Universidad de las Ciencias Informáticas
Facultad 8

Base de Datos de Gestor de Contenidos de Ajedrez

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Autores: Alexeider Rodríguez Rojas

Yaniel Pérez Pérez

Tutor: Ing. Abdanys Arias Suárez

Ciudad de La Habana, Junio 2008
"Año 50 de la Revolución"

La calidad nunca es un accidente; siempre es el resultado de un esfuerzo de la inteligencia.

John Ruskin

Declaración de autoría

Declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _2008.

_____ _____

Alexeider Rodríguez Rojas (Autor) Yaniel Pérez Pérez (Autor)

_____

Ing. Abdanys Arias Suárez (Tutor)

Agradecimientos

A mi papá y Judith por la ayuda y el apoyo que me han brindado y la confianza que han tenido en mí.

A mis abuelos por su cariño y educación que me han dado, así como a toda mi familia mencionando especialmente a mis tíos que son como mis hermanos Amauris, Aramis y Aracelis por su preocupación.

A mis mejores amigos: Emilio, Deiler, Yanirys y Dany por estar ahí cuando los necesitaba.

A todos aquellos que de una forma u otra han contribuido en mi formación.

Alexeider

A mi mamá que sin ella nuca hubiese sido posible su esfuerzo fue mi mayor fuerza.

A mi hermano por pensar siempre en mí y apoyarme en todo.

A mi abuelita que siempre estuvo presente en toda mi vida de estudiante y fue mi segunda mamá.

A mi abuelo que ayudo con todo sus esfuerzos a mi carrera y con su apoyo a mi mamá.

A mi abuelita Inés que siempre estuvo preocupándose por mi mamá y mi carrera.

A toda mi familia.

A Hilda y Lolo que ayudaron y cuidaron de mi mamá cuando estaba yo lejos.

A todos los que de una forma u otra ayudaron a mi formación.

Yaniel

A la Revolución y la Universidad de las Ciencias Informáticas.

Dedicatoria

Dedico este trabajo como representación de todos estos años de estudios:

A la memoria de mi madre.

A mi papá y Judith por ser mi principal apoyo.

A mis abuelos por su preocupación en mi educación y formación.

A mis hermanos, Alexito y Javier por ser parte del objetivo de mi esfuerzo.

Alexeider

A la memoria de mi papá que siempre fue mi ejemplo y mi guía.

Dedicado a lo mas especial en mi vida, mi mamá, que es el ejemplo más grande de entrega, fidelidad y amor hacia un hijo que he conocido.

A mi hermano que no me alcanza el corazón para quererlo, siempre estuvo a mi lado y fue como un padre.

Yaniel

Resumen

El ajedrez es un deporte que ha tomado gran auge en nuestro país y la Universidad de las Ciencias Informáticas (UCI) no se quiere quedar atrás en este tema, por lo que se ha trazado la tarea a través del proyecto INFODREZ de realizar un Gestor de Contenidos(CMS) capaz de gestionar la información necesaria respecto al tema del ajedrez. Es ahí donde toma partida el objetivo de este trabajo, dada la necesidad de almacenar toda esta información.

Este trabajo presenta una descripción de los pasos necesarios y las herramientas empleadas para la confección de la Base de Datos que brindará servicios a los módulos implementados hasta el momento, los cuales son: Ajedrez por Correspondencia, Estadística, Torneos, Torneo Online, Arbitraje y Cálculo de Elo.

En el presente documento se exponen una serie de aspectos relacionados con el tema de Base de Datos, que sirven de punto de partida para la selección de los elementos que conforman la propuesta. Así como su descripción, fundamentación y validación.

Índice de Contenido

Índice de Tablas

Índice de Figuras

Introducción

El ajedrez es un deporte milenario, y desde aquellos tiempos se ha tratado de almacenar la mayor cantidad de datos posibles para su análisis y con esto obtener frutos en el juego así como para estudiar el juego y a los contrarios.

Con el paso del tiempo la cantidad de jugadores y partidas son muchas y esto hace más importante el almacenar de la mejor forma los datos relacionados al ajedrez. Como deporte ciencia relaciona lo científico con lo competitivo y esto hace que sea necesaria la creación de un sistema que ayuden al proceso de almacenamiento de datos.

Como nuestro país y gobierno siempre han estado al lado del deporte sano y puro se ha dado a la tarea desde hace ya unos años de fomentar la práctica del ajedrez en nuestros centros de estudio y trabajo, con este fin se ha creado un proyecto en la Universidad de las Ciencias Informáticas para facilitar el conocimiento de este tema, así como la práctica de dicho deporte, cuyo nombre es: INFODREZ, a través de módulos y con una base de datos amplia que facilitará el estudio del ajedrez.

La importancia de un trabajo de este tipo es muy grande ya que se tendría almacenado una gran cantidad de partidas en las cuales se podría saber cual es la apertura más usada o los errores más cometidos por los jugadores, también sería de suma importancia el manejo de estructuras de torneos con esto sería más fácil cerciorarse de cual es más acorde con el torneo que de desearía realizar dependiendo de la cantidad de jugadores, tiempo de duración y otros aspectos vitales a la hora de definir este tema. Se tendría bien almacenados los Elos de los jugadores cubanos así como sus participaciones y resultados.

Por todo lo antes expuesto, se plantea como **problema científico** de la investigación: ¿Cómo realizar un óptimo diseño lógico y físico de la base de datos para el Gestor de Contenidos de Ajedrez?

Por lo cual el **objeto de estudio** de esta investigación es el proceso de almacenamiento y diseño lógico y físico de Bases de Datos para un Gestor de Contenidos.

De ello se deriva que el **campo de acción** que abarca este trabajo, es el diseño lógico y físico de la base de datos relacional para el Gestor de Contenidos de Ajedrez.

Por lo que el **objetivo general** consiste en realizar la base de datos para brindar servicios a los módulos de Ajedrez por Correspondencia, Estadística, Torneos, Torneo Online, Arbitraje y Cálculo de Elo.

Teniendo como **idea a defender**:

Si se realiza el diseño e implementación de la base de datos se podrá dar servicio a los módulos de Ajedrez por Correspondencia, Estadística, Torneos, Torneo Online, Arbitraje y Cálculo de Elo.

Para dar cumplimiento al principal objetivo de esta investigación se han planteado una serie de **tareas** a cumplir, las cuales son:

- Elaborar el diseño teórico de la investigación.
- Elaborar la fundamentación teórica de la investigación.
- Estudiar diseño de la base de datos de algunos CMS (Drupal, Joomla, E107, Plone).
- Definir elementos que puedan aportar al diseño de la base de datos del CMS de Ajedrez.
- Realizar el diseño lógico del Gestor de Contenidos de Ajedrez.
- Realizar el diseño físico del los módulos que se desarrollaran este curso en el Gestor de Contenidos de Ajedrez.
- Probar integridad de los datos en la base de datos.

El presente trabajo está estructurado de tres capítulos, los cuales están conformados por la siguiente información:

Capítulo 1. Fundamentación Teórica, contiene toda la información con respecto a la parte teórica de las bases de datos, como es el surgimiento y desarrollo de las mismas, así como sus principales componentes; se explican los diferentes tipos de base de datos por los diferentes criterios que existen. Además se describen los tipos de modelos para el diseño, se realiza un análisis y descripción de los diferentes Sistemas Gestores de Base de Datos, así como las metodologías y herramientas utilizadas para diseñar e implementar bases de datos.

Capítulo 2. Descripción y Análisis de la Solución Propuesta, se describe la solución propuesta, donde se trata acerca de la estrategia de integración de la solución con otros módulos o sistemas, la descripción de la arquitectura y fundamentación, haciendo énfasis en los requisitos funcionales y no funcionales, al igual que el diagrama de clases persistentes obtenido a partir del diagrama de clases

del diseño, extraídos a partir de la labor realizada por el analista del proyecto, el diagrama Entidad Relación de cada uno de los módulos, y la descripción de las tablas.

Capítulo 3. Validación del Diseño Realizado, se realiza la validación teórica del diseño que incluye análisis de integridad, normalización de las bases de datos, el análisis de redundancia de información y el análisis de la seguridad, así como la trazabilidad de las acciones realizadas en la base de datos.

En la parte final del documento se encuentra la bibliografía de apoyo para la realización de este trabajo, así como los anexos necesarios y un Glosario de Términos para una mejor comprensión del texto.

Capítulo 1.- Fundamentación Teórica.

Introducción

En el presente capítulo se realiza análisis respecto a las Bases de Datos (BD), su surgimiento y desarrollo, los diferentes tipos que existen actualmente según sus respectivas clasificaciones, los modelos definidos por forma de usabilidad así como los principales componentes. Además se hace un estudio de las principales tecnologías, metodologías, así como herramientas con especificidades como ventajas y desventajas para el desarrollo y tratamiento de las BD, así como de estrategias que se utilizan para el diseño.

1.1 Conceptos de Bases de Datos

Una BD es un conjunto de datos relacionados entre sí. Se entiende por *dato,* los hechos conocidos, que pueden registrarse y que tienen significado implícito. Una definición más rigurosa que la anterior es la siguiente:

Es un conjunto de datos lógicamente coherentes, con un cierto significado inherente. Una colección aleatoria de datos no puede considerarse propiamente una BD.

Otra podría ser:

Una BD se diseña, construye y puebla con datos para propósito específico. Está dirigida a un grupo de usuarios y tiene ciertas aplicaciones preconcebidas que interesan a distintos usuarios.

Constituyen una colección de datos interrelacionados que se puede utilizar por uno o más programas de aplicación, puede considerarse una colección de datos variables en el tiempo.

O sea, una BD tiene:

- Una fuente de la cual se derivan los datos.
- Cierto grado de interacción con los hechos del mundo real.

- Un público activamente interesado en el contenido de la BD.
- Su tamaño es variado.
- Debe ser posible buscar, obtener y actualizar los datos siempre que sea necesario.

1.2 Surgimiento y desarrollo de las Bases de Datos

La información era necesaria almacenarla para poder acceder a ella cuando se necesitara, por esta razón se crean los archivos secuenciales como almacenes de datos. Estos daban un acceso muy rápido pero sólo de forma secuencial (debía recorrer el archivo entero). Como solución aparecieron los archivos indexados, donde el acceso ya podía ser aleatorio (acceder de una vez a la posición deseada del archivo).

Por la complejidad de los programas y los volúmenes de datos que iban generando se requerían de condiciones de almacenamientos así como operaciones que facilitaran el trabajo con los datos. Debía existir una privacidad en las acciones de cada usuario que accediera a la BD es decir que las acciones de un usuario no interfirieran en las de otro usuario en dicha BD.

De esta forma surgen las BD, surgieron como primera topología la jerárquica donde los datos se situaban siguiendo una jerarquía. Las BD jerárquicas tenían el problema que los accesos a los datos eran unidireccionales, y era más complicado hacer el camino inverso (pero posible, aunque el tiempo de cálculo era mayor). Como variante a esta topología surgieron las BD de red, modelo ligeramente distinto del jerárquico, el cual permitía que un mismo nodo tenga varios padres, aún así no era una solución factible. Para dar absoluta libertad a las relaciones entre tablas surgieron las BD relacionales. Estas trajeron dos cosas muy importantes: las propiedades ACID y un lenguaje común de acceso a los datos: SQL.

Las propiedades ACID son:

- **Atomicidad.** Cada transacción del usuario debe tratarse de forma atómica. O se ejecuta todo o nada, es decir, el trabajo se realiza en su totalidad o no se realiza en ningún caso.

- **Consistencia.** Las transacciones han de cumplir las restricciones definidas dentro de la BD. Si no las pueden cumplir, se evita su ejecución. De esta forma se conserva la integridad y coherencia de los datos.
- **Aislamiento.** Una transacción es una unidad de aislamiento, permitiendo que transacciones concurrentes se comporten como si cada una fuera una única transacción que se ejecuta en el sistema. Las transacciones alcanzan el nivel más alto de aislamiento cuando se pueden serializar. En este nivel, los resultados obtenidos de un conjunto de transacciones concurrentes son idénticos a los obtenidos mediante la ejecución en serie de las transacciones.
- **Durabilidad**. Una vez se ha completado la transacción, los resultados de la misma han de ser permanentes y sobrevivir a posibles caídas del sistema o la BD.

1.3 Tipos de bases de datos

Las bases de datos pueden clasificarse de varias maneras, de acuerdo al criterio elegido:

1.3.1 Según la variabilidad de los datos almacenados

- **Bases de datos estáticas:**

Estas son bases de datos de sólo lectura, utilizadas primordialmente para almacenar datos históricos que posteriormente se pueden utilizar para estudiar el comportamiento de un conjunto de datos a través del tiempo, realizar proyecciones y tomar decisiones.

- **Bases de datos dinámicas:**

Estas son bases de datos donde la información almacenada se modifica con el tiempo, permitiendo operaciones como actualización y adición de datos, además de las operaciones fundamentales de consulta. Un ejemplo de esto puede ser la BD utilizada en un sistema de información de una tienda de abarrotes, una farmacia, un videoclub, o un negocio cualquiera.

1.3.2 Según el contenido

- **Bases de datos bibliográficas:**

Solo contienen un representante de la fuente primaria, que permite localizarla. Un registro típico de una BD bibliográfica contiene información sobre el autor, fecha de publicación, editorial, título, edición, de una determinada publicación. Puede contener un resumen o extracto de la publicación original, pero nunca el texto completo, porque sino se estaría en presencia de una BD a texto completo (o de fuentes primarias). Como su nombre lo indica, el contenido son cifras o números.

- **Bases de datos de texto completo:**

Almacenan las fuentes primarias, como por ejemplo, todo el contenido de todas las ediciones de una colección de revistas científicas.

- **Directorios:**

Un ejemplo son las guías telefónicas en formato electrónico.

1.4 Modelo para el diseño de Base de Datos

- **Jerárquicas**

Los datos se organizan en una forma similar a un árbol, donde un nodo padre de información puede tener varios hijos. El nodo que no tiene padres es llamado raíz, y a los nodos que no tienen hijos se llaman hojas. Este modelo no es utilizado porque una de sus principales limitaciones es: “la poca flexibilidad de este modelo puede obligar a la introducción de redundancia cuando es preciso instrumentar, mediante el modelo jerárquico, situaciones del mundo real que no responden a una jerarquía”. [1]

- **Red**

Modelo ligeramente distinto del jerárquico, su diferencia fundamental es la modificación del concepto de nodo; éste permite que un mismo nodo tenga varios padres.

Gran mejora con respecto al modelo jerárquico, ya que ofrecía una solución eficiente al problema de redundancia de datos; aún así, la dificultad que significa administrar la información en una BD de red ha significado que sea un modelo utilizado en su mayoría por programadores más que por usuarios finales, por lo que este modelo no se utiliza.

- **Orientado a objetos**

El modelo de objetos conjuga de forma centralizada los conceptos de abstracción, jerarquía, encapsulación, modularidad, persistencia, tipos y concurrencia, donde los primeros cuatro conceptos son fundamentales, lo cual significa que si un modelo carece de estos elementos pues no es orientado a objetos. Además se basa en los principios de la ingeniería. Este modelo refleja una nueva forma de pensar acerca de la de descomposición, haciendo que el diseño orientado a objetos sea diferente a la forma que existía de diseño estructurado hasta el momento, y que por consiguiente su arquitectura sea diferente también. Esto ocurre sencillamente porque los métodos de diseño orientado a objetos utilizan la programación orientada a objetos, mientras que el diseño estructurado se basa en la programación estructurada. El desarrollo del diseño orientado a objetos no niega las ideas existentes, sino que se basa en ellas y las mejora. Un modelo orientado a objetos tiene obligatoriamente que cumplir con lo siguiente: la estructura básica de trabajo son los objetos, no algoritmos; donde cada objeto no es más que una instancia de una clase ya definida y que dichas clases estarán relacionadas únicamente por relaciones de herencia. En caso de que no se cumpla uno de las condiciones antes mencionadas, pues no se estaría en presencia de un modelo orientado a objetos.

- **Relacional**

El modelo relacional de datos tiene varios aspectos que lo caracterizan, desde el punto de vista de estructural se puede plantear que este modelo organiza la información en forma de tablas y de esta manera es que sus usuarios la perciben y no de otra; desde el punto de vista de la integridad se puede decir que el modelo establece varias restricciones de integridad, las cuales deben ser cumplidas por todas sus tablas; y desde el punto de vista de la manipulación, el modelo relacional tiene definidos un grupo de operadores a través de los cuales interactúa con la información almacenada en las tablas de la BD y sus resultados los devuelven en tablas. Entre estos operadores se encuentran proyectar, restringir y juntar. Debido a que la información consultada en cada operación se encuentra en forma de tabla, al igual que la que devuelven, pues existe la posibilidad de concatenar operadores unos con otros. Este modelo además está capacitado para hacer procesamiento de conjunto. Esto significa que al devolver el resultado de cualquier operación, el resultado lo devuelve en tablas que están

conformadas por un conjunto de filas, no por filas individuales. El modelo relacional está compuesto por cinco componentes que lo identifican, los cuales se mencionan a continuación: un conjunto abierto de tipos escalares, un generador de tipos de relación y una interpretación propuesta de dichos tipos, herramientas para definir variables de dichos tipos de relación generados, una operación asignación relacional para asignar valores de relación a las variables de relación mencionadas, un conjunto abierto de operadores relacionales genéricos para derivar valores de relación de otros valores de relación.

1.5 Bases de datos y sus componentes principales

El volumen de información que puede almacenar una BD es muy variado, va desde una agenda, un contador, o un libro de visitas, hasta el de una tienda en línea, un sistema de noticias, un portal, o la información generada en una red corporativa.

Las BD son de vital importancia en muchos aspectos, contienen la información específica para poder llevar a cabo la estimación más exacta de situaciones por lo que ayuda a la toma de decisiones en empresas e instituciones. Estas están formadas por cuatro componentes principales: Datos, Hardware, Software y Usuarios.

Datos: se refiere a que los datos en la BD serán tanto integrados (unificación de varios archivos que de otro modo serían distintos) como compartidos (las piezas individuales de datos en la base pueden ser compartidas entre diferentes usuarios, con lo que cada uno puede acceder a la misma pieza de datos con fines diferentes).

Hardware: se refiere a los dispositivos de almacenamiento donde reside la BD, así como a los dispositivos periféricos (unidad de control, canales de comunicación) necesarios para su uso.

Software: constituido por un conjunto de programas que se conoce como Sistema Manejador de BD (DBMS: Data Base Management System). Este sistema maneja todas las solicitudes formuladas por los usuarios a la BD.

Usuarios: Existen tres clases de usuarios relacionados con una BD:

- El programador de aplicaciones, quien crea programas de aplicación que utiliza la BD.

