



## Guía de Base

### *Capítulo 3* *Tablas*

## Derechos de autor

---

Este documento tiene derechos de autor © 2020 por el equipo de documentación. Los colaboradores se listan más abajo. Se puede distribuir y modificar bajo los términos de la [GNU General Public License](#) versión 3 o posterior o la [Creative Commons Attribution License](#), versión 4.0 o posterior.

Todas las marcas registradas mencionadas en esta guía pertenecen a sus propietarios legítimos.

## Colaboradores

Este libro está adaptado de versiones anteriores del mismo.

### De esta edición

Pulkit Krishna  
Juan Peramos

Dan Lewis  
Ignacio Alcalá

Jean Hollis Weber  
B. Antonio Fernández

### De ediciones previas

Robert Großkopf  
Hazel Russman  
Steve Schwettman

Jost Lange  
Jean Hollis Weber

Jochen Schiffers  
Dan Lewis

## Comentarios y sugerencias

Puede dirigir cualquier clase de comentario o sugerencia acerca de este documento a: [documentation@es.libreoffice.org](mailto:documentation@es.libreoffice.org).



### Nota

Todo lo que envíe a la lista de correo, incluyendo su dirección de correo y cualquier otra información personal que escriba en el mensaje se archiva públicamente y no puede ser borrado.

---

## Fecha de publicación y versión del programa

Versión en español publicada el 28 de julio de 2021. Basada en la versión 6.2 de LibreOffice.

## Uso de LibreOffice en macOS

---

Algunas pulsaciones de teclado y opciones de menú son diferentes en macOS de las usadas en Windows y Linux. La siguiente tabla muestra algunas sustituciones comunes para las instrucciones dadas en este capítulo. Para una lista detallada vea la ayuda de la aplicación.

<b>Windows o Linux</b>	<b>Equivalente en Mac</b>	<b>Efecto</b>
<b>Herramientas &gt; Opciones</b> opción de menú	<b>LibreOffice &gt; Preferencias</b>	Acceso a las opciones de configuración
<i>Clic con el botón derecho</i>	<i>Control+clic o clic derecho</i> depende de la configuración del equipo	Abre menú contextual
<i>Ctrl (Control)</i>	⌘ ( <i>Comando</i> )	Utilizado con otras teclas
<i>F5</i>	<i>Mayúscula+⌘+F5</i>	Abre el navegador
<i>F11</i>	⌘+T	Abre la ventana de estilos y formato

## Contenido

---

<b>Derechos de autor</b> .....	<b>2</b>
Colaboradores.....	2
De esta edición.....	2
De ediciones previas.....	2
Comentarios y sugerencias.....	2
Fecha de publicación y versión del programa.....	2
<b>Uso de LibreOffice en macOS</b> .....	<b>2</b>
<b>Información general sobre tablas</b> .....	<b>5</b>
<b>Relaciones entre tablas</b> .....	<b>5</b>
Relaciones entre tablas en Bases de Datos.....	6
Relaciones uno a muchos.....	6
Relaciones de muchos a muchos.....	6
Relaciones uno a uno.....	7
Tablas y relaciones en la base de datos de ejemplo.....	8
Tabla de adición de artículos (Media).....	8
Tabla de préstamos (Loan).....	9
Tabla de administración de usuarios (Reader).....	10
<b>Crear tablas</b> .....	<b>11</b>
Crear una base de datos utilizando la interfaz gráfica de usuario.....	12
Claves primarias.....	16
Formatear campos.....	17
Crear un índice.....	18
Problemas al modificar tablas.....	20
Limitaciones del diseño de tablas gráficas.....	21
Entrada directa de órdenes SQL.....	22
Crear tabla.....	22
Modificar la tabla.....	25
Eliminar tablas.....	28
<b>Vincular tablas</b> .....	<b>28</b>
<b>Introducir datos en tablas</b> .....	<b>31</b>
Entrada utilizando la interfaz de base.....	32
Ordenar Tablas.....	34
Buscar en Tablas.....	34
Filtrar tablas.....	36
Entrada directa usando SQL.....	38
Introducir nuevos registros.....	38
Editar registros existentes.....	38
Eliminar registros existentes.....	39
Importar datos de otras fuentes.....	39
Agregar registros importados a una tabla existente.....	40
Crear una nueva tabla para datos importados.....	41
División de datos al importar.....	43
Problemas con estos métodos de entrada de datos.....	44

## Información general sobre tablas

---

Las bases de datos almacenan datos en tablas. La principal diferencia con las tablas en una hoja de cálculo normal es que los campos en los que se escriben los datos deben estar claramente definidos de antemano. Por ejemplo, una base de datos no permite que un campo de texto contenga números para usar en los cálculos. Dichos números se muestran, pero solo como cadenas, cuyo valor numérico real es cero. Del mismo modo, las imágenes no se pueden incluir en todos los tipos de campos.

Los detalles sobre qué tipos de datos están disponibles, se pueden obtener en Base desde la ventana Diseño de tabla. Se muestran en el «Apéndice» de esta guía.

Las bases de datos simples se basan en una sola tabla. Todos los elementos de datos se ingresan de forma independiente, lo que puede conducir a la entrada múltiple de los mismos datos. De este modo, se puede crear una libreta de direcciones simple para uso privado. Sin embargo, la libreta de direcciones de una escuela o una asociación deportiva podría contener tantas repeticiones de códigos postales y calles que estos datos se ubican mejor en una o incluso dos tablas separadas.

Almacenar datos en tablas separadas ayuda a:

- Reducir la entrada repetida del mismo contenido.
- Evitar errores de ortografía debido a la entrada repetida.
- Mejora el filtrado de datos en las tablas mostradas.

Al crear una tabla, siempre debe considerar si se van a producir múltiples repeticiones, especialmente de texto o imágenes (que consumen mucho almacenamiento) en la tabla. Si es así, debe exportarlos a otra tabla. Cómo hacerlo, en principio, se describe en el «Capítulo 1» de *Introducción a Base*, en la sección «Una base de datos simple: ejemplo de prueba en detalle».



### Nota

Una base de datos relacional es un grupo de tablas que están vinculadas entre sí a través de atributos comunes. El propósito de una base de datos relacional es en lo máximo la entrada duplicada de elementos de datos. Se deben evitar redundancias.

Esto se puede lograr al:

- Separar el contenido en tantos campos únicos como sea práctico (por ejemplo, en lugar de usar un campo para una dirección completa, use campos separados para el número de casa, calle, ciudad y código postal).
- Evitar datos duplicados para un campo en varios registros (por ejemplo, importando el código postal y la ciudad desde otra tabla).

Estos procedimientos se conocen como *Normalización de la base de datos*.

---

## Relaciones entre tablas

---

Este capítulo explica muchos de estos pasos en detalle, utilizando una base de datos de ejemplo para una biblioteca: **media\_without\_macros**. La construcción de las tablas para esta base de datos es un trabajo extenso, ya que cubre no solo la adición de elementos a una biblioteca, sino también el préstamo posterior de los mismos.

## Relaciones entre tablas en Bases de Datos

Las tablas en una base de datos interna HSQLDB siempre tienen un campo único y distintivo: la *clave primaria*. Este campo se tiene que definir antes de que se puedan escribir datos en la tabla. Usando este campo, se pueden encontrar registros concretos en una tabla.

En ciertos casos, se forma una clave primaria a partir de varios campos juntos. Estos campos tienen que ser únicos. Esto se conoce como una *clave primaria compuesta*.

La *Tabla 2* puede tener un campo que refiere a un registro en la *Tabla 1*. La clave primaria de la *Tabla 1* se escribe como un valor en el campo de la *Tabla 2*.

La *Tabla 2* ahora tiene un campo que apunta al campo clave de otra tabla, conocido como *clave externa*. Esta clave externa existe en la *Tabla 2* además de su clave primaria.

Cuantas más relaciones haya entre tablas, más compleja será la tarea de diseño.

La Figura 1 muestra la estructura general de la base de datos de ejemplo. Para visualizar mejor su contenido puede ampliar la página.

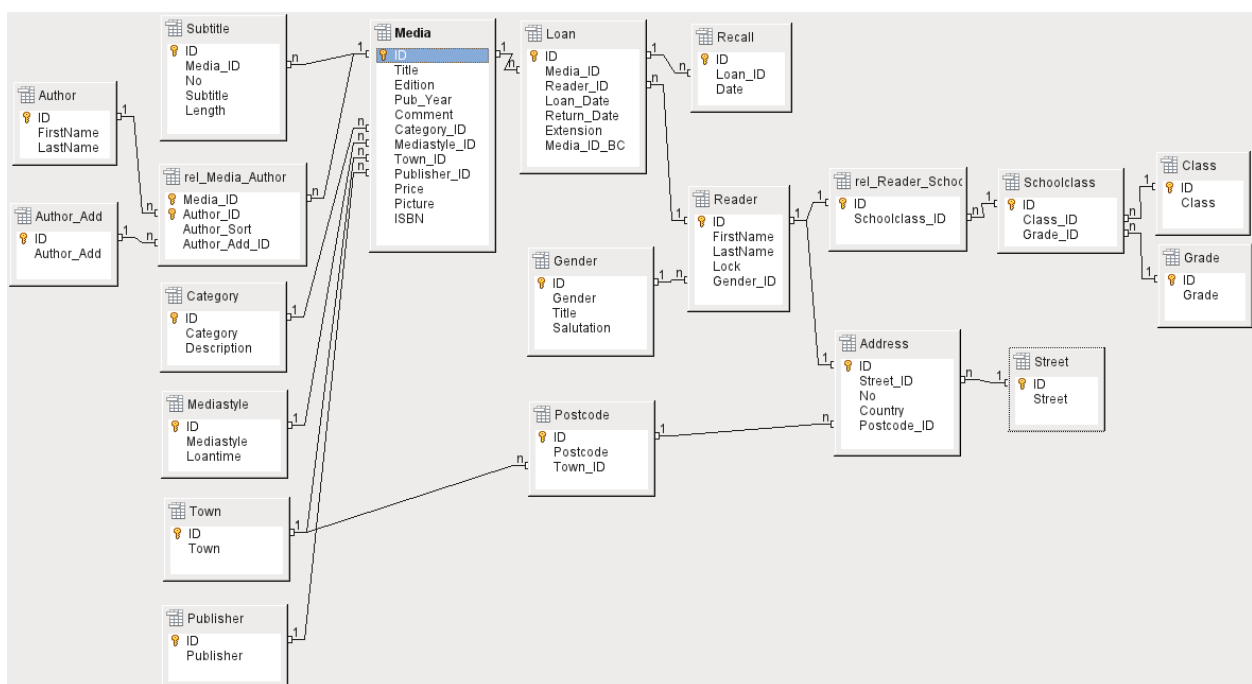


Figura 1: Diagrama de Relaciones en la base de datos de ejemplo: media\_without\_macros

### Relaciones uno a muchos

La base de datos **media\_without\_macros** de ejemplo, enumera los títulos de los artículos, en una tabla. Debido a que los títulos pueden tener múltiples subtítulos o, a veces, ninguno, los subtítulos se almacenan en una tabla separada.

Esta relación se conoce como *uno a muchos* (1 : n). Se pueden asignar muchos subtítulos a un artículo, por ejemplo, los títulos de muchas pistas para un CD de música. La clave principal para la tabla de artículos (*Media*) se almacena como una clave externa en la tabla de subtítulos (*Subtitle*). La mayoría de las relaciones entre tablas en una base de datos son relaciones de uno a muchos.

### Relaciones de muchos a muchos

Una base de datos para una biblioteca puede contener una tabla para los nombres de los autores y una tabla para sus obras, denominada *Media*. La conexión entre un autor y los libros que ha escrito, es obvia. La biblioteca puede contener más de un libro de un autor. Pero también puede

contener libros con el mismo título pero de diferentes autores. Esta relación se conoce como *muchos a muchos* ( $n : m$ ). Dichas relaciones necesitan una tabla que actúe como intermediaria entre las dos tablas en cuestión. En la Figura 2, la tabla *rel\_Media\_Author* es una tabla intermedia. Por lo tanto, en la práctica, la relación  $n : m$  se resuelve al tratarla como dos relaciones  $1 : n$ . En la tabla intermedia, *Media\_ID* puede aparecer más de una vez, al igual que *Author\_ID*. Pero al usarlos como un par, no hay duplicación: no hay dos pares idénticos. Por lo tanto, este par cumple los requisitos de una clave primaria para la tabla intermedia.

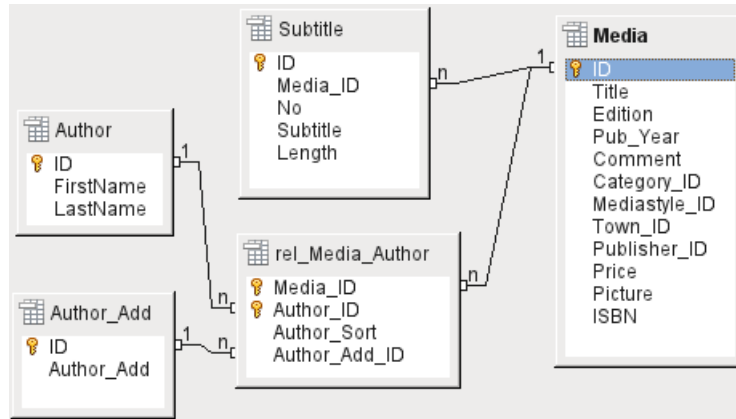


Figura 2: Ejemplo relación 1: n; y relación n: m



## Nota

Un autor es único al tener un nombre y apellido (aunque puede darse alguna rara coincidencia), así mismo un libro también es único al tener un número ISBN. Para un par determinado de estos valores, estas características hacen que el par sea único.

## Relaciones uno a uno

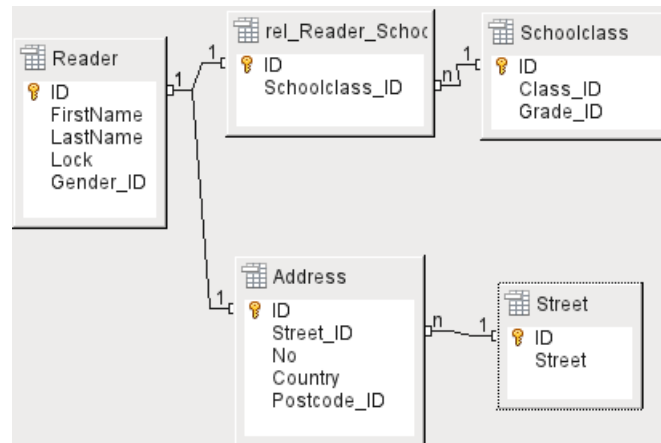


Figura 3: Ejemplo de relación 1: 1

La base de datos de la biblioteca, descrita anteriormente, requiere una tabla para lectores. En esta tabla solo se planificaron por adelantado los campos que son directamente necesarios.

Pero para una base de datos de una escuela, también se puede necesitar la clase escolar a la que pertenece el lector. Con los registros de la tabla, *Schoolclass*, se pueden encontrar las direcciones de los prestatarios cuando sea necesario. Por lo tanto, no es necesario incluir estas direcciones en la base de datos.

La relación entre la tabla de las clases escolares de los alumnos *Schoolclass* está separada de la tabla del lector, *Reader*, porque la asignación de las clases escolares no es siempre apropiada. De esto surge una relación 1 : 1 entre el lector y la clase escolar individual asignada.

En una base de datos para una biblioteca pública, se requieren las direcciones de los lectores. Para cada lector hay una sola dirección. Si hay varios lectores en la misma dirección, esta estructura requeriría que la dirección se ingrese nuevamente, ya que la clave primaria de la tabla *Reader* se ingresa directamente como clave principal en la tabla *Address*. La clave primaria y la clave externa son una y la misma en la tabla de direcciones. Por lo tanto esta es una relación 1 : 1.

Una relación 1 : 1 no significa que para cada registro en una tabla haya un registro correspondiente en otra tabla. Al menos habrá un único un registro correspondiente. Por lo tanto una relación 1 : 1 lleva a la exportación de campos que se completarán con contenido solo para algunos de los registros.

## Tablas y relaciones en la base de datos de ejemplo

La base de datos de ejemplo *media\_without\_macros.odt* debe atender a tres necesidades: adición y eliminación de artículos, gestión de préstamos y administración de usuarios.

### Tabla de adición de artículos (Media)

Primero deben agregarse los artículos a la base de datos para que una biblioteca pueda trabajar con ellos. Sin embargo, para un resumen simple de una colección de artículos en casa, puede crear bases de datos más fáciles con el asistente; eso podría ser suficiente para uso doméstico.

La tabla central para la adición de artículos es la tabla *Media* (ver Figura 4).

En esta tabla, se supone que todos los campos que se ingresan directamente no se usan también para otros artículos con el mismo contenido. Por lo tanto, se debe evitar la duplicación.

Por esta razón, los campos planificados en la tabla incluyen el título, el ISBN, una imagen de la portada y el año de publicación. La lista de campos se puede ampliar si es necesario. En este caso los administradores de la biblioteca pueden querer incluir, por ejemplo, campos para el tamaño (número de páginas), el título de la serie, etc.

La tabla *Subtitle* contiene el contenido detallado de los CD. Como un CD puede contener varias piezas de música, un registro de las piezas individuales en la tabla principal requeriría muchos campos adicionales (*Subtitle\_1*, *Subtitle\_2*, etc.) o el mismo elemento tendría que ingresarse muchas veces. La tabla *Subtitle*, por lo tanto, se encuentra en una relación n : 1 con la tabla *Media*.

