

---

# Proyecto Bases de Datos II

*Comunidades musicales distribuidas.*

---

Luca Conesa Martín-Aragón y Juan Pedro Bolívar Puente

---

Universidad de Granada

*Curso 07/08*

# Índice general

<b>1. Presentación</b>	<b>7</b>
1.1. Introducción . . . . .	7
1.2. ¿Qué ofrecemos? . . . . .	8
<b>2. Requisitos</b>	<b>11</b>
2.1. Requisitos funcionales . . . . .	11
2.1.1. Gestión de usuarios . . . . .	11
2.1.2. Gestión de música . . . . .	12
2.1.3. Gestión de Comunidades . . . . .	14
2.1.4. Listas de comunidades . . . . .	16
2.1.5. Gestión de entradas . . . . .	16
2.2. Requisitos no funcionales . . . . .	17
2.2.1. Hardware . . . . .	17
2.2.2. Software . . . . .	17
2.2.3. Seguridad . . . . .	17
2.2.4. Prestaciones . . . . .	18
<b>3. Diagramas de Flujos de Datos</b>	<b>19</b>
3.1. DFD de contexto . . . . .	19
3.2. DFD de nivel 1 . . . . .	20
3.3. DFD de nivel 2 . . . . .	21
3.3.1. Usuarios . . . . .	21
3.3.2. Musica . . . . .	21
3.3.3. Conciertos . . . . .	22
3.3.4. Artistas . . . . .	22
<b>4. Casos de uso</b>	<b>23</b>
4.1. Diagrama . . . . .	24

## ÍNDICE GENERAL

---

4.2. Alta en el sistema . . . . .	24
4.3. Identificarse en el sistema . . . . .	25
4.4. Modificar ficha usuario propia . . . . .	25
4.5. Modificar perfil usuario propio . . . . .	26
4.6. Acceder lista de comunidades existentes . . . . .	26
4.7. Acceder lista de comunidades del usuario . . . . .	27
4.8. Crear una comunidad . . . . .	27
4.9. Unirse a una comunidad . . . . .	28
4.10. Irse de una comunidad . . . . .	28
4.11. Acceder a una comunidad . . . . .	28
4.12. Dejar un mensaje a la comunidad . . . . .	29
4.13. Acceder a la información de usuarios de su comunidad . . . . .	29
4.14. Dejar mensaje privado a un usuario . . . . .	30
4.15. Dejar mensaje público a un usuario . . . . .	30
4.16. Expulsar a un usuario de una comunidad . . . . .	31
4.17. Nombrar administrador a un usuario de la comunidad . . . . .	31
4.18. Búsqueda de Canciones . . . . .	32
4.19. Mostrar Lista de reproducción . . . . .	32
4.20. Mostrar Lista de reproducción ajena . . . . .	33
4.21. Borrar Lista de reproducción . . . . .	33
4.22. Agregar canción Lista de reproducción . . . . .	34
4.23. Agregar canción a nueva Lista de reproducción . . . . .	34
4.24. Borrar canción de Lista de reproducción . . . . .	35
4.25. Puntuar Canción . . . . .	36
4.26. Añadir Tag a Canción . . . . .	36
4.27. Reproducir Canción . . . . .	37
4.28. Reproducir Canción en modo Smart-Radio . . . . .	37
4.29. Descargar Cancion . . . . .	38
4.30. Subir Cancion . . . . .	38

## ÍNDICE GENERAL

---

4.31. Descargar-isntallar plugin Plugin . . . . .	39
4.32. Acceder a la información del artista . . . . .	39
4.33. Añadir Tag a Artista . . . . .	40
4.34. Consultar la información del artista . . . . .	40
4.35. Modificar la información del artista . . . . .	40
4.36. Indica que la información del artista está equivocada . . . . .	41
4.37. Acceder a bandeja de correo . . . . .	41
4.38. Leer mensaje . . . . .	42
4.39. Borrar mensaje . . . . .	42
4.40. Acceder a zona de conciertos . . . . .	43
4.41. Vender Entrada . . . . .	43
4.42. Comprar Entrada . . . . .	44
4.43. Consultar información de conciertos . . . . .	45
4.44. Ingresar información de conciertos . . . . .	46
<b>5. Modelo Entidad/Relación</b>	<b>47</b>
<b>6. Diccionario de datos</b>	<b>49</b>
6.1. ENTIDADES . . . . .	49
6.1.1. MENSAJERO . . . . .	49
6.1.2. CANCIÓN . . . . .	49
6.1.3. USUARIO . . . . .	50
6.1.4. USR-ARTISTA . . . . .	50
6.1.5. ARTISTA . . . . .	50
6.1.6. MENSAJE . . . . .	51
6.1.7. COMUNIDAD . . . . .	51
6.1.8. TAG . . . . .	51
6.1.9. TAGGEABLE . . . . .	52
6.1.10. CONCIERTO . . . . .	52
6.1.11. LOCAL . . . . .	52

## ÍNDICE GENERAL

---

6.1.12. ENTRADA . . . . .	53
6.2. RELACIONES . . . . .	53
6.2.1. ESCUCHA . . . . .	53
6.2.2. ENLISTA . . . . .	53
6.2.3. PUNTUA . . . . .	54
6.2.4. TIENE . . . . .	54
6.2.5. CREA . . . . .	55
6.2.6. POSEE . . . . .	55
6.2.7. TOCA . . . . .	55
6.2.8. LUGAR . . . . .	56
6.2.9. COMPRA . . . . .	56
6.2.10. ES-MIEMBRO . . . . .	56
6.2.11. ADMINISTRA . . . . .	57
6.2.12. ENVIA . . . . .	57
6.2.13. Recibe . . . . .	58
6.2.14. Bandeja . . . . .	58
<b>7. Normalización</b>	<b>59</b>
7.1. De Modelo E/R a Relacional . . . . .	59
7.2. Análisis de la Forma Normal de las relaciones del sistema .	59
7.2.1. Relaciones . . . . .	60
7.3. Código de creación de la base de datos . . . . .	62

# 1

## Presentación

### 1.1 Introducción

---

La revolución de los sistemas de información y la democratización de internet han provocado cambios sustanciales en la distribución de contenidos culturales, poniendo en tela de juicio la validez de los modelos de negocio que se habían implantado con la masificación del vinilo y la explosión de la cultura pop a mediados del siglo pasado. Algunas implicaciones de éste hecho son las siguientes:

1. La libre transmisión de información de internet ha provocado la aparición de medios de compartición directa de ficheros entre pares, las redes P2P, en las que se transmiten grandes volúmenes de contenidos multimedia rompiendo la propiedad intelectual.
2. Por otro lado, los sistemas de grabación se han abaratado y mejorado enormemente, lo que ha causado una gran proliferación de nuevos artistas ajenos a los circuitos de las grandes discográficas.
3. Las estadísticas confirman que paralelamente a las descargas ilegales, han aumentado considerablemente el número de actuaciones en directo.
4. La promoción a través de internet y las comunidades virtuales han causado la proliferación y diversificación de los estilos musicales.

Los modelos clásicos de distribución musical han tenido problemas para adaptarse a este nuevo contexto. Como alternativa, proponemos un sistema que sirva como base de negocio compatible con el devenir de los nuevos tiempos, basada en la actuación de intermediario amistoso entre los creadores de *música libre* y los usuarios y melómanos.

### 1.2 ¿Qué ofrecemos?

---

Éstas ideas ya han sido abordadas por muchos proyectos innovadores como pueden ser *LastFM*, *Jamendo* o *Deezer*, que destilan la efervescencia juvenil de las nuevas alternativas de música en internet. Nosotros proponemos un enfoque integrador que lleve el mundo de los negocios de música libre en internet a su madurez.

Para ello, proponemos la creación de un sistema que fomente la participación activa de usuarios y creadores. El sistema debe permitir, por parte del **usuario**:

1. Usar un buscador de música para obtener un catálogo de canciones, artistas, álbumes o géneros relacionados.
2. Escuchar música on-line en streaming, tanto a través del portal web como mediante un cliente standalone.
3. La descarga de música libre.
4. Votar las canciones.
5. Crear listas de reproducción.
6. Obtener recomendaciones de grupos parecidos al escuchado por el usuario.
7. Usar una radio-inteligente, donde a partir de una canción el sistema obtiene automáticamente otras canciones que supone que te gustarán.
8. Descargar un plugin para su reproductor de escritorio que envíe información sobre la música escuchada sobre el usuario. Esta idea es fundamental en el sistema para permitir la estimación de preferencias musicales, afinidad en gustos musicales entre usuarios, cálculo de estadísticas por comunidades de usuarios, generación automática de listas de reproducción, etcétera.
9. Consultar el calendario de conciertos, obteniendo recomendaciones en cuanto a sus gustos.
10. Creación y consulta de calendarios de conciertos.
11. Comprar entradas para diferentes conciertos.

Por otro lado, para el **artista**:

## 1.2. ¿QUÉ OFRECEMOS?

---

1. Subir canciones propias y categorizarlas.
2. Crear una lista donde enlazarse otros artistas.
3. Programar su propia agenda de conciertos pudiendo utilizar el sistema para la venta de entradas.
4. Crear su propio espacio dónde puede modificar el diseño e interfaz. El artista tiene la opción de permitir a los usuarios participar activamente en la construcción de su sección -incorporando ideas wiki- lo cual será el comportamiento por defecto para artistas no registrados.
5. Y evidentemente todas las funciones de un oyente.

Creemos que el sistema es sostenible y rentable económicamente, ya que permite beneficiarse económicamente por distintas vías:

1. El sistema se sustenta enormemente en la interacción a través del portal web, lo que permite obtener cuantiosos beneficios en concepto de publicidad.
2. El cobro de comisiones por la venta online de entradas a conciertos y otros servicios de *management* que puedan ofrecerse a los artistas.



# 2

## Requisitos

### 2.1 Requisitos funcionales

---

#### 2.1.1 Gestión de usuarios

##### Alta

1. Se mostrará un formulario al usuario para que lo rellene con sus datos personales.
  2. Nombre, email y contraseña serán siempre campos obligatorios.
  3. Se podrán modificar los datos después del alta.
- 
1. Se podrá acceder a cualquiera de las comunidades del sistema.
  2. Se podrá acceder a una lista con todas las comunidades ordenadas por cercanía en gustos al usuario.

##### Seguridad para el alta

1. Se comprobará que no existe ningún usuario con el mismo e-mail que el recién registrado, tal caso se advertirá al usuario en proceso de registro de que su email ya está siendo utilizado y necesitará uno nuevo para registrarse.
2. A la hora de cambiar la contraseña en la actualización de datos se pedirá confirmación mediante el ingreso de la antigua contraseña.
3. Existe la posibilidad de recuperar una contraseña olvidada a través del email del usuario.

