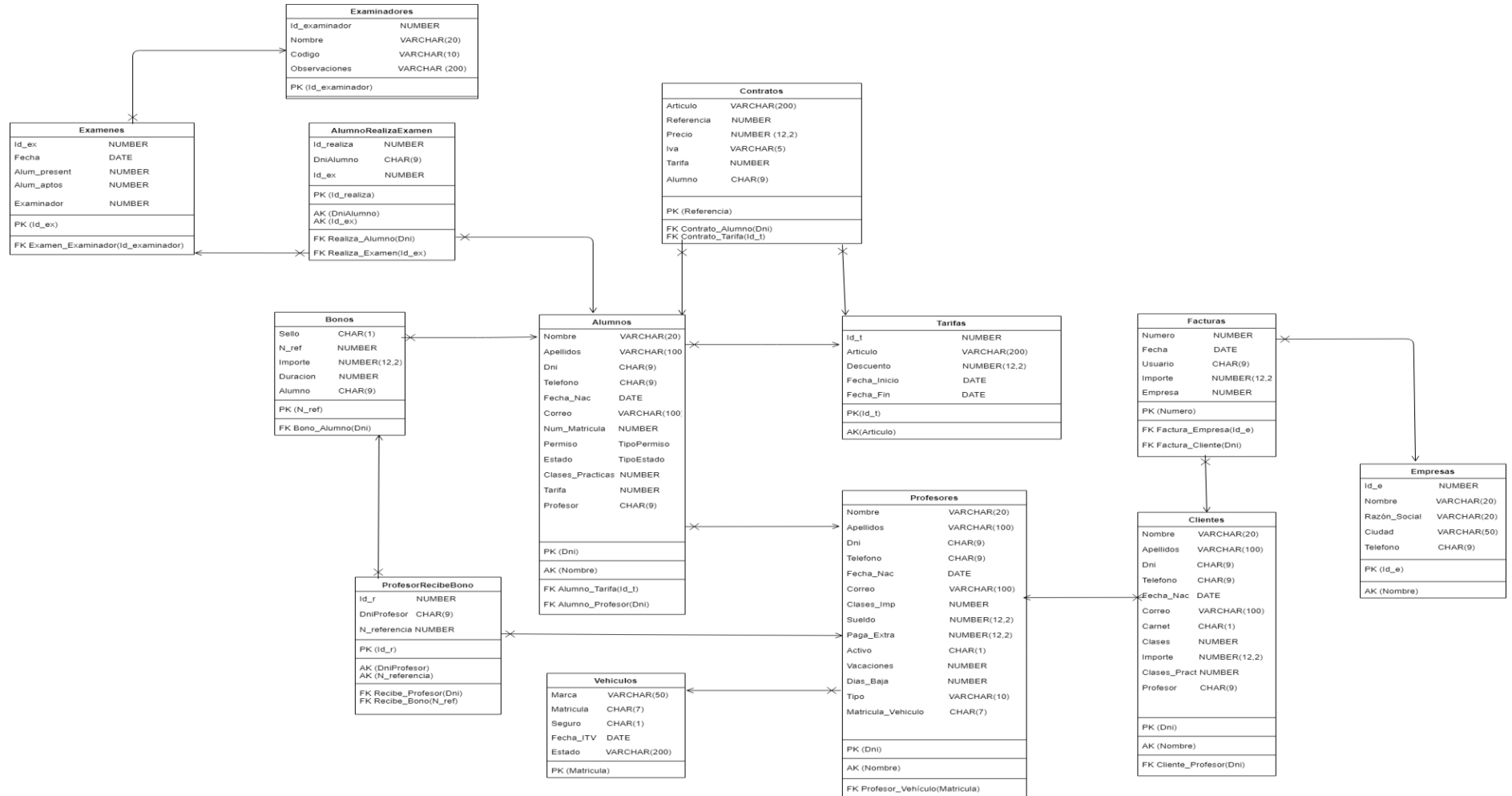
El alumno **aL1** elige un contrato **c1**. Realizará un examen **Examen**, que estará supervisado por el examinador **Examinador**.



8. MATRIZ DE TRAZABILIDAD

Requisitos→ Clases/A sociaciones/Restricciones	RI-001 INFORMACIÓN DE LOS ALUMNOS	RI-002 INFORMACIÓN SOBRE LOS HORARIOS	RI-003 INFORMACIÓN SOBRE FECHAS DE EXAMENES	RI-004 INFORMACIÓN SOBRE INGRESOS	RI-005 INFORMACIÓN SOBRE PROFESORES	RI-006 INFORMACIÓN SOBRE EL VEHÍCULO	RI-007 INFORMACIÓN SOBRE PERMISOS DE CONDUCIR	RI-008 INFORMACIÓN SOBRE EL NÚMERO DE ALUMNOS	RI-009 INFORMACIÓN SOBRE LAS CLASES IMPARTIDAS	RI-010 INFORMACIÓN SOBRE APROBADOS	RI-011 INFORMACIÓN SOBRE VACACIONES	RI-012 INFORMACIÓN SOBRE TARIFAS	RI-013 INFORMACIÓN DE IMPORTES	RI-014 INFORMACIÓN SOBRE EL SUELDO Y PAGAS EXTRA	RN-001 USO DEL VEHÍCULO	RN-002 ENTREGA DE BONOS	RN-003 CONTROL DE VACACIONES	RN-004 DÍAS DE BAJA	RN-005 CLASES IMPARTIDAS
EMPRESA				✓									✓						
CONTRATO				✓			✓	✓				✓	✓						
TIPOUSUARIO	✓	✓	✓		✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
VEHÍCULO						✓													
CLIENTE				✓	✓	✓							✓						✓
PROFESOR		✓		✓	✓	✓			✓		✓		✓	✓	✓	✓	✓	✓	✓
ALUMNO	✓		✓	✓	✓	✓	✓	✓		✓		✓	✓	✓					✓
EXAMINADOR										✓									
EXAMEN	✓		✓							✓									
BONO				✓	✓								✓			✓			✓
FACTURA				✓	✓								✓						
paga				✓	✓								✓						
supervisa										✓									
realiza	✓		✓							✓									
compra				✓	✓								✓			✓			✓
recibe				✓	✓								✓			✓			✓
imparte				✓	✓	✓							✓						✓
enseña				✓	✓	✓							✓						✓
usa				✓	✓	✓													✓
abona				✓															
elige				✓	✓		✓	✓				✓	✓						
corresponde				✓	✓	✓									✓				✓
R1-Vehículo				✓	✓	✓									✓				✓
R2-Bono		✓											✓			✓			
R3-Vacaciones		✓							✓		✓			✓			✓		
R4-Días baja									✓		✓			✓				✓	

