



Clase 08. SQL

TABLAS

***RECUERDA PONER A GRABAR LA
CLASE***





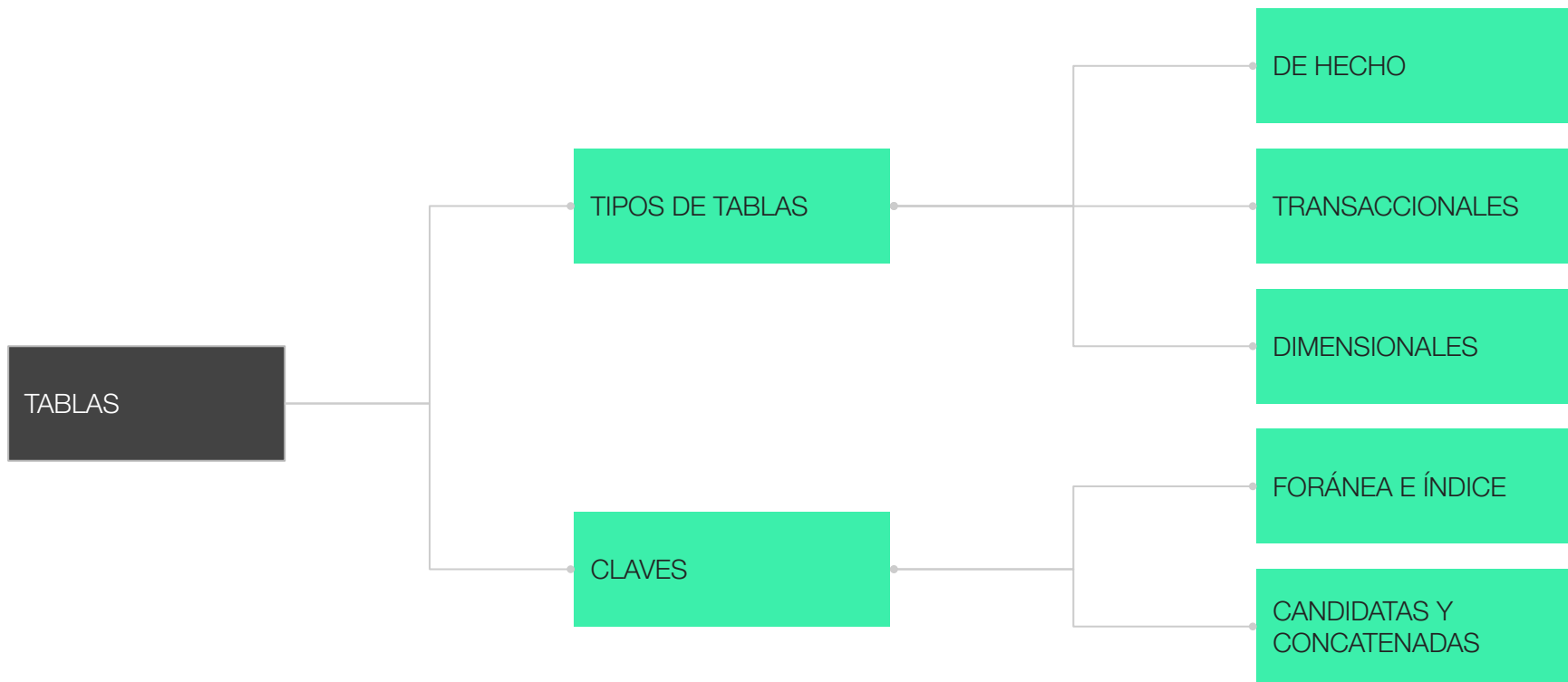
OBJETIVOS DE LA CLASE

- Conocer los conceptos generales.
- Identificar los diferentes tipos de tablas.
- Identificar los tipos de relación.

MAPA DE CONCEPTOS

MAPA DE CONCEPTOS CLASE 8

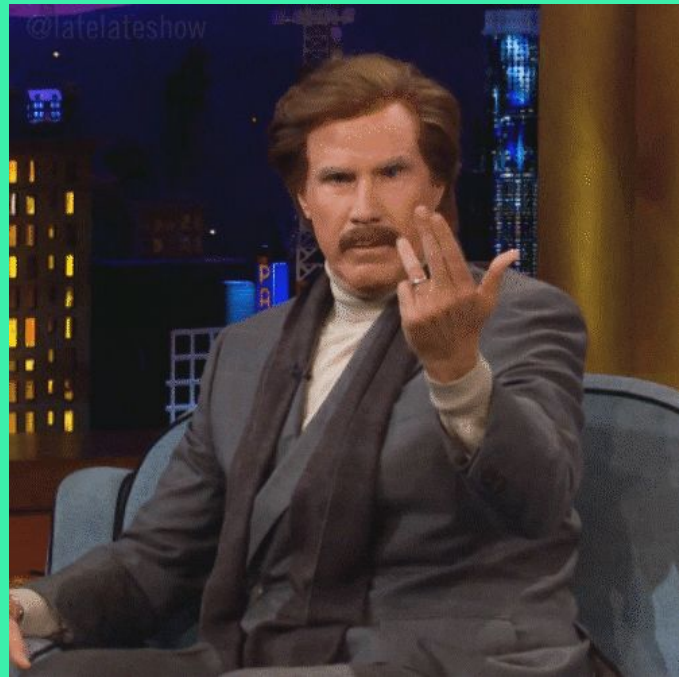
¡Para
recordar!



Realicemos un **repaso general** por los diferentes tipos de tabla que conforman una DB.

+

También veremos las **claves**, su concepto, tipos e implementación de cada una de ellas.

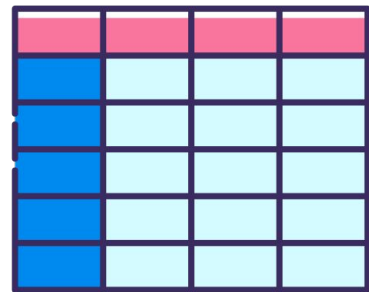


TABLAS

CONCEPTO GENERAL

RECORDEMOS: *concepto de tablas SQL*

Las tablas almacenan la información en forma de registros o tuplas. Para ello, respetan la estructura de cada dato de un registro, el cual condice con la definición del campo que lo almacena.