- El usuario final, quien accede a la BD por medio de un lenguaje de consulta o de programas de aplicación.
- El administrador de la BD (DBA Data Base Administrator), quien se encarga del control general del Sistema de BD.

1.6 Análisis y descripción de los diversos Sistemas Gestores de Bases de Datos.

Los Sistemas Gestores de Bases de Datos (SGBD) son un tipo de software muy específico, dedicado a servir de interfaz entre las bases de datos y las aplicaciones que la utilizan. En la actualidad existe una gran variedad de SGBD, las principales funciones que debe cumplir se relacionan con la creación y mantenimiento de la BD, el control de accesos, la manipulación de datos de acuerdo con las necesidades del usuario, el cumplimiento de las normas de tratamiento de datos, así como evitar redundancias e inconsistencias y mantener la integridad.

1.6.1 Principales objetivos de un Sistema Gestor de Base de Datos.

- **Independencia de los datos y los programas de aplicación:**

Ya se pudo notar que con archivos tradicionales la lógica de la aplicación contempla la organización de los archivos y el método de acceso. Por ejemplo, si por razones de eficiencia se utiliza un archivo secuencial indexado, el programa de aplicación debe considerar la existencia de los índices y la secuencia del archivo. Entonces es imposible modificar la estructura de almacenamiento o la estrategia de acceso sin afectar el programa de aplicación (naturalmente, lo que se afecta en el programa son las partes de éste que tratan los archivos, lo que es ajeno al problema real que el programa de aplicación necesita resolver). En un SGBD sería indeseable la existencia de aplicaciones y datos dependientes entre sí, por dos razones fundamentales:

- Diferentes aplicaciones necesitarán diferentes aspectos de los mismos datos (por ejemplo, puede requerirse la representación decimal o binaria).

- Se debe poder modificar la estructura de almacenamiento o el método de acceso según los cambios en el fenómeno o proceso de la realidad sin necesidad de modificar los programas de aplicación (también para buscar mayor eficiencia).

La independencia de los datos se define como la inmunidad de las aplicaciones a los cambios en la estructura de almacenamiento y en la estrategia de acceso y constituye el objetivo fundamental de los SGBD.

- **Minimización de la redundancia.**

Con los ficheros tradicionales, se produce redundancia de la información. Uno de los objetivos de los SGBD es minimizar la redundancia de los datos. Se dice disminuir la redundancia, no eliminarla, pues, aunque se definen las BD como no redundantes, en realidad existe redundancia en un grado no significativo para disminuir el tiempo de acceso a los datos o para simplificar el método de direccionado. Lo que se trata de lograr es la eliminación de la redundancia superflua.

- **Integración y sincronización de las bases de datos.**

La integración consiste en garantizar una respuesta a los requerimientos de diferentes aspectos de los mismos datos por diferentes usuarios, de forma que, aunque el sistema almacene la información con cierta estructura y cierto tipo de representación, debe garantizar entregar al programa de aplicación los datos que solicita y en la forma en que lo solicita.

Está vinculada a la sincronización, que consiste en la necesidad de garantizar el acceso múltiple y simultáneo a la BD, de modo que los datos puedan ser compartidos por diferentes usuarios a la vez. Están relacionadas, ya que lo usual es que diferentes usuarios trabajen con diferentes enfoques y requieran los mismos datos, pero desde diferentes puntos de vista.

- **Integridad de los datos.**

Consiste en garantizar la no contradicción entre los datos almacenados de modo que, en cualquier momento del tiempo, los datos almacenados sean correctos, es decir, que no se detecte inconsistencia entre los datos. Está relacionada con la minimización de la redundancia, ya que es más fácil garantizar la integridad si se elimina la redundancia.

- **Seguridad y recuperación.**

Seguridad (también llamada protección): garantizar el acceso autorizado a los datos, de forma de interrumpir cualquier intento de acceso no autorizado, ya sea por error del usuario o por mala intención.

Recuperación: que el SGBD disponga de métodos que garanticen la restauración de las bases de datos al producirse alguna falla técnica, interrupción de la energía eléctrica.

- **Facilidad de manipulación de la información**

Los usuarios de una BD pueden acceder a la misma con solicitudes para resolver muchos problemas diferentes. El SGBD debe contar con la capacidad de una búsqueda rápida por diferentes criterios, permitir que los usuarios planteen sus demandas de una forma simple, aislándolo de las complejidades del tratamiento de los archivos y del direccionamiento de los datos. Los SGBD actuales brindan lenguajes de alto nivel con diferentes grados de facilidad para el usuario no programador que garantizan este objetivo, los llamados sublenguajes de datos.

- **Control centralizado**

Uno de los objetivos más importantes de los SGBD es garantizar el control centralizado de la información. Permite controlar de manera sistemática y única los datos que se almacenan en la BD, así como el acceso a ella.

Lo anterior implica que debe existir una persona o conjunto de personas que tenga la responsabilidad de los datos operacionales: el administrador de la BD, que puede considerarse parte integrante del SGBD. Existen otros objetivos que deben cumplir los SGBD que en muchos casos dependen de las condiciones o requerimientos específicos de utilización del sistema.

Dentro de las principales funciones que debe cumplir un SGBD, se relacionan con la creación y mantenimiento de la BD, el control de accesos, la manipulación de datos de acuerdo con las necesidades del usuario, el cumplimiento de las normas de tratamiento de datos, evitar redundancias e inconsistencias y mantener la integridad.

Los SGBD, pueden definirse como un paquete generalizado de software, que se ejecuta en un sistema computacional anfitrión, centralizando los accesos a los datos y actuando de interfaz entre los datos

físicos, el usuario y la aplicación que lo utiliza; ejemplos de estos, se explicaran a continuación detallando cada ventaja y desventaja.

1.6.2 Sistemas Gestores de Base de Datos.

- **Oracle**

Se considera a Oracle como uno de los sistemas de bases de datos más completos que existe.

Ventajas:

- Soporte de transacciones.
- Gran estabilidad y seguridad.
- Escalabilidad.
- Multiplataforma.

Inconvenientes:

- Su mayor defecto es su enorme precio, que es de varios miles de euros (según versiones y licencias).
- Otro aspecto a criticar es la seguridad de la plataforma, y las políticas de suministro de parches de seguridad, que incrementan el nivel de exposición de los usuarios. En los parches de actualización provistos durante el 2005 fueron corregidas 22 vulnerabilidades públicamente conocidas, algunas de ellas con una antigüedad de más de 2 años.

Aunque ha tenido un amplio dominio en el mercado de servidores sufre la competencia del Microsoft SQL Server de Microsoft y con licencia libre como PostgreSQL y MySql.

- **MySQL**

Es un SGBD cliente/servidor, multihilo y multiusuario. Se compone de un servidor SQL, varios programas clientes y bibliotecas, herramientas administrativas, y una gran variedad de interfaces de programación (APIs). Se puede obtener también como una biblioteca multihilo que se puede enlazar

dentro de otras aplicaciones para obtener un producto más pequeño, más rápido, y más fácil de manejar. Es el servidor de BD relacionales más popular, es una idea originaria de la empresa opensource MySQL AB, desde enero de 2008 MySQL AB pertenece a Sun Microsystems. Esta es una empresa cuyo negocio consiste en proporcionar servicios en torno al servidor de BD MySQL.

MySQL es desarrollado como software libre en un esquema de licenciamiento dual. Cualquier empresa o persona interesada puede descargar el software de MySQL de Internet y usarlo sin pagar por ello. Además, cualquiera que lo necesite puede estudiar el código fuente y cambiarlo de acuerdo a sus necesidades. Usa la licencia GPL (Licencia Pública General GNU), para definir qué es lo que se puede y no se puede hacer con el software para diferentes situaciones. Sin embargo, si no se está de acuerdo con la licencia GPL o tiene la necesidad de incorporar código de MySQL en una aplicación comercial es posible comprar una versión con una licencia comercial. [2]

Ventajas:

- Diseñado en vistas a la velocidad.
- Consume muy pocos recursos de CPU y memoria. Muy buen rendimiento.
- Tamaño del registro sin límite.
- Buena integración con PHP.
- Utilidades de administración (phpMyAdmin).
- Buen control de acceso usuarios-tablas-permisos.
- Trabaja bajo diferentes plataformas: AIX 4x 5x, Amiga, BSDI, Digital Unix 4x, FreeBSD 2x 3x 4x, HP-UX 10.20 11x, Linux 2x, Mac OS, NetBSD, Novell NetWare 6.0, OpenBSD 2.5, OS/2, SCO OpenServer, SCO UnixWare 7.1.x, SGI Irix 6.x, Solaris 2.5, SunOS 4.x, Tru64 Unix y Windows 9x, Me, NT, 2000, XP, 2003.
- Desarrollo de APIs para C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, y Tcl.

Inconvenientes:

- No soporta subconsultas.
- No soporta transacciones.

- No soporta vistas.

- **PostgreSQL**

Está considerado el SGBD de código abierto más avanzado del mundo. PostgreSQL proporciona un gran número de características que normalmente sólo se encontraban en las BD comerciales de alto calibre tales como Oracle.

Es un SGBD objeto-relacional, ya que aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Su avanzada funcionalidad se pone de manifiesto con las consultas SQL declarativas, el control de concurrencia multiversión, soporte multiusuario, transacciones, optimización de consultas, herencia y valores no atómicos (atributos basados en vectores y conjuntos).

Es altamente extensible: soporta operadores y tipos de datos definidos por el usuario. Soporta la especificación SQL99 e incluye características avanzadas tales como las uniones (joins) SQL92. Cuenta con una API (del inglés Application Program Interface) flexible lo cual ha permitido dar soporte para el desarrollo con PostgreSQL en diversos lenguajes de programación.

Ventajas:

- Costo.
- Estabilidad, Confiabilidad.
- Funcionalidad (es un motor de BD *Avanzado y* con herramientas gráficas como PgAdmin, phpPgAdmin la administración de la BD es sencilla).

Inconvenientes:

- Consume gran cantidad de recursos y carga más el sistema.
- Límite del tamaño de cada fila de las tablas a 8k (se puede ampliar a 32k recompilando, pero con un coste añadido en el rendimiento).
- Es de 2 a 3 veces más lenta que MySQL.

- Menos funciones en PHP.

- **Microsoft SQL Server.**

SQL Server es el SGBD ideal para un amplio espectro de clientes corporativos y productores independientes de software (ISV) inmersa en la creación de aplicaciones empresariales. Las necesidades y requisitos del cliente han dado lugar a innovaciones significativas en el producto SQL Server versión 2000, entre las que se incluyen la facilidad de uso, escalabilidad, fiabilidad, y almacenamiento de datos. SQL Server 2000 es la oferta completa de BD y análisis. Tanto por la capacidad para consultar la BD mediante un explorador como por la compatibilidad con el Lenguaje de marcado extensible (XML, Extensible Markup Language), SQL Server 2000 es la BD totalmente habilitada para Web. [3]

Es un gestor de BD relacionales desarrollado por Microsoft. Para el desarrollo de aplicaciones más complejas (tres o más capas), Microsoft SQL Server incluye interfaces de acceso para la mayoría de las plataformas de desarrollo, incluyendo .NET, además de ser uno de los mejores SGBD basada en el lenguaje SQL, capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea. Las necesidades y requerimientos de los clientes han llevado a la creación de innovaciones de producto significativas para facilitar la utilización, escalabilidad, confiabilidad y almacenamiento de datos.

***Ventajas**:*

- Soporta la configuración automática y la auto-optimización.
- Administración multiservidor para un gran número de servidores.
- Gran variedad de opciones de duplicación de cualquier BD.
- Acceso universal a los datos (Universal Data Access).
- Fácil de usar.
- Escalabilidad. Permite realizar un escalamiento hasta 32 CPU y 64 gigabytes (GB) de RAM, siendo capaz de manejar al máximo multiprocesamiento simétrico aprovechando al máximo el hardware.

- Potencia: Microsoft SQL Server es la mejor BD para Windows NT Server.
- Posee los mejores registros de los benchmarks independientes (TCP) tanto en transacciones totales como en coste por transacción.
- Gestión: Con una completa interfaz gráfica que reduce la complejidad innecesaria de las tareas de administración y gestión de la BD.
- Múltiples instancias y Failover.
- Integración con la Web Acceso Web a datos.
- Soporte para XML (cláusula FOR XML).

Inconvenientes:

- Licencias con costos altos.
- Plataformas Windows.

1.7 Análisis y descripción de herramientas para los diagramas relacionales.

Herramientas CASE

Las Herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero. Estas herramientas nos pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de realizar un diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores.

1.7.1 Herramientas CASE para diseño de BD en Software Libre.

- **GNU Ferret**

GNU Ferret(Free Entity Relationship and Reverse Engineering Tool, Herramienta de Ingeniería Inversa y Entidad-Relación Libre) anteriormente conocido como GerWin, antes de un cambio de nombre para evitar problemas legales con Computer Associates es un clon del programa (privativo) ErWin(TM), que sirve para construir modelos de datos mediante diagramas Entidad-Relación, y generar el SQL correspondiente al modelo. [4]
Características:

- Diagramas Entidad Relación tipo Chen.
- Generación on-the-fly del diagrama de tablas correspondiente al modelo.
- Generación de varios tipos de outputs:
  - SQL 92
  - PostgreSQL
  - MySQL
  - GerwinML

- **DBDesigner**

DBDesigner es una herramienta de software libre que te permite diseñar BD para aplicaciones Web ó cualquier BD para una aplicación en Java, C++, Visual c# ó Visual Basic, Perl ó PHP, para el uso en la Web.

Ahora ya que un buen diseño de la BD no es trivial y de esto depende el buen funcionamiento de la aplicación es muy recomendable usar una herramienta que nos permita hacer el diseño y generar en código SQL, también llamados “SQL Scripts“. Hay muchas herramientas en el mercado. Pero a veces son muy costosas y en realidad no necesitamos pagar tanto dinero si queremos hacer una BD estándar para nuestra aplicación.

Oracle tiene sus propias herramientas y también la BD de Microsoft SQL Server viene con las herramientas apropiadas para diseñar la BD y administrarla hoy en día pero, el uso de la BD MySQL es muy común porque muchos proveedores de Hosting tienen MySQL ya que es gratis y muy eficiente. Es precisamente en este caso que podemos usar esta herramienta llamada DBDesigner.

Está comparado con productos como Oracle's Designer©, IBM's Rational Rose©, Computer Associates's ERwin© y theKompany's DataArchitect.

Con esta herramienta se pueden realizar las siguientes tareas en el momento de construir la BD:

- Crear y modificar las diferentes tablas que va a tener nuestra aplicación.
- Crear y modificar las diferentes relaciones.
- Crear y generar las llaves primarias y secundarias (llamadas primary keys y foreign keys).
- Crear y modificar los índices que nos sirven para acelerar enormemente la eficiencia de los resultados en un comando select.
- Crear y generar las condiciones que deben existir para asegurar la integridad relacional.
- Crear y visualizar el diagrama de entidad relacional.
- Crear y generar el código que nos permite crear la BD en el servidor, los llamados CREATE-Scripts. Que también generan los índices y se encargan de asegurar la integridad relacional.
- Permite decir que tipo de datos DATA TYPE tendrá cada columna de una tabla específica (INTEGER, FLOAT, VARCHAR, TEXT).
- Tener una herramienta para manejar las versiones de la BD, ya que esta puede cambiar con el tiempo y debemos tener cuidado de tener un versión que nos permita trabajar eficazmente, la aplicación por su parte trabaja con una versión específica de la BD y así podemos evitar problemas de que la aplicación no este usando con la versión indicada de la BD.

- A través del Diagrama Entidad Relación se puede visualizar todos los datos y así es mucho mas fácil saber si todo esta en su sitio y si las relaciones de las diferentes tablas están creadas correctamente.

- Podemos crear un diagrama a partir de una BD que no hallamos creado nosotros, sino que ya estaba existente en el momento de comenzar el proyecto. A este proceso se le conoce con el nombre de “Reverse Engineering”.

- Este programa nos permite conectarnos directamente a una BD existente para hacer operaciones importantes.

- Nos permite diseñar selecciones que sirve para extraer datos de la BD á través de varias tablas usando inner joins ó outer joins. De hecho hasta tiene “shortcuts” que nos permiten crear condiciones usando las cláusulas FROM, WHERE, GROUP, HAVING y ORDER.[5]

1.8 Metodologías.

➢ **XP (Extreme Programing)**

Es una de las metodologías de desarrollo de software más exitosas en la actualidad utilizada para proyectos de corto plazo y corto equipo. Consiste en una programación rápida o extrema, pues uno de los requisitos para llegar al éxito del proyecto es tener como parte del equipo al usuario final.

- “Se basa en las pruebas realizadas a los principales procesos, de tal manera que adelantándonos en algo hacia el futuro, podamos hacer pruebas de las fallas que pudieran ocurrir. Es como si nos adelantáramos a obtener los posibles errores.

- Se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.

- Propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento”. [6]

➢ **RUP (Rational Unified Process)**

La metodología de desarrollo empleada para la realización de esta investigación es RUP, llamada así por sus siglas en inglés Rational Unified Process, la cual organiza el proceso de desarrollo de software en cuatro fases y nueve flujos de trabajo.

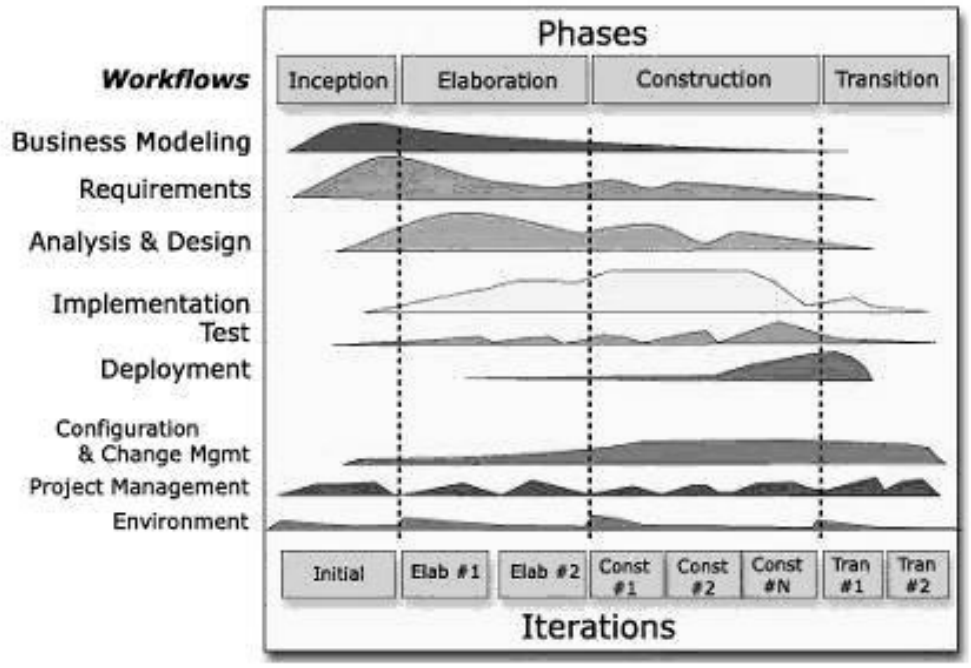

Figura 1 Fases e iteraciones de la metodología RUP.