9. MODELO RELACIONAL



9.1. CAMBIOS DE UN MODELO A OTRO

Tabla	Atributos	Primary Keys	Alternate Keys	Foreign Keys
Vehiculos	Marca, Matricula, Seguro, Fecha_itv, Estado	Matricula	-	-
ProfesorRecibeBono	Id_r, DniProfesor, N_referencia	Id_r	DniProfesor, N_referencia	DniProfesor/Profesores (Dni) N_referencia/Bonos (N_ref)
Empresas	Id_e, Nombre, Razón_Social, Ciudad, Telefono	Id_e	Nombre	-
AlumnoRealizaExamen	Id_realiza, DniAlumno, Id_ex	Id_realiza	DniAlumno, Id_ex	DniAlumno/Alumnos (Dni) Id_ex/Exámenes (Id_ex)
Profesores	Nombre, Apellidos, Dni, Teléfono, Fecha_Nac, Correo, Clases_Imp, Sueldo, Paga_Extra, Activo, Vacaciones, Días_Baja, Tipo, Matricula_Vehiculo	Dni	Nombre	Matricula_Vehiculo/Vehiculos
Facturas	Numero, Fecha, Usuario, Importe, Empresa	Numero	-	Empresa/Empresas (Nombre), Usuario/Clientes (Nombre)
Alumnos	Nombre, Apellidos, Dni, Telefono, Fecha_Nac, Correo, Num_Matricula, Permiso, Estado, Clases_Practicas, Tarifa, Profesor	Dni	Nombre	Tarifa/Tarifas (Id_t) Profesor/Profesores (Nombre)
Clientes	Nombre, Apellidos, Dni, Telefono, Fecha_Nac, Correo, Carnet, Clases, Importe, Clases_Pract, Profesor	Dni	Nombre	Profesor/Profesores (Nombre)
Tarifas	Id_t, Artículo, Descuento, Fecha_Inicio, Fecha_Fin	Id_t	Artículo	-
Examinadores	Id_examinador, Nombre, Código, Observaciones	Id_examinador	Nombre	
Bonos	Sello, N_ref, Importe, Duración, Alumno	N_Ref	-	Alumno/Alumnos (Dni)
Contratos	Artículo, Referencia, Precio, Iva, Tarifa	Referencia	-	Alumno/Alumnos (Dni) Tarifa/Tarifas (Artículo)
Exámenes	Id_ex, Fecha, Alum_present, Alum_aptos, Examinador	Id_ex	-	Examinador/Examinadores (Id_examinador)

-Transformación de entidades:

Hemos transformado el modelo conceptual al modelo relacional, para ello cada entidad se ha transformado en una relación cuyo nombre ha pasado a estar en plural, con sus correspondientes atributos necesarios.

Los atributos que eran claves sencillas, se han usado para crear claves primarias. En los que no se cumplía esto se crearon IDs para definir las claves primarias.

En las relaciones donde el atributo era clave semántica y no se definió como primaria, se ha usado como clave alternativa.

Los enumerados han pasado a ser restricciones asociados al atributo de su correspondiente tabla.

-Transformación de asociaciones:

En las asociaciones con cardinalidad 1:X, se ha creado un atributo en la tabla con participación (X) que será una clave ajena apuntando al atributo con clave primaria de la tabla con participación (1).

En las asociaciones con cardinalidad 1:1, se ha creado un atributo en una de las tablas que será una clave ajena apuntando al atributo con clave primaria de la otra tabla.

En las asociaciones con cardinalidad X:Y, se ha creado una tabla auxiliar que relacione ambas entidades, la cual tendrán atributos que serán una clave primaria compuesta y ajenas que apuntan al atributo con clave primaria de cada tabla de cada lado de la asociación.

-Transformación de clasificaciones:

El modelo conceptual cuenta con entidades que son heredadas de otras entidades jerárquicamente superiores como es el caso de TipoUsuario, que hereda a Cliente, Profesor y Alumno, la cual se trata de una clasificación disjunta y completa.

-Normalización del modelo relacional:

Está en 1FN porque los atributos son mono-valuados.

Está en 2FN porque está en 1FN y todos los atributos que no forman parte de ninguna clave candidata son completamente dependientes de las claves candidatas de la relación.

Está en 3FN porque está en 2FN y no existen dependencias transitivas.

10. SQL

10.1. CREACIÓN Y POBLACIÓN DE TABLAS

```
DROP TABLE ALUMNOREALIZAEXAMEN;  
DROP TABLE PROFESORRECIBEBONO;  
DROP TABLE FACTURAS;  
DROP TABLE BONOS;  
DROP TABLE CONTRATOS;  
DROP TABLE CLIENTES;  
DROP TABLE ALUMNOS;  
DROP TABLE PROFESORES;  
DROP TABLE EXAMENES;  
DROP TABLE EXAMINADORES;  
DROP TABLE TARIFAS;  
DROP TABLE EMPRESAS;  
DROP TABLE VEHICULOS;
```

```
CREATE TABLE Vehiculos  
(  
  Marca VARCHAR(50) NOT NULL,  
  Matricula CHAR(7) NOT NULL,  
  Seguro CHAR(1) NOT NULL,  
  Fecha_ITV DATE NOT NULL,  
  Estado VARCHAR(200) NOT NULL,  
  PRIMARY KEY (Matricula)  
);
```

```
CREATE TABLE Empresas  
(  
  Id_e NUMBER NOT NULL,  
  Nombre VARCHAR(20) NOT NULL,  
  Razon_Social VARCHAR(20),  
  Ciudad VARCHAR(50) NOT NULL,
```

```
Telefono CHAR(9) NOT NULL,  
PRIMARY KEY (Id_e),  
UNIQUE (Nombre)  
);
```

```
CREATE TABLE Tarifas  
(  
Id_t NUMBER NOT NULL,  
Articulo VARCHAR(200) NOT NULL,  
Descuento NUMBER(12,2) NOT NULL,  
Fecha_Inicio DATE NOT NULL,  
Fecha_Fin DATE NOT NULL,  
CONSTRAINT fechas CHECK (Fecha_Inicio<Fecha_Fin),  
PRIMARY KEY (Id_t),  
UNIQUE (Articulo)  
);
```

```
CREATE TABLE Examinadores  
(  
Id_examinador NUMBER NOT NULL,  
Nombre VARCHAR(20) NOT NULL,  
Codigo VARCHAR(10) NOT NULL,  
Observaciones VARCHAR(200),  
PRIMARY KEY (Id_examinador),  
UNIQUE (Nombre)  
);
```

```
CREATE TABLE Examenes  
(  
Id_ex NUMBER NOT NULL,  
Fecha DATE NOT NULL,  
Alum_present NUMBER NOT NULL,  
Alum_aptos NUMBER,  
Examinador NUMBER NOT NULL,  
CONSTRAINT diferencia CHECK (Alum_aptos<=Alum_present),  
PRIMARY KEY (Id_ex),  
FOREIGN KEY (Examinador) REFERENCES Examinadores (Id_examinador)
```