Los campos de la tabla *Subtitle* son (además del subtítulo en sí) el número de secuencia del subtítulo y la duración de la pista. El campo *Length* debe definirse primero como un campo de tiempo. De esta manera, la duración total del CD se puede calcular y mostrar en un resumen si fuese necesario.

Los autores tienen una relación n : m con los artículos. Un artículo puede tener varios autores, y un autor puede haber creado varios artículos. Esta relación está controlada por la tabla *rel\_Media\_Author*. La clave principal de esta tabla de vinculación es la clave externa, formada a partir de las tablas *Author* y *Media*. La tabla *rel\_Media\_Author* incluye una ordenación adicional (*Author\_Sort*) de autores, por ejemplo, por la secuencia en la que se nombran en el artículo. Además, se agrega una etiqueta complementaria como Productor, Fotógrafo, etc. al autor cuando sea necesario.

*Category*, *Mediastyle*, *Town*, y *Publisher* tienen una relación 1 : n.

Para *Category*, una pequeña biblioteca puede usar algo como Arte o Biología. Para bibliotecas más grandes, están disponibles sistemas generales para bibliotecas. Estos sistemas proporcionan



abreviaturas y descripciones completas. Por lo tanto, ambos campos aparecen en la tabla *Category*.

El *Mediastyle* está vinculado al período de préstamo en tiempo real. Por ejemplo, los DVD de video pueden, en principio, tener un período de préstamo de 7 días, pero los libros pueden prestarse por 21 días. Si el período del préstamo se quiere vincular a algún otro criterio, habrá que hacer los cambios oportunos en su metodología.

La tabla *Town* no solo sirve para almacenar datos de ubicación de los artículos, sino también para almacenar las ciudades usadas en las direcciones de los usuarios.

Dado que *Publishers* también se repiten con frecuencia, se les proporciona una tabla separada.

La tabla *Media* tiene en total cuatro claves externas y una clave primaria, que se utiliza como clave externa en dos tablas, como se muestra en la Figura 4.

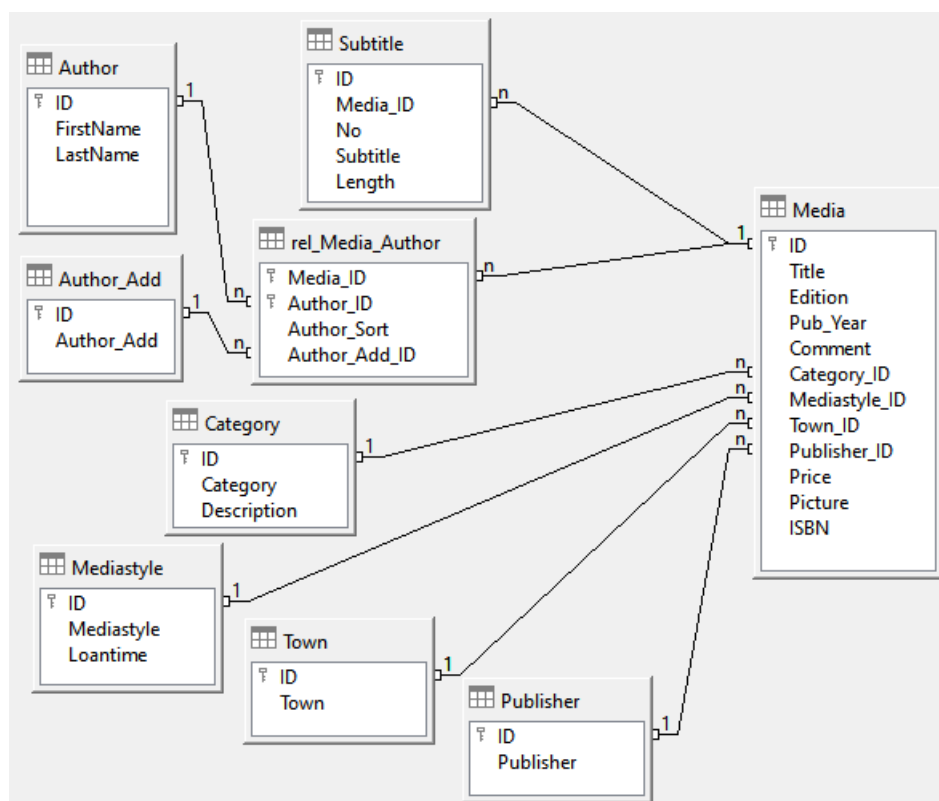


Figura 4: Adición de artículos (*Media*)

### Tabla de préstamos (*Loan*)

La tabla central es *Loan* (ver Figura 5). Es el enlace entre las tablas *Media* y *Reader*.

Cuando se devuelve un artículo, muchos de sus datos se pueden eliminar ya que ya no son necesarios. Pero dos de esos campos no deberían ser eliminados: *ID* y *Loan\_Date*. La primera es la clave principal. El segundo indica cuándo se prestó el artículo. Tiene dos propósitos. Primero, es útil para determinar los artículos más populares. En segundo lugar, si se observa que el artículo está dañado cuando se entrega de nuevo, este campo mostrará quién fue la última persona que lo tomó prestado. Además, el *Return\_Date* se registra cuando se devuelve el artículo.

Del mismo modo, los avisos de vencimiento de préstamo se integran en el procedimiento de préstamo. Cada aviso se ingresa por separado en la tabla *Recall* para que se pueda determinar el número total de avisos.

Además de un período de extensión en semanas, hay un campo adicional en el registro del préstamo que permite que los artículos se presten usando un escáner de código de barras

(*Media\_ID\_BC*). Los códigos de barras contienen, además del *Media\_ID* individual, un dígito de verificación que el escáner puede usar para determinar si el valor escaneado es correcto. Este campo de código de barras se incluye aquí solo para fines de prueba. Sería mejor si la clave principal de la tabla de artículos (*Media*) se pudiera ingresar directamente en forma de código de barras, o si se usara una macro para eliminar el dígito de verificación del número de código de barras ingresado antes del almacenamiento.

Finalmente, necesitamos conectar *Reader* con *Loan*. En la tabla del lector, solo se incluyen en el plan el nombre, un bloqueo opcional, y una clave externa que vincula a la tabla *Gender*.

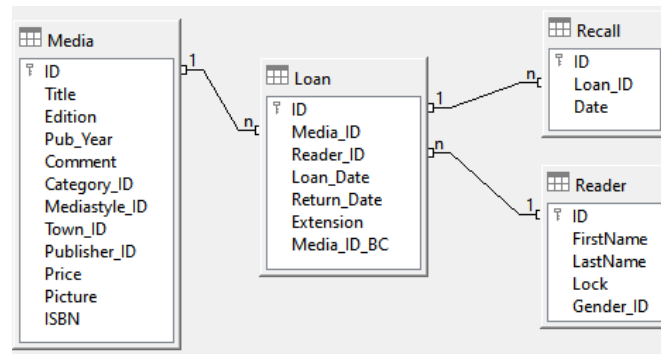


Figura 5: Préstamos (*Loan*)

### Tabla de administración de usuarios (*Reader*)

Para este diseño de tabla, se prevén dos escenarios. La cadena de tablas que se muestra en la figura 6 está diseñada para bibliotecas escolares. Aquí no hay necesidad de direcciones, ya que los alumnos pueden ser contactados a través de la escuela. Los avisos no necesitan enviarse por correo postal, sino que pueden distribuirse internamente. La cadena de direcciones es necesaria en el caso de las bibliotecas públicas. Aquí debe ingresar los datos que serán necesarios para la creación de cartas de comunicación con el lector.

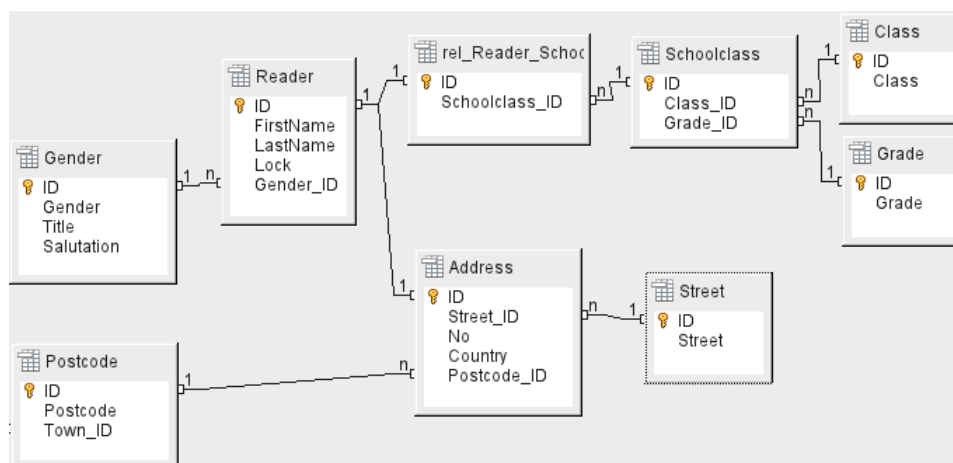


Figura 6: Lectores: una cadena de clase escolar y otra de direcciones

La tabla *Gender* garantiza que se use el saludo correcto en los avisos. La redacción de avisos se puede automatizar en la medida de lo posible. Además, algunos nombres de pila pueden ser igualmente masculinos o femeninos. Por lo tanto, se requiere una lista separada de género incluso cuando los avisos se escriben a mano.

La tabla *rel\_Reader\_Schoolclass*, como la tabla *Address*, tiene una relación 1:1 con la tabla *Reader*. Esto se eligió porque podría ser necesaria la clase de la escuela o la dirección. De lo contrario, el *Schoolclass\_ID* podría colocarse directamente en la tabla del alumno, Lo mismo ocurriría con el contenido completo de la tabla de direcciones en un sistema de biblioteca pública.

Una *School class* generalmente contiene una designación de año y un sufijo de nivel. En una escuela de 4 niveles, este sufijo puede ir de la **a** a la **d**. El sufijo se ingresa en la tabla *Class*. El año está en una tabla *Grade* separada. De esa manera, si los lectores ascienden de clase al final de cada año escolar, simplemente puede cambiar la entrada del año para todos.

La tabla *Address* también está dividida. La calle se almacena por separado porque los nombres de las calles dentro de un área a menudo se repiten. El código postal y la ciudad están separados porque a menudo hay varios códigos postales para una sola área y, por lo tanto, hay más códigos postales que ciudades. Por eso en comparación con la tabla *Address*, la tabla *Postcode* contiene significativamente menos registros y la tabla *Town* aún menos.

La forma en que se utiliza esta estructura de tabla se explica más adelante en el Capítulo 4, «Formularios», de esta guía.

## Crear tablas

La mayoría de los usuarios de LibreOffice generalmente usarán la interfaz gráfica de usuario (GUI) exclusivamente para crear tablas. La entrada directa de órdenes SQL se hace necesaria cuando, por ejemplo, un campo debe insertarse posteriormente en una posición particular, o un valor estándar debe establecerse después de que se haya guardado la tabla.

Terminología de la tabla: la siguiente imagen muestra la división estándar de las tablas en columnas y filas.

Tabla (TABLE)								
	Columna (COLUMN)				Columna (COLUMN)			
	Nombre campo (FIELD)	Tipo campo (TYPE)	Vacío (NULL)	Predeterminado (DEFAULT)	Nombre campo (FIELD)	Tipo campo (TYPE)	Vacío (NULL)	Predeterminado (DEFAULT)
Fila (ROW)	Registro							

Los registros de datos se almacenan en una sola fila de la tabla. Las columnas individuales se definen en gran medida por el campo, el tipo y las reglas que determinan si el campo puede estar vacío. Según el tipo, también se puede determinar el tamaño del campo en caracteres. Además, se puede especificar un valor predeterminado para usar cuando no se ingresó nada en el campo.

En la interfaz de Base, los términos para una columna se describen de una manera diferente, como se muestra a continuación.

Notaciones en la interfaz de Base			
Columna (COLUMN)			
		Propiedades de campo	
Nombre campo (FIELD)	Tipo campo (TYPE)	Entrada requerida (NULL / NOT NULL)	Valor predeterminado (DEFAULT)

Field se convierte en Field Name, Type se convierte en Field Type. Field Name y Field Type se ingresan en el área superior de la ventana Table Design. En el área inferior, tiene la oportunidad de establecer, bajo las propiedades de Field, las otras propiedades de columna, en la medida en que se puedan establecer utilizando la interfaz. Las limitaciones incluyen establecer el valor predeterminado de un campo de fecha en la fecha real de entrada. Esto solo es posible mediante el uso de la orden SQL apropiada (consulte “Entrada directa de órdenes SQL” en la página 22).



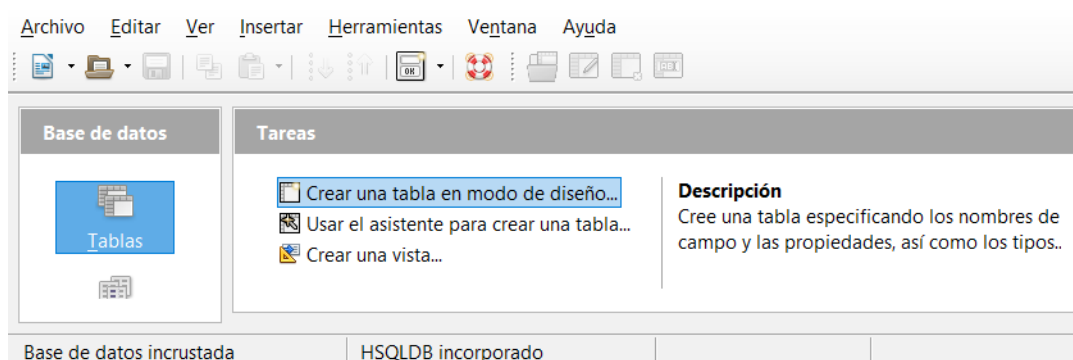
## Nota

Valores predeterminados: Este término en la interfaz no significa lo que el usuario de la base de datos generalmente entiende como un valor predeterminado. La interfaz muestra un cierto valor visualmente que se guarda con los datos.

El valor predeterminado en una base de datos se almacena en la definición de la tabla. Luego se escribe en un campo de un nuevo registro de datos siempre que esté vacío. Los valores predeterminados de SQL no aparecen al editar las propiedades de la tabla.

## Crear una base de datos utilizando la interfaz gráfica de usuario.

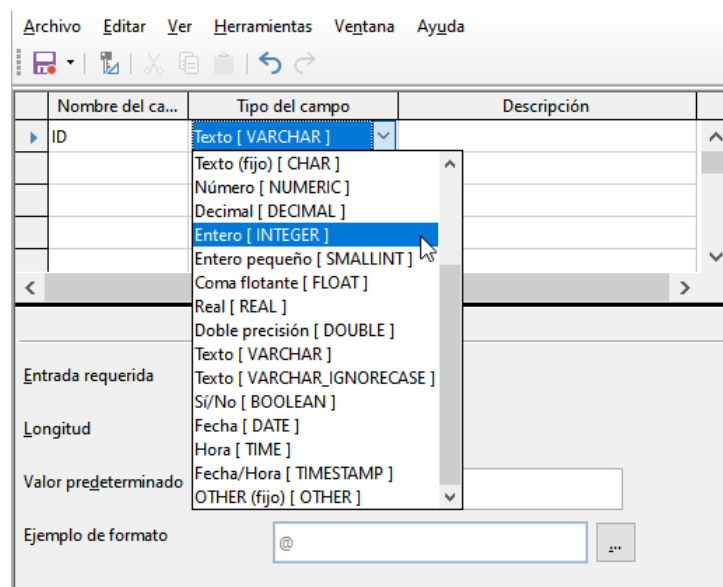
La creación de la base de datos utilizando la interfaz de usuario gráfica (GUI) se explica paso a paso, utilizando la tabla *Media* como ejemplo.



Inicie el editor de tablas haciendo clic en *Crear una tabla en modo de diseño*.

### 1) Campo ID:

- a) En la primera columna, ingrese la ID del nombre del campo. Luego presione la tecla Tab para moverse a la columna Tipo del campo. A continuación haga clic con el ratón en la siguiente columna para seleccionarla o presione la tecla Intro.



- b) El valor predeterminado es para *Típo del campo* es Texto [VARCHAR], Cámbielo a Entero [INTEGER] mediante la lista desplegable.

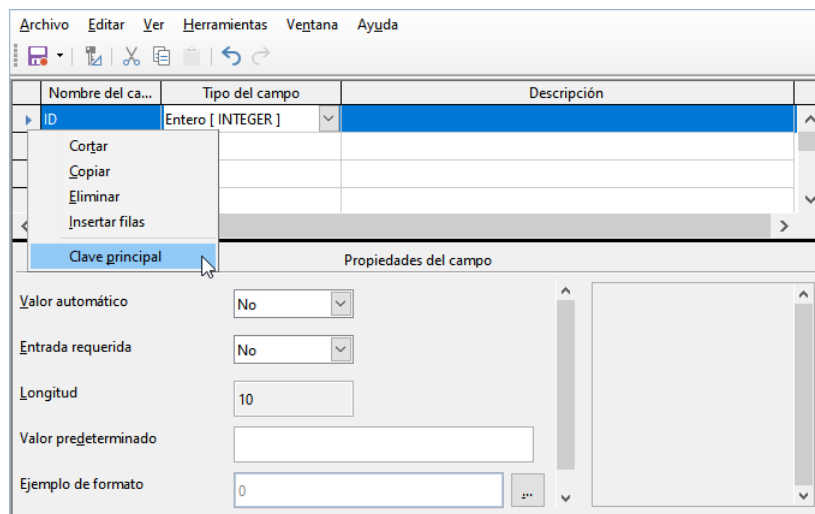
Los enteros pueden almacenar un número de hasta 10 dígitos. Además, Integer es el único tipo disponible en la interfaz que puede recibir un valor de incremento automático.