El ciclo de vida de esta metodología se caracteriza por:

- "Dirigido por casos de uso: Los casos de uso reflejan lo que los usuarios futuros necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. A partir de aquí los casos de uso guían el proceso de desarrollo ya que los modelos que se obtienen, como resultado de los diferentes flujos de trabajo, representan la realización de los casos de uso (cómo se llevan a cabo).

- Centrado en la arquitectura: La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente.

RUP se desarrolla mediante iteraciones, comenzando por los casos de uso relevantes desde el punto de vista de la arquitectura.

- Iterativo e Incremental: RUP propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros. Las iteraciones hacen referencia a pasos en los flujos de trabajo, y los incrementos, al crecimiento del producto". [6]

1.9 Herramientas de desarrollo y administración de Base de Datos.

Generalmente en una aplicación Web se dispone de un servidor remoto que tiene instalado el motor de BD MySQL y el lenguaje PHP, entre otras funcionalidades, estas opciones varían, aunque en términos generales se da esta combinación. "La BD MySQL se puede administrar perfectamente conectándose al servidor en forma remota, mediante SSH (**S**ecure **SH**ell). Como herramienta administrativa MySQL vía SSH, se torna bastante poco funcional, sobre todo para un usuario no experimentado, el manejo es similar al de una pantalla del sistema operativo DOS. Para solucionar este tema, se comenzaron a desarrollar distintas aplicaciones Web, que permiten manejar desde el navegador Web las BD. Entre estas aplicaciones que se desarrollaron, se encuentra phpMyAdmin." [7]

- **phpMyAdmin**

"phpMyAdmin es una herramienta escrita en PHP con la intención de manejar la administración de MySQL a través de páginas Web, utilizando Internet. Actualmente puede crear y eliminar Bases de Datos, crear, eliminar y alterar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios, exportar datos en varios formatos y está disponible en 50 idiomas. Se encuentra disponible bajo la licencia GPL." [7]

Es un programa de libre distribución en PHP, creado por una comunidad sin ánimo de lucro. Es una herramienta muy completa que permite acceder a todas las funciones típicas de la BD MySQL a través de una interfaz Web muy intuitiva.

La aplicación en si no es más que un conjunto de archivos escritos en PHP que se pueden copiar en un directorio de un servidor Web, de modo que, cuando se acceda a esos archivos, muestren unas páginas donde se pueda encontrar las BD a las que se tiene acceso en el servidor de BD y todas sus

tablas. phpMyAdmin, no sólo es un programa OpenSource, sino que es “el programa” para la administración de bases de datos MySQL en forma remota. No solamente por las funcionalidades que nos ofrece y que van mejorando con cada nueva versión, sino también porque a lo largo de los años ha sabido ganarse su lugar, y estar presente en casi todos los proveedores de hosting a nivel mundial, además de haber ganado numerosos premios.

1.10 Ejemplo de gestor de Base de Datos para ajedrez.

ChessBase

ChessBase es un gestor de bases de datos de ajedrez, personal y autónomo que se ha convertido en estándar mundial. Todo el mundo usa ChessBase, desde el Campeón Mundial hasta cualquier aficionado. Es el programa elegido por todos los que aman este juego y quieren conocerlo más a fondo. Lo que se puede hacer con ChessBase: introducir, comentar y archivar partidas, con variantes, comentarios en forma de texto, grabaciones de voz, inserción de imágenes y secuencias de vídeo; analizar posiciones con módulos tan fuertes como un Gran Maestro (GM), como Fritz y Crafty (ambos se incluyen); localizar partidas por aperturas, jugadores y torneos; generar tablas de torneos y estadísticas con gráficos de jugadores o aperturas; fundir partidas sobre la marcha en un árbol de apertura; generar el dossier de un jugador, con toda la información disponible en una BD; localizar una novedad en una partida con un clic de ratón; generar un amplio informe de aperturas con la línea principal y las variantes críticas, los planes así como las partidas más importantes.

Conclusiones

Después de este análisis sobre las BD y sus características además de las necesidades de la investigación así como las herramientas y métodos que se pueden usar para llevarla cabo se ha optado por usar como metodología de desarrollo el RUP. Se selecciona esta metodología porque es la que mejor se ajusta a la investigación del proyecto en cuestión. El rol que se desarrollará es el de diseñador principal de la BD del proyecto (responsable del diseño de la BD), jugando su papel principal en el flujo de trabajo de análisis y diseño en la fase de elaboración; construyendo el diagrama de clases persistentes y modelo de datos para el desarrollo de la BD, siendo esta metodología la que

permite llegar a tan alto nivel. Es la más adaptable para proyectos a largo plazo, además de realizar análisis de riesgos y prever las acciones a realizar en cada caso.

Para escoger las herramientas se tuvo como principal característica de que sea software libre, ya que el proyecto INFODREZ se trazó la estrategia de trabajar sobre software con dicha característica, por lo que la herramienta CASE escogida para el diseño de la BD (diagrama entidad-relación) es DBDesigner, y como SGBD el MySQL porque como principal característica tiene su buena integración con el PHP ya que los módulos a desarrollar en esta etapa han sido programados en dicho lenguaje. Además se escogió como herramienta administrativa de BD a phpMyAdmin ya que está diseñada específicamente para administrar a MySQL.

Capítulo 2.- Descripción y análisis de la solución propuesta.

Introducción

En este capitulo se expone los procesos más importantes por los cuales se pasó para el diseño y desarrollo de la BD del gestor de contenido de ajedrez por cada módulo al cual se le brinda servicio. Se hace una descripción de los requisitos funcionales y no funcionales que debe cumplir la BD, se muestra los diagramas de clases persistentes de cada módulo así como el diagrama Entidad-Relación de cada uno, luego se describen cada una de las clases y tablas que se encuentran en los diagramas antes mencionados. Además se hace una fundamentación de la descripción de la arquitectura propuesta.

2.1 Integración con otros módulos y sistemas.

Los módulos Arbitraje, Torneo, Ajedrez por Correspondencia, Torneo Online, Estadísticas y Cálculo de Elo se integran teniendo una interacción directa con la BD física haciendo peticiones a través de clases de acceso a datos que facilita el acceso a los datos desde la aplicación.

2.2 Descripción de la arquitectura y fundamentación.

La BD propuesta contiene 43 tablas en modelo físico, las cuales serán utilizadas por los seis módulos del proyecto: Estadísticas, Torneo, Torneo Online, Ajedrez por Correspondencia, Arbitraje y Cálculo de Elo, haciendo uso principalmente de las tablas *jugador, torneo, partida, arbitro, ronda, usuario*, las cuales recogen la información de mayor peso. Por otra parte el sistema se encuentra estructurado en tres capas siguiendo el patrón de arquitectura layers (capas), de ahí que exista una capa de acceso a datos encargada de gestionar la información.

2.3 Descripción de los Requisitos Funcionales y No Funcionales que debe cumplir la solución propuesta.

2.3.1 Requisitos Funcionales.

RF 1 Gestionar jugador.

1.1 Insertar jugador.

1.2 Actualizar jugador.

1.3 Listar jugador.

1.3.1 Listar jugador por ELO.

1.3.2 Listar jugador por federación.

1.3.3 Lista jugadores en un torneo determinado.

1.3.4 Listar partidas ganadas por un jugador determinado.

1.3.5 Listar estadísticas de un jugador.

RF 2 Gestionar árbitro.

2.1 Crear árbitro.

2.2 Eliminar árbitro.

2.3 Actualizar árbitro.

2.4 Listar árbitro.

2.4.1 Listar árbitro por categoría.

2.4.2 Listar árbitro por federación.

2.4.3 Listar árbitro por torneo.

2.4.4 Listar árbitro por partida.

RF 3 Gestionar torneo.

3.1 Crear torneo.

3.2 Eliminar torneo.

3.3 Modificar torneo.

3.4 Listar torneo.

3.4.1 Listar torneo por federación.

3.4.2 Listar torneo por tipo.

RF 4 Gestionar partida.

4.1 Crear partida.

4.2 Modificar partida.

4.3 Listar partida.

4.3.1 Listar partida por jugadores.

4.3.2 Listar partidas por aperturas.

4.3.3 Listar partidas por torneos.

4.3.4 Listar resultado de partidas.

RF 5 Gestionar ronda.

5.1 Crear ronda.

5.2 Modificar ronda.

5.3 Listar Ronda.

5.3.1 Listar ronda por torneo y fecha.

5.3.2 Listar ronda de un jugador en un torneo.

5.3.3 Mostrar tabla de pareo.

5.4 Realizar el reporte de cada ronda.

RF 6 Devolver jugadores de un torneo determinado.

RF 7 Gestionar usuario.

7.1 Insertar usuario.

7.2 Modificar datos del usuario.

7.3 Eliminar usuario.

7.4 Seleccionar usuario(s).

RF 8 Gestionar roles a usuarios.

8.1 Insertar asignación de rol a los usuarios.

8.2 Modificar asignación de rol.

8.3 Eliminar asignación de rol (si no esta asignado a un usuario).

8.4 Seleccionar rol.

8.5 Seleccionar roles asignados a un usuario.

8.6 Seleccionar listado de roles asignados.

RF 9 Gestionar jugadores que participaran en el torneo.

9.1 Adicionar un nuevo jugador al torneo.

9.2 Listar jugadores que inscritos en el torneo.

9.3 Eliminar jugadores de la lista de participantes de torneo.

RF 10 Gestionar árbitros que participaran en el torneo.

10.1 Adicionar un nuevo árbitro.

10.2 Listar árbitros del torneo.

10.3 Eliminar árbitros de la lista del torneo.

RF 11 Gestionar modelos de logística.

11.1 Adicionar modelos de la logística.

11.2 Actualizar modelos de logística.

11.3 Listar modelos de logística.

RF 12 Gestionar correo electrónico.

12.1 Adicionar correo electrónico.

12.2 Listar correos.

12.2.1 Listar correos electrónicos por asunto.

12.2.2 Listar correos electrónicos por tipo de mensaje.

RF 13 Gestionar Noticia para ronda.

13.1 Adicionar Noticia.

13.2 Modificar Noticia.

13.3 Eliminar Noticia.

13.4 Mostrar noticia.

13.4.1 Mostrar noticia por una ronda determinada.

13.4.1 Listar noticias por un torneo determinado.

13.4.2 Listar noticias por una fecha determinada.

RF 14 Gestionar Imagen.

14.1 Adicionar imagen.

14.2 Cambiar imagen.

14.3 Eliminar imagen.

RF 15 Gestionar evento.

15.1 Adicionar evento.

15.2 Modificar evento.

15.3 Eliminar evento.

15.4 Listar eventos.

15.4.1 Listar eventos por un torneo determinado.

15.4.2 Listar evento por una fecha determinada.

RF 16 Consultar Torneo.

16.1 Consultar listado de jugadores de por torneo.

16.2 Consultar listado de árbitros por torneo.

16.3 Consultar resultados de una ronda determinada.

16.4 Consultar bases de un torneo.

16.5 Consultar objetivos de un torneo.

16.6 Consultar partidas de un torneo determinado

RF 17 Gestionar partida online aplazada.

17.1 Adicionar partida aplazada.

17.2 Eliminar partida aplazada.

17.3 Listar partidas aplazadas.

17.3.1 Listar partidas aplazadas por jugador.

17.3.2 Listar resultado de una partida aplazada.

RF 18 Gestionar jugador expulsado.

18.1 Adicionar jugador expulsado de partida online.

18.2 Eliminar jugador expulsado.

18.3 Listar jugadores expulsados.

18.3.1 Listar causa de expulsión de un jugador determinado.

2.3.2 Requisitos No Funcionales.

Los requisitos no funcionales mencionados a continuación son los que tienen correspondencia con el desarrollo de la BD.

1. Rendimiento.

La aplicación debe estar concebida para el consumo mínimo de recursos y el sistema debe ser capaz de formular la respuesta lo más rápido posible.

2. Soporte.

Se requiere que esté instalado un Gestor de BD que soporte grandes volúmenes de datos y velocidad de procesamiento.

3. Portabilidad.

El sistema deberá ser compatible con el sistema operativo Linux y con Windows (Versiones como 2000 y XP), siendo además accesible principalmente con el navegador Mozilla.

4. Hardware.

Para el servidor de BD se requiere tarjeta de red, que tenga al menos 256MB de RAM y 1 GB de disco duro; y que el procesador tenga 1.2 GHz de velocidad como mínimo.

5. Software.

Los servidores, incluido el de la BD, deben tener instalados el sistema operativo Windows 2000 o superior, o Linux preferencialmente. La BD será implementada en el Gestor de Base de Datos MySQL.

6. Seguridad.

El sistema debe comunicarse usando un protocolo seguro, (https). Los datos no pueden viajar de forma transparente por la red, deben ser encriptados. Se debe mantener la integridad de la información, es decir que no se pierda durante su almacenamiento o transporte. También se deben realizar auditorias a los principales eventos dentro del sistema, registrando al usuario y los eventos efectuados.

7. Confiabilidad.

La información manejada por el sistema deberá estar protegida de acceso no autorizado y divulgación.

8. Integridad.

La información manejada por el sistema será objeto de cuidadosa protección contra la corrupción.

9. Fiabilidad.

Debe garantizarse el resguardo de la información, así como la grabación periódica de la BD, de forma tal que se posibilite la reinstalación del sistema y los datos, en caso de algún problema presentado en la explotación del mismo.

2.4 Diagramas de clases persistentes.

Las clases persistentes por lo general tienen como origen las clases clasificadas como entidad porque ellas modelan la información y el comportamiento asociado de algún fenómeno o concepto, como una persona, un objeto del mundo real o un suceso. Todas las clases identificadas en el dominio del análisis no son persistentes. La persistencia es la capacidad de un objeto de mantener su valor en el espacio y en el tiempo.

El Diagrama de Clase se utiliza para modelar la estructura lógica de la BD, con clases representando tablas, y atributos de clase representando columnas. Las clases persistentes y sus atributos hacen referencia directamente a las entidades lógicas y a sus atributos.

A continuación, en la siguiente página, se muestran los diagramas de clases persistentes por cada uno de los módulos.

2.4.1. Módulo Arbitraje.

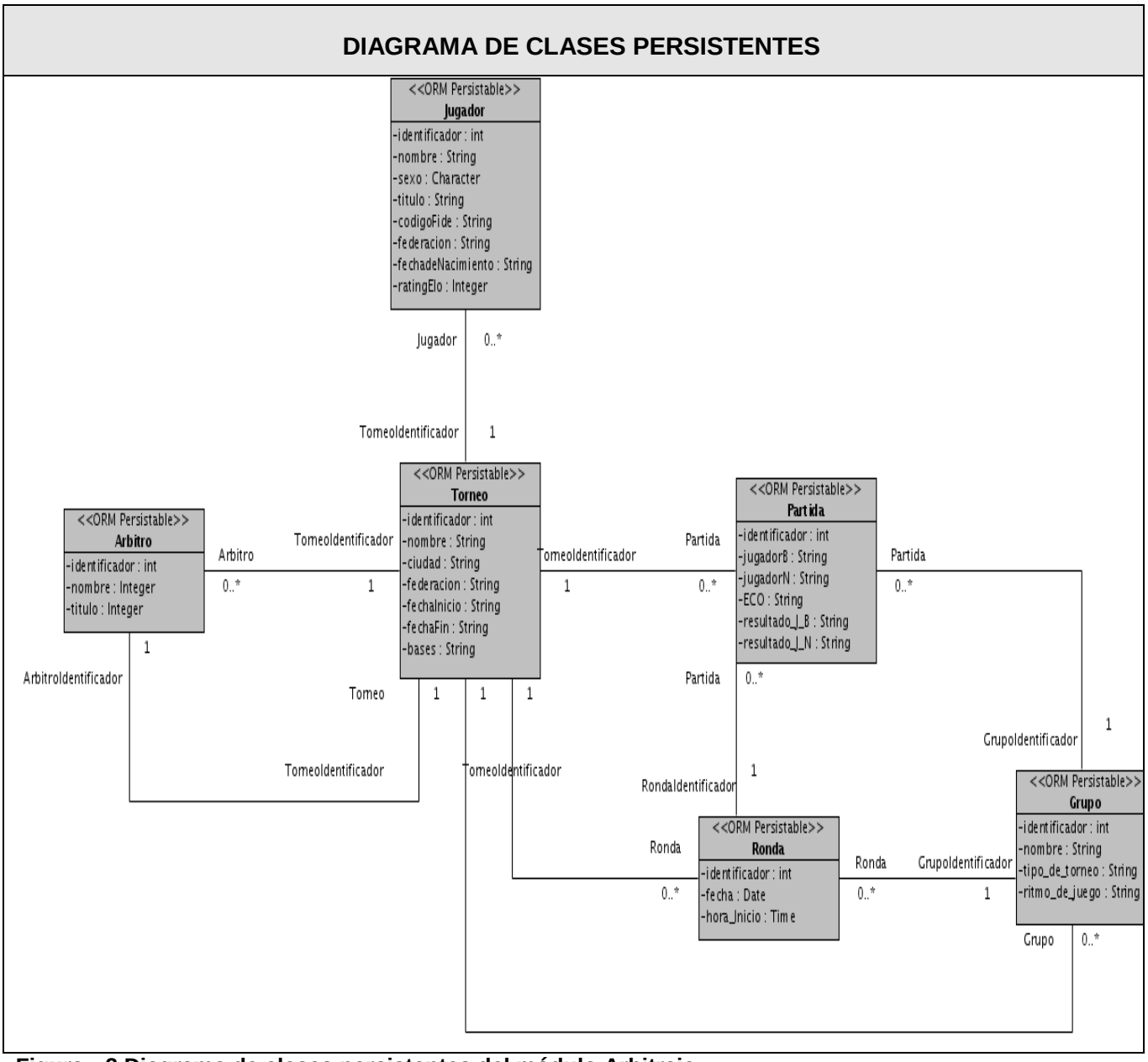

Figura 2 Diagrama de clases persistentes del módulo Arbitraje.