Propiedades de una tabla

Entendemos por **propiedades** a las columnas, tipos de datos, claves e índices de una tabla.

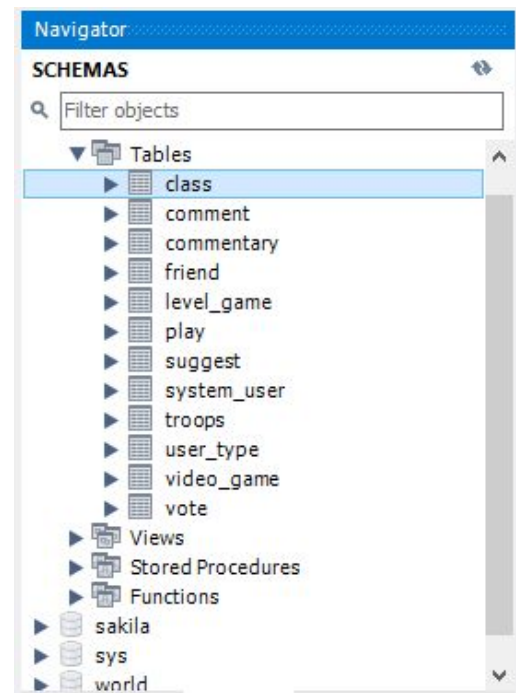
Y **cuando utilizamos un SGBD** como **Mysql Workbench**, podemos observar toda esta información mucho más cómodamente.

Árbol de navegación

En el panel Navigator-Schemas de Mysql Workbench, encontrarán un árbol de características desplegable, que permite acceder a las

propiedades de una tabla donde encontrarán:

- Columns.
- Indexes.
- Foreign Keys.
- Triggers.



Árbol de navegación: Columns

En el nodo **Columns**, podrán apreciar al desplegarlo todas las columnas de la tabla seleccionada.

Luego, pulsando sobre cualquiera de ellas, verán en el panel inferior **Object Info** la información de las propiedades que se le asignaron.

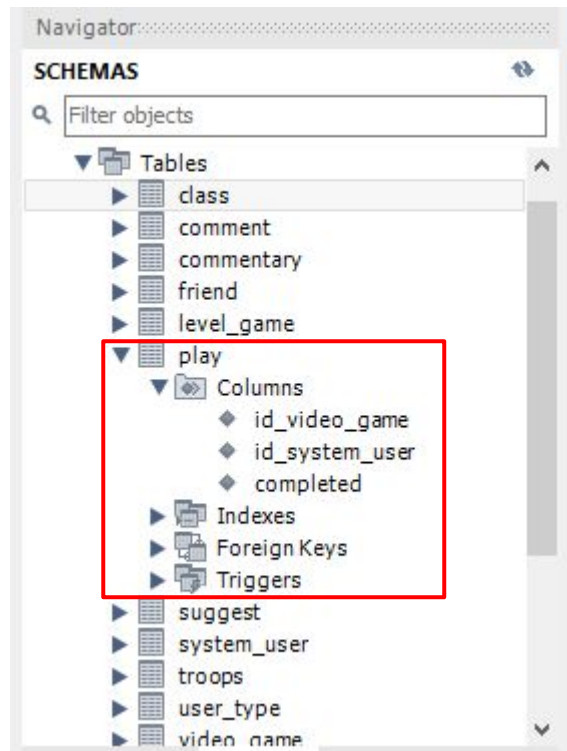


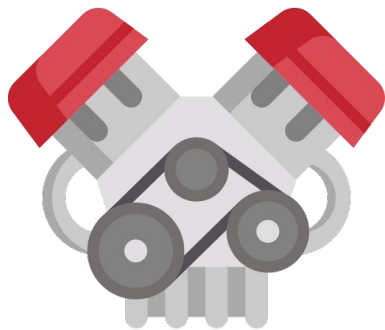
Table Inspector

También realizando clic derecho sobre la tabla se tiene la herramienta **Table Inspector**, donde encontramos el apartado diferentes solapas (Info, Columns, Indexes, Triggers, entre otras), con posibilidad de ver el detalle completo de sus valores.

Query 1 gammers_model.play x									
Info Columns Indexes Triggers Foreign keys Partitions Grants DDL									
Column	Type	Default Value	Nullable	Character Set	Collation	Privileges	Extra	Comments	
completed	tinyint(1)		NO			select,insert,update,references			
id_system_user	int		NO			select,insert,update,references			
id_video_game	int		NO			select,insert,update,references			

TABLAS TRANSACCIONALES

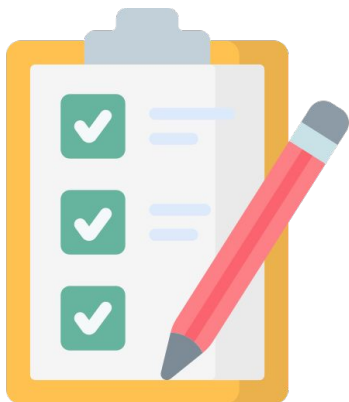
Tablas transaccionales: concepto



Mysql soporta diversos motores de almacenamiento, y estos a su vez soportan diferentes tipos de tablas.

Algunos trabajan con tablas transaccionales, mientras que la mayoría de los otros no.

Tablas transaccionales: concepto



Entre estos motores podemos encontrar:

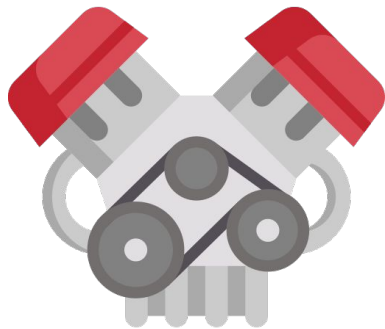
- MyISAM.
- MEMORY.
- InnoDB y BDB.
- EXAMPLE.
- NDB Cluster.
- ARCHIVE.
- CSV.
- FEDERATED.