);

```
CREATE TABLE Profesores
(
Nombre VARCHAR(20) NOT NULL,
Apellidos VARCHAR(100) NOT NULL,
Dni CHAR(9) NOT NULL,
Telefono CHAR(9) NOT NULL,
Fecha_Nac Date NOT NULL,
Correo VARCHAR(100),
Clases_Imp NUMBER NOT NULL,
Sueldo NUMBER(12,2) NOT NULL,
Paga_Extra NUMBER(12,2),
Activo CHAR(1) NOT NULL,
Vacaciones NUMBER NOT NULL,
Dias_Baja NUMBER,
Tipo VARCHAR(10) NOT NULL CHECK(Tipo IN('Prácticas','Teóricas')),
Matricula_Vehiculo CHAR(7) NOT NULL,
PRIMARY KEY (Dni),
UNIQUE (Nombre),
FOREIGN KEY (Matricula_Vehiculo) REFERENCES Vehiculos (Matricula),
CONSTRAINT diasBaja CHECK (Dias_Baja<61),
CONSTRAINT diasVacaciones CHECK (Vacaciones<30)
);
```

```
CREATE TABLE Alumnos
(
Nombre VARCHAR(20) NOT NULL,
Apellidos VARCHAR(100) NOT NULL,
Dni CHAR(9) NOT NULL,
Telefono CHAR(9) NOT NULL,
Fecha_Nac Date NOT NULL,
Correo VARCHAR(100) NOT NULL,
Num_Matricula NUMBER NOT NULL,
Permiso VARCHAR(100) NOT NULL CHECK( Permiso IN ('AM','A1','A2','A','B')),
Estado VARCHAR(100) NOT NULL CHECK (Estado IN ('Finalizado','Teórico','Práctico','Baja')),
Clases_Practicas NUMBER NOT NULL,
```



```
Tarifa NUMBER NOT NULL,  
Profesor CHAR(9) NOT NULL,  
PRIMARY KEY (Dni),  
UNIQUE (Nombre),  
FOREIGN KEY (Tarifa) REFERENCES Tarifas (Id_t),  
FOREIGN KEY (Profesor) REFERENCES Profesores (Dni)  
);
```

```
CREATE TABLE Clientes  
(  
Nombre VARCHAR(20) NOT NULL,  
Apellidos VARCHAR(100) NOT NULL,  
Dni CHAR(9) NOT NULL,  
Telefono CHAR(9),  
Fecha_Nac Date NOT NULL,  
Correo VARCHAR(100),  
Carnet CHAR(1) NOT NULL,  
Clases NUMBER NOT NULL,  
Importe NUMBER(12,2) NOT NULL,  
Clases_Pract NUMBER NOT NULL,  
Profesor CHAR(9) NOT NULL,  
PRIMARY KEY (Dni),  
UNIQUE (Nombre),  
FOREIGN KEY (Profesor) REFERENCES Profesores (Dni)  
);
```

```
CREATE TABLE Contratos  
(  
Articulo VARCHAR(200) NOT NULL,  
Referencia NUMBER NOT NULL,  
Precio NUMBER(12,2) NOT NULL,  
Iva VARCHAR(5) NOT NULL,  
Tarifa NUMBER NOT NULL,  
Alumno CHAR(9) NOT NULL,  
PRIMARY KEY (Referencia),  
FOREIGN KEY (Alumno) REFERENCES Alumnos (Dni),  
FOREIGN KEY (Tarifa) REFERENCES Tarifas (Id_t)
```

```
);
```

```
CREATE TABLE Bonos
```

```
(  
Sello CHAR(1) NOT NULL,  
N_ref NUMBER NOT NULL,  
Importe NUMBER (12,2) NOT NULL,  
Duracion NUMBER NOT NULL,  
Alumno CHAR(9) NOT NULL,  
PRIMARY KEY (N_ref),  
FOREIGN KEY (Alumno) REFERENCES Alumnos (Dni)  
);
```

```
CREATE TABLE Facturas
```

```
(  
Numero NUMBER NOT NULL,  
Fecha DATE NOT NULL,  
Usuario CHAR(9) NOT NULL,  
Importe NUMBER (12,2) NOT NULL,  
Empresa NUMBER NOT NULL,  
PRIMARY KEY (Numero),  
FOREIGN KEY (Empresa) REFERENCES Empresas (Id_e),  
FOREIGN KEY (Usuario) REFERENCES Clientes (Dni)  
);
```

```
CREATE TABLE ProfesorRecibeBono
```

```
(  
Id_r NUMBER NOT NULL,  
DniProfesor CHAR(9) NOT NULL,  
N_referencia NUMBER NOT NULL,  
PRIMARY KEY (Id_r),  
UNIQUE (DniProfesor),  
UNIQUE (N_referencia),  
FOREIGN KEY (DniProfesor) REFERENCES Profesores (Dni),  
FOREIGN KEY (N_referencia) REFERENCES Bonos (N_ref)  
);
```

```
CREATE TABLE AlumnoRealizaExamen
```

```
(  
Id_realiza NUMBER NOT NULL,  
DniAlumno CHAR(9) NOT NULL,  
Id_ex NUMBER NOT NULL,  
PRIMARY KEY (Id_realiza),  
UNIQUE(DniAlumno),  
UNIQUE(Id_ex),  
FOREIGN KEY (DniAlumno) REFERENCES Alumnos(Dni),  
FOREIGN KEY (Id_ex) REFERENCES Examenes(Id_ex)  
);
```

```
INSERT INTO Vehiculos VALUES ('Audi', '4157HJL', 'S', TO_DATE('02/02/2019', 'dd/mm/yyyy'),'Vehículo en perfectas condiciones para su uso');
```

```
INSERT INTO Vehiculos VALUES ('Volswagen', '5177DGR', 'S', TO_DATE('03/04/2019', 'dd/mm/yyyy'),'Vehículo en perfectas condiciones para su uso');
```

```
INSERT INTO Vehiculos VALUES ('Toyota', '7237HJL', 'S', TO_DATE('12/02/2019', 'dd/mm/yyyy'),'Vehículo en perfectas condiciones para su uso');
```

```
INSERT INTO Vehiculos VALUES ('Audi', '4128FJV', 'S', TO_DATE('22/10/2019', 'dd/mm/yyyy'),'Vehículo en perfectas condiciones para su uso');
```

```
INSERT INTO Vehiculos VALUES ('Suzuki', '1127GJM', 'S', TO_DATE('21/05/2019', 'dd/mm/yyyy'),'Rueda delatara derecha con presión baja, pero apto para su uso');
```

```
INSERT INTO Empresas VALUES (1,'Anemia.S.L', 'gasolina', 'Guatemala', '631942195');
```

```
INSERT INTO Empresas VALUES (2,'ADDACARRITO', 'Miguel Toro', 'US', '613849120');
```

```
INSERT INTO Empresas VALUES (3,'TARFIA', 'Mala Vida', 'Sevilla', '631948301');
```

```
INSERT INTO Empresas VALUES (4,'GALLO', 'polvorón', 'Estepa', '758291402');
```

```
INSERT INTO Empresas VALUES (5,'ANGELA', 'PADILLA', 'MI CAMA', '656421240');
```

```
INSERT INTO Tarifas VALUES (1,'Descuento Teórico', '21,21', TO_DATE('06/11/1999', 'dd/mm/yyyy'),TO_DATE('02/07/2040', 'dd/mm/yyyy'));
```

```
INSERT INTO Tarifas VALUES (2,'Descuento Teórico+Práctico', '50', TO_DATE('22/06/2015', 'dd/mm/yyyy'),TO_DATE('09/07/2016', 'dd/mm/yyyy'));
```

```
INSERT INTO Tarifas VALUES (3,'Descuento por acompañante', '10,1', TO_DATE('31/01/2006', 'dd/mm/yyyy'),TO_DATE('02/07/2010', 'dd/mm/yyyy'));
```

```
INSERT INTO Tarifas VALUES (4,'Descuento folletos', '45,6', TO_DATE('15/08/2009', 'dd/mm/yyyy'),TO_DATE('14/11/2011', 'dd/mm/yyyy'));
```

```

INSERT INTO Tarifas VALUES (5,'Descuento por guapo', '32,99', TO_DATE('11/04/2015',
'dd/mm/yyyy'),TO_DATE('13/10/2016', 'dd/mm/yyyy'));

INSERT INTO Examinadores VALUES (1,'Carmen', 'asdg1237','la bella del salón');
INSERT INTO Examinadores VALUES (2,'Carlos', 'zxcyugt12', 'puntúa de manera rara');
INSERT INTO Examinadores VALUES (3,'Josema', 'uytuyta74', 'le gustan las cabras');
INSERT INTO Examinadores VALUES (4,'Chema', 'asd7yihg', 'es muy estricto');
INSERT INTO Examinadores VALUES (5,'Pablo', 'asuidhu887', 'buen examinador');

INSERT INTO Examenes VALUES (1,TO_DATE('06/11/1998', 'dd/mm/yyyy'), '20', '12',1);
INSERT INTO Examenes VALUES (2,TO_DATE('12/08/2014', 'dd/mm/yyyy'), '32', '20',2);
INSERT INTO Examenes VALUES (3,TO_DATE('06/03/2018', 'dd/mm/yyyy'), '26', '15',3);
INSERT INTO Examenes VALUES (4,TO_DATE('23/11/2008', 'dd/mm/yyyy'), '29', '19',4);
INSERT INTO Examenes VALUES (5,TO_DATE('28/05/2016', 'dd/mm/yyyy'), '16', '15',5);

INSERT INTO Profesores VALUES ('JuanCarlos', 'Cáliz Maroto', '24356712K', '619642737', TO_DATE('27/04/1988',
'dd/mm/yyyy'), 'juanitoca@gmail.com', '876', '1200,00', '300,00', 'S', '8', '4', 'Prácticas', '4157HJL');
INSERT INTO Profesores VALUES ('Paula', 'Molina Alonso', '71824584T', '612408142', TO_DATE('01/04/1956',
'dd/mm/yyyy'), 'paulina@gmail.com', '95', '1000,00', '250,00', 'N', '13', '3', 'Teóricas', '7237HJL');
INSERT INTO Profesores VALUES ('Juanjo', 'Ferrero Rocher', '23498107Q', '691256623', TO_DATE('16/07/1975',
'dd/mm/yyyy'), 'Juanito@hotmail.com', '147', '1200,00', '300,00', 'S', '6', '5', 'Prácticas', '5177DGR');
INSERT INTO Profesores VALUES ('Aitana', 'Barba Roque', '345812940', '712818761', TO_DATE('22/11/1981',
'dd/mm/yyyy'), 'aitañoa@gmail.com', '410', '1000,00', '250', 'S', '10', '1', 'Teóricas', '1127GJM');
INSERT INTO Profesores VALUES ('David', 'Lozano De La Torre', '52734812L', '612398781', TO_DATE('12/10/1991',
'dd/mm/yyyy'), 'DavidPetit@hotmail.com', '10', '1000,00', '300,00', 'S', '0', '0', 'Teóricas', '4128FJV');

INSERT INTO Alumnos VALUES ('Sonia', 'Padilla Granadal', '12349861F', '683742184', TO_DATE('06/11/1993',
'dd/mm/yyyy'), 'tumorenita@hotmail.com', '26384762', 'AM', 'Teórico', '0',1, '71824584T');
INSERT INTO Alumnos VALUES ('Jorge', 'Martínez Méndez', '78324681H', '612349790', TO_DATE('14/08/1994',
'dd/mm/yyyy'), 'jorgitoguapete@gmail.com', '87234691', 'B', 'Práctico', '5',2, '24356712K');
INSERT INTO Alumnos VALUES ('Antonia', 'Fernández Mari', '23857697J', '612409711', TO_DATE('06/12/1988',
'dd/mm/yyyy'), 'Antonia@gmail.com', '23467587', 'A1', 'Finalizado', '13',3, '23498107Q');
INSERT INTO Alumnos VALUES ('Juanjo', 'Alonso Huertas', '51325781V', '612080031', TO_DATE('21/02/2000',
'dd/mm/yyyy'), 'juanito@gmail.com', '12437869', 'B', 'Finalizado', '17',4, '23498107Q');
INSERT INTO Alumnos VALUES ('María', 'Merchan López', '72935691S', '661298712', TO_DATE('24/05/1976', 'dd/mm/yyyy'),
'marieta@hotmail.com', '698237654', 'B', 'Baja', '8',5, '52734812L');