## 2.1. REQUISITOS FUNCIONALES

---

### Identificación

1. Para cada sesión del usuario se pedirá obligatoriamente nombre y contraseña.
2. La introducción de 5 contraseñas consecutivas no válidas provocará el lanzamiento de un e-mail al usuario avisándole sobre la incidencia.
3. Una vez dentro del sistema no se volverán a pedir los datos en toda la sesión.

### Información de artistas

1. Si un artista no está registrado en el Sistema cualquier otro usuario registrado puede modificar su información.
2. Si un artista está registrado en el sistema solamente él podrá modificar su información.

## 2.1.2 Gestión de música

### Búsqueda y reproducción de canciones

1. La barra de búsqueda, donde introducir las palabras clave de la canción a buscar , devuelve una lista con todas las coincidencias de tags, título, autor o álbum ordenadas en páginas de 20 entradas.
2. Se mostrará un sistema de ordenado de canciones de forma que se puedan ordenar las canciones obtenidas en la búsqueda por título, autor, álbum o puntuación.
3. Se deben poder puntuar las canciones de 0 a 5 estrellas en la tabla obtenida tras una búsqueda, esta puntuación se añadirá a las anteriores en dicha canción.
4. Cada usuario podrá puntuar cada canción una sola vez.
5. La opción de reproducción de una de las canciones de una lista supone su inmediata reproducción tomando la lista de búsqueda como una lista de reproducción que comienza en la marcada.
6. La reproducción total de cualquier canción por parte del usuario supondrá una correspondiente actualización en los gustos de este.

## 2.1. REQUISITOS FUNCIONALES

---

7. La opción de reproducción en Smart-Radio de una de las canciones de una lista supone su inmediata reproducción, no obstante las siguientes canciones en el modo Smart-Radio no tiene porqué coincidir con el resto de canciones de la lista.
8. La aplicación del reproductor estará continuamente disponible con sus opciones (stop, play, siguiente o anterior canción).
9. Se podrá iniciar una Smart-Radio a partir de la canción actualmente en reproducción, esto no supondrá la rebobinación de esta pero , a partir de ella se deben ir reproduciendo canciones acordes con los gustos del usuario.
10. Sobre cada una de las canciones de la Smart-Radio se puede marcar si es o no del gusto del usuario.
11. Así mismo si el usuario marca la canción en reproducción como fuera de sus gustos, el sistema primero actualizará los datos sobre sus gustos y a continuación cambiará a la siguiente canción.

### Listas de Reproducción

1. Cualquier canción puede ser agregada a una de las listas de reproducción del usuario existentes o ser usada para crear una nueva.
2. Las listas usan un nombre configurable por el usuario.
3. Un usuario puede acceder a las listas de otros usuarios de su comunidad.
4. Un usuario NO puede modificar ninguna lista de otros usuarios.
5. Una lista de reproducción no puede tener dos veces la misma canción.

### Plugin

1. El usuario podrá descargar el plugin para su reproductor, este plugin buscará (si existe) el reproductor adecuado en el equipo del usuario y se instalará allí.
2. El plugin puede ser desinstalado en cualquier momento, dejando de estar en su computadora y de enviar información al sistema.
3. El plugin enviará datos acerca de la música que escucha el usuario al sistema siempre que la conexión sea posible.
4. El sistema actualizará los datos sobre los gustos del usuario una vez recibidos.

## 2.1. REQUISITOS FUNCIONALES

---

### **Puntuar Canciones**

1. Cada usuario puede puntuar una sola vez cada canción
2. La puntuación mostrada será la media de todas las puntuaciones obtenidas por la canción.

### **Subir Canciones**

Esta opción sólo está disponible para Artistas.

1. Solo los artistas pueden subir canciones.
2. El sistema tras comprobar que el usuario es un artista ofrecerá una aplicación donde introducir el path de la canción que desea subir junto con su información asociada (título, álbum, autores).
3. Se podrá cambiar la información de la canción cuando el artista desee.
4. No se podrá subir una canción ya existente en el sistema.
5. La canción solo puede ser subida por su propietario y pasará a formar parte de su información personal (discografía).

## **2.1.3 Gestión de Comunidades**

### **Creación**

1. No se podrá crear una comunidad con un nombre ya existente.
2. El creador de la comunidad será el primer administrador.
3. Cualquier administrador podrá dar permisos de administración a otro usuario.
4. La comunidad se borrará cuando todos los administradores hayan decapitado.

### **Ingreso**

1. Se podrá acceder a cualquiera de las comunidades del sistema (si no se pertenece ya).
2. Se podrá acceder a una lista con todas las comunidades ordenadas por cercanía en gustos al usuario.

### Mensajes

1. Se podrán enviar y recibir mensajes dentro de la comunidad, así como recibirlos por email, mostrándose a todos los usuarios en la página de la comunidad por orden de envío y en una cantidad mayor que 20 (a partir de aquí se borran los más antiguos).
2. Todo remitente recibirá una copia de su mensaje en su bandeja de correo.
3. Sólomente el destinatario y el remitente pueden leer sus mensajes privados, los cuales son copias y el borrado de uno de ellos no supondrá la pérdida del otro.
4. Cada mensaje puesto en la comunidad será también enviado automáticamente por mail a sus usuarios.
5. El e-mail contendrá un enlace directo a la comunidad, sin necesidad siquiera de identificación.
6. Todo aquel que acceda a la ficha de un usuario podrá leer sus mensajes públicos.

### Usuarios

1. Se podrá obtener una lista con los integrantes de la comunidad ordenados por cercanía en gustos al usuario.
2. Sobre cada integrante se podrá consultar sus gustos, nickname, perfil y mensajes.

### Administración

1. Sólo el administrador tendrá acceso a la expulsión de usuarios de su comunidad.
2. El administrador puede otorgar de sus mismos poderes a otro usuario de la misma comunidad que administra.
3. Ninguno de los administradores puede expulsar a otro administrador en su misma comunidad.
4. Con el abandono del administrador de una comunidad, será obligatorio el nombramiento de un nuevo administrador.
5. Con el abandono del administrador de una comunidad, si no existieran usuarios en los que delegar esta se elimina.

## 2.1. REQUISITOS FUNCIONALES

---

### 2.1.4 Listas de comunidades

1. La lista de comunidades a las que pertenece un usuario deberá actualizarse con cada ingreso/borrado/expulsión del usuario de alguna comunidad.
2. Ningun usuario expulsado puede volver a la comunidad de la que le borraron.
3. La lista de comunidades existentes debe aparecer ordenada por orden de gustos parecidos al usuario

### 2.1.5 Gestión de entradas

#### Compra de entradas

Para la compra de entradas se necesita tarjeta de crédito.

1. Se muestra previamente un formulario con los distintos conciertos, elegido uno de ellos se pasa a la aplicación de pagos donde se introduce el número de tarjeta de crédito para extraer el dinero.
2. Los conciertos se mostrarán ordenados por fecha desde la actual en adelante.
3. Sólo se venderán entradas para conciertos del día actual en adelante.
4. Se enviará un email de confirmación al usuario con los datos de su entrada, así como la posibilidad de impresión de esta.

#### Venta de entradas

Esta opción sólo está disponible para Artistas.

1. Se mostrará un formulario donde el Artista introducirá los datos (lugar, fecha, hora, plazas, descripción, precios) de su concierto.
2. Sólo podran venderse entradas con fecha 3 días por delante de la actual.
3. Se ingresará el 90 por ciento del dinero obtenido de la venta en una cuenta bancaria ofrecida por el Artista previamente.
4. El 10 por ciento restante se ingresará en la cuenta bancaria del propietario del sistema.

## 2.2. REQUISITOS NO FUNCIONALES

---

5. Todas las entradas desfasadas se eliminarán del sistema.
6. Si se ingresa información con venta de entradas de un concierto ya existente, el nuevo remplazará al del sistema por ser añadido por el propio artista.
7. La lista de conciertos existentes contendrá siempre a la lista de conciertos con entrada, pues pueden registrarse conciertos sólo de forma informativa o gratuitos.

## 2.2 Requisitos no funcionales

---

### 2.2.1 Hardware

1. La aplicación web funcionará sobre cualquier ordenador personal con una conexión a internet superior a 56kb.
2. La base de datos y el resto del sistema se almacenarán en un servidor privado de más de 20 TB de capacidad. Se proporcionarán actualizaciones de almacenamiento si la base de canciones así lo requiriese.

### 2.2.2 Software

1. El sistema funcionará sobre Oracle 8 por imposición técnica de los servidores disponibles. A ser posible se portará la aplicación a MySQL 5.2.
2. Las aplicaciones web funcionarán seguirán el estándar XHTML 1.0 de la W3C. Debe comprobarse su compatibilidad con los motores Gecko y WebKit. Con menos prioridad se asegurará la compatibilidad con Opera 8.0 o superior e Internet Explorer 7.0 o superior.
3. El Plugin funcionará sobre los reproductores XMMS, Audacious, Amarok y Winamp.
4. El reproductor web se creará sobre la herramienta Flash 7.0, intentando ofrecer compatibilidad con Gnash.

### 2.2.3 Seguridad

1. Ante una posible caída del sistema su operatividad estará garantizada en menos de una hora, sin pérdida de datos en la base.

## 2.2. REQUISITOS NO FUNCIONALES

---

2. Se comprobarán los ingresos de los usuarios para evitar varias sesiones desde distintos lugares de la misma cuenta.

### 2.2.4 Prestaciones

1. El tiempo de respuesta en cualquiera de las operaciones será siempre inferior a 2 segundos con menos de 400 usuarios conectados.
2. En el caso de operaciones importantes para el usuario (ingreso de cuenta bancaria) se mostrará inmediatamente un mensaje informativo donde se confirme la tramitación de la operación.
3. Cualquier usuario será capaz de manejar el sistema con fluidez tras media hora de dedicación, sin cometer errores.