Tablas transaccionales: concepto



De todos ellos, **InnoDB** y **BDB** son motores transaccionales, siendo el primero el más elegido en Mysql, gracias a que **InnoDB** soporta bloqueos y capacidades de **COMMIT**, así como **ROLLBACK**, además de **Recuperación ante Fallas**.

Tablas transaccionales: concepto

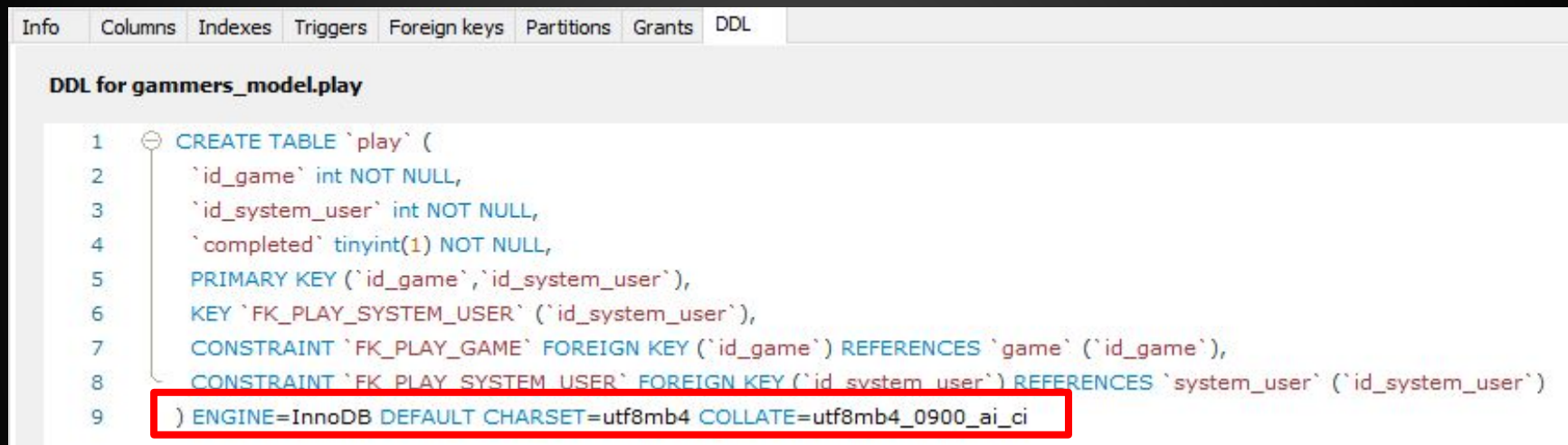


InnoDB también permite **bloqueo a nivel de filas**, mejorando su rendimiento y **soportando múltiples usuarios en simultáneo**.

También fue diseñado para procesar de manera ágil **grandes volúmenes de datos**.

Tablas transaccionales

Puedes encontrar referencia del motor utilizado a través de la opción **Table Inspector** en la solapa **DDL**. Allí verás la opción por defecto de MySQL **ENGINE=InnoDB**.