```

```

INSERT INTO Clientes VALUES ('María', 'Franco CuloAbierto', '93617824U', '712301846', TO_DATE('18/09/1972',
'dd/mm/yyyy'), 'franquito@gmail.com', 'S', '0', '300,00', '10', '71824584T');
INSERT INTO Clientes VALUES ('Dani', 'Chen Hu ', '38723648B', '691496921', TO_DATE('02/10/1988', 'dd/mm/yyyy'),
'dani@hotmail.com', 'S', '3', '80,00', '4', '24356712K');
INSERT INTO Clientes VALUES ('Lourdes', 'Martínez Ruiz', '56896182E', '612837421', TO_DATE('08/03/1981',
'dd/mm/yyyy'), 'lour@gmail.com', 'N', '21', '130,00', '0', '71824584T');
INSERT INTO Clientes VALUES ('Martin', 'Alonso Borrás', '76123924Y', '692391612', TO_DATE('20/11/1950',
'dd/mm/yyyy'), 'Marti@gmail.com', 'S', '0', '40,00', '6', '52734812L');
INSERT INTO Clientes VALUES ('Marina', 'Ming Tian', '12164589F', '612859324', TO_DATE('29/10/1986', 'dd/mm/yyyy'),
'marinita01@gmail.com', 'S', '0', '80,00', '9', '24356712K');

INSERT INTO Contratos VALUES ('Descuento Teórico', '1245741', '150', '21',1,'12349861F');
INSERT INTO Contratos VALUES ('Descuento Teórico+Práctico', '1324785', '150', '21',2,'78324681H');
INSERT INTO Contratos VALUES ('Descuento por guapo', '2147184', '150', '21',3,'23857697J');
INSERT INTO Contratos VALUES ('Descuento por acompañante', '0124742', '150', '21',4,'51325781V');
INSERT INTO Contratos VALUES ('Descuento folletos', '9875423', '150', '21',5,'72935691S');

INSERT INTO Bonos VALUES ('S', '742398798', '31,42',1, '78324681H');
INSERT INTO Bonos VALUES ('S', '5452', '45,4',2, '78324681H');
INSERT INTO Bonos VALUES ('N', '78798', '99,99',3, '23857697J');
INSERT INTO Bonos VALUES ('S', '09812312', '32,5',4, '23857697J');
INSERT INTO Bonos VALUES ('N', '423', '32,2',5, '51325781V');

INSERT INTO Facturas VALUES (1, TO_DATE('09/01/2012', 'dd/mm/yyyy'), '93617824U', '123,00',1);
INSERT INTO Facturas VALUES (2, TO_DATE('27/02/1999', 'dd/mm/yyyy'), '38723648B', '88,00',2);
INSERT INTO Facturas VALUES (3, TO_DATE('11/11/2011', 'dd/mm/yyyy'), '76123924Y', '215,50',3);
INSERT INTO Facturas VALUES (4, TO_DATE('02/10/2016', 'dd/mm/yyyy'), '56896182E', '188,00',4);
INSERT INTO Facturas VALUES (5, TO_DATE('13/07/2007', 'dd/mm/yyyy'), '12164589F', '93,00',5);

INSERT INTO ProfesorRecibeBono VALUES (1,'24356712K','742398798');
INSERT INTO ProfesorRecibeBono VALUES (2,'71824584T', '5452');
INSERT INTO ProfesorRecibeBono VALUES (3,'23498107Q', '78798');
INSERT INTO ProfesorRecibeBono VALUES (4,'345812940', '09812312');
INSERT INTO ProfesorRecibeBono VALUES (5,'52734812L', '423');

INSERT INTO AlumnoRealizaExamen VALUES (1,'12349861F',1);
INSERT INTO AlumnoRealizaExamen VALUES (2,'78324681H',2);