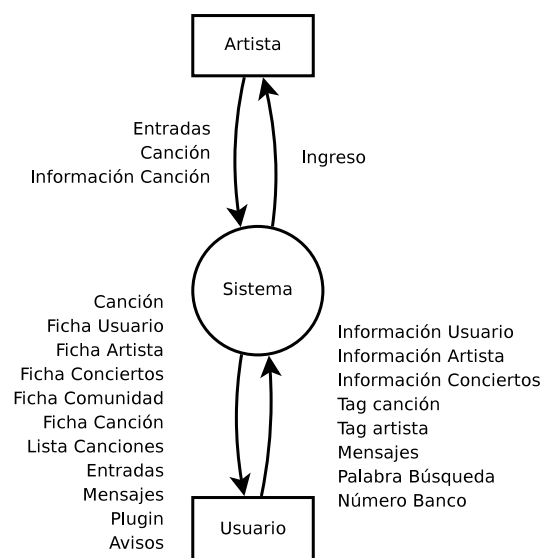


# 3

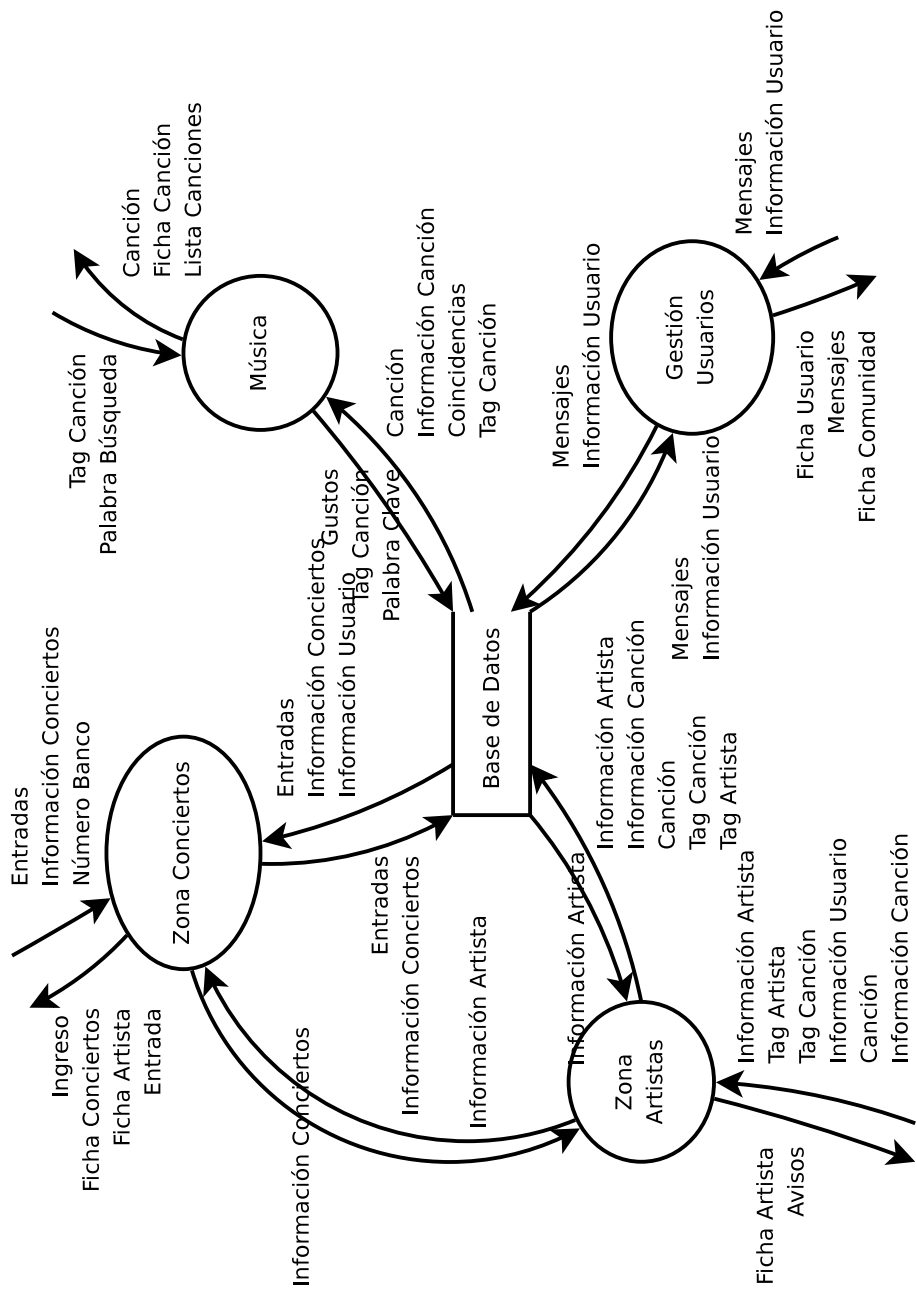
## Diagramas de Flujos de Datos

### 3.1 DFD de contexto

---

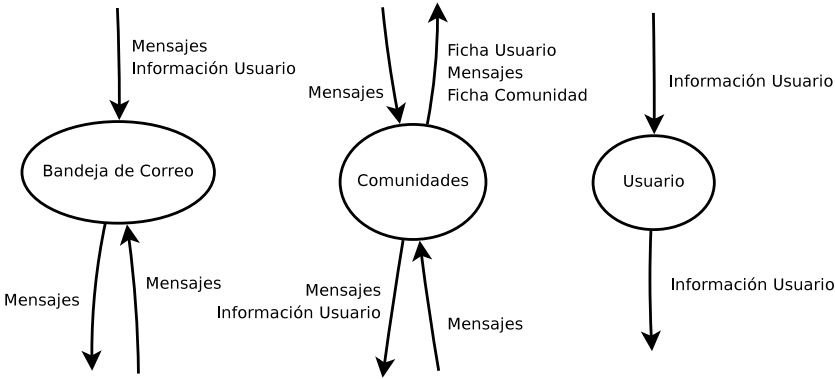


3.2 DFD de nivel 1

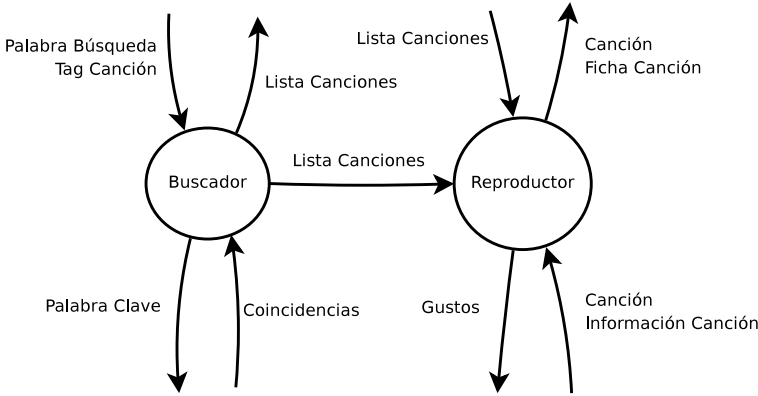


### 3.3 DFD de nivel 2

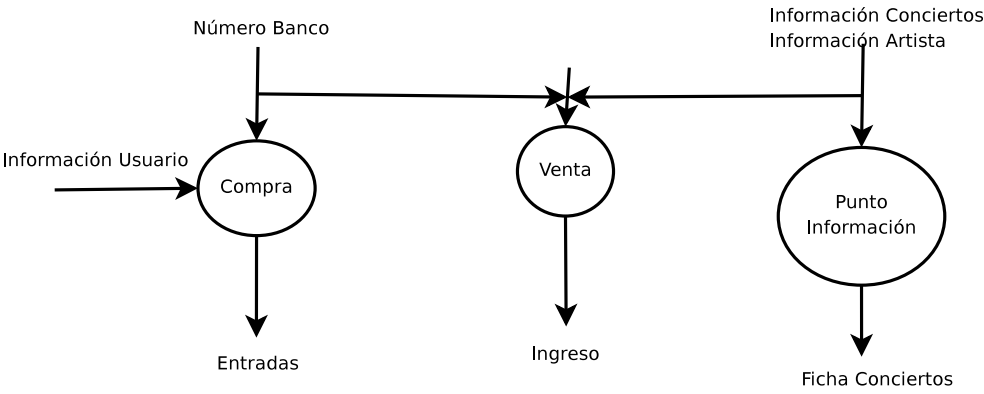
#### 3.3.1 Usuarios



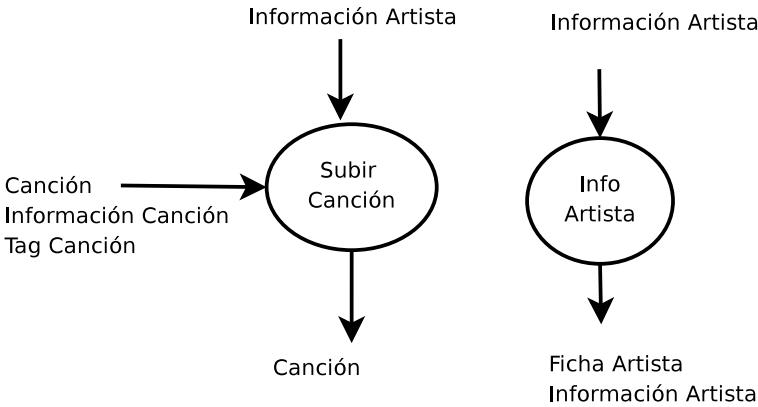
#### 3.3.2 Musica



3.3.3 Conciertos



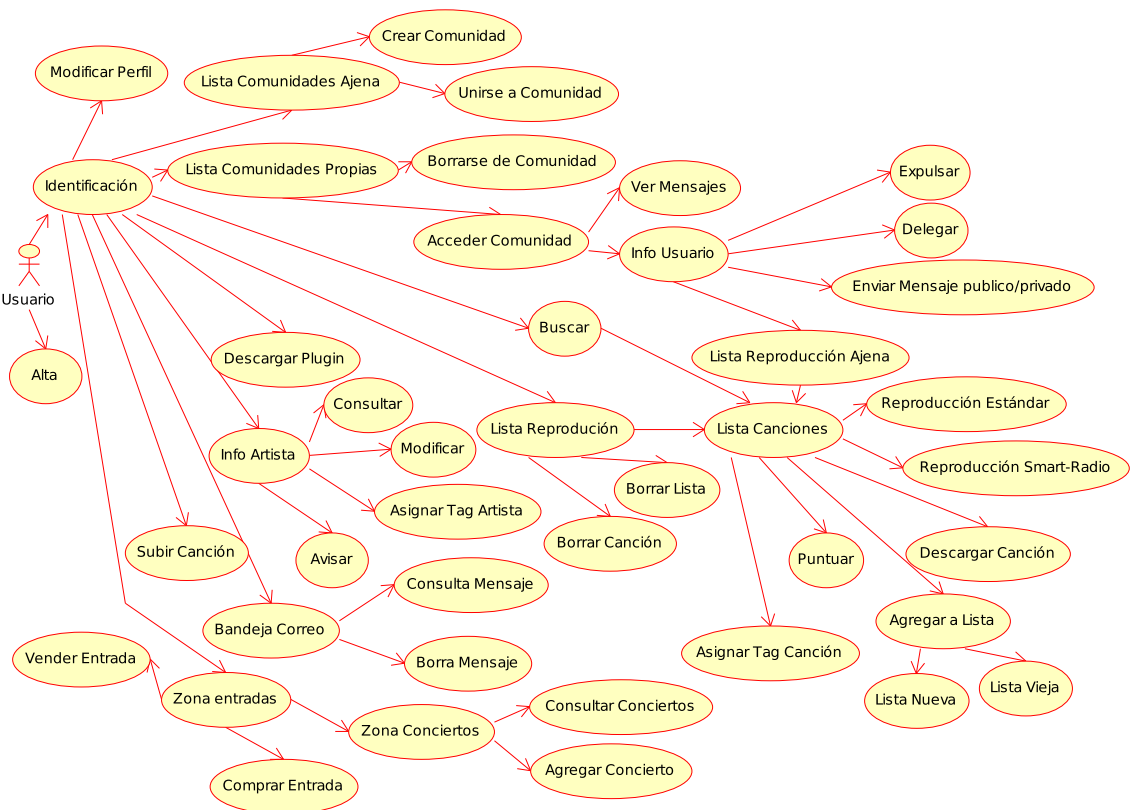
3.3.4 Artistas



# 4

## Casos de uso

## 4.1 Diagrama



## 4.2 Alta en el sistema

- **Resumen:** El usuario se registra en el sistema.
- **Actores:** Oyente o Artista.
- **Precondiciones:** El actor no debe existir en el sistema.