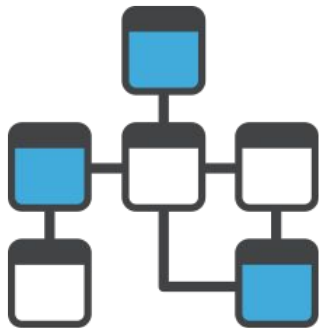


```
DDL for gammers_model.play

1 CREATE TABLE `play` (
2   `id_game` int NOT NULL,
3   `id_system_user` int NOT NULL,
4   `completed` tinyint(1) NOT NULL,
5   PRIMARY KEY (`id_game`,`id_system_user`),
6   KEY `FK_PLAY_SYSTEM_USER` (`id_system_user`),
7   CONSTRAINT `FK_PLAY_GAME` FOREIGN KEY (`id_game`) REFERENCES `game` (`id_game`),
8   CONSTRAINT `FK_PLAY_SYSTEM_USER` FOREIGN KEY (`id_system_user`) REFERENCES `system_user` (`id_system_user`)
9 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

TABLAS DE HECHO

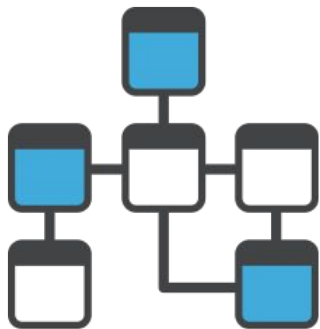
Tablas de hecho: concepto



Este tipo de tablas se ocupan de capturar información de un único hecho de negocios.

Su principal objetivo es generar un evento de medición, específico de la información clave que se debe analizar.

Tablas de hecho: concepto



Son utilizadas para el modelo analítico de un sistema de información de base de datos.

El segmento de **Analítica de Datos**, basado en **Business Intelligence**, se apoya en tablas de hecho y dimensionales para obtener las métricas de reportes que los usuarios buscan.

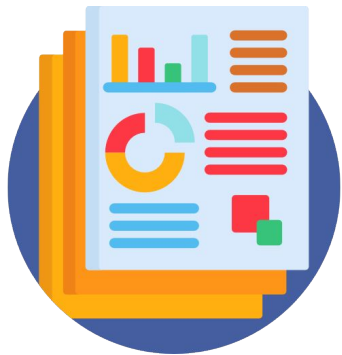
Tablas de hecho: tipos de columnas



En las tablas de hecho existen tres tipos de columnas:

- Aditivas.
- Semi-aditivas.
- No-aditivas

Tablas de hecho: aplicación



Su principal beneficio surge cuando debemos obtener métricas para armar, por ejemplo, reportes utilizando Power BI, Tableau, u otra herramienta similar.

Nuestra DB podría incluir tablas de hecho sin relación alguna con el **Schema**, albergando sólo los “*números importantes*”.

VEAMOS UN EJEMPLO

Tablas de hecho: ejemplo

Imaginemos que existe una tabla de **games_completed** (*no normalizada*), que contiene los datos de **first_name**, **last_name** de los jugadores, **name** de los videojuegos que han finalizado, y la cantidad de juegos completados que tiene cada uno. Esta información podría reflejarse directamente en una **tabla de hecho**:

	first_name	last_name	name	games_completed
►	Abigael	Haversham	The Dark Pictures: House Of Ashes	1
	Adeline	Bartell	Voice of Cards	2
	Aguie	Pavelin	Grand Theft Auto: The Trilogy - The Definitive E...	2
	Aguie	Pavelin	Fast & Furious: Spy Racers El Retorno de SH1F...	2
	Alameda	Ludlow	Far Cry 6	1
	Alanna	Mapston	Poppy Playtime	2
	Alano	Semrad	Far Cry 6	3
	Alexio	Ruddell	Far Cry 6	5
	Alexio	Ruddell	Project Zero: Maiden of Black Water	5
	Alia	Michele	Lost Ark	1
	Ame	Lassell	Pokémon Diamante Brillante / Perla Reluciente	1

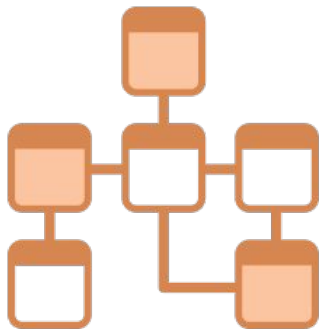
TABLAS DIMENSIONALES

Tablas dimensionales

Ya tenemos las **tablas de hecho**, las cuales concentran la información más importante de un negocio.

Veamos ahora cómo las **tablas dimensionales** se ocupan de nutrir y complementar el detalle de la información que hacen a las primeras.

Tablas dimensionales: concepto



Las **tablas dimensionales** brindan información uniforme con datos asociados a otras tablas.