## Consejo

Para realizar una selección rápida de la lista Tipo del campo con el teclado, presione la tecla correspondiente a la primera letra que elija. Presionar repetidamente esta tecla puede usarse para cambiar la selección. Por ejemplo, presionar D puede cambiar su selección de Fecha a Fecha / Hora o a Decimal.

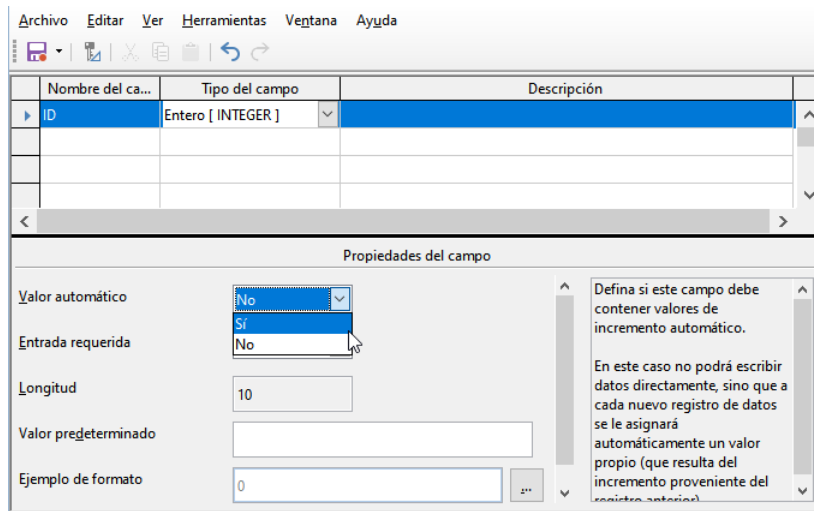
- c) Establezca el campo *ID* como la clave principal haciendo clic con el botón derecho del ratón en el triángulo verde a la izquierda de su fila y seleccionando *Clave principal* en el menú contextual. Aparecerá el símbolo de una llave antes de la identificación.



## Nota

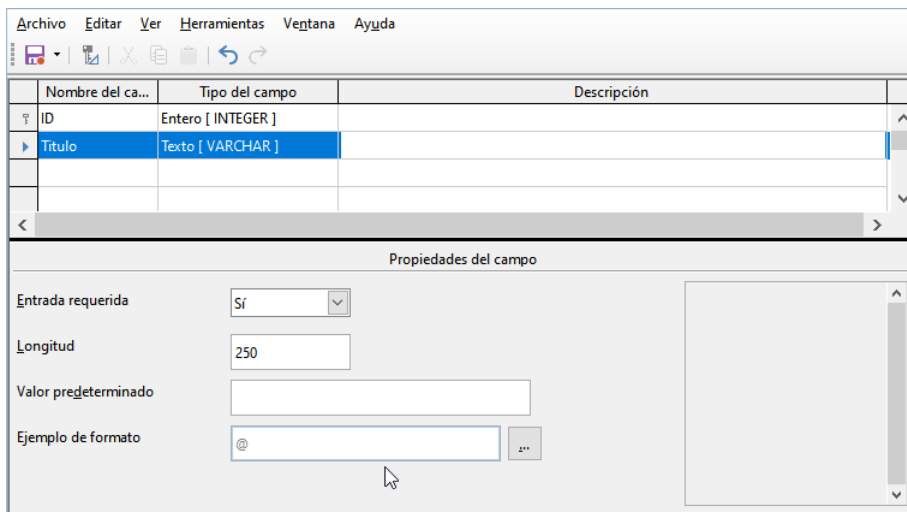
La clave principal tiene un solo propósito: identificar de forma exclusiva el registro. Por lo tanto, puede usar un nombre arbitrario para este campo. En el ejemplo, hemos usado el nombre más usado: ID (abreviatura de identificación).

- d) En *Propiedades del campo* para el campo *ID*, cambie el *Valor automático* a *Sí*. Esto hace que la clave primaria se incremente automáticamente. En la base de datos interna, la cuenta comienza en 0.



El valor automático solo se puede configurar para un campo en una tabla. Al elegir Sí, este campo se configura automáticamente como la clave primaria si aún no se había configurado una.

- 2) El campo siguiente es *Title*.
  - a) El nombre del campo *Title* se ingresa en la columna Nombre del campo.
  - b) El tipo de campo no necesita cambiarse, ya está establecido como Texto [VARCHAR].



- c) En Propiedades del campo, cambie *Entrada requerida* de No a Sí. Un artículo sin título no tendría sentido.
- d) Ajuste también la longitud del campo. La longitud predeterminada es de 100 en las versiones recientes de LibreOffice, pero debe aumentarse a 250 para los títulos de artículos.

	Nombre del campo	Tipo del campo	Descripción
PK	ID	Entero [ INTEGER ]	
	Title	Texto [ VARCHAR ]	
	Edition	Texto [ VARCHAR ]	No. de edición. nueva edición, etc.
	Pub_Year	Entero pequeño [ SMALLINT ]	Año de publicación

- e)
  - 3) La columna *Descripción* para todos los campos podrá ser cualquier cosa. Esta columna también se puede dejar vacía. Solo sirve para explicar el contenido del campo para quienes deseen ver la definición de la tabla.

- 4) Para el campo *Pub\_Year*, se ha elegido el tipo Entero pequeño [SMALLINT]. Puede contener un número entero con un tamaño máximo de 5 dígitos. La fecha de publicación no se utiliza en los cálculos, pero definirla como entero asegura que no contendrá caracteres alfabéticos.

	Nombre del campo	Tipo del campo	Descripción
☒	ID	Entero [ INTEGER ]	
	Title	Texto [ VARCHAR ]	
	Edition	Texto [ VARCHAR ]	No. de edición. nueva edición, etc.
	Pub_Year	Entero pequeño [ SMALLINT ]	Año de publicación
	Comment	Texto [ VARCHAR ]	
▶	Category_ID	Entero [ INTEGER ]	Clave externa "Category"

- 5) Para el campo *Category\_ID*, se elije el tipo Entero [ INTEGER ]. Este campo almacenará el mismo valor que la clave primaria de la tabla *Category*, (*ID*) las claves primarias tienen que ser de este tipo, por lo que lo que se ingrese aquí como clave externa tiene que ser del mismo tipo. Esto también se aplica a los siguientes campos de claves externas *Mediastyle\_ID*, *Town\_ID* y *Publisher\_ID*.

	Comment	Texto [ VARCHAR ]	
	Category_ID	Entero [ INTEGER ]	Clave externa "Category"
	Mediastyle_ID	Entero [ INTEGER ]	Clave externa "Mediastyle"
	Town_ID	Entero [ INTEGER ]	Clave externa "Town"
	Publisher_ID	Entero [ INTEGER ]	Clave externa "Publisher"
▶	Price	Número [ NUMERIC ]	Declaración del valor (€, \$, £)

Propiedades del campo

Entrada obligatoria	No
Longitud	6
Decimales	2

- 6) Para el campo *Price*, use el tipo Número [ NUMERIC ] o [ DECIMAL ]. Ambos tipos de campo pueden contener valores con un punto decimal. En Propiedades del campo, establezca una longitud de 6 caracteres. Esto debería ser suficiente para los precios de nuestros artículos.
- El número de lugares decimales se establece en 2. Esto proporciona un precio máximo de 9999.99 ya que el punto decimal en sí no está incluido en el dato del campo.
  - No es necesario poner el carácter \$ en el formato, ya que una fórmula lo manejará.

	Publisher_ID	Entero [ INTEGER ]	Clave externa "Publisher"
	Price	Número [ NUMERIC ]	Declaración del valor (€, \$, £)
	Picture	Imagen [ LONGVARBINARY ]	
▶	ISBN	Número [ NUMERIC ]	max. 13 - lugar del número ISBN

Propiedades del campo

Entrada obligatoria	No
Longitud	13

- 7) Para *ISBN*, use el tipo Número [NUMERIC]. Se puede establecer con exactitud la longitud de campo correcta para este campo. Los ISBN tienen 10 o 13 caracteres de longitud. Se almacenarán como números sin un separador. La longitud se establece en un máximo de 13 caracteres. El número de decimales se establece en cero.
- 8) Guarde la tabla con el nombre *Media*.

La tabla principal para la base de datos de ejemplo ya está pues creada.

Todas las otras tablas se pueden crear de manera similar. Tenga cuidado de que los tipos de campo y las propiedades de campo coincidan con lo que se va a almacenar en esos campos. Esta es la diferencia con una hoja de cálculo, en la que una columna puede contener una mezcla de propiedades.



## Nota

El orden de los campos en la tabla solo se podrá cambiar cuando la tabla se guarde por primera vez en la interfaz. Cuando los datos se ingresan posteriormente directamente en la tabla, el orden de los campos es fijo. Sin embargo, el orden todavía se puede cambiar libremente en consultas, formularios e informes.

## Claves primarias

Al guardar una tabla creada en modo diseño, si no se establece una clave principal, aparecerá un cuadro de diálogo preguntándole si se debe crear una clave primaria. Esto indica que falta un campo significativo en la tabla. Sin una clave primaria, la base de datos HSQLDB no puede acceder a la tabla. Este campo generalmente se denomina *ID* y se le da el tipo INTEGER con *Valor automático* para incrementar automáticamente su valor cuando se añade un registro. Al hacer clic en *Sí* en el cuadro de diálogo, se crea automáticamente un campo de clave principal. Al hacer clic en *No* o *Cancelar* en el cuadro de diálogo **no** se creará un nuevo campo de clave primaria, pero, puede designar un campo existente, haciendo clic con el botón derecho en la flecha verde a la izquierda del campo correspondiente y entonces la tabla estará operativa.

También puede usar una combinación de campos como su clave principal. Los campos deben declararse como clave principal juntos (mantenga presionada la tecla *Control* o *Shift*). Luego, al hacer clic con el botón derecho, la combinación de los campos resaltados será la clave principal.

Si la información se importa a esta tabla desde otras (por ejemplo, una base de datos de direcciones con códigos postales y ciudades almacenadas externamente), se debe incluir un campo con el mismo tipo de datos que la clave primaria de la otra tabla. Supongamos que la tabla *Postcode* tiene como clave primaria un campo llamado *ID* con el tipo Entero minúsculo. La tabla de direcciones debe tener un campo *Postcode\_ID* con el mismo tipo. Lo que se ingresa en la tabla de direcciones es siempre el número que sirve como clave primaria para la ubicación dada en la tabla *Postcode*. Esto significa que la tabla *Address* ahora tiene una clave externa además de su propia clave primaria.

Una regla fundamental para nombrar campos en una tabla es que no hay dos campos que puedan tener el mismo nombre. Por lo tanto, no puede tener un segundo campo llamado *ID* como una clave externa en la tabla *Address*.

El tipo de campo solo puede modificarse de forma limitada. Siempre se permite aumentar una propiedad (por ejemplo la longitud de un campo), ya que todos los valores ya ingresados coincidirán con las nuevas condiciones. Al disminuir una propiedad pueden surgir problemas y puede conducir a una pérdida de datos.

Los campos de hora en el diseño de tablas no se pueden crear para contener fracciones de segundo. Para eso, necesita hacerlo en dos pasos: Primero crear un campo de Marca de Tiempo. en el diseño y posteriormente modificar este campo usando **Herramientas > SQL**.



```
ALTER TABLE "Table_name" ALTER COLUMN "Field_name" TIMESTAMP (6)
```

El parámetro "6" hace que el campo Timestamp sea capaz de almacenar fracciones de segundo.

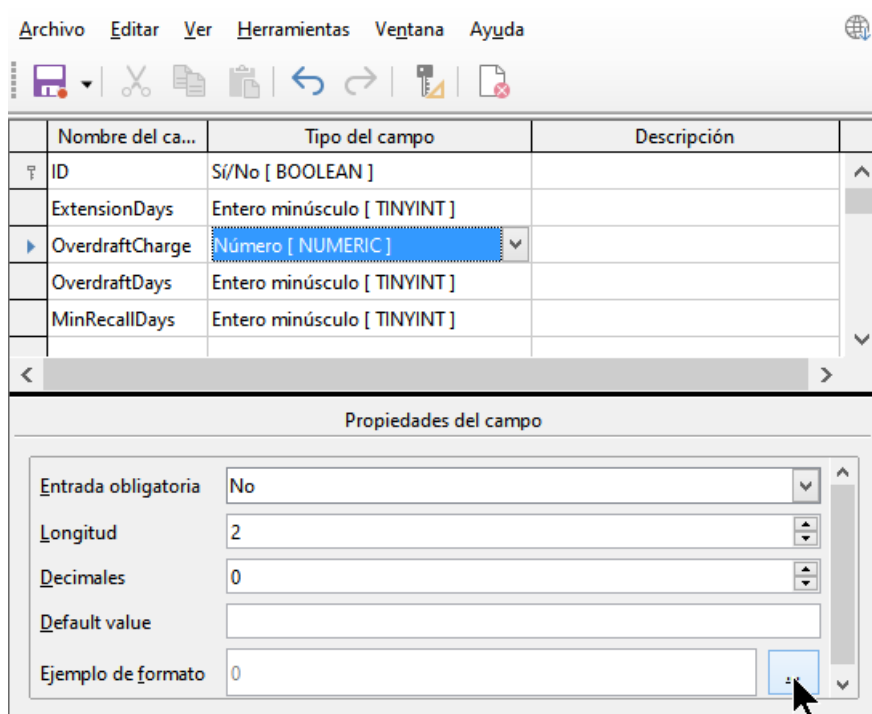
### Formatear campos

El formateo presenta los valores de la base de datos al usuario y permite la entrada de valores dependiendo de las convenciones de entrada normales en ese país. Sin un formato establecido, las cifras decimales se representarán con un punto, cuando la mayoría de los países europeos usan una coma (4.21 en lugar de 4,21). Los valores de fecha se presentan en la forma 2014-12-22. Al configurar el formato, debe tener en cuenta las normas locales del país.

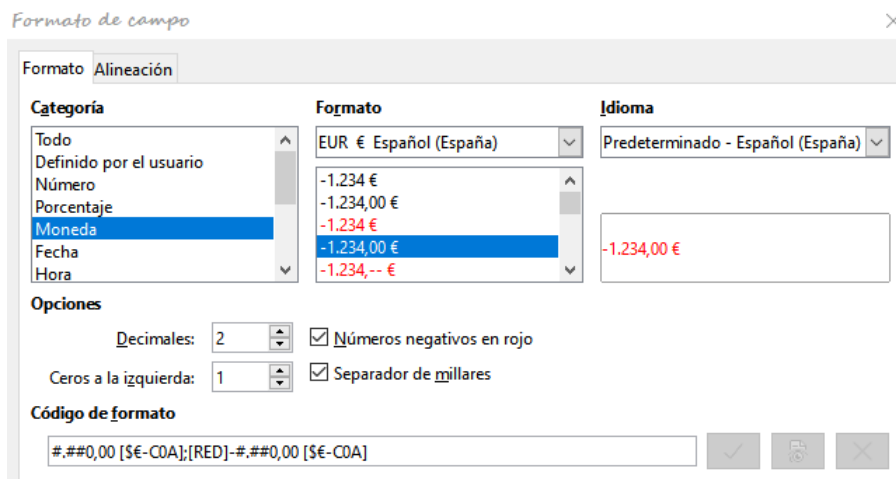
El formateo solo proporciona una representación de los contenidos. Una fecha representada por un número de año de dos caracteres se almacenará como un año de cuatro caracteres. Si se crea un campo para un número con dos decimales, como el cargo vencido (*OverdraftCharge* en el siguiente ejemplo), el número se almacena con dos decimales, incluso si no se ha configurado el formato para mostrarlos. Se puede ingresar un número con dos decimales en un campo formateado sin decimales. La parte decimal parece desaparecer en la entrada pero se vuelve visible si se omite el formato.

Los formularios se pueden configurar para mostrar solo la información necesaria, para mostrar solo la hora, de un campo Fecha/Hora [TIMESTAMP]. En el caso de almacenar tiempo desde un cronómetro, por ejemplo, los minutos, segundos y fracciones de segundo en un campo de marca de tiempo se pueden mostrar usando MM:SS.00 en el formato de visualización. Un formato sin fecha se puede establecer más adelante en formularios utilizando el campo formateado, pero no directamente en el campo.

El formato de los campos cuando se crea la tabla o posteriormente, a través de las propiedades del campo, utiliza un diálogo separado:



En *Propiedades del campo*, el botón junto a *Ejemplo de formato* abre el cuadro de diálogo para cambiar el formato.



Al crear campos de moneda, tenga cuidado de que el campo numérico tenga dos lugares decimales establecidos. El formato se puede realizar al crear la tabla en la interfaz para usar la moneda adecuada durante la entrada. Esto solo afecta la entrada en la tabla y en las consultas que usan el valor de entrada sin cálculos. En los formularios, la designación de moneda debe formatearse en las propiedades del control.



## Nota

Base guarda el formato de las tablas cuando se crean los campos o durante la entrada de datos si los formatos de columna se modifican haciendo clic con el botón derecho en los encabezados de las columnas. Los anchos de columna en la pantalla de entrada también se guardan cuando se modifican durante la entrada de datos.

En consultas, formularios o informes, el formato de visualización se puede modificar según sea necesario.

