

Práctica Grupal

Bufete de Abogados Pearson Hardman

Diseño y Gestión de BBDD

Curso 2020/21



Alberto Aguilar Domingo

Vicente Candela Pérez

Índice

1. Descripción del sistema de información.....	3
2. Diagrama E-R.....	4
3. Diseño lógico.....	5
4. Cambios y respuestas a los comentarios de la Parte 1	7
5. Funciones, procedimientos y disparadores.....	10
6. Otros elementos.....	18
7. Bibliografía.....	22

1. Descripción del sistema de información

El fabuloso y prestigioso bufete de abogados Pearson Hardman es un bufete que cuenta con los mejores abogados del mundo. Tal es esta su fama, que la prestigiosa universidad de Harvard cede a los alumnos más aventajados de la carrera de derecho para que se formen con ellos en el mundo de la abogacía.

Todos los trabajadores del bufete poseen un código único para poder acceder al edificio de oficinas. Además, de ellos también sabemos su nombre, apellidos y fecha de cumpleaños.

Nuestro bufete cuenta con dos tipos de trabajadores: abogados y estudiantes de Harvard.

De los abogados queremos saber el código que reciben del colegio de abogados una vez son matriculados, el cual es único, ya que confirma que está acreditado como abogado.

En nuestro bufete hay una cuantiosa cantidad de grandísimos abogados, pero cabe destacar que las especialidades que más facturan son: penal, civil y mercantil. Nuestros abogados son tan buenos que pueden estar especializados en una o varias ramas del derecho, como por ejemplo, alguna de las anteriormente mencionadas. Cabe mencionar que nuestros abogados de mercantil, tendrán una serie puntos que irán aumentando según vayan ganando casos, de modo que a final de año el abogado que más tenga, recibirá un ascenso. Y también decir que los abogados de penal tendrán un sistema de comisiones en función de los casos ganados que tengan.

Los estudiantes en prácticas estarán asignados a un solo abogado mentor, el cual se encargará de enseñarles. Puede darse el caso de que un abogado puede tener a su cargo varios becarios (estudiantes), sin embargo, no puede haber ningún estudiante sin mentor.

Los estudiantes poseen un código único que les proporciona Harvard, el cual los acredita que son estudiantes de dicha universidad.

Nuestra maravillosa empresa representa a aquellos clientes que demandan de abogados. Estos clientes pueden requerir de los conocimientos de varios de nuestros abogados, ya que algunos casos son extremadamente tediosos. También se puede dar el caso de que nuestros abogados pueden estar estudiando los casos de varios de nuestros clientes.

De nuestros clientes queremos saber su DNI, nombre, apellidos, correo electrónico y su dirección de residencia.

Por cada cliente se puede crear uno o varios expedientes según el delito o delitos del que se le culpe. Dicho expediente está asociado al código del cliente (el cual sirve para identificarlo) y también presentará un código de expediente para saber que expediente es de todos los que se pueden haberle abierto a dicho cliente. También de cada expediente es necesario saber si la falta es leve, grave o muy grave.

Solemos evitar los juicios por lo que nos gusta ir a visitar al demandante. (no hace falta mencionar que para que vayamos a visitar al demandante debe existir primero una relación entre el abogado de nuestro bufete y el demandado, si no, no iríamos a hacer la visita) y así intentar llegar a un acuerdo previo.

Cuando realizamos una visita puede haber varios demandantes, de igual modo que puede ser que hagamos la visita de varios expedientes.