A diferencia de las **tablas de hecho**, estas portarán datos adicionales relacionados con la información de las tablas de hecho.

Tablas dimensionales: información



La información que podrán alojar las tablas dimensionales abarca desde datos básicos que contenga el esquema, hasta por ejemplo:

- Datos demográficos detallados.
- Datos de perfil de los usuarios.
- Preferencias de los usuarios.

Entre otros.

Tablas dimensionales: beneficios



Toda la información adicional y detallada permitirá tener **métricas efectivas**, y generar **aportes de valor** a las campañas publicitarias y/o de Marketing.

Además, mantener una topología del tipo estrella entre todo el cúmulo de información, permite aplicar la normalización de bases de datos de la forma más óptima.

VEAMOS UN EJEMPLO

Tablas dimensionales: ejemplo

La tabla del centro representa la tabla de hecho **games_completed**, mientras que el resto de las tablas, que aportan información detallada, son las **Tablas Dimensionales**.

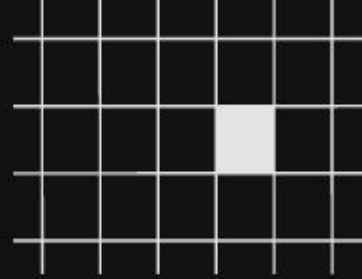


Ejemplo
en vivo



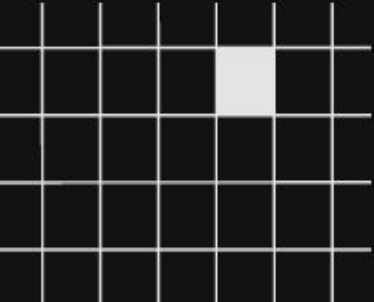
¡VAMOS A PRACTICAR UN POCO!

CODER HOUSE



¡VAMOS A PRACTICAR UN POCO!

Ejemplo en vivo de Tablas Dimensionales.





BREAK

¡5/10 MINUTOS Y VOLVEMOS!

CLAVES

IMPLEMENTACIÓN DE LAS DIFERENTES CLAVES E ÍNDICES

Claves e índices

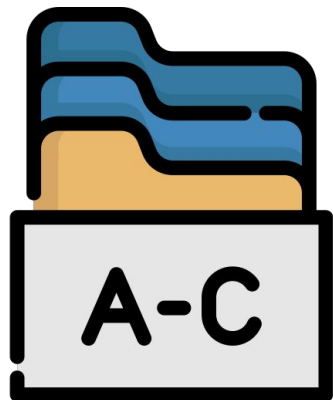
¡Para
recordar!



En la **Clase 02** realizamos una aproximación teórica de la importancia de los índices y claves en las tablas de datos.

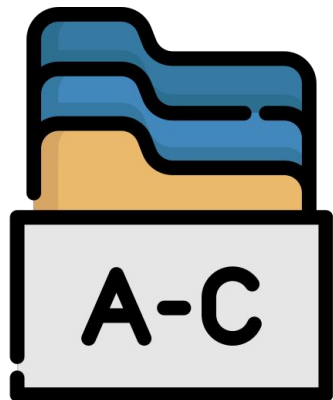
Profundicemos ahora un poco más la funcionalidad que aportan.

Claves e índices



En el **ecosistema de la DB** los índices son fundamentales, tanto para la normalización de la información, como para su posterior búsqueda y relación entre tablas de almacenamiento.

Claves e índices



Se los considera como un grupo de datos que vinculan a una o varias columnas de diferentes tablas, creando entre ellas una relación de contenido.

Claves e índices: beneficio



Podemos destacar la importancia de los mismos para acelerar las consultas de información (optimización de consultas), algo que se vuelve mucho más efectivo cuando una tabla con datos crece considerablemente.

ÍNDICES

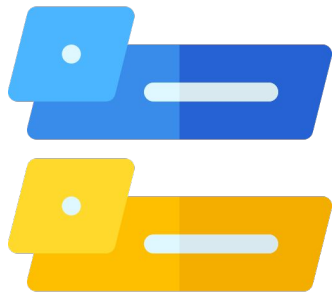
Índices



En el momento en el que definimos una tabla, es cuando el índice debe ser especificado correctamente.

Si bien las tablas pueden modificarse, aplicar un índice desde el principio es la mejor opción para todo el **schema**.

Índices



Podemos recurrir a la sentencia **DESCRIBE**, además de **Table Inspector**, para acceder a la información de los índices de una tabla.

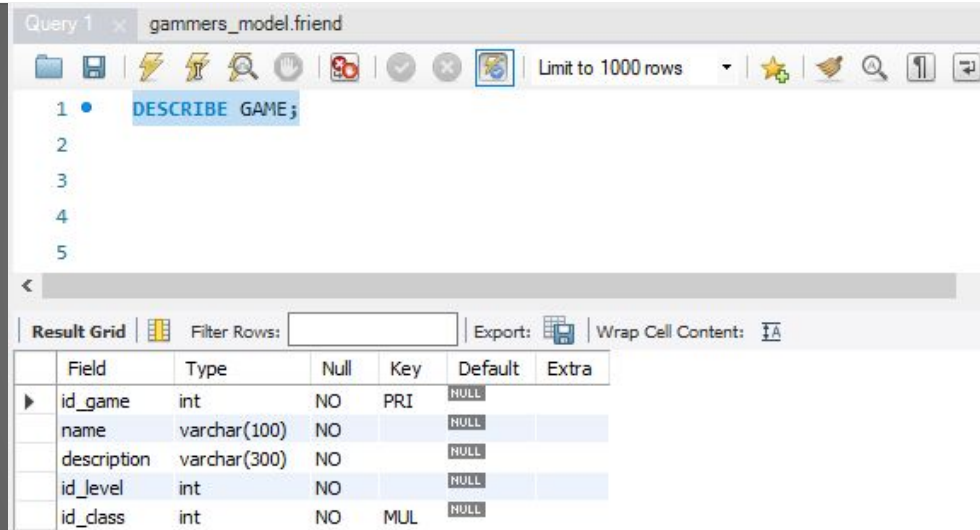
```
describe nombre_de_tabla;
```

¡Para
recordar!



Índices: describe <tabla>

Con **DESCRIBE** accedemos a la información propia de la construcción de dicha tabla, y veremos la columna **Key** con la información de los campos con índice, más la columna **Extra** con la información adicional de los mismos.



Query 1 x gammers_model.friend

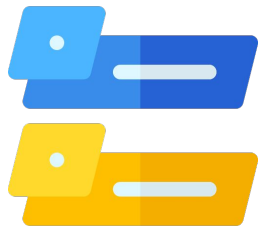
Limit to 1000 rows