### 4.3. IDENTIFICARSE EN EL SISTEMA

- **Postcondiciones:** El actor se convierte en usuario registrado.

Usuario	Sistema
Accede al programa de registro	Muestra un formulario de registro(Nombre completo, email, nickname, contraseña, descripción)
Rellena los campos del formulario con sus datos	Comprueba que el email y nickname no existen en el sistema
	Confirma el alta al usuario

### 4.3 Identificarse en el sistema

- **Resumen** El actor entra al sistema como usuario identificado.
- **Actores** Oyente o Artista.
- **Precondiciones** El actor no está ya identificado en el sistema.
- **Postcondiciones** El actor pasa a estar identificado.

Usuario	Sistema
Accede al programa de identificación	Pide al usuario su nickname y contraseña
Introduce su nickname y usuario	Comprueba la validez de los datos
	Identifica al usuario

### 4.4 Modificar ficha usuario propia

- **Resumen** El usuario cambia sus datos personales en el sistema.
- **Actores** Oyente o Artista.
- **Precondiciones** El usuario debe estar identificado.
- **Postcondiciones** Se cambian los datos personales del usuario en el sistema.

#### 4.5. MODIFICAR PERFIL USUARIO PROPIO

Usuario	Sistema
Accede a su perfil	Muestra sus datos personales y la bandeja de correo
Cambia los campos que desee de sus datos personales	Comprueba la validez de los nuevos datos
	Actualiza el perfil

#### 4.5 Modificar perfil usuario propio

- **Resumen** El usuario cambia sus datos personales en el sistema.
- **Actores** Oyente o Artista.
- **Precondiciones** El usuario debe estar identificado.
- **Postcondiciones** Se cambian los datos personales del usuario en el sistema.

Usuario	Sistema
Accede a su perfil	Muestra sus datos personales
Cambia los campos que desee	Comprueba la validez de los nuevos datos
	Actualiza el perfil

#### 4.6 Acceder lista de comunidades existentes

- **Resumen** El usuario consulta una lista con las comunidades existentes.
- **Actores** Oyente o Artista.
- **Precondiciones** El usuario debe estar identificado.
- **Postcondiciones**

Usuario	Sistema
Accede al programa de comunidades	Muestra una lista con las comunidades existentes ordenadas por parentesco musical con el usuario



## 4.7. ACCEDER LISTA DE COMUNIDADES DEL USUARIO

### 4.7 Acceder lista de comunidades del usuario

- **Resumen** El usuario consulta una lista con las comunidades a las que pertenece.
- **Actores** Oyente o Artista.
- **Precondiciones** El usuario debe estar identificado.
- **Postcondiciones**

Usuario	Sistema
Accede al programa de sus comunidades	Muestra una lista con las comunidades (nombre, descripción, número de miembros y gustos) a las que pertenece ordenadas por parentesco musical con el usuario más una opción "comunidad nueva"

### 4.8 Crear una comunidad

- **Resumen** El usuario crea una nueva comunidad de la que será administrador.
- **Actores** Oyente o Artista
- **Precondiciones** Que no exista otra comunidad con el mismo nombre
- **Postcondiciones** El usuario queda registrado en una comunidad como administrador.

Usuario	Sistema
Accede a la lista de comunidades existentes (ver caso de uso)	
Elige "comunidad nueva"	Muestra un formulario donde rellenar datos sobre la nueva comunidad (nombre, descripción)
Cumplimenta el formulario	Comprueba la veracidad de los datos
	Si está correcto crea la nueva comunidad y nombra administrador al fundador

### 4.9 Unirse a una comunidad

- **Resumen** El usuario pasa a formar parte de una comunidad de usuarios.
- **Actores** Oyente o Artista .
- **Precondiciones**
- **Postcondiciones** El usuario queda registrado en una comunidad

Usuario	Sistema
Accede a la lista de comunidades existentes (ver caso de uso)	
Elige la comunidad que desea	Muestra una serie de opciones
Elige Registrar"	Ingresa al usuario en la comunidad

### 4.10 Irse de una comunidad

- **Resumen** El usuario deja una comunidad de la que forma parte.
- **Actores** Oyente o Artista.
- **Precondiciones** El usuario debe formar parte de alguna comunidad.
- **Postcondiciones** El usuario deja de formar parte de la comunidad.

Usuario	Sistema
Accede a la lista de comunidades del usuario (ver caso de uso)	
Elige la comunidad que desea abandonar	Despliega un menú de opciones sobre la comunidad
Elige abandonarla	Espera confirmación
Confirma el abandono	Elimina al usuario de la comunidad

### 4.11 Acceder a una comunidad

- **Resumen** El usuario entra en una comunidad de la que forma parte.
- **Actores** Oyente o Artista.

#### 4.12. DEJAR UN MENSAJE A LA COMUNIDAD

- **Precondiciones** El usuario debe formar parte de alguna comunidad.
- **Postcondiciones** El usuario deja de formar parte de la comunidad y esta deja de estar en la lista de sus comunidades.

Usuario	Sistema
Accede a la lista de comunidades del usuario (ver caso de uso)	
Elige la comunidad a la que desea acceder	Despliega un menú de opciones sobre la comunidad
Elige acceder	Muestra la página de la comunidad

#### 4.12 Dejar un mensaje a la comunidad

- **Resumen:** El usuario deja un mensaje que pueden leer todos los integrantes de su comunidad.
- **Actores:** Oyente o Artista.
- **Precondiciones:** El usuario debe pertenecer a alguna comunidad.
- **Postcondiciones:** Se crea un nuevo mensaje en la comunidad

Usuario	Sistema
Accede a una de sus comunidades (ver caso de uso)	
Escribe un mensaje en la zona a tal efecto y acepta	Muestra el mensaje en la página de la comunidad y envía una copia al email de los integrantes

#### 4.13 Acceder a la información de usuarios de su comunidad

- **Resumen:** El usuario consulta la información de los integrantes de sus comunidades.
- **Actores:** Oyente o Artista.
- **Precondiciones:** El usuario debe pertenecer a alguna comunidad.
- **Postcondiciones:**

#### 4.14. DEJAR MENSAJE PRIVADO A UN USUARIO

Usuario	Sistema
Accede a una de sus comunidades (ver caso de uso)	
Elige la lista de usuarios de la comunidad	Muestra los diferentes usuarios de la comunidad en una lista
Elige uno de los usuarios	Muestra la ficha del usuario con su foto, nickname, mensajes, listas de reproducción y opciones

#### 4.14 Dejar mensaje privado a un usuario

- **Resumen:** El usuario envía un mensaje privado a otro de su comunidad.
- **Actores:** Oyente o Artista.
- **Precondiciones:** Ambos usuarios sean de la misma comunidad.
- **Postcondiciones:** Se crea un nuevo mensaje en la bandeja de correo del destinatario (más una copia en la del remitente).

Usuario	Sistema
Accede a la información de usuarios de su comunidad (ver caso de uso)	
Selecciona enviar mensaje privado en las opciones	Muestra un campo donde escribir el mensaje
Escribe el mensaje y pulsa aceptar	Envía el mensaje al destinatario

#### 4.15 Dejar mensaje público a un usuario

- **Resumen:** El usuario envía un mensaje público a otro de su comunidad.
- **Actores:** Oyente o Artista.
- **Precondiciones:** Ambos usuarios sean de la misma comunidad.
- **Postcondiciones:** Se crea un nuevo mensaje en la ficha de usuario del destinatario.

#### 4.16. EXPULSAR A UN USUARIO DE UNA COMUNIDAD

Usuario	Sistema
Accede a la información de usuarios de su comunidad (ver caso de uso)	
Selecciona enviar mensaje público en las opciones	Muestra un campo donde escribir el mensaje
Escribe el mensaje y pulsa aceptar	Envía el mensaje al destinatario y lo muestra en su ficha de usuario

#### 4.16 Expulsar a un usuario de una comunidad

- **Resumen:** El usuario/administrador de comunidad elimina a un usuario de su comunidad.
- **Actores:** Oyente o Artista.
- **Precondiciones:** El usuario debe ser el administrador, el usuario a expulsar pertenece a su comunidad.
- **Postcondiciones:** Se borra al usuario expulsado de la comunidad.

Usuario	Sistema
Accede a la información de usuarios de su comunidad (ver caso de uso)	
Selecciona eliminar en las opciones	Envía mensaje de confirmación si es el administrador, si no vuelve atrás
Confirma la expulsión	Borra al usuario de la comunidad

#### 4.17 Nombrar administrador a un usuario de la comunidad

- **Resumen:** El usuario/administrador de comunidad delega en uno de los usuarios de esta.
- **Actores:** Oyente o Artista.
- **Precondiciones:** El usuario debe ser el administrador, el usuario sobre el que delegar pertenece a su comunidad.

#### 4.18. BÚSQUEDA DE CANCIONES

- **Postcondiciones:** Un nuevo usuario pasa a ser el administrador de la comunidad junto al anterior/es.