2.4.2. Módulo Estadísticas.

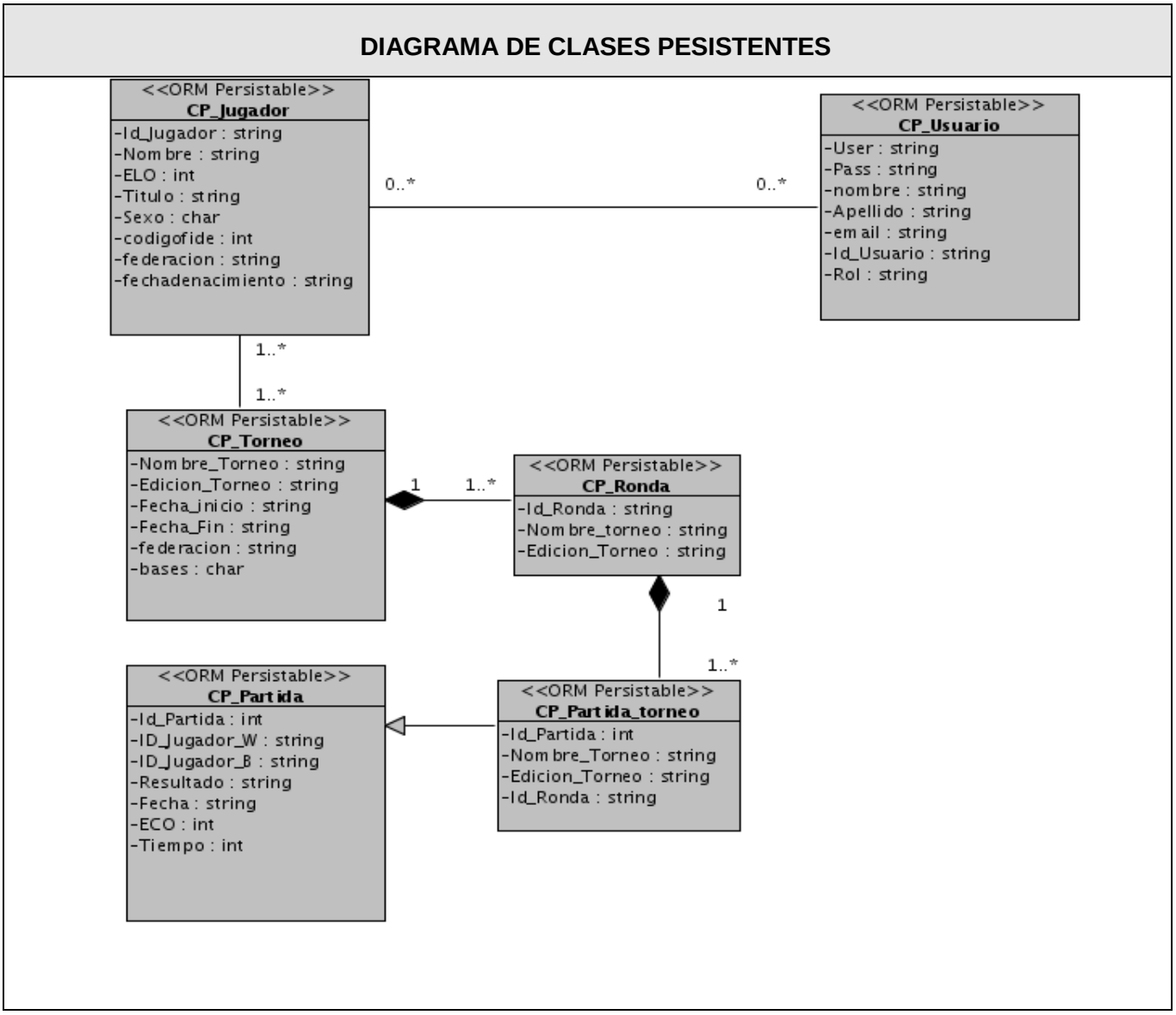

Figura 3 Diagrama de clases persistentes del módulo Estadísticas.

2.4.3. Módulo Torneo.

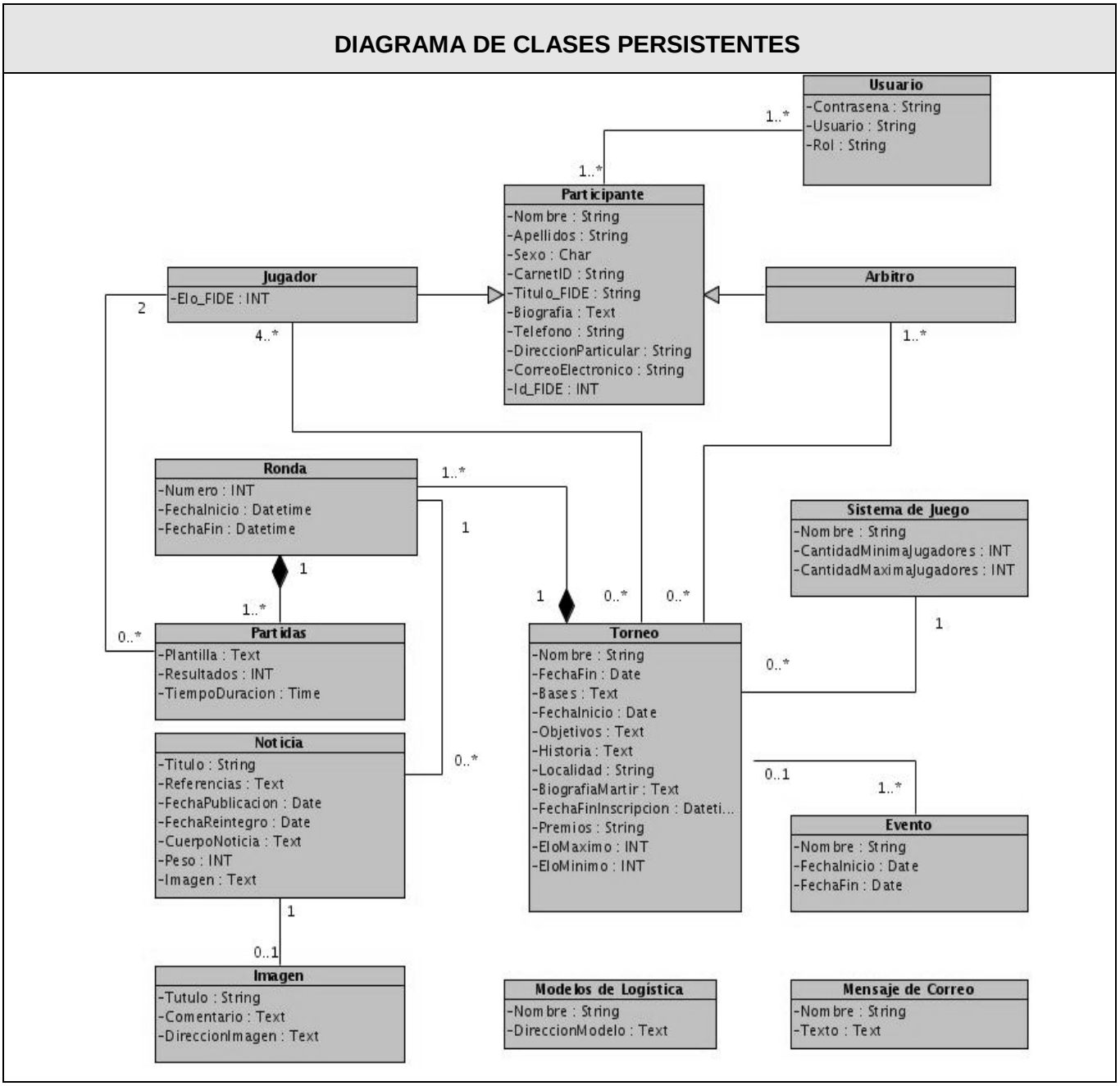

Figura 4 Diagrama de clases persistentes del módulo Torneo.

2.4.4. Módulo Ajedrez por Correspondencia.

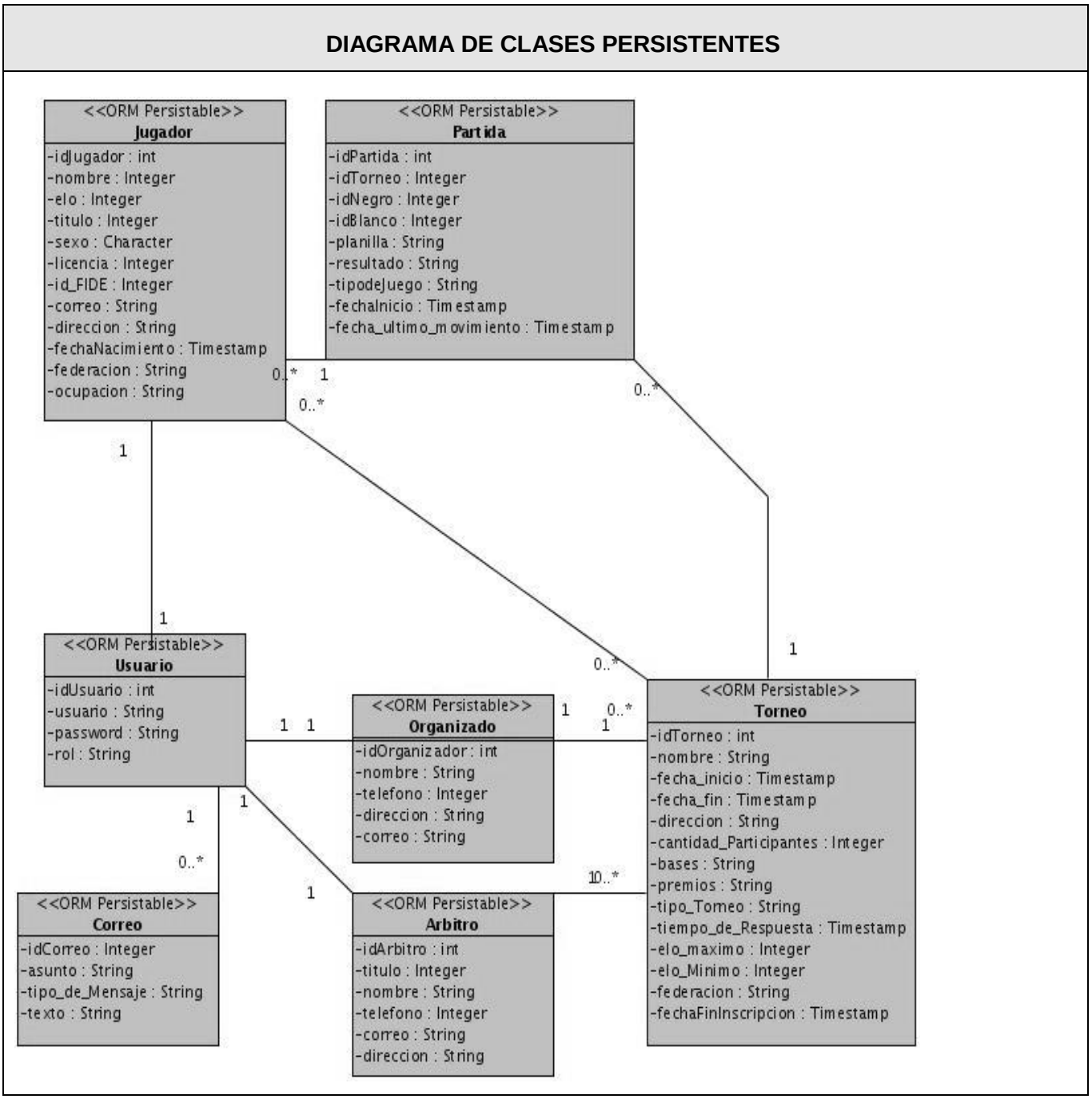

Figura 5 Diagrama de clases persistentes del módulo Ajedrez por correspondencia.

2.4.5. Módulo Torneo Online.

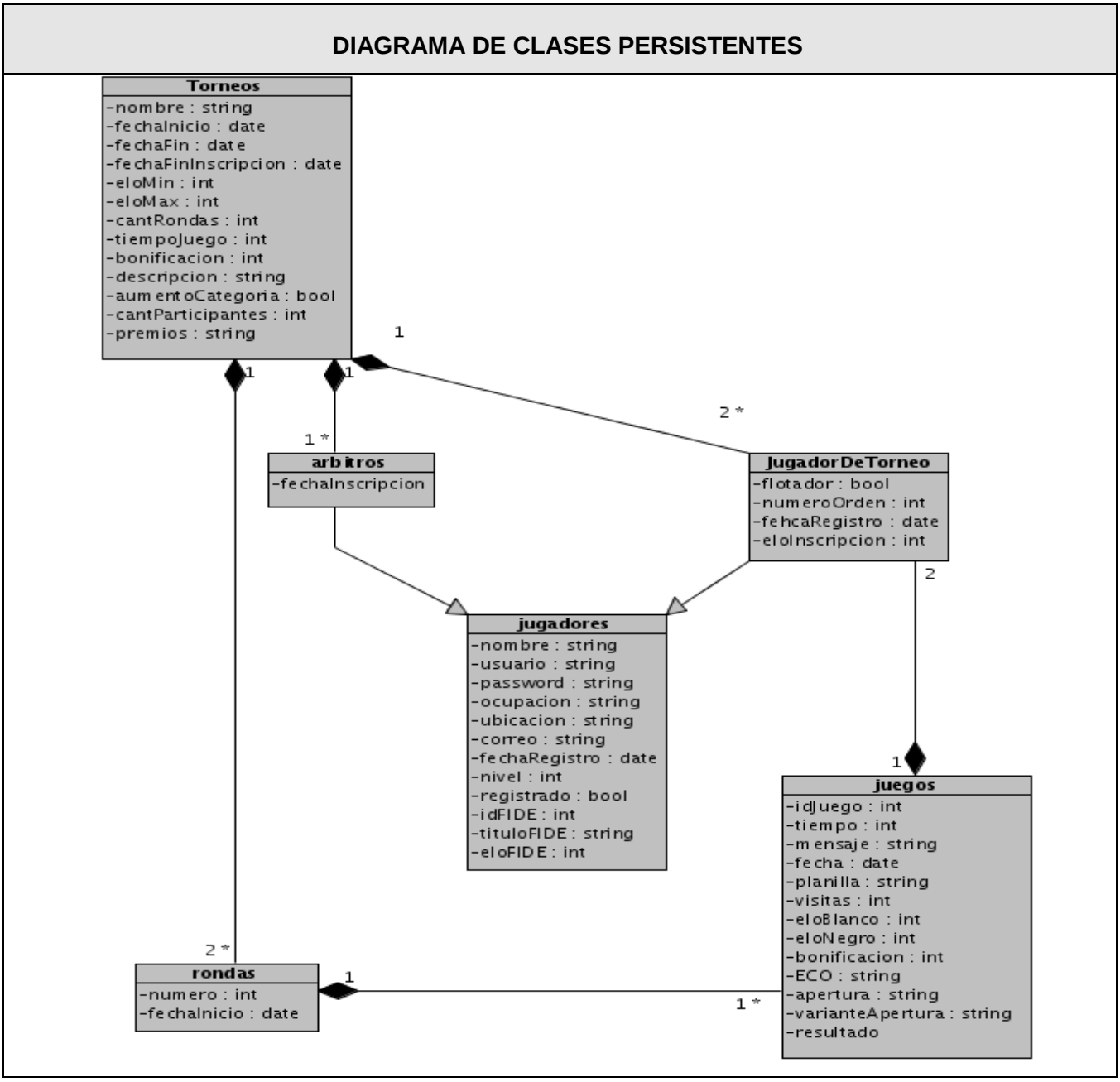

Figura 6 Diagrama de clases persistentes del módulo Torneo Online.

2.5 Descripción de las clases persistentes.

A continuación se representan las principales clases persistentes con sus atributos y tipos de datos correspondientes, las demás se pueden encontrar en el Anexo 1.

Tabla 1: Clase Torneo

Nombre: Torneo	
Atributo	**Tipo**
idTorneo	integer
nombre	string
fechaInicio	timestamp
fechaFin	timestamp
ciudad	string
cantParticipantes	integer
bases	string
premios	string
tipoTorneo	string
eloMaximo	integer
eloMinimo	integer
federacion	string
fechaFinInscripcion	timestamp
arbitroPrincipal	string
bonificacion	string

Tabla 2: Clase Ronda

Nombre: Ronda	
Atributo	**Tipo**
idRonda	integer
idTorneo	integer
fechaInicio	timestamp

Tabla 3: Clase Jugador

Nombre: jugador	
Atributo	**Tipo**
idJugador	integer
nombre	string
eloFIDE	integer
titulo	string
sexo	char
idFIDE	string
federación	string
fechaNacimiento	timestamp
telefono	integer
correo	string

Tabla 4 : Clase Partida

Nombre: Partida	
Atributo	**Tipo**
idPartida	integer
idBlanco	integer
idNegro	integer
fechaInicio	timestamp
eco	string
fechaFin	timestamp
resultado	string
tipoJuego	string

Tabla 5: Clase Arbitro.

Nombre: Arbitro	
Atributo	**Tipo**
idArbitro	integer
nombre	string
titulo	string
direccion	string
telefono	integer

2.6 Diagramas Entidad Relación.

A partir de la siguiente página se muestran los diagramas Entidad Relación separados por cada uno de los módulos a los cuales se le debe prestar servicio, debido a la cantidad de clases que conformaban el diagrama principal y para una mayor compresión del mismo; el diagrama general se encuentra seguido de los módulos.

