TFG-Base de datos FIA

Clemente Iturriaga, José Alfredo

Grado en Informática - itinerario Computación

Ferrer Duran, Jordi

15-06-2015

C) Copyright

FICHA DEL TRABAJO FINAL

Título del trabajo:	TFG- Bases de datos FIA
Nombre del autor:	José Alfredo Clemente Iturriaga
Nombre del consultor:	Jordi Ferrer Duran
Fecha de entrega (mm/aaaa):	06/2015
Área del Trabajo Final:	Bases de datos
Titulación:	Grado en Ingeniería Informática- itinerario Computación

Resumen del Trabajo

Este Trabajo se basa en el desarrollo de una base de datos para la FIA Federación Internacional de Automovilismo partiendo de cero.

El proceso ha sido el seguido para la elaboración de una base de datos:

- Recogida de requisitos por parte del cliente (FIA).
- Elaboración del esquema entidad-relación.
- Estudio lógico de la base de datos.
- Diseño físico de la base de datos.
- Creación de los datos para insertar en las tablas de la base de datos.
- Implementación de las restricciones en la base de datos.
- Conjunto de datos para pruebas.
- Elaboración del archivo log.
- Elaboración del data warehouse.
- Entrega de la base de datos junto con la documentación.
- Elaboración de la presentación.

Summary of the Work

This Work is based on the development of a database for the FIA International Federation of Motoring starting from the beginning.

The process has been continued for the making of a database:

- Collection of requisites on the part of the client (FIA).
- Making of the scheme entity - relation.
- Logical study of the database.
- Physical design of the database.
- Creation of the information to insert in the stage of the database.
- Implementation of the restrictions in the database.
- Set of information for tests.
- Making of the file log.
- Making of the data warehouse.
- Delivery of the database together with the papers.
- Making of the presentation.

Palabras clave

FIA, log, bases de datos, relacional, diseño físico, diseño lógico.

INDICE

1. INTRODUCCION

1.1 CONTEXTO Y JUSTIFICACIÓN DEL TRABAJO

Hoy en día, la información es algo muy importante en cualquier área del trabajo, todos generamos información continuamente; la compra que realizamos, el uso del teléfono, el uso de las redes sociales, en el trabajo, etc…

Dada esta gran importancia de los datos, si no tenemos un sistema adecuado que los gestione, sólo tenemos un montón de datos y nada más, es como tener una biblioteca, en que los libros estén en cualquier sitio, sin orden ni identificación, sólo tenemos libros y nada más. No podemos sacar conclusiones de que libros de matemáticas hay o cuantos tenemos repetidos, etc…

El trabajo realizado, es un proyecto de base de datos para la FIA, cuyo objetivo final es proporcionar un sistema organizado, para que puedan extraer información útil de estos datos, con el objetivo final de ser más eficientes y poder desarrollar mejor la organización y poder mejorar continuamente.

El contexto en el que se desarrolla la base de datos es la propia FIA, es decir, la relación entre las categorías de competiciones, carreras, coches, escuderías, pilotos, patrocinadores, etc… Todo lo que tiene que ver con esta organización.

1.2 OBJETIVOS DEL TRABAJO

- Cumplir con los requerimientos del cliente, FIA.
- Elaborar una base de datos relacional normalizada.
- Obtener tablas claras y concisas.
- Eliminar registros duplicados en las tablas.
- Facilitar la labor de consulta.
- Crear una documentación bien elaborada.
- Comprobar que la base de datos está libre de errores.
- Crear un archivo de registro de actividad de la base de datos, reflejando en todo momento la actividad que se ha realizado sobre la misma y la fecha y hora de la misma.
- Elaboración de una presentación en power point sobre la base de datos.

1.3 ENFOQUE Y METODO SEGUIDO

Enfoque:

- Lo primero es leer la documentación inicial proporcionada para saber qué tipo de bases de datos hay que desarrollar.
- Elaborar un listado de requisitos que se han de cumplir.
- Buscar información sobre la FIA y su entorno [1].
- Desarrollar la base de datos de forma cíclica.
- Realizar pruebas de integridad en todo el desarrollo del proceso.

Método seguido:

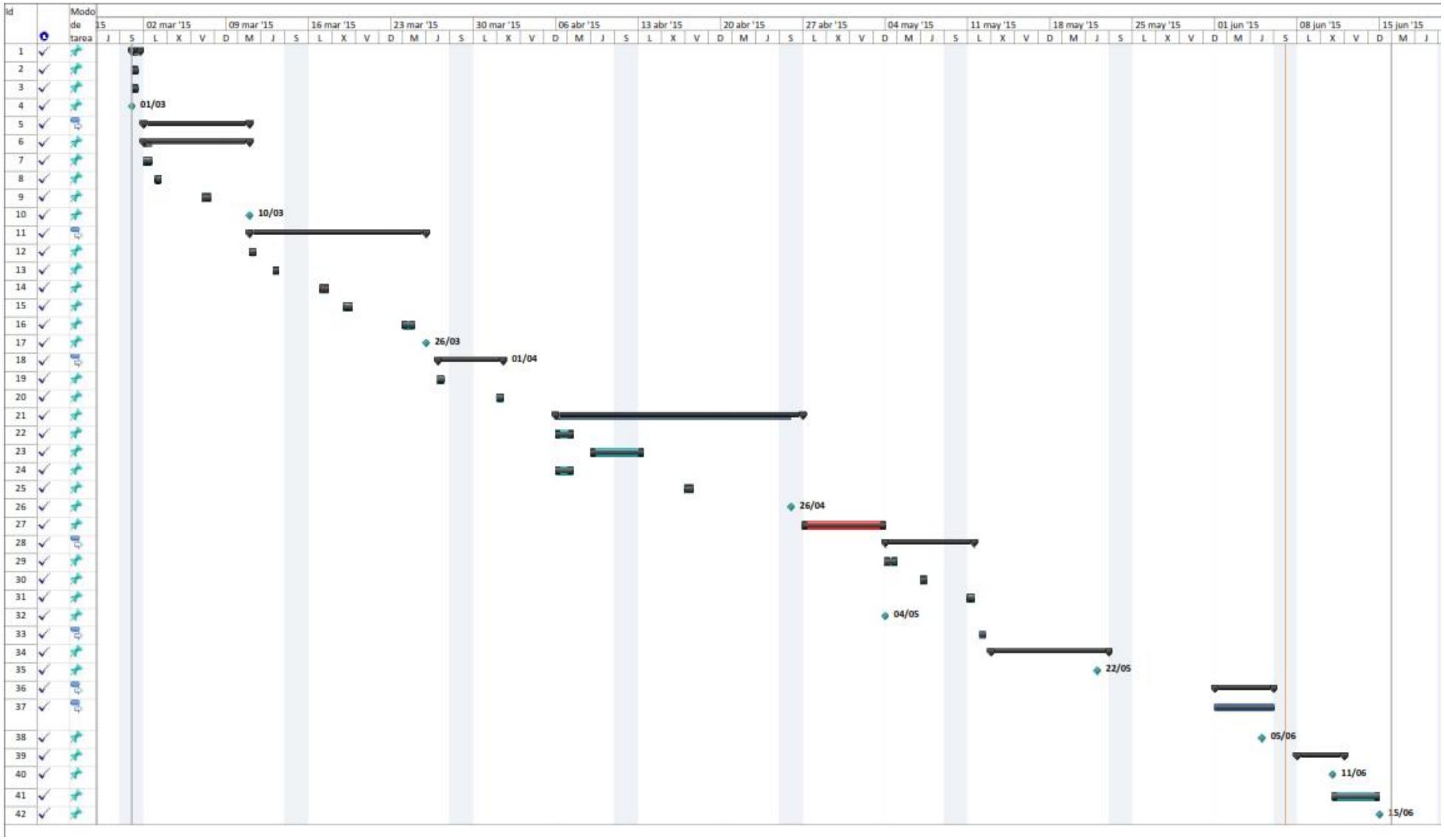
- Se parte de la creación de un producto nuevo, que será la base de datos para la FIA.
- Primero se obtiene la información del documento proporcionado para saber los requisitos que se han de cumplir.
- Buscar información sobre la FIA [1].
- Desarrollar un primer esquema UML y sobre cada entidad seleccionar los atributos mínimos necesarios.
- Buscar información sobre cada entidad, por ejemplo buscar información sobre las carreras de la FIA y ver qué tipo de información se proporciona y cual es relevante [2].
- Desarrollar las relaciones entre las entidades.
- Reestructurar el esquema UML si es necesario.
- Seleccionar los atributos mínimos necesarios para cada entidad presente en el esquema UML y las relaciones.
- Realizar el diseño lógico de la base de datos.
- Comprobar que la base de datos se encuentra normalizada.
- Comprobar si hay que reestructurar el diseño de la base de datos. En caso de que así fuese, se vuelve a comprobar cómo afecta el rediseño a la base de datos y que entidades o atributos hay que volver a normalizar.
- Diseño a nivel físico.
- Se crean las tablas en Oracle. Con las restricciones necesarias (CHECK, PRIMARY KEY , FOREIGN KEY, etc...) y la creación de índices si hiciese falta (CREATE INDEX).
- Se introducen datos en las tablas.
- Se crean los disparadores que controlan la entrada de datos, para mantener la integridad de la base de datos.
- Se comprueban que los disparadores funcionan correctamente.
- Se crea el archivo de registro de actividad en la base de datos.
- Se crean las consultas y se verifica el resultado mostrado.
- Se crea el repositorio estadístico.
- Por último se realiza la documentación del proyecto.

1.4 PLANIFICACION DEL TRABAJO

1.4.1 PLANIFICACIÓN DE TAREAS

		Mo de tar	Nombre de tarea	Duración	Comienzo	Fin	% completad
1	✓		**Inicio del proyecto**	**6 horas**	**dom 01/03/15**	**dom 01/03/15**	**100%**
2	✓		Lectura del documento Enunciat_TFG_BD_20142	3 horas	dom 01/03/15	dom 01/03/15	100%
3	✓		Instalación del software	3 horas	dom 01/03/15	dom 01/03/15	100%
4	✓		Hito inicio de proyecto	0 horas	dom 01/03/15	dom 01/03/15	100%
5	✓		**Desarrollo a nivel conceptual**	**56 horas**	**lun 02/03/15**	**mar 10/03/15**	**100%**
6	✓		Extracción de la información	56 horas	lun 02/03/15	mar 10/03/15	100%
7	✓		Resumen de la información extraida	5 horas	lun 02/03/15	lun 02/03/15	100%
8	✓		Esquema ER inicial	3 horas	mar 03/03/15	mar 03/03/15	100%
9	✓		Esquema UML inicial	5 horas	sáb 07/03/15	sáb 07/03/15	100%
10	✓		Hito recopilación de la información	0 horas	mar 10/03/15	mar 10/03/15	100%
11	✓		**Optimización del esquema ER y UML**	**88 horas**	**mié 11/03/15**	**jue 26/03/15**	**100%**
12	✓		Comprobar si faltan o sobran entidades	3 horas	mié 11/03/15	mié 11/03/15	100%
13	✓		Establecer relación entre ellas	2 horas	vie 13/03/15	vie 13/03/15	100%
14	✓		Vacaciones	2 días	mar 17/03/15	mar 17/03/15	100%
15	✓		Establecer los atributos necesarios para cada entidad	6 horas	jue 19/03/15	jue 19/03/15	100%
16	✓		Optimizar el diagrama UML	8 horas	mar 24/03/15	mar 24/03/15	100%
17	✓		Hito esquema UML	0 horas	jue 26/03/15	jue 26/03/15	100%
18	✓		**Desarrollo a nivel lógico**	**28 horas**	**vie 27/03/15**	**mié 01/04/15**	**100%**
19	✓		Comprobacion de normalizacion de la BD	3 horas	vie 27/03/15	vie 27/03/15	100%
20	✓		Desarrollo de la memoria del trabajo	4 horas	mié 01/04/15	mié 01/04/15	100%
21	✓		**Implementación de la base de datos**	**128 horas**	**lun 06/04/15**	**dom 26/04/15**	**100%**
22	✓		Creación de las tablas	11 horas	lun 06/04/15	mar 07/04/15	100%
23	✓		Creación de datos para las tablas	18 horas	jue 09/04/15	lun 13/04/15	100%
24	✓		Creación de los Triggers	10 horas	lun 06/04/15	mar 07/04/15	100%
25	✓		Comprobación de la integridad de la base de datos	5 horas	vie 17/04/15	vie 17/04/15	100%
26	✓		Hito desarrollo a nivel físico	0 horas	dom 26/04/15	dom 26/04/15	100%
27	✓		Dias no disponibles	48 horas	lun 27/04/15	dom 03/05/15	100%
28	✓		Creación del archivo log	44 horas	lun 04/05/15	lun 11/05/15	100%
29	✓		buscar información para implementar el log	8 horas	lun 04/05/15	lun 04/05/15	100%
30	✓		implementación del log	3 horas	jue 07/05/15	jue 07/05/15	100%
31	✓		corregir errores de ejecución	4 horas	lun 11/05/15	lun 11/05/15	100%
32	✓		Hito archivo log	0 horas	lun 04/05/15	lun 04/05/15	100%
33	✓		Desarrollo de la memoria del trabajo	3 horas	mar 12/05/15	mar 12/05/15	100%
34	✓		Creación de las consultas	64 horas	mié 13/05/15	vie 22/05/15	100%
35	✓		Hito creación de las consultas	0 horas	vie 22/05/15	vie 22/05/15	100%
36	✓		Comprobación de la calidad del producto	40 horas	lun 01/06/15	vie 05/06/15	100%
37	✓		Elaboración de un plan de pruebas para comprobar que la base de datos funciona bien	40 horas	lun 01/06/15	vie 05/06/15	100%
38	✓		Hito comprobación del buen funcionamiento	0 horas	vie 05/06/15	vie 05/06/15	100%
39	✓		**Presentación virtual**	**32 horas**	**lun 08/06/15**	**jue 11/06/15**	**100%**
40	✓		Hito presentación virtual	0 horas	jue 11/06/15	jue 11/06/15	100%
41	✓		Desarrollo de la memoria del trabajo	24 horas	jue 11/06/15	dom 14/06/15	100%
42	✓		Entrega del proyecto	0 horas	lun 15/06/15	lun 15/06/15	100%

1.4.2 DIAGRAMA DE GRANTT



1.4.3 HITOS PARCIALES CONSEGUIDOS

Inicio de proyecto: 01/03/2015

Recopilación de información: 10/03/2015

Esquema UML: 26/03/2015

Desarrollo a nivel físico: 26/04/2015

Archivo log: 04/05/2015

Creación de las consultas: 22/05/2015

Comprobación de buen funcionamiento: 05/06/2015

Presentación virtual: 11/06/2015

Entrega del producto: 15/06/2015

1.4.4 PLAN DE CONTINGENCIA

Riesgos observados y solución planteada.

- Perdida de datos del proyecto.
  Solución. Todos los datos relativos al proyecto son guardados en un medio externo con objeto de asegurar el proyecto ante una eventual pérdida de los datos en el ordenador, debido a un virus, inutilización del disco duro, etc…

- El diseño del proyecto no se ajusta a lo establecido inicialmente.
  Solución. Establecer bien los requisitos iniciales y como afectan estos al resto del proyecto.

- Rehacer parte del proyecto ya realizado.
  Solución. Comprobar que cada tarea realizada se ajusta a lo establecido y no se quedan requisitos iniciales sin establecer ni cumplir. Realizar una retroalimentación en cada fase del proyecto antes de pasar a la siguiente.

- Diseño de tablas con información reiterativa.
  Solución. Establecer primero la normalización de las tablas que compondrán la base de datos.

- Falta de tiempo en las tareas establecidas.
  Solución. El proyecto sólo se desarrolla en días laborables, dejando los fines de semana como tiempo extra en caso de cumplir los objetivos en la semana.

1.4.5 RECURSOS UTILIZADOS

- Magic Draw.
- SQL Developer.
- Microsoft Office.
- Paint.
- Microsoft Project.
- Internet

1.5 SUMARIO DE PRODUCTOS OBTENIDOS

Los productos obtenidos son los siguientes:

- Diagrama UML.
- La base de datos FIA en Oracle.
- Tablas de las bases de datos.
- Disparadores de las tablas.
- Presentación en Power Point.
- Documentación del proyecto.

1.6 COSTE ECONÓMICO

Horas empleadas:

-	Inicio del proyecto	6
-	Desarrollo a nivel conceptual	13
-	Optimización del esquema UML	19
-	Desarrollo nivel lógico	7
-	Implementación base de datos	128
-	Archivo log	45
-	Repositorio	64
-	Calidad del producto	40
-	Presentación virtual	32
-	Desarrollo memoria de trabajo	24

Total: 378 horas

De esas 378 horas muchas de ellas han sido debido a buscar información sobre como implementar alguna cosa en la base de datos, es tiempo de aprendizaje del manejo del programa y de SQL, aproximadamente el 30% del tiempo se ha llevado este trabajo.

Así el tiempo real empleado a margen del aprendizaje es 378 -30% =264 horas si calculamos a 10 euros la hora, tenemos un coste de 2640euros en el proyecto.

El proyecto es viable para una institución como la FIA.

El coste del software es 0, debido a que se han usado programas gratuitos.

2. DESARROLLO DEL PROYECTO

2.1 LECTURA DEL DOCUMENTO

2.1.1 INFORMACIÓN EXTRAIDA DEL DOCUMENTO

- Se ha de implementar un sistema de Bases de datos para la FIA (Federación Internacional de Automovilismo).
- La base de datos ha de gestionar todas las competiciones de la FIA con el objetivo de aumentar el nivel competitivo y profesional que tiene hasta este momento.
- El sistema ha de controlar principalmente tres aspectos fundamentales: registro de todas las entidades que participan en las competiciones (pilotos, coches, circuitos, etc…), registro de los resultados de las carreras y por último el registro de datos relativos al rendimiento de coches y pilotos.
- El trabajo a desarrollar es dar una solución al problema planteado aplicando una estructura de bases de datos como medio para ello.
- Se indican una serie de requisitos mínimos que se han de establecer, por ejemplo: han de existir las entidades carreras, competiciones, coches, etc…
- Se indican una serie de requerimientos que se han de cumplir, por ejemplo: un piloto puede conducir cualquier coche de la escudería durante el año de competición pero no el de otra escudería durante ese año.
- Se ha de definir un repositorio de alta baja y modificación de todas las entidades que se consideran relevantes.
- Definir un repositorio estadístico para realizar consultas.
- La base de datos ha de ser escalable.
- Recogida en un log, toda la actividad en la base de datos.
- Habrá tratamiento de excepciones.

2.2 INSTALACIÓN DEL SOFTWARE

Después de leer el documento y realizado un primer análisis, el software necesario será el siguiente:

- Oracle Database 11g-R express Edition.
  SGDB para la gestión de la base de datos.
  Url de descarga: http://www.oracle.com/technetwork/database/database-technologies/express-edition/downloads/index.html

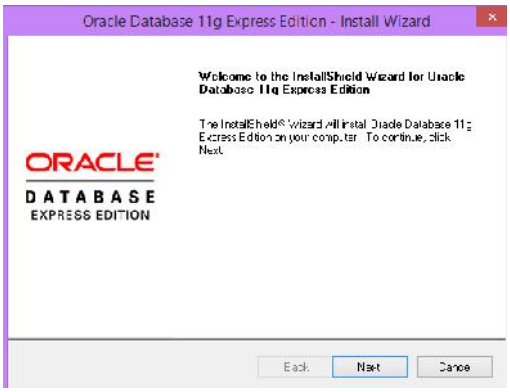
- Magic Draw 18.1 FR versión de prueba.
  Programa para el diseño UML de la base de datos.
  Url de descarga: http://www.nomagic.com/products/magicdraw.html

- Microsoft Office 2010 Home edition. Excel y Word.
  Para la realización de este documento.

