**Daniel Nugent – 18326304**

**Information Management II – Database SQL Report**

**Description**

I decided to model my SQL database based on the online chat forum Reddit. Reddit is an “anonymous” chat forum where users can post and comment on various different topics. Posts are posted in various different “subreddits” which essentially categorizes posts into different topics. Subreddits are reachable via adding “r/${subreddit\_name}” after the base Reddit URL. Examples of subreddits include “politics”, “pics”, “xbox”, “gaming” and so on. Users can comment on posts, upvote or downvote posts (essentially rating them), or even give an award to the post with gold. Different awards cost different amounts of gold and have different pictures to represent them. Users can comment in reply to another comment, upvote/downvote other comments and also give awards to them. Users can follow other users. Users also have an avatar picture to show off their personality. It can be customized on the reddit site. Users get “karma” when their posts and comments are upvoted more than they are downvoted. I.e. a comment gets 20 upvotes and 3 downvotes, the user gets 17 comment karma. This can be viewed by other users to see what users have more popular comments or to discern legit users from trolls.

I made 7 different tables. They are user, subreddit, user\_follower, comment, post, award and award\_reference. This encompassed the main core features of reddit without some of the new more complicated features that were added recently such as livestreaming which would have in itself added several tables to model it properly. Most attributes are set to not null unless specified otherwise.

The user table had seven attributes, the user\_id, username, comment\_karma, post\_karma, followers (number of followers they have), avatar\_url and cake day (what Reddit calls the day your account was created). I decided to drop the followers attribute because we can get how many followers the user has by looking in the user\_follower table to find out (using the SQL COUNT keyword). The primary key is user\_id. It has no foreign keys.

The user\_follower table is used to represent the one directional following of a user by another user. i.e. if one user follows another, it doesn’t mean the person being followed also follows the user. It has four attributes, id, date, user\_id and follower\_id. The primary key is id and the foreign keys are user\_id and follower\_id.

The comment table has eight attributes, comment\_id, text, upvotes, downvotes, parent\_comment\_id, parent\_post\_id, author\_id and date. The parent\_comment\_id and parent\_post\_id have a constraint which means one of them must not be null. (a comment can be in reply to a post or another comment). The primary key is comment\_id and the foreign keys are parent\_comment\_id, parent\_post\_id and author\_id.

The post table has eight attributes, post\_id, title, body, upvotes, downvotes, subreddit\_id, author\_id and date. The parent\_comment\_id and parent\_post\_id have a constraint which means one of them must not be null. (a comment can be in reply to a post or another comment). The primary key is post\_id and the foreign keys are subreddit\_id and author\_id.

The subreddit table has five attributes, subreddit\_id, handle, followers, title, online. The handle refers to the URL parameter to reach the subreddit. i.e. “r/${handle}”. The title is the current title of the subreddit. Followers refers to the number of people following the subreddit. The primary key is subreddit\_id.

The award table has six attributes, id, type\_id, giver\_id, comment\_id, post\_id and date. The type\_id refers to what type of award it is (gold, silver etc). The giver\_id is the id of the user which gave the award. There is a constraint on comment\_id and post\_id that one must be null and the other must not be null, as a award can be given to a comment or a post. The primary key is id and the foreign keys are type\_id, giver\_id, comment\_id and post\_id.

The award\_reference table is used to reference an award\_id to the award which pertains to it. It has four attributes, award\_id, name, gold and image\_url. The primary key is award\_id and it has no foreign keys.

**Entity relation diagram**

**Diagram, schematic

Description automatically generated**

**Mapping to relational schema**

**A picture containing text, computer, indoor, computer

Description automatically generated**

**Functional dependencies**

Underlines are primary keys, highlighted are foreign keys

**A picture containing timeline

Description automatically generated**



**Semantic Contraints**

Every primary key in the database is set to NOT NULL and to AUTO INCREMENT so it automatically gets assigned an ID. This is important so that it can uniquely identify a row in the table and to avoid integrity constraint violations, thus creating an entity integrity constraint. Foreign keys are also set to NOT NULL exempt in the case of comment and award table as an award can be given to a comment or post and a comment can be made in response to another comment or a post. For this we put a constraint on the award and comment table that either the comment\_id or post\_id in the award table should be NOT NULL and the other should be NULL and in the comment table we put a constraint on either the parent\_post\_id or parent\_comment\_id being NOT NULL and the other being NULL.