```
1 DESCRIBE GAME;
```

Result Grid

	Field	Type	Null	Key	Default	Extra
▶	id_game	int	NO	PRI	NULL	
	name	varchar(100)	NO		NULL	
	description	varchar(300)	NO		NULL	
	id_level	int	NO		NULL	
	id_class	int	NO	MUL	NULL	

Índices



La columna **Key** muestra si el campo está indexado, a través de los indicadores:

- **PRI**: indica que el campo es una clave primaria.
- **UNI**: indica que el campo es un índice UNIQUE.
- **MUL**: indica si se permiten múltiples ocurrencias.

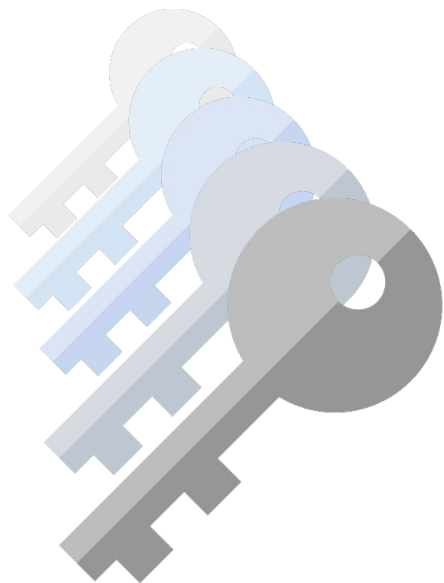
CLAVES PRIMARIAS

Claves Primarias: Primary Key



Este tipo de clave fue creado para poder **establecer**
consultas realmente veloces sobre los datos
almacenados en una tabla.

Primary Key: Único



Otra característica de las claves primarias, es que **cada uno de los registros que se almacene en la tabla deberá ser único.**

Por consiguiente, no podrá repetirse su valor en ningún otro registro que allí se guarde.

Primary Key: AUTOINCREMENT

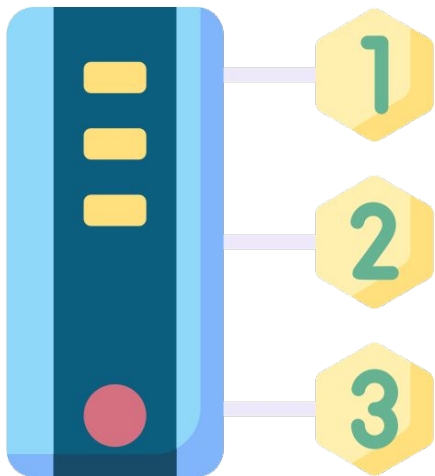


Para evitar la duplicidad de datos en estas columnas, debemos agregar la propiedad

AUTOINCREMENT al definirlas.

Nos evitará tener que implementar un proceso de validación previo por cada nuevo dato a agregar.

Primary Key: AUTOINCREMENT



Genera un número incremental consecutivo en el campo definido como **Primary Key**. Inicia su numeración en **1**, y va de forma incremental por cada nuevo registro añadido.

Primary Key: AUTOINCREMENT

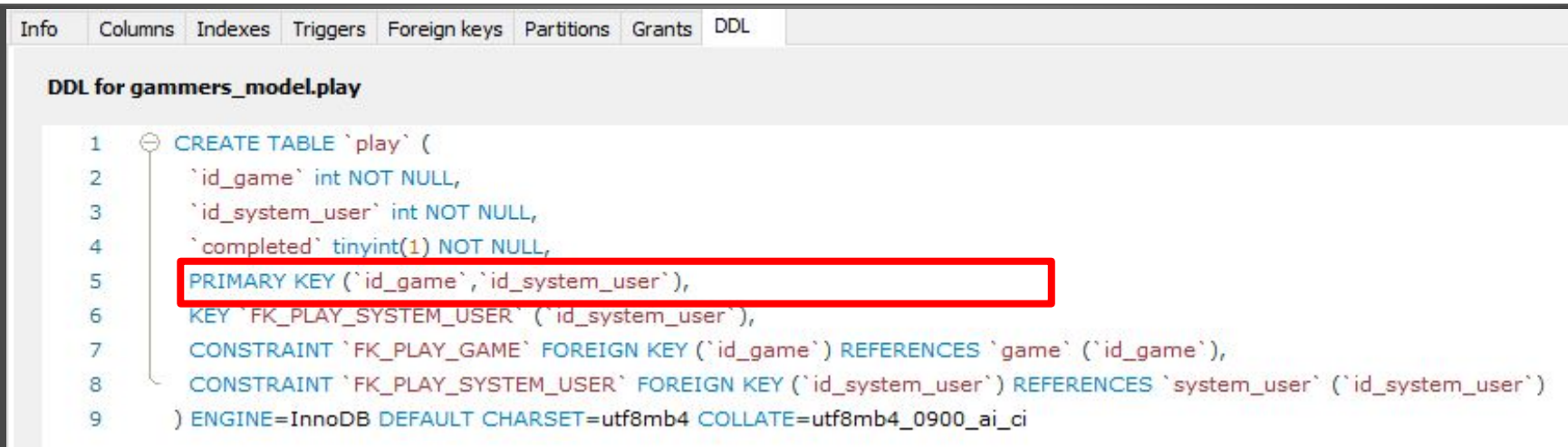
Aún cuando eliminamos un registro de la tabla, al agregar el siguiente SQL respetará el número consecutivo que le correspondía, tal como si el anterior aún existiese.

	id_system_user	first_name	last_name	email
▶	1	Tyson	Besse	tbesse0@aboutads.info
	2	Tam	Fransewich	tfransewich1@who.int
	3	Rosamund	Barcroft	rbarcroft2@furl.net
	4	Bent	Husher	bhusher3@cbc.ca
	5	Averell	Alliot	aalliot4@opensource.org
	6	Aurora	Hannabus	ahannabus5@uiuc.edu
	7	Somerset	Fairlamb	sfairlamb6@webnode.com
	8	Victor	Maughan	vmaughan7@bloglovin.com
	9	Coletta	Winkless	cwinkless8@narod.ru
	10	Carce	Petracco	cpetracco9@flickr.com

VEAMOS UN EJEMPLO DE CÓDIGO

Primary Key: ejemplo

Editamos la tabla **play** a través de **Mysql Workbench > Table Inspector > pestaña DDL**. En el código SQL identificamos la columna **id_video_game**, definida como **Primary Key**

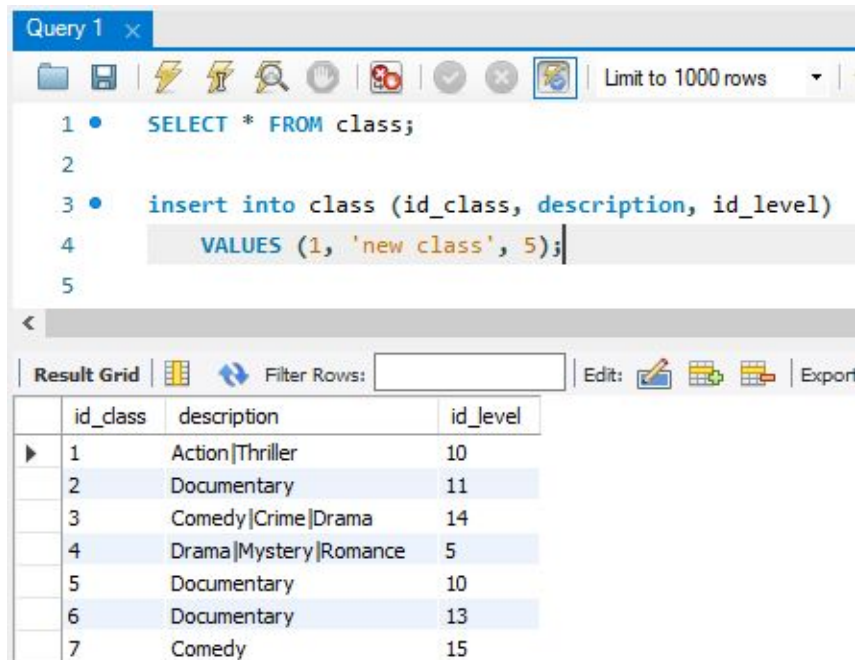