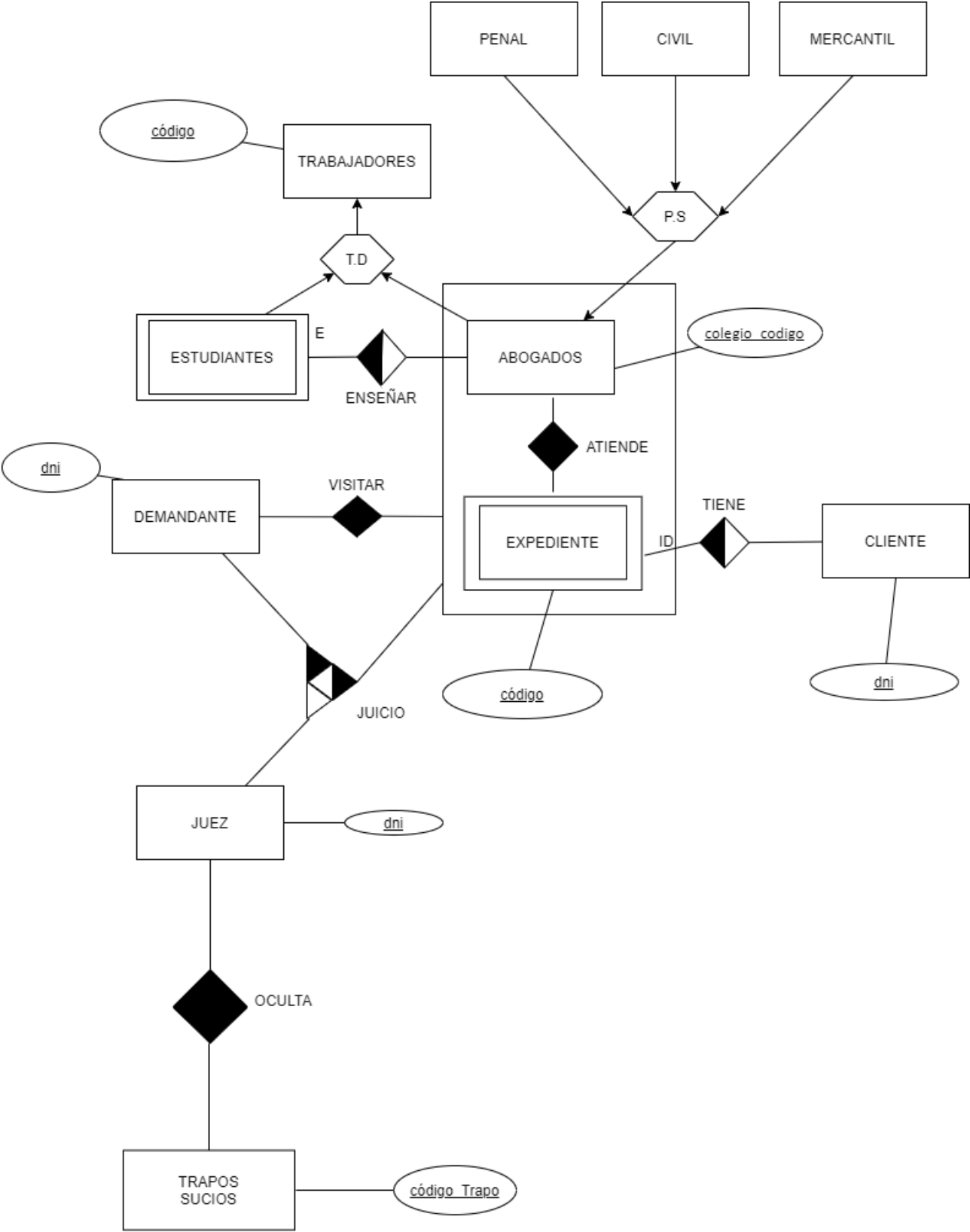
Del demandante deseamos conocer su nombre completo y su DNI.

En el caso de que el demandante no retire su querrela y un juez requiera de un juicio para solventar la demanda, se deben presentar en el juicio tanto el juez, como el/los demandantes, como el abogado defensor con sus clientes.

Por último, del juez sabemos su DNI, código único de juez, nombre, apellidos, nombre de familiares y si tiene o no trapos sucios.

Los trapos sucios se identificarán por un código de trapo sucio, el cual el propio bufete asigna y una descripción detallada del lio en el que se ha metido nuestro juez.

2. Diagrama E-R



3. Diseño lógico

TRABAJADORES (codigo, nombre, apellidos, fecha_ini, fecha_nacimiento)

C.P: código

CLIENTE (dni, nombre, apellidos, correo_elec y direccion)

C.P: DNI

DEMANDANTE (dni_demandante, nombre_demandante,)

C.P: dni_demandante

JUEZ (dni_juez, nombre, apellidos, familiares, codigo)

C.P: DNI

C.Alternativa: código

TRAPOS SUCIOS (codigo_trapo, descripcion)

CP: codigo_trapo

ABOGADOS (trab_codigo, horario, colegio_codigo)

C.P: trab_codigo

C.Alternativa: colegio_codigo

C.Ajena: trab_codigo → TRABAJADORES

ESTUDIANTES (trab_codigo, abo_trab_codigo)

C.P: trab_codigo, abo_trab_codigo

C.Ajena: abo_trab_codigo → ABOGADOS VNN

C.Ajena: trab_codigo → TRABAJADORES

PENAL (comisiones, ab_codigo)

C.P: ab_codigo,

C.Ajena: ab_codigo → ABOGADOS

CIVIL (ab_codigo)

C.P: ab_codigo

C.Ajena: ab_codigo → ABOGADOS

MERCANTIL (puntos, ab_codigo)

C.P: ab_codigo

C.Ajena: ab_codigo → ABOGADOS

EXPEDIENTE (tipo_de_falta, codigo_expediente, clien_dni)

C.P: código_expediente, clien_dni

C.Ajena: clien_dni → CLIENTE

ATIENDE (abo_trab_codigo, clien_dni, cod_expediente)

C.P: abo_trab_codigo, clien_dni, cod_expediente

C.Ajena: abo_trab_codigo → ABOGADO

C.Ajena: clien_dni, cod_expediente → EXPEDIENTE

VISITAR (dni_demandante, abo_trab_codigo, clien_dni, cod_expediente, fecha_visita)