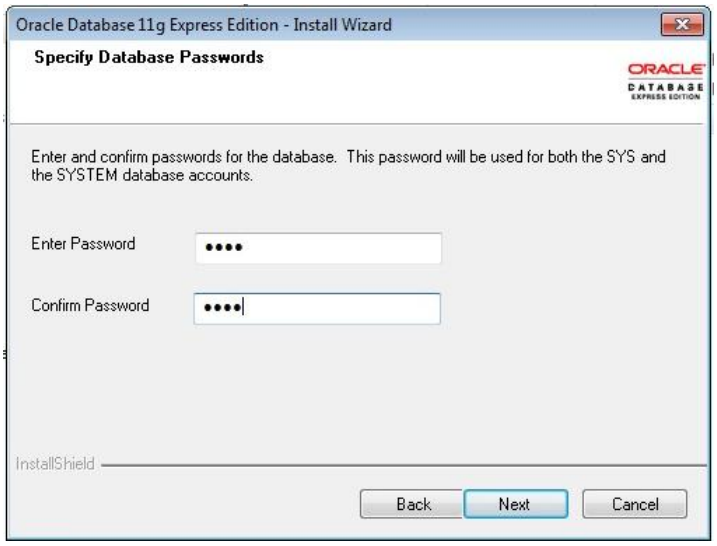
2.2.1 Instalación de Oracle Database 11g-R express EDITION

1. Se descarga el programa de la dirección: http://www.oracle.com/technetwork/database/database-technologies/express-edition/downloads/index.html

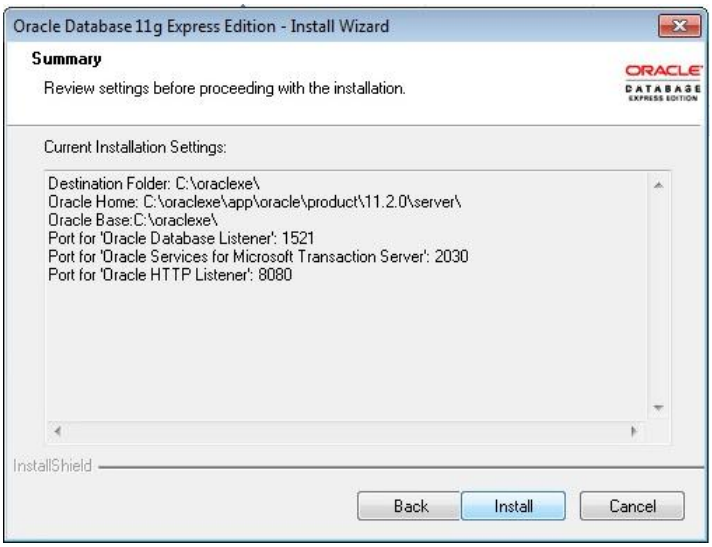
2. Se ejecuta el archivo descargado y se procede a su instalación.



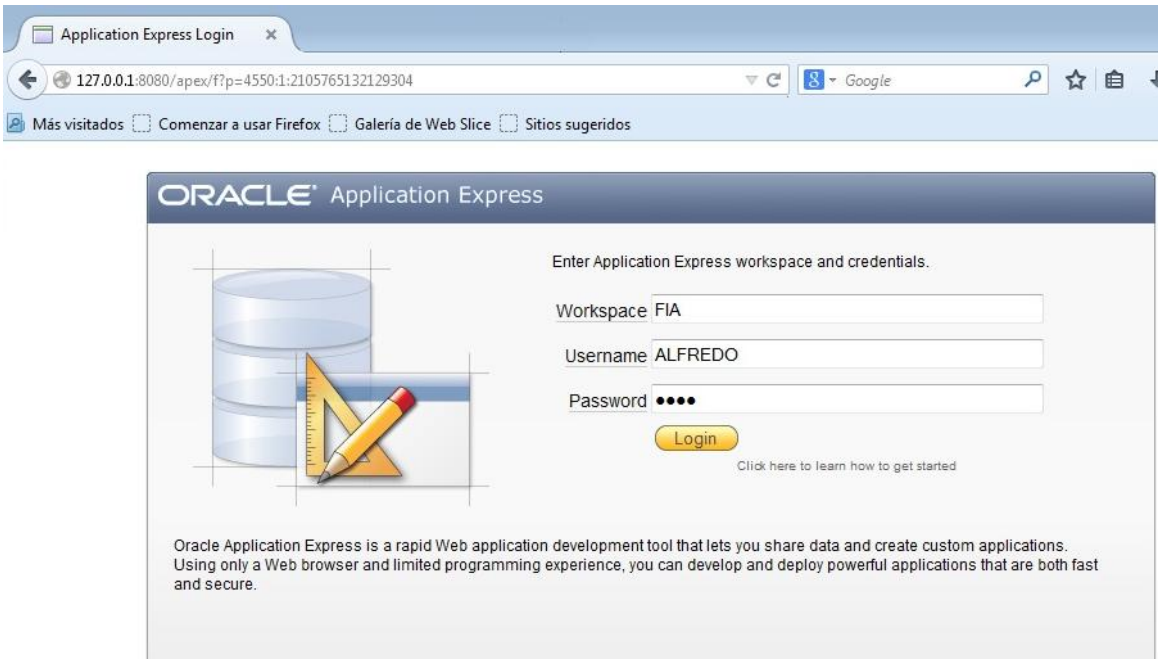
3. En un momento de la instalación el programa nos pide una contraseña para acceder a la base de datos en modo system (administrador).



4. Se dejan las opciones de configuración de los puertos por defecto:

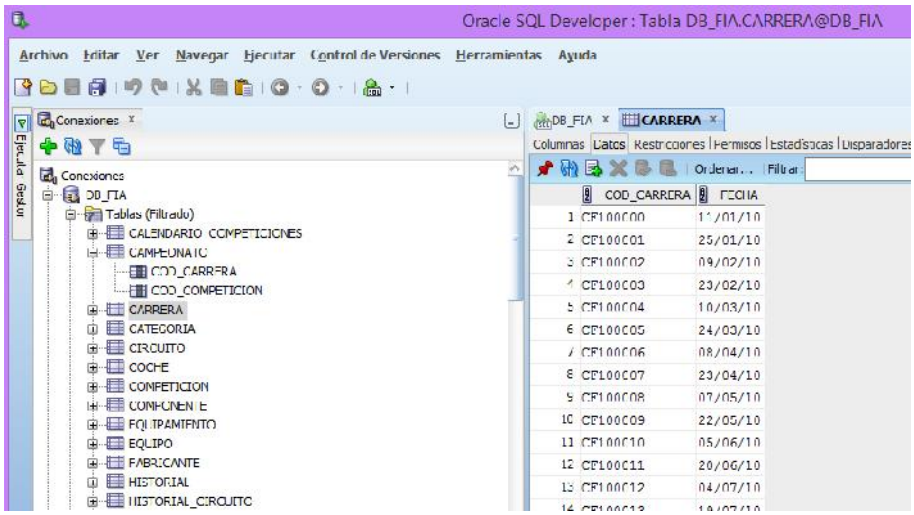


5. Ejecutamos el programa y procedemos a reservar el espacio de trabajo para la base de datos FIA.



El programa funciona correctamente.

También se utiliza el programa SQL Developer de Oracle que nos permite interactuar con Oracle Application Express.

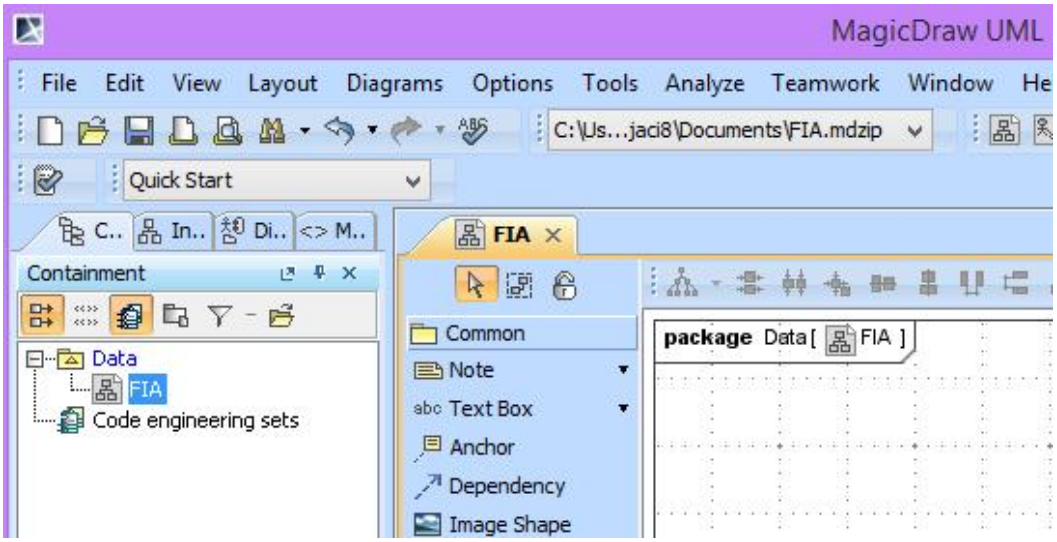


2.2.2 INSTALACIÓN DE MAGIC DRAW 18.1 FR VERSIÓN DE PRUEBA

1. Se descarga el programa desde la url: http://www.nomagic.com/products/magicdraw.html

2. Se instala el programa con las opciones por defecto.

3. Se consigue un serial valido de prueba, el cual es enviado a la dirección de correo electrónico proporcionada al registrarse.
4. Se ejecuta el programa y se crea el diseño UML para la base de datos FIA:



2.3 DESARROLLO DEL MODELO CONCEPTUAL

2.3.1 REQUISITOS:

- Implementar un sistema de bases de datos.
- La base de datos es para controlar todas las competiciones de la Federación Internacional de Automovilismo.
- Se busca seguir la evolución de los pilotos de una forma más analítica.
- Se han de controlar tres aspectos fundamentales de las competiciones:
  - Registro de todas las entidades que participan en las competiciones.
  - Registro de los resultados de las competiciones.
  - Registrar los datos de rendimiento de los pilotos y de los coches, proporcionados por la telemetría de los coches.

2.3.2 RESTRICCIONES:

- Un piloto puede conducir cualquier coche durante la competición.
- Un piloto no puede cambiar de equipo dentro del mismo año de competición.
- Cada equipo sólo puede tener 2 pilotos como máximo en una competición.

2.3.3 INFORMACIÓN INICIAL APORTADA:

- Entidades y atributos:
  - Pilotos: nombre, nacionalidad, código_licencia,…
  - Equipos: código_equipo, nombre, sede, fecha de debut en competición…
  - Componentes: código_componente, descripción, utilidad…
  - Coches: código_coche, modelo, fecha primera carrera…
  - Patrocinadores: código_patrocinador, nombre, sector…
  - Carrera: código_carrera, fecha, longitud circuito…
  - Competiciones: código_competicion, descripción, fecha inicio, fecha fin
  - Resultados: código_resultado, código_coche, posición, puntos obtenidos…
  - Datos telemétricos: código_dato, momento (fecha hora con milisegundos), ámbito, valor, unidad de medida…

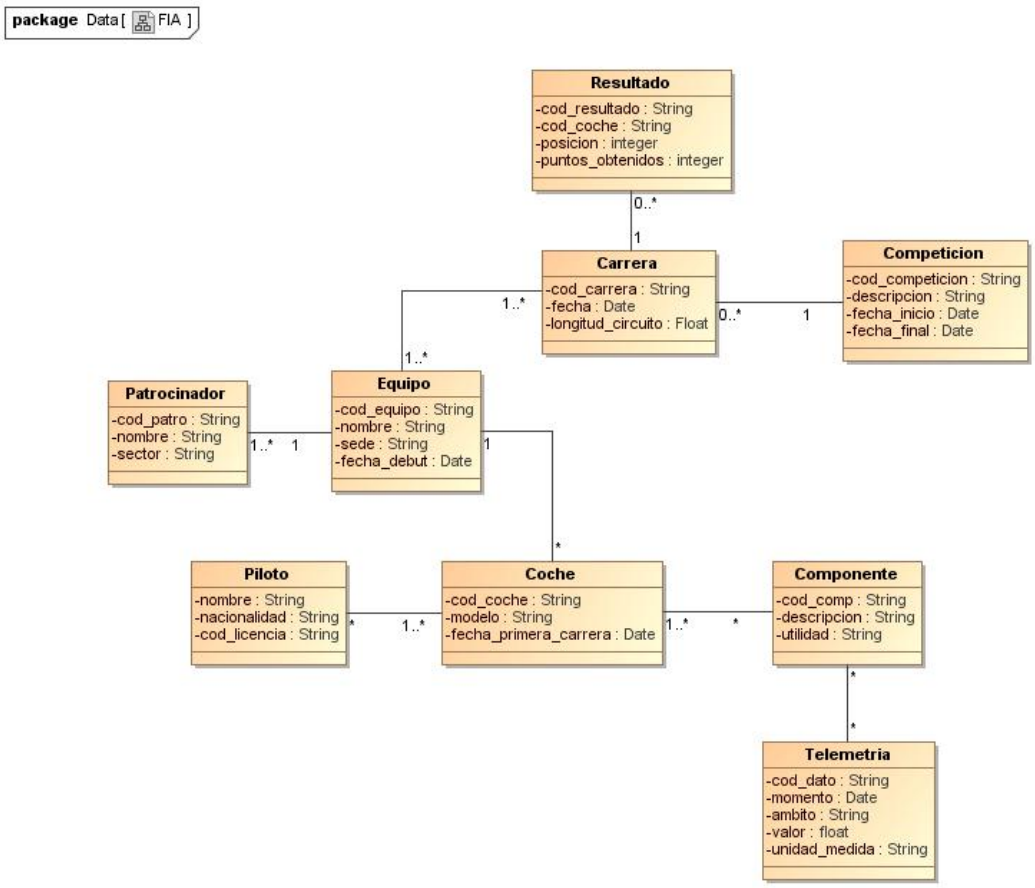
2.3.4 CONSULTAS EN EL REPOSITORIO ESTADISTICO

- Porcentaje de componentes defectuosos.
- Resultado de cada piloto en cada competición disputada.
- Listado de los 10 pilotos que han terminado más carreras de todos los datos registrados.
- Clasificación de cada competición –suma de puntos de los pilotos-.
- Dado un circuito, las 10 vueltas más rápidas que se han dado en él –se ha de indicar el tiempo dela vuelta, el piloto, la fecha, y el equipo.
- Temperatura más alta registrada en el coche, indicar la fecha, el circuito, el coche, el pilot y el valor.
- Dado un año, top 5 de los patrocinadores que aportan más dinero a los equipos de cualquier categoría.
- Volumen de datos telemétricos guardados, indicar cuál es el equipo que ha recogido más datos telemétricos y el número de registros dado en un año concreto.
- Lista de los 5 fabricantes que aportan más componentes a la competición.

- Top 10 de los pilotos que han ganado más carreras entre todos los datos guardados.
- Datos del piloto que ha tenido una mejor evolución entre dos momentos de tiempo indicados.
- Coche que ha consumido más combustible en una competición y año concreto, indicar los datos del coche y los litros de combustible consumidos.

2.4 DIAGRAMA UML

2.4.1 DIAGRAMA UML INICIAL



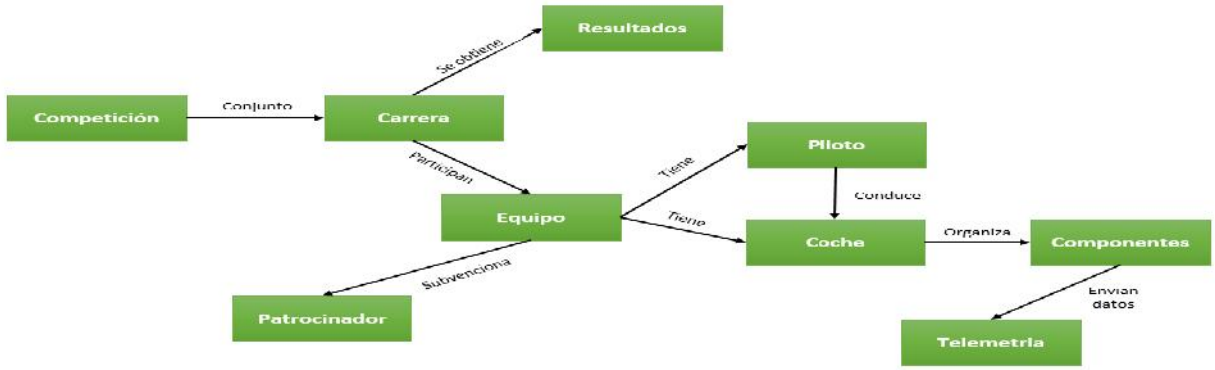
2.4.2 OPTIMIZACIÓN DEL ESQUEMA UML

2.4.2.1 ESTUDIO DE LAS ENTIDADES

Primero hay que estudiar cómo se relacionan las entidades iniciales.

Organizando una relación de las entidades iniciales obtenemos el siguiente resultado de relación en lenguaje natural:

- Una competición es un conjunto de carreras, de las cuales se obtienen unos resultados, en estas carreras participan equipos que están subvencionados por un patrocinador, el equipo se compone de pilotos que conducen coches, estos coches se organizan en componentes que envían unos datos de telemetría al equipo.

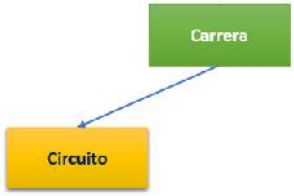


Una vez que ya se tiene una idea de la que partimos se desarrolla a su alrededor el resto de entidades que se necesitan o que hay que reorganizar.

Análisis de las entidades:

- Partimos de la entidad Competición.
  - La FIA es la organización que crea las competiciones, al tener la FIA distintas categorías [5] (Formula 1, Formula 2, Formula E, Rallye, Karting, etc…). Hace falta incluir la entidad categoría para especificar a qué tipo de categoría pertenece la competición a la que nos referimos.
  - Cada categoría de la FIA tendrá una serie de competiciones propias, por norma general una cada año.

- La entidad Carrera.
  - Cada carrera tiene lugar en un circuito [16] [17] [10] [11] (en el caso de fórmula 1, 2, etc…, suele ser en un circuito de automovilismo, urbano o mixto con un trazado definido y que no cambia normalmente a lo largo de las competiciones) en el caso de rallies las carreras se disputan en circuitos que pueden cambiar el trazado a lo largo de las temporadas.
  - Por este motivo se necesita definir la clase circuito, para tener una idea más clara del lugar donde se ha desarrollado la carrera.

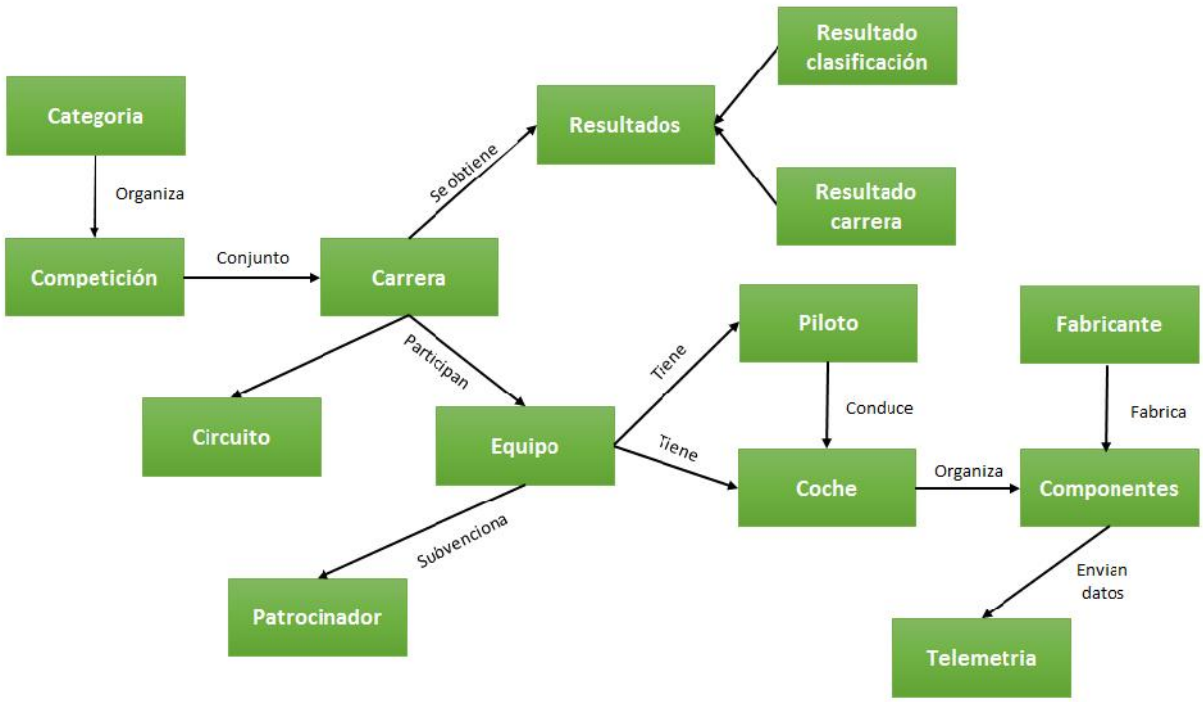


- La entidad Resultado.
  - En una carrera se obtienen dos tipos de resultados, el primero es para definir la parrilla de salida y el segundo el de la carrera en sí misma [3].
  - Así que en la entidad resultado hay una composición con dos entidades más, la entidad resultado clasificación y resultado carrera.
  - Es una composición porque el resultado siempre se va a originar a partir de un resultado de clasificación que conforma la parrilla de salida y un resultado de la carrera.

- La entidad Componente.
  - Los componentes de los coches los tienen que fabricar alguna empresa, sea la propia escudería (chasis, alerones, etc…) como empresas ajenas a la misma escudería, por ejemplo los neumáticos y el combustible son comprados, así como los frenos, etc… [18].

- Las entidades equipo [13] [15], patrocinador, piloto, coche y telemetría no hay que reestructurar.