En el caso de los campos que deben contener un porcentaje, tenga en cuenta que el resultado de 1% debe almacenarse como 0.01. Por lo tanto, escribir porcentajes requiere al menos dos lugares decimales. Si es necesario almacenar porcentajes fraccionarios como 3,45, el valor numérico almacenado requiere cuatro decimales (0.0345).

## Crear un índice

A veces es útil indexar otros campos o una combinación de otros campos además de la clave primaria. Un índice acelera la búsqueda y también se puede utilizar para evitar entradas duplicadas.

Cada índice tiene un orden de clasificación definido. Si se muestra una tabla sin ordenar, el orden de clasificación estará de acuerdo con el contenido de los campos especificados en el índice.

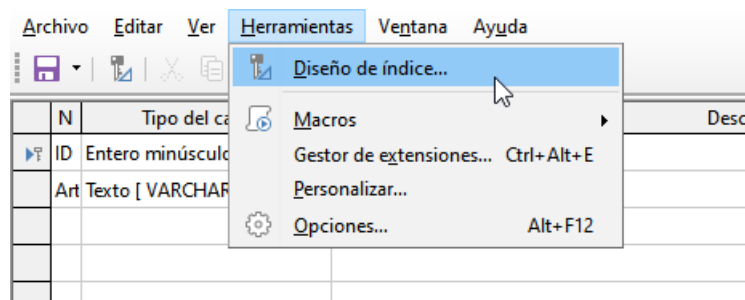


Figure 7: Acceso al Diseño de índice

Abra la tabla para editar haciendo clic derecho y usando el menú contextual. Luego puede acceder a la creación de índice con **Herramientas > Diseño de índice**.

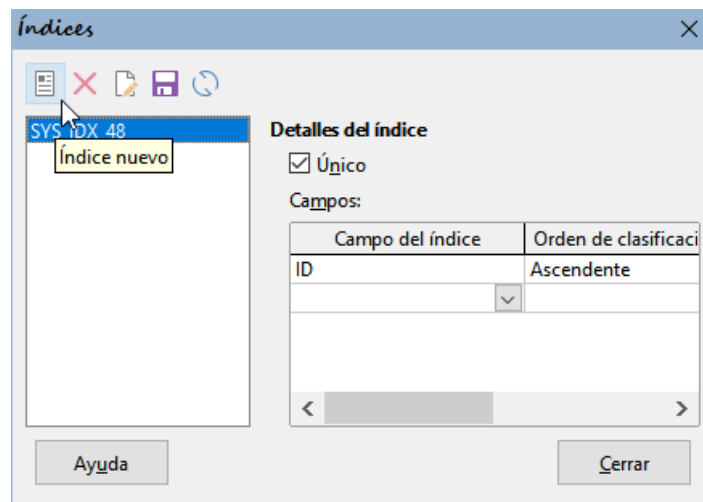


Figura 8: Crear un nuevo índice

En el cuadro de diálogo Índices (Figura 8) haga clic en el botón *Índice nuevo* para crear un índice, además de la clave principal.

El nuevo índice recibe automáticamente el nombre *index1*. El Índice especifica qué campo o campos se utilizarán para este índice. Al mismo tiempo, puede elegir el orden de clasificación.

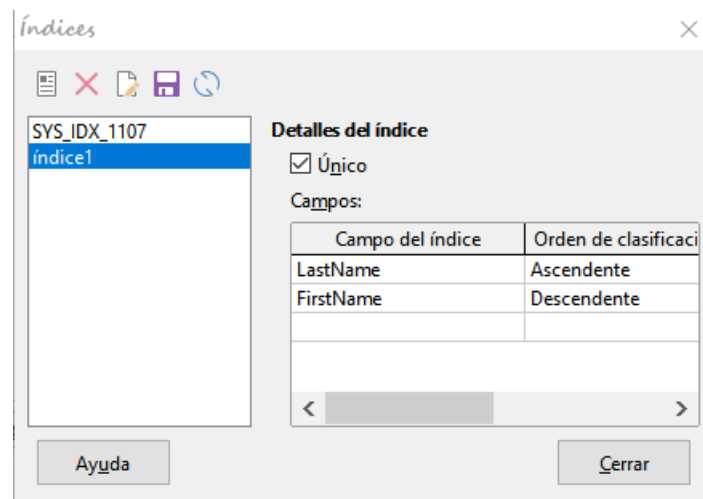


Figura 9: El índice se define como único

En principio, también se puede crear un índice a partir de campos de tabla que no contienen valores únicos. Sin embargo, en la Figura 9, en Detalles de índice se ha marcado Único, de modo que el campo *LastName* junto con el campo *FirstName* solo puede tener entradas que aún no se encuentren en esa combinación. Por ejemplo, Robert Miller y Robert Maier son posibles, y también Robert Miller y Eva Miller. Pero nunca dos Robert Miller.

Si se crea un índice solo para un campo, la unicidad se aplica a ese campo. Tal índice suele ser la clave principal. En este campo, cada valor puede aparecer solo una vez. Además, en el caso de claves primarias, el campo nunca puede ser NULL.

Una circunstancia excepcional para un índice único es cuando no hay entrada en un campo (el campo es NULL). Como NULL puede tener cualquier valor arbitrario, un índice que usa dos campos siempre puede tener la misma entrada repetidamente en uno de los campos siempre que no haya entrada en el otro.



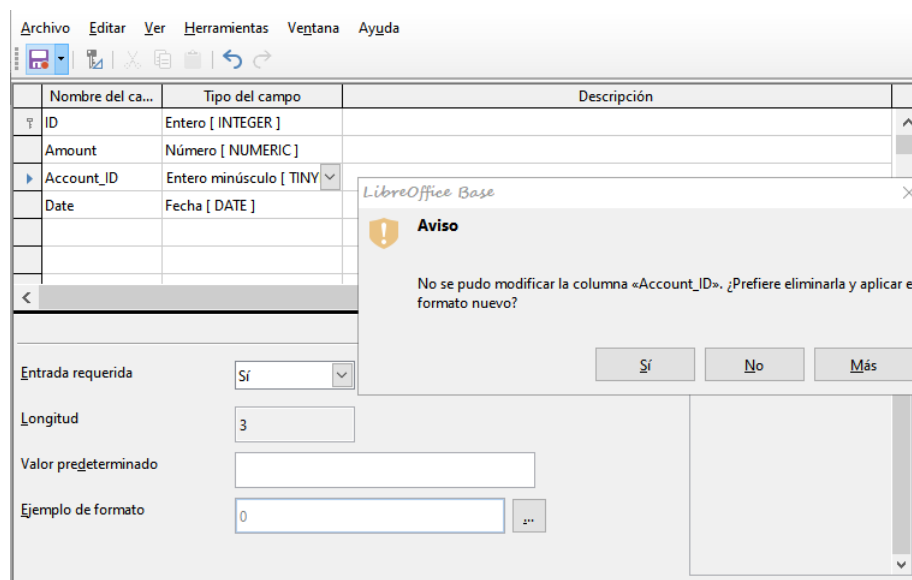
## Nota

**NULL** se usa en bases de datos para designar una celda vacía, que no contiene nada. No es posible ningún cálculo usando un campo NULL. Esto contrasta con las hojas de cálculo, en las que a las celdas vacías se les asigna automáticamente el valor 0 (cero).

Ejemplo: en una base de datos, el número de artículos y la fecha del préstamo se ingresan cuando se presta el artículo. Cuando se devuelve, se ingresa una fecha de devolución. En teoría, un índice que utiliza los campos *Media\_ID* y *ReturnDate* podría evitar fácilmente que se preste el mismo elemento repetidamente sin que se indique la fecha de devolución. Lamentablemente, esto no funcionará porque la fecha de devolución inicialmente no tiene ningún valor. El índice evitará que un artículo se marque como devuelto dos veces con la misma fecha, pero no hará nada más.

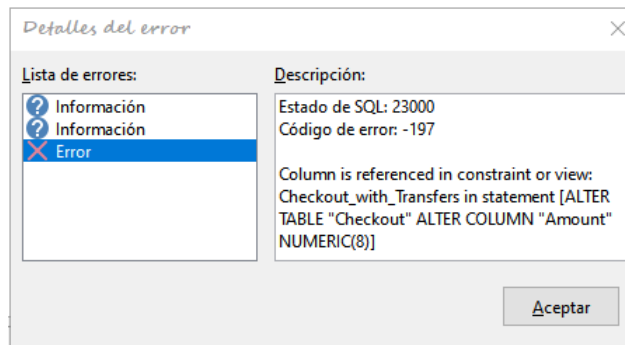
## Problemas al modificar tablas

Es mejor crear tablas completas con todas las configuraciones necesarias, antes de establecer relaciones o crear consultas, de modo que los cambios en la configuración de la tabla no sean necesarios más adelante. Cuando las propiedades de los campos (nombre de campo, entrada obligatoria, etc.) se cambian más tarde, puede generar mensajes de error que no se deben a la interfaz, sino a un intento de modificar la base de datos subyacente de una manera no deseada.

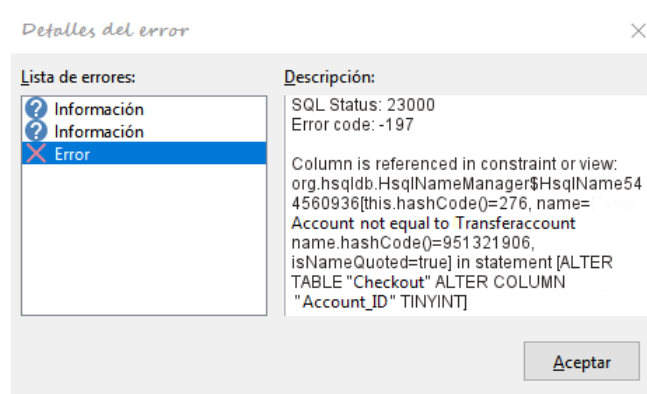


En este caso, el campo *Amount* se intenta cambiar a una *Entrada requerida* = Sí. El símbolo de advertencia nos notifica que este cambio puede conducir a la pérdida de datos. No es posible un cambio simple porque puede haber registros que ya tengan una entrada en este campo.

Al hacer clic en *Sí* aparece otro aviso de error, ya que la estructura de la base de datos no permite que se elimine este campo. Al hacer clic en *No*, se cancela toda la operación. El botón *Más* se proporciona siempre que sea posible para brindarle información adicional sobre la resolución del problema.



La columna con el nombre de campo *Amount* se menciona en otra parte de la base de datos. Esto podría ser una definición restrictiva o una vista de tabla que fue creada por algún usuario después de que se creó la tabla misma. La ilustración anterior muestra que el nombre de la restricción o vista es *View\_Checkout\_with\_Transfers*. Esto deja claro dónde deben realizarse cambios en la base de datos. Por ejemplo, el código SQL para la vista podría guardarse primero como una consulta, y luego la vista podría destruirse y hacer un nuevo intento para modificar el campo.



En este caso, el nombre de la restricción *Account not equal to Transferaccount* nos lleva hasta la definición de esa restricción. La condición es que el valor en el campo *Account\_ID* no puede ser el mismo que el valor en el campo *TransferAccount\_ID*. La columna solo puede modificarse si se elimina esta condición.

Ahora, si se produce un error adicional, lo más probable es que sea causado por el campo correspondiente que está vinculado a un campo en otra tabla por una relación definida. En este caso, el enlace debe romperse utilizando *Herramientas > Relaciones* antes de que se pueda realizar el cambio.

### Limitaciones del diseño de tablas gráficas.

La secuencia de campos en una tabla no se puede cambiar una vez que se ha guardado la base de datos. Para mostrar una secuencia diferente se necesita una consulta.

Solo mediante órdenes SQL directas puede insertar un campo en una posición específica en la tabla. Sin embargo, los campos ya creados no se pueden mover con este método.

Las propiedades de las tablas deben establecerse al principio: por ejemplo, qué campos no deben ser estar vacíos (NULL) y cuáles deben contener un valor predeterminado. Estas propiedades no se pueden cambiar posteriormente con la interfaz.

Los valores predeterminados que puede establecer en la interfaz no son tan potentes como los posibles valores predeterminados dentro de la base de datos. Por ejemplo, no puede definir el valor predeterminado para un campo de fecha como la fecha de entrada. Eso solo es posible con órdenes SQL directas.

## Entrada directa de órdenes SQL

Para ingresar órdenes SQL directas, vaya a **Herramientas > SQL**.

Las órdenes se ingresan en el área superior de la ventana (Figura 10). En el área de *Estado*, se muestra el éxito o la razón del fracaso. Los resultados de las instrucciones SELECT se pueden mostrar en el área *Salida* si la casilla de verificación está seleccionada.

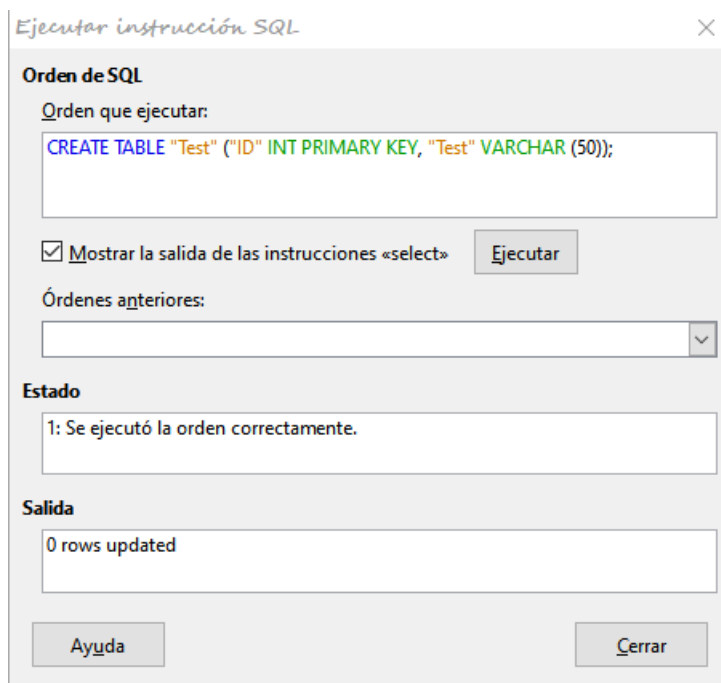


Figura 10: Entrada directa de órdenes SQL

Puede encontrar un resumen de las posibles órdenes para el motor HSQLDB incorporado en <http://www.hsqldb.org/doc/1.8/guide/ch09.html>. Los contenidos se describen en las siguientes secciones. Algunas órdenes solo tienen sentido cuando se trata de una base de datos HSQLDB externa (Especificar usuario, etc.). Cuando las necesite, puede consultar la sección «Conectar una base de datos a un HSQLDB externo» en el *Apéndice A* de esta guía.



### Nota

LibreOffice se basa en la versión 1.8.0 de HSQLDB. La versión del servidor actualmente disponible es 2.5. Las funciones de la nueva versión son más amplias. Se pueden encontrar en <http://hsqldb.org/web/hsqldbDocsFrame.html>. La Versión 1.8 se describe en <http://www.hsqldb.org/doc/1.8/guide/>. Una descripción adicional en los paquetes de instalación para HSQLDB, se pueden descargar desde <http://sourceforge.net/projects/hsqldb/files/hsqldb/>.

### Crear tabla

Una orden simple para crear una tabla utilizable es:

```
CREATE TABLE "Test" ("ID" INT PRIMARY KEY, "Texto" VARCHAR(50));
```

Desglose de esta orden:

```
CREATE TABLE "Test": Crear una tabla de nombre "Test".
```

Los nombres de campo, tipos de campo y opciones especificados se insertan entre paréntesis ( ): ("ID" INT PRIMARY KEY, "Text" VARCHAR(50)): campo de nombre ID de tipo numérico entero como clave principal; campo de nombre Texto de tipo texto variable y la longitud limitada a 50 caracteres.

Parámetros de la orden CREATE:

```
CREATE [MEMORY | CACHED | [GLOBAL] TEMPORARY | TEMP | TEXT] TABLE
"Nombre_Tabla" ( <Definiciones_Campos> [, ...] [, <Restricciones_Campos>...] ) [ON
COMMIT {DELETE | PRESERVE} ROWS];
```

#### **[MEMORY | CACHED | [GLOBAL] TEMPORARY | TEMP | TEXT]:**

Especifica la ubicación de la tabla recién creada. La configuración predeterminada es MEMORY. HSQLDB crea todas las tablas en la memoria central. Esta configuración también se aplica a las tablas que LibreOffice Base escribe en la base de datos incrustada. Otra posibilidad sería escribir las tablas en el disco duro y usar la memoria solo para almacenar el acceso al disco duro (CACHED).



#### **Nota**

```
CREATE TEXT TABLE "Test" ("ID" INT PRIMARY KEY, "Text" VARCHAR(50));
```

Crear una tabla test en HSQLDB. Tiene que estar vinculada a un archivo de texto externo (por ejemplo, un archivo \*.csv): SET TABLE "Text" SOURCE "Text.csv";

Naturalmente, el archivo Text.csv debe tener los campos correspondientes en el orden correcto. Al crear el enlace, se pueden seleccionar varias opciones adicionales. Para más detalles, consulte

[http://www.hsqldb.org/doc/1.8/guide/guide.html#set\\_table\\_source-section](http://www.hsqldb.org/doc/1.8/guide/guide.html#set_table_source-section)

Las tablas de texto no están protegidas de escritura, contra otros programas. Por lo tanto, puede suceder que otro programa o usuario altere la tabla justo cuando Base accede a ella. Las tablas de texto se utilizan principalmente para el intercambio de datos entre diferentes programas.