Usuario	Sistema
Accede a la información de usuarios de su comunidad (ver caso de uso)	
Selecciona delegar en las opciones	Envía mensaje de confirmación si es el administrador, si no vuelve atrás
Confirma el trasvase de poder	Nombra al nuevo administrador y degrada al anterior a usuario

#### 4.18 Búsqueda de Canciones

- **Resumen** El usuario busca una canción.
- **Actores** Oyente o Artista.
- **Precondiciones** El usuario debe estar identificado.
- **Postcondiciones**

Usuario	Sistema
Accede al programa de búsqueda	Muestra una barra de búsqueda
Introduce la palabra clave	Busca tags, canciones, autores o álbumes que coincidan con la palabra clave
	Devuelve una lista con las coincidencias (título, autor, álbum, puntuación)

#### 4.19 Mostrar Lista de reproducción

- **Resumen** El usuario recibe una de sus listas de canciones para la reproducción.
- **Actores** Oyente o Artista.
- **Precondiciones** El usuario debe estar identificado y tener alguna lista de reproducción.

#### 4.20. MOSTRAR LISTA DE REPRODUCCIÓN AJENA

- **Postcondiciones**

Usuario	Sistema
Accede al programa de listas de reproducción	Muestra una lista con las diferentes listas de reproducción del usuario
Selecciona una de las listas	Muestra una serie de opciones sobre la lista de reproducción
Elige mostrar	Devuelve una lista con las canciones de la lista

#### 4.20 Mostrar Lista de reproducción ajena

- **Resumen** El usuario recibe una de las listas de canciones para la reproducción de un integrante de su comunidad.
- **Actores** Oyente o Artista.
- **Precondiciones** El debe pertenecer al menos a una comunidad.
- **Postcondiciones**

Usuario	Sistema
ccede a la información de usuarios de su comunidad (ver caso de uso)	
Elige la lista de reproducción del usuario	Muestra una lista con las diferentes listas de reproducción del usuario
Selecciona una de las listas	Devuelve una lista con las canciones de la lista

#### 4.21 Borrar Lista de reproducción

- **Resumen** El usuario elimina una de sus listas de canciones para la reproducción.
- **Actores** Oyente o Artista.
- **Precondiciones** El usuario debe estar identificado y tener alguna lista de reproducción.
- **Postcondiciones** Se elimina una lista de reproducción del usuario.

#### 4.22. AGREGAR CANCIÓN LISTA DE REPRODUCCIÓN

Usuario	Sistema
Accede al programa de listas de reproducción	Muestra una lista con las diferentes listas de reproducción del usuario
Selecciona una de las listas	Muestra una serie de opciones sobre la lista de reproducción
Elige eliminar	Envía un mensaje para confirmación
Confirma el borrado	Elimina la lista de reproducción

#### 4.22 Agregar canción Lista de reproducción

- **Resumen** El usuario incluye una nueva canción a una de sus listas de reproducción.
- **Actores** Oyente o Artista.
- **Precondiciones** El usuario debe tener alguna lista de reproducción.
- **Postcondiciones** Se agrega una lista de reproducción con una canción elegida al usuario.

Usuario	Sistema
Accede a una lista de canciones: Buscando, desde una de sus listas de reproducción o desde una lista de reproducción ajena (ver casos de uso)	
Selecciona una de las canciones	Muestra una serie de opciones sobre la canción
Elige agregarla a una lista de reproducción	Muestra la lista de listas de reproducción del usuario más "nueva lista"
Selecciona una de las listas	Agrega esta canción a la lista elegida

#### 4.23 Agregar canción a nueva Lista de reproducción

- **Resumen** El usuario incluye una nueva canción a una nueva listas de reproducción.



#### 4.24. BORRAR CANCIÓN DE LISTA DE REPRODUCCIÓN

- **Actores** Oyente o Artista.
- **Precondiciones**
- **Postcondiciones** Se agrega una canción a una nueva lista de reproducción del usuario.

Usuario	Sistema
Accede a una lista de canciones: Buscando, desde una de sus listas de reproducción o desde una lista de reproducción ajena (ver casos de uso)	
Selecciona una de las canciones	Muestra una serie de opciones sobre la canción
Elige agregarla a una lista de reproducción	Muestra la lista de listas de reproducción del usuario más "nueva lista"
Selecciona "nueva lista"	Pide un nombre para la nueva lista
Introduce el nombre	Crea la lista nueva con la canción elegida.

#### 4.24 Borrar canción de Lista de reproducción

- **Resumen** El usuario borra una canción de una de sus listas de reproducción.
- **Actores** Oyente o Artista.
- **Precondiciones** El usuario debe tener alguna lista de reproducción.
- **Postcondiciones** Se elimina una canción de una lista de reproducción del usuario.

Usuario	Sistema
Accede a una de sus listas de reproducción (ver caso de uso)	
Elige la canción que desea borrar	Muestra una serie de opciones
Selecciona "borrar"	Envía un mensaje para confirmación
Confirma el borrado	elimina la canción de la lista

## 4.25 Puntuar Canción

---

- **Resumen** El usuario puntua una canción.
- **Actores** Oyente o Artista.
- **Precondiciones**
- **Postcondiciones** Se actualiza la puntuación de la canción

Usuario	Sistema
Accede a una lista de canciones: Buscando, desde una de sus listas de reproducción o desde una lista de reproducción ajena (ver casos de uso)	
Selecciona una de las canciones	Muestra una serie de opciones so- bre la canción
Elige puntuarla	Muestra un campo donde introdu- cir la puntuación
Introduce la puntuación	Actualiza el valor de la puntuación de la canción

## 4.26 Añadir Tag a Canción

---

- **Resumen** El usuario da una palabra que describa a la canción paa  
ayudar en su búsqueda/categorización.
- **Actores** Oyente o Artista.
- **Precondiciones**
- **Postcondiciones** Se actualizan los tags de la canción

#### 4.27. REPRODUCIR CANCIÓN

Usuario	Sistema
Accede a una lista de canciones: Buscando, desde una de sus listas de reproducción o desde una lista de reproducción ajena (ver casos de uso)	
Selecciona una de las canciones	Muestra una serie de opciones sobre la canción
Elige dar Tag	Muestra un campo donde introducir el Tag
Introduce la palabra	Actualiza los tags de la canción

#### 4.27 Reproducir Canción

- **Resumen** El usuario reproduce una canción.
- **Actores** Oyente o Artista.
- **Precondiciones**
- **Postcondiciones**

Usuario	Sistema
Accede a una lista de canciones: Buscando, desde una de sus listas de reproducción o desde una lista de reproducción ajena (ver casos de uso)	
Selecciona una de las canciones	Muestra una serie de opciones sobre la canción
Elige reproducirla	Reproduce la canción

#### 4.28 Reproducir Canción en modo Smart-Radio

- **Resumen** El usuario reproduce una canción en una Smart-Radio.
- **Actores** Oyente o Artista.
- **Precondiciones**
- **Postcondiciones** Se actualizan los datos sobre los gustos del usuario.

#### 4.29. DESCARGAR CANCION

Usuario	Sistema
Accede a una lista de canciones: Buscando, desde una de sus listas de reproducción o desde una lista de reproducción ajena (ver casos de uso)	
Selecciona una de las canciones	Muestra una serie de opciones sobre la canción
Elige reproducirla en modo Smart-Radio	Reproduce la canción en una Smart-Radio

#### 4.29 Descargar Cancion

- **Resumen:** El usuario descarga una canción.
- **Actores:** Oyente o Artista.
- **Precondiciones:** La canción debe existir en el sistema.
- **Postcondiciones:**

Usuario	Sistema
Accede a una lista de canciones: Buscando, desde una de sus listas de reproducción o desde una lista de reproducción ajena (ver casos de uso)	
Selecciona una de las canciones	Muestra una serie de opciones sobre la canción
Elige Descargar	Envía la canción al usuario

#### 4.30 Subir Cancion

- **Resumen:** El artista sube una canción de su propiedad.
- **Actores:** Oyente o Artista.
- **Precondiciones:** El usuario debe estar identificado como artista, la canción no debe estar ya en el sistema.
- **Postcondiciones:** Se añade una nueva canción a la base de datos.

#### 4.31. DESCARGAR-ISNTALLAR PLUGIN PLUGIN

Usuario	Sistema
Accede al programa de subida de canciones	Muestra una zona donde introducir la ruta de la canción e información(título,fecha,álbum)
Introduce la ruta y la información requerida	Sube la canción al sistema
	Actualiza los datos del autor

#### 4.31 Descargar-isntallar plugin Plugin

- **Resumen:** El usuario obtiene el plugin para su reproductor.
- **Actores:** Oyente o Artista.
- **Precondiciones:** El usuario debe estar identificado en el sistema.
- **Postcondiciones:** Se actualizan los datos sobre gustos del usuario.

Usuario	Sistema
Accede al programa de descarga del plugin	Envía el mensaje de aceptación para iniciar la descarga
Acepta y descarga el plugin	Una vez bajado se lanza el instalable
Instala el plugin en su reproductor favorito	El Plugin envía datos sobre los gustos del usuario al sistema

#### 4.32 Acceder a la información del artista

- **Resumen:** El usuario consulta la información de un artista.
- **Actores:** Oyente o Artista.
- **Precondiciones:** El usuario debe estar identificado.
- **Postcondiciones:**

Usuario	Sistema
Introduce el nombre del artista en la barra de búsqueda	Muestra una lista con las coincidencias
Selecciona al artista buscado (si existe)	Muestra una serie de opciones

### 4.33 Añadir Tag a Artista

- **Resumen** El usuario da una palabra que describa al artista ayudar en su búsqueda/categorización.
- **Actores** Oyente o Artista.
- **Precondiciones**
- **Postcondiciones** Se actualizan los tags del artista

Usuario	Sistema
Accede a la información del artista (ver caso de uso)	
Selecciona asignar tag en las opciones	Muestra un campo donde introducir el nuevo tag
Introduce la palabra que defina al artista	Actualiza los tags del artista

### 4.34 Consultar la información del artista

- **Resumen:** El usuario consulta la información de un artista.
- **Actores:** Oyente o Artista.
- **Precondiciones:**
- **Postcondiciones:**

Usuario	Sistema
Accede a la información del artista (ver caso de uso)	
Selecciona "Información. <sup>en</sup> las opciones	Muestra la información (Nombre, biografía, discografía) del artista

### 4.35 Modificar la información del artista

- **Resumen:** El usuario modifica la información de un artista.
- **Actores:** Oyente o Artista.

#### 4.36. INDICA QUE LA INFORMACIÓN DEL ARTISTA ESTÁ EQUIVOCADA

- **Precondiciones:**
- **Postcondiciones:** Se actualiza la información del artista.

