1. Start with abstract model
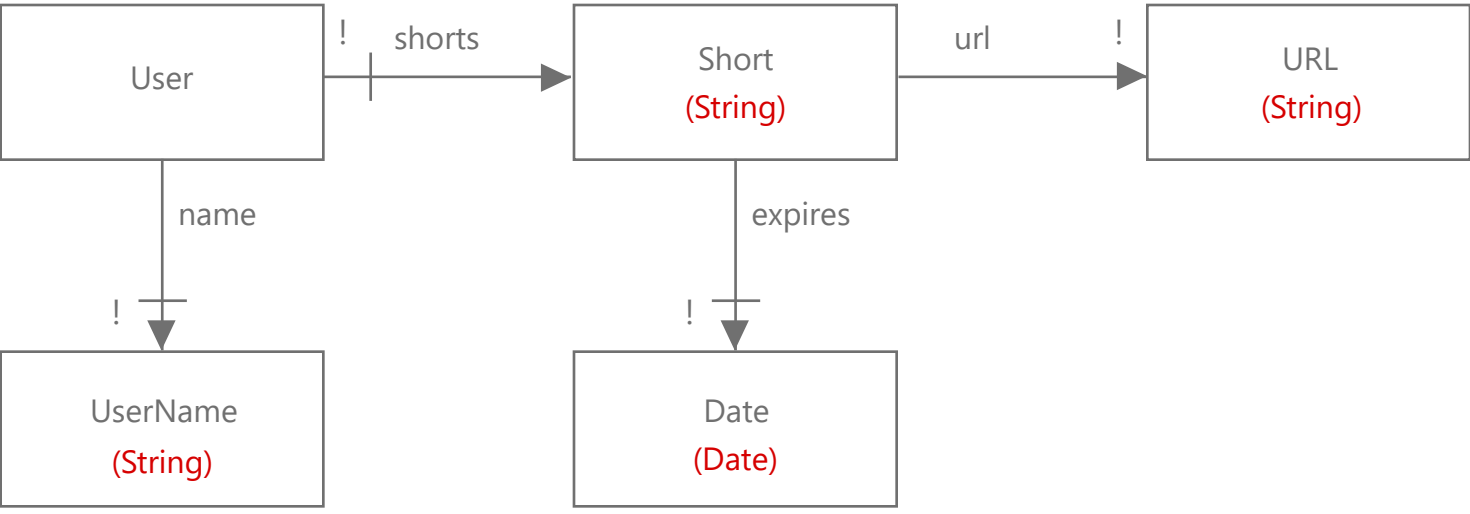
User —! shorts— Short —url—! URL

Short —expires—! Date
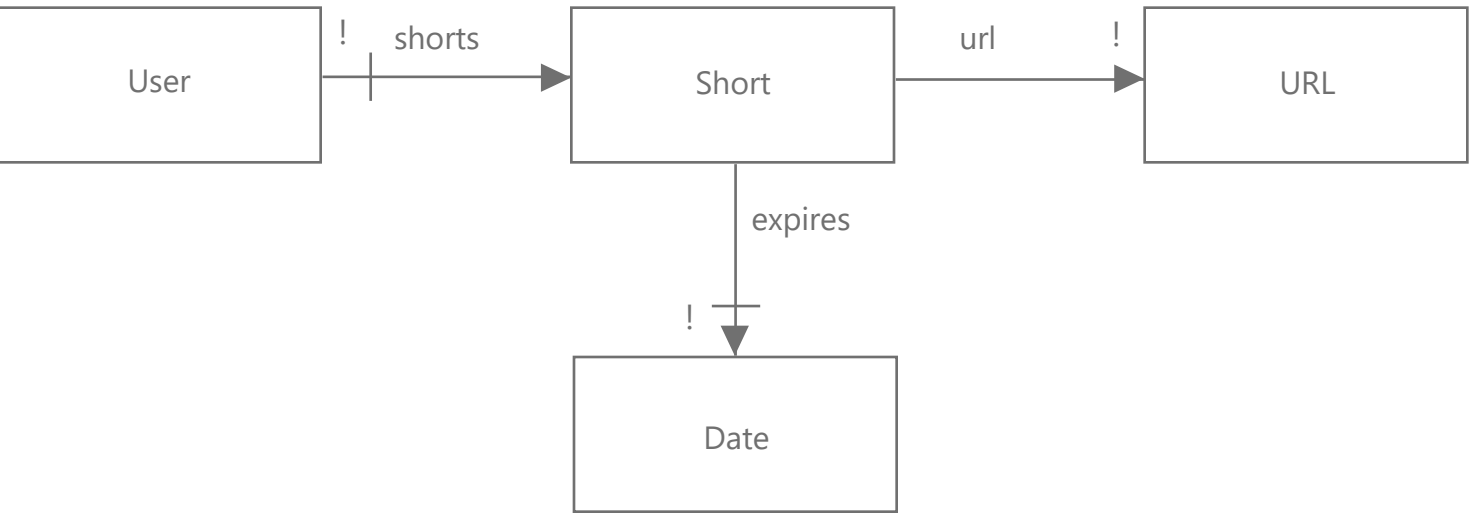
2. Add attributes
 - will it work as just a primitive value?