Las tablas en formato texto (TEXT) (como CSV) no se pueden escribir en bases de datos internas que se configuran únicamente en MEMORIA, mientras que Base no puede acceder a las tablas TEMPORALES o TEMP. Las órdenes SQL se ejecutan en este caso, pero las tablas no se muestran (y, por lo tanto, no se pueden eliminar) utilizando la interfaz, y los datos ingresados a través de SQL tampoco son visibles para el módulo de consulta de la interfaz, a menos que se elimine automáticamente el contenido después de que se evita la confirmación final (con ON COMMIT PRESERVE ROWS). Cualquier solicitud en este caso mostrará una tabla sin ningún contenido.

Las tablas creadas directamente con SQL no se muestran inmediatamente. Debe usar **Ver > Actualizar tablas** o simplemente cerrar la base de datos y luego volver a abrirla.

#### **<Definiciones\_Campos>:**

```
"Nombre_Campo" Tipo de Datos[(Número de caracteres[,Lugares decimales])] [{DEFAULT
"Valor_Predeterminado" | GENERATED BY DEFAULT AS IDENTITY (START WITH <n>[,
INCREMENT BY <m>]}] | [[NOT] NULL] [IDENTITY] [PRIMARY KEY]
```

Permite que los valores predeterminados se incluyan en la definición del campo.

Para los campos de texto, puede ingresar texto entre comillas simples o NULL. La única función SQL permitida es CURRENT\_USER. Esto solo tiene sentido si HSQLDB se está utilizando como una base de datos de servidor externa con varios usuarios.

Para los campos de fecha y hora, se puede ingresar una fecha, una hora o una combinación de las dos entre comillas simples o NULL. Debe asegurarse de que la fecha sigue las convenciones americanas (aaaa-mm-dd), que la hora tiene el formato hh:mm:ss, y que un valor combinado de fecha/hora [TIMESTAMP] tiene el formato aaaa-mm-dd hh:mm:ss .

Funciones SQL permitidas:

para la fecha actual `CURRENT_DATE, TODAY, CURDATE ()`

para la hora actual `CURRENT_TIME, NOW, CURTIME ()`

para la marca de tiempo de datos actual `CURRENT_TIMESTAMP, NOW.`

Para los campos booleanos (sí / no) se pueden ingresar las expresiones `FALSE, TRUE, NULL.` Tienen que ingresarse sin comillas simples.

Para los campos numéricos, es posible cualquier número válido en el rango o NULL. Aquí también, si ingresa NULL, tampoco use comillas. Al ingresar decimales, asegúrese de que el punto decimal sea un punto y no una coma. (Algunas personas de habla inglesa usan una coma como separador decimal).

Para campos binarios (imágenes, etc.) es posible cualquier cadena hexadecimal válida entre comillas simples o NULL. Una cadena de ejemplo hexadecimal es: 0004ff, que representa 3 bytes: primero 0, luego 4 y finalmente 255 (0xff). Como los campos binarios en la práctica solo necesitan ingresarse para las imágenes, debe conocer el código binario de la imagen que se utilizará como predeterminado.



## Nota

Sistema hexadecimal: los números tienen 16 como base de numeración. Un sistema mixto que consiste en los números del 0 al 9 y las letras de la **a** a la **f** proporciona 16 dígitos posibles para cada columna. Con dos columnas, puede tener  $16 \times 16 = 256$  valores posibles. Esto corresponde a 1 Byte ( $2^8$ ).

**NOT NULL:** El valor del campo no puede ser NULL. Esta condición solo puede darse en la definición de campo.

Ejemplo:

```
CREATE TABLE "Test" ("ID" INT GENERATED BY DEFAULT AS IDENTITY (START WITH 10),  
"Name" VARCHAR(50) NOT NULL, "Date" DATE DEFAULT TODAY);
```

Se crea una tabla llamada *Test*. El campo *ID*, clave primaria se define como *Valor automático*, cuyo valor comienza en 10. El campo *Name* es un campo de texto con longitud máxima de 50 caracteres. No debe estar vacío. Finalmente tenemos el campo de fecha *Date* que almacena la fecha actual, si no se ingresa ninguna otra fecha. Este valor predeterminado solo se hace efectivo cuando se crea un nuevo registro. Eliminar una fecha en un registro existente deja el campo vacío.

### <Restricciones\_Campos>:

```
[CONSTRAINT "Nombre_Restricción"  
UNIQUE ( "Nombre_Campo1" [,"Nombre_Campo2"...] ) |  
PRIMARY KEY ( "Nombre_Campo1" [,"Nombre_Campo2"...] ) |  
FOREIGN KEY ( "Nombre_Campo1" [,"Nombre_Campo2"...] )  
REFERENCES "otro_Nombre_Tabla" ( "Nombre_Campo1" [,"Nombre_Campo2"...])  
[ON {DELETE | UPDATE}  
{CASCADE | SET DEFAULT | SET NULL}] |  
CHECK(<Condición_Búsqueda>)
```



**CONSTRAINT** define las condiciones que deben cumplirse cuando se ingresan los datos. Las restricciones pueden recibir un nombre. **UNIQUE** ("Nombre\_Campo"): el valor del campo debe ser único dentro de ese campo

**PRIMARY KEY** ("Nombre\_Campo"): el valor del campo debe ser único y no puede ser NULL (clave primaria)

**FOREIGN KEY** ("Nombre\_Campo") **REFERENCES** <"otro\_Nombre\_Tabla"> "Nombre\_Campo"): Los campos especificados en esta tabla están vinculados a los campos de otra tabla. El valor del campo debe ser probado para integridad referencial como claves foráneas; es decir, debe haber una clave primaria correspondiente en la otra tabla, si se ingresa un valor aquí.

[**ON {DELETE | UPDATE} {CASCADE | SET DEFAULT | SET NULL}**]: En el caso de una clave externa, esto especifica qué sucederá si, por ejemplo, se elimina el registro externo. No tiene sentido, en una tabla de préstamos para una biblioteca, tener un número de usuario para el cual el usuario ya no existe. El registro correspondiente debe modificarse para que la relación entre las tablas siga siendo válida. Por lo general, el registro simplemente se elimina. Esto sucede si selecciona **ON DELETE CASCADE**.

**CHECK**(<Condición\_Búsqueda>): Formulado como una condición **WHERE**, pero solo para el registro actual.

```
CREATE TABLE "Time_measurement" ("ID" INT PRIMARY KEY, "Start_time" TIME, "End_time" TIME, CHECK ("Start_time" <= "End_time"));
```

La condición **CHECK** excluye la entrada de un valor de *End\_time* a la hora *Start\_time*. Un intento de hacer esto produce un mensaje de error similar a:

```
Check constraint violation SYS_CT_357 table: Time_measurement ...
```

A la restricción de búsqueda se le asigna un nombre que no es muy informativo. En lugar de eso, el nombre podría definirse en la definición de la tabla:

```
CREATE TABLE "Time_measurement" ("ID" INT PRIMARY KEY, "Start_time" TIME, "End_time" TIME, CONSTRAINT "Start_time<=End_time" CHECK ("Start_time" <= "End_time"));
```

Esto da un mensaje de error algo más claro en el que aparece el nombre de la restricción involucrada.

Se deben respetar las restricciones al establecer relaciones entre tablas o la indexación para campos particulares. Las restricciones se establecen usando la condición «**CHECK**», en la interfaz usando **Herramientas > Relaciones**, y también en con los índices creados al editar la Tabla ya diseñada con **Herramientas > Diseño de índices**.

[**ON COMMIT {DELETE | PRESERVE} ROWS**]:

El contenido de las tablas del tipo **TEMPORARY** o **TEMP** se borra de manera predeterminada cuando ha terminado de trabajar con un registro en particular (**ON COMMIT DELETE ROWS**). Esto le permite crear registros temporales, que contienen información para otras acciones que se llevarán a cabo al mismo tiempo.

Si desea que una tabla de este tipo contenga datos disponibles para una sesión completa (desde abrir una base de datos hasta cerrarla), elija **ON COMMIT PRESERVE ROWS**.

## Modificar la tabla

A veces es posible que desee insertar un campo adicional en una posición particular de la tabla. Supongamos que tiene una tabla llamada *Direcciones* con campos *ID*, *Apellido*, *Calle*, etc. Se observa que se nos ha olvidado incluir el campo *Nombre*.

```
ALTER TABLE "Direcciones" ADD "Nombre" VARCHAR(25) BEFORE "Apellido";
```

`ALTER TABLE "Direcciones"`: Cambiar la tabla *Direcciones*.

`ADD "Nombre" VARCHAR(25)`: Insertar el campo *Nombre* con una longitud de 25 caracteres.

`BEFORE "Apellido"`: Antes del campo *Apellido*.

La posibilidad de especificar la posición de los campos adicionales después de la creación de la tabla no está disponible en la interfaz.

```
ALTER TABLE "Nombre_Tabla" ADD [COLUMN] <Definición_Campo> [BEFORE  
"Nombre_Campo_ya_existente"];
```

La designación adicional `COLUMN` no es necesaria en casos donde no hay opciones alternativas disponibles.

```
ALTER TABLE "Nombre_Tabla" DROP [COLUMN] "Nombre_Campo";
```

El campo *Nombre\_Campo* se borra de la tabla *Nombre\_Tabla*. Sin embargo, esto no ocurre si el campo está involucrado en una vista o como una clave externa en otra tabla.

```
ALTER TABLE "Nombre_Tabla" ALTER COLUMN "Nombre_Campo" RENAME TO  
"Nuevo_Nombre_Campo";
```

Cambia el nombre de un campo.

```
ALTER TABLE "Nombre_Tabla" ALTER COLUMN "Nombre_Campo" SET DEFAULT  
<Valor_Predeterminado>;
```

Establece un valor predeterminado específico para el campo. `NULL` elimina un valor predeterminado existente.

```
ALTER TABLE "Nombre_Tabla" ALTER COLUMN "Nombre_Campo" SET [NOT] NULL;
```

Establece o elimina una condición `NOT NULL` para un campo.

```
ALTER TABLE "Nombre_Tabla" ALTER COLUMN <Definición_Campo>;
```

La definición del campo corresponde a Crear tabla con las siguientes restricciones:

- El campo tiene que ser ya un campo de clave principal para aceptar la propiedad `IDENTITY`. `IDENTITY` significa que el campo tiene la propiedad Valor automático. Esto solo es posible para los campos `INTEGER` o `BIGINT`. Para estas descripciones de tipo de campo, consulte el Apéndice de este manual.
- Si el campo ya tiene la propiedad `IDENTITY` pero no se repite en la definición del campo, se elimina la propiedad `IDENTITY` existente.
- El valor predeterminado será el especificado en la nueva definición de campo. Si la definición del valor predeterminado se deja en blanco, se elimina cualquier valor predeterminado ya definido.
- La propiedad `NOT NULL` continúa en la nueva definición, si no se define de otra manera. Esto está en contraste con el valor predeterminado.
- En algunos casos, según el tipo de modificación, la tabla debe estar vacía para que se produzca el cambio. En todos los casos, el cambio tendrá efecto solo si es posible en principio (por ejemplo, un cambio de `NOT NULL` a `NULL`) y los valores existentes se pueden traducir (por ejemplo, un cambio de `TINYINT` a `INTEGER`).

```
ALTER TABLE "Nombre_Tabla" ALTER COLUMN "Nombre_Campo" RESTART WITH  
<Nuevo_Valor_Campo>;
```

Esta orden se usa exclusivamente para un campo **IDENTITY**. Determina el siguiente valor para un campo con el conjunto de funciones Valor automático. Se puede usar, por ejemplo, cuando una base de datos se usa inicialmente con datos de prueba y luego se proporciona con datos reales. Esto requiere que se elimine el contenido de las tablas y que se establezca un nuevo valor como "1" para el campo.

```
ALTER TABLE "Nombre_Tabla" ADD [CONSTRAINT "Nombre_Condición"] CHECK (<Condición_Búsqueda>);
```

Esto agrega una condición de búsqueda introducida por la palabra CHECK. Tal condición no se aplicará retrospectivamente a los registros existentes, pero se aplicará a todos los cambios posteriores y los registros ingresados recientemente. Si no se define un nombre de restricción, se asignará uno automáticamente. Ejemplo:

```
ALTER TABLE "Loan" ADD CHECK (IFNULL("Return_Date","Loan_Date")>="Loan_Date");
```

La tabla Loan necesita protegerse de los errores de entrada. Por ejemplo, debe evitar que se proporcione una fecha de devolución anterior a la fecha del préstamo. Si este error ocurre durante el proceso de devolución, recibirá un mensaje de error **Check constraint violation...**

```
ALTER TABLE "Nombre_Tabla" ADD [CONSTRAINT "Nombre_Restricción"] UNIQUE ("Nombre_Campo1", "Nombre_Campo2"...);
```

Aquí se agrega una condición que obliga a los campos nombrados a tener valores diferentes en cada registro. Si se nombran varios campos, esta condición se aplica a la combinación en lugar de los campos individuales. NULL no cuenta aquí. Por lo tanto, un campo puede tener el mismo valor repetidamente sin causar ningún problema, si el otro campo en cada uno de los registros es NULL. Esta orden no funcionará si ya existe una condición ÚNICA para la misma combinación de campo.

```
ALTER TABLE "Nombre_Tabla" ADD [CONSTRAINT "Nombre_Restricción"] PRIMARY KEY ("Nombre_Campo1", "Nombre_Campo2"...);
```

Agrega una clave primaria, opcionalmente con una restricción, a una tabla. La sintaxis de la restricción es la misma que cuando se crea una tabla.

```
ALTER TABLE "Nombre_Tabla" ADD [CONSTRAINT "Nombre_Restricción"] FOREIGN KEY ("Nombre_Campo1", "Nombre_Campo2"...)  
REFERENCES "Nombre_de_otra_Tabla" ("Nombre_Campo1_otra_Tabla", "Nombre_Campo2_otra_Tabla"...)  
[ON {DELETE | UPDATE} {CASCADE | SET DEFAULT | SET NULL}];
```

Agrega una clave externa (**FOREIGN KEY**) a la tabla. La sintaxis es la misma que cuando se crea una tabla. La operación finalizará con un mensaje de error si algún valor en la tabla no tiene un valor correspondiente en la tabla que contiene esa clave primaria.

Ejemplo: las tablas *Nombre* y *Direcciones* deben vincularse. La tabla *Nombre* contiene un campo con el nombre *Direcciones\_ID*. El valor de este debe estar vinculado al campo *ID* en la tabla *Direcciones*. Si el valor 1 se encuentra en *Direcciones\_ID* pero no en el campo *ID* de la tabla *Direcciones*, el enlace no funcionará. Tampoco funcionará si los dos campos son de diferentes tipos.

**ALTER TABLE "Nombre\_Tabla" DROP CONSTRAINT "Nombre\_Restricción";** Esta orden elimina la restricción nombrada (**UNIQUE**, **CHECK**, **FOREIGN KEY**) de una tabla.

**ALTER TABLE "Nombre\_Tabla" RENAME TO "Nuevo\_Nombre\_Tabla";** Finalmente, esta orden cambia solo el nombre de una tabla.



## Nota

Cuando cambia una tabla usando SQL, el cambio afecta la base de datos pero no es necesariamente aparente o efectivo en la interfaz. Cuando la base de datos se cierra y se vuelve a abrir, los cambios también aparecen en la interfaz.

Los cambios también se muestran si elige **Ver > Actualizar tablas** en el contenedor de la tabla.

---

## Eliminar tablas

```
DROP TABLE "Nombre_Tabla" [IF EXISTS] [RESTRICT | CASCADE];
```

Elimina la tabla *Nombre\_Tabla*.

**IF EXISTS** evita que ocurra un error si esta tabla no existe. **RESTRICT** es la disposición predeterminada y no necesita ser elegida explícitamente; significa que la eliminación no se produce si la tabla está vinculada a otra tabla mediante el uso de una clave externa o si hay una vista activa de esta tabla. Las consultas no se ven afectadas, ya que no se almacenan en HSQLDB.

Si, en cambio, elige **CASCADE**, se eliminan todos los enlaces a la tabla *Nombre\_Tabla*. En las tablas vinculadas, todas las claves externas se establecen en **NULL**. Todas las vistas que se refieren a la tabla nombrada también se eliminan por completo.

## Vincular tablas

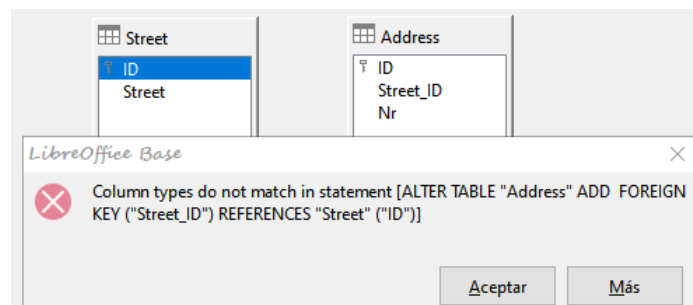
---

En principio, puede tener una base de datos sin enlaces entre tablas. El usuario tiene que asegurarse durante la entrada de datos, que las relaciones entre las tablas permanecen correctas. Esto generalmente se hace mediante el uso de formularios de entrada adecuados que lo manejan.

Eliminar registros en tablas vinculadas no es una cuestión simple. Suponga que desea eliminar una calle en particular de la tabla *Street* en la Figura 6, donde el campo *ID* está vinculado con la tabla *Address* como clave externa en esa tabla. Las referencias en la tabla *Address* desaparecerían. La base de datos no permite esto, una vez que se ha creado la relación. Para eliminar la calle, se debe cumplir la condición previa, que ya no se haga referencia en la tabla de direcciones.