Ahora ya se tiene la estructuración final para poder rediseñar el esquema UML:



El razonamiento del esquema en lenguaje natural sería el siguiente:

La FIA es una organización que tiene distintas categorías de automovilismo para las cuales organiza competiciones que consisten en un conjunto de carreras de las que se obtienen unos resultados, tanto de clasificación como de competición, estas carreras tienen lugar en circuitos, que pueden ser de distinto tipo dependiendo de la competición, y participan unos equipos con unos pilotos que conducen coches estructurados por distintos componentes -realizados por unos fabricantes- de los que se recogen unos datos telemétricos, todo esto apoyados económicamente por un patrocinador.

2.4.3 ESTUDIO DEL ATRIBUTO DE LAS ENTIDADES

Para cada entidad, se formula el atributo que hace falta, el dominio, descripción y ejemplo.

Entidad Categoría:

ATRIBUTO	DOMINIO	DESCRIPCIÓN	EJEMPLO
nombre	String	Proporciona el nombre de la categoría.	Formula 1
abreviatura	String	Abreviatura del nombre de la categoría.	F1
tipo_automovil	String	Indica el tipo de automóvil que se usa.	Monoplaza, kart, serie…
descripción	String	Breve descripción de la categoría.	

Entidad Competición:

ATRIBUTO	DOMINIO	DESCRIPCIÓN	EJEMPLO
cod_competicion	String	Identifica una competición de forma univoca.	F1_2016
descripción	String	Breve descripción de la competición.	Campeonato de F1
num_carreras	Integer	Indica el número de carreras que habrá.	15
fecha_inicio	Date	Indica la fecha de inicio de la competición.	24-01-2016
fecha_final	Date	Indica la fecha final de la competición.	14-11-2016

Entidad Carrera:

ATRIBUTO	DOMINIO	DESCRIPCIÓN	EJEMPLO
cod_carrera	String	Identifica una carrera de forma univoca.	F1_2016_24-01-2016
fecha	Date	Fecha de la carrera	24-01-2016

Entidad Circuito:

ATRIBUTO	DOMINIO	DESCRIPCIÓN	EJEMPLO
cod_circuito	String	Identifica un circuito de forma univoca.	F1_2016_24-01-2016
nombre	String	Nombre del circuito	Silverstone
país	String	Nombre del país donde se encuentra.	Gran Bretaña
tipo	String	Tipo de circuito; urbano, mixto, automovilístico.	automovilistico
longitud	Integer	Longitud del circuito en metros.	5891
Inauguracion	Date	Año en que se corre la primera carrera.	01-01-1935

Entidad Resultado:

ATRIBUTO	DOMINIO	DESCRIPCIÓN	EJEMPLO
cod_resultado	String	Identifica el resultado de forma univoca.	F1_2016_24-01-2016
cod_coche	String	Código del coche	R846
cod_piloto	String	Código del piloto	FAL
cod_equipo	String	Código del equipo	REN

Entidad Resultado_clasificación:

ATRIBUTO	DOMINIO	DESCRIPCIÓN	EJEMPLO
pos_clasificación	Integer	Posición obtenida en la clasificación	1
tiempo	Time	Tiempo en minutos:segundos:milesimas	1:54:1234785

Entidad Resultado_carrera:

ATRIBUTO	DOMINIO	DESCRIPCIÓN	EJEMPLO
pos_carrera	Integer	Posición obtenida en la carrera	4
tiempo	Time	Tiempo horas: minutos: segundos: milesimas	2:14:9:2546
vuelta_rapida	Time	Vuelta rápida en carrera min:seg:milesimas	4:3:2145
puntos_obtenidos	Integer	Puntos obtenidos en la carrera.	25
puntos_escuderia	Integer	Puntos obtenidos por el equipo.	20

Entidad Equipo:

ATRIBUTO	DOMINIO	DESCRIPCIÓN	EJEMPLO
cod_equipo	String	Código que identifica al equipo	4
nombre	String	Nombre de la escudería.	Ferrari
sede	String	Sede de la escudería.	Manarelo
pais	String	Pais de la escudería.	Italia
fecha_debut	Date	Fecha de inicio en las competiciones.	21-05-1950

Entidad Patrocinador:

ATRIBUTO	DOMINIO	DESCRIPCIÓN	EJEMPLO
cod_patrocinador	String	Código que identifica al patrocinador	AM12
nombre	String	Nombre del patrocinador	América móvil
sector	String	Sector del patrocinador	Telefonía
pais	String	Pais del patrocinador	EEUU

Entidad Piloto:

ATRIBUTO	DOMINIO	DESCRIPCIÓN	EJEMPLO
nombre	String	Nombre del piloto	Fernando Alonso
nacionalidad	String	Nacionalidad del piloto	Española
cod_licencia	String	Código de la licencia del piloto.	1981_FE_AL_ES_F1
fecha_nacimiento	Date	Fecha de nacimiento del piloto.	29-07-1981
sexo	String	Sexo del piloto (V: varón, M: mujer)	V

Entidad Coche:

ATRIBUTO	DOMINIO	DESCRIPCIÓN	EJEMPLO
cod_coche	String	Código que identifica al coche.	FE655_1
tipo	String	Tipo de coche.	Monoplaza
modelo	String	Modelo del coche	FE655
fecha_fs_carrera	Date	Fecha de entrada en carrera.	15-03-1986
fecha_fin_uso	Date	Fecha en la que se deja usar el coche.	23-05-1987
motivo	String	Motivo por el que el coche se deja de usar.	Cambio motor

Entidad Componente:

ATRIBUTO	DOMINIO	DESCRIPCIÓN	EJEMPLO
cod_componente	String	Código del componente.	FR_100_D
descripción	String	Descripción del componente.	Freno delantero 100
utilidad	String	Para que se usa el componente en el coche.	Sirve para frenar.
ambito	String	Ambito de la medida del dato	Temperatura, presión
unidad medida	String	Unidad en que se mide el dato	Grados, Kgf,etc…
min	Integer	Valor mínimo del dato	0
max	Integer	Valor máximo del dato	900

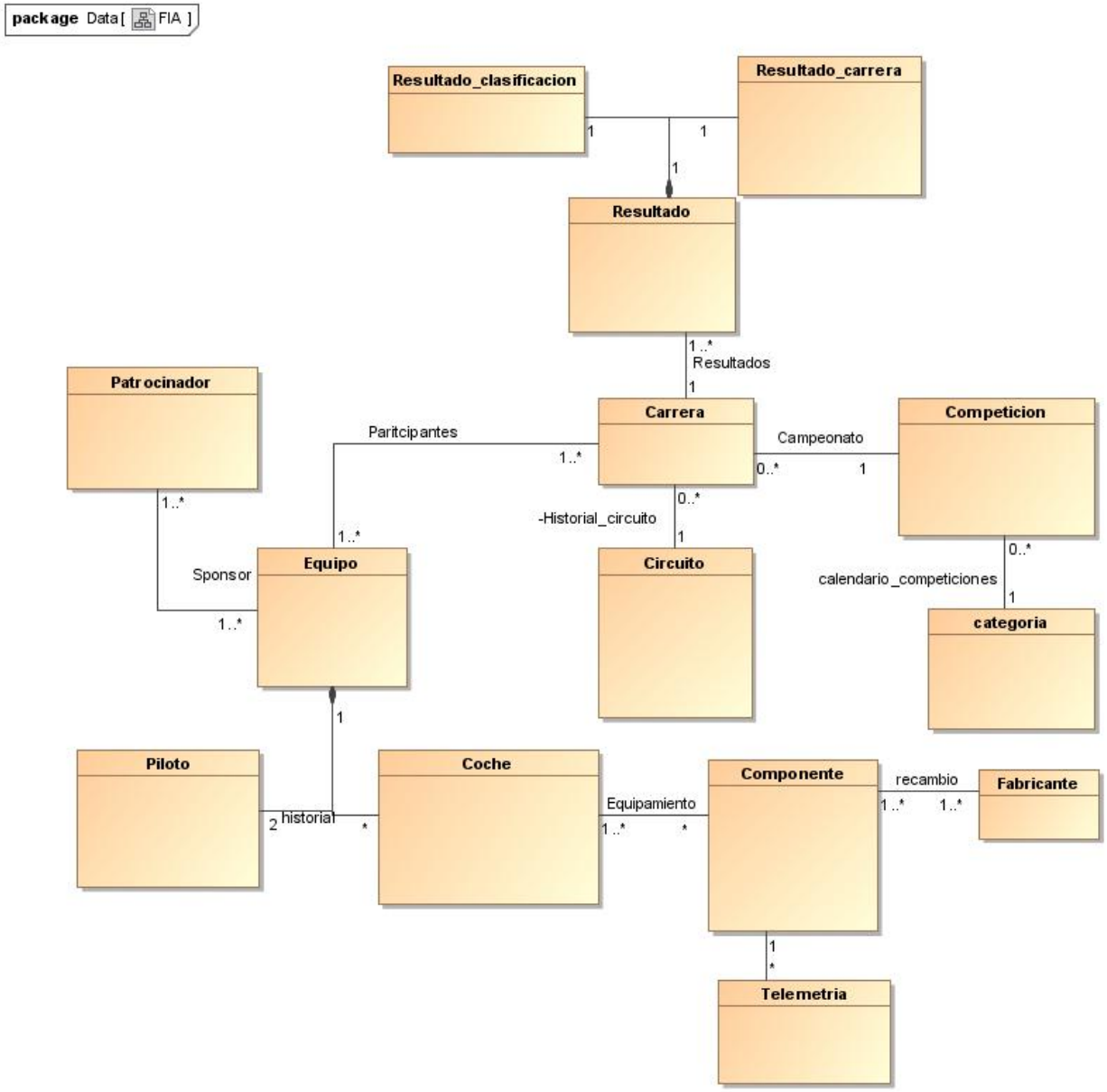
Entidad Telemetría:

ATRIBUTO	DOMINIO	DESCRIPCIÓN	EJEMPLO
Cod_dato (CIC)	String	Código del dato.	T1
Momento	Date/Time	Fecha y hora en milésimas.	15-03-1995 15:24:32:2655
Valor	Float	Valor del dato recibido.	26,0

Entidad Fabricante:

ATRIBUTO	DOMINIO	DESCRIPCIÓN	EJEMPLO
nombre	String	Nombre del fabricante	Japan Electronics
sector	String	Sector de fabricación	Neumáticos

2.4.4 RELACIÓN ENTRE LAS ENTIDADES



Calendario_competiciones

Relaciona la entidad categoría con la entidad competición.

Una categoría puede tener cero competiciones, porque se acaba de crear, o muchas a lo largo de su existencia. Cada competición es exclusiva de una categoría.

Campeonato

Relaciona la entidad competición y carrera.

Una competición puede tener muchas carreras, pero una carrera sólo puede pertenecer a una competición.

Resultado

Relaciona la entidad carrera y la composición de la entidad resultado.

Cada carrera tiene una serie de resultados correspondientes a cada uno de los participantes en la misma, estos resultados son de clasificación y de carrera. Por otro lado cada resultado es único de cada carrera.

Si la carrera se suspende el resultado es NULL, siempre habrá un resultado como mínimo.

Historial_circuito

Relaciona las carreras y los circuitos.

En un circuito se pueden haber celebrado ninguna carrera o muchas y cada carrera se celebra en un solo circuito.

Participantes

Relaciona cada carrera con los participantes en la misma; equipo, coche, piloto y patrocinador.

En cada carrera pueden participar un equipo (en caso de catástrofe) o muchos equipos, y estos equipos pueden haber participado en una carrera (acaba de debutar) o en muchas carreras.

Sponsor

Relaciona al equipo con el patrocinador.

Para cada competición cada equipo tiene al menos un patrocinador, que puede ser la propia escudería, hasta muchos patrocinadores.

Cada patrocinador por regla general sólo patrocina un equipo en concreto, pero puede haber casos en que tenga participación en varios equipos, por lo cual un patrocinador puede estar en uno o varios equipos.

Historial

Relaciona las entidades equipo, piloto y coche, para una determinada competición.

Así una escudería puede tener sólo dos pilotos, que pueden conducir varios coches pertenecientes a la escudería a lo largo de una temporada (competición).

Equipamiento

Relaciona las entidades coche, componente para una determinada carrera.

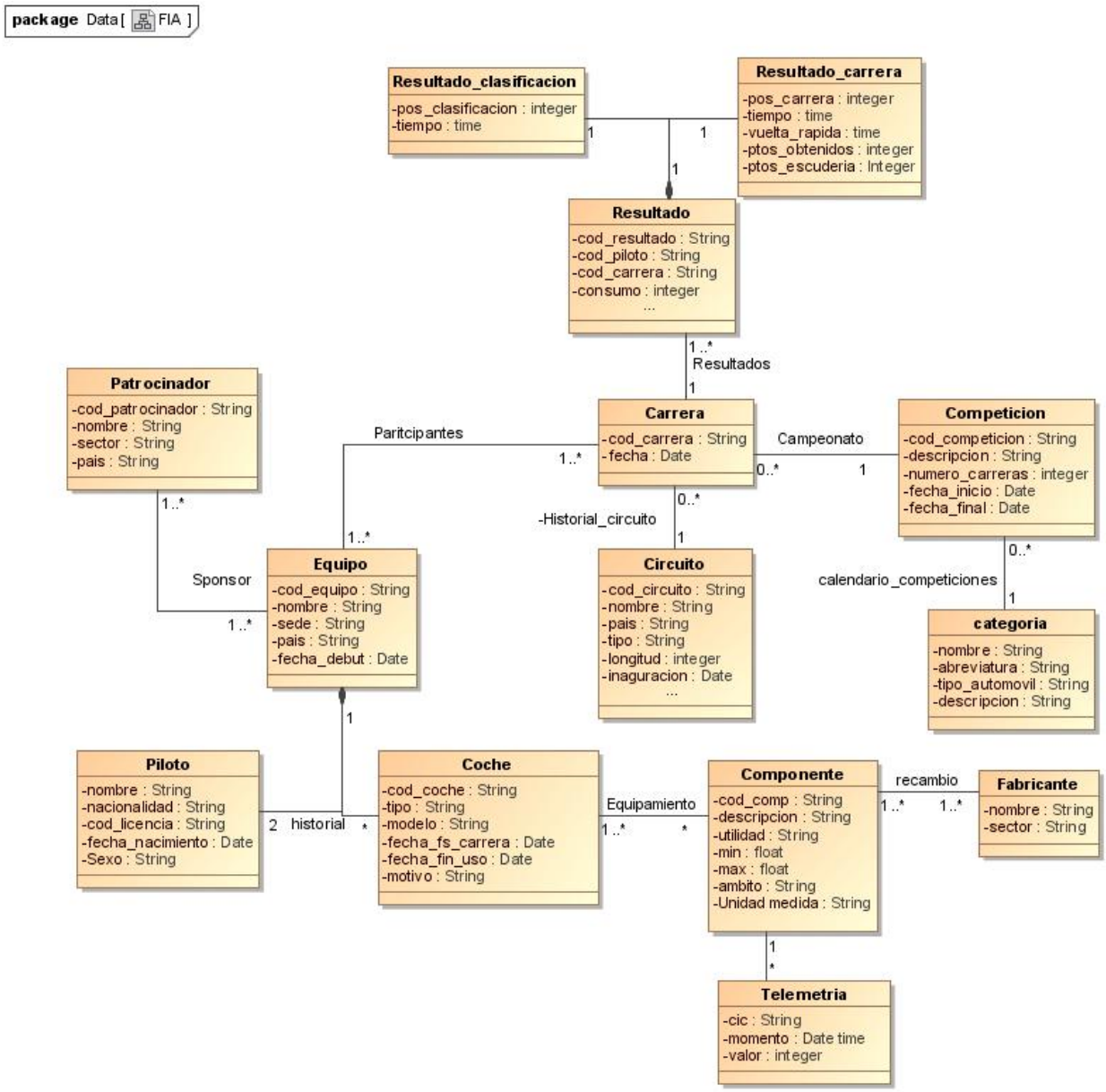
Puede ocurrir que un componente se use en un mismo coche o en otro coche distinto de la escudería para una única carrera o para varías, por eso es importante relacionar la entidad coche y componente con la entidad carrera.

Recambio

Relaciona las entidades componente y fabricante.

Se relaciona los componentes que van en los coches con sus correspondientes fabricantes.

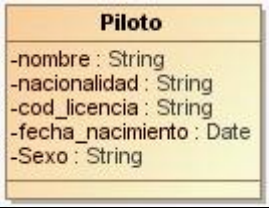
2.4.5 DIAGRAMA UML DEFINITIVO

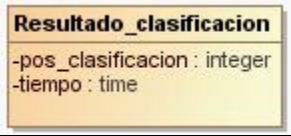
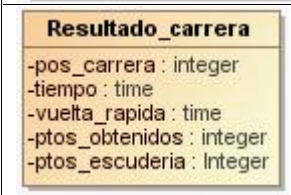


2.5 ESTUDIO LÓGICO

2.5.1 INSTANCIAS

En subrayado continuo las claves primarias, en subrayado discontinuo las claves foráneas y en negrita los valores que no pueden ser nulos.

	Patrocinador (cod_patrocinador, nombre, sector, pais)
	Equipo (cod_equipo, nombre, sede, pais, fecha_debut)
	Piloto (nombre, nacionalidad, cod_licencia, fecha_nacimiento, sexo)
	Coche (cod_coche, tipo, modelo, fecha_st_carrera, fecha_fin_uso, motivo)
	Componente (cod_comp, descripcion, utilidad, ámbito, unidad_medida, min, max)
	Telemetria (cic, momento, valor)
	Circuito (cod_circuito, nombre, pais, tipo, longitud, inaguracion)
	Carrera (cod_carrera, fecha)
	Competicion (cod_competicion, descripción, numero_carreras, fecha_inicio, fecha final)

	Categoria (nombre, abreviatura, tipo_automovil, descripción)
	Resultado_clasificacion (pos_clasificacion, tiempo)
	Resultado_carrera (pos_carrera, tiempo, vuelta_rapida, ptos_obtenidos, ptos_escuderia)
	La entidad resultado es una composición de las entidades resultado_clasficación y resultado_carrera. Resultado (cod_resultado, cod_piloto, cod_carrera, pos_clasificacion, tiempo_clas, pos_carrera, tiempo_carrera, vuelta_rapida, ptos_obtenidos, ptos_escuderia) Como hay competiciones en las que no hay una carrera previa para determinar la línea de salida de la carrera, los atributos pos_clasificación, tiempo_clas pueden ser NULL. Hay carreras como los rallies, que no se dan vueltas a un circuito, por lo tanto el atributo vuelta_rápida puede ser NULL. Hay competiciones en que no se otorgan puntos a la escudería, por lo tanto ptos_escudería puede ser NULL. El atributo tiempo_carrera, puede ser NULL, porque el piloto puede abandonar y por lo tanto no terminar la carrera.
	Componente(cod_comp, descripcion, utilidad, min, max, ámbito, unidad_medida) Los valores mínimo y máximo que arroja un componente son necesarios para saber si el mismo funciona correctamente. Ámbito es el campo de la medición, por ejemplo temperatura, presión, etc… Unidad de medida será grado centígrado, Kgf, etc…, esto depende del ámbito de la medida.
	Fabricante (nombre, sector) Sector indica cual es la especialidad del fabricante; neumáticos, válvulas, detectores de velocidad, etc…

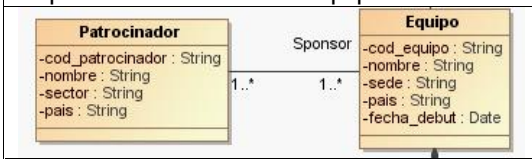
2.5.2 RELACIONES

Sponsor

La relación sponsor, relaciona a los equipos con los patrocinadores, indicando el porcentaje de participación del patrocinador en el equipo.

Un equipo puede tener 1 patrocinador o muchos y un patrocinador puede hacerlo a 1 o varios equipos de la misma categoría de competición de la FIA o de distintas categorías.

El patrocinador lo es del equipo durante todo el periodo de la competición.



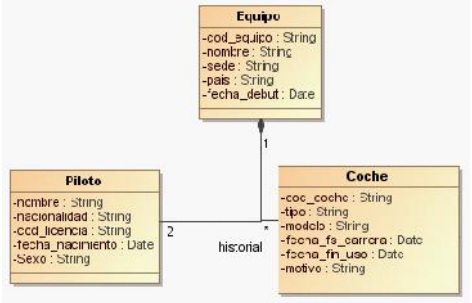
Sponsor (cod_patrocinador, cod_equipo, cod_competicion, participacion)

Historial

Relaciona el historial de los pilotos-equipos-coches.

En una competición (temporada de carreras), cada piloto sólo puede estar en un equipo, cada equipo sólo puede tener dos pilotos y tanto los equipos como los pilotos pueden tener varios coches para las carreras.

Un piloto pertenece al equipo con el que comienza la competición aunque ya no corra en él. Sólo puede cambiar de equipo al finalizar la temporada de carreras.