CP: dni_demandante, abo_trab_codigo, clien_dni, cod_expediente, fecha_visita

C.Ajena: abo_trab_codigo, clien_dni, cod_expediente → ATIENDE

C.Ajena: dni_demandante → DEMANDANTE

JUICIO (juez_dni, dni_demandante, abo_trab_codigo, clien_dni, cod_expediente)

C.P: juez_dni, dni_demandante, abo_trab_codigo, clien_dni, cod_expediente

C.Ajena: juez_dni → JUEZ **VNN**

C.Ajena: dni_demandante → DEMANDANTE

C.Ajena: abo_trab_codigo, abo_colegio_codigo, clien_dni, cod_expediente → ATIENDE

OCULTA (codigo_trapo, juez_dni)

CP: codigo_trapo, juez_dni

C.Ajena: juez_dni → JUEZ

C.Ajena: código_trapo → TRAPOS SUCIOS

4. Cambios y respuestas a los comentarios de la Parte 1

Cambios adicionales

No se ha hecho ningún cambio adicional

Comentario 1:

Respecto al formato, en el diagrama los nombres de las relaciones van en mayúsculas y en la modelización en las claves ajenas los nombres de las tablas también en mayúsculas.

Respuesta 1:

Este cambio se ha realizado y se puede observar en el diagrama E-R de la página 4.

Comentario 2:

En la modelización de las especializaciones ABOGADOS y ESTUDIANTES ¿por qué incluís en la clave principal más de un atributo? Se supone que el código del trabajador identifica a cada trabajador de la empresa.

Tened en cuenta que al cambiar la CP de una entidad habrá que revisar la modelización de las tablas relacionadas para adecuarla. Indicad los cambios que hacéis

Respuesta 2:

Hemos considerado que cada abogado deberá de identificarse con su código de empresa y su código de colegiado así, en el caso de que el día de mañana ese abogado por x motivo haya sido despedido y su código de empresa haya podido ser reutilizado para otro abogado, no haya problemas a la hora de identificar que abogado llevaba ese caso.

Comentario 3:

Revisad la definición de las claves principales de las especializaciones PENAL, CIVIL y MERCANTIL.

Respuesta 3:

Hacia un error de atributos, en la lista de atributos que se indicaba para la entidad no estaba el atributo que formaba la clave principal de la misma.

Solucionado.

Comentario 4:

Respecto a la especialización ABOGADO, habéis incluido el mismo atributo horario para cada una de ellas y especificáis que dependiendo del tipo este atributo tiene un valor u otro, pero en realidad el atributo es el mismo. Se puede eliminar la especialización y plantear una relación M:M entre ABOGADO y una nueva entidad TIPOS donde se especifique el código de la especialidad y el número de horas que se trabaja. Valorad ambas opciones y decidid cuál de ellas es más adecuada y por qué.

Respuesta 4:

Ahora horario es un atributo de abogado.

Penal ha adquirido un nuevo atributo que se llama comisiones y ese varia en función de los casos ganados que haga nuestro abogado, de manera que, si nuestro abogado no gana ningún caso, no recibirá ninguna comisión, mientras que si gana casos ganara comisión, a más casos ganados mayor comisión.

Civil ahora no tiene el atributo horario.

Mercantil ha adquirido un nuevo atributo de puntos, de manera que por cada caso mercantil ganado nuestro abogado recibirá una bonificación en puntos, a final de año el abogado mercantil con más puntos en la empresa recibirá un ascenso.

Comentario 5:

En el enunciado redactad más detalladamente el tema de la identificación de EXPEDIENTE porque según se lee se entiende que el expediente se identifica exclusivamente por el código del cliente. Por otro lado, entiendo que en el expediente habrá que indicar algún otro campo como descripción del delito cometido, fecha...

Respuesta 5:

El cambio se ha indicado en la descripción del problema que se encuentra en la página 3.

Comentario 6:

¿Cómo se sabe qué expediente tiene asignado cada abogado? ¿Consideráis adecuado que esta información se sepa? En caso afirmativo, mirad qué cambios podéis hacer para reflejarlo.

Respuesta 6:

Hemos corregido en el diagrama E-R las entidades EXPEDIENTE y CLIENTE de manera que ahora el abogado se relaciona con el expediente a través de la relación M:M llamada ATIENDE.

Sí, consideramos adecuado que se sepa la información para que si en la relación VISITAR el demandante quiera llevar la iniciativa de reunirse con nosotros sepa con que abogado debe hacerlo.

Comentario 7:

Respecto a la visita para acuerdo previo entre DEMANDANTE y CLIENTE, considerad la idea de realizar una relación con EXPEDIENTE para saber sobre cuál de los expedientes se quiere conseguir el acuerdo con el DEMANDANTE. Entiendo que dependiendo del EXPEDIENTE será un DEMANDANTE u otro.

Por otro lado. ¿puede darse el caso que haya un mismo DEMANDANTE sobre varios EXPEDIENTES?

Respuesta 7:

Como hemos comentado ya en la respuesta 6, ahora la entidad EXPEDIENTE se sitúa ya dentro de la agregación. Haciendo así, que el demandante tenga una relación con el expediente.