User —! shorts— Short —url—! URL

User —name—! UserName

Short —expires—! Date

3. Pick primitive types

User —! shorts— Short **(String)** —url—! URL **(String)**

User —name—! UserName **(String)**

Short **(String)** —expires—! Date **(Date)**
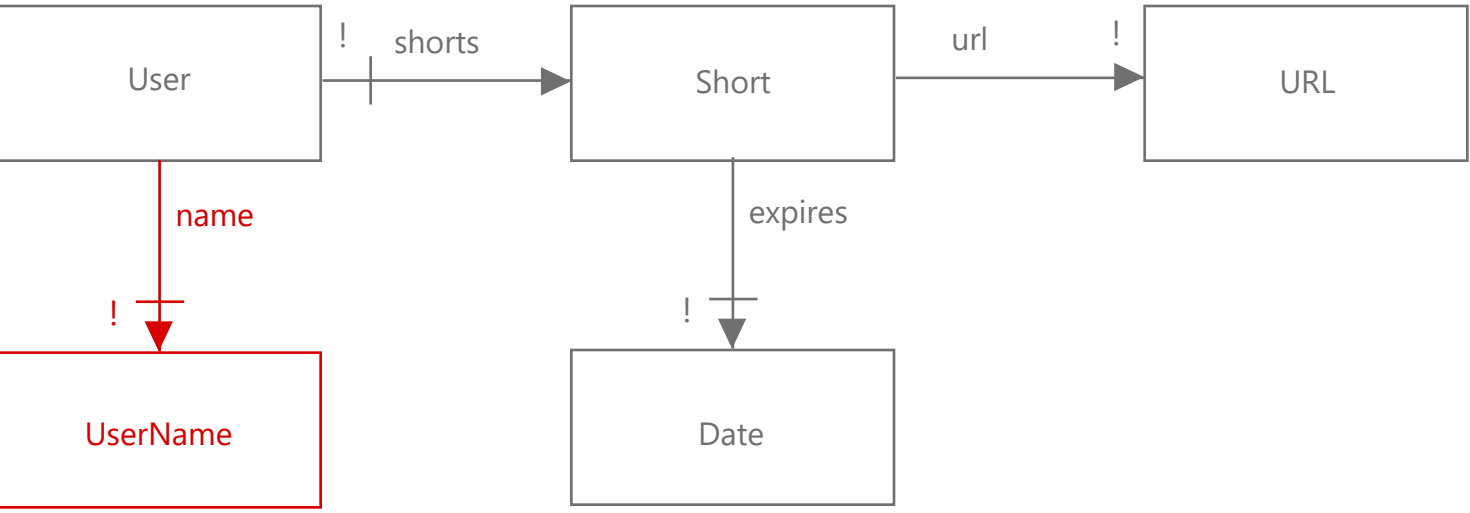
4. Check each relation
 - flip one-to-many relations