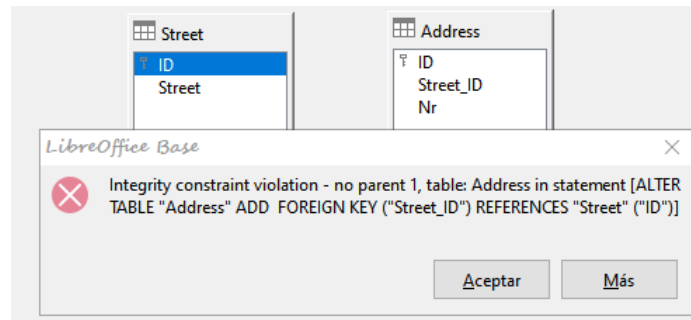
Los enlaces básicos se hacen usando **Herramientas > Relaciones**. Esto crea una línea de conexión desde la clave primaria en una tabla a la clave externa definida en la otra.

Puede recibir el siguiente mensaje de error al crear dicho enlace:

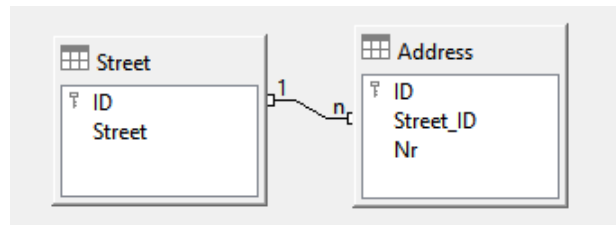


Este mensaje muestra el error que ocurrió y la orden interna de SQL que causó el error: *Column types do not match in statement*— Como también se muestra la orden SQL, la referencia es claramente a las columnas *Address.street\_ID* y *Street.ID*. Para fines de prueba, uno de estos

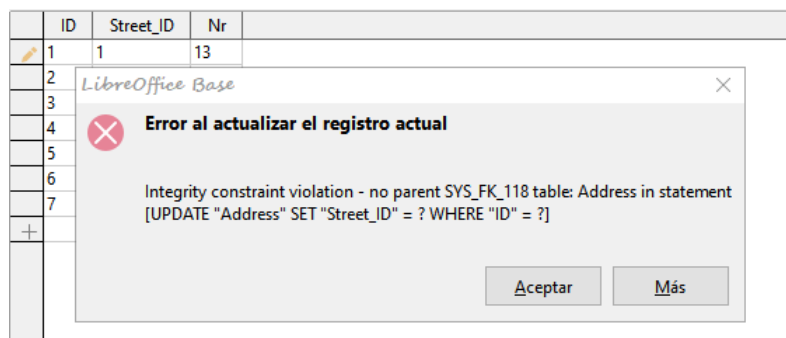
campos se definió como un Entero, el otro como Entero minúsculo. Por lo tanto, no se pudo crear ningún enlace, ya que un campo no puede ser de distinto tipo que el otro.



En este caso, los tipos de columna coinciden. La instrucción SQL es la misma que en el primer ejemplo. Pero de nuevo hay un error: **Integrity constraint violation – no parent 1, table: Address** — No se pudo establecer la integridad de la relación. En el campo *Street\_ID* de la tabla *Address*, hay un número **1**, que no está presente en el campo *ID* de la tabla *Street*. La tabla principal aquí es *Street*, ya que su clave principal es la que debe existir. Este error es muy común, cuando se van a vincular dos tablas y algunos campos de la tabla con la clave externa prospectiva ya contienen datos. Si el campo de clave externa contiene una entrada que no está presente en la tabla principal (la tabla que contiene la clave primaria), esta es una entrada no válida.



Si la vinculación se lleva a cabo con éxito y posteriormente se intenta introducir un registro igualmente inválido en la tabla, aparece el siguiente mensaje de error:



De nuevo esto es una violación de integridad. Base se niega a aceptar el valor 1 para el campo *street\_ID* después de que se haya realizado el enlace porque la tabla *Street* no contiene dicho valor en el campo *ID*.

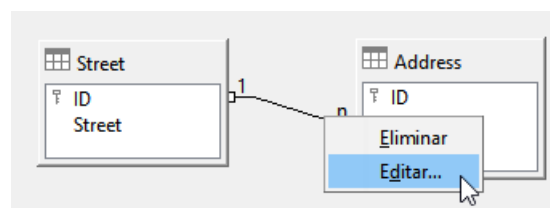


Figura 11: Los enlaces se editan con clic derecho

Las propiedades de un enlace se pueden editar para que la eliminación de un registro de la tabla *Street*, establezca simultáneamente en NULL las entradas correspondientes en la tabla *Address*.

Las propiedades que se muestran en la Figura 11 siempre se relacionan con una acción vinculada al cambio en un registro de la tabla que contiene la clave primaria correspondiente. En nuestro caso esta es la tabla *Street*.

Si se modifica la clave principal de un registro en esta tabla (*Opciones de actualización*), podrían ocurrir las siguientes acciones:

### Sin acción

Cambiar la clave primaria *Street.ID* no está permitido en este caso, ya que rompería la relación entre las tablas.

### Actualizar en cascada

Si se cambia la clave principal *Street.ID*, la clave externa se cambia automáticamente a su nuevo valor. Esto asegura que el enlace no esté dañado. Por ejemplo, si un valor se cambia de 3 a 4, todos los registros de la tabla *Address* que contienen la clave externa *Address.Street\_ID* con el valor 3, se cambian a 4.

### Definir NULL

Todos los registros que contienen esta clave primaria en particular ahora no tendrán entrada en el campo de clave externa *Address.Street\_ID*; El campo será NULL.

Relaciones

Tablas involucradas

Address Street

Campos involucrados

Address	Street
Street_ID	ID

Opciones de actualización

Ninguna acción

Actualización en cascada

Definir NULL

Predefinir

Opciones de eliminación

Sin acción

Eliminar en cascada

Definir NULL

Definir predeterminado

Ayuda Aceptar Cancelar

Figura 12: Editar las propiedades de la relación

### Definir predeterminado

Si se cambia la clave principal *Street\_ID*, el valor de *Address.Street\_ID* originalmente vinculado a ella se establece en el valor predeterminado previamente definido. Para este propósito, necesitamos una definición inequívoca de un valor predeterminado. Si el valor predeterminado se establece utilizando la instrucción SQL:

```
ALTER TABLE "Address" ALTER COLUMN "Street_ID" SET DEFAULT 1;
```

la definición del enlace asegura que el campo volverá a este valor en el caso de una Actualización. Por lo tanto, si se cambia la clave principal en la tabla *Street*, la clave externa correspondiente en la tabla *Address* se establecerá en 1. Esto es útil cuando se requiere un registro para tener un campo *street*, en otras palabras, este campo no puede ser NULL. Pero

tenga cuidado: si 1 no está en uso, habrá creado un enlace a un valor inexistente. Por lo tanto, es posible destruir la integridad de la relación.



## Precaución

Si el valor predeterminado en un campo de clave externa no está vinculado a un valor de clave primaria de la tabla externa, se creará un enlace a un valor que no es posible. La integridad referencial de la base de datos sería destruida.

---

Sería mejor **no utilizar** la posibilidad de establecer el valor predeterminado.

Si se elimina un registro de la tabla *Street* (*Opciones de eliminación*), las siguientes opciones están disponibles:

### Sin acción

No se lleva a cabo ninguna acción. Si la eliminación solicitada afecta un registro en la tabla *Address*, la solicitud será rechazada.

### Eliminar en cascada

Si se elimina un registro de la tabla *Street* y esto afecta a un registro en la tabla *Address*, ese registro también se eliminará.

Puede parecer extraño en este contexto, pero hay otras estructuras de tabla en las que tiene sentido. Supongamos que tiene una tabla *CD* y una tabla que almacena los títulos en estos CD. Ahora, si se elimina un registro en la tabla *CD*, muchos títulos en la otra tabla no tienen sentido ya que ya no están disponibles para usted. En tales casos, una eliminación en cascada tiene sentido. Significa que no necesita eliminar todos los títulos antes de eliminar el CD de la base de datos.

### Definir NULL

Esto es lo mismo que para la opción de actualización.

### Definir predeterminado

Esto es lo mismo que para la opción de actualización y requiere las mismas precauciones.



## Consejo

La opción *Sin acción* debe evitarse en la mayoría de los casos para que no se muestren mensajes de error de la base de datos al usuario, ya que estos no son fácilmente comprensibles.

---

Las relaciones a claves foráneas que se refieren a un solo campo de otra tabla se establecen arrastrando con el ratón un campo seleccionado hacia el otro campo de la otra tabla.

Para vincular a una tabla que tiene una clave principal compuesta, vaya a **Herramientas > Relaciones**, luego **Insertar > Nueva relación**, o use el botón correspondiente. Aparece un cuadro de diálogo para editar las propiedades de una relación con la posibilidad de elegir las tablas disponibles.

## Introducir datos en tablas

---

Las bases de datos que consisten en una sola tabla generalmente no requieren un formulario de entrada a menos que contengan un campo para imágenes. Sin embargo, cuando una tabla contiene claves externas de otras tablas, los usuarios deben recordar qué números de clave ingresar o mirar las otras tablas simultáneamente. En tales casos, un formulario es más útil.

## Entrada utilizando la interfaz de base

Las tablas en el *contenedor de tablas* se abren haciendo doble clic en su nombre. Si la clave primaria es un campo que se incrementa automáticamente, uno de los campos visibles contendrá el texto *Valor automático*. No es posible introducir ningún valor en el ese campo. Su valor asignado puede modificarse si es necesario, pero solo después de que se haya confirmado el registro.

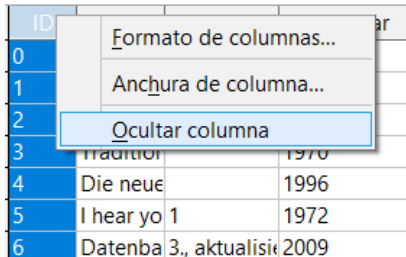


Figura 13: Ocultar columnas

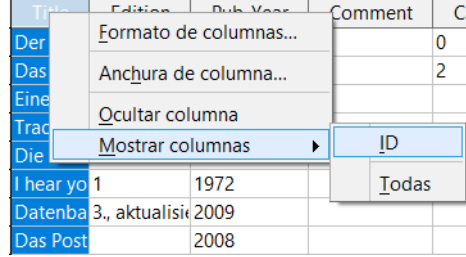


Figura 14: Mostrar columnas

Las columnas individuales en la Vista de datos de la tabla se pueden ocultar. Por ejemplo, si el campo de la clave primaria no necesita ser visible, esto se puede especificar en la tabla en la vista de ingreso de datos haciendo clic derecho en el encabezado de la columna. Esta configuración se almacena en los ajustes de la interfaz. La columna sigue existiendo en la tabla y siempre se puede volver a hacer visible.

La entrada de registros en la tabla generalmente se realiza de izquierda a derecha usando el teclado (teclas *Tab* o *Intro*) o con el ratón.

Cuando se alcanza el último campo de un registro, el cursor salta automáticamente al siguiente registro. La entrada anterior se almacena. El almacenamiento adicional usando **Archivo > Guardar** no es necesario y, de hecho, no es posible. Los datos ya están en la base de datos.



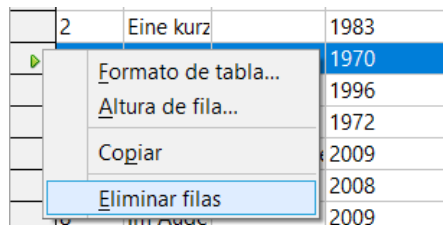
### Precaución

Para HSQLDB, los datos están en la memoria de trabajo. Solo se transferirán al disco duro cuando Base esté cerrada (desafortunadamente es una debilidad desde el punto de vista de la seguridad de los datos). Si Base por alguna razón no se cierra de manera ordenada, puede conducir a la pérdida de datos.

Si no se ingresan datos en un campo que se haya definido previamente durante el diseño de la tabla como obligatorio (NOT NULL), se muestra el correspondiente mensaje de error: `Attempt to insert null into a non-nullable column.`

También se muestran la columna correspondiente, la tabla y la orden SQL (según lo traducido por la interfaz).

Cambiar un registro es fácil: busque el campo, ingrese un valor diferente y abandone el registro.



Para eliminar un registro, seleccione la fila haciendo clic en su encabezado (el área gris de la izquierda), haga clic con el botón derecho y elija Eliminar filas.

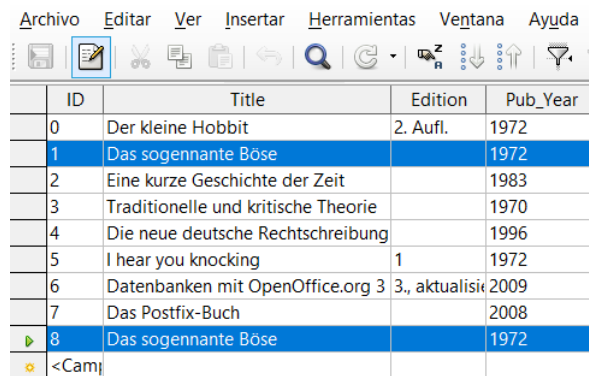


Hay un método, no visible, para copiar filas completas. Para que esto funcione, la clave primaria de la tabla debe estar definida como *Valor automático*.



ID	Title	Edition	Pub_Year	Comment	Category_ID	Mediastyle_ID
0	Der klein	2. Aufl.	1972		0	0
1	Das soge		1972		2	0
2	Eine kurz		1983			0
3	Tradition		1970			1
4	Die neue		1996			0
5	I hear yo	1	1972		1	1
6	Datenba	3., aktualis	2009			0
7	Das Post		2008			0
8	Im Auge		2009		2	1
9	Das soge		1972		2	0
10	Eine kurz		1983			0
11	Eine kurz		1983			0

Primero se resalta el encabezado de la fila con el botón izquierdo del ratón. Mantenga presionado el botón y arrastre el ratón. El cursor cambiará a un símbolo con un signo +. Esto significa que el registro debe ser copiado. Tan pronto aparezca el símbolo se puede soltar el botón del ratón.



ID	Title	Edition	Pub_Year
0	Der kleine Hobbit	2. Aufl.	1972
1	Das sogenannte Böse		1972
2	Eine kurze Geschichte der Zeit		1983
3	Traditionelle und kritische Theorie		1970
4	Die neue deutsche Rechtschreibung		1996
5	I hear you knocking	1	1972
6	Datenbanken mit OpenOffice.org 3	3., aktualis	2009
7	Das Postfix-Buch		2008
8	Das sogenannte Böse		1972

En el ejemplo, el registro con la clave primaria **1** se inserta como un nuevo registro con la nueva clave primaria **9**. Si utiliza la tecla control o shift para resaltar un grupo de registros, estos se copiarán como un grupo.

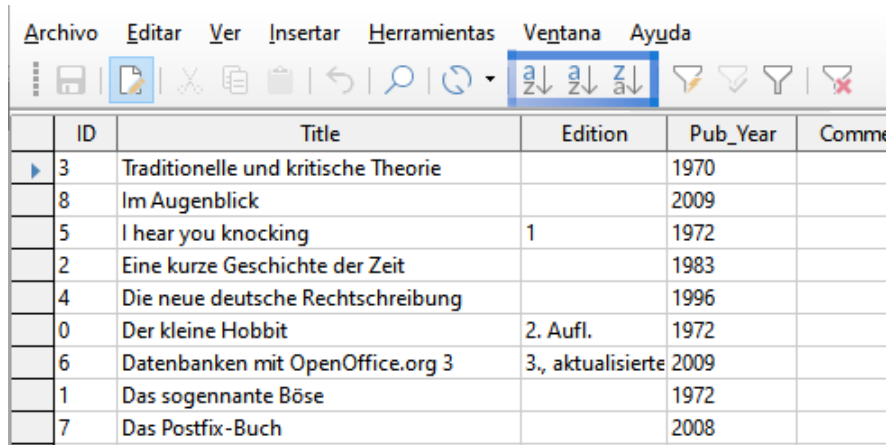
### Consejo

Los encabezados de columna se pueden arrastrar para obtener un ancho adecuado para la entrada. Si esto se hace en una tabla, Base guarda automáticamente el nuevo ancho para la columna en la tabla.

Los anchos de columna en las tablas afectan a los de las consultas. Si las columnas de una consulta son demasiado estrechas, ampliarlas en la consulta solo tendrá un efecto temporal. El nuevo ancho no se guardará. Se debe ampliar la columna en la tabla para que aparezca correctamente en las consultas.

Las funciones *Ordenar*, *Buscar* y *Filtrar* son muy útiles para recuperar registros concretos.