En relación a la segunda pregunta, previamente a cambiar el diagrama E-R no cabía esta posibilidad en nuestra base de datos, pero ahora hemos rectificado y hemos decidido poner una relación M:M para que sí se contemple la opción de que haya un mismo demandante sobre varios expedientes.

Comentario 8:

Respecto a la relación JUICIO yo lo haría en vez de con CLIENTE con el EXPEDIENTE para saber sobre cuál de los expedientes del CLIENTE se realiza el JUICIO. Valorad la idea y decidid al respecto.

Respuesta 8:

En relación al comentario anterior y dado que el cliente siempre tiene la razón (en este caso usted ahora mismo), hemos optado por cambiar la ubicación de las entidades CLIENTE y EXPEDIENTE para que así tenga un contacto directo con la relación JUICIO.

Comentario 9:

Respecto a los TRAPOS SUCIOS, si los vais a codificar de forma genérica, pondría una relación M:M con JUEZ. Para identificar los tipos genéricos de TRAPOS SUCIOS que tiene cada JUEZ. Valorad la idea y decidid al respecto.

Respuesta 9:

Hemos modificado la relación entre JUEZ y TRAPOS SUCIOS para que se trate de una relación M:M y debido a eso ha sido necesario eliminar la entidad ID debido a la incompatibilidad de una relación M:M con este tipo de entidad.

5. Funciones, procedimientos y disparadores.

A continuación, procedemos a explicar todos los procedimientos, funciones y disparadores que se han desarrollado para la base de datos.

5.1 Función TOTAL_ESTUDIANTES

Como se explica en la descripción del problema, cada abogado tiene a cargo una serie de estudiantes (se podría decir que son los becarios del bufete. La función TOTAL_ESTUDIANTES recibe como parámetro el código identificativo de un abogado (num_abo) y obtiene el número de estudiantes que están a cargo de dicho abogado.

Como podemos observar, en nuestra base de datos existen un total de tres estudiantes que se encuentran supervisados por el abogado cuyo código es X00001.

```
SQL> SELECT * FROM ESTUDIANTES WHERE ABO_TRAB_CODIGO = 'X00001';
```

TRAB_CODIG	ABO_TRAB_C
X00004	X00001
X00006	X00001
X00007	X00001

A continuación, mostramos como la función creada en SQL developer devuelve que son 3 los alumnos que se encuentran supervisados por el abogado indicado anteriormente.

```
SQL> SELECT TOTAL_ESTUDIANTES('X00001') FROM DUAL;
```

TOTAL_ESTUDIANTES('X00001')
3

5.2 Función TOTAL_CASOS

Según la modelización de nuestro problema, un abogado es capaz de atender varios casos al mismo tiempo. Esta función lo que hará, será recibir el código de un abogado y a partir de él obtendrá el número total de casos que en ese momento dicho abogado esté atendiendo.

Como ejemplo, podemos ver en la siguiente captura que el abogado cuyo código es X00002 está atendiendo 3 casos en este momento.

```
SQL> SELECT COUNT(*) FROM ATIENDE WHERE ABO_TRAB_CODIGO='X00002';

COUNT(*)
-----
3
```

Si en SQL plus llamamos a la función, podemos ver que esta también nos devolverá que dicho abogado está llevando 3 casos a la vez.

```
SQL> SELECT TOTAL_CASOS ('X00002') FROM DUAL;

TOTAL_CASOS('X00002')
-----
3
```

5.3 Función NUM_TRA

Esta función es muy sencillita, simplemente nos mostrara por pantalla el número de trapos sucios que tenga el juez que le pedimos.

A continuación, podemos observar una pequeña muestra de lo que hace.

```
SQL> select distinct JUEZ_DNI, NUM_TRA(JUEZ_DNI) from oculta;

JUEZ_DNI  NUM_TRA(JUEZ_DNI)
-----
66666666Y          3
55555555G          2
44444444B          1
```

Esta función nos servirá también para un proceso que usaremos más adelante.

5.4 Función EDAD_REAL

Esta función, ha sido implementada como una función auxiliar que será llamada por otra función más adelante.

Esta función recibe una fecha de nacimiento, y a partir de ella devuelve la edad real de la persona.

Por ejemplo, una persona nacida en 15/4/2000 a día de hoy tiene 20 años de edad. Si pasamos esta fecha, como podemos observar a continuación, la función devolverá que la persona tiene 20 años.

```
SQL> SELECT EDAD_REAL('15/4/2000') FROM DUAL;  
  
EDAD_REAL('15/4/2000')  
-----  
20
```

5.5 Función MEDIA_TRABAJADA

Esta función calcula y devuelve la media de los años trabajados por todos los empleados que están en el bufete.

Como podemos observar, nuestros empleados han trabajado de media 24,25 años en el bufete.

```
Function MEDIA_TRABAJADA compilado
```

```
MEDIA_TRABAJADA  
-----  
24,25
```

5.6 Procedimiento TOTAL_EXPEDIENTES_ABOGADO

Este procedimiento recibirá como parámetro un código de identificación de un abogado y mostrará por pantalla el DNI de los clientes de dicho abogado junto al código del expediente asociado a dicho cliente.