Historial (cod_licencia, cod_coche, cod_equipo, cod_carrera)

Participantes

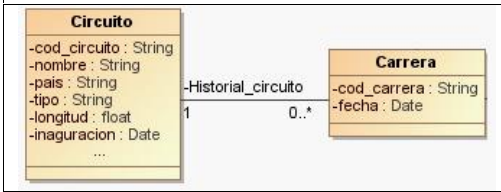
Relaciona los equipos que han participado en las carreras.

Esta relación se puede eliminar porque esta información se puede sacar a través de la relación historial.

Historial_circuito

Relaciona las carreras con los circuitos.

En un circuito se pueden haber disputado muchas carreras o ninguna, debido a que es nuevo o es un nuevo trazado de una carrera de rally, por ejemplo y a su vez cada carrera sólo puede disputarse en un circuito determinado.

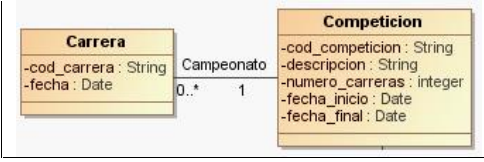


Historial_circuito (cod_circuito , cod_carrera)

Campeonato

Relaciona las carreras que se realizan en cada competición.

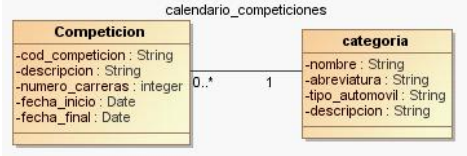
En una competición dada se pueden celebrar de cero carreras (si es una competición nueva) a muchas a lo largo de su historia, pero cada carrera pertenece a una competición en concreto.



Campeonato (cod_carrera , cod_competicion)

Calendario_competiciones

Relaciona las competiciones que ha habido en una categoría.



Calendario_competiciones (cod_competición, nombre)

Resultados

Relaciona los resultados de cada carrera, tanto de clasificación como de carrera.

Ya está definida anteriormente como composición de las clases resultado.

Hay que incluir el cod_carrera dentro de las claves foraneas.

La clave primaria es cod_resultado.

Se computa dentro de esta relación el consumo de combustible realizado.

Resultado (cod_resultado, cod_piloto, cod_carrera, pos_clasificacion, tiempo_clas, pos_carrera, tiempo_carrera, vuelta_rapida, ptos_obtenidos, ptos_escuderia, consumo).

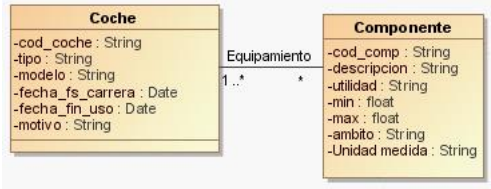
Equipamiento

Relaciona los tipos de componentes con cada coche.

En esta relación es importante poner la fecha de instalación del componente (equipamiento) en el coche, y la vida útil en kilómetros que tendrá el componente.

Cada componente tiene un código propio que los diferencia de los demás, el CIC (código de identificación de componente) y su vida útil es distinta para cada uno.

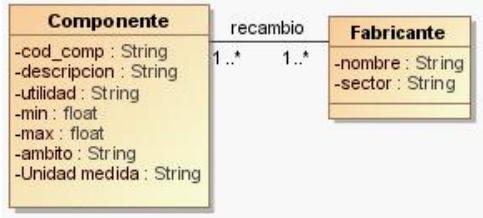
La clave primaria será el CIC.



Equipamiento (cod_coche, cod_comp, cic, fecha_instalacion, vida_util)

Recambio

Relaciona los componentes fabricados por cada fabricante



Recambio (fabricante,identificador)

El identificador es único para cada componente que se fabrica en el mundo.

Al final del análisis tenemos las siguientes relaciones:

Sponsor (cod_patrocinador, cod_equipo, cod_competicion, participacion)

Historial (cod_licencia, cod_coche, cod_equipo, cod_carrera)

Historial_circuito (cod_circuito , cod_carrera)

Campeonato (cod_carrera , cod_competicion)

Calendario_competiciones (cod_competición, nombre)

Resultado (cod_resultado, cod_piloto, cod_carrera, pos_clasificacion, tiempo_clas, pos_carrera, tiempo_carrera, vuelta_rapida, ptos_obtenidos, ptos_escuderia, consumo).

Equipamiento (cod_coche, cod_comp, cic, fecha_instalacion, vida_util)

Recambio (cod_comp, fabricante, identificador)

2.5.2.1 ANÁLISIS DE LAS RELACIONES

1ª forma normal.

Todas las relaciones tienen valores atómicos, por lo tanto están en 1ª forma normal.

2ª forma normal.

Las relaciones que tienen atributos que no pertenecen a la clave candidata son:

- Sponsor (cod_patrocinador, cod_equipo, cod_competicion, participacion)
  El atributo participación depende de las tres claves foráneas que forman la clave primaria de la relación, en cada tupla este valor depende de los otros tres atributos.
  Está en 2ª forma normal.

- Resultado (cod_resultado, cod_piloto, cod_carrera, pos_clasificacion, tiempo_clas, pos_carrera, tiempo_carrera, vuelta_rapida, ptos_obtenidos, ptos_escuderia).
  Todos los atributos que no pertenecen a la clave primaria (pos_clasificación, tiempo_clas, pos_carrera, tiempo_carrera, vuelta_rápida, ptos_obtenidos y ptos_escuderia) dependen de los atributos de la clave foránea (cod_piloto, cod_carrera).
  Estos resultados son particulares de cada carrera y piloto.

- Equipamiento (cod_coche, cod_comp, cic, fecha_instalacion, vida_util)
  El atributo fecha_instalación es independiente del resto de atributos, y vida_util depende del componente en particular.

3ª forma normal

Las relaciones que pueden tener atributos que están referidas por otros atributos que no pertenecen a las claves candidatas son:

- Sponsor (cod_patrocinador, cod_equipo, cod_competicion, participacion)
  Está en 3ª forma normal.

- Resultado (cod_resultado, cod_piloto, cod_carrera, pos_clasificacion, tiempo_clas, pos_carrera, tiempo_carrera, vuelta_rapida, ptos_obtenidos, ptos_escuderia, consumo)
  Los atributos dependen exclusivamente de las claves candidatas.
  Está en 3ª forma normal.

- Equipamiento (cod_coche, cod_comp, cic, fecha_instalacion, vida_util)
  Está en 3ª forma normal.

FNBC

Puesto que no hay dependencias entre los atributos que no son claves candidatas o primarias todas las relaciones se encuentran en FNBC.

4ª forma normal

Comprobar si no hay dependencias multivaluadas independientes.

No hay dependencias multivaluadas independientes en las relaciones.

Conclusión

Todas las relaciones se encuentran normalizadas.

2.6 DISEÑO FÍSICO

El proceso a seguir, es:

Crear la base de datos, crear el tablespace, ejecutar el comando spool para almacenar la información, creación de las tablas, creación de las restricciones, inserción de los datos en las tablas y por último comprobación de la integridad de la base de datos.

2.6.1 BASE DE DATOS

Se crea la base de datos que se va a llamar DB_FIA.

Utilizaré tanto la consola de comandos como la interface gráfica de Oracle Database XE 11.2.

Para ello ejecutamos Oracle Database XE 11.2 y se introducen los datos para crear la base de datos.

La base de datos se ha creado con éxito:

Entramos en la base de datos:

Ahora se crea el tablespace que se llamara TB_FIA, para ello usaré la consola de comandos del propio programa de Oracle Express 11g.

Entramos como usuario system de la base de datos:

```
SQL*Plus: Release 11.2.0.2.0 Production on Sθb Jun 13 18:11:52 2015

Copyright (c) 1982, 2014, Oracle.  All rights reserved.

SQL> conn sys / as sysdba
Enter password:
Connected.
SQL> _
```

Creamos el tablespace TB_FIA en la ruta C:\TB_FIA\tablespace:

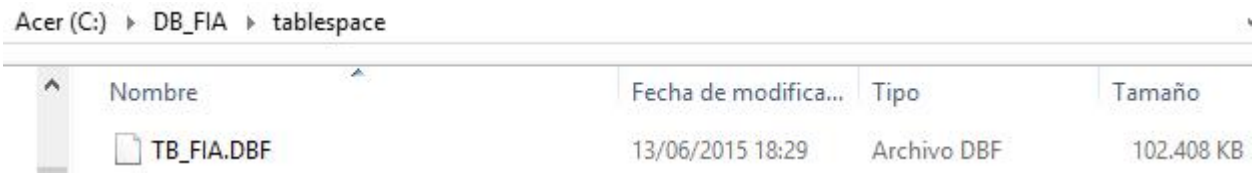
```
SQL> CREATE TABLESPACE TB_FIA
  2  DATAFILE 'C:\DB_FIA\tablespace\TB_FIA.dbf'
  3  SIZE 100M AUTOEXTEND ON NEXT 10M MAXSIZE 5000M
  4  LOGGING
  5  EXTENT MANAGEMENT LOCAL
  6  SEGMENT SPACE MANAGEMENT AUTO;

Tablespace created.
```

Inicialmente el tamaño de almacenamiento del archivo de la base de datos es de 100 megas y se va incrementando en 10 megas hasta un tamaño máximo de 5GB conforme se necesita más espacio de almacenamiento.

Se puede comprobar cómo se ha creado en tablespace en la base de datos DB_FIA:

Tablespace	Free Space (MB)	Used Space (MB)	Percent Used	Maximum (MB)
SYSAUX	39	691		32,768
SYSTEM	9	382		600
USERS	82	18		11,264
UNDOTBS1	366	14		500
TEMP	57	1		32,768
TB_FIA	99	1		5,000

El archivo físico se encuentra en C:\DB_FIA\tablespace\.



2.6.2 SPOOL

Se ejecuta el comando spool que monitoriza toda la actividad en la base de datos hacia una archivo de salida. Este proceso se describe en el apartado 2.7 Archivo log

2.6.3 TABLAS

Las fechas en las tablas serán del tipo DD/MM/YYYY.

Para crear las tablas utilizo SQL Developer, porque con Oracle express sólo me deja crear de una en una tabla, y en SQL developer las puedo crear todas a la vez. Descripción de las tablas que se van a crear:

2.6.3.1 PATROCINADOR

Diseño lógico

- Patrocinador (cod_patrocinador, nombre, sector, pais)

Consideraciones:

- El atributo cod_patrocinador es la clave primaría.
- Los atributos nombre y sector son del tipo VARCHAR2 y no pueden ser NULL.
- El atributo país puede ser NULL, debido a que una organización puede patrocinar un equipo y no se puede determinar el país de origen del patrocinador, por ejemplo la OPEP.

Restricciones:

- No puede haber dos patrocinadores con el mismo nombre.

```
CREATE TABLE Patrocinador (
cod_patrocinador VARCHAR2(10) PRIMARY KEY,
nombre VARCHAR2(25) NOT NULL,
sector VARCHAR2(25) NOT NULL,
pais VARCHAR2(25),
CONSTRAINT nombrePatrocinadorUnico UNIQUE (nombre)
) ;
```

2.6.3.2 EQUIPO

Diseño lógico

- Equipo (cod_equipo, nombre, sede, pais, fecha_debut)

Consideraciones:

- El atributo Cod_equipo es la clave primaria.
- Todos los atributos no admiten valores nulos.

Restricciones:

- El nombre del equipo ha de ser único, no puede haber dos equipos con el mismo nombre aunque sean en diferentes categorías de competición.
- La fecha de debut ha de ser anterior o igual a la fecha actual, pero como Oracle no permite utilizar CHECK con fechas dinámicas, habrá que crear un trigger que controle esta situación.

```
CREATE TABLE Equipo (
cod_equipo VARCHAR2(10) PRIMARY KEY,
nombre VARCHAR2(50) NOT NULL,
sede VARCHAR2(25) NOT NULL,
pais VARCHAR2(25),
```

```
fecha_debut DATE,
CONSTRAINT nombreEquipoUnico UNIQUE (nombre)
) ;
```

2.6.3.3 PILOTO

Diseño lógico

- Piloto (nombre, nacionalidad, cod_licencia, fecha_nacimiento, sexo)

Consideraciones:

- El atributo cod_licencia será la clave primaria.
- El resto de atributos no pueden ser nulos.
- El atributo sexo es necesario, porque hay pilotos femeninos y masculinos en los campeonatos de automovilismo.
- El nombre estará compuesto siempre que se pueda del nombre y apellidos, en el orden (apellido, nombre).

Restricciones:

- La edad mínima para competir en la FIA es de 18 años, así la edad del piloto ha de ser mayor o igual a 18 años, SYSDATE - fecha_nacimiento >=18. Crear tigger para controlar esta restricción.
- El valor para el sexo ha de ser 'V' =varon o 'H'=hembra.

```
CREATE TABLE Piloto (
nombre VARCHAR2(50) NOT NULL,
nacionalidad VARCHAR2(25) NOT NULL,
cod_licencia VARCHAR2(20) PRIMARY KEY,
fecha_nacimiento DATE,
sexo CHAR(1),
CONSTRAINT comprSexo CHECK (sexo='V' OR sexo='H')
);
```

2.6.3.4 COCHE

Diseño lógico

- Coche (cod_coche, tipo, modelo, fecha_st_carrera, fecha_fin_uso, motivo)

Consideraciones:

- En la entidad coche, sólo aparecen los coches que han participado en alguna carrera
- El atributo Cod_coche será la clave primaria.
- El resto de atributos no pueden ser nulos excepto el atributo motivo que hace referencia a la causa del cese de uso del coche, esta causa puede valer NULL y la fecha de fin de uso, porque se puede estar usando aún el coche.
- Cuando un coche es eliminado todos sus componentes son eliminados de la lista de componente, pero no los datos recogidos de los mismos.
- Hay que crear un trigger que controle la eliminación de los componentes.

Restricciones:

- La fecha de fin de uso será posterior o igual a la fecha de primera carrera. Fecha_fin_uso > = fecha_st_carrera, también puede ser NULL, porque el coche está en uso.

```
CREATE TABLE Coche (
cod_coche VARCHAR2(10) PRIMARY KEY,
tipo VARCHAR2(25) NOT NULL,
modelo VARCHAR2(25) NOT NULL,
fecha_st_carrera DATE NOT NULL,
fecha_fin_uso DATE ,
motivo VARCHAR2(50),
CONSTRAINT comprFechafinuso CHECK (fecha_fin_uso IS NULL OR fecha_fin_uso >= fecha_st_carrera)
) ;
```

2.6.3.5 COMPONENTE

Diseño lógico

- Componente (cod_comp, descripcion, utilidad, ámbito, unidad_medida, min, max)

Consideraciones:

- El atributo Cod_comp será la clave primaria.
- Los atributos descripción, fabricante, utilidad, ámbito, unidad_medida, min y max no pueden ser NULL.
- Todos los componentes tienen un intervalo de funcionamiento adecuado (Min y Max), fuera de estos valores el componente funciona mal, estos valores son numéricos.
- Dentro de la descripción de los componentes se incluye el combustible.

Restricciones:

- La medida min ha de ser menor que max.
- No hay restricciones.

```
CREATE TABLE Componente (
cod_comp VARCHAR2(10) PRIMARY KEY,
descripcion VARCHAR2(100) NOT NULL,
utilidad VARCHAR2(100) NOT NULL,
ámbito VARCHAR2(50) NOT NULL,
und_medida VARCHAR2(25) NOT NULL,
min NUMBER NOT NULL,
max NUMBER NOT NULL,
CONSTRAINT ck_minmax CHECK (min < max)
);
```

2.6.3.6 TELEMETRIA

Diseño lógico

- Telemetria (cic, momento, valor)

Consideraciones:

- El atributo cic y momento será la clave primaria
- El atributo valor no puede ser null.

Restricciones:

- No hay restricciones.

```
CREATE TABLE Telemetria (
cic VARCHAR2(20),
momento  VARCHAR2(10) NOT NULL,
valor INTEGER NOT NULL,
CONSTRAINT pk_telemetria PRIMARY KEY (cic,momento),
CONSTRAINT fk_telemetria_coche FOREIGN KEY (cic) REFERENCES Equipamiento (cic)
);
```

2.6.3.7 CIRCUITO

Diseño lógico

- Circuito (cod_circuito, nombre, pais, tipo, longitud, inauguracion)

Consideraciones:

- El atributo cod_circuito será la clave primaria.
- Los atributos nombre, país, tipo y longitud son atributos que no pueden ser NULL.
- La fecha de inauguración del circuito puede ser NULL, porque hay circuitos que son recorridos y por lo tanto no tiene sentido darle una fecha de inauguración ya que cada temporada pueden cambiar.

Restricciones:

- No hay restricciones.

```
CREATE TABLE Circuito (
cod_circuito VARCHAR2(10) PRIMARY KEY,
nombre VARCHAR2(25) NOT NULL,
pais  VARCHAR2(25) NOT NULL,
tipo VARCHAR2(25) NOT NULL,
longitud NUMBER NOT NULL,
inauguración DATE
);
```

2.6.3.8 CARRERA

Diseño lógico

- Carrera (cod_carrera, fecha)

Consideraciones:

- El atributo cod_carrera será la clave primaria.
- Fecha no puede ser NULL y representa cuando se ha disputado o se disputará la carrera.

Restricciones:

- No hay restricciones.

```
CREATE TABLE Carrera (
cod_carrera VARCHAR2(10) PRIMARY KEY,
fecha DATE NOT NULL
);
```

2.6.3.9 COMPETICION

Diseño lógico

- Competicion (cod_competicion, descripción, numero_carreras, fecha_inicio, fecha final)

Consideraciones:

- El atributo cod_competicion será la clave primaria.
- El resto de atributos no pueden ser NULL.

Restricciones:

- La fecha final ha de ser posterior o igual a la fecha de inicio, puede haber competiciones que se hacen una vez cada temporada y sólo hay una carrera.

```
CREATE TABLE Competicion (
cod_competicion VARCHAR2(10) PRIMARY KEY,
descripcion VARCHAR2(25) NOT NULL,
numero_carreras NUMBER NOT NULL,
fecha_inicio DATE NOT NULL,
fecha_final DATE NOT NULL,
CONSTRAINT comprFechasCompeticion CHECK (fecha_final >= fecha_inicio),
CONSTRAINT comprNCarreras CHECK (numero_carreras>0)
);
```

2.6.3.10 CATEGORIA

Diseño lógico

- Categoria (nombre, abreviatura, tipo_automovil, descripción)

Consideraciones:

- El atributo nombre será la clave primaria, no puede haber dos categorías de automovilismo con el mismo nombre.
- El atributo abreviatura puede ser la clave alternativa.

- abreviatura se corresponde al nombre de la carrera abreviado, ejemplo; Formula 1 → F1.
- Abreviatura, tipo de automóvil y descripción son atributos que han de estar presentes para poder describir la categoría a la que se refieren.

Restricciones:

- No hay restricciones.

```
CREATE TABLE Categoria (
nombre  VARCHAR2(50) PRIMARY KEY,
abreviatura VARCHAR2(7) NOT NULL,
tipo_automovil VARCHAR2(25) NOT NULL,
descripción VARCHAR(125) NOT NULL
);
```

2.6.3.11 FABRICANTE

Diseño lógico

- Fabricante (nombre)

Consideraciones:

- La clave primaria es el nombre del fabricante.

Restricciones:

- No hay restricciones.

```
CREATE TABLE Fabricante (
nombre VARCHAR2(50),
sector VARCHAR2 (50),
CONSTRAINT pk_fabricante PRIMARY KEY(nombre)
);
```

2.6.3.12 RESULTADO

Diseño lógico

- Resultado (cod_resultado, cod_piloto, cod_carrera, pos_clasificacion, tiempo_clas, pos_carrera, tiempo_carrera, vuelta_rapida, ptos_obtenidos, ptos_escuderia, consumo)

Consideraciones:

- El atributo cod_resultado será la clave primaria.
- Los atributos cod_piloto y cod_carrera son las claves foráneas.
- Los atributos pos_clasificacion, tiempo_clas, pueden ser nulos porque en algunas competiciones el orden de salida, es el orden en la clasificación y no hay carreras de clasificación ni tiempos de clasificación.
- En una carrera donde si haya carrera de clasificación como la fórmula 1, puede no haber podido disputarla por sanción, por avería, etc… y salir en último lugar sin que haya tiempo de clasificación.

- El atributo vuelta_rapida puede ser nulo, porque en una carrera de rally no se realiza en un circuito cerrado.
- El atributo ptos_escuderia puede ser nulo porque hay competiciones en que no hay competición de fabricantes.
- El atributo tiempo puede ser NULL, porque puede no haber terminado la carrera y no se puede computar el tiempo total.
- Los atributos pos_carrera y ptos_obtenidos no pueden ser NULL.
- Los tiempos se manejan como cadenas de texto del formato (H:MM:SS:SSS, horas:minutos:segundos:milésimas de segundo).
- Los puntos obtenidos por la escudería son la suma de los puntos obtenidos por sus pilotos.
- El consumo serán los litros de combustible consumidos en la carrera.