2.6.1 Módulo Estadísticas.

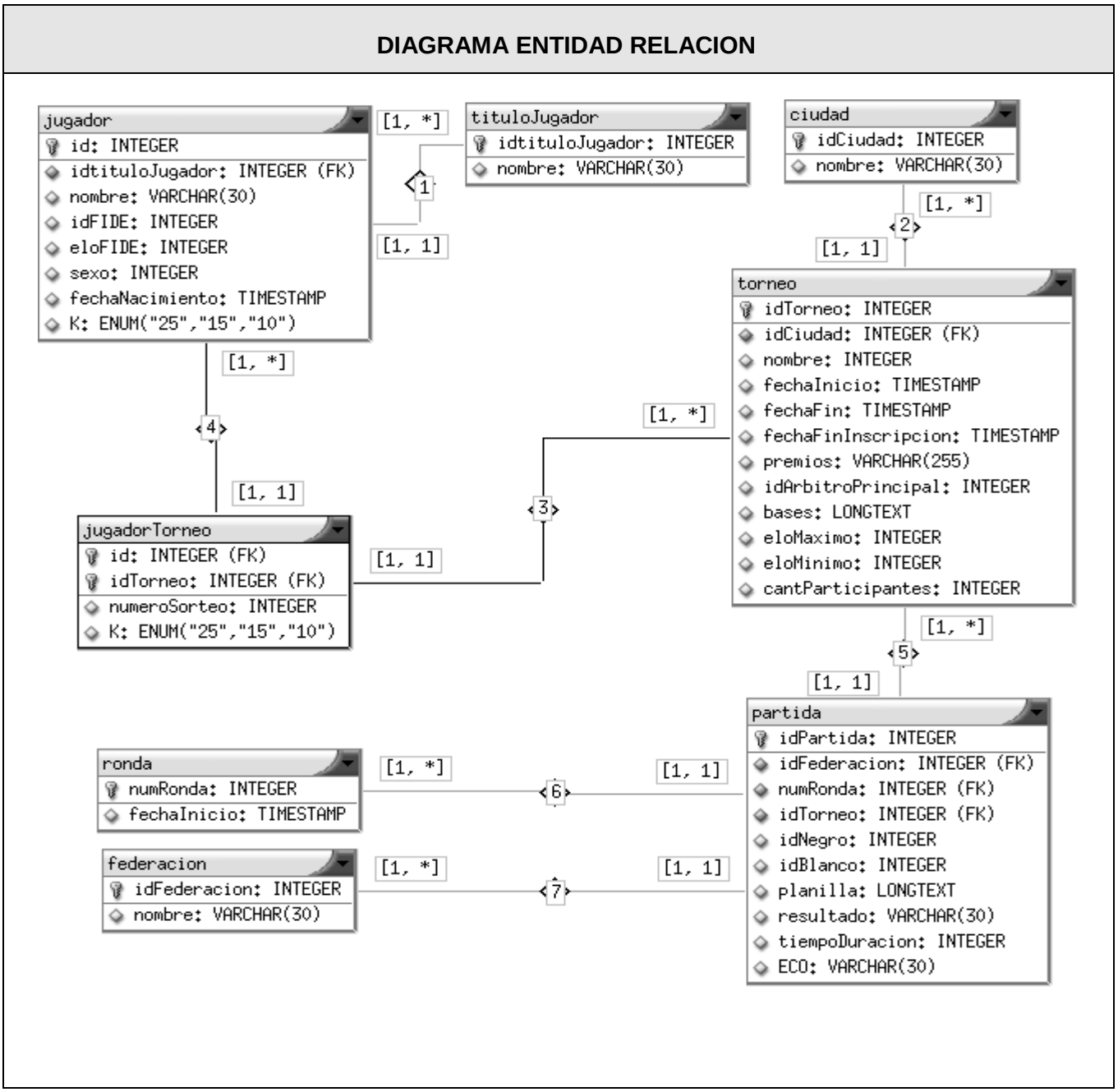

Figura 7 Diagrama Entidad-Relación del módulo Estadísticas.

2.6.2 Módulo Torneo.

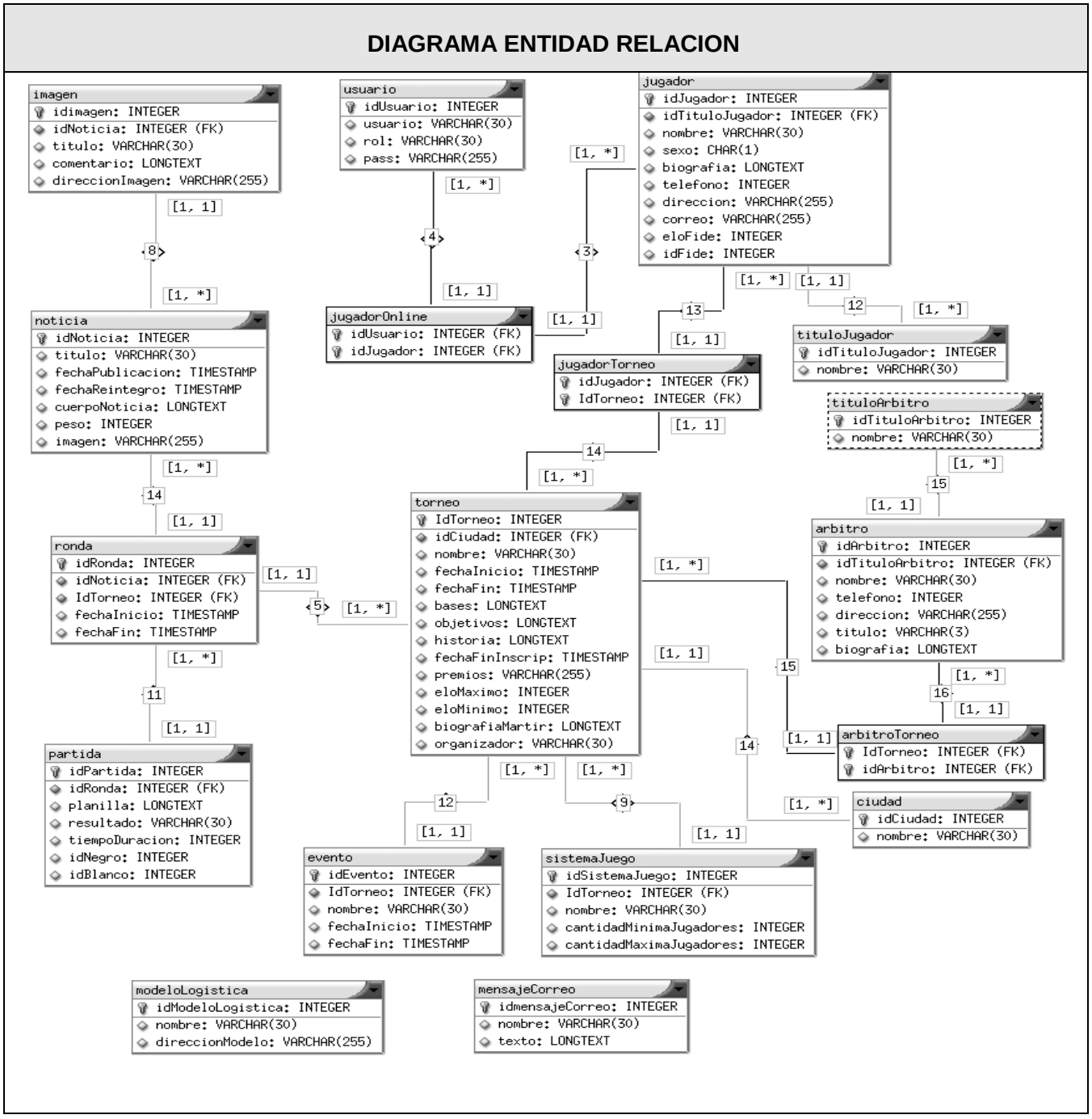

Figura 8 Diagrama Entidad-Relación del módulo Torneo.

2.6.3 Módulo Torneo Online.

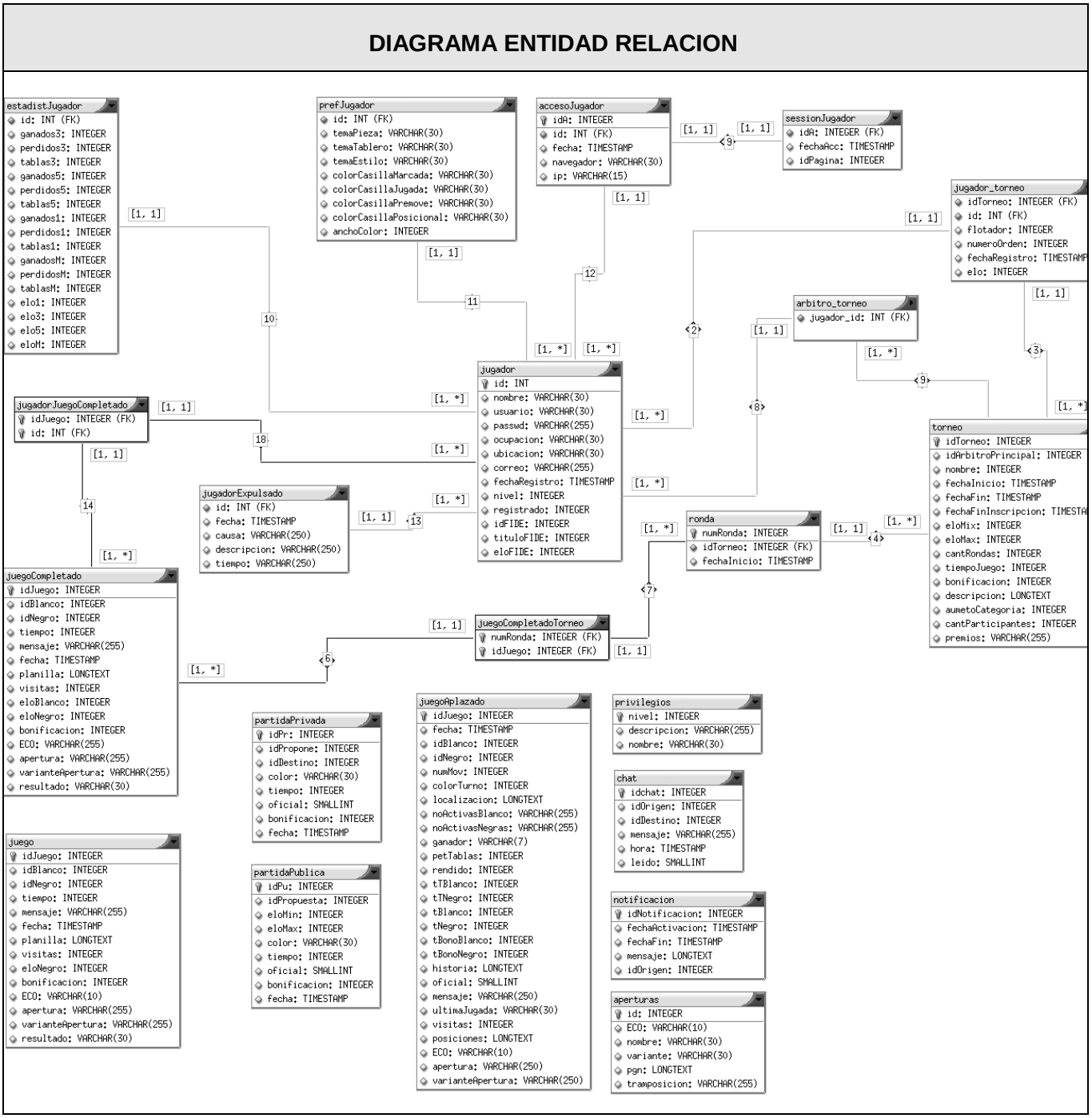

Figura 9 Diagrama Entidad-Relación del módulo Torneo Online.

2.6.4 Módulo Arbitraje.

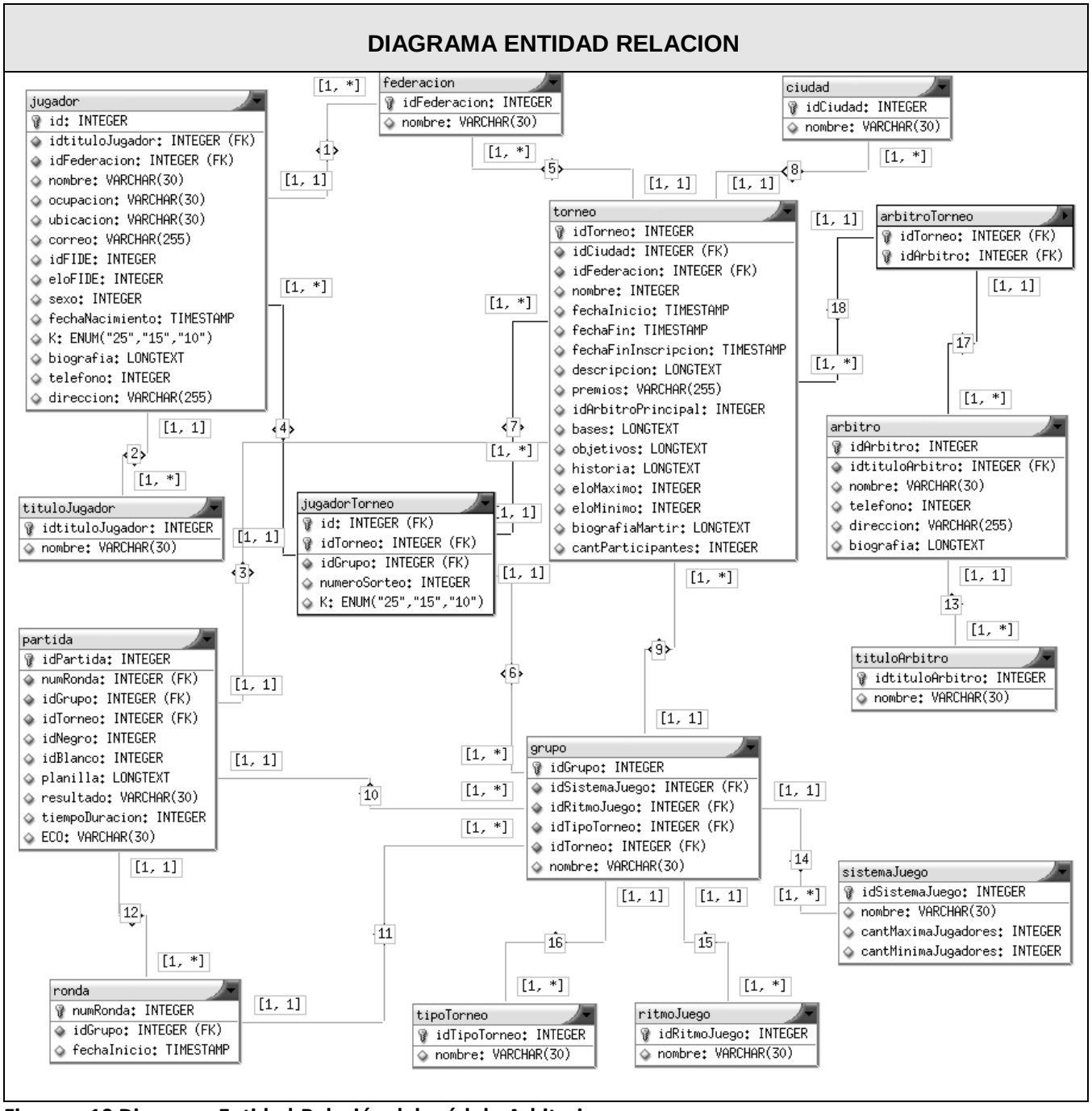

Figura 10 Diagrama Entidad-Relación del módulo Arbitraje.

2.6.5 Módulo Ajedrez por correspondencia.

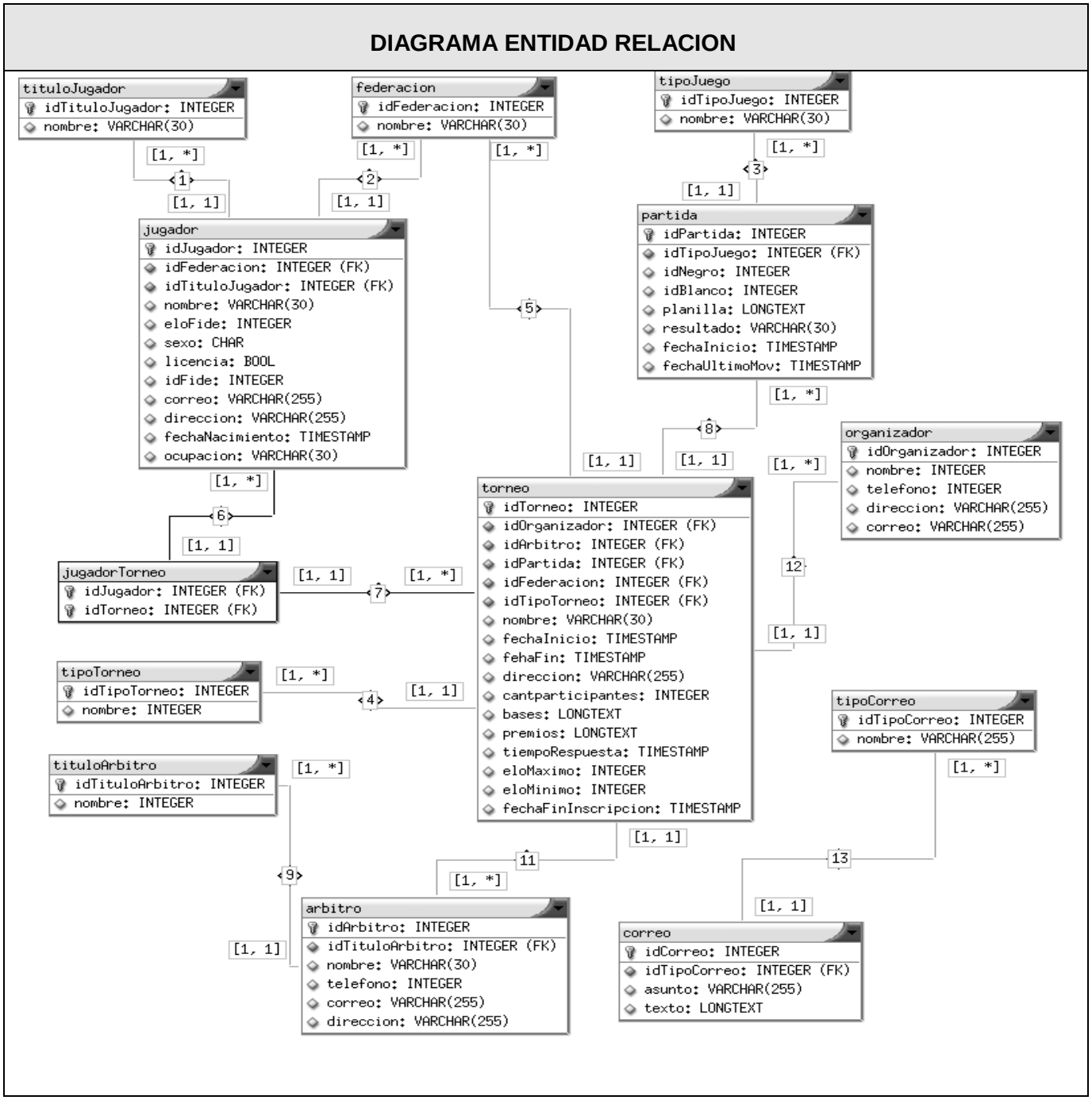

Figura 11 Diagrama Entidad-Relación del módulo Ajedrez por correspondencia.

