



# Normalización de la base de datos del proyecto

Link to notion: <https://hill-limburger-3c6.notion.site/Normalizaci-n-de-Proyecto-c16a0627b3864f74a8d635e1c66d81f3>

## Comprobación de normalización

Para asegurarnos de que las tablas están hechas de forma correcta aplicaremos las reglas de normalización, hasta la tercera forma normal.

### 1 **Primera forma**

Para cumplir con la primera forma normal las tablas deben de:

#### **Atomicidad**

🔗 Todos los atributos son atómicos, es decir, son indivisibles. Asimismo, utilizan los tipos de datos permitidos por MySQL.

#### **Level**

En esta tabla se puede observar que ningún elemento es divisible. La dificultad, número total de notas y máximo resultado posible son atómicos.

#### **Match/Game**

Ningún elemento de la tabla puede ser dividido en más secciones. “user\_id” y “level\_id” son llaves foráneas y no se pueden separar. El “score” es un solo número entero atómico. El número correcto de notas y fechas tampoco se puede dividir más.

#### **User**

Al observar esta tabla podemos notar que ninguno de sus atributos se podría dividir. El usuario, contraseña y clase son atributos que no haría sentido separar más. El nivel desbloqueado es un solo número, lo que lo hace lo más atómico posible.

## **Llaves**

🔑 Todas las tablas tienen llaves primarias.

0 La llave primaria no tiene atributos nulos.

## **Level**

Su llave primaria es "level\_number".

Es compuesta de solo 1 atributo y tiene restricción de que no puede ser nulo y autoincremental.

## **Match/Game**

Su llave primaria es "id".

Es compuesta de solo 1 atributo y tiene restricción de que no puede ser nulo y autoincremental.

## **User**

Su llave primaria es "user\_id".

Es compuesta de solo 1 atributo y tiene restricción de que no puede ser nulo y autoincremental.

## **Columnas**

! No puede haber variación en el número de columnas.

## **Level**

Todas las filas tendrán las columnas de "level\_number", "difficulty", "total\_number\_of\_notes" y "max\_possible\_score".

## **Match/Game**

Todas las filas tendrán las columnas de “id”, “user\_id”, “level\_id”, “score”, “correct\_number\_of\_notes” y “date”.

## User

Todas las filas tendrán las columnas de “id”, “class”, “username”, “password”, y “levels\_unlocked”.

## Dependencia



Los campos que no son llave deben identificarse por la llave.

## Level

La dificultad de un nivel, su número total de notas y resultado máximo dependen de que nivel sea.

## Match/Game

Qué usuario juega, qué nivel es, el resultado, el número correcto de notas y la fecha dependen todas de el “match” o “game”.

## User

La clase del usuario, su usuario, contraseña y hasta qué nivel tiene desbloqueado depende del usuario.

## Redundancia



No deben existir grupos de valores repetidos.

## Level

Existen los datos “level\_number”, “difficulty”, “total\_number\_of\_notes”, y “max\_possible\_score” los cuales no se repiten y tienen una columna individual cada uno de ellos.

## Match/Game

Existen los datos “id”, “user\_id”, “level\_id” los cuales todos (semántica y físicamente) identifican diferentes aspectos de la base de datos.

## User

Contiene datos como “class”, “username”, “password”, “id” y “levels\_unlocked” los cuales representan diferentes aspectos.

## Segunda Forma Normal

Para estar en la segunda forma debe cumplir:

### Dependencias funcionales

😞 No deben existir dependencias funcionales parciales. Es decir todos los valores de las columnas de una fila deben depender de la llave primaria.

## Level

Todos los atributos de la tabla dependen de la llave primaria “level\_number”, si el número de nivel cambia, también cambian el número total de notas y la puntuación más alta posible.

## Match/Game

En esta tabla, se utiliza una llave primaria id, de la cual dependen todos los valores de las columnas: “user\_id”, “level\_id”, “score”, “correct\_number\_of\_notes” y “date”.

## User

Todos los atributos de “user” están enlazados a la llave primaria id, en donde cada “id” único representa una combinación de “class”, “username”, “password”, y “levels\_unlocked” diferentes.

### Llaves

🔑 La llave primaria debe ser formada de solo 1 columna que tenga un valor indivisible.

## Level

Está la llave primaria “level\_number” la cual representa un número entero que aumenta, que no se repite y que no se puede dividir ya que solo existe en una columna. Adicionalmente, no es una llave compuesta, es decir, no esta conformada por los valores de las otras columnas de la tabla.

## Match/Game


En “match” o “game” está la llave primaria id la cual es un valor entero que aumenta, no se repite y solo existe en una columna. Adicionalmente, no es una llave compuesta, es decir, no esta conformada por los valores de las otras columnas de la tabla.

## User

En “user” está la llave primaria id la cual es un valor entero que aumenta, no se repite y solo existe en una columna. Adicionalmente, no es una llave compuesta, es decir, no esta conformada por los valores de las otras columnas de la tabla.