```

```
INSERT INTO AlumnoRealizaExamen VALUES (3, '72935691S', 3);  
INSERT INTO AlumnoRealizaExamen VALUES (4, '51325781V', 4);  
INSERT INTO AlumnoRealizaExamen VALUES (5, '23857697J', 5);
```

10.2. SECUENCIAS

```
CREATE SEQUENCE SEC_Empresas
INCREMENT BY 1
START WITH 1
MAXVALUE 99999;
CREATE SEQUENCE SEC_Tarifas
INCREMENT BY 1
START WITH 1
MAXVALUE 99999;
CREATE SEQUENCE SEC_Examinadores
INCREMENT BY 1
START WITH 1
MAXVALUE 99999;
CREATE SEQUENCE SEC_Examenes
INCREMENT BY 1
START WITH 1
MAXVALUE 99999;

CREATE SEQUENCE SEC_ProfesorRecibeBono
INCREMENT BY 1
START WITH 1
MAXVALUE 99999;
CREATE SEQUENCE SEC_AlumnoRealizaExamen
INCREMENT BY 1
START WITH 1
MAXVALUE 99999;
```

10.2.1 TRIGGERS ASOCIADOS A SECUENCIAS

```
CREATE OR REPLACE TRIGGER TR_SEC_Empresas
BEFORE INSERT ON Empresas
FOR EACH ROW
DECLARE
valorSec NUMBER := 0;
BEGIN
SELECT SEC_Empresas.NEXTVAL INTO valorSec FROM DUAL;
:NEW.Id_e := valorSec;
END;
/

CREATE OR REPLACE TRIGGER TR_SEC_Tarifas
BEFORE INSERT ON Tarifas
FOR EACH ROW
DECLARE
valorSec NUMBER := 0;
BEGIN
SELECT SEC_Tarifas.NEXTVAL INTO valorSec FROM DUAL;
:NEW.Id_t := valorSec;
END;
/

CREATE OR REPLACE TRIGGER TR_SEC_Examinadores
BEFORE INSERT ON Examinadores
FOR EACH ROW
DECLARE
valorSec NUMBER := 0;
BEGIN
SELECT SEC_Examinadores.NEXTVAL INTO valorSec FROM DUAL;
:NEW.Id_examinador := valorSec;
END;
/

CREATE OR REPLACE TRIGGER TR_SEC_Examenes
BEFORE INSERT ON Examenes
FOR EACH ROW
```

```

DECLARE
valorSec NUMBER := 0;
BEGIN
SELECT SEC_Examenes.NEXTVAL INTO valorSec FROM DUAL;
:NEW.Id_ex := valorSec;
END;
/

CREATE OR REPLACE TRIGGER TR_SEC_ProfesorRecibeBono
BEFORE INSERT ON ProfesorRecibeBono
FOR EACH ROW
DECLARE
valorSec NUMBER := 0;
BEGIN
SELECT SEC_ProfesorRecibeBono.NEXTVAL INTO valorSec FROM DUAL;
:NEW.Id_r := valorSec;
END;
/

CREATE OR REPLACE TRIGGER TR_SEC_AlumnoRealizaExamen
BEFORE INSERT ON AlumnoRealizaExamen
FOR EACH ROW
DECLARE
valorSec NUMBER := 0;
BEGIN
SELECT SEC_AlumnoRealizaExamen.NEXTVAL INTO valorSec FROM DUAL;
:NEW.Id_realiza := valorSec;
END;
/

```

10.2.2 TRIGGERS NO ASOCIADOS A SECUENCIAS

```
--trigger máx 4 alumnos
alter trigger alumnos disable;
CREATE OR REPLACE TRIGGER alumnos
BEFORE INSERT ON Alumnos
FOR EACH ROW
DECLARE numero INTEGER;
BEGIN
SELECT count(*) INTO numero
FROM Alumnos WHERE profesor = :new.profesor;
IF (numero > 4)
THEN raise_application_error
(-20600,:new.Profesor||' un profesor no `puede tener más de 4 alumnos');
END IF;
END;
/

INSERT INTO Vehiculos VALUES ('Audi', '4157XXX', 'S', TO_DATE('02/02/2019', 'dd/mm/yyyy'),'Vehículo en perfectas
condiciones para su uso');
INSERT INTO Profesores VALUES ('Pruebas', 'Cáliz Maroto', '78451236K', '619642737', TO_DATE('27/04/1988',
'dd/mm/yyyy'), 'juanitoca@gmail.com', '876', '1200,00', '9000,00', 'S', '8', '4', 'Prácticas', '4157XXX');
INSERT INTO Alumnos VALUES ('Jose', 'Martínez Méndez', '78324681A', '612349790', TO_DATE('14/08/1994',
'dd/mm/yyyy'), 'jorgitoguapete@gmail.com', '87234691', 'B', 'Práctico', '5', '1', '24356712K');
INSERT INTO Alumnos VALUES ('Jorges', 'Martínez Méndez', '78324681B', '612349790', TO_DATE('14/08/1994',
'dd/mm/yyyy'), 'jorgitoguapete@gmail.com', '87234691', 'B', 'Práctico', '5', '1', '24356712K');
INSERT INTO Alumnos VALUES ('Jorger', 'Martínez Méndez', '78324681C', '612349790', TO_DATE('14/08/1994',
'dd/mm/yyyy'), 'jorgitoguapete@gmail.com', '87234691', 'B', 'Práctico', '5', '1', '24356712K');
INSERT INTO Alumnos VALUES ('Jorget', 'Martínez Méndez', '78324681D', '612349790', TO_DATE('14/08/1994',
'dd/mm/yyyy'), 'jorgitoguapete@gmail.com', '87234691', 'B', 'Práctico', '5', '1', '24356712K');
INSERT INTO Alumnos VALUES ('Jorgey', 'Martínez Méndez', '78324681E', '612349790', TO_DATE('14/08/1994',
'dd/mm/yyyy'), 'jorgitoguapete@gmail.com', '87234691', 'B', 'Práctico', '5', '1', '24356712K');
INSERT INTO Alumnos VALUES ('Jorgeu', 'Martínez Méndez', '78324681F', '612349790', TO_DATE('14/08/1994',
'dd/mm/yyyy'), 'jorgitoguapete@gmail.com', '87234691', 'B', 'Práctico', '5', '1', '24356712K');
```

```

INSERT INTO Alumnos VALUES ('Jorgei', 'Martínez Méndez', '78324681G', '612349790', TO_DATE('14/08/1994',
'dd/mm/yyyy'), 'jorgitoguapete@gmail.com', '87234691', 'B', 'Práctico', '5', '1', '24356712K');
INSERT INTO Alumnos VALUES ('Jorgeo', 'Martínez Méndez', '78324681I', '612349790', TO_DATE('14/08/1994',
'dd/mm/yyyy'), 'jorgitoguapete@gmail.com', '87234691', 'B', 'Práctico', '5', '1', '24356712K');
INSERT INTO Alumnos VALUES ('Jorgep', 'Martínez Méndez', '78324681J', '612349790', TO_DATE('14/08/1994',
'dd/mm/yyyy'), 'jorgitoguapete@gmail.com', '87234691', 'B', 'Práctico', '5', '1', '24356712K');
INSERT INTO Alumnos VALUES ('Jorgec', 'Martínez Méndez', '78324681K', '612349790', TO_DATE('14/08/1994',
'dd/mm/yyyy'), 'jorgitoguapete@gmail.com', '87234691', 'B', 'Práctico', '5', '1', '24356712K');
INSERT INTO Alumnos VALUES ('Jorgev', 'Martínez Méndez', '78324681L', '612349790', TO_DATE('14/08/1994',
'dd/mm/yyyy'), 'jorgitoguapete@gmail.com', '87234691', 'B', 'Práctico', '5', '1', '24356712K');

--Trigger sueldo máximo de todos los profesores 20.000
Alter trigger sumaProf disable;
CREATE OR REPLACE TRIGGER sumaProf
BEFORE
INSERT ON Profesores
FOR EACH ROW
DECLARE
suma NUMBER(12,2);
BEGIN
SELECT SUM(Sueldo) INTO suma FROM Profesores;
suma := suma + :NEW.Sueldo;
IF (suma > 20000) THEN
raise_application_error
(-20600, :NEW.Sueldo || ' La suma de salarios no puede ser superior a 20000');
END IF;
END;
/
INSERT INTO Profesores VALUES ('JuanCarlos2', 'Cáliz Maroto', '25356713K', '619642737', TO_DATE('27/04/1988',
'dd/mm/yyyy'), 'juanitoca@gmail.com', '876', '1200,00', '9000,00', 'S', '8', '4', 'Prácticas', '4157HJL');
INSERT INTO Profesores VALUES ('Paula1', 'Molina Alonso', '71824584R', '612408142', TO_DATE('01/04/1956',
'dd/mm/yyyy'), 'paulina@gmail.com', '95', '1000,00', '9250,00', 'N', '13', '3', 'Teóricas', '7237HJL');
INSERT INTO Profesores VALUES ('Paula2', 'Molina Alonso', '71824584Z', '612408142', TO_DATE('01/04/1956',
'dd/mm/yyyy'), 'paulina@gmail.com', '95', '9000,00', '9250,00', 'N', '13', '3', 'Teóricas', '7237HJL');
INSERT INTO Profesores VALUES ('JuanCarlos3', 'Cáliz Maroto', '35356713K', '619642737', TO_DATE('27/04/1988',
'dd/mm/yyyy'), 'juanitoca@gmail.com', '876', '1200,00', '9000,00', 'S', '8', '4', 'Prácticas', '4157HJL');