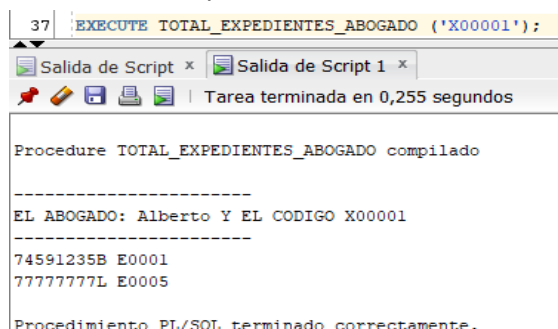
Para este procedimiento, se ha creado un cursor que devolverá los valores mencionados anteriormente siempre que se cumple la condición de que en la tabla ATIENDE el atributo ABO_TRAB_CODIGO coincide con el parámetro que recibe el procedimiento.

Como podemos observar, el abogado cuyo código es X00001 lleva los siguientes expedientes asociados a los DNI.

```
SQL> select CLIEN_DNI, COD_EXPEDIENTE from atiende where ABO_TRAB_CODIGO='X00001';

CLIEN_DNI COD_EXPEDI
-----
74591235B E0001
77777777L E0005
```

Y si ejecutamos la función, podemos observar que devuelve los mismos valores.



37 EXECUTE TOTAL_EXPEDIENTES_ABOGADO ('X00001');

Salida de Script x Salida de Script 1 x

Tarea terminada en 0,255 segundos

Procedure TOTAL_EXPEDIENTES_ABOGADO compilado

EL ABOGADO: Alberto Y EL CODIGO X00001

74591235B E0001
77777777L E0005

Procedimiento PL/SQL terminado correctamente.

5.7 Procedimiento LISTADO_VISITAS

Este procedimiento, lo que hará será devolver un listado con las visitas que un abogado en concreto (cuyo código identificativo será pasado por parámetro) ha realizado a un demandante en concreto (también pasado como parámetro).

También se ha implementado un contador que mostrará el número total de visitas realizadas. El procedimiento mostrará por pantalla el código del abogado, el DNI del demandante y la fecha de la visita junto al expediente del caso por el que se hizo la visita.

Como podemos observar en este ejemplo, en nuestra base de datos, el abogado X00002 ha visitado dos veces al demandante cuyo DNI es 78451296H.

```
SQL> SELECT COD_EXPEDIENTE,FECHA_VISITA FROM VISITAR WHERE DNI_DEMANDANTE='78451296H' AND ABO_TRAB_CODIGO='X00002';

COD_EXPEDI FECHA_VI
-----
E0006      20/12/20
E0006      23/12/20
```

Si ejecutamos el procedimiento anterior con los mismos datos indicados, mostrará los mismos datos.

```

Procedure LISTADO_VISITAS compilado

-----
EL ABOGADO: X00002(Vicente) SE REUNIO CON EL DEMANDANTE: 78451296H
-----
VISITA 1: E0006 23/12/20
VISITA 2: E0006 20/12/20
-----
TOTAL: 2

Procedimiento PL/SQL terminado correctamente.

```

5.8 Procedimiento NOVATO

Este procedimiento, lo que hará será recibir como parámetro el código identificativo de un abogado y a partir de la fecha en la que comenzó a trabajar en el bufete de abogados le asignará un rango.

Este rango podrá ser Novato (si ha trabajado en el bufete menos de 2 años), Avanzado (si ha trabajado más de tres años) o Intermedio (si se encuentra entre los 2 y 3 años).

Este procedimiento implementa un cursor que permitirá mostrar el nombre, apellidos, código y años trabajando en el bufete para cada uno de los trabajadores.

Vamos a hacer el ejemplo con el trabajador cuyo código identificativo es 'X00001'. Como podemos observar en la siguiente captura, dicho trabajador comenzó sus andanzas en la empresa un día 22 de diciembre del año 2018. Esto implica que lleva más de dos años en la empresa y menos de tres, por lo que su rango sería el intermedio.

```

SQL> select * from trabajadores where codigo = 'X00001';

CODIGO      NOMBRE      APELLIDOS      FECHA_IN  FECHA_NA
-----
X00001      Alberto      Aguilar Domingo      22/12/18  21/12/00

```

Para comprobar esto, vamos a ejecutar nuestro procedimiento pasándole como parámetro el valor 'X00001'.

```

NOMBRE APELLIDO CODIGO RANGO
-----
Alberto Aguilar Domingo X00001 2 INTERMEDIO

```

Como podemos observar, efectivamente muestra en el valor del campo rango el valor intermedio.