```
DDL for gammers_model.play

1 CREATE TABLE `play` (
2   `id_game` int NOT NULL,
3   `id_system_user` int NOT NULL,
4   `completed` tinyint(1) NOT NULL,
5   PRIMARY KEY (`id_game`, `id_system_user`),
6   KEY `FK_PLAY_SYSTEM_USER` (`id_system_user`),
7   CONSTRAINT `FK_PLAY_GAME` FOREIGN KEY (`id_game`) REFERENCES `game` (`id_game`),
8   CONSTRAINT `FK_PLAY_SYSTEM_USER` FOREIGN KEY (`id_system_user`) REFERENCES `system_user` (`id_system_user`)
9 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Primary Key: ejemplo

Listemos los datos de la tabla **class**, y
luego intentemos agregar un nuevo
registro con el identificador de **class**
igual a 1.



The screenshot shows a SQL query editor window titled "Query 1". The query consists of two statements: a SELECT statement to retrieve all data from the 'class' table, and an INSERT statement to add a new record with id_class 1, description 'new class', and id_level 5. Below the query, the "Result Grid" shows the current data in the 'class' table, which includes 7 rows with columns id_class, description, and id_level.

```
1 • SELECT * FROM class;  
2  
3 • insert into class (id_class, description, id_level)  
4   VALUES (1, 'new class', 5);  
5
```

	id_class	description	id_level
▶	1	Action Thriller	10
	2	Documentary	11
	3	Comedy Crime Drama	14
	4	Drama Mystery Romance	5
	5	Documentary	10
	6	Documentary	13
	7	Comedy	15

Primary Key: ejemplo

Nos toparemos con un **error**, dado que la clave primaria está definida como única, y por ello no puede repetirse algún número en ningún otro registro que se quiera agregar.

Output			
Action Output			
#	Time	Action	Duration / Fetch
1	09:36:44	insert into class (id_class, description, id_level) VALUES (1, 'new class', 5)	0.000 sec
Error Code: 1062. Duplicate entry '1' for key 'class.PRIMARY'			

CLAVES FORÁNEAS

CODER HOUSE

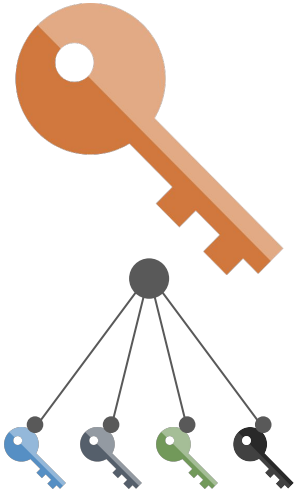
Claves foráneas: Foreign Key



La clave foránea establece una **limitación referencial entre dos tablas**, identificando un campo en una tabla que se referencia con el campo de otra.

Una oficia como **tabla Padre** (*o referenciada*), mientras que la otra será la **tabla Hija** (*o referendo*).

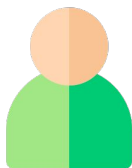
Claves foráneas: Foreign Key



El campo de la tabla Padre siempre es la **Clave Primaria**, y los registros de la tabla Hija no pueden tener valores que no existan en la tabla Padre.

Esto ocurre porque las referencias se crean para relacionar información, siendo ésta una **parte esencial de la Normalización de Base de Datos**.

Claves foráneas: condiciones



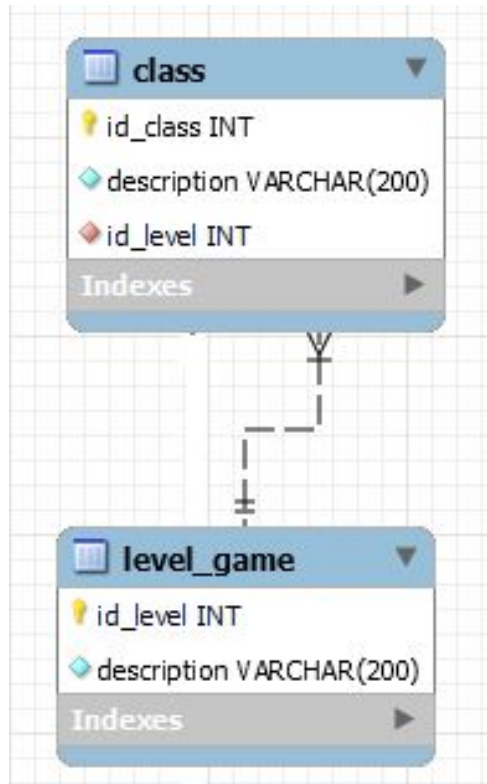
Para utilizar esta vinculación entre dos tablas, debemos:

- Definir **Foreign Key** al crear la tabla desde el principio.
- Las tablas relacionadas deben ser del tipo **InnoDB**.
- La tabla Padre debe crearse primero, con su **Primary Key**.
- La tabla Hija debe referenciar **Foreign Key** a la tabla Padre.
- Ambos campos clave deben ser del mismo tipo de dato.

Claves foráneas: ejemplo

La Tabla Padre (**level_game**) posee la **Clave Primaria**, y se relaciona con la Tabla Hija (**class**) mediante el campo **id_level**, siendo este la **Clave Foránea**.

¿Se imaginan la tabla **class** teniendo
campos sin relacionar?

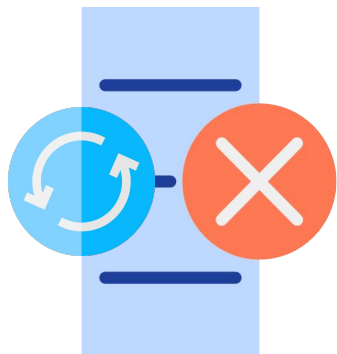


CONDICIONALES EN FOREIGN KEY

Claves foráneas: restricciones

En este tipo de relación de datos debemos prestar atención a eventos clave, como son las actualizaciones/eliminaciones de registros, o la inserción de nueva información.

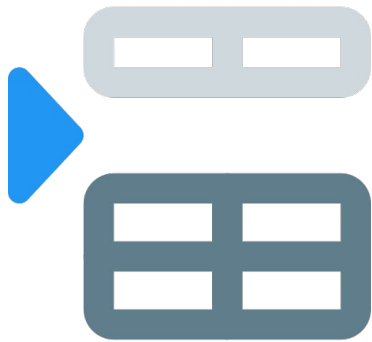
Restricciones ON UPDATE y ON DELETE



Existen sub-cláusulas denominadas **ON UPDATE** y **ON DELETE**, que se deben crear junto a la cláusula Foreign Key. Esto actuará en el momento en que un usuario desee eliminar o actualizar datos de la tabla padre.

Los tipos de respuestas que estas accionarán pueden ser **CASCADE, NO ACTION, RESTRICT.**

Restricciones ON INSERT



Se rechaza cualquier operación INSERT en una **Tabla Hija**, si no existe el valor de la clave foránea en la **Tabla Padre**.