```

```

INSERT INTO Profesores VALUES ('JuanCarlosBueno', 'Cáliz Maroto', '25356753K', '619642737', TO_DATE('27/04/1988',
'dd/mm/yyyy'), 'juanitoca@gmail.com', '876', '800,00', '9000,00', 'S', '8', '4', 'Prácticas', '4157HJL');
INSERT INTO Profesores VALUES ('JuanCarlos44', 'Cáliz Maroto', '25556753K', '619642737', TO_DATE('27/04/1988',
'dd/mm/yyyy'), 'juanitoca@gmail.com', '876', '800,00', '9000,00', 'S', '8', '4', 'Prácticas', '4157HJL');
INSERT INTO Profesores VALUES ('JuanCarlosFallo2', 'Cáliz Maroto', '25656753K', '619642737', TO_DATE('27/04/1988',
'dd/mm/yyyy'), 'juanitoca@gmail.com', '876', '800,00', '9000,00', 'S', '8', '4', 'Prácticas', '4157HJL');

--Trigger Profesor de prácticas recibe bono
Drop trigger TR_BONOS_PROFESORES;
CREATE OR REPLACE TRIGGER TR_BONOS_PROFESORES
BEFORE INSERT OR UPDATE ON ProfesorRecibeBono
FOR EACH ROW
DECLARE
    v_DniProfesor CHAR(9) := :NEW.DniProfesor;
    v_Tipo VARCHAR(10);
BEGIN
    SELECT Tipo INTO v_Tipo FROM Profesores WHERE Dni = v_DniProfesor;
    IF v_Tipo <> 'Prácticas' THEN RAISE_APPLICATION_ERROR(-20002, 'El profesor al que le entrega el bono, debe de ser
de prácticas');
    END IF;
END;
/

INSERT INTO Bonos VALUES ('S', '7423986557', '31,42', '45', '78324681H');
INSERT INTO Profesores VALUES ('CarlosBonoProfesor12', 'Cáliz Maroto', '78324681H', '619642737',
TO_DATE('27/04/1988', 'dd/mm/yyyy'), 'juanitoca@gmail.com', '876', '1200,00', '300,00', 'S', '8', '4', 'Teóricas',
'4157HJL');
INSERT INTO ProfesorRecibeBono VALUES ('6', '78324681H', '7423986557');

--Trigger número máximo de vehículos para cada profesor

drop trigger trigger_vehiculo_profesor_1;
create or replace trigger trigger_vehiculo_profesor_1
before insert or update on Profesores
for each row
declare

```

```

v_Matricula char(9) := :new.Matricula_Vehiculo;
cuenta number;
begin
select count(*) Matricula_Vehiculo into cuenta from Profesores where Tipo='Prácticas' and Matricula_Vehiculo =
v_Matricula;
if cuenta > 1 then RAISE_APPLICATION_ERROR(-20002,' Solo puede haber un coche asignado para cada profesor de
prácticas');
end if;
end;
/
INSERT INTO Profesores VALUES ('PruebaCoche', 'Cáliz Maroto', '12356712K', '619642737', TO_DATE('27/04/1988',
'dd/mm/yyyy'), 'juanitoca@gmail.com', '876', '1200,00', '300,00', 'S', '8', '4', 'Prácticas', '4157HJL');
INSERT INTO Profesores VALUES ('MatriculaRep', 'Molina Alonso', '01824584T', '612408142', TO_DATE('01/04/1956',
'dd/mm/yyyy'), 'paulina@gmail.com', '95', '1000,00', '250,00', 'N', '13', '3', 'Prácticas', '4157HJL');

--Trigger alumno a examen
Alter trigger TR_Alumnos_examen disable;
CREATE OR REPLACE TRIGGER TR_Alumnos_examen
BEFORE INSERT OR UPDATE ON AlumnoRealizaExamen
FOR EACH ROW
DECLARE
v_DniAlumno CHAR(9) := :NEW.DniAlumno;
v_Estado VARCHAR(100);
BEGIN
SELECT Estado INTO v_Estado FROM Alumnos WHERE Dni = v_DniAlumno;
IF v_Estado <> 'Práctico' THEN RAISE_APPLICATION_ERROR(-20002, ' El alumno no puede presentarse a examen porque aún
no ha comenzado las clases prácticas');
END IF;
END;
/

INSERT INTO Alumnos VALUES ('Antonio', 'Martínez Méndez', '78324681X', '612349790', TO_DATE('14/08/1994',
'dd/mm/yyyy'), 'jorgitoguapete@gmail.com', '87234691', 'B', 'Teórico', '5', '1', '24356712K');
INSERT INTO AlumnoRealizaExamen VALUES ('6', '78324681X', '5');