## Ordenar Tablas

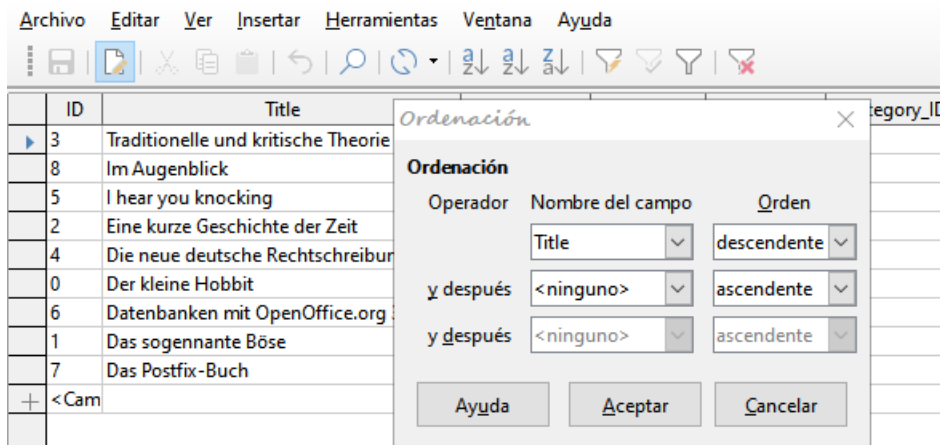


	ID	Title	Edition	Pub_Year	Comm
▶	3	Traditionelle und kritische Theorie		1970	
	8	Im Augenblick		2009	
	5	I hear you knocking	1	1972	
	2	Eine kurze Geschichte der Zeit		1983	
	4	Die neue deutsche Rechtschreibung		1996	
	0	Der kleine Hobbit	2. Aufl.	1972	
	6	Datenbanken mit OpenOffice.org 3	3., aktualisierte	2009	
	1	Das sogenannte Böse		1972	
	7	Das Postfix-Buch		2008	

Figura 15: Ordenación rápida

Los botones resaltados en la figura 15 son los botones de ordenación. El segundo y tercer botón permiten una ordenación rápida. Primero, seleccione un campo. Luego, haga clic en el botón correspondiente al orden ascendente o descendente, y los datos se ordenan por esa columna. En este caso se muestra un orden descendente por el campo *Title*.

La ordenación rápida solo es efectiva para una columna. Para ordenar por varias columnas simultáneamente, se proporciona una función de clasificación más avanzada con el primer botón de ordenación:

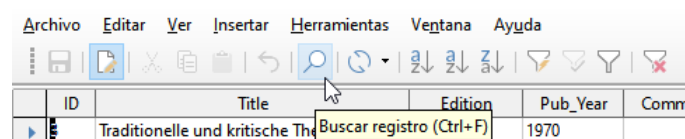


	ID	Title	Edition	Pub_Year	Comm	Category_ID
▶	3	Traditionelle und kritische Theorie		1970		
	8	Im Augenblick		2009		
	5	I hear you knocking	1	1972		
	2	Eine kurze Geschichte der Zeit		1983		
	4	Die neue deutsche Rechtschreibung		1996		
	0	Der kleine Hobbit	2. Aufl.	1972		
	6	Datenbanken mit OpenOffice.org 3	3., aktualisierte	2009		
	1	Das sogenannte Böse		1972		
	7	Das Postfix-Buch		2008		
+	<Cam					

Figura 16: Ordenar por más de una columna

Se seleccionan el nombre del campo de la columna y el orden de clasificación. Si se ha llevado a cabo una ordenación rápida previa, la primera fila ya contendrá los correspondientes nombre del campo y orden de clasificación.

## Buscar en Tablas



	ID	Title	Edition	Pub_Year	Comm
▶		Traditionelle und kritische Theorie		1970	

El botón Buscar registro es un método simple para localizar registros en una tabla grande. Sin embargo, la función de búsqueda es muy lenta para bases de datos grandes, ya que la búsqueda no utiliza una orden SQL dentro de la base de datos. Para una búsqueda más rápida, en lugar de

usar Buscar registro, use una consulta. Para evitar la modificación frecuente de la consulta, puede diseñarse para ejecutarse con parámetros. Consulte la sección «Uso de parámetros en consultas» en el «Capítulo 5, Consultas».

## Consejo

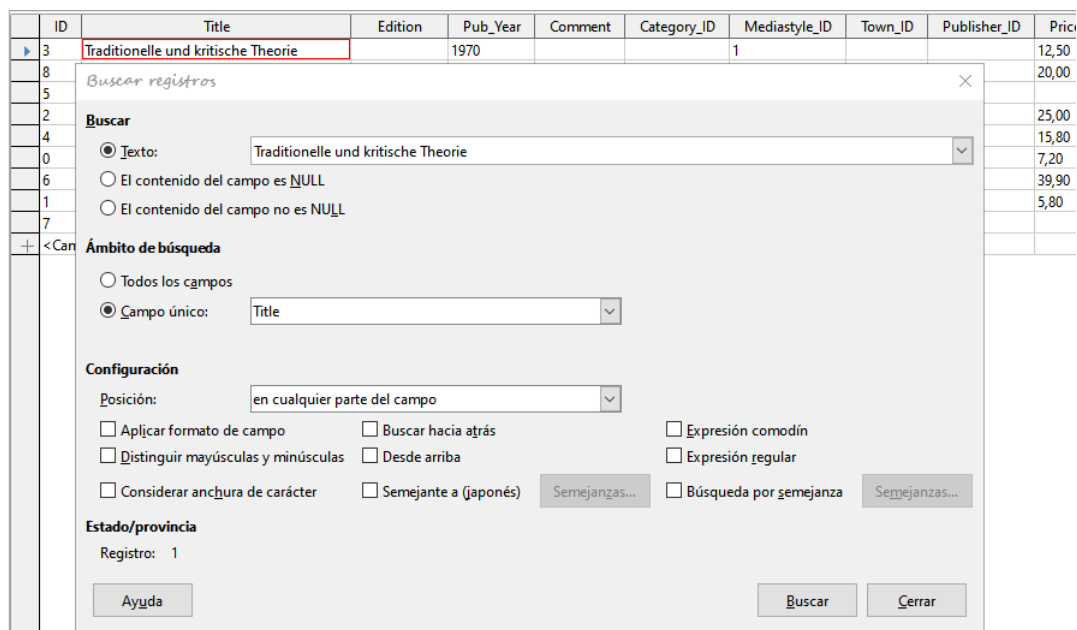
Antes iniciar la búsqueda, asegúrese de que las columnas en las que buscará sean lo suficientemente anchas como para mostrar correctamente los registros que se encuentren. La ventana de búsqueda permanece en primer plano y no podrá corregir la configuración del ancho de columna en la tabla subyacente. Para acceder a la tabla, debe interrumpir la búsqueda.

El botón Buscar registros rellena automáticamente el término de búsqueda con el contenido del campo seleccionado cuando se invoca.

Para que la búsqueda sea efectiva, el área de búsqueda debe ser lo más limitada posible. No tendría sentido buscar el texto de un campo *Título* en un campo *Author*. El nombre de campo *Título* ya se sugiere como el nombre del campo único.

A través de combinaciones específicas pueden obtenerse resultados más precisos. Puede usar los marcadores de posición SQL normales ("\_" para un carácter variable, "%" para un número arbitrario de caracteres variables o "\" como carácter de escape para permitir que se busquen caracteres especiales).

Las expresiones regulares se describen en detalle en la *Ayuda de LibreOffice*. Aparte de estas expresiones, la Ayuda disponible para este módulo es bastante escasa.



ID	Title	Edition	Pub_Year	Comment	Category_ID	Mediastyle_ID	Town_ID	Publisher_ID	Price
3	Traditionelle und kritische Theorie		1970			1			12,50
8									20,00
5									25,00
2									15,80
4									7,20
0									39,90
6									5,80
1									
7									

Figura 17: Máscara de entrada para buscar registros

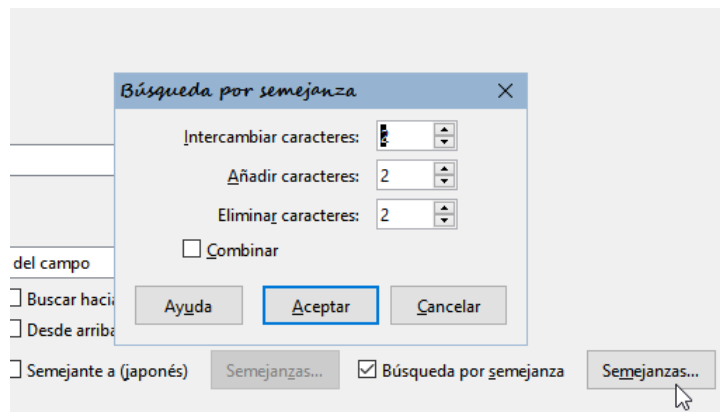


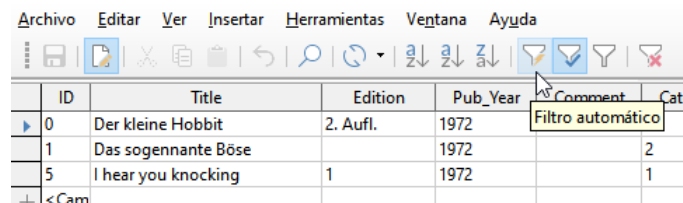
Figura 18: Limitar la búsqueda de similitudes

La función de búsqueda por semejanza es útil cuando necesita excluir errores ortográficos. Cuanto más altos sean los valores que establezca, más registros se mostrarán en la lista final.

Este módulo de búsqueda es el más adecuado para las personas que saben, por el uso regular, exactamente cómo lograr un resultado dado. La mayoría de los usuarios tienen más probabilidades de encontrar registros utilizando un filtro.

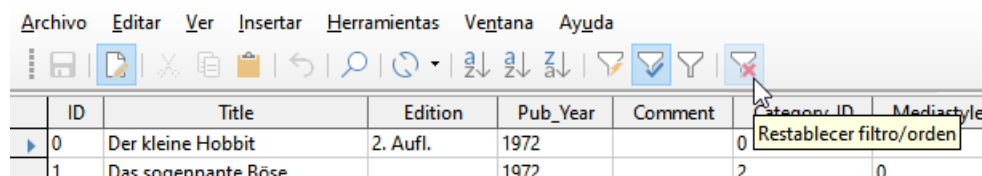
El «Capítulo 4» de esta guía describe el uso de formularios para la búsqueda y cómo el uso de SQL y macros puede lograr una búsqueda con palabras clave.

## Filtrar tablas



Puede filtrar una tabla rápidamente utilizando el *Filtro automático*. Coloque el cursor en un campo, y un clic en el icono hace que el filtro se haga cargo del contenido de este campo. Solo se muestran aquellos registros para los que el campo elegido tiene el mismo contenido. La siguiente figura muestra el filtrado de acuerdo con una entrada en la columna *Pub\_Year*.

El filtro está activo, como lo muestra el icono del filtro con una marca de verificación. El símbolo de filtro se muestra presionado. Este botón es una palanca, por lo que si se hace clic nuevamente, el filtro continúa existiendo, pero ahora se muestran todos los registros. Por lo tanto, si lo desea, siempre puede volver al estado filtrado.

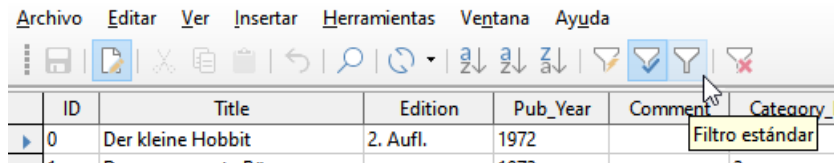


Al hacer clic en el icono *Restablecer filtro/orden* en el extremo derecho, se eliminan todos los filtros y tipos existentes. Los filtros se vuelven inactivos y ya no se pueden recuperar con sus valores anteriores.



## Consejo

Todavía se pueden ingresar registros normalmente en una tabla filtrada o en una que haya sido restringida por una búsqueda. Permanecen visibles en la vista de tabla hasta que la tabla se actualiza presionando el botón *Actualizar*.



El icono *Filtro estándar* abre un cuadro de diálogo en el que puede filtrar utilizando varios criterios simultáneos, de forma similar a la ordenación. Si el *Filtro automático* está en uso, la primera línea del Filtro estándar ya mostrará este criterio de filtro existente.

Operador	Nombre del campo	Condición	Valor
	Pub_Year	=	1972
Y	- ninguno -		
Y	- ninguno -		

Figura 19: Filtrado de datos múltiples con el filtro estándar

El filtro estándar proporciona muchas de las funciones de filtrado de datos SQL. Las siguientes órdenes SQL en el desplegable *Condición* están disponibles:

Condición	Descripción
=	Igualdad exacta corresponde a como, pero sin ningún marcador de posición adicional
<>	No igual
<	Menor que
<=	Menor o igual que
>	Mayor que
>=	Mayor o igual que
como	Para texto, escrito entre comillas (""); "_" para un carácter variable. "% " para un número arbitrario de caracteres variables
no como	Opuesto a como. En SQL <code>NOT LIKE</code>
nulo	Sin entrada, ni siquiera un carácter de espacio. En SQL, <code>NULL</code>
No nulo	Opuesto a nulo. En SQL <code>NOT NULL</code>

Antes de que un criterio de filtro se pueda combinar con otro, la siguiente fila debe tener al menos un nombre de campo seleccionado. En la Figura 19, la palabra - ninguno - se muestra en lugar de un nombre de campo, por lo que la combinación no está activa. Los operadores de combinación disponibles son **Y**(AND) y **O** (OR).

El nombre del campo puede ser un nuevo nombre de campo o uno previamente seleccionado.

Incluso para grandes masas de datos, el número de registros recuperados puede reducirse a un conjunto más manejable con un filtrado hábil utilizando estas tres posibles condiciones.

En el caso de los formularios de filtrado también, hay algunas posibilidades adicionales (descritas en el «Capítulo 4») que no están disponibles en la interfaz.

## Entrada directa usando SQL

La entrada directa de datos usando SQL es útil para ingresar, cambiar o eliminar múltiples registros con una sola orden.

### Introducir nuevos registros

```
INSERT INTO "Nombre_Tabla" [( "Nombre_Campo" [,...] )] { VALUES("Valor_campo" [,...] ) | <Fórmula_Selección>};
```

Si no se especifica *Nombre\_Campo*, todos los campos se deben completar y en el orden correcto (como se establece en la tabla). Eso incluye el campo de clave primaria incrementado automáticamente, donde está presente. Los valores ingresados también pueden ser el resultado de una consulta (<Fórmula\_Selección>). A continuación se proporciona información más exacta.

```
INSERT INTO "Nombre_Tabla" ("Nombre_Campo") VALUES ('Test');  
CALL IDENTITY();
```

En la tabla, en la columna *Nombre\_Campo*, se inserta el valor *Test*. No se toca la *ID* del campo de la clave primaria incrementada automáticamente. El valor correspondiente para *ID* debe crearse por separado utilizando `CALL IDENTITY ()`. Esto es importante cuando se utilizan macros, para que el valor de este campo clave se pueda utilizar más adelante.

```
INSERT INTO "Nombre_Tabla" ("Nombre_Campo") SELECT "Otro_nombre_Campo" FROM  
"Nombre_de_otra_Tabla";
```

En la primera tabla, se insertan tantos registros nuevos en *Nombre\_Campo*, como están presentes en la columna *Otro\_nombre\_Campo* de la segunda tabla. Naturalmente, para limitar el número de entradas, se puede utilizar aquí una fórmula de selección.

### Editar registros existentes

```
UPDATE "Nombre_Tabla" SET "Nombre_Campo" = <Expresión> [, ...] [WHERE  
<Expresión>];
```

Cuando está modificando muchos registros a la vez, es muy importante verificar cuidadosamente la orden SQL que está ingresando. Suponga que todos los estudiantes en una clase deben ascender un año:

```
UPDATE "Nombre_Tabla" SET "Año" = "Año"+1;
```

Nada podría ser más rápido: todos los registros de datos se modifican con una sola orden. Pero, imagine que ahora debe determinar qué estudiantes no deberían haberse visto afectados por este cambio. Hubiera sido más sencillo usar un campo *Repetidor* tipo *SÍ/No* para los repetidores y luego subir solo a aquellos estudiantes para los que **no** se marcó este campo:

```
UPDATE "Nombre_Tabla" SET "Año" = "Año"+1 WHERE "Repeticion" = FALSE;
```

Estas condiciones funcionan cuando el campo en cuestión solo puede tomar dos valores: TRUE o FALSE, pero en este caso puede tomar un valor vacío NULL. Sería más seguro si la condición se formulara como `WHERE "Repeticion" <> TRUE`.

Si posteriormente desea que se ingrese un valor predeterminado en un campo en particular donde esté vacío, puede hacerlo con la orden:

```
UPDATE "Tabla" SET "Campo" = 1 WHERE "Campo" IS NULL;
```

Puede modificar varios campos a la vez asignándoles directamente valores. Suponga que una tabla para libros incluye los nombres de sus autores. Se descubre que *Erich Kästner* ha sido frecuentemente ingresado como *Eric Käschtner*.

```
UPDATE "Libros" SET "Nombre_Autor" = 'Erich', "Apellido_Autor" = 'Kästner' WHERE "Nombre_Autor" = 'Eric' AND "Apellido_Autor" = 'Käschtner';
```

También es posible hacer cálculos con `UPDATE`. Si, por ejemplo, las mercancías que cuestan más de 150,00 € se incluyen en una oferta especial y el precio se reduce en un 10%, se puede llevar a cabo de la siguiente manera:

```
UPDATE "Nombre_Tabla" SET "Precio" = "Precio"*0.9 WHERE "Precio" >= 150.00
```

Cuando elige el tipo de datos `CHAR`, el campo tiene un ancho fijo. Cuando haga falta, el texto se rellenará con caracteres nulos. Si se convierte a `VARCHAR`, estos caracteres nulos permanecen. Para eliminarlos, use la función `RTRIM` para eliminar caracteres de vacíos o de control de la derecha:

```
UPDATE "Nombre_Tabla" SET "Nombre_Campo" = RTRIM("Nombre_Campo");
```

## Eliminar registros existentes

```
DELETE FROM "Nombre_Tabla" [WHERE <Expresión>];
```

Sin la expresión condicional, la orden...

```
DELETE FROM "Nombre_Tabla";
```

elimina todo el contenido de la tabla.

