Blog Post Model Hierarchy

User objects can have multiple blog posts and multiple comments

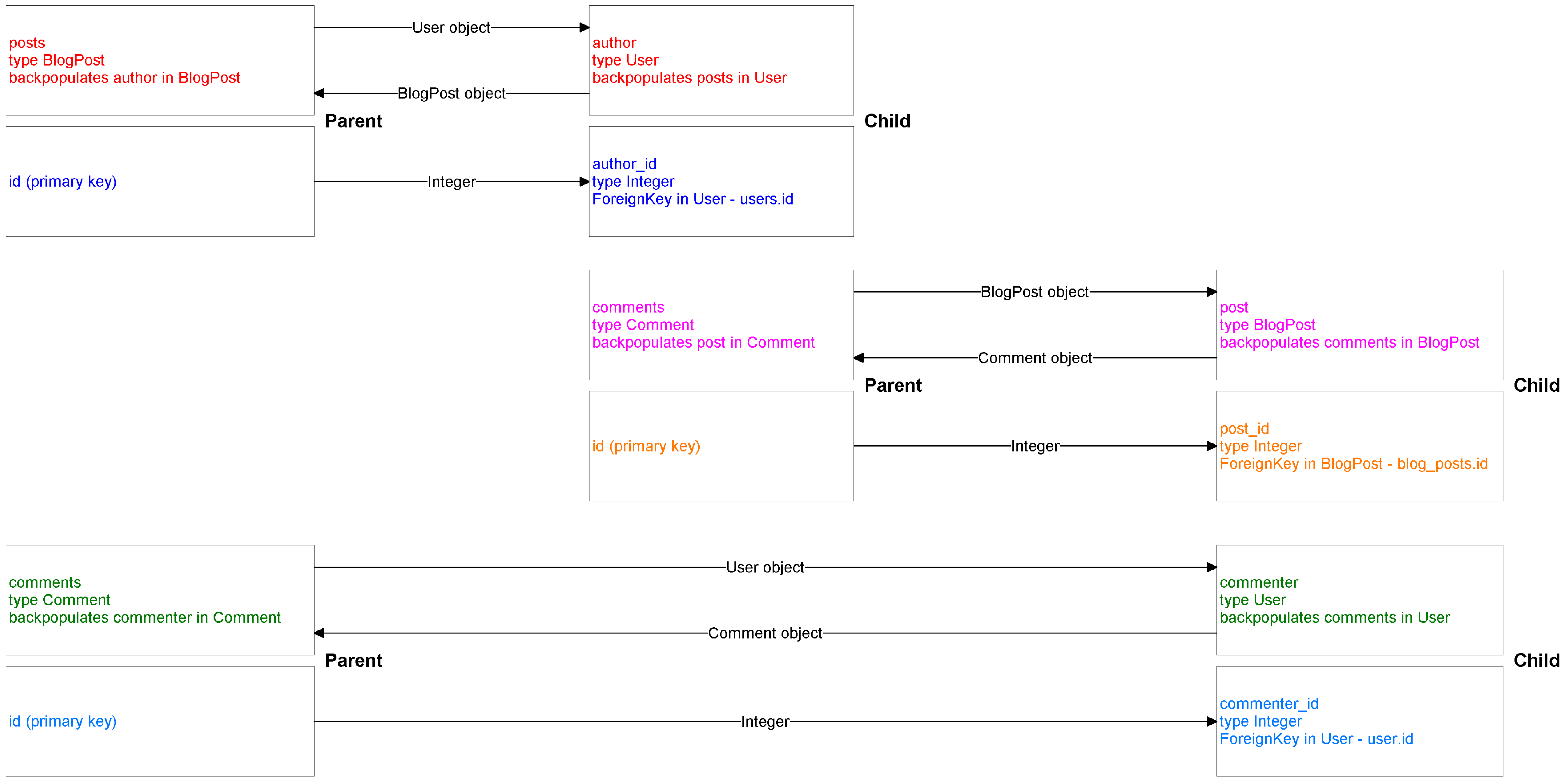
BlogPost objects have one user and can have multiple comments

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
User object
BlogPost object
author
type User
backpopulates posts in User
Parent
Child
id (primary key)
Integer
author_id
type Integer
ForeignKey in User - users.id
comments
type Comment
backpopulates post in Comment
BlogPost object
Comment object
post
type BlogPost
backpopulates comments in BlogPost
Parent
Child
id (primary key)
Integer
post_id
type Integer
ForeignKey in BlogPost - blog_posts.id
comments
type Comment
backpopulates commenter in Comment
User object
Comment object
commenter
type User
backpopulates comments in User
Parent
Child
id (primary key)
Integer
commenter_id
type Integer
ForeignKey in User - user.id