2.6.6 Diagrama general.

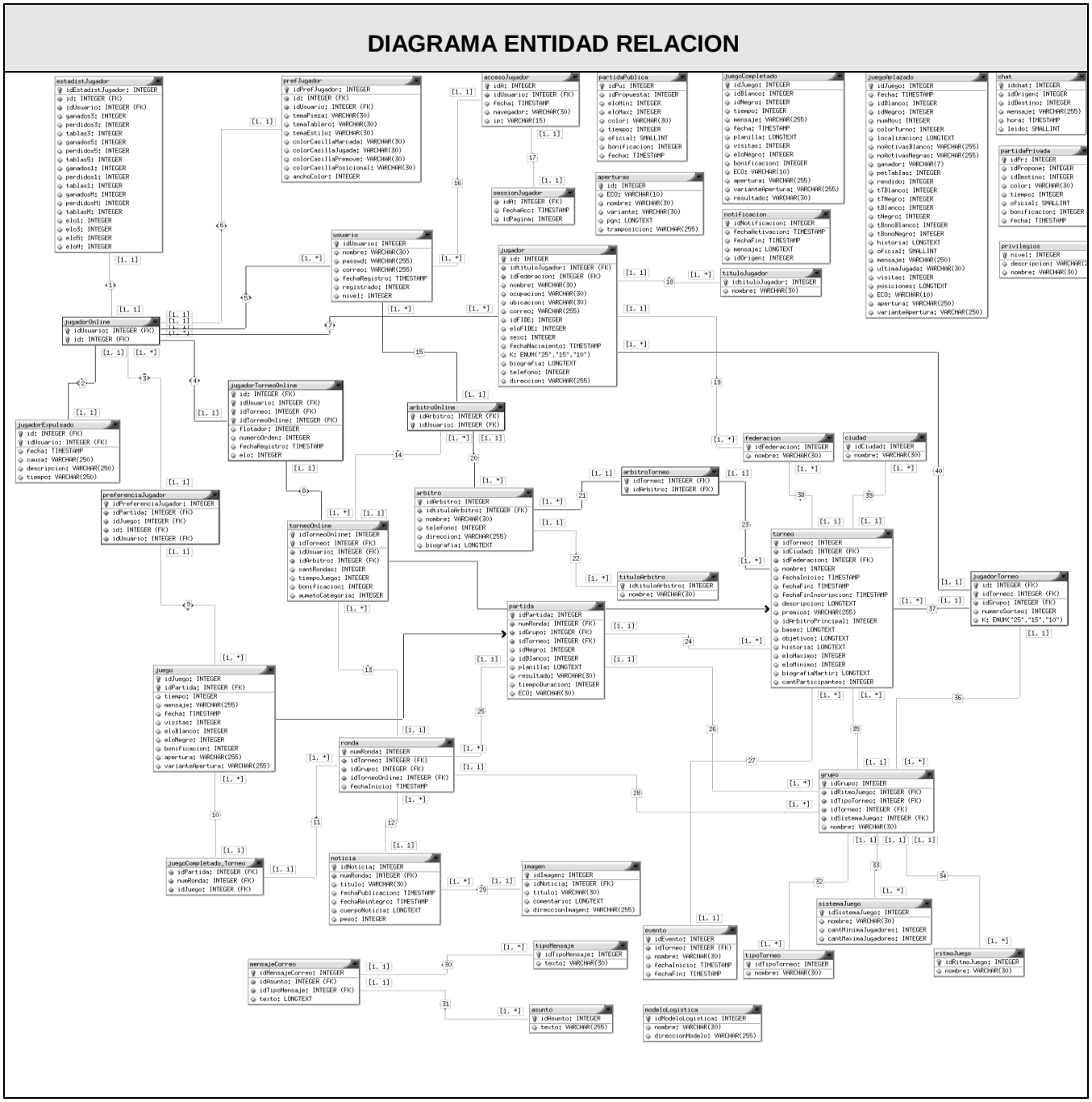

Figura 12 Diagrama Entidad-Relación General.

2.7 Descripción de las tablas.

A continuación se muestra la descripción de las principales tablas sobre las cuales giran los principales procesos del sistema, sus campos con el tipo de dato y la descripción textual de cada uno, las demás tablas se encuentran en el Anexo 2.

Se partió de definir las clases persistentes. Las clases simples se representaron como tablas y en las relaciones de 1...n se añadió una llave extranjera al extremo n, que se corresponde con la llave de la clase del extremo 1.

En el caso de las relaciones n...m se crea una nueva tabla que tiene como llave las clases que conforman a relación.

Tabla 6.- torneoOnline.

Nombre: torneoOnline		
Descripción: Almacena los torneos online.		
Atributo	**Tipo**	**Descripción**
idTorneoOnline	INTEGER	Id del torneo online.
idTorneo(FK)	INTEGER	Id del torneo.
cantRondas	INTEGER	Cantidad de rondas que tiene el torneo.
tiempoJuego	INTEGER	Tiempo que va a tener cada juego.
bonificacion	INTEGER	Bonificación que se da al ganador.
aumentoCategoria	INTEGER	Aumento de categoría como jugador online.
cantParticipantes	INTEGER	Cantidad de participantes del torneo.
causa	VARCHAR(255)	Causa por la cual fue expulsado.
descripcion	VARCHAR(255)	Descripción de la causa.
tiempo	TIMESTAMP	Tiempo que fue expulsado.

Tabla 7 .- torneo.

Nombre: torneo		
Descripción: Almacena toda la información referente a un torneo.		
Atributo	**Tipo**	**Descripción**
IdTorneo	INTEGER	Id del torneo.
idFederacion(FK)	INTEGER	Id de la Federación a la cual pertenece el torneo.
idCiudad(FK)	INTEGER	Id de la ciudad donde se desarrolla el torneo.
Nombre	VARCHAR(30)	Nombre del torneo.
fechaInicio	TIMESTAMP	Fecha de inicio del torneo.
FechaFin	TIMESTAMP	Fecha de terminación del torneo.
fechaFinIncripcion	TIMESTAMP	Fecha limite de inscripción.
cantParticipantes	INTEGER	Cantidad de participantes del torneo.
descripcion	LONGTEXT	Descripción del torneo.
premios	VARCHAR(255)	Premios que otorgados por el torneo.
idArbritoPrincipal	INTEGER	Id del árbitro principal.
bases	LONGTEXT	Bases del torneo.
objetivos	LONGTEXT	Objetivos que persigue el torneo.
historia	LONGTEXT	Historia del torneo.
eloMaximo	INTEGER	Rating Elo máximo para entrar al torneo.
eloMinimo	INTEGER	Rating Elo mínimo para entrar al torneo.
biografiaMartir	LONGTEXT	La biografía del mártir del torneo.

Tabla 8.- jugador.

Nombre: jugador		
Descripción: Almacena información sobre los jugadores.		
Atributo	**Tipo**	**Descripción**
id	INTEGER	Id del jugador.
idTituloJugador(FK)	INTEGER	Hace referencia al titulo.
idFederacion(FK)	INTEGER	Id de la federación.
nombre	VARCHAR (30)	Nombre del jugador.
sexo	CHAR (1)	Sexo del jugador.
idFIDE	INTEGER	Código que asigna la FIDE a los jugadores.
fechaNacimiento	TIMESTAMP	Fecha de nacimiento del jugador.
K	ENUM(“25”,”15”,”10”)	Constante para el cálculo de ELO.
eloFIDE	INTEGER	Elo FIDE del jugador.
ocupacion	VARCHAR(30)	Ocupación del jugador.
ubicacion	VARCHAR(30)	Centro laboral o de estudios.
correo	VARCHAR(255)	Correo electrónico del jugador.
biografia	LONGTEXT	Biografía del jugador.
direccion	VARCHAR(255)	Dirección donde reside el jugador permanentemente.
telefono	INTEGER	Teléfono del jugador.

Tabla 9.-grupo

Nombre: grupo		
Descripción: Almacena información sobre los grupos.		
Atributo	**Tipo**	**Descripción**
IdGrupo	INTEGER	Id del grupo.
IdTorneo(FK)	INTEGER	Id del torneo al que pertenece el grupo.
IdRitmoJuego(FK)	INTEGER	Es el ritmo de juego del grupo.

idTipoTorneo(FK)	INTEGER	Es el tipo de torneo en que esta el grupo.
idSistemaJuego(FK)	INTEGER	Es el sistema de juego del grupo.
nombre	VARCHAR(30)	Nombre del grupo.

Tabla 10.- partida**.**

Nombre: partida		
Descripción: Almacena los datos de las partidas desarrolladas.		
Atributo	**Tipo**	**Descripción**
IdPartida	INTEGER	Id de la partida.
numRonda(FK)	INTEGER	Id de la ronda en que se desarrolló la partida.
IdGrupo(FK)	INTEGER	Id del grupo en la cual se desarrolló la partida.
IdTorneo(FK)	INTEGER	Id del torneo al cual pertenece la partida.
IdNegro	INTEGER	Id del jugador de las piezas negras.
IdBlanco	INTEGER	Id del jugador de las piezas blancas.
planilla	LONGTEXT	Planilla donde se guardan los movimientos realizados en la partida.
resultado	VARCHAR(30)	Resultado de la partida.
tiempoDuracion	INTEGER	Tiempo que dura la partida.
ECO	VARCHAR(30)	Estrategia de apertura.

Tabla 11.- arbitro.

Nombre: arbitro		
Descripción: Almacena toda la información de un árbitro.		
Atributo	**Tipo**	**Descripción**
IdArbitro	INTEGER	Id del árbitro.
idTituloArbitro(FK)	INTEGER	Id del titulo de árbitro relacionado con el árbitro.
nombre	VARCHAR(30)	Nombre del árbitro.

telefono	INTEGER	Teléfono del árbitro.
direccion	VARCHAR(255)	Dirección del árbitro.
biografia	LONGTEXT	Biografía del árbitro.

Tabla 12.- ronda.

Nombre: ronda		
Descripción: Almacena la información general de una ronda.		
Atributo	**Tipo**	**Descripción**
numRonda	INTEGER	Id de la ronda.
idTorneo(FK)	INTEGER	Id del torneo al cual pertenece la ronda.
idGrupo(FK)	INTEGER	Id del grupo al cual pertenece la ronda.
idTorneoOnline(FK)	INTEGER	Id del torneo online al que pertenezca la ronda.
fechaInicio	TIMESTAM	Hora de inicio de la ronda.

2.8 Nomencladores.

En programación de sistemas, un nomenclador no suele ser otra que una tabla (o vector 2D de al menos 2 columnas) que te permite, en principio, el ingreso de datos seleccionando un elemento (por ejemplo usando una ventana de selección) minimizando con ello la probabilidad de que el operador ingrese algún dato con errores (por ejemplo una localidad con errores ortográficos).

Un nomenclador puede ser usado desde con el prehistórico GWBASIC, pasando por COBOL, Clipper, VFP, y cualquiera de los actuales lenguajes, incluso las bases de datos más potentes tales como Oracle. Su uso es imprescindible.

Lo que debe quedar claro es que no es el profesional en informática quien arma el nomenclador: es el usuario, quien lo conoce. El desarrollador lo ayudará a depurarlo (por ejemplo, eliminando entradas superfluas, erróneas y/o redundantes). El usuario deberá dar su conformidad por escrito antes de implementarlo.

Su uso es público (cualquier operador lo debe usar para cargar datos usando los forms adecuados) pero no cualquiera puede modificarlo: una vez más es el usuario - responsable del dato quien debe tener los privilegios de ABM sobre el nomenclador, y todas sus actividades deben quedar convenientemente registradas. Si posees la posibilidad de deshacer, mejor que mejor.

Imagina que el nomenclador tiene 2 columnas:

COD_POSTAL	DETALLE
.....	.....
.....	.....
ABC001	SmallVille
.....	.....
.....	.....

Si el operador escoge ABC001 podrías pensar en almacenar ABC001 en vez de SmallVille, dado que ocupa menos espacio:

1) Hoy los discos son monstruosamente grandes.

2) Si algún día el encargado de mantener el nomenclador cambia (por motivos válidos o no) el detalle SmallVille por GigaVille, si se ha almacenado ABC001 y se hace listados o consultas históricos (o sea, hacia atrás en el tiempo) aparecerá la localidad de GigaVille que antaño no existía. Por lo contrario, si se almacena el Detalle asociado al código, aparecerá lo que el operador escogió cuando grabó la información, haya habido los cambios que se quiera en el nomenclador.

Conclusiones

En este capítulo se definieron los requisitos funcionales y no funcionales con los que debe cumplir la BD. Se mostró el diagrama de clases persistentes de cada módulo del proyecto, analizando todas las clases persistentes encontradas, definiendo un total de 15 clases, de las cuales se ha explicado brevemente cada uno de sus atributos y su tipo de dato.

Se mostró el diagrama entidad relación para cada módulo del proyecto, explicando brevemente cada una de las entidades representadas y cada uno de sus campos y una pequeña descripción de los mismos, en el cual se definieron 43 entidades de las cuales 9 son nomencladores, con esto se facilita el entendimiento de la BD diseñada.

Capitulo 3.- Validación del diseño realizado.

Introducción

En este capítulo se brinda información sobre la validación teórica realizada al diseño descrito en el capítulo 2, las reglas de integridad, así como la normalización y el análisis de la redundancia de la información. Se toman aspectos como la seguridad y control de acceso a la BD.

3.1 Validación teórica del diseño.

3.1.1 Integridad

Mantener la integridad de una BD es asegurarse de que los datos que contiene son correctos, evitando datos inconsistentes o erróneos de cualquier otro tipo. Normalmente, la integridad se expresa mediante restricciones o reglas que no se pueden violar. Estas restricciones se pueden aplicar tanto a los datos, como a sus relaciones, y es el SGBD quien se encarga de mantenerlas.

Integridad de entidad: la clave primaria de una entidad no puede tener valores nulos y siempre deberá ser única. Un ejemplo de las buenas prácticas lo constituye la elección del campo de la tabla que será o formará parte de la llave primaria, partiendo de la obligación de que todos los valores del atributo son únicos y nunca nulos (**NULL**).

Integridad de Transacciones: Define los estados por los que una tupla puede pasar válidamente, como son: introducido, pendiente, seleccionado, enviado, cancelado y terminado. Está encargada de asegurar que el estado de una determinada tupla, no pase de un estado inicial a uno final, sin haber pasado por los estados intermedios. En nuestra BD no se encuentran restricciones de este tipo.

Integridad Referencial: Otra de las reglas, es la llamada restricción de integridad referencial que se especifica entre dos relaciones y sirve para mantener la consistencia entre tuplas de las dos relaciones. Establece que una tupla en una relación que haga referencia a otra, deberá referirse a un valor existente en esa relación. Un ejemplo de aplicación de este concepto en el modelo de datos es el siguiente: se asegura que los atributos idTorneo e idJugador de la tabla jugadorTorneo sean llaves

foráneas, debido a que provienen y son llaves primarias de las entidades torneo y jugador respectivamente. Además cumplen con las restricciones siguientes:

1. Los idJugador e idTorneo en la tabla jugadorTorneo tienen los mismos dominios que en sus entidades de origen.

2. Al introducir las llaves de torneo y jugador en la tabla jugadorTorneo, estos tienen que encontrarse en su entidad de origen, es decir, jugadorTorneo no puede tener un idTorneo e idJugador que no estén presentes en la tabla torneo y jugador respectivamente.

Por lo planteado anteriormente se comprueba que en el modelo de datos se cumple con la regla de integridad referencial, que es una de las más importantes dentro de la integridad de la BD.

3.1.2 Normalización

La normalización es el proceso mediante el cual se transforman datos complejos a un conjunto de estructuras de datos más pequeñas, que además de ser más simples y más estables, son más fáciles de mantener. También se puede entender la normalización como una serie de reglas que sirven para ayudar a los diseñadores de BD a desarrollar un esquema que minimice los problemas de lógica. Cada regla está basada en la que le antecede. La normalización se adoptó porque el viejo estilo de poner todos los datos en un solo lugar, como un archivo o una tabla de la BD, era ineficiente y conduce a errores de lógica cuando se trataban de manipular los datos. La normalización también hace las cosas fáciles de entender. Los seres humanos tenemos la tendencia de simplificar las cosas al máximo. Se hace con casi todo, desde los animales hasta con los automóviles. Se ve una imagen de gran tamaño y se hace más simple agrupando cosas similares juntas. Las guías que la normalización provee crean el marco de referencia para simplificar una estructura de datos compleja. [8]

Otra ventaja de la normalización de BD es el consumo de espacio. Una BD normalizada ocupa menos espacio en disco que una no normalizada. Hay menos repetición de datos, lo que tiene como consecuencia un mucho menor uso de espacio en disco.

El proceso de normalización tiene un nombre y una serie de reglas para cada fase. Esto puede parecer un poco confuso al principio, pero poco a poco se va entendiendo el proceso, así como las razones para hacerlo de esta manera.

Las reglas de normalización se relacionan entre ellas de forma dependiente, lo que indica que para que una BD se encuentre en una determinada forma normal, primeramente tiene que cumplir con las reglas de los niveles inferiores de normalización. Esto quiere decir que una BD esta en 2da Forma Normal si previamente cumple con las reglas de normalización del 1er nivel. Cada regla que se cumple aumenta el grado de normalización del esquema de relación. Si una regla no se cumple, el esquema se debe descomponer en varios esquemas de relación que si la cumplan por separado. Se dice que una BD se encuentra en determinada forma normal si cumple con las restricciones correspondientes a dicha forma normal.

Para los esquemas de relación propuesto para la BD se tuvo en cuenta la normalización donde se aplico al diagrama Modelo Entidad Relación (DER), basado en sus formas normales con el objetivo de analizar la relación de la BD, las dependencias funcionales entre sus atributos, de cumplir con una serie de requisitos para evitar las anomalías en las actualización, inserción, y eliminación, asegurando la consistencia de los datos, con vista a garantizar un adecuado diseño de la BD.

Existen varios niveles de normalización:

- Primera Forma Normal (1FN).
- Segunda Forma Normal (2FN).
- Tercera Forma Normal (3FN).
- Forma Normal Boyce-Codd.
- Cuarta Forma Normal.
- Quinta Forma Normal o Forma Normal de Proyección-Unión.
- Forma Normal de Proyección-Unión Fuerte.
- Forma Normal de Proyección-Unión Extra Fuerte y Forma Normal de Clave de Dominio.

Primera Forma Normal

La Primera Forma Normal, o 1FN, es la más elemental de todas. Una tabla está en 1FN si el valor que contiene un atributo de un registro, un campo, es único y elemental. En cada uno de los atributos solo se puede incluir un dato, aunque sea compuesto, pero no se pueden incluir una lista de datos.

Segunda Forma Normal

Una tabla está en Segunda Forma Normal o 2FN cuando está en 1FN y todo atributo que no pertenece a la clave primaria tiene una dependencia funcional de la clave completa y no de parte de ella. Luego, si la clave principal está formada por un solo atributo y ya está en 1FN, ya estará en 2FN.

Tercera Forma Normal

Una tabla está en Tercera Forma Normal o 3FN si está en 2FN y no existen atributos que no pertenezcan a la clave primaria que puedan ser conocidos mediante otro atributo que no forma parte de la clave primaria, es decir, no hay dependencias funcionales transitivas.