## Tercera Forma Normal

Para estar en la tercera forma debe cumplir:

 No deben existir dependencias transitivas entre las columnas de una tabla. Es decir las columnas que no forman parte de la llave primaria deben depender solo de esa llave, nunca de otra columna.

## Level

Tanto “difficulty”, “total\_number\_of\_notes” y “max\_possible\_score” dependen únicamente de la llave primaria “level\_number”. A pesar de que se podría argumentar que la dificultad depende del número total de notas, esto no es del todo cierto, ya que podría haber niveles más difíciles con menos notas. Asimismo, se puede observar que existe algún tipo de dependencia entre “max\_possible\_score” y “total\_number\_of\_notes”. Sin embargo, existen casos en donde esto no aplica del todo. Por ejemplo, si en un futuro se decide optar por que diferentes notas tengan diferentes valores de puntaje, como una nota larga que se debe de dejar presionada.

## Match/Game

No existen dependencias transitivas en la tabla. En primera instancia se podría creer que “score” depende de “correct\_number\_of\_notes”, esto no es cierto ya que la score depende de el multiplicador que el jugador tenga.

## User

No existen dependencias transitivas en la tabla. Se podría pensar que la “password” depende del “username”, sin embargo, dos usuarios pueden tener la misma

contraseña.

## Restricciones de integridad

### Level

- level\_number SMALLINT UNSIGNED NOT NULL AUTO\_INCREMENT,

Esto hace que level\_number siempre sea un número chico (no habrá muchos niveles), no sea nulo y se autoincrementa automáticamente.

- difficulty SMALLINT NOT NULL,

Hace que la dificultad sea un número chico y no pueda ser nulo.

- total\_number\_of\_notes SMALLINT NOT NULL,

Hace que la que el numero total de notas sea un número chico

- max\_possible\_score INT NOT NULL,

Hace que el puntaje máximo sea un número no nulo.

- PRIMARY KEY (level\_number)

Hace irrepetible el número de nivel, lo hace index y referenceable por otras tablas.

### Match/Game

- game\_id INT UNSIGNED NOT NULL AUTO\_INCREMENT,

La restricción de integridad NOT NULL hace que este atributo no sea negable, es decir, siempre requiere de un valor. Adicionalmente, la restricción AUTO\_INCREMENT hace que el atributo game\_id se inicialice automáticamente de manera incremental.

- user\_id INT UNSIGNED NOT NULL,

Estas restricciones se aseguran de que el user\_id sea un entero sin signo y no sea nulo.

- level\_id SMALLINT UNSIGNED NOT NULL,

Estas restricciones se aseguran de que el user\_id sea un entero pequeño sin signo y no sea nulo.

- score INT NOT NULL,

Estas restricciones se aseguran de que el user\_id sea un entero sin signo y no sea nulo.

- correct\_number\_of\_notes SMALLINT NOT NULL,

Estas restricciones se aseguran de que el user\_id sea un entero pequeño sin signo y no sea nulo.

- date\_played TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

Estas restricciones determinan que la fecha en la que se jugó sea de tipo TIMESTAMP y que por defecto ponga la fecha en la que se agrega el dato a la base de datos, esto también ocurre cuando se actualiza el dato.

- PRIMARY KEY (game\_id),

Se determina a game\_id como la llave primaria de la tabla, de esta manera se puede referenciar por otras tablas.

- FOREIGN KEY(user\_id) REFERENCES user(id) ON DELETE RESTRICT ON UPDATE CASCADE,

Se determina a user\_id como llave secundaria y se referencia desde la tabla user. Adicionalmente, se actualiza en cascada en caso de que se cambie su valor en otra de las tablas, tampoco se pueden eliminar datos de esta columna por la restricción de ON DELETE RESTRICT.

- FOREIGN KEY(level\_id) REFERENCES level(level\_number) ON DELETE RESTRICT ON UPDATE CASCADE

Se determina a level\_id como llave secundaria y se referencia desde la tabla level. Adicionalmente, se actualiza en cascada en caso de que se cambie su valor en otra de las tablas, tampoco se pueden eliminar datos de esta columna por la restricción de ON DELETE RESTRICT.

## User

- id INT UNSIGNED NOT NULL AUTO\_INCREMENT,

Hace que el id sea un número, no nulo que se autoincrementa solo.

- class VARCHAR(20) NOT NULL,

Lo hace una palabra de máximo 20 caracteres.

- username VARCHAR(20) NOT NULL,

Lo hace una palabra de máximo 20 caracteres.

- pwd VARCHAR(20) NOT NULL,

Lo hace una palabra de máximo 20 caracteres.

- levels\_unlocked INT NOT NULL,

Hace que los niveles que ha desbloqueado sea un numero no nulo.

- PRIMARY KEY (id),

Hace irrepetible el id lo hace index y referenceable por otras tablas.

- INDEX idx\_class (class)

Hace más rápido el acceso de la clase en la que esta la persona.