Vamos a probar con otra persona, en este caso con el trabajador cuyo código es 'X00008' y su fecha de incorporación es el 12/05/2016 y, por lo tanto, lleva más de tres años en la empresa (como podemos ver en la siguiente captura).

```

SQL> select * from trabajadores where codigo = 'X00008';

CODIGO      NOMBRE      APELLIDOS      FECHA_IN  FECHA_NA
-----
X00008      Carlos      Rodriguez Parse      12/05/16  07/04/97

```

Comprobamos mediante una llamada a la función que efectivamente, se le asigna el rango 'Avanzado'.

```
NOMBRE APELLIDO CODIGO RANGO
```

```
-----  
Carlos Rodriguez Parse X00008 4 AVANZADO
```

5.9 Procedimiento LISTADO_CLIEN

Este procedimiento nos resulta bastante útil en el día a día dentro del bufete, de esta manera podemos llevar un historial delictivo de nuestros clientes. De modo que cuando llega un cliente con un caso nuevo, ya sabemos cuál es su pasado y si han sido errores tontos o ha cometido fallos más graves.

A continuación, vamos a mostrar el historial delictivo de uno de nuestros clientes.

```
Procedure LISTADO_CLIEN compilado
```

```
-----  
EL CLIENTE: 11111111A
```

```
-----  
1. Grave E0002
```

```
Total casos: 1
```

Como podemos comprobar este cliente ya tiene un pasado delictivo.

5.10 Procedimiento LISTADO_JUECES

Este procedimiento nos facilita bastante la vida a la hora de ir a juicio porque podemos ver gracias a este procedimiento un poco de información privilegiada para ganar el caso. En este procedimiento si nosotros ponemos el DNI del juez se nos mostrara quien es, su código de juez y si es poco o mucho en lo referido a la corrupción.

Aquí muestro una pequeña muestra de el gran potencial que tiene este procedimiento.

```
SQL> EXECUTE LISTADO_JUECES('55555555G');  
EL JUEZ CON DNI: 55555555G  
-----  
Jesuja Rodriguez 00002J POCO CORRUPTO  
  
PL/SQL procedure successfully completed.
```

5.11 Disparador INSER_ESTUDIANTE

Este disparador se encargará de asegurarse que un trabajador del bufete que ya es abogado, no puede ser añadido en la tabla de estudiante.

Para comprobar su funcionamiento, vamos a mostrar la lista de los abogados del bufete.

```
SQL> select * from abogados;

TRAB_CODIG HORARIO   COLEGIO_CO
-----
X00001      L/M/X      C00001
X00002      L/M/J      C00002
X00003      X/J/V      C00003
```

Y a continuación, vamos a intentar insertar uno de ellos en la tabla de estudiante.

```
SQL> insert into estudiantes values ('X00001','X00002');
insert into estudiantes values ('X00001','X00002')
*
ERROR at line 1:
ORA-20001: NO SE PUDO REALIZAR EL ALTA.
EL EMPLEADO CON CODIGO: X00001YA ES ABOGADO.
ORA-06512: at "PRUEBA.INSER_ESTUDIANTE", line 11
ORA-04088: error during execution of trigger 'PRUEBA.INSER_ESTUDIANTE'
```

5.12 Disparador INSER_ABOGADO

Este disparador se podría definir como el contrario del caso anterior. En el caso de que se intente añadir un estudiante en la tabla de abogado, el disparador mostrará un error impidiendo que esto pase.

Mostramos todos los registros de la tabla estudiantes:

```
SQL> select * from estudiantes;

TRAB_CODIG ABO_TRAB_C
-----
X00004      X00001
X00005      X00002
X00006      X00001
X00007      X00001
```

E intentamos añadir un registro a la tabla de abogados indicando un código perteneciente a la tabla de estudiantes:

```
SQL> INSERT INTO ABOGADOS VALUES ('X00007','L\M\X', 'C00007');
INSERT INTO ABOGADOS VALUES ('X00007','L\M\X', 'C00007')
*
ERROR at line 1:
ORA-20001: NO SE PUDO REALIZAR EL ALTA.
EL EMPLEADO CON CODIGO: X00007YA ES ESTUDIANTE.
ORA-06512: at "PRUEBA.INSER_ABOGADO", line 11
ORA-04088: error during execution of trigger 'PRUEBA.INSER_ABOGADO'
```

Como podemos observar, el disparado impide realizar dicha acción.

5.13 Disparador INSER_JUICIO

Este disparador se encargará de asegurarse que al menos se haya hecho una visita al demandante. Como ya quedó claro en la descripción somos un bufete que destaca sobre todo por ganar juicios sin ir a ellos, es decir, haciendo tratos previos con el demandante, para que ambas partes estén conformes.

Para comprobar su funcionamiento, vamos a mandar a juicio a un expediente que aún no ha realizado ninguna visita.

```
SQL> INSERT INTO JUICIO VALUES('44444444B', '56748912Z', 'X00001', '11111111A', 'E0007');
INSERT INTO JUICIO VALUES('44444444B', '56748912Z', 'X00001', '11111111A', 'E0007')
*
ERROR at line 1:
ORA-20001: NO SE PUEDE IR A JUICIO AUN.
EL EXPEDIENTE CON CODIGO: E0007 NO HA REALIZADO NINGUNA VISITA.
ORA-06512: at "PRUEBA.INSER_JUICIO", line 17
ORA-04088: error during execution of trigger 'PRUEBA.INSER_JUICIO'
```

Como podemos comprobar el disparador impide realizar la acción, y nos muestra un mensaje por pantalla de cuál es el motivo del error.