Se puede afirmar que los diseños de los módulos de la BD propuestos se encuentran normalizados hasta Tercera Forma Normal. Se puede plantear que los esquemas de relación para estos módulos están en 1ra Forma Normal puesto que se puede asegurar que cada tupla contiene exactamente un valor para cada atributo de las tablas de la BD, es decir, no existen campos multivaluados. También se cumple que los esquemas de relación están en 2da Forma Normal, porque primeramente se encuentran en 1ra Forma Normal, y además todos los atributos que no son claves en las tablas, dependen totalmente de la clave primaria. Finalmente se puede plantear que los esquemas de relación se encuentran en 3ra Forma Normal, porque primeramente se encuentran en 2da Forma Normal, y además que no existen dependencias transitivas entre llaves candidatas y atributos no primos.

Solo se llegará hasta la 3ra Forma Normal, debido a que un esquema relacional muy grande como el que se está desarrollando es muy difícil llevarlo a las otras formas normales y durante el proceso se puede crear características no deseadas.

3.1.3 Trazabilidad de las Acciones

La Trazabilidad es la cualidad que permite que todas las acciones realizadas sobre un sistema de tecnología de la información sean asociadas de modo inequívoco a un individuo o entidad. Además es

la capacidad que tiene una organización o sistema para rastrear, reconstruir o establecer relaciones entre objetos monitoreados, para identificar y analizar situaciones específicas o generales en los mismos. [9]

Para llevar un control de las acciones realizadas por los usuarios que interactúan con la BD se creó un grupo de tablas que funcionan como un historial donde se guardan todas las partidas con la hora de la partida los oponentes y las jugadas que se hicieron y con ellas se puede seguir la traza de un usuario dentro de la BD desempeñando el rol de jugador online, en este conjunto de tablas se guardaran los datos necesarios para poder hacer una auditoria a cualquier usuario y saber las acciones realizadas por el mismo con fecha y hora. Además se puede verificar con tablas como AccesoJugador se puede tener datos importantes de conexiones de un usuario como es el IP de donde realizó la conexión y la fecha de la cual hizo uso del servicio.

3.1.4 Análisis de redundancias

Se dice que hay redundancia de datos cuando la misma información es almacenada varias veces en la misma BD. Esto es siempre algo que se debe evitar, la redundancia dificulta la tarea de modificación de datos, y es el motivo más frecuente de inconsistencia de datos. Además requiere un mayor espacio de almacenamiento, que influye en mayor coste y mayor tiempo de acceso a los datos.

Después de llevar la BD a 3FN quedó libre de redundancias aunque en caso que se quisiera agrupar en relaciones más grandes la BD se podría utilizar la técnica de Síntesis Relacional que precisamente consiste en agrupar las relaciones en otras más grandes a conveniencia del proceso de negocio.

3.2 Análisis de la seguridad de la Base de Datos

La seguridad es un punto esencial en las BD para evitar ataques, impedir cualquier acceso no autorizado, con la intensión de modificar, usar y/o difundir información almacenada en las BD. El SGBD a utilizar para diseñar las BD incluye formas de restringir el acceso al sistema. Esta función se denomina control de acceso y se pone en prácticas creando cuentas de usuarios y contraseñas para que el SGBD controle el proceso de entrada al sistema.

El administrador de la BD (ABD) juega un papel importante en la seguridad de la BD porque es quien otorga privilegios a los usuarios, los clasifica, así como a los datos. Las órdenes de los ABD incluyen las siguientes acciones:

- Creación de cuentas.
- Concesión de privilegios.
- Revocación de privilegios.

La creación de cuentas sirve para controlar el acceso al SGBD en general, la concesión y revocación de privilegios para controlar autorizaciones discrecionales. Estas autorizaciones constituyen un importante mecanismo de seguridad.

Conclusiones

En el capítulo se han analizado una serie de aspectos y consideraciones que se deben tener en cuenta para realizar un diseño eficiente de una BD, como son: la integridad del diseño de la BD, la redundancia de información presente en la misma, además de la normalización, la cual en la BD propuesta se llevó hasta la tercera forma normal (3NF). También se analizó la seguridad que debe presentar la misma, realizando un control de acceso a la BD, usando la asignación de derechos a los usuarios como principal método de seguridad. Se realizó el análisis de la trazabilidad de las acciones para tener constancia de las acciones de los usuarios en la BD.

Conclusiones

En la UCI se desarrolla un proyecto basado en la informatización del ajedrez para la Universidad y el país, que ayudaría de forma inigualable al desarrollo de este deporte en Cuba, dicho proyecto requería de una BD que le permitiera archivar toda la información necesaria. Esta BD fue creada totalmente funcional, respecto a rendimiento, integridad, seguridad, libre de redundancias e inconsistencias. También permite seguir la trazabilidad de las acciones que se realizan sobre ella. Para la realización de este proyecto fue necesario resolver problemas de tipo organizativo, por lo que se hizo un estudio que dio como resultado la aprobación y la utilización de RUP como metodología de desarrollo de software, para realizar el modelado y el diseño de la BD se usó el DBDesigner, y el SGBD utilizado fue MySQL.

Al empezar esta investigación se trazaron tres objetivos principales que fueron vencidos, ellos son:

1. El estudio y preparación sobre el diseño e implementación de las BD llegando a conocer las mejores formas o vías de diseñar una BD acorde con lo que se necesita en el proyecto INFODREZ.

2. Se realizó el diseño de la BD relacional analizando detalladamente cada aspecto para que la misma cumpliera con los requerimientos planteados por la problemática.

3. Se realizó la implementación de acuerdo con el diseño realizado y se comprobó las utilidades de dicha base de datos.

Se puede concluir que este trabajo ha llevado a la realización satisfactoria de una BD capaz de satisfacer los requerimientos del proyecto.

Recomendaciones

El alcance de este trabajo está definido para darle servicio a los módulos del proyecto INFODREZ a desarrollar en esta etapa, por lo que se recomienda:

- Realizar el diseño de la BD que va a prestar servicios a los módulos que se van a desarrollar en un futuro del proyecto.
- El estudio del ajedrez como deporte para un mejor entendimiento de lo que necesitaría para el desarrollo posterior de dicha BD.

Referencias Bibliográficas

1. **Ruiz, Francisco.** El Modelo de Datos Gerárquico. [En línea] 21 de 04 de 2001. [Citado el: 03 de 02 de 2008.] http://alarcos.inf-cr.uclm.es/doc/bda/doc/trab/T0001_MAMoraga.pdf

2. **Dubois, Poul.** Las principales características de MySQL. [En línea] [Citado el: 20 de 04 de 2008.] http://dev.mysql.com/doc/refman/5.0/es/features.html

3. Características de SQL Server. [En línea] 2001. [Citado el: 20 de 03 de 2008.] http://www.sqlmax.com/caracter.asp

4. GNU Ferret. [En línea] 2001. [Citado el: 21 de 03 de 2008.] http://alts.homelinux.net/libreapp.php?id=83.

5. DBDesigner . [En línea] 2003. [Citado el: 10 de 04 de 2008.] http://www.fabforce.net/dbdesigner4/features.php .

6. **Sanchez, María A. Mendoza.** Metodologias de desarrollo de Software. [En línea] 07 de 06 de 2004. [Citado el: 29 de 03 de 2008.] http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

7. Software Libre Para Sitios Web.phpMyAdmin. [En línea] 2005. [Citado el: 15 de 03 de 2008.] http://img.redusers.com/imagenes/libros/lpcu068/capitulogratis.pdf .

8. **Eduardo.** Qué es la normalización. [En línea] 2003. [Citado el: 02 de 05 de 2008.] http://www.mysql-hispano.org/page.php?id=16&pag=1 .

9. **Cano, Jeimy J.** Trazabilidad de las Operaciones Electrónicas. [En línea] 2005. [Citado el: 07 de 05 de 2008.] http://www.isaca.org/Template.cfm?Section=Home&CONTENTID=24655&TEMPLATE=/ContentManagement/ContentDisplay.cfm .

Bibliografía

1. Análisis de Optimización de Consultas. [En línea] 2003. [Citado el: 26 de 03 de 2008.] http://dis.um.es/~mjortin/ibd_temas/ibd_t6_opti.pdf.

2. Bases de Datos. [En línea] 2002. [Citado el: 15 de 03 de 2008.] http://www.silicontower.net/alojamiento- web/caracteristicas-detalladas/bases-de-datos .

3. **Cano, Jeimy J.** Trazabilidad de las Operaciones Electrónicas. [En línea] 2005. [Citado el: 07 de 05 de 2008.] http://www.isaca.org/Template.cfm?Section=Home&CONTENTID=24655&TEMPLATE=/ContentManagement/ContentDisplay.cfm .

4. Características de SQL Server. [En línea] 2001. [Citado el: 20 de 03 de 2008.] http://www.sqlmax.com/caracter.asp

5. DBDesigner . [En línea] 2003. [Citado el: 10 de 04 de 2008.] http://www.fabforce.net/dbdesigner4/features.php .

6. DBDesigner. The Linux/KDE2 visual database designer. [En línea] 2005. [Citado el: 26 de 03 de 2008.] http://dbdesigner.sourceforge.net/ .

7. **D. Francisco Ruiz, M. Ángeles Moraga de la Rubia.** EL MODELO DE DATOS JERÁRQUICO. [En línea] 2001. [Citado el: 12 de 03 de 2008.] http://alarcos.inf-cr.uclm.es/doc/bda/doc/trab/T0001_MAMoraga.pdf .

8. **Dubois, Poul.** Las principales características de MySQL. [En línea] [Citado el: 20 de 04 de 2008.] http://dev.mysql.com/doc/refman/5.0/es/features.html

9. **Eduardo.** Qué es la normalización. [En línea] 2003. [Citado el: 02 de 05 de 2008.] http://www.mysql-hispano.org/page.php?id=16&pag=1 .

10. **Gary W. Hansen, James V. Hasen.** *Diseño y Administración de Base de Datos.* 2da Ed. s.l. : Prentice Hall, 1997.

11. GNU Ferret. [En línea] 2001. [Citado el: 21 de 03 de 2008.] http://alts.homelinux.net/libreapp.php?id=83.

12. **GUZMÁN, IGNACIO GARCÍA RODRÍGUEZ DE.** BASES DE DATOS. MODELO EN RED GENERAL. [En línea] 2001. [Citado el: 25 de 02 de 2008.] http://alarcos.inf-cr.uclm.es/doc/bda/doc/trab/T0001_IGarcia.pdf .

13. **Hernán.** Tutorial básico de bases de datos. [En línea] 2005. [Citado el: 21 de 04 de 2008.] http://www.cristalab.com/tutoriales/75/tutorial-basico-de-bases-de-datos .

14. **I. Jacobson, G Booch.** *El proceso unificado de desarrollo de software.* 2000.

15. Instalación de programas Open source. [En línea] 2004. [Citado el: 12 de 03 de 2008.] http://www.dixian.info/cms/es/instalacion-de-programas-open-source.html .

16. Modelo de red. [En línea] 2005. [Citado el: 20 de 02 de 2008.] http://enciclopedia.us.es/index.php/BD_de_red .

17. MySQL 5.0 Reference Manual. [En línea] 2007. [Citado el: 03 de 05 de 2008.] http://dev.mysql.com/doc/refman/5.0/es/index.html .

18. **Rodolfo Franco.** BASES DE DATOS RELACIONALES. [En línea] 2007. [Citado el: 10 de 03 de 2008.] http://www.udistrital.edu.co/comunidad/profesores/rfranco/bd_rel.htm .

19. **Ruiz, Francisco.** El Modelo de Datos Gerárquico. [En línea] 21 de 04 de 2001. [Citado el: 03 de 02 de 2008.] http://alarcos.inf-cr.uclm.es/doc/bda/doc/trab/T0001_MAMoraga.pdf

20. **Sanchez, María A. Mendoza.** Metodologias de desarrollo de Software. [En línea] 07 de 06 de 2004. [Citado el: 29 de 03 de 2008.] http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

21. Software Libre Para Sitios Web.phpMyAdmin. [En línea] 2005. [Citado el: 15 de 03 de 2008.] http://img.redusers.com/imagenes/libros/lpcu068/capitulogratis.pdf .

22. The phpMyAdmin Project. [En línea] 2005. [Citado el: 16 de 02 de 2008.] http://www.phpmyadmin.net/home_page/index.php .

Anexos

Anexo 1. Descripción de las clases persistentes.

Tabla 13.- Clase Correo.

Nombre: Correo	
Atributo	Tipo
idCorreo	integer
tipoCorreo	string
asunto	string
texto	string

Tabla 14.- Clase Organizador.

Nombre: Organizador	
Atributo	**Tipo**
idOrganizador	integer
nombre	string
correo	string
direccion	string

Tabla 15.- Clase ModeloLogística.

Nombre: ModeloLogistica	
Atributo	**Tipo**
idModelo	integer
nombre	string
direccionModelo	string

Tabla 16.- Clase Noticia.

Nombre: Noticia	
Atributo	**Tipo**
idNoticia	integer
titulo	string
referencia	string
cuerpoNoticia	string
fechaPublicacion	timestamp
fechaReintegro	timestamp
imagen	string

Tabla 17.- Clase Usuario.

Nombre: Usuario	
Atributo	**Tipo**
idUsuario	integer
usuario	string
password	string
rol	string

Tabla 18.- Clase Evento.

Nombre: Evento	
Atributo	**Tipo**
idEvento	integer
nombre	string
fechaInicio	timestamp
fechaFin	timestamp

Tabla 19.- Clase jugadores (online).

Nombre: jugadores	
Atributo	**Tipo**
nombre	string
usuario	string
password	string
ocupacion	string
ubicacion	string
correo	string
fechaRegistro	date
nivel	int
registrado	bool
idFIDE	int
tituloFIDE	string
eloFIDE	int

Tabla 20.- Clase jugadorDeTorneo.

Nombre: jugadorDeTorneo	
Atributo	**Tipo**
flotador	bool
numeroOrden	int
fechaRegistro	date
eloInscripcion	int

Tabla 21.- Clase Grupo.

Nombre: Grupo	
Atributo	**Tipo**
idGrupo	integer
nombre	string
tipoTorneo	string
ritmoJuego	string

Tabla 22.- Clase juegos (online).

Nombre: juegos	
Atributo	**Tipo**
idJuego	int
tiempo	int
mensajes	string
fecha	date
planilla	string
visitas	int
eloNegro	int
eloBlanco	int
Bonificacion	Int
ECO	string
apertura	string
varianteApertura	string
resultado	string

Anexo 2. Descripción de las tablas de la Base de Datos.

Tabla 23.-federacion.

Nombre: federacion		
Descripción: Es un nomenclador que almacena toda la información referente a una Federación.		
Atributo	**Tipo**	**Descripción**
IdFederacion	INTEGER	Id de la federación.
Nombre	VARCHAR(30)	Nombre de la federación.

Tabla 24.- tituloJugador.

Nombre: tituloJugador		
Descripción: Tiene la función de nomenclador con los títulos que puede tener un jugador.		
Atributo	**Tipo**	**Descripción**
idTituloJugador	INTEGER	Id del título del jugador.
Nombre	VARCHAR(30)	Nombre del título de jugador.

Tabla 25.- usuario.

Nombre: usuario		
Descripción: Almacena la información sobre los usuarios del sistema.		
Atributo	**Tipo**	**Descripción**
IdUsuario	INTEGER	Id del usuario.
nombre	VARCHAR(30)	Nombre del usuario.
passwd	VARCHAR(255)	Contraseña del usuario.
correo	VARCHAR(255)	Correo del usuario.
fechaRegistro	TIMESTAMP	Fecha de registrado el usuario.
registrado	INTEGER	Es para conocer si el usuario es del dominio.
nivel	INTEGER	Nivel del usuario.

Tabla 26.- tituloArbitro.

Nombre: tituloArbitro		
Descripción: Tiene la función de nomenclador con los títulos que puede tener un árbitro.		
Atributo	**Tipo**	**Descripción**
idTituloArbitro	INTEGER	Id del titulo de árbitro.
Nombre	VARCHAR(30)	Nombre del titulo del árbitro.

Tabla 27.- aperturas.

Nombre: aperturas		
Descripción: Almacena información sobre las aperturas.		
Atributo	**Tipo**	**Descripción**
Id	INTEGER	Id de la apertura.
nombre	VARCHAR(30)	Nombre de la apertura.
ECO	VARCHAR(10)	Estrategia de apertura.
variante	VARCHAR(30)	Variante de la apertura.
pgn	LONGTEXT	Preguntar.
transposicion	VARCHAR(30)	Si es por transposición o no la apertura.

Tabla 28.- imagen.

Nombre: imagen		
Descripción: Almacena información sobre las imágenes que tendrán las noticias.		
Atributo	**Tipo**	**Descripción**
IdImagen	INTEGER	Id de la imagen.
idNoticia(FK)	INTEGER	Id de la noticia la que pertenece.
Titulo	VARCHAR(30)	Titulo de la imagen.
comentario	LONGTEXT	Comentario sobre la imagen.
direccionImagen	VARCHAR(255)	Dirección de la imagen.

Tabla 29.- mensajeCorreo.

Nombre: mensajeCorreo		
Descripción: Almacena información sobre los mensajes de correo.		
Atributo	**Tipo**	**Descripción**
idMensajeCorreo	INTEGER	Id del mensaje.
idAsunto(FK)	INTEGER	Id del asunto.
idTipoMensaje(FK)	INTEGER	Id del tipo de mensaje.
Texto	LONGTEXT	Texto del mensaje.

Tabla 30.- arbitroTorneo.

Nombre: arbitroTorneo		
Descripción: Se crea de la relación entre tabla árbitro y torneo.		
Atributo	**Tipo**	**Descripción**
IdTorneo	INTEGER	Id del torneo.
IdArbitro	INTEGER	Id del árbitro.

Tabla 31.- notificacion.

Nombre: notificacion		
Descripción: Almacena las notificaciones hechas a los usuarios..		
Atributo	**Tipo**	**Descripción**
idNotificacion	INTEGER	Id de la notificación.
fechaActivacion	TIMESTAMP	Fecha donde se activa la notificación.
FechaFin	TIMESTAMP	Fecha final de la notificación.
Mensaje	LONGTEXT	Mensaje de la notificación.
IdOrigen	INTEGER	De donde viene la notificación.

Tabla 32.- evento.

Nombre: evento		
Descripción: Almacena información sobre los eventos y actividades que se realizan en los torneos.		
Atributo	**Tipo**	**Descripción**
IdEvento	INTEGER	Id del evento.
idTorneo(FK)	INTEGER	Id del torneo.
Nombre	VARCHAR(30)	Nombre del evento.
fechaInicio	TIMESTAMP	Fecha de inicio del evento.
FechaFin	TIMESTAMP	Fecha de fin del evento.

Tabla 33.- modeloLogistica.

