**Writing Game**

**Mélisandre Schofield e2395207**

<https://la-fin.org/writing-game>

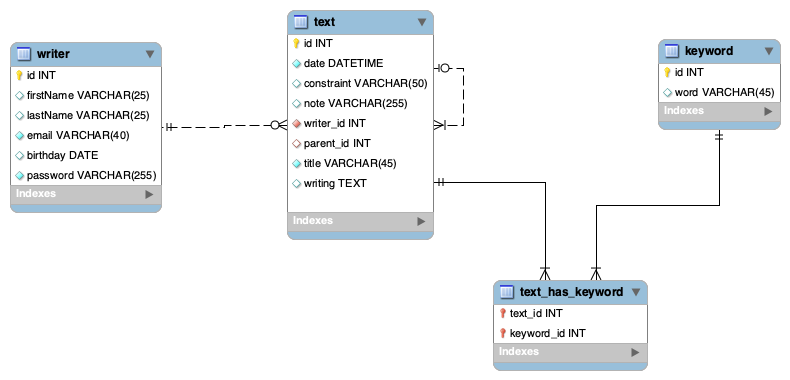
https://github.com/melisandres/writing-game

**a propos**

Les utilisateurs peuvent ecrire un court text, ou ils peuvent itérer sur le text d’un autre utilisateur. S’il choisisent d’itérer, ceci crée un nouveau text. Ce nouveau text garde une relation avec le text précédent. Les texts peuvent être itérés a l’infini. Éventuellement, il y aurrait la possibilité d’ajouter des contraintes ou des notes à une itération pour ses “enfants”, une façon pour un utilisateur de proposer un challenge aux autres utilisateures. J’imagine aussi un system où un utilisateur peut proposer un text à itérer à un autre utilisateur. Et bien sure, je voudrais éventuelement montrer les relations entre les texts dans la facon de les afficher.

Due a un manque de temp, les requetes fournis ne sont pas tous optimales, je ne préviens pas toujours les injections SQLs, et il y a un petit bug qui ajoute parfois un keyword vide a un text. Aussi, le code est un peu pêle-mêle. Idéalement, j’aurais le temp de le netoyer.

**diagram**



**requêtes SQL pour bâtir la bd**

-- -----------------------------------------------------

-- Table `mcandide\_tag`.`writer`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `mcandide\_tag`.`writer` ;

CREATE TABLE IF NOT EXISTS `mcandide\_tag`.`writer` (

`id` INT NOT NULL AUTO\_INCREMENT,

`firstName` VARCHAR(25) NULL,

`lastName` VARCHAR(25) NULL,

`email` VARCHAR(40) NOT NULL,

`birthday` DATE NULL,

`password` VARCHAR(255) NOT NULL,

PRIMARY KEY (`id`),

UNIQUE INDEX `email\_UNIQUE` (`email` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `mcandide\_tag`.`text`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `mcandide\_tag`.`text` ;

CREATE TABLE IF NOT EXISTS `mcandide\_tag`.`text` (

`id` INT NOT NULL AUTO\_INCREMENT,

`title` VARCHAR(45) NULL,

`date` DATETIME NOT NULL,

`constraint` VARCHAR(50) NULL,

`note` VARCHAR(255) NULL,

`writing` TEXT NULL,

`writer\_id` INT NOT NULL,

`parent\_id` INT NULL,

PRIMARY KEY (`id`),

INDEX `fk\_text\_writer\_idx` (`writer\_id` ASC),

INDEX `fk\_text\_text1\_idx` (`parent\_id` ASC),

CONSTRAINT `fk\_text\_writer`

FOREIGN KEY (`writer\_id`)

REFERENCES `mcandide\_tag`.`writer` (`id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_text\_text1`

FOREIGN KEY (`parent\_id`)

REFERENCES `mcandide\_tag`.`text` (`id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `mcandide\_tag`.`keyword`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `mcandide\_tag`.`keyword` ;

CREATE TABLE IF NOT EXISTS `mcandide\_tag`.`keyword` (

`id` INT NOT NULL AUTO\_INCREMENT,

`word` VARCHAR(30) NULL,

PRIMARY KEY (`id`),

UNIQUE INDEX `word\_UNIQUE` (`word` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `mcandide\_tag`.`text\_has\_keyword`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `mcandide\_tag`.`text\_has\_keyword` ;

CREATE TABLE IF NOT EXISTS `mcandide\_tag`.`text\_has\_keyword` (

`text\_id` INT NOT NULL,

`keyword\_id` INT NOT NULL,

PRIMARY KEY (`text\_id`, `keyword\_id`),

INDEX `fk\_text\_has\_keyword\_keyword1\_idx` (`keyword\_id` ASC),

INDEX `fk\_text\_has\_keyword\_text1\_idx` (`text\_id` ASC),

CONSTRAINT `fk\_text\_has\_keyword\_text1`

FOREIGN KEY (`text\_id`)

REFERENCES `mcandide\_tag`.`text` (`id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_text\_has\_keyword\_keyword1`

FOREIGN KEY (`keyword\_id`)

REFERENCES `mcandide\_tag`.`keyword` (`id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;