Usuario	Sistema
Accede a la información del artista (ver caso de uso)	
Selecciona "Modificar Información. <sup>en</sup> las opciones	Muestra la información (Nombre, biografía, discografía) del artista editable
Modifica los campos que desea y pulsa "guardar los cambios"	Actualiza la información del artista

#### 4.36 Indica que la información del artista está equivocada

- **Resumen:** El usuario consulta la información de un artista.
- **Actores:** Oyente o Artista.
- **Precondiciones:** El artista no debe estar registrado en el sistema como usuario.
- **Postcondiciones:** Muestra un aviso a los usuarios que accedan a la información del artista sobre posibles erratas.

Usuario	Sistema
Accede a la información del artista (ver caso de uso)	
Selecciona . <sup>E</sup> quivocado. <sup>en</sup> las opciones	Coloca un aviso en la página de información del artista avisando de la existencia de erratas

#### 4.37 Acceder a bandeja de correo

- **Resumen:** El usuario accede a su bandeja de correo.
- **Actores:** Oyente o Artista.
- **Precondiciones:** El usuario debe estar identificado.

#### 4.38. LEER MENSAJE

- **Postcondiciones:**

Usuario	Sistema
Accede a su perfil	Muestra sus datos personales y la bandeja de correo
Selecciona la bandeja de correo	Muestra una lista con sus mensajes enviados y recibidos

#### 4.38 Leer mensaje

- **Resumen:** El usuario lee un mensaje privado.
- **Actores:** Oyente o Artista.
- **Precondiciones:** Que él sea el destinatario o remitente del mensaje.
- **Postcondiciones:**

Usuario	Sistema
Accede a su bandeja de correo (ver caso de uso)	
Selecciona el mensaje que desee leer	Muestra una serie de opciones
Selecciona "leer"	Abre el mensaje para su lectura

#### 4.39 Borrar mensaje

- **Resumen:** El usuario borra un mensaje privado.
- **Actores:** Oyente o Artista.
- **Precondiciones:** Que él sea el destinatario del mensaje.
- **Postcondiciones:** Se elimina el mensaje de la bandeja de correo.



#### 4.40. ACCEDER A ZONA DE CONCIERTOS

Usuario	Sistema
Accede a su bandeja de correo (ver caso de uso)	
Selecciona el mensaje que desee eliminar	Muestra una serie de opciones
Selecciona "eliminar"	Muestra un mensaje de confirmación
Confirma el borrado	Elimina el mensaje

#### 4.40 Acceder a zona de conciertos

- **Resumen:** El usuario accede a la zona de compra, venta e información sobre entradas y conciertos.
- **Actores:** Oyente o Artista.
- **Precondiciones:** El usuario debe estar identificado.
- **Postcondiciones:**

Usuario	Sistema
Accede al programa de conciertos	Muestra Una lista de los conciertos registrados y un punto de venta de entradas.

#### 4.41 Vender Entrada

- **Resumen:** El artista informa de sus conciertos y vende sus entradas.
- **Actores:** Oyente o Artista.
- **Precondiciones:** El usuario debe estar identificado y ser un artista.
- **Postcondiciones:** Se añaden nuevas entradas al sistema e información de conciertos.

#### 4.42. COMPRAR ENTRADA

Usuario	Sistema
Usuario	Sistema
Accede a la zona de conciertos (ver caso de uso)	
Accede al punto de venta	Muestra las opciones comprar y vender entradas
Selecciona vender entradas	Pide información sobre el concierto (lugar, fecha, hora, precio, aforo, descripción)
Introduce la información	Comprueba la veracidad de los datos y si es correcto muestra una pantalla donde colocar la cuenta bancaria para situar las ganancias del artista
Introduce la cuenta bancaria	Envía un mensaje para esperar confirmación donde aparecen todos los datos introducidos, así como posibilidad de impresión
Si está de acuerdo acepta	Pone en venta las entradas y renueva la información de conciertos

#### 4.42 Comprar Entrada

- **Resumen:** El usuario adquiere entradas para un concierto.
- **Actores:** Oyente o Artista.
- **Precondiciones:**
- **Postcondiciones:** Se eliminan las entradas compradas del sistema.

#### 4.43. CONSULTAR INFORMACIÓN DE CONCIERTOS

Usuario	Sistema
Usuario	Sistema
Accede a la zona de conciertos (ver caso de uso)	
Accede al punto de venta	Muestra las opciones comprar y vender entradas
Selecciona comprar entradas	Muestra una lista con los conciertos y su información (lugar, fecha, hora, precio, palzas disponibles, descripción)
Selecciona el concierto que desea	Pide el número de entradas a comprar y el número de tarjeta de crédito con el que el usuario pagará
Introduce los datos pedidos	Muestra una pantalla general con un resumen del concierto e información de la venta y espera confirmación, así como posibilidad de impresión
Confirma la compra	Envía un mensaje de aceptación y avisa de que se descontará el dinero de la tarjeta y recibirá las entradas en su email
	Descuenta el dinero de la tarjeta y envía las entradas via email al usuario

#### 4.43 Consultar información de conciertos

- **Resumen:** El usuario recibe información sobre los conciertos.
- **Actores:** Oyente o Artista.
- **Precondiciones:**
- **Postcondiciones:**

#### 4.44. INGRESAR INFORMACIÓN DE CONCIERTOS

Usuario	Sistema
Usuario	Sistema
Accede a la zona de conciertos (ver caso de uso)	
Selecciona uno de los conciertos de la lista	Muestra la información sobre el concierto (lugar, fecha, hora, precio, palzas disponibles, descripción)

#### 4.44 Ingresar información de conciertos

- **Resumen:** El usuario inscribe información sobre los conciertos.
- **Actores:** Oyente o Artista.
- **Precondiciones:**
- **Postcondiciones:** Se genera información sobre nuevos conciertos.

Usuario	Sistema
Usuario	Sistema
Accede a la zona de conciertos (ver caso de uso)	
Escribe la información sobre el concierto(lugar, fecha, hora, precio, palzas disponibles, descripción)	Se añade a la lista de conciertos

# 5

## Modelo Entidad/Relación



# 6

## Diccionario de datos

### 6.1 ENTIDADES

---

#### 6.1.1 MENSAJERO

Entidad que interacciona con mensajes

<i>Atributo</i>	<i>Descripción</i>
ID	Número identificativo del mensajero
Fecha-creacion	fecha el que el usuario se dio de alta en el sistema

**Claves Candidatas:** ID

#### 6.1.2 CANCIÓN

Canción en el sistema.

<i>Atributo</i>	<i>Descripción</i>
ID	Número identificativo de la canción en el sistema
Titulo	Nombre de la canción (NOT NULL)
Album	Nombre del album donde aparece
Fecha	Año de edición de la canción
Url-descarga	Dirección donde se aloja la canción (NOT NULL)
Descripción	Breve comentario acerca de la canción

**Claves Candidatas:** ID

### 6.1.3 USUARIO

Usuario del sistema.

<i>Atributo</i>	<i>Descripción</i>
ID	Numero identificativo del usuario en el sistema.
E-mail	Dirección de correo electrónico
Nombre	Nombre del usuario en el sistema (NOT NULL)
Hash-pass	Hash de la contraseña de seguridad (NOT NULL)
Descripcion	Descripcion textual del usuario y sus gustos musicales.
Sexo	Sexo del usuario.
Fecha na	Fecha de nacimiento del usuario.
Avatar Url	La URL a la imagen del usuario.
Homepage Url	La URL de la web personal del usuario.

**Claves Candidatas:** ID, E-mail, Nombre

**Restricciones Adicionales:**

$Sexo \in \{M, F\}$

### 6.1.4 USR-ARTISTA

Usuario del sistema que, además, crea canciones.

<i>Atributo</i>	<i>Descripción</i>
ID	Numero identificativo del usuario artista en el sistema.

**Claves Candidatas:** ID

### 6.1.5 ARTISTA

Creador de canciones.

<i>Atributo</i>	<i>Descripción</i>
ID	número identificativo del artista
Nombre	Nombre del artista (NOT NULL)
Descripcion	Breve comentario sobre el artista



**Claves Candidatas:** ID.

### **6.1.6 MENSAJE**

Texto enviado de un usuario a otro / otros.

<i>Atributo</i>	<i>Descripción</i>
ID	Número identificativo del mensaje en el sistema
Asunto	Breve enunciado que describe el cuerpo del mensaje
Contenido	Cuerpo del mensaje
Fecha	Fecha de envío del mensaje (NOT NULL)
Publico	Indica si el mensaje es público o privado (Si/No)

**Claves Candidatas:** ID.

### **6.1.7 COMUNIDAD**

Conjunto de Usuarios.

<i>Atributo</i>	<i>Descripción</i>
ID	Número identificativo de la comunidad en el sistema
Nombre	Nombre de la Comunidad (NOT NULL)
Descripcion	Breve comentario explicando el tema de la comunidad

**Claves Candidatas:** ID, Nombre

### **6.1.8 TAG**

Palabra usada para describir canciones o a artistas.

<i>Atributo</i>	<i>Descripción</i>
Palabra Tag	Palabra descriptora

**Claves Candidatas:** Palabra Tag

## 6.1. ENTIDADES

### 6.1.9 TAGGEABLE

Entidad que puede recibir un tag.

<i>Atributo</i>	<i>Descripción</i>
ID	Numero identificador del objeto taggeable

**Claves Candidatas:** ID

### 6.1.10 CONCIERTO

Evento musical en directo.

<i>Atributo</i>	<i>Descripción</i>
ID	Número identificador del concierto en el sistema
Nombre	Nombre del concierto
Fecha	Fecha en la que ocurre el evento (NOT NULL)
Num. Entradas	Cantidad de entradas disponibles

**Claves Candidatas:** ID.

### 6.1.11 LOCAL

Edificio donde se realizan conciertos.

<i>Atributo</i>	<i>Descripción</i>
ID	Número identificador del local en el sistema
Nombre	Nombre del local
Direccion	Vía y número donde está el local (NOT NULL)
Ciudad	Localidad donde está el local (NOT NULL)
CP	Código postal (NOT NULL)
Pais	Pais donde está el local (NOT NULL)
Aforo	Cantidad de público que puede entrar en el local
Teléfono	Telefono de contacto con el local.
Fax	Fax de contacto con el local.

**Claves Candidatas:** ID, (Nombre, Direccion, CP, Pais)

### 6.1.12 ENTRADA

Elemento que permite el acceso a un concierto.

<i>Atributo</i>	<i>Descripción</i>
Numero	Número de la entrada
Precio	Costo de la entrada (NOT NULL)

**Claves Candidatas:** Numero.

## 6.2 RELACIONES

### 6.2.1 ESCUCHA

Relación de los usuarios con las canciones que escuchan.