Restricciones:

- Los puntos obtenidos (ptos_obtenidos) no pueden ser distintos de 25,15, 5 o 0.
- Definir las claves foráneas.
- No puede haber dos tuplas con datos distintos para un mismo piloto en la misma carrera .

```
CREATE TABLE Resultado (
cod_resultado VARCHAR2(15) PRIMARY KEY,
cod_piloto VARCHAR2(20),
cod_carrera VARCHAR2(10),
pos_clasificacion NUMBER,
tiempo_clas  VARCHAR2(12),
pos_carrera NUMBER NOT NULL,
tiempo_carrera VARCHAR2(11) ,
vuelta_rapida VARCHAR2(12),
ptos_obtenidos NUMBER NOT NULL,
ptos_escuderia NUMBER NOT NULL,
consumo INTEGER NOT NULL,
CONSTRAINT ckpuntos  CHECK (ptos_obtenidos=25 OR ptos_obtenidos=15 OR ptos_obtenidos=5 OR ptos_obtenidos=0),
CONSTRAINT ck_ptos_escuderia CHECK (ptos_escuderia=25 OR ptos_escuderia=15 OR ptos_escuderia=5 OR
ptos_escuderia=0),
CONSTRAINT fkpiloto_resultado FOREIGN KEY (cod_piloto) REFERENCES Piloto (cod_licencia),
CONSTRAINT fkequipo_resultado FOREIGN KEY (cod_carrera) REFERENCES Carrera (cod_carrera)
);
```

2.6.3.13 SPONSOR

Diseño lógico

- Sponsor (cod_patrocinador, cod_equipo, cod_competicion, participacion)

Consideraciones:

- El atributo cod_patrocinador será la clave primaria.
- Los atributos cod_equipo, cod_competicion y cod_patrocinador serán las claves foráneas.
- El atributo participación será el dinero de patrocinio en el equipo en miles de euros. Así un millón de euros será 1000.

Restricciones:

- El atributo participación será un entero mayor o igual a cero.

```
CREATE TABLE Sponsor (
cod_patrocinador VARCHAR2(10) PRIMARY KEY,
cod_equipo VARCHAR2(10) ,
cod_competicion VARCHAR2(10) ,
participacion NUMBER NOT NULL,
CONSTRAINT ckparticipacion_sponsor  CHECK (participación>=0),
CONSTRAINT fkpatroc_sponsor FOREIGN KEY (cod_patrocinador) REFERENCES Patrocinador (cod_patrocinador),
CONSTRAINT fkequipo_sponsor FOREIGN KEY (cod_equipo) REFERENCES Equipo (cod_equipo),
CONSTRAINT fkcompeticion_sponsor FOREIGN KEY (cod_competicion) REFERENCES Competicion (cod_competicion)
);
```

2.6.3.14 HISTORIAL

Diseño lógico

- Historial (cod_licencia, cod_coche, cod_equipo, cod_carrera)

Consideraciones:

- Todos los atributos son claves foráneas.
- La clave primaria es el conjunto de las cuatro claves foráneas.

Restricciones:

- No hay restricciones.

```
CREATE TABLE Historial (
cod_licencia VARCHAR2(20),
cod_coche VARCHAR2(10) ,
cod_equipo VARCHAR2(10) ,
cod_carrera VARCHAR2(10),
CONSTRAINT pk_historial PRIMARY KEY(cod_licencia,cod_coche,cod_equipo,cod_carrera),
CONSTRAINT fklicencia_historial FOREIGN KEY (cod_licencia) REFERENCES Piloto (cod_licencia),
CONSTRAINT fkcoche_historial FOREIGN KEY (cod_coche) REFERENCES Coche (cod_coche),
CONSTRAINT fkequipo_historial FOREIGN KEY (cod_equipo) REFERENCES Equipo (cod_equipo),
CONSTRAINT fkcarrera_historial FOREIGN KEY (cod_carrera) REFERENCES Carrera (cod_carrera)
);
```

2.6.3.15 HISTORIAL_CIRCUITO

Diseño lógico

- Historial_circuito (cod_circuito , cod_carrera)

Consideraciones:

- Ambos atributos son claves foráneas.
- La clave primaria está compuesta por ambas claves foráneas.

Restricciones:

- No hay restricciones.

```
CREATE TABLE Historial_circuito (
cod_circuito VARCHAR2(10) ,
cod_carrera VARCHAR2(10) ,
CONSTRAINT pk_HistorialCircuito PRIMARY KEY(cod_circuito, cod_carrera),
CONSTRAINT fkcircuito_Historialcircuito FOREIGN KEY (cod_circuito) REFERENCES Circuito (cod_circuito),
CONSTRAINT fkcarrera_Historialcircuito FOREIGN KEY (cod_carrera) REFERENCES Carrera (cod_carrera)
);
```

2.6.3.16 CAMPEONATO

Diseño lógico

- Campeonato (cod_carrera , cod_competicion)

Consideraciones:

- Ambos atributos son claves foráneas.
- La clave primaria está compuesta por las claves foráneas cod_carrera y cod_competición.

Restricciones:

- No hay restricciones.

```
CREATE TABLE Campeonato (
cod_carrera VARCHAR2(10) ,
cod_competicion VARCHAR2(10) ,
CONSTRAINT pk_campeonato PRIMARY KEY(cod_carrera,cod_competicion),
CONSTRAINT fkcarrera_campeonato FOREIGN KEY (cod_carrera) REFERENCES Carrera (cod_carrera),
CONSTRAINT fkcompeticion_campeonato FOREIGN KEY (cod_competicion) REFERENCES Competicion (cod_competicion)
);
```

2.6.3.17 CALENDARIO_COMPETICIONES

Diseño lógico

- Calendario_competiciones (cod_competición, nombre)

Consideraciones:

- Ambos atributos son claves foráneas.
- La clave primaria es la compuesta por las claves foráneas.

Restricciones:

- No hay restricciones.

```
CREATE TABLE Calendario_competiciones (
cod_competicion VARCHAR2(10) ,
nombre VARCHAR2(50) ,
CONSTRAINT pk_calendario_competiciones PRIMARY KEY (cod_competicion, nombre),
CONSTRAINT fkcompeticion_calendario FOREIGN KEY (cod_competicion) REFERENCES Competicion (cod_competicion),
CONSTRAINT fknombre_calendario FOREIGN KEY (nombre) REFERENCES Categoria (nombre)
);
```

2.6.3.18 EQUIPAMIENTO

Diseño lógico

- Piezas (cod_coche, cod_comp, cic, fecha_instalacion, vida_util)

Consideraciones:

- La clave primaria será el código CIC del componente.
- Las claves foráneas serán cod_coche y cod_comp.
- El atributo vida útil será dado en kilómetros de uso.
- Los atributos fecha_instalación y vida_util no pueden ser NULL.

Restricciones:

- El atributo vida útil ha de ser mayor de cero.
- La fecha de instalación será anterior a la fecha de entrada de datos, esto se controla mediante un trigger.

```
CREATE TABLE Equipamiento (
cod_coche VARCHAR2(10) ,
cod_comp VARCHAR2(10) ,
cic VARCHAR2(15) PRIMARY KEY,
fecha_instalacion DATE NOT NULL,
vida_util NUMBER NOT NULL,
CONSTRAINT ck_vidautil CHECK (vida_util>0),
CONSTRAINT fkcoche_componentes FOREIGN KEY (cod_coche) REFERENCES Coche (cod_coche),
CONSTRAINT fkcomponente_componentes FOREIGN KEY (cod_comp) REFERENCES Componente (cod_comp)
);
```

2.6.3.19 RECAMBIO

Diseño lógico

- Recambio (fabricante, identificador)

Consideraciones:

- La clave primaria es el identificador que en cada componente fabricado es único.
- Fabricante es clave foránea.

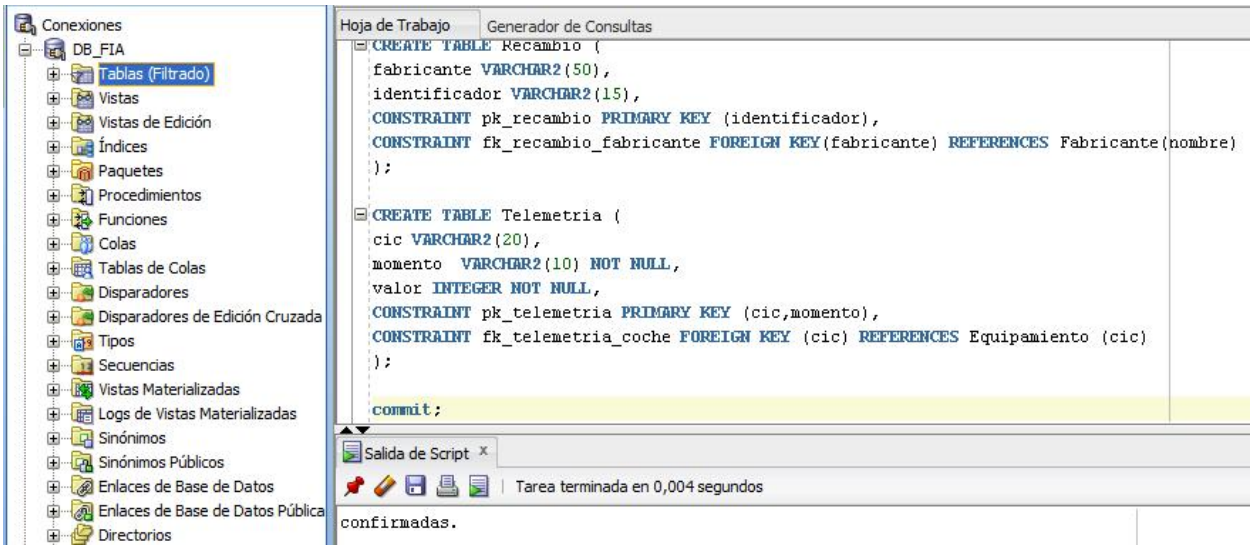
Restricciones:

- Identificador sólo es un valor numérico.

```
CREATE TABLE Recambio (
fabricante VARCHAR2(50),
identificador VARCHAR2(15),
CONSTRAINT pk_recambio PRIMARY KEY (identificador),
CONSTRAINT fk_recambio_fabricante FOREIGN KEY(fabricante) REFERENCES Fabricante(nombre),
);
```

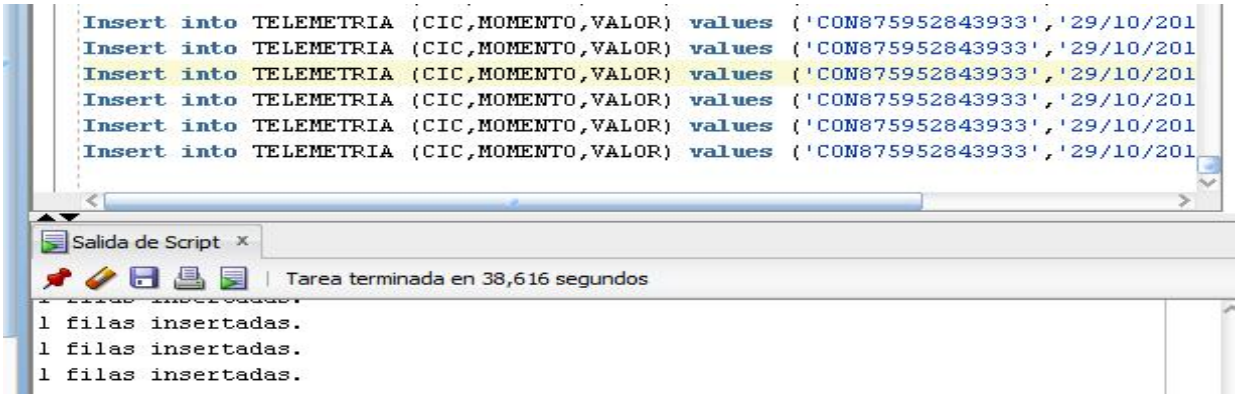
2.6.4 CREACION DE LAS TABLAS

Se carga el archivo tablas.txt dentro de una instrucción SQL para ejecutar la creación de tablas.



2.6.5 INSERCIÓN DE LOS DATOS EN LAS TABLAS

Se hace como en el caso anterior, se carga el archivo datos.txt dentro de una instrucción SQL.



2.6.6 PROCEDIMIENTO ABM (ALTA, BAJA, MODIFICACIÓN)

Los procesos de alta, baja y modificación sólo hay que tenerlos en cuenta en aquellas clases que hacen referencia a claves ajenas en alguno de sus atributos.

No hay problema en dar de alta registros siempre que cumplan con las especificaciones de la tabla donde van a ir.

Relación de las clases que tienen claves ajenas, en rojo las tablas que no se pueden adaptar o borrar los datos que tienen, en verde las que sí:

TABLA	TABLA REFERENCIADA	PROCEDIMIENTO	
		UPDATE	DELETE
Recambio	Fabricante	No se puede hacer update o delete sobre al atributo nombre de la clase fabricante. Hay que tener un historial del recambio fabricado y el fabricante que lo hizo en su momento. Si borramos a un fabricante de la tabla, tendremos recambios sin saber quién los fabrico. Sólo se puede borrar si no está referenciado en otras tablas.	
Equipamiento	Coche Componente	Se puede adaptar el código del coche y el código del componente en cascada mediante un trigger.	Solo se puede borrar si no hay datos referenciados en otras tablas.
Resultado	Piloto Carrera	Se puede hacer un update. Controlado mediante un trigger.	Solo se puede borrar si no hay datos referenciados en otras tablas.
Telemetría	Equipamiento	Se puede hacer un update sobre el cic del equipamiento. Se controla mediante un trigger.	Solo se puede borrar si no hay datos referenciados en otras tablas.
Sponsor	Patrocinador Equipo Competición	No se puede hacer ni update ni delete sobre patrocinador, si se puede sobre Equipo y si se puede hacer update sobre competición.	Solo se puede borrar si no hay datos referenciados en otras tablas.
Historial	Carrera Coche Piloto Equipo	Se pueden adaptar.	Solo se puede borrar si no hay datos referenciados en otras tablas.
Historial circuito	Carrera Circuito	Se pueden adaptar.	Solo se puede borrar si no hay datos referenciados en otras tablas.
Campeonato	Carrera Competición	Se pueden adaptar.	Solo se puede borrar si no hay datos referenciados en otras tablas.
Calendario competiciones	Categoría Competicion	No se puede adaptar, porque tendríamos el historial de carreras mal, por ejemplo.	Solo se puede borrar si no hay datos referenciados en otras tablas.

En resumen, las clases que hacen referencia a claves foráneas que son códigos de identificación (código de equipo, licencia de piloto, etc) si se puede hacer un update y las que hacen referencia a nombres, no se pueden cambiar.

No se pueden borrar datos de una tabla si están referenciados en otras.

Para controlar los updates hay que hacerlo mediante un trigger en cada una de las tablas en las que se hace la actualización y que son referenciadas por atributos de otras tablas, estas tablas son:

TABLA	TABLAS QUE REFERENCIAN
Coche	Equipamiento, historial
Componente	Equipamiento
Piloto	Resultado, historial
Carrera	Resultado, historial, historial circuito, campeonato
Equipamiento	Telemetría
Equipo	Sponsor, historial
Competicion	Sponsor, campeonato, calendario_competiciones
Circuito	Historial circuito

2.6.6.1 DISPARADORES DE LAS TABLAS

Tabla Coche

```
create or replace trigger cascade_update_coche
after update of cod_coche on Coche
for each row
begin
  update equipamiento
  set equipamiento.cod_coche = :new.cod_coche
  where equipamiento.cod_coche = :old.cod_coche;

  update historial
  set historial.cod_coche = :new.cod_coche
  where historial.cod_coche = :old.cod_coche;
end;
```

Tabla Componente

```
create or replace trigger cascade_update_componente
after update of cod_comp on componente
for each row
begin
  update equipamiento
  set equipamiento.cod_comp = :new.cod_comp
  where equipamiento.cod_comp = :old.cod_comp;
end;
```

Tabla Piloto

```
create or replace trigger cascade_update_piloto
after update of cod_licencia on Piloto
for each row
begin
  update resultado
  set resultado.cod_piloto = :new.cod_licencia
  where resultado.cod_piloto = :old.cod_licencia;
```

```
  update historial
  set historial.cod_licencia = :new.cod_licencia
  where historial.cod_licencia = :old.cod_licencia;
end;
```

Tabla Carrera

```
create or replace trigger cascade_update_carrera
after update of cod_carrera on Carrera
for each row
begin
  update resultado
  set resultado.cod_carrera = :new.cod_carrera
  where resultado.cod_carrera = :old.cod_carrera;

  update historial
  set historial.cod_carrera = :new.cod_carrera
  where historial.cod_carrera = :old.cod_carrera;

  update historial_circuito
  set historial_circuito.cod_carrera = :new.cod_carrera
  where historial_circuito.cod_carrera = :old.cod_carrera;

  update campeonato
  set campeonato.cod_carrera = :new.cod_carrera
  where campeonato.cod_carrera = :old.cod_carrera;

end;
```

Tabla Equipamiento

```
create or replace trigger cascade_update_equipamiento
after update of cod_comp on equipamiento
for each row
begin
  update telemetria
  set telemetria.cod_comp = :new. cod_comp
  where telemetria.cod_comp = :old.cod_comp;
end;
```

Tabla equipo

```
create or replace trigger cascade_update_equipo
after update of cod_equipo on Equipo
for each row
begin
  update Sponsor
  set Sponsor.cod_equipo = :new.cod_equipo
  where Sponsor.cod_equipo = :old.cod_equipo;

  update historial
```

```
  set historial.cod_equipo = :new.cod_equipo
  where historial.cod_equipo = :old.cod_equipo;
end;
```

Tabla competición

```
create or replace trigger cascade_update_competicion
after update of cod_competicion on Competicion
for each row
begin
  update Sponsor
  set Sponsor.cod_competicion = :new.cod_competicion
  where Sponsor.cod_competicion = :old.cod_competicion;

  update campeonato
  set campeonato.cod_competicion = :new.cod_competicion
  where campeonato.cod_competicion = :old.cod_competicion;

  update calendario_competiciones
  set calendario_competiciones.cod_competicion = :new.cod_competicion
  where calendario_competiciones.cod_competicion = :old.cod_competicion;

end;
```

Tabla Circuito

```
create or replace trigger cascade_update_circuito
after update of cod_circuito on circuito
for each row
begin
  update circuito
  set circuito.cod_circuito = :new.cod_circuito
  where circuito.cod_circuito = :old.cod_circuito;
end;
```

2.6.6.2 EJEMPLO DE FUNCIONAMIENTO

Tomando como ejemplo la tabla equipo, si modificamos el código de equipo, se debían adaptar los datos en el resto de tablas que referencian este atributo.

Antes de hacer nada, tenemos:

Tabla equipo:

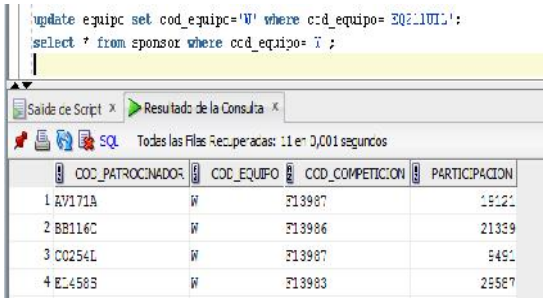
```
select * from equipo
```

Salida de Script | Resultado de la Co... | Todas las Filas Recuperadas: 9 en 0,003 segundos

	COD_EQUIPO	NOMBRE	SEDE	PAIS	FECHA_
1	EQ713MER	Mercedes-Benz Grand Prix Limited	Barckley	Reino Unido	03/12/54
2	EQ496SCU	Scuderia Ferrari	Maranello	Italia	01/01/50
3	EQ211WIL	Williams F1 Team	Grove	Reino Unido	04/12/75
4	EQ401INF	Infiniti Red Bull Racing	Milton Keynes	Reino Unido	09/06/05
5	EQ275MCL	McLaren-Honda	Woking	Reino Unido	08/08/66
6	EQ202LOT	Lotus F1 Team	Enstone	Reino Unido	11/05/77
7	EQ700SAU	Sauber F1 Team	Hinwil	Suiza	04/03/93
8	EQ1098AH	Sahara Force India F1 Team	Silverstone	Reino Unido	08/08/08
9	EQ567SCU	Scuderia Toro Rosso	Faenza	Italia	06/05/06

Modifico el código del equipo Williams F1 team por W.