```

10.3. CONSULTAS

```
--Tarifa que no se haya seleccionado nunca
INSERT INTO Tarifas VALUES ('6','Descuento por guapos', '32,99', TO_DATE('11/04/2015',
'dd/mm/yyyy'),TO_DATE('13/10/2016', 'dd/mm/yyyy'));
select Artículo from Tarifas where Id_t NOT IN (select Tarifa from Alumnos);

--Tipo de alumno más usual
select Estado, count(*) from Alumnos group by Estado
having count(*) = (select max(mas) from (select Estado, count(*) as mas
from Alumnos group by Estado));
select Estado, count(*) from Alumnos group by Estado
having count(*) >= All (select count(*)
from Alumnos group by Estado);

--Alumnos con clases prácticas más de 15
select Nombre, Apellidos ,Dni from Alumnos where Clases_Practicas >15;

--Profesor que tiene más alumnos asignados

select Profesor, count(*) from Alumnos group by Profesor
having count(*) = (select max(mas) from (select Profesor, count(*) as mas
from Alumnos group by Profesor));
select Profesor, count(*) from Alumnos group by Profesor
having count(*) >= All (select count(*)
from Alumnos group by Profesor);

--Vehiculos con fecha itv pasada
INSERT INTO Vehiculos VALUES ('Audi', '4159HJL', 'S', TO_DATE('02/02/2018', 'dd/mm/yyyy'),'Vehículo en perfectas
condiciones para su uso');
select Matricula, Fecha_ITV from Vehiculos where Fecha_ITV < sysdate;
```

```

--Sueldo total a pagar profesores prácticas
select sum(Sueldo) from Profesores where Tipo = 'Prácticas';

--Sueldo total a pagar profesores teóricos
select sum(Sueldo) from Profesores where Tipo = 'Teóricas';

--Número total de alumnos matriculados en la autoescuela
CREATE OR REPLACE PROCEDURE PR_Alumnos_Matriculados
IS
CURSOR C IS
SELECT DNI, Nombre, Apellidos, Fecha_Nac, Correo FROM Alumnos;
v_Alumnos C%ROWTYPE;
v_TotalAlumnos NUMBER;
BEGIN
SELECT COUNT(*) INTO v_TotalAlumnos FROM Alumnos;
DBMS_OUTPUT.PUT_LINE(v_TotalAlumnos || ' alumnos matriculados en la autoescuela.' || CHR(13) || CHR(13));
OPEN C;
FETCH C INTO v_Alumnos;
DBMS_OUTPUT.PUT_LINE(RPAD('DNI:', 25) || RPAD('Nombre:', 25) || RPAD('Apellidos:', 25) || RPAD('Fecha de
nacimiento:', 25) || RPAD('Email:', 25));
DBMS_OUTPUT.PUT_LINE(LPAD('-', 120, '-'));
WHILE C%FOUND LOOP
DBMS_OUTPUT.PUT_LINE(RPAD(v_Alumnos.Dni, 25) || RPAD(v_Alumnos.Nombre, 25) || RPAD(v_Alumnos.Apellidos, 25) ||
RPAD(v_Alumnos.Fecha_Nac, 25) ||
RPAD(v_Alumnos.Correo, 25));
FETCH C INTO v_Alumnos;
END LOOP;
CLOSE C;
END;
/

```

10.4. PRUEBAS

-Prueba 1:

```
create or replace
FUNCTION ASSERT_EQUALS (salida BOOLEAN, salida_esperada BOOLEAN) RETURN VARCHAR2 AS
BEGIN
    IF (salida=salida_esperada) THEN
        RETURN 'EXITO';
    ELSE
        RETURN 'FALLO';
    END IF;
END ASSERT_EQUALS;
/
```

```
create or replace
PACKAGE PRUEBAS_VEHICULOS AS

    PROCEDURE inicializar;
    PROCEDURE insertar
        (nombre_prueba VARCHAR2, v_Marca VARCHAR, v_Matricula CHAR, v_Seguro CHAR, v_Fecha_ITV Date, v_Estado
VARCHAR, salidaEsperada BOOLEAN);
    PROCEDURE actualizar
        (nombre_prueba VARCHAR2, v_Marca VARCHAR, v_Matricula CHAR, v_Seguro CHAR, v_Fecha_ITV Date, v_Estado
VARCHAR, salidaEsperada BOOLEAN);
    PROCEDURE eliminar
        (nombre_prueba VARCHAR2, v_Matricula CHAR, salidaEsperada BOOLEAN);

END PRUEBAS_VEHICULOS;
/
create or replace
PACKAGE BODY PRUEBAS_VEHICULOS AS

    PROCEDURE inicializar AS
```

```

BEGIN
    DELETE FROM ALUMNOREALIZAEXAMEN;
    DELETE FROM PROFESORRECIBEBONO;
    DELETE FROM FACTURAS;
    DELETE FROM BONOS;
    DELETE FROM CONTRATOS;
    DELETE FROM CLIENTES;
    DELETE FROM ALUMNOS;
    DELETE FROM PROFESORES;
    DELETE FROM EXAMENES;
    DELETE FROM EXAMINADORES;
    DELETE FROM TARIFAS;
    DELETE FROM Vehiculos;

END inicializar;

PROCEDURE insertar (nombre_prueba VARCHAR2, v_Marca VARCHAR, v_Matricula CHAR, v_Seguro CHAR, v_Fecha_ITV Date,
v_Estado VARCHAR, salidaEsperada BOOLEAN) AS
    salida BOOLEAN := true;
    vehiculo vehiculos%ROWTYPE;
BEGIN
    INSERT INTO Vehiculos VALUES (v_Marca, v_Matricula, v_Seguro, v_Fecha_ITV, v_Estado);

    SELECT MARCA ,MATRICULA ,SEGURO ,FECHA_ITV ,ESTADO INTO vehiculo FROM Vehiculos WHERE Matricula =
v_Matricula;
    If (vehiculo.Marca<>v_Marca AND vehiculo.Matricula<>v_Matricula AND vehiculo.Seguro<>v_Seguro AND
vehiculo.Fecha_ITV<>v_Fecha_ITV AND vehiculo.Estado<>v_Estado) THEN
        salida := false;
    END IF;
    COMMIT WORK;

    DBMS_OUTPUT.put_line(nombre_prueba || ':' || ASSERT_EQUALS(salida, salidaEsperada));

EXCEPTION

```

```

        WHEN OTHERS THEN
            DBMS_OUTPUT.put_line(nombre_prueba || ':' || ASSERT_EQUALS(false, salidaEsperada));
            ROLLBACK;
    END insertar;

    PROCEDURE actualizar (nombre_prueba VARCHAR2, v_Marca VARCHAR, v_Matricula CHAR, v_Seguro CHAR, v_Fecha_ITV
Date, v_Estado VARCHAR, salidaEsperada BOOLEAN) AS

        salida BOOLEAN := true;
        vehiculo vehiculos%ROWTYPE;
    BEGIN

        UPDATE Vehiculos SET Marca = v_Marca, Matricula= v_Matricula, Seguro = v_Seguro, Fecha_ITV = v_Fecha_ITV,
Estado = v_Estado WHERE Matricula = v_Matricula;

        SELECT MARCA ,MATRICULA ,SEGURO ,FECHA_ITV ,ESTADO INTO vehiculo FROM Vehiculos WHERE Matricula =
v_Matricula;
        If (vehiculo.Marca<>v_Marca AND vehiculo.Matricula<>v_Matricula AND vehiculo.Seguro<>v_Seguro AND
vehiculo.Fecha_ITV<> v_Fecha_ITV AND vehiculo.Estado<>v_Estado) THEN
            salida := false;
        END IF;
        COMMIT WORK;

        DBMS_OUTPUT.put_line(nombre_prueba || ':' || ASSERT_EQUALS(salida, salidaEsperada));

    EXCEPTION
        WHEN OTHERS THEN
            DBMS_OUTPUT.put_line(nombre_prueba || ':' || ASSERT_EQUALS(false, salidaEsperada));
            ROLLBACK;
    END actualizar;

    PROCEDURE eliminar (nombre_prueba VARCHAR2, v_Matricula CHAR, salidaEsperada BOOLEAN) AS

        salida BOOLEAN := true;
        n_vehiculos INTEGER;
    BEGIN