User ←! creator— Short **(String)** —url—! URL **(String)**

User —name—! UserName **(String)**

Short **(String)** —expires—! Date **(Date)**
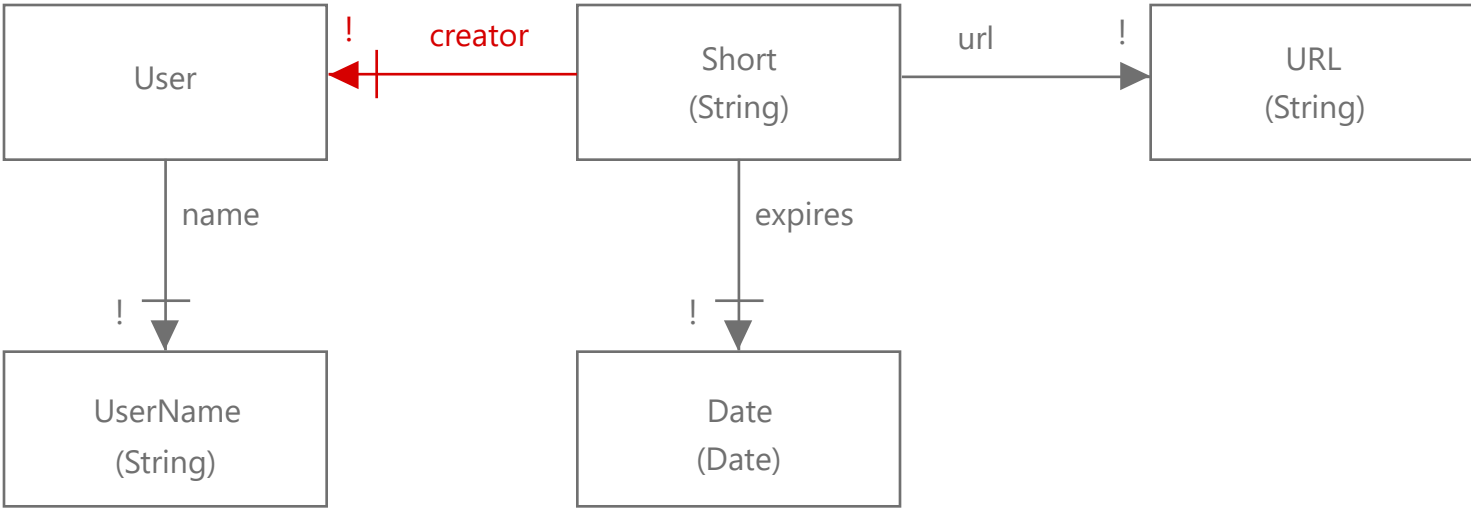
5. Group into tables
 - flip one-to-many relations

User ←! creator— Short **(String)** —url—! URL **(String)**

User —name—! UserName **(String)**

Short **(String)** —expires—! Date **(Date)**
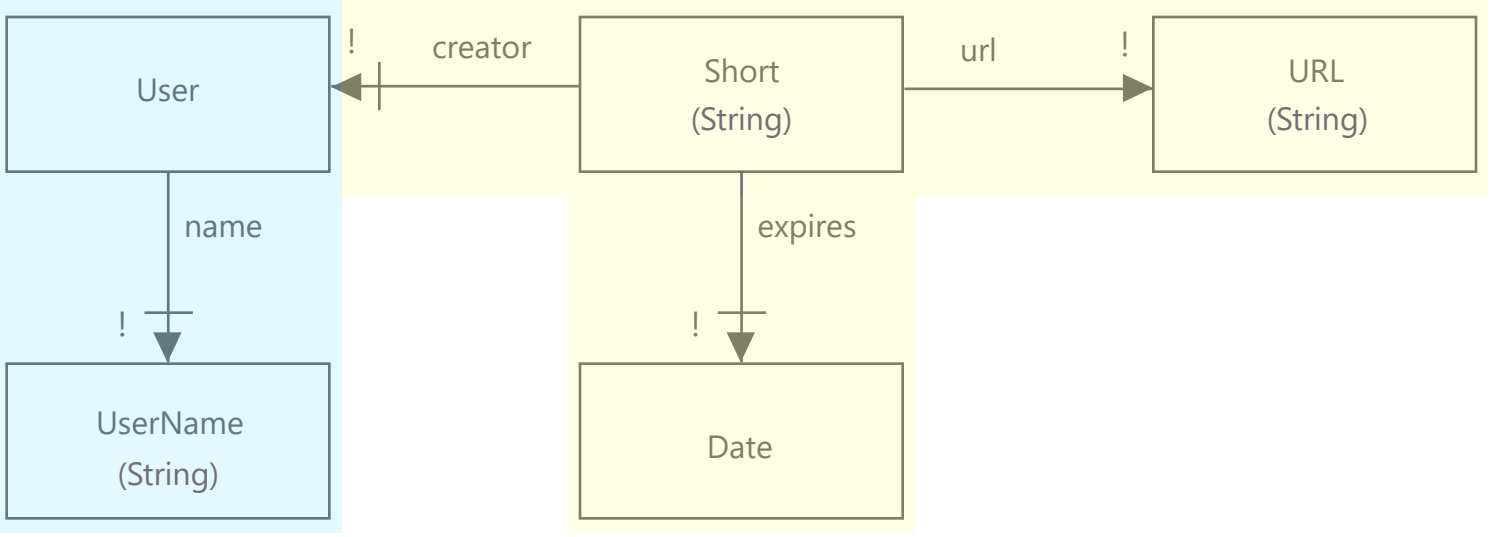
5. Write out tables
 - root becomes key of table
 - relations to other tables become references
 - sets without primitive types become ids