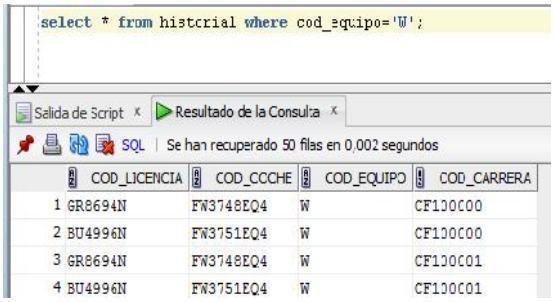
Ahora se muestra el resultado del cambio en las tablas afectadas (sponsor y campeonato).



```
update equipo set cod_equipo='W' where cod_equipo='EQ211WIL';
select * from sponsor where cod_equipo='W';
```

Salida de Script | Resultado de la Consulta

	COD_PATROCINADOR	COD_EQUIPO	COD_COMPETICION	PARTICIPACION
1	AV171A	W	F13987	19121
2	BB116C	W	F13986	21339
3	CO254L	W	F13987	9491
4	EL458S	W	F13983	29587



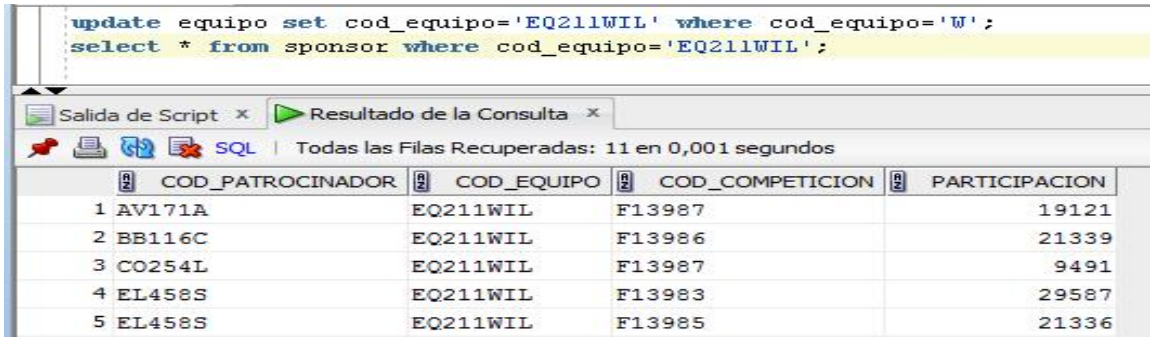
```
select * from historial where cod_equipo='W';
```

Salida de Script | Resultado de la Consulta

SQL | Se han recuperado 50 filas en 0,002 segundos

	COD_LICENCIA	COD_COCHE	COD_EQUIPO	COD_CARRERA
1	GR8694N	FW3748EQ4	W	CF100C00
2	BU4996N	FW3751EQ4	W	CF100C00
3	GR8694N	FW3748EQ4	W	CF100C01
4	BU4996N	FW3751EQ4	W	CF100C01

Ahora se vuelve a poner los datos como estaban:



```
update equipo set cod_equipo='EQ211WIL' where cod_equipo='W';
select * from sponsor where cod_equipo='EQ211WIL';
```

Salida de Script | Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 11 en 0,001 segundos

	COD_PATROCINADOR	COD_EQUIPO	COD_COMPETICION	PARTICIPACION
1	AV171A	EQ211WIL	F13987	19121
2	BB116C	EQ211WIL	F13986	21339
3	CO254L	EQ211WIL	F13987	9491
4	EL458S	EQ211WIL	F13983	29587
5	EL458S	EQ211WIL	F13985	21336

2.6.7 DISPARADORES

Hay que comprobar para cada tabla y para cada relación como controlar la entrada de los datos para que estos sean fiables y la base de datos sea consistente.

Debido a que Oracle no permite crear restricciones que operen con fechas dinámicas, los controles sobre las fechas de las distintas tablas hay que realizarlas con disparadores.

Todos los disparadores son para realizar control sobre los datos de fecha, estos datos nunca pueden ser posteriores a la fecha actual. Por ejemplo, la fecha de debut de un coche en competición no puede ser posterior a la fecha actual, así como un piloto no puede haber nacido posteriormente a la fecha actual.

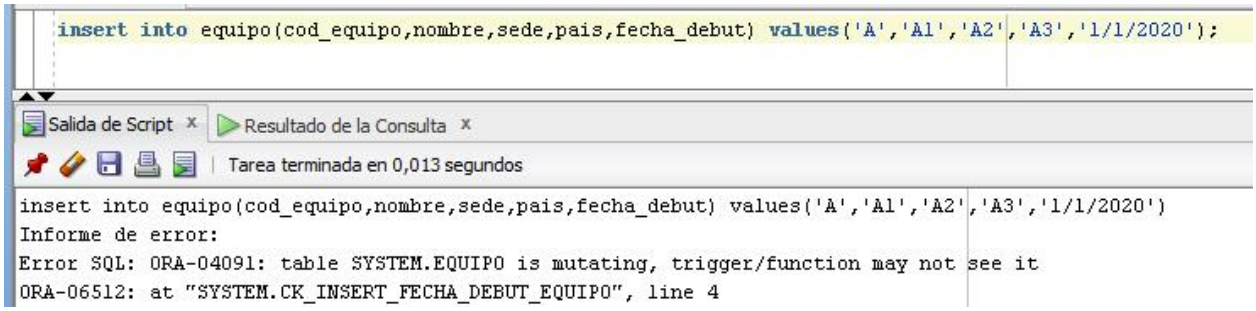
Tablas en las que se debe usar un disparador:

2.6.7.1 EQUIPO

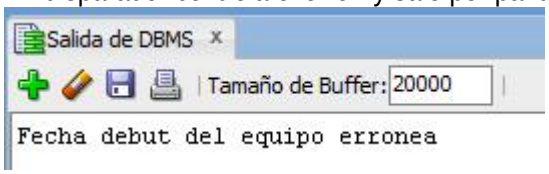
Se debe controlar la fecha de debut.

```
CREATE OR REPLACE TRIGGER ck_insert_fecha_debut_equipo
AFTER INSERT OR UPDATE ON EQUIPO
FOR EACH ROW
begin
IF( :NEW.fecha_debut > SYSDATE)THEN
    DBMS_OUTPUT.PUT_LINE('Fecha debut del equipo erronea');
    DELETE FROM EQUIPO
    WHERE EQUIPO.COD_EQUIPO=:NEW.COD_EQUIPO;
END IF;
IF( :NEW.fecha_debut < SYSDATE)THEN
    DBMS_OUTPUT.PUT_LINE('RSP: OK');
END IF;
END;
```

Comprobación:
Se intenta insertar datos nuevos con la fecha errónea:



EL disparador controla el error y sale por pantalla el mensaje de error.



Ahora se introduce con la fecha correcta:

```
insert into equipo(cod_equipo,nombre,sede,pais,fecha_debut) values('A','A1','A2','A3','1/1/2010');
```

Salida de Script | Resultado de la Consulta

Tarea terminada en 0,001 segundos

```
1 filas insertadas.
OK
```

Salida de DBMS

Tamaño de Buffer: 20000

```
OK
```

2.6.7.2 PILOTO

Se debe controlar la fecha de nacimiento del piloto, como la edad mínima para conducir es de 16 años, el piloto no puede haber nacido después del año actual menos 16 años.

```
CREATE OR REPLACE  TRIGGER ck_insert_fecha_nac_piloto
AFTER INSERT OR UPDATE ON PILOTO
FOR EACH ROW
begin
IF( (:NEW.fecha_nacimiento + 16) > SYSDATE)THEN
    DBMS_OUTPUT.PUT_LINE('Fecha nacimiento del piloto, erronea.');
    DELETE FROM PILOTO
    WHERE PILOTO.COD_LICENCIA=:NEW.COD_LICENCIA;
END IF;
IF( (:NEW.fecha_nacimiento + 16) < SYSDATE)THEN
    DBMS_OUTPUT.PUT_LINE('RSP: OK');
END IF;
END;
```

Comprobación

Se intentan entrar datos de un piloto nuevo con la fecha errónea.

Hoja de Trabajo | Generador de Consultas

```
insert into piloto (nombre,nacionalidad,cod_licencia,fecha_nacimiento,sexo) values('B','B1','B00','1/1/2000','V');
```

Salida de Script

Tarea terminada en 0,01 segundos

```
insert into piloto (nombre,nacionalidad,cod_licencia,fecha_nacimiento,sexo) values('B','B1','B00','1/1/2000','V')
Informe de error:
Error SQL: ORA-04091: table SYSTEM.PILOTO is mutating, trigger/function may not see it
ORA-06512: at "SYSTEM.CK_INSERT_FECHA_NAC_PILOTO", line 4
```

El trigger funciona y no deja insertar los datos. Sale el mensaje de error en pantalla.

Ahora se introducen los datos correctos:

Hoja de Trabajo | Generador de Consultas

```
insert into piloto (nombre,nacionalidad,cod_licencia,fecha_nacimiento,sexo) values('B','B1','B00','1/1/1999','V');
```

Salida de Script

Tarea terminada en 0 segundos

```
1 filas insertadas.
OK
```

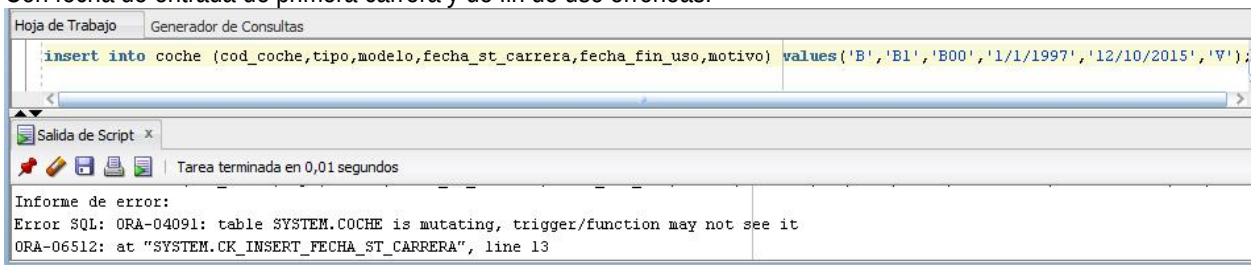
2.6.7.3 COCHE

Se debe controlar la fecha de debut en competición.

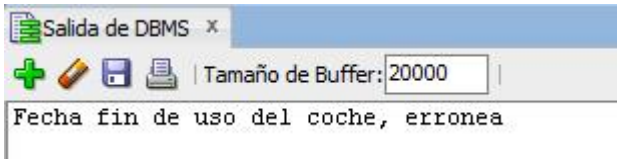
```
CREATE OR REPLACETRIGGER ck_insert_fecha_coche
AFTER INSERT OR UPDATE ON COCHE
FOR EACH ROW
DECLARE
  errores integer;
begin
errores:=0;
IF( :NEW.fecha_st_carrera > SYSDATE)THEN
       DBMS_OUTPUT.PUT_LINE('Fecha primera carrera del coche, erronea');
       errores:=1;
     elsif(:NEW.fecha_fin_uso > SYSDATE)then
       DBMS_OUTPUT.PUT_LINE('Fecha fin de uso del coche, erronea');
       errores:=1;
END IF;
if errores=1 then
     DELETE FROM COCHE
     WHERE COCHE.COD_COCHE=:NEW.COD_COCHE;
end if;
IF( errores=0)THEN
     DBMS_OUTPUT.PUT_LINE('RSP: OK');
END IF;
END;
```

Comprobación

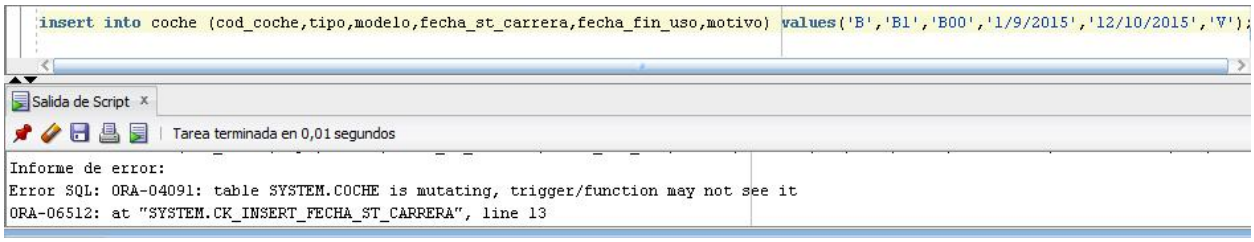
Con fecha de entrada de primera carrera y de fin de uso erróneas.



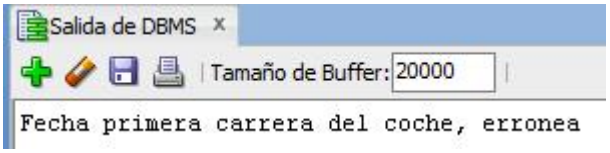
El trigger funciona y avisa de la fecha de fin de uso del coche errónea.



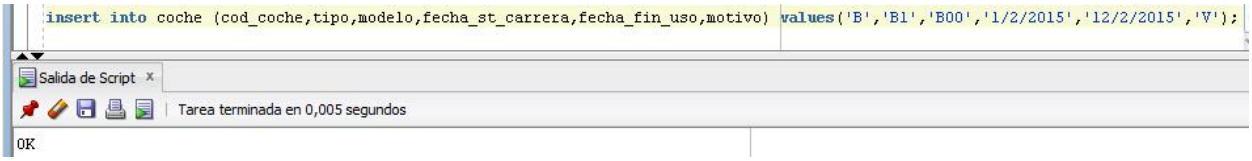
Con la fecha de la primera carrera errónea.



El trigger funciona avisa de la fecha de la primera carrera errónea.



Con todo correcto se inserta bien la información



2.6.7.4 CIRCUITO

Se debe controlar la fecha de inauguración.

```
CREATE OR REPLACE TRIGGER ck_insert_inagu_circuito
AFTER INSERT OR UPDATE ON CIRCUITO
FOR EACH ROW
begin
IF( :NEW.INAUGURACION > SYSDATE)THEN
      DBMS_OUTPUT.PUT_LINE('Fecha inauguración del circuito, erronea');
      DELETE FROM CIRCUITO
      WHERE CIRCUITO.INAUGURACION=:NEW.INAUGURACION;
END IF;
IF( :NEW.INAUGURACION < SYSDATE)THEN
    DBMS_OUTPUT.PUT_LINE('RSP: OK');
END IF;
END;
```

2.6.7.5 CARRERA

Se debe controlar la fecha de la carrera.

```
CREATE OR REPLACE TRIGGER ck_fecha_carrera
AFTER INSERT OR UPDATE ON CARRERA
FOR EACH ROW
begin
IF( :NEW.fecha > SYSDATE)THEN
      DBMS_OUTPUT.PUT_LINE('Fecha carrera, erronea');
      DELETE FROM CARRERA
      WHERE CARRERA.fecha=:NEW.fecha;
END IF;
IF( :NEW.fecha < SYSDATE)THEN
    DBMS_OUTPUT.PUT_LINE('RSP: OK');
END IF;
END;
```

2.6.7.6 COMPETICIÓN

Se debe controlar la fecha de inicio.

```
CREATE OR REPLACE TRIGGER ck_fecha_comp_inicio
AFTER INSERT OR UPDATE ON COMPETICION
FOR EACH ROW
begin
IF( :NEW.fecha_inicio > SYSDATE)THEN
      DBMS_OUTPUT.PUT_LINE('Fecha inicio carrera, erronea');
      DELETE FROM COMPETICION
      WHERE COMPETICION.fecha_inicio=:NEW.fecha_inicio;
END IF;
IF( :NEW.fecha_inicio < SYSDATE)THEN
    DBMS_OUTPUT.PUT_LINE('RSP: OK');
END IF;
END;
```

2.6.7.7 EQUIPAMIENTO

Se debe controlar la fecha de instalación del componente.

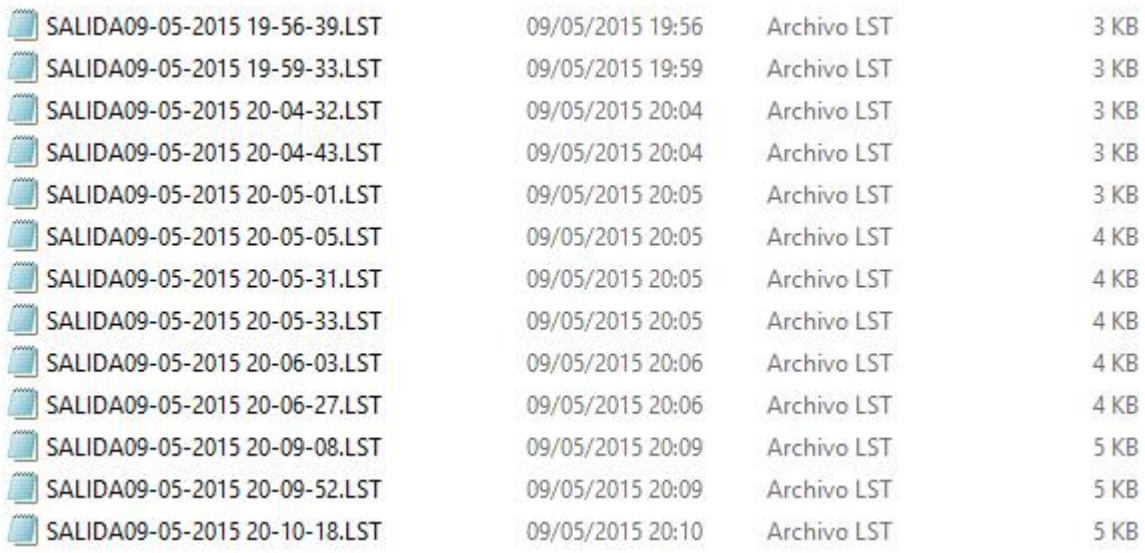
```
CREATE OR REPLACE TRIGGER ck_fecha_inst_equip
AFTER INSERT OR UPDATE ON EQUIPAMIENTO
FOR EACH ROW
begin
IF( :NEW.fecha_instalacion > SYSDATE)THEN
      DBMS_OUTPUT.PUT_LINE('Fecha instalación equipamiento, erronea');
      DELETE FROM EQUIPAMIENTO
      WHERE EQUIPAMIENTO.fecha_instalacion=:NEW.fecha_instalacion;
END IF;
IF( :NEW.fecha_instalacion < SYSDATE)THEN
    DBMS_OUTPUT.PUT_LINE('RSP: OK');
END IF;
END;
```

2.7 ARCHIVO LOG

El archivo log reúne la información de las acciones realizadas en la base de datos.

Cada vez que se realiza una acción, se crea un archivo de extensión log, el cual recoge las acciones realizadas, las salidas de esas acciones y los mensajes de error si se producen, así como el mensaje OK si todo ha ido bien.

Cada archivo creado tendrá un nombre, por ejemplo: SALIDA y la fecha y hora del archivo creado, así podemos tener los siguientes archivos guardados:



De esta forma nos aseguramos que todas las acciones sobre la base de datos quedan registradas.

2.7.1 PROCESO

1. Ejecutamos la siguiente instrucción para que se pueda crear el archivo con el formato especificado anteriormente:

```
column date_column new_value today_var
select to_char(sysdate,'dd-mm-yyyy HH24-MI-SS') date_column
from dual
/
spool SALIDA&today_var
```

2. Activamos la consola para que los mensajes puedan salir por pantalla:

```
set serveroutput on;
```

3. Se utiliza la instrucción de Oracle para sacar mensajes a pantalla:

```
DBMS_OUTPUT.PUT_LINE(' texto a sacar en consola');
```

2.7.2 EJEMPLO DE EJECUCIÓN:

Órdenes a ejecutar:

```
set serveroutput on;
column date_column new_value today_var
select to_char(sysdate,'dd-mm-yyyy HH24-MI-SS') date_column
from dual;
spool SALIDA &today_var;
```

Órdenes para iniciar la salida de mensajes y dar formato al nombre del archivo de salida

```
select * from equipo
order by cod_equipo;
select * from patrocinador;
delete from carrera where cod_carrera='PR1';
insert into carrera(cod_carrera,fecha) values('a00000','1/1/2020');
```

Acciones sobre la base de datos; selección, borrado e inserción con fecha errónea.

```
> select * from equipo
order by cod_equipo
COD_EQUIPO NOMBRE                                          SEDE                PAIS                FECHA_DEBUT
---------- -------------------------------------------------- ------------------------- ------------------------- -----------
EQ109SAH   Sahara Force India F1 Team                      Silverstone         Reino Unido         08/08/08
EQ202LOT   Lotus F1 Team                                   Enstone             Reino Unido         11/05/77
EQ211WIL   Williams F1 Team                                Grove               Reino Unido         04/12/75
-          -          -          -          -          -          -          -          -          -          -
EQ700SAU   Sauber F1 Team                                  Hinwill             Suiza               04/03/93
EQ713MER   Mercedes-Benz Grand Prix Limited                Barckley            Reino Unido         03/12/54
1          1                                               1                   1                   21/05/15

 10 filas seleccionadas

> select * from patrocinador
COD_PATROCINADOR NOMBRE                SECTOR                PAIS
---------------- ------------------------- ------------------------- -------------------------
PI302I           Pirelli               Neumaticos            Italia
RO398X           Rolex                 Joyeria               Suiza
FL144S           Fly Emirates          Aerolineas            Arabia Saudi
-          -          -          -          -          -          -          -
SH440P           sharp                 Tecnologia            Japon
AV171A           Avianca               Aerolineas            Colombia
BB116C           BBC                   Comunicaciones        Reino Unido
GU318F           Gulf                  Petroquimico          EEUU

 39 filas seleccionadas

> delete from carrera where cod_carrera='PR1'
0 filas eliminado
> insert into carrera(cod_carrera,fecha) values('a00000','1/1/2020')

Error que empieza en la línea 17 del comando:
insert into carrera(cod_carrera,fecha) values('a00000','1/1/2020')
Informe de error:
Error SQL: ORA-04091: table SYSTEM.CARRERA is mutating, trigger/function may not see it
ORA-06512: at "SYSTEM.CK_FECHA_CARRERA", line 4
ORA-04088: error during execution of trigger 'SYSTEM.CK_FECHA_CARRERA'
04091. 00000 -  "table %s.%s is mutating, trigger/function may not see it"
*Cause:    A trigger (or a user defined plsql function that is referenced in
           this statement) attempted to look at (or modify) a table that was
           in the middle of being modified by the statement which fired it.
*Action:   Rewrite the trigger (or function) so it does not read that table.

Fecha carrera, errónea.
```

2.8 REPOSITORIO

Como los datos introducidos en las tablas del repositorio estadístico proceden de consultas a la base de datos, no hace falta poner restricciones de integridad, porque estas ya están cubiertas al introducir los datos en las tablas de la base de datos.

