

Blog Post Model Hierarchy

User objects can have multiple blog posts and multiple comments

BlogPost objects have one user and can have multiple comments

Comment objects have one user and one blog post

```
class User(UserMixin, db.Model):
    __tablename__ = "users"
    id = db.Column(db.Integer, primary_key=True)
    ...id = db.Column(db.Integer, primary_key=True)...
    email = db.Column(db.String(100), unique=True, nullable=False)
    password = db.Column(db.String(1000), nullable=False)
    name = db.Column(db.String(100), nullable=False)
    posts = db.relationship('BlogPost', back_populates='author')
    comments = db.relationship('Comment', back_populates='commenter')
```

```
class BlogPost(db.Model):
    __tablename__ = "blog_posts"
    id = db.Column(db.Integer, primary_key=True)
    title = db.Column(db.String(250), unique=True, nullable=False)
    subtitle = db.Column(db.String(250), nullable=False)
    date = db.Column(db.String(250), nullable=False)
    body = db.Column(db.Text, nullable=False)
    img_url = db.Column(db.String(250), nullable=False)
    author_id = db.Column(db.Integer, db.ForeignKey('users.id'), nullable=False)
    author = db.relationship('User', back_populates='posts')
    comments = db.relationship('Comment', back_populates='post')
```

```
class Comment(db.Model):
    __tablename__ = "comments"
    id = db.Column(db.Integer, primary_key=True)
    body = db.Column(db.Text, nullable=False)
    date = db.Column(db.String(250), nullable=False)
    post_id = db.Column(db.Integer, db.ForeignKey('blog_posts.id'), nullable=False)
    post = db.relationship('BlogPost', back_populates='comments')
    commenter_id = db.Column(db.Integer, db.ForeignKey('users.id'), nullable=False)
    commenter = db.relationship('User', back_populates='comments')
```

posts
type BlogPost
backpopulates author in BlogPost

id (primary key)

—User object—

Relationship

←BlogPost object

Parent

Child

—Integer—

ForeignKey

author
type User
backpopulates posts in User

author_id
type Integer
ForeignKey in User - users.id

add_new_post()

```
new_post = BlogPost(
    title=form.title.data,
    subtitle=form.subtitle.data,
    body=form.body.data,
    img_url=form.img_url.data,
    author=current_user,
    date=date.today().strftime("%d/%b/%Y")
)
```

comments
type Comment
backpopulates post in Comment

id (primary key)

—BlogPost object—

Relationship

←Comment object

Parent

Child

—Integer—

ForeignKey

post
type BlogPost
backpopulates comments in BlogPost

post_id
type Integer
ForeignKey in BlogPost - blog_posts.id

show_post(post_id)

```
new_comment = Comment(
    body=form.body.data,
    post=requested_post,
    commenter=current_user,
    date=date.today().strftime("%d/%b/%Y"),
)
```

comments
type Comment
backpopulates commenter in Comment

id (primary key)

—User object—

Relationship

←Comment object

Parent

Child

—Integer—

ForeignKey

commenter
type User
backpopulates comments in User

commenter_id
type Integer
ForeignKey in User - user.id