ALTER TABLE `reddit`.`award`

ADD CONSTRAINT CheckOnlyOneColumnIsNull

CHECK

(

( CASE WHEN comment\_id IS NULL THEN 0 ELSE 1 END

+ CASE WHEN post\_id IS NULL THEN 0 ELSE 1 END

) = 1

);

Similarly, for comment

ALTER TABLE `reddit`.`comment`

ADD CONSTRAINT CheckOnlyOneColumnIsNullComment

CHECK

(

( CASE WHEN parent\_comment\_id IS NULL THEN 0 ELSE 1 END

+ CASE WHEN parent\_post\_id IS NULL THEN 0 ELSE 1 END

) = 1

);

We check that a post doesn’t have an empty string as a title

ALTER TABLE `reddit`.`post`

ADD CONSTRAINT title\_not\_empty

CHECK (title != '');

And that all comments have text.

ALTER TABLE `reddit`.`comment`

ADD CONSTRAINT text\_not\_empty

CHECK (text != '');

**Views**

1. I have created two useful views for the database. Even though we store the user karma as separate attributes in comment\_karma and post\_karma, Reddit likes to combine this total into total\_karma for a quick glance at how much karma in total a user has. For this I made a view.

CREATE VIEW TotalUserKarma AS

SELECT user\_id, (comment\_karma + post\_karma) as total\_karma

FROM reddit.user;

1. Another view I made was to reference the details of the award by using the type\_id to join the specified award to an award given to a post or comment. This is useful to shorten the time it takes to get the details of the type of award given. We join the award reference table to the award table of the type\_id of the award referencing to the award\_id on the award\_reference table.

CREATE VIEW AwardInfoJoined AS

SELECT a.id, a.giver\_id, ar.name as award\_name, ar.gold, ar.image\_url, a.comment\_id, a.post\_id, a.date

FROM award a

LEFT JOIN award\_reference ar ON a.type\_id = ar.award\_id;

**Triggers, Update and Deletes**

In most instances we can use cascades to delete rows in tables when its foreign key pointing at a primary key table row gets deleted. So if a post gets deleted, all comments in response to that post gets automatically deleted. This is the normal behaviour in Reddit. If a user deletes his comment or post, all comments in response are also deleted. We need to create triggers to set the upvotes to one after a post or comment is made. This is the user upvoting his own comment/post which is done automatically on Reddit.

To set upvotes to one before inserting the comment in MySQL workbench

DROP TRIGGER IF EXISTS `reddit`.`comment\_BEFORE\_INSERT`;

DELIMITER $$

USE `reddit`$$

CREATE DEFINER = CURRENT\_USER TRIGGER `reddit`.` comment\_BEFORE\_INSERT` BEFORE INSERT ON `comment` FOR EACH ROW

BEGIN

SET NEW.upvotes = 1;

END$$

DELIMITER ;

END

And similarly for posts

DROP TRIGGER IF EXISTS `reddit`.`post\_BEFORE\_INSERT`;

DELIMITER $$

USE `reddit`$$

CREATE DEFINER = CURRENT\_USER TRIGGER `reddit`.`post\_BEFORE\_INSERT` BEFORE INSERT ON `post` FOR EACH ROW

BEGIN

SET NEW.upvotes = 1;

END$$

DELIMITER ;

**Appendix**

**Create**

CREATE TABLE `award` (

`id` int NOT NULL AUTO\_INCREMENT,

`type\_id` int NOT NULL,

`giver\_id` int DEFAULT NULL COMMENT 'can be null if giver wants to remain anonymous',

`comment\_id` int DEFAULT NULL COMMENT 'can be null if award is for post ',

`post\_id` int DEFAULT NULL COMMENT 'can be null if award is for comment',

`date` date NOT NULL,

PRIMARY KEY (`id`),

KEY `award\_ibfk\_2` (`giver\_id`),

KEY `award\_ibfk\_3` (`comment\_id`),

KEY `award\_ibfk\_4` (`post\_id`),

CONSTRAINT `award\_ibfk\_1` FOREIGN KEY (`id`) REFERENCES `award\_reference` (`award\_id`) ON DELETE CASCADE,