Creación de las tablas que contendrán el repositorio estadístico, para la consulta -les pongo una w delante para que queden al final en el listado de tablas-:

2.8.1 CREACION DE LAS TABLAS DEL REPOSITORIO

Porcentaje de componentes defectuosos.

- Atributos: código del coche (cod_coche) y porcentaje (porcentaje).
- Creación tabla:

```
CREATE TABLE wrepo_prctj_comp_defec (
cod_coche VARCHAR2(10),
porcentaje number(*,3));
```

Resultado de cada piloto en cada competición disputada.

- Atributos: nombre de piloto (nombre), competición en la que compite (competición) y puntos obtenidos (puntos).
- Creación tabla:

```
CREATE TABLE wrepo_clasif_pilotos (
nombre VARCHAR2(50),
competicion VARCHAR2(10),
puntos NUMBER(*,0));
```

Listado de los 10 pilotos que han terminado más carreras de todos los datos registrados.

- Atributos: licencia el piloto (cod_piloto), nombre del piloto (nombre) y carreras terminadas.
- Creación tabla:

```
CREATE TABLE wrepo_carr_pilotos (
cod_piloto VARCHAR2(20),
nombre VARCHAR2(50),
carreras_terminadas NUMBER(*,0));
```

Clasificación de cada competición –suma de puntos de los pilotos-.

- Atributos: tipo de competición (descripción), nombre del piloto (nombre) y puntos obtenidos (puntos).
- Creación tabla:

```
CREATE TABLE wrepo_clasificacion (
descripcion VARCHAR2(25),
nombre VARCHAR2(50),
puntos NUMBER(*,0));
```

Dado un circuito, las 10 vueltas más rápidas que se han dado en él –se ha de indicar el tiempo de la vuelta, el piloto, la fecha, y el equipo.

- Atributos: nombre del circuito (nom_circuito), tiempo de la vuelta rápida (vuelta_rapida), fecha en que se produjo la vuelta rápida (fecha), nombre del piloto (piloto), escudería a la que pertenecía el piloto (escudería).
- Creación tabla:

```
CREATE TABLE wrepo_10_vult_rapi_circu (
nom_circuito VARCHAR2(25),
vuelta_rapida VARCHAR2(12),
fecha DATE,
piloto VARCHAR2(50),
escuderia VARCHAR2(50));
```

Temperatura más alta registrada en el coche, indicar la fecha, el circuito, el coche, el piloto y el valor.

- Atributos: momento en que se registró la temperatura (momento), nombre del circuito (circuito), tipo de coche (tipo_coche), modelo del coche (modelo_coche), nombre del piloto (piloto), temperatura alcanzada (temperatura).
- Creación tabla:

```
CREATE TABLE wrepo_temperatura (
momento VARCHAR2(23),
circuito VARCHAR2(25),
tipo_coche VARCHAR2(25),
modelo_coche VARCHAR2(25),
piloto VARCHAR2(50),
temperatura NUMBER(*,0));
```

Dado un año, top 5 de los patrocinadores que aportan más dinero a los equipos de cualquier categoría.

- Atributos: nombre del patrocinador (patrocinador), nombre del equipo (equipo), dinero aportado (importe).
- Creación tabla:

```
CREATE TABLE wrepo_top5_patro (
patrocinador VARCHAR2(25),
equipo VARCHAR2(50),
importe NUMBER(*,0));
```

Volumen de datos telemétricos guardados, indicar cuál es el equipo que ha recogido más datos telemétricos y el número de registros dado en un año concreto.

- Atributos: nombre del equipo (nombre), datos registrados por el equipo (total), total de datos de todos los equipos (datos_totales).
- Creación tabla:

```
CREATE TABLE wrepo_eq_datos_tele (
nombre VARCHAR2(50),
total NUMBER(*,0),
datos_totales  NUMBER(*,0));
```

Lista de los 5 fabricantes que aportan más componentes a la competición.

- Atributos: nombre del fabricante (fabricante), total de componentes aportados (total).
- Creación tabla:

```
CREATE TABLE wrepo_top5_fabr (
fabricante VARCHAR2(50),
total NUMBER(*,0));
```

Top 10 de los pilotos que han ganado más carreras entre todos los datos guardados.

- Atributos: código de licencia del piloto(licencia), nombre del piloto (nombre), carreras terminadas (carreras).
- Creación tabla:

```
CREATE TABLE wrepo_pilo_carr_term (
licencia VARCHAR2(20),
nombre VARCHAR2(50),
carreras_terminadas number(*,0));
```

Datos del piloto que ha tenido una mejor evolución entre dos momentos de tiempo indicados.

- Atributos: nombre del piloto (nombre), diferencia de puntos (evolución).
- Creación tabla:

```
CREATE TABLE wrepo_evo_piloto (
nombre VARCHAR2(50),
evolucion number(*,0));
```

Coche que ha consumido más combustible en una competición y año concreto, indicar los datos del coche y los litros de combustible consumidos.

- Atributos: código del coche (cod_coche), litros de combustible consumidos (litros_combustible).
- Creación tabla:

```
CREATE TABLE wrepo_coche_combus (
cod_coche VARCHAR2(10),
litros_combustible number(*,0));
```

2.8.2 ACTUALIZACIÓN DE LAS TABLAS DEL REPOSITORIO

En los procedimientos que listan un número determinado de registros, se puede poner una variable si estos van a cambiar, ningún procedimiento tiene esta variable.

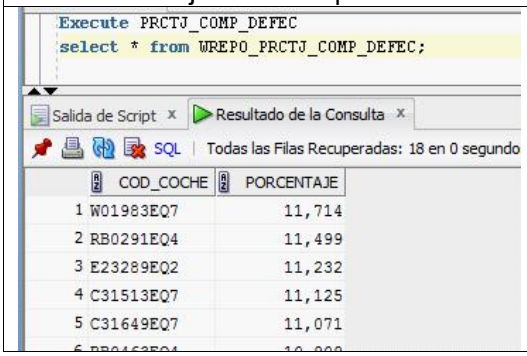
En el anexo 1 se explica cuál ha sido el proceso para seleccionar los datos de una tabla que después van a una tabla del repositorio estadístico.

Porcentaje de componentes defectuosos.

- Primero borramos los datos de la tabla para insertar los datos.
- Procedimiento creado:

```
create or replace procedure PRCTJ_COMP_DEFEC
            as
            begin
   delete from WREPO_PRCTJ_COMP_DEFEC;
            insert into WREPO_PRCTJ_COMP_DEFEC
            (COD_COCHE,PORCENTAJE)
   select * from(
                        select R3.cod_coche,round((malos/(buenos+malos))*100,3) as porcentaje from
                        (select R1.cod_coche,count(*) as malos from
                        (select cic,componente.cod_comp,cod_coche,mini,maxi from componente inner join
equipamiento
                        on componente.cod_comp=equipamiento.cod_comp)R1 inner join telemetria
                        on R1.cic=telemetria.cic
                        where valor < mini or valor >maxi
                        group by cod_coche)R2 inner join (select R1.cod_coche,count(*) as buenos from
                        (select cic,componente.cod_comp,cod_coche,mini,maxi from componente inner join
equipamiento
                        on componente.cod_comp=equipamiento.cod_comp)R1 inner join telemetria
                        on R1.cic=telemetria.cic
                        where valor > mini or valor <maxi
                        group by cod_coche) R3
                        on R2.cod_coche=R3.cod_coche
                        order by porcentaje desc);
 end;
```

Resultado de ejecución del procedimiento

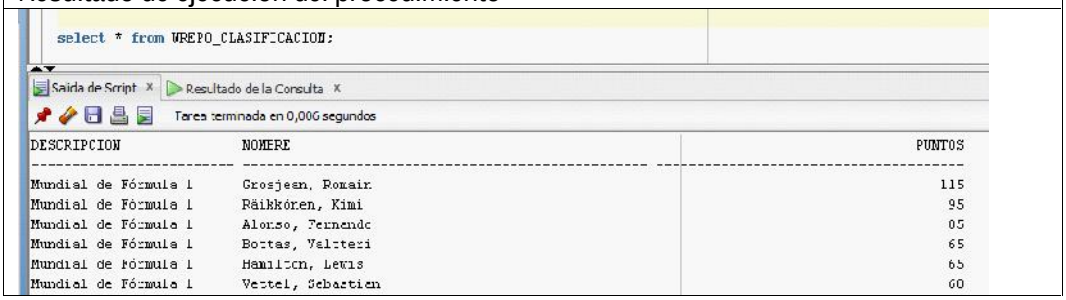


Resultado de cada piloto en cada competición disputada.

- Primero borramos los datos de la tabla para insertar los datos.
- Procedimiento creado:

```
create or replace procedure CLASIFICACION
        as
        begin
  delete from WREPO_CLASIFICACION;
        insert into WREPO_CLASIFICACION
        (DESCRIPCION,NOMBRE,PUNTOS)
  select * from(
                select descripcion,nombre,puntos from(
                select  nombre, cod_competicion, sum(ptos_obtenidos) puntos
                from (resultado inner join campeonato
                on resultado.cod_carrera=campeonato.cod_carrera) inner join
                piloto on resultado.cod_piloto=piloto.cod_licencia
                group by nombre,cod_competicion
                order by cod_competicion, puntos desc)res inner join competicion
                on res.cod_competicion=competicion.cod_competicion);
end;
```

Resultado de ejecución del procedimiento



```
select * from WREPO_CLASIFICACION;
```

Saida de Script | Resultado de la Consulta

Tarea terminada en 0,006 segundos

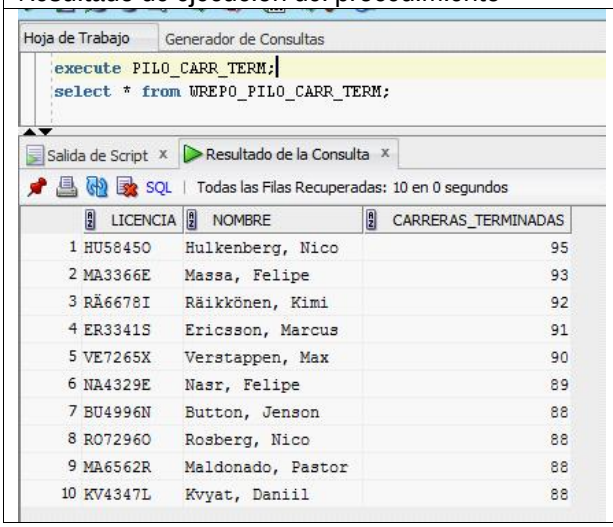
DESCRIPCION	NOMBRE	PUNTOS
Mundial de Fórmula 1	Grosjean, Romain	115
Mundial de Fórmula 1	Räikkönen, Kimi	95
Mundial de Fórmula 1	Alonso, Fernando	85
Mundial de Fórmula 1	Bottas, Valtteri	65
Mundial de Fórmula 1	Hamilton, Lewis	65
Mundial de Fórmula 1	Vettel, Sebastian	60

Listado de los 10 pilotos que han terminado más carreras de todos los datos registrados.

- Como sólo tenemos que tener 10 pilotos, primero borramos los datos de la tabla para insertar los datos.
- Procedimiento creado:

```
create or replace procedure PILO_CARR_TERM
        as
        begin
  delete from WREPO_PILO_CARR_TERM;
        insert into WREPO_PILO_CARR_TERM
        (LICENCIA,NOMBRE,CARRERAS_TERMINADAS)
  select * from(
                select * from (
                select resultado.cod_piloto, piloto.nombre,count(tiempo_carrera)  carreras_terminadas
                from resultado inner join piloto
                on resultado.cod_piloto=piloto.cod_licencia
                group by cod_piloto,nombre
                order by carreras_terminadas desc)
                where rownum<11);
end;
```

Resultado de ejecución del procedimiento



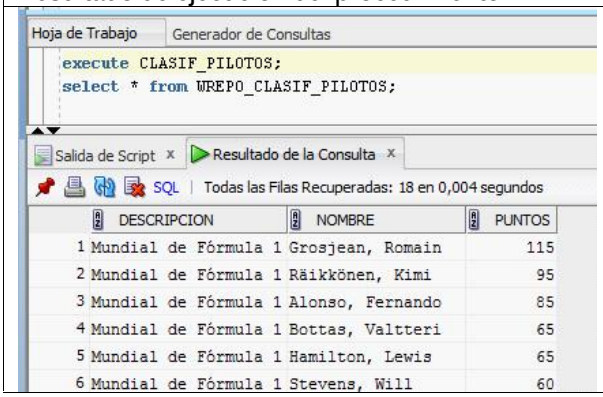
	LICENCIA	NOMBRE	CARRERAS_TERMINADAS
1	HU58450	Hulkenberg, Nico	95
2	MA3366E	Massa, Felipe	93
3	RÄ6678I	Räikkönen, Kimi	92
4	ER3341S	Ericsson, Marcus	91
5	VE7265X	Verstappen, Max	90
6	NA4329E	Nasr, Felipe	89
7	BU4996N	Button, Jenson	88
8	RO72960	Rosberg, Nico	88
9	MA6562R	Maldonado, Pastor	88
10	KV4347L	Kvyat, Daniil	88

Clasificación de cada competición –suma de puntos de los pilotos-.

- Se da el resultado de las competiciones en el año en curso.
- Procedimiento creado:

```
create or replace procedure CLASIF_PILOTOS
            as
            begin
  delete from WREPO_CLASIF_PILOTOS;
            insert into WREPO_CLASIF_PILOTOS
            (NOMBRE,COMPETICION,PUNTOS)
  select * from(
                  select descripcion,nombre,sum(ptos_obtenidos)puntos from piloto inner join
                  (select * from
                  resultado inner join
                  (select * from
                  campeonato inner join
                  (select * from competicion
                  where to_char(fecha_inicio,'YYYY')=to_char(sysdate,'YYYY'))comp
                  on comp.cod_competicion=campeonato.cod_competicion)carreras_curso_actual
                  on carreras_curso_actual.cod_carrera=resultado.cod_carrera) resul
                  on resul.cod_piloto=piloto.cod_licencia
                  group by descripcion,nombre
                  order by descripcion,puntos desc);
end;
```

Resultado de ejecución del procedimiento



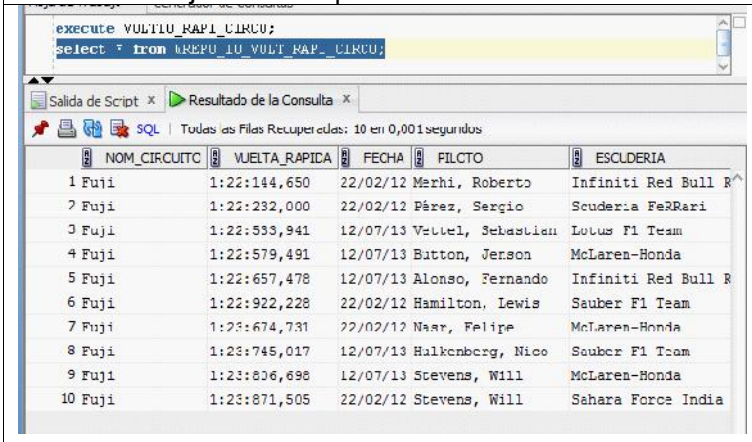
	DESCRIPCION	NOMBRE	PUNTOS
1	Mundial de Fórmula 1	Grosjean, Romain	115
2	Mundial de Fórmula 1	Räikkönen, Kimi	95
3	Mundial de Fórmula 1	Alonso, Fernando	85
4	Mundial de Fórmula 1	Bottas, Valtteri	65
5	Mundial de Fórmula 1	Hamilton, Lewis	65
6	Mundial de Fórmula 1	Stevens, Will	60

Dado un circuito, las 10 vueltas más rápidas que se han dado en él –se ha de indicar el tiempo de la vuelta, el piloto, la fecha, y el equipo.

- Para seleccionar el circuito, se declara la variable circu que contendrá el nombre del circuito, en este caso el circuito es el de Fuji. Antes de introducir los datos se borra la tabla.
- Procedimiento creado:

```
create or replace procedure VULT10_RAPI_CIRCU
        as
  circu VARCHAR2(25);
        begin
  circu:='Fuji';
  delete from WREPO_10_VULT_RAPI_CIRCU;
        insert into WREPO_10_VULT_RAPI_CIRCU
        (NOM_CIRCUITO,VUELTA_RAPIDA,FECHA,PILOTO,ESCUDERIA)
  select * from(
                select nomCircuito, vuelta_rapida,fecha,nombre_piloto,nombre as nombre_escuderia from
                (select nomCircuito, vuelta_rapida,fecha,nombre as nombre_piloto,cod_equipo from
                (select nomCircuito,cod_piloto,fecha,vuelta_rapida,cod_equipo from
                (select * from
                (select nomCircuito,codCarrera,cod_piloto,fecha,vuelta_rapida from
                (select nomCircuito,codCarrera,fecha from
                (select nombre as nomCircuito, cod_carrera as codCarrera from
                historial_circuito inner join circuito
                on historial_circuito.cod_circuito=circuito.cod_circuito
                where nombre=circu) carrerasCircuito inner join carrera
                on carrerasCircuito.codCarrera=carrera.cod_carrera) inner join resultado
                on resultado.cod_carrera=codCarrera
                order by vuelta_rapida asc)
                where rownum < 11) inner join historial
                on codCarrera=historial.cod_carrera
                where cod_piloto=historial.cod_licencia
                order by vuelta_rapida asc) inner join piloto
                on cod_piloto=piloto.cod_licencia)RF inner join equipo
                on RF.cod_equipo=equipo.cod_equipo
                order by vuelta_rapida asc);
end;
```

Resultado de ejecución del procedimiento



	NOM_CIRCUITO	VUELTA_RAPIDA	FECHA	PILOTO	ESCUDERIA
1	Fuji	1:22:144,650	22/02/12	Merhi, Roberto	Infiniti Red Bull R
2	Fuji	1:22:232,000	22/02/12	Pérez, Sergio	Scuderia FeRRari
3	Fuji	1:22:533,941	12/07/13	Vettel, Sebastian	Lotus F1 Team
4	Fuji	1:22:579,491	12/07/13	Button, Jenson	McLaren-Honda
5	Fuji	1:22:657,478	12/07/13	Alonso, Fernando	Infiniti Red Bull R
6	Fuji	1:22:922,228	22/02/12	Hamilton, Lewis	Sauber F1 Team
7	Fuji	1:23:674,731	22/02/12	Nasr, Felipe	McLaren-Honda
8	Fuji	1:23:745,017	12/07/13	Hulkenberg, Nico	Sauber F1 Team
9	Fuji	1:23:806,698	12/07/13	Stevens, Will	McLaren-Honda
10	Fuji	1:23:871,505	22/02/12	Stevens, Will	Sahara Force India

Temperatura más alta registrada en el coche, indicar la fecha, el circuito, el coche, el piloto y el valor.

- Como sólo tenemos que tener 5 patrocinadores, primero borramos los datos de la tabla para insertar los datos.
- Procedimiento creado:

```
create or replace procedure EQ_DATOS_TELE
```

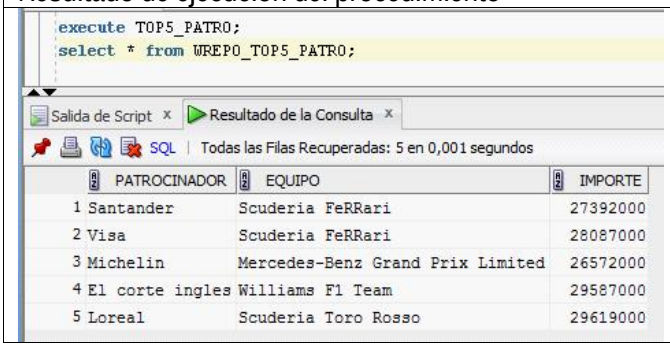
Resultado de ejecución del procedimiento

Dado un año, top 5 de los patrocinadores que aportan más dinero a los equipos de cualquier categoría.