5.14 Disparador COMPROBAR_FECHA

Este disparador se encargará de impedir que cualquier registro (persona) que se desee incluir en la tabla trabajador y cuya edad sea inferior a 18 años no pueda ser dado de alta dentro de la base de datos.

Para mostrar que funciona, vamos a intentar insertar un registro cuyo valor para el atributo FECHA_NACIMIENTO sea '05/12/2005'. Por lo tanto, su edad actual es de 15 años y no debería dejar insertarlo dentro de nuestra base de datos.

```
SQL> INSERT INTO TRABAJADORES VALUES ('X00009', 'Pablo', 'Rodriguez Pastor', '22/12/2018', '05/12/2005');
INSERT INTO TRABAJADORES VALUES ('X00009', 'Pablo', 'Rodriguez Pastor', '22/12/2018', '05/12/2005')
*
ERROR at line 1:
ORA-20010: NO SE PUEDE REALIZAR EL ALTA.
EL ESTUDIANTE CON EL CODIGO: X00009 NO TIENE
LA EDAD CORRECTA
ORA-06512: at "PRUEBA.COMPROBAR_FECHA", line 3
ORA-04088: error during execution of trigger 'PRUEBA.COMPROBAR_FECHA'
```

Como podemos observar, el disparador impide añadir el registro a la base de datos.

6. Otros Elementos

A continuación, procedemos a la explicación de otros elementos que se han indicado en la base de datos.

6.1 Gestión de usuarios

Para nuestra base de datos, vamos a crear tres tipos de usuarios.

El primer de todos ellos, recibirá el nombre de e3Jefe y tendrá permisos de administrador sobre la base de datos. El segundo recibirá el nombre de e3Abogados y tendrá permisos de conexión, modificación e inserción de datos y creación de vistas dentro de la base de datos. Por último, encontramos el usuario e3Estudiante el cual solo tendrá permisos de conexión.

Esta captura muestra cómo se han creado estos usuarios (los permisos se asignarán en el siguiente apartado).

```
SQL> connect system/bdadmin;
Connected.
SQL> CREATE USER e3Jefe IDENTIFIED BY DIOS;
User created.
SQL> CREATE USER e3Abogados IDENTIFIED BY ABO;
User created.
SQL> CREATE USER e3Estudiante IDENTIFIED BY ESTU;
User created.
```

6.2 Permisos

A continuación, vamos a indicar los permisos que cada usuario tiene sobre la base de datos.

El usuario e3Jefe tendrá permisos de administrador, por lo que se los asignamos.

```
SQL> GRANT DBA TO JEFE;

Grant succeeded.
```

El usuario e3Abogados tiene permisos para conectarse, de recursos y para crear vistas.

```
SQL> GRANT CONNECT, RESOURCE, CREATE VIEW TO ABOGADOS;

Grant succeeded.
```

Y, por último, el usuario e3Estudiante solo tendrá permisos de conexión.

```
SQL> GRANT CONNECT TO ESTUDIANTES;

Grant succeeded.
```

6.3 Vistas

6.3.1 Vista EXPO_GRAV

Esta vista lo que hará será devolver el tipo de falta, el código del expediente y el DNI del cliente de aquellos acusados que hayan cometido faltas catalogadas como “Muy grave”.

Esta captura muestra cuales son los expedientes catalogados como “Muy grave”.

```
TIPO_DE_FA CODIGO_EXP CLIEN_DNI
-----
Muy Grave E0001 74591235B
Muy Grave E0005 77777777L
```

Podemos observar que la vista muestra los mismos valores.

```
SQL> select * from EXPO_GRAV;

TIPO          EXPEDIENTE CLIENTE
-----
Muy Grave E0001 74591235B
Muy Grave E0005 77777777L
```

6.3.2 Vista LIST_CLIEN

Esta vista devuelve el DNI, nombre, apellidos y correo electrónico de todos los clientes de la base de datos.

Esta captura muestra todos los datos de los clientes.

```
SQL> select dni, nombre, apellidos, correo_elec from cliente;

DNI          NOMBRE      APELLIDOS
-----
CORREO_ELEC
-----
74591235B Luis      Diaz
luisdiaz@gmail.com

11111111A Matteo  Fechi
matteofechi@gmail.com

22222222F Valentina Fontalvo
valentinafontalvo@gmail.com

33333333R Fran      Quiles
franquiles@gmail.com

77777777L Aina      Caselles
ainacaselles@gmail.com
```

Y si llamamos a la vista, podremos observar que se obtienen los mismos datos.

```
SQL> select * from LIST_CLIEN;

DNI          NOMBRE      APELLIDOS
-----
CORREO_ELEC
-----
74591235B Luis      Diaz
luisdiaz@gmail.com

11111111A Matteo  Fechi
matteofechi@gmail.com

22222222F Valentina Fontalvo
valentinafontalvo@gmail.com

33333333R Fran      Quiles
franquiles@gmail.com

77777777L Aina      Caselles
ainacaselles@gmail.com
```

6.6.3 Vista TRAJOS

Esta vista devolverá el número de veces que se ha cometido un trajo sucio por parte de un juez.