```

```

DELETE FROM Vehiculos WHERE Matricula = v_Matricula;

SELECT COUNT(*) INTO n_vehiculos FROM Vehiculos WHERE Matricula = v_Matricula;
  If (n_vehiculos<>0) THEN
    salida := false;
  END IF;
COMMIT WORK;

DBMS_OUTPUT.put_line(nombre_prueba || ':' || ASSERT_EQUALS(salida, salidaEsperada));

EXCEPTION
WHEN OTHERS THEN
  DBMS_OUTPUT.put_line(nombre_prueba || ':' || ASSERT_EQUALS(false, salidaEsperada));
  ROLLBACK;
END eliminar;
END;
/
SET SERVEROUTPUT ON;
BEGIN
  PRUEBAS_VEHICULOS.INICIALIZAR;
  PRUEBAS_VEHICULOS.INSERTAR('Prueba 1- Inserción vehículo','Audi', '4157CCC', 'S', TO_DATE('02/02/2019',
'dd/mm/yyyy'),'Vehículo en perfectas condiciones para su uso',true);
  PRUEBAS_VEHICULOS.ACTUALIZAR ('Prueba 2 - Actualización estado','Audi', '4157CCC', 'S',
TO_DATE('02/02/2019', 'dd/mm/yyyy'),'Vehículo mal',true);
  PRUEBAS_VEHICULOS.ELIMINAR ('Prueba 3 - Eliminar vehículo', '4157CCC',true);
END;

```

-Prueba 2:

```
create or replace
PACKAGE PRUEBAS_BONOS AS

    PROCEDURE inicializar;
    PROCEDURE insertar
        (nombre_prueba VARCHAR2, v_Sello CHAR, v_N_ref NUMBER, v_Importe NUMBER, v_Duracion NUMBER, v_Alumno CHAR,
salidaEsperada BOOLEAN);
    PROCEDURE actualizar
        (nombre_prueba VARCHAR2, v_Sello CHAR, v_N_ref NUMBER, v_Importe NUMBER, v_Duracion NUMBER, v_Alumno CHAR,
salidaEsperada BOOLEAN);
    PROCEDURE eliminar
        (nombre_prueba VARCHAR2, v_N_ref NUMBER, salidaEsperada BOOLEAN);
END PRUEBAS_BONOS;
/
create or replace
PACKAGE BODY PRUEBAS_BONOS AS

    PROCEDURE inicializar AS
BEGIN
    DELETE FROM ALUMNOREALIZAEXAMEN;
    DELETE FROM PROFESORRECIBEBONO;
    DELETE FROM FACTURAS;
    DELETE FROM BONOS;
    DELETE FROM CONTRATOS;
    DELETE FROM CLIENTES;
    DELETE FROM ALUMNOS;
```

```

DELETE FROM PROFESORES;
DELETE FROM EXAMENES;
DELETE FROM EXAMINADORES;
DELETE FROM TARIFAS;
DELETE FROM Vehiculos;

END inicializar;

PROCEDURE insertar (nombre_prueba VARCHAR2, v_Sello CHAR, v_N_ref NUMBER, v_Importe NUMBER, v_Duracion NUMBER,
v_Alumno CHAR, salidaEsperada BOOLEAN) AS
    salida BOOLEAN := true;
    bono Bonos%ROWTYPE;
BEGIN
    INSERT INTO Bonos VALUES (v_Sello, v_N_ref, v_Importe, v_Duracion, v_Alumno);

    SELECT Sello ,N_ref ,Importe ,Duracion ,Alumno INTO bono FROM Bonos WHERE N_ref = v_N_ref;
    If (bono.Sello<>v_Sello AND bono.N_ref<>v_N_ref AND bono.Importe<>v_Importe AND bono.Duracion<>v_Duracion
AND bono.Alumno<>v_Alumno) THEN
        salida := false;
    END IF;
    COMMIT WORK;

    DBMS_OUTPUT.put_line(nombre_prueba || ':' || ASSERT_EQUALS(salida, salidaEsperada));

EXCEPTION
WHEN OTHERS THEN
    DBMS_OUTPUT.put_line(nombre_prueba || ':' || ASSERT_EQUALS(false, salidaEsperada));
    ROLLBACK;
END insertar;

PROCEDURE actualizar (nombre_prueba VARCHAR2, v_Sello CHAR, v_N_ref NUMBER, v_Importe NUMBER, v_Duracion NUMBER,
v_Alumno CHAR, salidaEsperada BOOLEAN) AS
    salida BOOLEAN := true;
    bono Bonos%ROWTYPE;
BEGIN

```

```

        UPDATE Bonos SET Sello = v_Sello, N_ref= v_N_ref, Importe = v_Importe, Duracion = v_Duracion, Alumno =
v_Alumno WHERE N_ref = v_N_ref;

        SELECT Sello ,
N_ref ,
Importe ,
Duracion ,
Alumno INTO bono FROM Bonos WHERE N_ref = v_N_ref;
        If (bono.Sello<>v_Sello AND bono.N_ref<>v_N_ref AND bono.Importe<>v_Importe AND bono.Duracion<>v_Duracion
AND bono.Alumno<>v_Alumno) THEN
            salida := false;
        END IF;
        COMMIT WORK;

        DBMS_OUTPUT.put_line(nombre_prueba || ':' || ASSERT_EQUALS(salida, salidaEsperada));

    EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.put_line(nombre_prueba || ':' || ASSERT_EQUALS(false, salidaEsperada));
        ROLLBACK;
END actualizar;

PROCEDURE eliminar (nombre_prueba VARCHAR2, v_N_ref NUMBER, salidaEsperada BOOLEAN) AS

    salida BOOLEAN := true;
    n_bonos INTEGER;
BEGIN

    DELETE FROM Bonos WHERE N_ref = v_N_ref;

    SELECT COUNT(*) INTO n_bonos FROM Bonos WHERE N_ref = v_N_ref;
    If (n_bonos<>0) THEN
        salida := false;
    END IF;
    COMMIT WORK;

    DBMS_OUTPUT.put_line(nombre_prueba || ':' || ASSERT_EQUALS(salida, salidaEsperada));

```

```
        EXCEPTION
        WHEN OTHERS THEN
            DBMS_OUTPUT.put_line(nombre_prueba || ':' || ASSERT_EQUALS(false, salidaEsperada));
            ROLLBACK;
    END eliminar;
END;
/
SET SERVEROUTPUT ON;
BEGIN
    PRUEBAS_BONOS.INICIALIZAR;
    PRUEBAS_BONOS.INSERTAR('Prueba 1- Inserción bono','S', '45','55', '41', '78324681H',true);
    PRUEBAS_BONOS.ACTUALIZAR ('Prueba 2 - Actualización Sello','N', '45', '45','41', '78324681H',true);
    PRUEBAS_BONOS.ELIMINAR ('Prueba 3 - Eliminar Bono', '45',true);
END;
```