- Como sólo tenemos que tener 5 patrocinadores, primero borramos los datos de la tabla para insertar los datos.
- Procedimiento creado:

```
create or replace procedure TOP5_PATRO
            as
            begin
  delete from WREPO_TOP5_PATRO;
            insert into WREPO_TOP5_PATRO
            (PATROCINADOR,EQUIPO,IMPORTE)
  select * from(
                        select patrocinador,nombre as equipo, participacion*1000 as euros from
                        (select nombre as patrocinador,cod_equipo, participacion from
                        (select * from
                        (select cod_patrocinador,cod_equipo,participacion from sponsor inner join competicion
                        on sponsor.cod_competicion=competicion.cod_competicion
                        where TO_CHAR(fecha_final,'yyyy')='2010'
                        order by participacion desc)
                        where rownum  <6)part inner join patrocinador
                        on part.cod_patrocinador=patrocinador.cod_patrocinador)patron inner join equipo
                        on patron.cod_equipo=equipo.cod_equipo);
end;
```

Resultado de ejecución del procedimiento



	PATROCINADOR	EQUIPO	IMPORTE
1	Santander	Scuderia FeRRari	27392000
2	Visa	Scuderia FeRRari	28087000
3	Michelin	Mercedes-Benz Grand Prix Limited	26572000
4	El corte ingles	Williams F1 Team	29587000
5	Loreal	Scuderia Toro Rosso	29619000

Volumen de datos telemétricos guardados, indicar cuál es el equipo que ha recogido más datos telemétricos y el número de registros dado en un año concreto.

- Como sólo tenemos que tener 1 equipo, primero borramos los datos de la tabla para insertar los datos.
- Procedimiento creado:

```
create or replace procedure EQ_DATOS_TELE
        as
        begin
  delete from WREPO_EQ_DATOS_TELE;
        insert into WREPO_EQ_DATOS_TELE
        (NOMBRE,TOTAL,DATOS_TOTALES)
  select * from(
                select nombre as equipo,total,total_datos from
                (select distinct cod_equipo,total,total_datos from
                (select * from
                (select * from
                (select cod_coche,count(valor) as total from
                (select cic,valor from telemetria
                where substr(momento,7,4)='2010'
                group by cic,valor) tele inner join equipamiento
                on tele.cic=equipamiento.cic
                group by cod_coche
                order by total desc)
                where rownum<2) natural join
                (select count(valor)as total_datos from telemetria
                where substr(momento,7,4)='2010'))datos inner join historial
                on datos.cod_coche=historial.cod_coche)eq inner join equipo
                on eq.cod_equipo=equipo.cod_equipo);
end;
```

Resultado de ejecución del procedimiento

Lista de los 5 fabricantes que aportan más componentes a la competición.

- Como sólo tenemos que tener el top 5 de los fabricantes, primero borramos los datos de la tabla para insertar los datos.
- Procedimiento creado:

```
create or replace procedure TOP5_FABR
        as
        begin
  delete from wrepo_top5_fabr;
        insert into WREPO_TOP5_FABR
        (fabricante,total)
  select * from(
                select * from
                (select fabricante,count(*)as total from recambio inner join equipamiento
                on recambio.identificador=equipamiento.cic
                where fabricante <>'Escuderia'
                group by fabricante
```

```
                order by total desc)
                where rownum<6);
end;
```

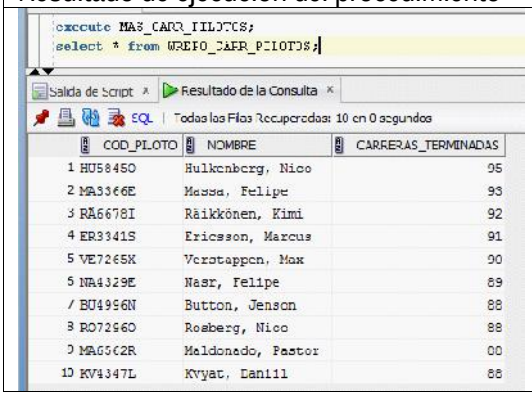
Resultado de ejecución del procedimiento

Top 10 de los pilotos que han ganado más carreras entre todos los datos guardados.

- Como sólo tenemos que tener el top 10 de los pilotos, primero borramos los datos de la tabla para insertar los datos.
- Procedimiento creado:

```
create or replace procedure MAS_CARR_PILOTOS
        as
        begin
  delete from WREPO_CARR_PILOTOS;
        insert into WREPO_CARR_PILOTOS
        (COD_PILOTO,NOMBRE,CARRERAS_TERMINADAS)
  select * from(
                select * from (
                select resultado.cod_piloto, piloto.nombre, count(tiempo_carrera)  carreras_terminadas
                from resultado inner join piloto
                on resultado.cod_piloto=piloto.cod_licencia
                group by cod_piloto,nombre
                order by carreras_terminadas desc)
                where rownum<11);
end;
```

Resultado de ejecución del procedimiento

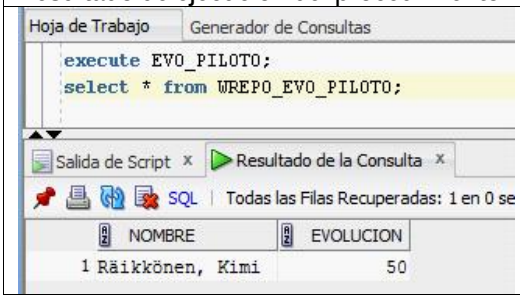


Datos del piloto que ha tenido una mejor evolución entre dos momentos de tiempo indicados.

- Como sólo tenemos que tener 1 piloto, primero borramos los datos de la tabla para insertar los datos.
- Procedimiento creado:

```
create or replace procedure EVO_PILOTO
            as
            begin
   delete from WREPO_EVO_PILOTO;
            insert into WREPO_EVO_PILOTO
            (NOMBRE,EVOLUCION)
   select * from(
   select * from(
                        select nombre,evolucion from
                        (select ptos1.cod_piloto,puntos2-puntos1 as evolucion from
                        (select cod_piloto,sum(resultado.ptos_obtenidos) as puntos1 from resultado inner join
carrera
                        on resultado.cod_carrera=carrera.cod_carrera
                        where carrera.fecha<'5/7/2010' and carrera.fecha>'1/1/2010'
                        group by cod_piloto)ptos1

                        inner join

                        (select cod_piloto,sum(resultado.ptos_obtenidos) as puntos2 from resultado inner join
carrera
                        on resultado.cod_carrera=carrera.cod_carrera
                        where carrera.fecha<'5/11/2010' and carrera.fecha>'1/1/2010'
                        group by cod_piloto)ptos2

                        on ptos1.cod_piloto=ptos2.cod_piloto)evo inner join piloto
                        on evo.cod_piloto = piloto.cod_licencia
                        order by evolucion desc)
                        where rownum <2);
 end;
```

Resultado de ejecución del procedimiento



Coche que ha consumido más combustible en una competición y año concreto, indicar los datos del coche y los litros de combustible consumidos.

- Como sólo tenemos que tener 1 piloto, primero borramos los datos de la tabla para insertar los datos.
- Procedimiento creado:

```
create or replace procedure COCHE_COMBUS
            as
 ano integer;
 competicio VARCHAR2(25);
            begin
  delete from WREPO_COCHE_COMBUS;
            insert into WREPO_COCHE_COMBUS
            (COD_COCHE,LITROS_COMBUSTIBLE)
  select * from(
                        select cod_coche,litros_combustible from
                        (select distinct cod_piloto,cod_coche,sum(consumo)as litros_combustible from
                        (select carr.cod_carrera,cod_piloto,consumo from
                        (select cod_carrera from
                        (select cod_competicion
                        from competicion
                        where descripcion=competicio and to_char(fecha_final,'yyyy')=ano) co
                        inner join campeonato
                        on co.cod_competicion=campeonato.cod_competicion)carr
                        inner join resultado
                        on carr.cod_carrera=resultado.cod_carrera) pi_car
                        inner join historial
                        on historial.cod_carrera=pi_car.cod_carrera
                        group by cod_coche,cod_piloto
                        order by litros_combustible desc)
                        where rownum <2);
end;
```

Resultado de ejecución del procedimiento

3. CONCLUSIONES

- La mayor parte del tiempo la he pasado aprendiendo a cómo se maneja el lenguaje SQL, debido a que había muchas cosas que ya no recordaba y el programa de Oracle.

- El manejo del tiempo para el proyecto, yo creía que no sería difícil encontrar tiempo para trabajar en el proyecto, menos mal que sólo contabilice los días laborables como días de trabajo y dejar el fin de semana para trabajar si fuese necesario.

- El ritmo del proyecto lo he podido lograr mantener, y la mayor parte del tiempo iba un poco por delante de lo planificado.

- A pesar de muy bien que se planifica algo, siempre hay algo que te hace reacondicionar el proyecto desde el principio, eso me ha pasado al no declarar una tabla que después me hizo falta para que el diseño de la base de datos estuviese normalizada.

- No ha sido nada fácil desarrollar el proyecto, yo pensaba que sería más sencillo.

- Una cosa importante es que antes de dedicar tiempo a desarrollar el proyecto hay que emplear tiempo a conocer y aprender el fucionamiento del entorno de trabajo (Office, Oracle SQL, el lenguaje SQL,etc…).

- Me ha sorprendido mucho que no he necesitado ayuda del tutor para el desarrollo del proyecto, lo cual me sorprende mucho.

4. GLOSARIO

No he considerado palabras o términos relevantes.

5. ANEXO 1

En este anexo se va a explicar cuál ha sido el procedimiento para obtener los datos de las consultas a las distintas tablas de la base de datos, que después van a una tabla del repositorio.

Consideremos que queremos obtener los cinco patrocinadores que más dinero han aportado a los equipos en cualquier categoría en un año concreto.

Código que se ejecuta:

```
select patrocinador,nombre as equipo, participacion*1000 as euros from
        (select nombre as patrocinador,cod_equipo, participacion from
                (select * from
                        (select cod_patrocinador,cod_equipo,participacion from sponsor inner join competición
                        on sponsor.cod_competicion=competicion.cod_competicion
                        where TO_CHAR(fecha_final,'yyyy')='2010'
                        order by participacion desc)
                where rownum  <6)part
        inner join patrocinador
        on part.cod_patrocinador=patrocinador.cod_patrocinador)patrón
 inner join equipo
on patron.cod_equipo=equipo.cod_equipo);
```

Consideraciones:

- Cada categoría tiene distintas competiciones, como no se indica que categoría concreta se quiere no hace falta usar esta tabla, en caso de querer una competición concreta ya relacionaríamos las competicones con las categorías.

Tablas implicadas:

- Sponsor. Relaciona las competiciones, los equipos, los patrocinadores y la cantidad de dinero aportada.
- Competicion. Porque hay que relacionar todos los sponsors de todas las categorías.
- Patrocinador. Para obtener los datos del patrocinador (nombre, marca, etc.).
- Equipo. Para obtener el nombre del equipo.

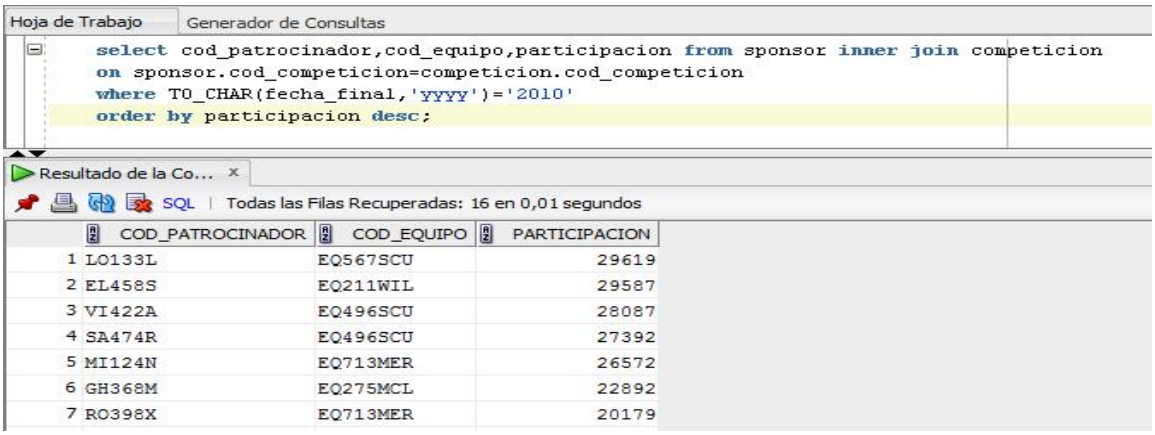
Criterios para obtener los datos:

- Lo primero es seleccionar de entrada los datos mínimos para que la selección no se haga muy grande cuando se haga un inner join.

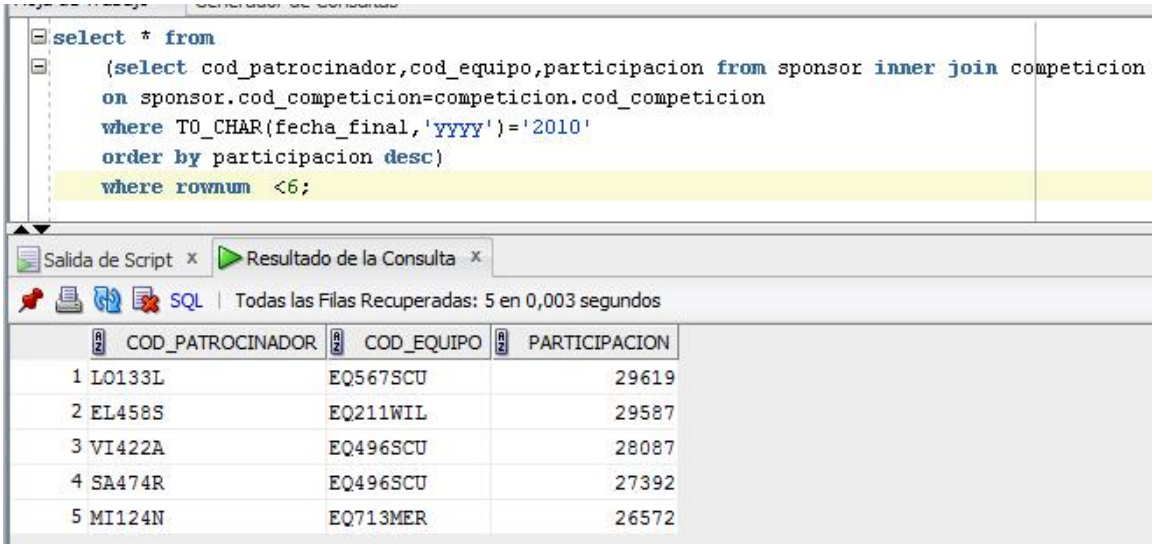
Proceso de selección de los datos de las tablas.

- Como lo que queremos es obtener los 5 patrocinadores que aportan más dinero, es ir a la tabla de sponsor y seleccionar los 5 que más dinero aportan, seleccionando los atributos (cod_patrocinador, cod_equipo y participación)

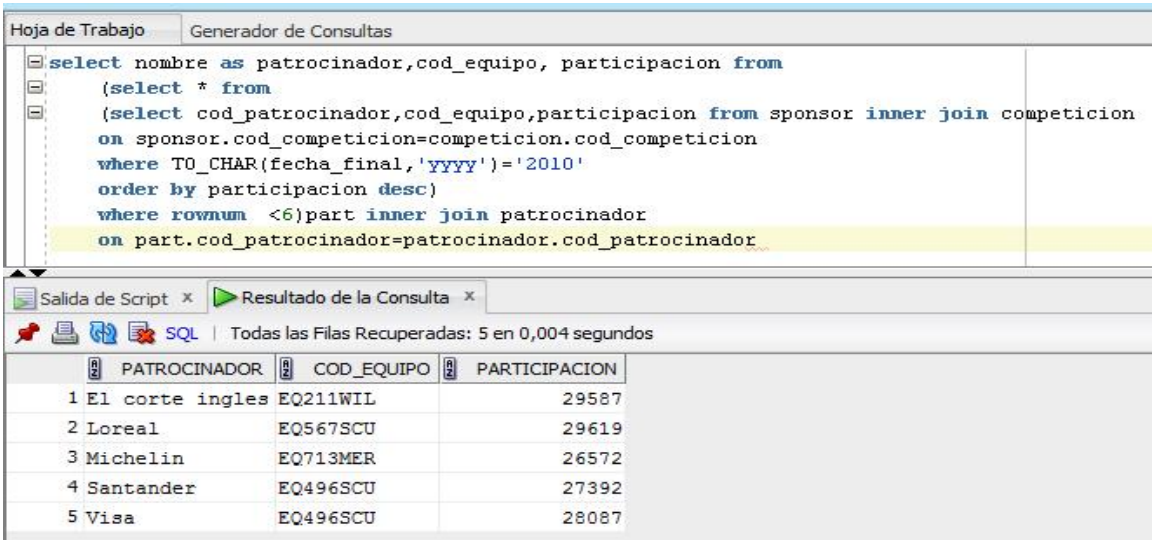
- Para seleccionar las 5 aportaciones mayores, hago un select de la participación en orden descendente, de esta manera la mayor participación está en el row 1.
  Código ejecutado y resultado:



- Para escoger los 5 que más aportan sólo hago un rownum <6.



- Ahora que ya tenemos los datos mínimos necesarios, sólo nos queda hacer un inner join con las tablas del patrocinador y del equipo para saber los nombres de ambos y no los códigos que tienen dentro de la FIA.
- Primero se hace un inner join con la tabla patrocinador.



- Por último se hace un inner join con la tabla equipo.

Hoja de Trabajo | Generador de Consultas

```
select patrocinador,nombre as equipo, participacion*1000 as euros from
  (select nombre as patrocinador,cod_equipo, participacion from
    (select * from
    (select cod_patrocinador,cod_equipo,participacion from sponsor inner join competicion
    on sponsor.cod_competicion=competicion.cod_competicion
    where TO_CHAR(fecha_final,'yyyy')='2010'
    order by participacion desc)
    where rownum  <6)part inner join patrocinador
    on part.cod_patrocinador=patrocinador.cod_patrocinador)patron inner join equipo
    on patron.cod_equipo=equipo.cod_equipo;
```

Salida de Script | Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 5 en 0,005 segundos

	PATROCINADOR	EQUIPO	EUROS
1	El corte ingles	Williams F1 Team	29587000
2	Loreal	Scuderia Toro Rosso	29619000
3	Michelin	Mercedes-Benz Grand Prix Limited	26572000
4	Santander	Scuderia FeRRari	27392000
5	Visa	Scuderia FeRRari	28087000

Con este mismo criterio de seleccionar en primer lugar los mínimos datos posibles se han diseñado el resto de consultas para el repositorio estadístico.

6. BIBLIOGRAFIA

[1] Federación Internacional del Automóvil. 2015. http://www.fia.com/

[2] Calendario oficial de la FIA 2015. http://www.fia.com/sites/default/files/basicpage/file/20150316/2015%20FullCalendar.pdf

[3] Bases de datos de fórmula 1. http://www.formulaf1.es/foros/tema/base-de-datos/

[4] Circuitos de fórmula 1. http://es.wikipedia.org/wiki/Anexo:Circuitos_de_F%C3%B3rmula_1

[5] Categorías. http://es.wikipedia.org/wiki/Automovilismo

[6] Importar datos a tabla Oracle. http://www.forosdelweb.com/f100/importar-datos-tabla-oracle-330009/

[7] Base de datos de pilotos. http://www.palomatica.info/luis/tutorialSQL/principal.htm

[8] Consulta Oracle. http://www.ajpdsoft.com/modules.php?name=news&file=article&sid=131

[9] Base de datos creada. http://www.palomatica.info/luis/tutorialSQL/principal.htm

[10] Circuitos de fórmula 1. http://www.formulaf1.es/circuito/spa-francorchamps-belgica/

[11] Circuitos de fórmula 1. http://es.wikipedia.org/wiki/Anexo:Circuitos_de_F%C3%B3rmula_1

[12] Escuderías F1. http://formula1.autobild.es/escuderias/scuderia-ferrari

[13] Autobid.es F1. 2015. http://formula1.autobild.es/escuderias

[14] Autobid.es F1. 2015. http://formula1.autobild.es/pilotos

[15] Autobid.es F1. 2015. http://formula1.autobild.es/escuderias

[16] Circuitos de fórmula 1. http://www.formulaf1.es/circuito/spa-francorchamps-belgica/

[17] Circuitos de fórmula 1. http://es.wikipedia.org/wiki/Anexo:Circuitos_de_F%C3%B3rmula_1

[18] El árbol de levas.2015. http://levasybalancines.blogspot.com.es/