Primero en SQL mostraremos mediante el comando SELECT cuantas veces se ha cometido el trajo sucio.

```
SQL> select codigo_trajo, count(codigo_trajo) from oculta group by codigo_trajo;
```

CODIGO_TRA	COUNT(CODIGO_TRAJO)
00001T	2
00002T	1
00003T	1
00004T	1
00005T	1

Y si llamamos a la vista, podremos comprobar que se obtienen los mismos resultados.

```
SQL> select * from trajos;
```

CODIGO_TRA	NUM_VECES
00001T	2
00002T	1
00003T	1
00004T	1
00005T	1

6.6.4 Vista TRAB_00

Esta vista se encargará de mostrar todos los trabajadores cuyo año de nacimiento se encuentra comprendido entre 1991 y el 2000.

Esta captura nos muestra cuales son los trabajadores que cumplen la condición del año de nacimiento.

```
SQL> SELECT *
2 FROM TRABAJADORES
3 WHERE TO_NUMBER(TO_CHAR(FECHA_NACIMIENTO, 'YYYY')) BETWEEN 1991 AND 2000
4 ;
```

CODIGO	NOMBRE	APELLIDOS	FECHA_IN	FECHA_NA
X00001	Alberto	Aguilar Domingo	22/12/18	21/12/00
X00002	Vicente	Candela Perez	17/05/18	11/01/00
X00003	Xavi	Dura Mas	18/05/18	15/01/00
X00004	Lazaro	Ortega	08/12/16	25/11/99

Y si llamamos a la vista, obtenemos la misma salida.

```
SQL> SELECT * FROM TRAB_00;
```

CODIGO	NOMBRE	APELLIDOS	FECHA_IN	FECHA_NA
X00001	Alberto	Aguilar Domingo	22/12/18	21/12/00
X00002	Vicente	Candela Perez	17/05/18	11/01/00
X00003	Xavi	Dura Mas	18/05/18	15/01/00
X00004	Lazaro	Ortega	08/12/16	25/11/99

Debido a que se añadió la opción With Check Option, la vista no permitirá insertar ningún trabajador cuyo año de nacimiento no esté comprendido entre estos dos valores.

6.6.5 Vista DATOS_ESTUDIANTES

Esta vista mostrará el código de expediente, el DNI, el nombre y los apellidos de los clientes del bufete. Esta vista se ha creado expresamente para que sea consultada por los estudiantes y puedan obtener información sobre los clientes.

Mostramos los datos que mostraría la vista mediante una instrucción select:

```
SQL> SELECT CODIGO_EXPEDIENTE, C.DNI, C.NOMBRE, C.APELLIDOS
2 FROM CLIENTE C, EXPEDIENTE E
3 WHERE E.CLIENTEN_DNI = C.DNI
4 ORDER BY CODIGO_EXPEDIENTE;

CODIGO_EXP DNI      NOMBRE  APELLIDOS
-----
E0001      74591235B Luis    Diaz
E0002      11111111A Matteo  Fечи
E0003      22222222F Valentina Fontalvo
E0004      33333333R Fran    Quiles
E0005      77777777L Aina    Caselles
E0006      77777777L Aina    Caselles

6 rows selected.
```

Y vemos que, si llamamos a la vista, se muestra la misma información:

```
SQL> select * from DATOS_ESTUDIANTES;

CODIGO_EXP DNI      NOMBRE  APELLIDOS
-----
E0001      74591235B Luis    Diaz
E0002      11111111A Matteo  Fечи
E0003      22222222F Valentina Fontalvo
E0004      33333333R Fran    Quiles
E0005      77777777L Aina    Caselles
E0006      77777777L Aina    Caselles

6 rows selected.
```

6.6.6 Vista TOTAL_COMISIONES

Esta vista mostrará la suma de todas las comisiones que los abogados del tipo Penal han cobrado.

Podemos observar que la tabla Penal, esta rellena con los siguientes datos.

```
SQL> select * from penal;

COMISIONES AB_CODIGO
-----
5000 X00001
4000 X00002
```

Como podemos observar, las comisiones suman 9000. Vamos a llamar a la vista para observar que efectivamente muestra el valor 9000 de comisiones.

```
SQL> select * from total_comisiones;

TOTAL
-----
9000
```

7. Bibliografía

Apuntes

Título: 01_Tema1_DiseñoBDRelacionales

Autor/-es: Yolanda Marhuenda

Sitios de Referencia

Página de Wikipedia de Suits: <https://es.wikipedia.org/wiki/Suits>

Serie completa: <https://www.netflix.com/es/title/70195800>