La operación correcta deberá ser: *insertar un dato en la Tabla Padre, y luego insertar el o los datos asociados en la Tabla Hija.*

CLAVES CANDIDATAS

CODER HOUSE

Claves candidatas: definición

Una clave candidata es un conjunto de atributos que identifican de manera unívoca cada tupla (*fila*) de una tabla SQL. La conforma un conjunto de columnas cuyos valores no se repiten en ninguna otra fila de dicha tabla.

Claves candidatas: definición



En una tabla que almacena datos de **personas**, encontraremos seguramente dos o más con **nombre** y **apellido** coincidente. Si deseamos que el conjunto de datos de un registro no se repita, podemos optar por un tercer dato que conforme la **clave candidata**.

Claves candidatas: definición



Por ejemplo, combinando **nombre** + **apellido** + **documento**, tendríamos una clave candidata que haga el registro; único.

- *¡Pero se puede duplicar un documento!*

Sí, es verdad. Para esto podríamos elegir, en su lugar, la dirección de **correo electrónico** o su **teléfono móvil**.

CLAVES CONCATENADAS

Claves concatenadas: definición



Una **clave concatenada**, también conocida como **clave compuesta**, es una clave primaria formada por más de un campo. Se usa en situaciones donde una clave primaria no satisface la necesidad de identificar unívocamente un registro o situación.

Claves concatenadas: implementación



Un caso óptimo sería un estacionamiento vehicular, donde guardamos nuestro vehículo de forma frecuente.

Si bien **patente** podría definir una clave única, si ingresamos varias veces en un día la tabla de registro de ingreso romperá su exclusividad en el registro.

Claves concatenadas: SQL

Su implementación en una tabla SQL se realiza a través del atributo **PRIMARY KEY**, combinando en éste el nombre de cada columna, separado por una coma.



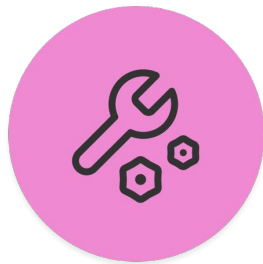
The screenshot displays a database management interface with a tabbed menu at the top: Info, Columns, Indexes, Triggers, Foreign keys, Partitions, Grants, and DDL. The 'DDL' tab is selected, showing the 'DDL for gammers_model.play'.

The SQL code for creating the 'play' table is shown in a text editor with line numbers 1 through 9. The code is as follows:

```
1 CREATE TABLE `play` (  
2   `id_game` int NOT NULL,  
3   `id_system_user` int NOT NULL,  
4   `completed` tinyint(1) NOT NULL,  
5   PRIMARY KEY (`id_game`, `id_system_user`),  
6   KEY `FK_PLAY_SYSTEM_USER` (`id_system_user`),  
7   CONSTRAINT `FK_PLAY_GAME` FOREIGN KEY (`id_game`) REFERENCES `game` (`id_game`),  
8   CONSTRAINT `FK_PLAY_SYSTEM_USER` FOREIGN KEY (`id_system_user`) REFERENCES `system_user` (`id_system_user`)  
9 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

The line containing the **PRIMARY KEY** definition is highlighted with a red rectangular box.

To the right of the code editor, a sidebar for the 'play' table is visible. It lists the columns: 'id_game' (INT), 'id_system_user' (INT), and 'completed' (TINYINT(1)). Below the columns, there is an 'Indexes' section with a right-pointing arrow.



MODIFICAR ESTRUCTURA DE TABLAS

Continuamos trabajando con las tablas creadas en la clase 06.

Tiempo estimado: 15 minutos

MODIFICAR ESTRUCTURA DE TABLAS

Desafío
generico



Retomando el diseño de tablas que realizamos en las prácticas de la **Clase 06**, modificar dichas entidades aplicando:

- Claves primarias.
- Claves foráneas.
- Índices.

También debes generar el modelo relacional utilizando **Reverse Engineering**.



DESCRIPCIÓN DE TABLAS

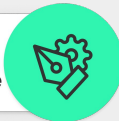
Describiremos las tablas que componen nuestro proyecto final.

DESCRIPCIÓN DE TABLAS

Formato: Archivo en **formato .PDF** nombrado como **"Diagrama+Apellido"**.

Sugerencia: Si lo publican en su Drive Personal, recuerden compartir el acceso.

Desafío
entregable



>> Consigna:

- Armar en formato grilla, los nombres de las tablas de nuestro proyecto final. *(pueden armarlo en una planilla de cálculo y luego exportar su formato a PDF).*

>> Aspectos a incluir en el entregable:

- Definir las tablas de su proyecto
- Incluir una descripción de cada tabla
- Deben incluir los campos que las componen, detallando:
 - claves primarias
 - claves foráneas
 - campos abreviados
 - nombre del campo completo
 - tipos de datos de cada campo

¿PREGUNTAS?



MÁS EJEMPLOS PRÁCTICOS

Material
ampliado



Reverse Engineering con Mysql Workbench

Ejemplos de ingeniería inversa utilizando Mysql Workbench:

- <https://www.youtube.com/watch?v=RMxr5j83dcg>
- <https://www.youtube.com/watch?v=XPvUImHhBhw>

Cuándo usar una clave compuesta o concatenada:

- https://www.youtube.com/watch?v=3CCQm_qslE0

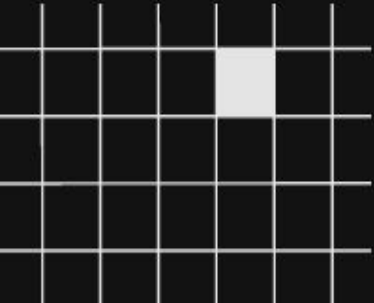
Otro ejemplo de Foreign Key:

- https://www.youtube.com/watch?v=Ja4ImbxDv_0



¡MUCHAS GRACIAS!

Resumen de lo visto en clase hoy:

- Tablas: concepto, Tablas de hecho, transaccionales, dimensionales.
 - Claves: Primarias, Foráneas, Índice
 - claves candidatas, claves concatenadas
- 



OPINA Y VALORA ESTA CLASE

#DEMOCRATIZANDO LA EDUCACIÓN

CODER HOUSE