CONSTRAINT `award\_ibfk\_2` FOREIGN KEY (`giver\_id`) REFERENCES `user` (`user\_id`) ON DELETE CASCADE,

CONSTRAINT `award\_ibfk\_3` FOREIGN KEY (`comment\_id`) REFERENCES `comment` (`comment\_id`) ON DELETE CASCADE,

CONSTRAINT `award\_ibfk\_4` FOREIGN KEY (`post\_id`) REFERENCES `post` (`post\_id`) ON DELETE CASCADE,

CONSTRAINT `award\_chk\_1` CHECK (((`comment\_id` is not null) or (`post\_id` is not null))),

CONSTRAINT `CheckOnlyOneColumnIsNull` CHECK ((((case when (`comment\_id` is null) then 0 else 1 end) + (case when (`post\_id` is null) then 0 else 1 end)) = 1))) ;

CREATE TABLE `award\_reference` (

`award\_id` int NOT NULL AUTO\_INCREMENT,

`name` varchar(100) NOT NULL,

`gold` int NOT NULL,

`image\_url` varchar(200) NOT NULL,

PRIMARY KEY (`award\_id`),

UNIQUE KEY `award\_UNIQUE` (`name`)

);

CREATE TABLE `comment` (

`comment\_id` int NOT NULL AUTO\_INCREMENT,

`text` varchar(10000) NOT NULL,

`upvotes` int NOT NULL DEFAULT '1',

`downvotes` int DEFAULT '0',

`parent\_comment\_id` int DEFAULT NULL COMMENT 'parent comment id can be NULL if there is no parent to the comment ie the comment is a direct reply to the post',

`parent\_post\_id` int DEFAULT NULL,

`author\_id` int NOT NULL,

`date` date NOT NULL,

PRIMARY KEY (`comment\_id`),

KEY `comment\_ibfk\_1\_idx` (`parent\_comment\_id`),

KEY `comment\_ibfk\_2\_idx` (`parent\_post\_id`),

KEY `comment\_ibfk\_3` (`author\_id`),

CONSTRAINT `comment\_ibfk\_1` FOREIGN KEY (`parent\_comment\_id`) REFERENCES `comment` (`comment\_id`) ON DELETE CASCADE,

CONSTRAINT `comment\_ibfk\_2` FOREIGN KEY (`parent\_post\_id`) REFERENCES `post` (`post\_id`) ON DELETE CASCADE,

CONSTRAINT `comment\_ibfk\_3` FOREIGN KEY (`author\_id`) REFERENCES `user` (`user\_id`) ON DELETE CASCADE,

CONSTRAINT `CheckOnlyOneColumnIsNullComment` CHECK ((((case when (`parent\_comment\_id` is null) then 0 else 1 end) + (case when (`parent\_post\_id` is null) then 0 else 1 end)) = 1)),

CONSTRAINT `parent\_not\_null` CHECK (((`parent\_comment\_id` is not null) or (`parent\_post\_id` is not null))),

CONSTRAINT `text\_not\_empty` CHECK ((`text` <> \_utf8mb4''))

);

CREATE TABLE `post` (

`post\_id` int NOT NULL AUTO\_INCREMENT,

`title` varchar(100) NOT NULL,

`body` varchar(10000) NOT NULL DEFAULT '',

`upvotes` int NOT NULL DEFAULT '1',

`downvotes` int NOT NULL DEFAULT '0',

`subreddit\_id` int NOT NULL COMMENT 'Subreddit ID is not required as you can post in a subreddit or on your wall.',

`author\_id` int NOT NULL,

`date` date NOT NULL,

PRIMARY KEY (`post\_id`),

KEY `post\_ibfk\_1` (`subreddit\_id`),

KEY `post\_ibfk\_2` (`author\_id`),

CONSTRAINT `post\_ibfk\_1` FOREIGN KEY (`subreddit\_id`) REFERENCES `subreddit` (`subreddit\_id`),