Por esta razón, es preferible que la orden sea más específica. Por ejemplo, si se proporciona el valor de la clave primaria, solo se eliminará el registro con esa clave.

```
DELETE FROM "Nombre_Tabla" WHERE "ID" = 5;
```

Si, en el caso de un préstamo, el registro de un artículo se debe eliminar cuando se devuelva el artículo, se puede hacer usando:

```
DELETE FROM "Nombre_Tabla" WHERE NOT "Return_date" IS NULL;
```

o de manera alternativa con:

```
DELETE FROM "Nombre_Tabla" WHERE "Return_date" IS NOT NULL;
```

## Importar datos de otras fuentes

A veces necesitamos importar a Base conjuntos de datos de otro programa a través del portapapeles. Esto implica crear una nueva tabla o agregar registros a una existente.



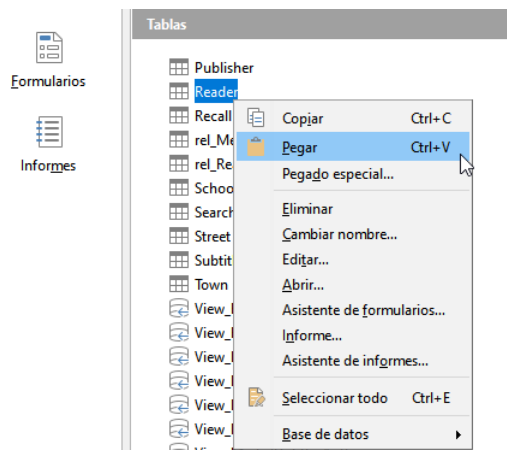
## Nota

Para importar datos usando el portapapeles, Base debe poder leer el formato de datos. Este siempre será el caso para los archivos de datos abiertos en LibreOffice.

Por ejemplo, si las tablas de una base de datos externa deben leerse en un archivo \*.odb, esa base de datos primero debe abrirse en LibreOffice o registrarse con LibreOffice como fuente de datos. Consulte «Acceso a bases de datos externas» en el Capítulo 2, «Creación de una base de datos».

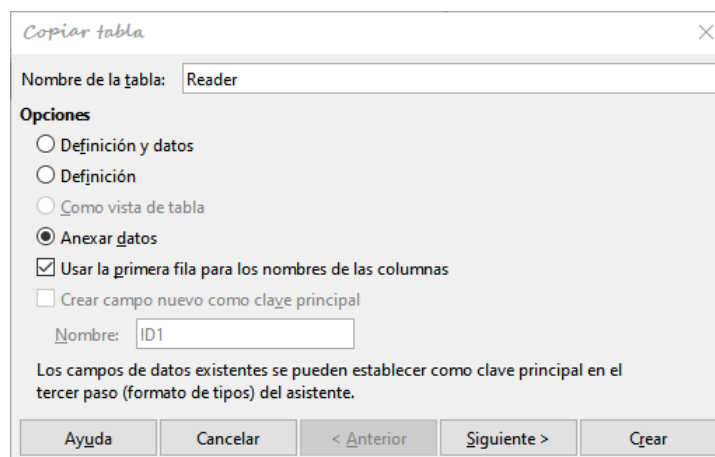
	A	B	C	D
1	ID	FirstName	LastName	
2	10	Robert	Großkopf	
3	11	Maike	Longfoot	
4	12	Georte	Orwell	

Aquí se ha copiado una pequeña tabla de ejemplo de una hoja de cálculo Calc en el portapapeles. Luego se pega en el contenedor de la tabla de Base. Por supuesto, esto también podría haberse hecho seleccionándolo con el botón izquierdo del ratón y luego arrastrándolo.



En el contenedor de tablas, haga clic con el botón derecho para abrir el menú contextual de la tabla a la que se agregarán los registros.

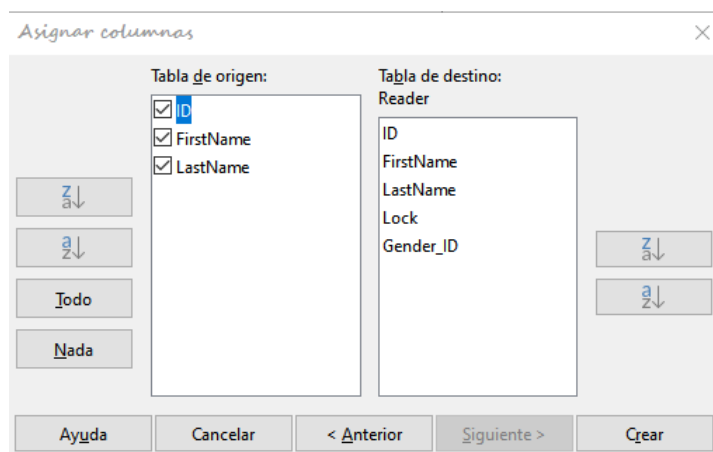
## Agregar registros importados a una tabla existente



El nombre de la tabla aparece en el asistente de importación. Seleccionar Al mismo tiempo *Anexar datos* y *Usar la primera fila para los nombres de las columnas*, puede ser necesario o no,



dependiendo de su versión de LibreOffice. Si se van a agregar los registros, no se requiere definición de datos. Una clave principal también tiene que estar disponible para su uso.



Las columnas de la tabla de origen Calc y la tabla de destino en Base no tienen que coincidir en su secuencia, nombres o número general. Solo se transfieren los elementos seleccionados del lado izquierdo. La correspondencia entre las tablas de origen y destino debe ajustarse utilizando los botones de flecha a cada lado.

Esto completa la importación.

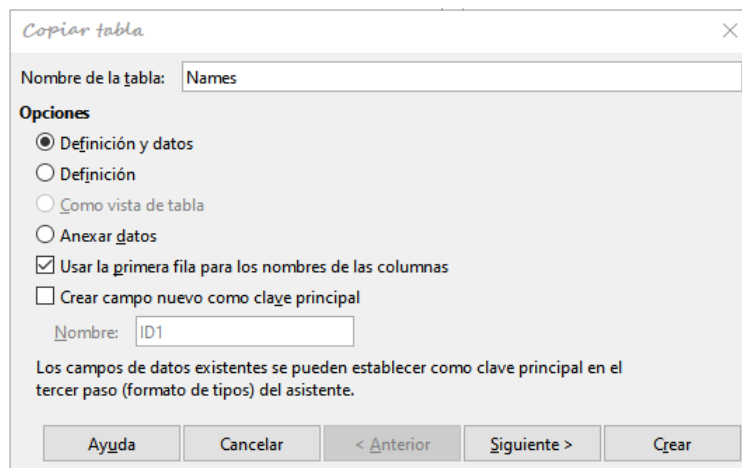
La importación puede generar problemas si:

- Los campos en la tabla de destino requieren una entrada obligatoria, pero la tabla de origen no proporciona datos para ellos.
- Las definiciones de campo en la tabla de destino no están en concordancia con las de la tabla de origen (por ejemplo, se debe ingresar un nombre en un campo numérico, o el campo de destino tiene muy pocos caracteres para los datos).
- La tabla de origen proporciona datos incongruentes con los de la tabla de destino, por ejemplo, valores no únicos para claves primarias u otros campos definidos como únicos.

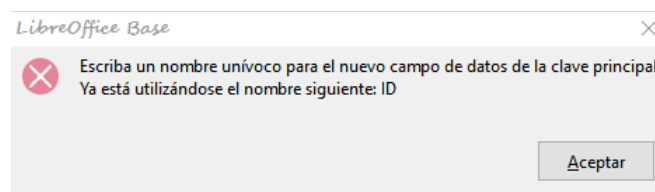
### **Crear una nueva tabla para datos importados**

Se puede crear una nueva tabla seleccionando por ejemplo un rango de una hoja de cálculo Calc, copiándolo al portapapeles y al hacer clic derecho en el contenedor de tablas seleccionando *Pegar* del menú emergente.

Cuando se inicia el asistente de copia, el nombre de la tabla que estaba seleccionada anteriormente en el contenedor de tablas aparece automáticamente. Debe cambiar el nombre si está creando una nueva tabla, ya que no es posible tener dos tablas con el mismo nombre en la base de datos. El nombre de la nueva tabla en el ejemplo es *Names*. En la importación, se transfieren la definición de la tabla y los datos, y se usa la primera fila para los nombres de las columnas.



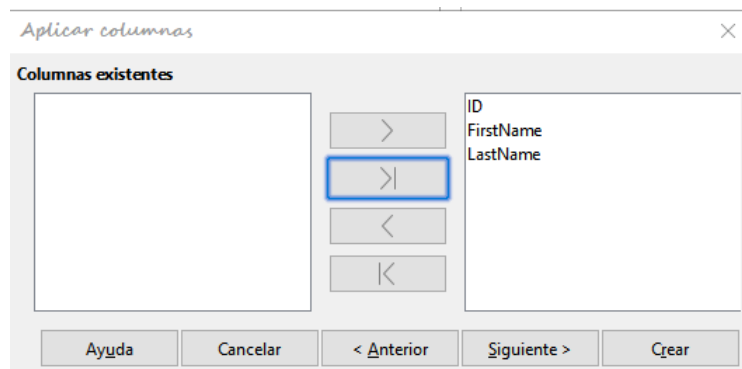
En este punto, puede crear un nuevo campo adicional para una clave primaria. El nombre de este campo no debe existir como encabezado de columna en la tabla Calc. De lo contrario, recibirá el mensaje de error:



Lamentablemente, este mensaje no explica la situación correctamente.

Si desea utilizar un campo existente del rango de datos como su clave principal, no marque *Crear campo nuevo como clave principal*. Establecerá su campo de clave principal en la tercera página del Asistente (figura 20).

Todas las columnas disponibles se transfieren usando el segundo botón y pasarán del área izquierda a la derecha (>|).



El formato de los tipos de tabla a menudo requiere un ajuste. De manera predeterminada, los campos se definen como campos de texto con un tamaño muy grande.

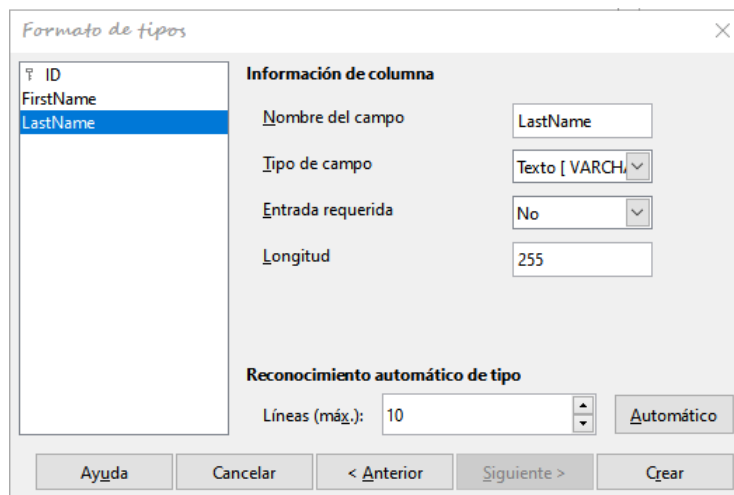
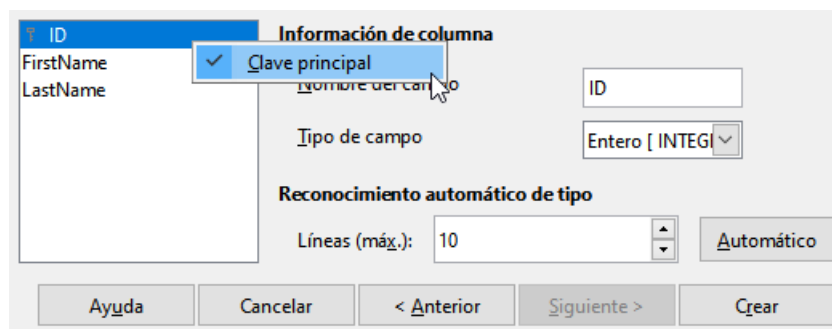


Figura 20: Tercera página del Asistente

Por lo tanto, deberá ajustar su tamaño si así lo desea. Los campos numéricos o de fecha se deben establecer del tipo adecuado con el desplegable *Tipo de campo*. En el caso de los números decimales, deberá también verificar el número de lugares decimales.



La opción para elegir una clave principal está presente, pero algo oculta, en el menú contextual del campo que queramos designar. En este ejemplo, el campo *ID* se ha formateado de manera que permita su uso como clave principal. Después debe establecerse la clave primaria usando el menú contextual del nombre del campo, si no se eligió *crear campo nuevo como clave principal* en el primer paso del asistente (aparecerá un icono de una llave a la izquierda del campo).

Cuando hace clic en el botón *Crear*, la tabla se crea y se llena con los datos copiados. La nueva clave primaria no es una clave de Valor automático. Debe editar la tabla para cambiarla si desea un valor automático y también si desea realizar más operaciones de formateo en el resto de campos.

### División de datos al importar

A veces, los datos de origen no están disponibles en la forma deseada. Las direcciones, por ejemplo, a menudo se introducen en hojas de cálculo como un solo registro (calle, número, ciudad y código postal). Al importarlos, es posible que desee colocar la ciudad y el código postal en una tabla separada, que luego se puede vincular a la tabla principal *Addresses*.

La siguiente es una forma posible de crear esta relación directamente:

- 1) La tabla completa con toda la información de la dirección se importa a Base como una tabla llamada *Addresses*. (Vea las secciones anteriores para más detalles).
- 2) Los datos del campo Código postal y Ciudad se leen con una consulta,

```
SELECT DISTINCT "Postcode", "Town" FROM "Addresses";
```

- 3) Se copian y almacenan como una tabla separada *Postcode\_Town*. Para ello, se agrega un campo *ID* y se especifica como una clave principal con Valor automático. Aquí está la consulta:
- 4) Se agrega un nuevo campo llamado *Postcode\_ID* a la tabla *Addresses* del mismo tipo que el campo *ID* de la tabla *Postcode\_Town*
- 5) Usando **Herramientas > SQL**, se realiza una actualización de la tabla *Addresses*:

```
UPDATE "Addresses" AS "a" SET "a"."Postcode_ID" = (SELECT "ID" FROM "Postcode_Town" WHERE "Postcode"||"Town" = "a"."Postcode"||"a"."Town");
```

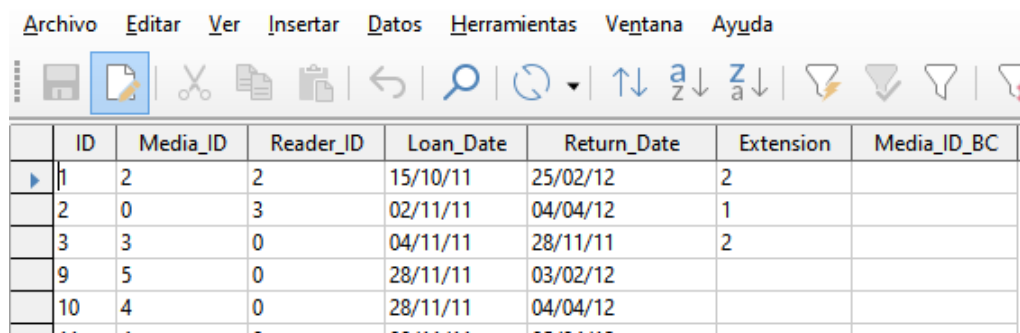
- 6) La tabla *Addresses* se abre para editar y los campos Código postal y Ciudad se eliminan. Este cambio se guarda y la tabla se cierra nuevamente.

Una vez separadas las tablas se crea una relación 1 : n entre la tabla *Postcode\_Town* y la tabla *Addresses*. Esta relación se define usando **Herramientas > Relaciones**.

Para obtener detalles sobre el código SQL, consulte también el Capítulo 5, «Consultas».

## Problemas con estos métodos de entrada de datos

La entrada utilizando una tabla sola no tiene en cuenta los enlaces a otras tablas. Esto queda claro en un ejemplo de un préstamo de artículos.



ID	Media_ID	Reader_ID	Loan_Date	Return_Date	Extension	Media_ID_BC
1	2	2	15/10/11	25/02/12	2	
2	0	3	02/11/11	04/04/12	1	
3	3	0	04/11/11	28/11/11	2	
9	5	0	28/11/11	03/02/12		
10	4	0	28/11/11	04/04/12		

La tabla *Loan* contiene claves externas para el artículo prestado (*Media\_ID*) y el lector correspondiente (*Reader\_ID*), así como una fecha de préstamo (*Loan\_Date*). En la tabla, por lo tanto, debemos ingresar en el momento del préstamo dos valores numéricos (número de artículo y número de lector) y una fecha. La clave principal se ingresa automáticamente en el campo *ID*. Si el lector realmente corresponde al número no es manifiesto a menos que una segunda tabla para los lectores esté abierta al mismo tiempo. Si el artículo se prestó con el número correcto tampoco es evidente. Aquí el préstamo debe basarse en la etiqueta del artículo o en otra tabla abierta.

Todo esto es mucho más fácil de lograr usando formularios. Los lectores y los artículos se pueden buscar utilizando controles de cuadro de lista. En los formularios, los nombres de usuario y artículo son visibles y sus identificadores numéricos están ocultos. Además, un formulario puede diseñarse de modo que se pueda seleccionar primero un usuario, luego una fecha de préstamo, y cada conjunto de artículos se asigna a esta fecha por número. En otros lugares, estos números pueden hacerse visibles con descripciones exactas que correspondan a los artículos.

La entrada directa en tablas es útil solo para bases de datos con tablas simples. Cuando existan relaciones entre tablas, es mejor utilizar un formulario especialmente diseñado. En los formularios, estas relaciones se pueden manejar utilizando subformularios o campos de lista.