<i>Entidad</i>	<i>Cardinalidad</i>	<i>Rol</i>
Usuario	*	Usuario que reproduce una canción
Cancion	*	Canción reproducida por el usuario

Atributos específicos de la relación:

<i>Atributo</i>	<i>Descripción</i>
Fecha	Última vez que el usuario escuchó la canción

**Claves Candidatas:** Usuario.mail-Escucha.fecha-Cancion.ID

### 6.2.2 ENLISTA

Relación que permite al usuario crear sus propias listas de reproducción.

<i>Entidad</i>	<i>Cardinalidad</i>	<i>Rol</i>
Usuario	*	Usuario que enlista una canción
Cancion	*	Canción que pasa a estar en alguna de las listas del usuario

## 6.2. RELACIONES

Atributos específicos de la relación:

<i>Atributo</i>	<i>Descripción</i>
Nombre_lista	Nombre de la lista del usuario donde se agrega la canción
Posicion_Lista	Posicion que ocupa la canción en la lista

**Claves Candidatas:** Enlista.Nombre\_lista-Enlista.Posicion\_lista-Usuario.mail-Cancion.ID

### 6.2.3 PUNTUA

Usuarios votan las canciones que escuchan.

<i>Entidad</i>	<i>Cardinalidad</i>	<i>Rol</i>
Usuario	*	Usuario que juzga la canción
Cancion	*	Canción juzgada por el usuario

Atributos específicos de la relación:

<i>Atributo</i>	<i>Descripción</i>
Puntuacion	Voto del usuario respecto de la canción

**Claves Candidatas:** Usuario.mail-Cancion.ID

### 6.2.4 TIENE

Usuarios describen mediante tags los objetos taggeables.

<i>Entidad</i>	<i>Cardinalidad</i>	<i>Rol</i>
Usuario	*	Usuario que describe una canción
Tag	*	Palabra descriptora para el objeto
Taggeable	*	Recibe el tag

**Claves Candidatas:** Usuario.mail-Tag.Palabra\_tag-Taggeable.ID

## 6.2. RELACIONES

### 6.2.5 CREA

Creación de una canción en el sistema por parte de un artista.

Entidad	Cardinalidad	Rol
Artista	*	Creador de la canción
Canción	*	Canción creada por e artista

**Claves Candidatas:** Artista.ID-Usuario.mail

### 6.2.6 POSEE

Artistas registrados se encarga de administrar sus propios grupos (artistas).

Entidad	Cardinalidad	Rol
Usr_artista	*	Usuario registrado que es artista
Artista	*	Conjunto de información de un artista de la que usr-artista toma el control

**Restricciones Adicionales:** Usr\_Artista debe ser un integrante de Artista

**Claves Candidatas:** Usr\_Artista.ID-Artista.ID

### 6.2.7 TOCA

Artistas celebran conciertos.

Entidad	Cardinalidad	Rol
Artista	*	Artista que dió el concierto
Concierto	*	Concierto que celebró el artista

Atributos específicos de la relación:

Atributo	Descripción
Porcentaje	Parte de las ganancias que recibe el artista mediante la venta de entradas

## 6.2. RELACIONES

**Claves Candidatas:** Artista.ID-Concierto.ID

### 6.2.8 LUGAR

Localización de los concierots.

<i>Entidad</i>	<i>Cardinalidad</i>	<i>Rol</i>
Local	1	Edificio o lugar donde se da el concierto
Concierto	*	Concierto que se celebra

**Claves Candidatas:** Concierto.ID

### 6.2.9 COMPRA

Un usuario compra entradas para un concierto.

<i>Entidad</i>	<i>Cardinalidad</i>	<i>Rol</i>
Usuario	1	Usuario que compra la entrada
Entrada	*	Pase para el concierto que desea el usuario

Atributos específicos de la relación:

<i>Atributo</i>	<i>Descripción</i>
Estado-pago	Estado de la transacción monetaria para conseguir la entrada (sin pagar / en trámite / pagado) (NOT NULL)

**Claves Candidatas:** Concierto.ID-Entrada.Numero-Usuaio.Mail

### 6.2.10 ES-MIEMBRO

Usuarios que forman parte de comunidades.

## 6.2. RELACIONES

<i>Entidad</i>	<i>Cardinalidad</i>	<i>Rol</i>
Usuario	*	Integrante de la comunidad
Comunidad	*	Agrupación de usuarios

Atributos específicos de la relación:

<i>Atributo</i>	<i>Descripción</i>
Fecha-alta	Fecha en la que entra el usuario en la comunidad por primera vez

**Claves Candidatas:** Usuario.Mail-Comunidad.ID

### 6.2.11 ADMINISTRA

Un usuario administra una comunidad.

<i>Entidad</i>	<i>Cardinalidad</i>	<i>Rol</i>
Usuario	*	Usuario que administra la comunidad
Comunidad	*	Agrupación de usuarios que adminsitra el usuario

**Claves Candidatas:** Usuario.Mail-Comunidad.ID

### 6.2.12 ENVIA

Envío de mensajes a mensajeros.

<i>Entidad</i>	<i>Cardinalidad</i>	<i>Rol</i>
Mensaje	*	Mensaje que recibe la comunidad
Mensajero	*	Objeto que envia el mensaje

**Claves Candidatas:** Mensaje.ID

**6.2.13 Recibe**

Recepción de mensajes por parte del mensajero.

<i>Entidad</i>	<i>Cardinalidad</i>	<i>Rol</i>
Mensaje	*	Mensaje que recibe el usuario
Mensajero	*	Objeto que recibe el mensaje

**Claves Candidatas:** Mensaje.ID-Mensajero.ID

**6.2.14 Bandeja**

Manipulación de la bandeja de entrada por los mensajeros

<i>Entidad</i>	<i>Cardinalidad</i>	<i>Rol</i>
Mensaje	*	Mensaje que envia el usuario
Mensajero	*	Objeto que manipula mensajes

**Claves Candidatas:** Mensaje.ID-Mensajero.ID



# 7

## Normalización

### 7.1 De Modelo E/R a Relacional

---

En el paso desde el modelo de entidad relación al modelo relacional deben simplificarse las tablas resultantes para evitar redundancias o para reducir el tamaño de la base de datos.

Con este fin, en ocasiones se ha incluido la información de una relación en la tabla propia de la entidad. Esto puede hacerse cuando las relaciones tienen una cardinalidad 1 en alguno de sus extremos, y cuando la relación no tenga datos asociados que se redunden al realizar la fusión. También pueden realizarse este tipo de simplificaciones en algunas relaciones de herencia. Aún así, no existen reglas estrictas en la simplificación de las tablas y el diseñador de la Base de Datos debe aplicar su ingenio y experiencia para saber en qué condiciones es conveniente.

En el próximo apartado se enumeran y estudian todas las tablas que integran finalmente la base de datos con las simplificaciones que se han realizado.

### 7.2 Análisis de la Forma Normal de las relaciones del sistema

---

Para el correcto funcionamiento del sistema es preferible normalizar las relaciones que se establecen entre los datos.

De los 6 posibles niveles de normalización será preferible alcanzar, al menos, la FNBC (Forma Normal de Boyce y Codd) ya que teniendo una tabla en FNBC generalmente se va a cumplir que lo esté también en 4FN y 5FN, si no se da el caso entonces pasar a estas formas normales superiores es de gran dificultad.

## 7.2. ANÁLISIS DE LA FORMA NORMAL DE LAS RELACIONES DEL SISTEMA

---

Por tanto se intentará normalizar las relaciones hasta la FNBC sabiendo que una tabla lo está en los siguientes casos:

1. Cuando una relación está en 3FN y además para toda dependencia funcional no-trivial su determinante es una CK completa.
2. Siempre que tengamos una relación en 3FN solamente tenemos que ver si no tiene llaves solapadas (atributos comunes entre CK), y si no tiene CK múltiples (si las hubiera habría que comprobar si contienen interdependencias entre atributos primos que no son CK).

¿Cuándo está una tabla en 3FN? Cuando lo está en 2FN (cada atributo no primo es dependiente de toda la clave primaria al completo) y además no existen dependencias transitivas entre los atributos no-primos (A-¿B siendo ambos atributos no-primos) por lo tanto basta con mirar si tenemos uno o menos atributos no primos o, en el caso de que haya 2 o más, comprobar que no existen dependencias entre estos.

### 7.2.1 Relaciones

- `hm_person(id, create_date)`  
Única CK y un solo atributo no primo → FNBC
- `hm_message(id, sender_id, content, send_date, private)`  
Única CK, Sin relaciones transitivas entre atributos no primos, → FNBC
- `hm_user(id, nick, pass_hash, email, confirm_code, real_name, sex, description, birth_date, homepage_url, avatar_url)`  
Múltiples CK independientes y mono-atributo, PK final mono-atributo, Sin relaciones transitivas entre atributos no primos, → FNBC
- `hm_community(id, com_name, description, avatar_url)`  
Múltiples CK independientes y mono-atributo, PK final mono-atributo, Sin relaciones transitivas entre atributos no primos, → FNBC
- `hm_user_artist(id)`  
Un solo atributo, → FNBC
- `hm_taggeable(id)`  
Un solo atributo, → FNBC
- `hm_artist(id, art_name, description)`  
Única CK mono-atributo, Sin dependencias transitivas entre atributos no primos, → FNBC

## 7.2. ANÁLISIS DE LA FORMA NORMAL DE LAS RELACIONES DEL SISTEMA

---

- `hm_song(id, title, album, exe_date, down_url)`  
Múltiples CK multi-atributo totalmente independientes entre sí, PK final mono-atributo, Sin dependencias entre atributos primos no-CK, → FNBC
- `hm_club(id, club_name, address, town, zip_code, country, region, phone, fax)`  
Múltiples CK multiatributo independientes entre sí, PK final mono-atributo, Sin dependencias entre atributos primos no-CK, Sin dependencias transitivas entre atributos no primos, → FNBC
- `hm_concert(id, title, con_date, max_ticket, club_id)`  
Única CK mono-atributo, Sin dependencias transitivas entre atributos no-primos, → FNBC
- `hm_receives(message_id, receiver_id)`  
Única CK multiatributo, No existen atributos no-primos, → FNBC
- `hm_inbox(message_id, person_id)`  
Única CK multiatributo, No existen atributos no-primos, → FNBC
- `hm_comm_member(comm_id, user_id, member_date)`  
Única CK multiatributo, Un único atributo no-primo, → FNBC
- `hm_comm_admin(comm_id, user_id)`  
Única CK multiatributo, No existen atributos no-primos, → FNBC
- `hm_playlist(user_id, song_id, pos, list_name)`  
Única CK multiatributo, Un único atributo no-primo, → FNBC
- `hm_rating(user_id, song_id, rating)`  
Única CK multiatributo, Un único atributo no-primo, → FNBC
- `hm_listen(user_id, song_id, listen_date)`  
Única CK multiatributo, No existen atributos no-primos, → FNBC
- `hm_tag(user_id, taggeable_id, tag)`  
Única CK multiatributo, No existen atributos no-primos, → FNBC
- `hm_author(artist_id, song_id)`  
Única CK multiatributo, No existen atributos no-primos, → FNBC
- `hm_artist_admin(user_artist_id, artist_id)`  
Única CK multiatributo, No existen atributos no-primos, → FNBC
- `hm_plays_on(artist_id, concert_id, percentage)`  
Única CK multiatributo, Un único atributo no-primo, → FNBC
- `hm_ticket_buy(concert_id, ticket_num, user_id, payment)`  
Única CK multiatributo, Un único atributo no-primo, → FNBC