CONSTRAINT `post\_ibfk\_2` FOREIGN KEY (`author\_id`) REFERENCES `user` (`user\_id`) ON DELETE CASCADE,

CONSTRAINT `title\_not\_empty` CHECK ((`title` <> \_utf8mb4''))

);

CREATE TABLE `subreddit` (

`subreddit\_id` int NOT NULL AUTO\_INCREMENT,

`handle` varchar(100) DEFAULT NULL,

`followers` int DEFAULT NULL,

`title` varchar(100) DEFAULT NULL,

`online` int DEFAULT NULL,

PRIMARY KEY (`subreddit\_id`),

UNIQUE KEY `name\_UNIQUE` (`handle`)

);

CREATE TABLE `user` (

`user\_id` int NOT NULL AUTO\_INCREMENT,

`username` varchar(45) NOT NULL,

`comment\_karma` int NOT NULL DEFAULT '0',

`post\_karma` int NOT NULL DEFAULT '0',

`avatar\_url` varchar(200) NOT NULL DEFAULT 'https://www.reddit.com/',

`cake\_day` date NOT NULL,

PRIMARY KEY (`user\_id`)

);

CREATE TABLE `user\_follower` (

`id` int NOT NULL AUTO\_INCREMENT,

`date` date NOT NULL,

`user\_id` int NOT NULL,

`follower\_id` int NOT NULL,

PRIMARY KEY (`id`),

KEY `follower\_id` (`follower\_id`),

KEY `user\_follower\_ibfk\_1` (`user\_id`),

CONSTRAINT `user\_follower\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `user` (`user\_id`) ON DELETE CASCADE,

CONSTRAINT `user\_follower\_ibfk\_2` FOREIGN KEY (`follower\_id`) REFERENCES `user` (`user\_id`),

CONSTRAINT `Follow\_self` CHECK ((`user\_id` <> `follower\_id`))

);

**Insert**

INSERT INTO `award` VALUES (1,1,4,3,NULL,'2020-08-30'),(2,3,1,NULL,1,'2020-09-02');

INSERT INTO `award\_reference` VALUES (1,'Gold',500,'https://i.redd.it/4osxoc5mp5h21.jpg'),(2,'Silver',100,'https://www.redditstatic.com/desktop2x/img/gold/badges/award-silver-large.png'),(3,'Platinum',1800,'https://www.redditstatic.com/desktop2x/img/gold/badges/award-platinum-large.png');

INSERT INTO `comment` VALUES (1,'this post is very funny',4,0,NULL,1,1,'2020-09-02'),(2,'this comment is very funny!',2,1,1,NULL,2,'2020-09-03'),(3,'I think you should talk to a councellor',40,3,NULL,4,3,'2020-08-30'),(4,'I think this is great advice!',20,1,3,NULL,4,'2020-09-01');

INSERT INTO `post` VALUES (1,'This is a funny post!','A very funny post body',10,3,1,1,'2020-09-01'),(2,'New league of legends champion released!','Yone was released today',6800,390,2,4,'2020-05-10'),(3,'very cool pic','https://verycoolpic.com',46,1,3,3,'2020-07-10'),(4,'My marriage is breaking up','Can someone giive me some advice',100,10,4,2,'2020-08-29');

INSERT INTO `subreddit` VALUES (1,'funny',34000000,'funny content here!',25000),(2,'leagueoflegends',4500000,'League of Doublelift',20400),(3,'pics',26300000,'Reddit Pics',19600),(4,'relationship\_advice',4000000,'Relationship Advice',8200);

INSERT INTO `user` VALUES (1,'randomredditor',100,200,'https://www.google.com','2019-01-12'),(2,'coolcakes',1000,2000,'https://www.google.com/2','2018-03-10'),(3,'redditiscool',69,420,'https://www.google.com/3','2020-01-10'),(4,'throwaway2',1885,13441,'https://picsum.photos/id/237/200/300','2013-05-01');

INSERT INTO `user\_follower` VALUES (6,'2020-02-10',1,2),(8,'2020-04-10',4,2),(9,'2020-06-28',2,1),(10,'2020-03-20',3,2);