User ←! creator— Short **(String)** —url—! URL **(String)**

User —name—! UserName **(String)**

Short **(String)** —expires—! Date
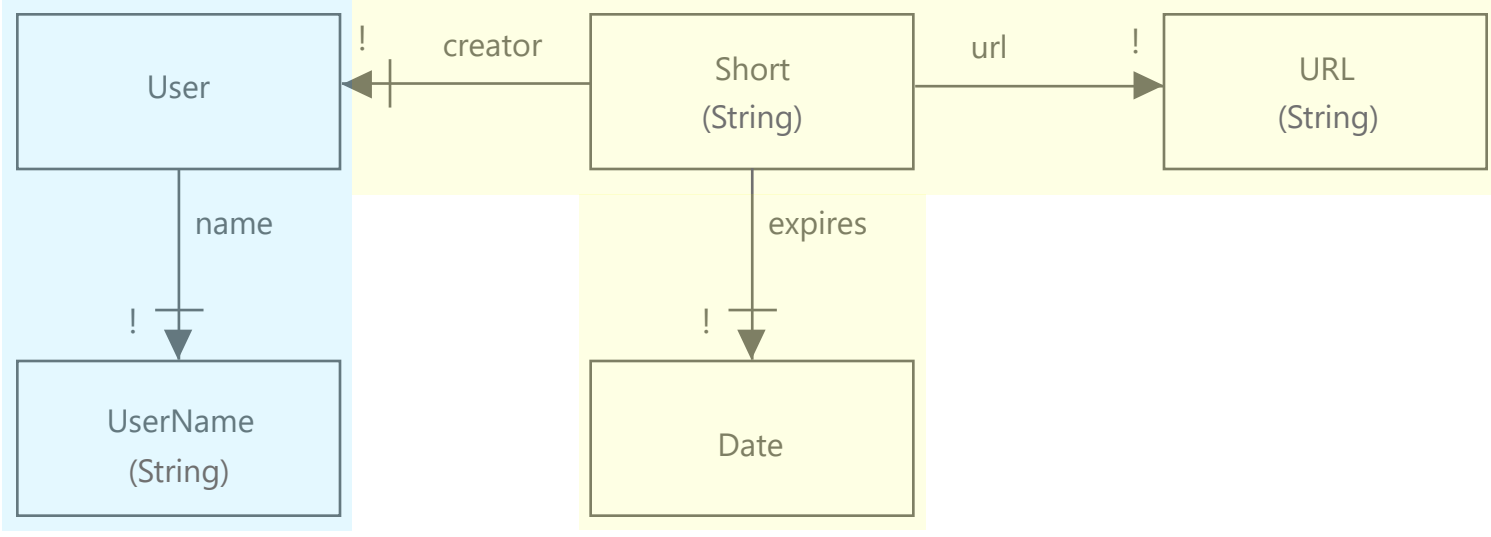
```
CREATE TABLE users (
  id int primary key,
  name text
)
```

```
CREATE TABLE shorts (
  short text primary key,
  url text,
  creator int references users(id)
)
```