### 7.3 Código de creación de la base de datos

---

```
1
2  /*
3   Proyecto Harmonic
4   =====
5   Autores:
6       Juan Pedro Bolivar Puente
7       Luca Mefistofeles Conesa Martin-Aragon
8
9   Instalacion de las tablas de la base de datos.
10  */
11
12
13  drop table hm_plays_on;
14  drop table hm_ticket_buy;
15  drop table hm_ticket;
16  drop table hm_concert;
17  drop table hm_club;
18  drop table hm_inbox;
19  drop table hm_receives;
20  drop table hm_comm_member;
21  drop table hm_comm_admin;
22  drop table hm_playlist;
23  drop table hm_rating;
24  drop table hm_listen;
25  drop table hm_tag;
26  drop table hm_author;
27  drop table hm_artist_admin;
28  drop table hm_song;
29  drop table hm_artist;
30  drop table hm_taggeable;
31  drop table hm_user_artist;
32  drop table hm_user;
33  drop table hm_community;
34  drop table hm_person;
35  drop table hm_message;
36
37  drop sequence hm_person_id_seq;
38  drop sequence hm_taggeable_id_seq;
39  drop sequence hm_song_id_seq;
40  drop sequence hm_artist_id_seq;
41  drop sequence hm_club_id_seq;
```

### 7.3. CÓDIGO DE CREACIÓN DE LA BASE DE DATOS

---

```
42 drop sequence hm_concert_id_seq;
43 drop sequence hm_message_id_seq;
44
45 /*
46 ** ENTITIES
47 */
48
49 create table hm_person
50 (
51     id                int ,
52     create_date       date ,
53     primary key (id)
54 );
55
56 create table hm_message
57 (
58     id                int ,
59     sender_id         int ,
60     content           varchar2(2048),
61     send_date         date ,
62     private           char check (private in ('t', 'f')),
63     primary key (id)
64 );
65
66 create table hm_user
67 (
68     /* system data */
69     id                int ,
70     nick              varchar(32) unique not null ,
71     pass_hash         varchar(32) not null ,
72     email             varchar(32) unique not null ,
73     confirm_code      varchar(32),
74     /* profile data */
75     real_name         varchar(128),
76     sex               char(1) check (sex in ('M', 'F')),
77     description       varchar(512),
78     birth_date        date ,
79     homepage_url      varchar(256),
80     avatar_url        varchar(256),
81     foreign key (id) references hm_person(id)
82                         on delete cascade ,
83     primary key (id)
84 );
85
```

### 7.3. CÓDIGO DE CREACIÓN DE LA BASE DE DATOS

---

```
86 create table hm_community
87 (
88     id            int ,
89     com_name      varchar(128) unique not null ,
90     description   varchar(2048),
91     avatar_url    varchar(256),
92     foreign key (id) references hm_person(id)
93                 on delete cascade ,
94     primary key (id)
95 );
96
97 create table hm_user_artist
98 (
99     id            int ,
100    foreign key (id) references hm_user(id)
101                    on delete cascade ,
102    primary key(id)
103 );
104
105 create table hm_taggeable
106 (
107     id            int ,
108     primary key(id)
109 );
110
111 create table hm_artist
112 (
113     id            int ,
114     art_name      varchar(48) not null ,
115     description   varchar(2048),
116     foreign key (id) references hm_taggeable(id)
117                 on delete cascade ,
118     primary key (id)
119 );
120
121 create table hm_song
122 (
123     id            int ,
124     title         varchar(1024) not null ,
125     album         varchar(1024),
126     exe_date      date ,
127     down_url      varchar(256),
128     foreign key (id) references hm_taggeable(id)
129                 on delete cascade ,
```

### 7.3. CÓDIGO DE CREACIÓN DE LA BASE DE DATOS

---

```
130         primary key (id)
131     );
132
133 create table hm_club
134 (
135     id            int ,
136     club_name     varchar(128),
137     address       varchar(512),
138     town          varchar(32),
139     zip_code      varchar(6),
140     region        varchar(32),
141     country       varchar(32),
142     phone         varchar(16),
143     fax           varchar(16),
144     room_size     int ,
145     primary key (id)
146 );
147
148 create table hm_concert
149 (
150     id            int ,
151     title         varchar(1024),
152     con_date      date ,
153     max_ticket    int ,
154     club_id       int ,
155     foreign key (club_id) references hm_club (id)
156                             on delete cascade ,
157     primary key (id)
158 );
159
160 create table hm_ticket
161 (
162     concert_id    int ,
163     num           int ,
164     price         int , /* TODO!!! */
165     foreign key (concert_id) references hm_concert(id)
166                             on delete set null ,
167     primary key (concert_id , num)
168 );
169
170 /*
171 ** RELATIONSHIPS
172 */
173
```

### 7.3. CÓDIGO DE CREACIÓN DE LA BASE DE DATOS

---

```
174 create table hm_receives
175 (
176     message_id int,
177     receiver_id int,
178     foreign key (message_id) references hm_message (id)
179                                     on delete cascade,
180     foreign key (receiver_id) references hm_person (id)
181                                     on delete cascade,
182     primary key (message_id, receiver_id)
183 );
184
185 create table hm_inbox
186 (
187     message_id int,
188     person_id int,
189     foreign key (message_id) references hm_message (id)
190                                     on delete cascade,
191     foreign key (person_id) references hm_person (id)
192                                     on delete cascade,
193     primary key (message_id, person_id)
194 );
195
196 create table hm_comm_member
197 (
198     comm_id int,
199     user_id int,
200     member_date date,
201     foreign key (comm_id) references hm_community (id)
202                                     on delete cascade,
203     foreign key (user_id) references hm_user (id)
204                                     on delete cascade,
205     primary key (comm_id, user_id)
206 );
207
208 create table hm_comm_admin
209 (
210     comm_id int,
211     user_id int,
212     foreign key (comm_id) references hm_community (id)
213                                     on delete cascade,
214     foreign key (user_id) references hm_user (id)
215                                     on delete cascade,
216     primary key (comm_id, user_id)
217 );
```



### 7.3. CÓDIGO DE CREACIÓN DE LA BASE DE DATOS

---

```
218
219 create table hm_playlist
220 (  
221     user_id    int ,  
222     song_id    int ,  
223     pos        int ,  
224     list_name  varchar(32),  
225     foreign key (user_id) references hm_user (id)  
226                     on delete cascade ,  
227     foreign key (song_id) references hm_song (id)  
228                     on delete cascade ,  
229     primary key (user_id , song_id , pos , list_name)  
230 );  
231  
232 create table hm_rating  
233 (  
234     user_id    int ,  
235     song_id    int ,  
236     rating     int check (rating between 0 and 5),  
237     foreign key (user_id) references hm_user (id)  
238                     on delete cascade ,  
239     foreign key (song_id) references hm_song (id)  
240                     on delete cascade ,  
241     primary key (user_id , song_id)  
242 );  
243  
244 create table hm_listen  
245 (  
246     user_id      int ,  
247     song_id      int ,  
248     listen_date  date ,  
249     foreign key (user_id) references hm_user (id)  
250                     on delete cascade ,  
251     foreign key (song_id) references hm_song (id)  
252                     on delete cascade ,  
253     primary key (user_id , song_id , listen_date)  
254 );  
255  
256 create table hm_tag  
257 (  
258     user_id      int ,  
259     taggeable_id int ,  
260     tag          varchar(16),  
261     foreign key (user_id) references hm_user (id)
```

### 7.3. CÓDIGO DE CREACIÓN DE LA BASE DE DATOS

---

```
262                                     on delete cascade ,
263     foreign key (taggeable_id) references hm_taggeable (id)
264                                     on delete cascade ,
265     primary key (user_id , taggeable_id , tag)
266 );
267
268 create table hm_author
269 (
270     artist_id    int ,
271     song_id      int ,
272     foreign key (artist_id) references hm_artist (id)
273                                     on delete cascade ,
274     foreign key (song_id) references hm_song (id)
275                                     on delete cascade ,
276     primary key (artist_id , song_id)
277 );
278
279 create table hm_artist_admin
280 (
281     user_artist_id int ,
282     artist_id      int ,
283     foreign key (user_artist_id) references hm_user_artist (id)
284                                     on delete cascade ,
285     foreign key (artist_id) references hm_artist (id)
286                                     on delete cascade ,
287     primary key (user_artist_id , artist_id)
288 );
289
290 create table hm_plays-on
291 (
292     artist_id    int ,
293     concert_id   int ,
294     percentage   int , /* TODO!! */
295     foreign key (artist_id) references hm_artist (id)
296                                     on delete cascade ,
297     foreign key (concert_id) references hm_concert (id)
298                                     on delete cascade ,
299     primary key (artist_id , concert_id)
300 );
301
302 create table hm_ticket_buy
303 (
304     concert_id    int ,
305     ticket_num    int ,
```

### 7.3. CÓDIGO DE CREACIÓN DE LA BASE DE DATOS

---

```
306         user_id      int ,
307         payment      char, /* TODO!! */
308         foreign key (concert_id , ticket_num)
309                     references hm_ticket (concert_id , num)
310                     on delete cascade ,
311         foreign key (user_id) references hm_user (id)
312                     on delete cascade ,
313         primary key (concert_id , ticket_num , user_id)
314     );
315
316     create sequence hm_person_id_seq
317         start with 1000
318         increment by 1;
319
320     create sequence hm_taggeable_id_seq
321         start with 1000
322         increment by 1;
323
324     create sequence hm_club_id_seq
325         start with 1000
326         increment by 1;
327
328     create sequence hm_concert_id_seq
329         start with 1000
330         increment by 1;
331
332     create sequence hm_song_id_seq
333         start with 1000
334         increment by 1;
335
336     create sequence hm_artist_id_seq
337         start with 1000
338         increment by 1;
339
340     create sequence hm_message_id_seq
341         start with 1000
342         increment by 1;
```