Nombre: modeloLogistica		
Descripción: Almacena información sobre los mensajes de correo.		
Atributo	**Tipo**	**Descripción**
idModeloLogistica	INTEGER	Id del modelo de logística.
Nombre	VARCHAR(30)	Nombre del modelo.
direccionModelo	VARCHAR(255)	Dirección del modelo donde se encuentra el modelo que es un documento.

Tabla 34.- asunto.

Nombre: asunto		
Descripción: Tiene la función de nomenclador con los asuntos que puede tener un mensaje.		
Atributo	**Tipo**	**Descripción**
IdAsunto	INTEGER	Id del asunto.
Texto	VARCHAR(255)	Texto del asunto.

Tabla 35.- partidaPrivada.

Nombre: partidaPrivada		
Descripción: Almacena información sobre las partidas privadas.		
Atributo	**Tipo**	**Descripción**
IdPr	INTEGER	Id de la partida pública.
IdPropone	INTEGER	Id del jugador online que propone la partida.
IdDestino	INTEGER	Id del jugador online al que se propone la partida.
Color	VARCHAR(30)	Color de las piezas del que propone la partida.
Tiempo	INTEGER	Tiempo máximo de duración de la partida.
Oficial	SMALLINT	Si es oficial o no la partida.
bonificacion	INTEGER	Es la bonificación de la apartida que se da en la partida.
Fecha	TIMESTAMP	Fecha de la partida.

Tabla 36.- chat.

Nombre: chat		
Descripción: Tiene la función de almacenar información del Chat.		
Atributo	**Tipo**	**Descripción**
IdChat	INTEGER	Id del chat.
IdOrigen	INTEGER	Id del usuario que manda el mensaje.
IdDestino	INTEGER	Id del usuario que recibe.
Mensaje	VARCHAR(255)	Texto del mensaje.
Hora	TIMESTAMP	Hora del mensaje.
Leido	SMALLINT	Si fue leido o no el mensaje.

Tabla 37.- partidaPublica.

Nombre: partidaPublica		
Descripción: Almacena información sobre las partidas públicas.		
Atributo	**Tipo**	**Descripción**
IdPu	INTEGER	Id de la partida pública.
idPropuesta	INTEGER	Id del jugador que la propone que la propone.
eloMaximo	INTEGER	Elo máximo.
EloMinimo	INTEGER	Elo mínimo.
Color	VARCHAR(30)	Color de las piezas del que propone la partida.
Tiempo	INTEGER	Tiempo de duración de la partida.
Oficial	SMALLINT	Si es oficial o no la partida.
bonificacion	INTEGER	Es la bonificación que se da en la partida al ganador.
Fecha	TIMESTAMP	Fecha de la partida.

Tabla 38.- jugadorTorneo.

Nombre: jugadorTorneo		
Descripción: Es la relación entre las tablas jugador y torneo.		
Atributo	**Tipo**	**Descripción**
id	INTEGER	Id del jugador.
idTorneo	INTEGER	Id del torneo en que ha participado cada jugador.
idGrupo(FK)	INTEGER	Id del grupo al que pertenece el jugador en dicho torneo.
numeroSorteo	INTEGER	Número de sorteo del jugador en dicho torneo.
K	ENUM('25', '15', '10')	Constante para el cálculo del Elo.

Tabla 39.- noticia.

Nombre: noticia		
Descripción: Almacena las noticias relacionadas con los sucesos de un torneo.		
Atributo	**Tipo**	**Descripción**
IdNoticia	INTEGER	Id de la noticia.
numRonda	INTEGER	Id de la ronda de la cual se realiza la noticia.
Titulo	VARCHAR(30)	Título de la noticia publicada.
fechaPublicación	TIMESTAMP	Fecha de publicación de la noticia.
fechaReintegro	TIMESTAMP	Preguntar
cuerpoNoticia	LONGTEXT	Es el cuerpo de la noticia, el texto completo.
Peso	INTEGER	Peso de la noticia.

Tabla 40.- tipoTorneo.

Nombre: tipoTorneo		
Descripción: Es un nomenclador con sobre los tipos de torneo que existen.		
Atributo	**Tipo**	**Descripción**
idTipoTorneo	INTEGER	Id del tipo de torneo.
Nombre	VARCHAR (30)	Nombre del tipo torneo.

Tabla 41.- privilegios.

Nombre: privilegios		
Descripción: Tiene datos de los privilegios de los usuarios del sistema.		
Atributo	**Tipo**	**Descripción**
Nivel	INTEGER	Nivel de privilegios.
descripcion	VARCHAR(255)	Descripción del nivel.
Nombre	VARCHAR(30)	Nombre del privilegio.

Tabla 42.- ritmoJuego.

Nombre: ritmoJuego		
Descripción: Tabla que se usa para nomenclador el ritmo de juego.		
Atributo	**Tipo**	**Descripción**
idRitmoJuego	INTEGER	Id del ritmo de juego.
Nombre	VARCHAR(30)	Nombre del ritmo de juego.

Tabla 43.- sessionJugador.

Nombre: sessionJugador		
Descripción: Almacena los datos de la sesión de un usuario.		
Atributo	**Tipo**	**Descripción**
IdA	INTEGER	Id de la sesión.
FechaAcc	TIMESTAMP	Fecha de acceso de la sesión al sistema.
IdPagina	INTEGER	Id de la página en que se encuentra el usuario.

Tabla 44.- jugadorExpulsado.

Nombre: jugadorExpulsado		
Descripción: Almacena datos de los jugadores online expulsados por mala conducta.		
Atributo	**Tipo**	**Descripción**
id	INTEGER	Id del jugador expulsado.
idUsuario	INTEGER	Id del usuario del jugador.
fecha	TIMESTAMP	Fecha en que fue expulsado el jugador.
causa	VARCHAR(255)	Causa por la cual fue expulsado.
descripcion	VARCHAR(255)	Descripción de la causa.
tiempo	TIMESTAMP	Tiempo que fue expulsado.

Tabla 45.- sistemaJuego.

Nombre: sistemaJuego		
Descripción: Almacena toda la información de los sistemas de juego que existen.		
Atributo	**Tipo**	**Descripción**
idSistemaJuego	INTEGER	Id del sistema de juego.
Nombre	VARCHAR(30)	Nombre del sistema de juego.
cantidadMinimaJugadores	INTEGER	Es la cantidad mínima de jugadores del sistema de juego.
cantidadMaximaJugadores	INTEGER	Es la cantidad máxima de jugadores del sistema de juego.

Tabla 46.- jugadorOnline.

Nombre: jugadorOnline		
Descripción: Guarda los datos de la unión de la tabla jugador y usuario que aquí se convierte en Jugador Online.		
Atributo	**Tipo**	**Descripción**
idUsuario	INTEGER	Id del usuario.
id	INTEGER	Id del jugador.

Tabla 47.- ritmoJuego.

Nombre: ritmoJuego		
Descripción: Es un nomenclador que almacena información sobre los ritmos de juegos que existen.		
Atributo	**Tipo**	**Descripción**
idRitmoJuego	INTEGER	Ritmo de juego su id.
Nombre	VARCHAR (30)	Nombre del ritmo de juego.

Tabla 48.- ciudad.

Nombre: ciudad		
Descripción: Es un nomenclador que almacena información sobre las ciudades en que se realizan los torneos.		
Atributo	**Tipo**	**Descripción**
IdCiudad	INTEGER	Id de la ciudad.
Nombre	VARCHAR (30)	Nombre de la ciudad.

Tabla 49.- accesoJugador.

Nombre: accesoJugador		
Descripción: Almacena los datos de conexión de un jugador al juego Online.		
Atributo	**Tipo**	**Descripción**
IdUsuario	INTEGER	Id del usuario del sistema que se ha conectado.
Fecha	TIMESTAMP	Fecha en la cual se realizó la conexión.
navegador	VARCHAR(30)	Navegador con el cual se conectó el usuario.
Ip	VARCHAR(15)	Ip de donde se realizó la conexión.

Tabla 50.- preferenciaJugador.

Nombre: preferenciaJugador		
Descripción: Se almacena la relación del jugador online con los juegos que prefiere.		
Atributo	**Tipo**	**Descripción**
IdJuego	INTEGER	Id del juego que prefiere el jugador.
Id	INTEGER	Id del jugador.
IdUsuario	INTEGER	Id del usuario del jugador.

Tabla 51.- arbitroOnline.

Nombre: arbitroOnline		
Descripción: Almacena los datos de árbitros Online.		
Atributo	**Tipo**	**Descripción**
IdArbitro	INTEGER	Id del árbitro online.
IdUsuario	INTEGER	Id del usuario que utiliza el árbitro.

Tabla 52.- prefJugador.

Nombre: prefJugador		
Descripción: Almacena las preferencias en el tablero del jugador online.		
Atributo	**Tipo**	**Descripción**
Id	INTEGER	Id del jugador.
IdUsuario	INTEGER	Id del usuario del jugador.
temaPieza	VARCHAR(30)	Tema de la pieza que utiliza el jugador.
temaTablero	VARCHAR(30)	Tema del tablero que utiliza el jugador.
temaEstilo	VARCHAR(30)	Tema del estilo de la página.
colorCasillaMarcada	VARCHAR(30)	Color de la casilla marcada.
colorCasillaJugada	VARCHAR(30)	Color de la casilla jugada.
colorCasillaPremove	VARCHAR(30)	Color de la casilla que se marca para la próxima jugada.
colorCasillaPosicional	VARCHAR(30)	Color de la casilla en que se encuentra la pieza.
anchoColor	INTEGER	Ancho del color de la celda.

Tabla 53.- tipoMensaje.

Nombre: tipoMensaje		
Descripción: Almacena los tipos de mensaje que puede ser un correo.		
Atributo	**Tipo**	**Descripción**
idTipoMensaje	INTEGER	Id del tipo de mensaje.
Texto	VARCHAR(30)	Texto del tipo de mensaje.

Tabla 54.- juegoCompletado.

Nombre: juegoCompletado		
Descripción: Se guardan los datos de las partidas online completadas.		
Atributo	**Tipo**	**Descripción**
idJuego	INTEGER	Id del juego completado.
idPartida	INTEGER	Id de partida.
tiempo	INTEGER	Tiempo utilizado en el juego.
mensaje	VARCHAR(255)	Mensaje enviado por el sistema al terminar la partida.
fecha	TIMESTAMP	Fecha en que se jugó la partida.
visitas	INTEGER	Cantidad de visitas realizadas a la partida.
eloBlanco	INTEGER	Elo del jugador de piezas blancas.
eloNegro	INTEGER	Elo del jugador de piezas negras.
bonificacion	INTEGER	Bonificación obtenida por el jugador ganador.
ECO	VARCHAR(255)	Estrategia de apertura.
apertura	VARCHAR(255)	Apertura utilizada en la partida.
varianteApertura	VARCHAR(255)	Variante de la apertura utilizada.

Tabla 55.- juegoCompletadoTorneo.

Nombre: juegoCompletadoTorneo		
Descripción: Es el resultado de la relación entre las tablas juegoCompletado y torneo.		
Atributo	**Tipo**	**Descripción**
idPartida	INTEGER	Id de la partida.
numRonda	INTEGER	Id de la ronda.
idJuego	INTEGER	Id del juego completado.

Tabla 56.- jugadorTorneoOnline.

Nombre: jugadorTorneoOnline		
Descripción: Es la relación entre las tablas jugador y torneoOnline y guarda los datos de dicha relación.		
Atributo	**Tipo**	**Descripción**
Id	INTEGER	Id del jugador.
idUsuario	INTEGER	Id del usuario de dicho jugador.
idTorneo	INTEGER	Id del torneo.
idTorneoOnline	INTEGER	Id del torneo online.
flotador	INTEGER	Condición de que sea flotador en el torneo.
numeroOrden	INTEGER	Número de orden del jugador.
fechaRegistro	TIMESTAMP	Fecha de registro del jugador en el torneo.
elo	INTEGER	Elo alcanzado por el jugador en el torneo online.

Tabla 57.- estadistJugador.

Nombre: estadistJugador		
Descripción: Almacena las estadísticas de los jugadores Online, como juegos ganados, perdidos y hechos tabla.		
Atributo	**Tipo**	**Descripción**
Id	INTEGER	Id del jugador.
IdUsuario	INTEGER	Id del usuario del jugador.
ganados1	INTEGER	Juegos ganados de un minuto de duración.
Perdidos1	INTEGER	Juegos perdidos de un minuto de duración.
tablas1	INTEGER	Juegos hechos tabla de un minuto de duración.
ganados3	INTEGER	Juegos ganados de 3 minutos de duración.
Perdidos3	INTEGER	Juegos perdidos de 3 minutos de duración.
tablas3	INTEGER	Juegos hechos tabla de 3 minutos de duración.

ganados5	INTEGER	Juegos ganados de 5 minutos de duración.
Perdidos5	INTEGER	Juegos perdidos de 5 minutos de duración.
tablas5	INTEGER	Juegos hechos tabla de 5 minutos de duración.
ganadosM	INTEGER	Juegos ganados de duración ilimitada.
perdidosM	INTEGER	Juegos perdidos de duración ilimitada.
TablasM	INTEGER	Juegos hechos tabla de duración ilimitada.
elo1	INTEGER	Elo del jugador Online en juegos de un minuto.
elo3	INTEGER	Elo del jugador Online en juegos de 3 minutos.
elo5	INTEGER	Elo del jugador Online en juegos de 5 minutos.
Eloy	INTEGER	Elo del jugador Online en juegos de tiempo ilimitado.

Tabla 58.- juegoAplazado.

Nombre: juegoAplazado		
Descripción: Almacena los datos de las partidas online que han sido aplazadas.		
Atributo	**Tipo**	**Descripción**
idJuego	INTEGER	Id del juego aplazado.
fecha	TIMESTAMP	Fecha en que fue aplazada la partida.
idBlanco	INTEGER	Id del jugador de piezas blancas.
idNegro	INTEGER	Id del jugador de piezas negras.
numMov	INTEGER	Cantidad de movimientos que se han realizado en la partida.
colorTurno	INTEGER	Color de las piezas que le corresponden jugar.
localizacion	LONGTEXT	Localización de las piezas en el tablero.
noActivasBlanco	VARCHAR(255)	Almacena las piezas blancas que están fuera de juego.
noActivasNegro	VARCHAR(255)	Almacena las piezas negras que están fuera de juego.
ganador	VARCHAR(255)	Al retomarse la partida guarda el ganador.

petTablas	INTEGER	Al retomarse la partida guarda jugador que hace petición de tablas.
rendido	INTEGER	Al retomarse la partida guarda jugador que se rinde.
tTBlanco	INTEGER	Tiempo total con que comenzó a jugar la partida el jugador de piezas blancas.
tTNegro	INTEGER	Tiempo total con que comenzó a jugar la partida el jugador de piezas negras.
tBlanco	INTEGER	Tiempo en que empezó a jugar la partida el jugador de piezas blancas.
tNegro	INTEGER	Tiempo en que empezó a jugar la partida el jugador de piezas negras.
tBonoBlanco	INTEGER	Bonificación del jugador de piezas blancas antes de comenzar la partida.
tBonoNegro	INTEGER	Bonificación del jugador de piezas negras antes de comenzar la partida.
historia	LONGTEXT	Historia de todos los sucesos de la partida.
oficial	SMALLINT	Verificar si la partida es oficial que influya en el Elo del jugador.
mensaje	VARCHAR(255)	Mensaje enviado por el sistema al terminar la partida.
ultimaJugada	VARCHAR(30)	Ultima jugada realizada en la partida.
visitas	INTEGER	Cantidad de visitas realizadas a la partida.
posiciones	LONGTEXT	Posiciones de las piezas en el tablero.
ECO	VARCHAR(10)	Estrategia de apertura.
apertura	VARCHAR(255)	Apertura que se utilizó en la partida.
varianteApertura	VARCHAR(255)	Variante de la apertura que se utilizó en la partida.

Glosario

ACID (Atomicity, Consistency, Isolation, Durability): En español: Atomicidad, Consistencia, Aislamiento y Durabilidad, propiedades importantes de las Base de Datos relacionales.

Atributo: Es cada una de las cualidades, propiedades o características de un elemento.

Clase: Es la declaración o abstracción de objetos, lo que significa, que una clase es la definición de un objeto. Cuando se programa un objeto y se definen sus características y funcionalidades, realmente se programa una clase.

CMS (Content Management System): Sistema de gestión de contenidos. Aplicación de software que simplifica hacer el diseño, las pruebas y el envío de contenidos en páginas Web.

Elo: Es un método para calcular la fuerza relativa de los jugadores de juegos como el ajedrez.

GPL: Licencia de regulación de los derechos de autor de los programas de software libre (free software).

Hardware: Componentes físicos de un ordenador o de una red, en contraposición con los programas o elementos lógicos que los hacen funcionar.

Herramienta CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador): Las Herramientas CASE son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero.

Hosting: Alojamiento. Servicio ofrecido por algunos proveedores, que brindan a sus clientes (individuos o empresas) un espacio en su servidor para alojar un sitio Web.

Normalización: Proceso de reducción sobre una estructura de datos que procura aumentar la integridad, disminuir la redundancia y las dependencias funcionales de esa estructura.

OpenSource (Código Abierto): Calificación de software que cumple una serie de requisitos, principalmente aquel que permite una libre redistribución, distribuye el código fuente, y permite modificaciones y trabajos derivados.

Redundancia: Repetición de una información ya dada en el mensaje.

Rol: Papel desempeñado por las personas en la sociedad, de esta misma forma funciona en los sistemas informáticos.

Scripts: Un conjunto de comandos escritos en un lenguaje interpretado para automatizar ciertas tareas de aplicación.

Servidor: Aplicación informática o programa que realiza algunas tareas en beneficio de otras aplicaciones llamadas clientes.

SGBD (Sistema Gestor de Base de Datos): Conjunto coordinado de programas, procedimientos, lenguajes. Que suministra tanto a usuarios no informáticos como a los analistas, programadores o al administrador, los medios necesarios para describir, recuperar y manipular los datos, manteniendo su integridad, confidencialidad y seguridad.

Software: Componente intangible de una computadora, es decir, un conjunto de programas y procedimientos necesarios para hacer posible la realización de una tarea específica.

Software libre: Es el software que, una vez obtenido, puede ser usado, copiado, estudiado, modificado y redistribuido libremente.

SQL (Structured Query Language): Es un lenguaje de acceso a las Base de Datos, permite especificar todas las operaciones sobre la Base de Datos como por ejemplo: Inserción, Borrado, Actualización. Utiliza características de álgebra y cálculo relacional permitiendo de esta forma realizar consultas a la Base de Datos de forma sencilla.

SSH (Secure SHell): Protocolo de conexión segura a los servidores. Igual que Telnet pero los datos se envían y reciben encriptados.

Tecnología: Conjunto de teorías y de técnicas que permiten el aprovechamiento práctico del conocimiento científico.

Tupla: Es una hilera o fila en una tabla.