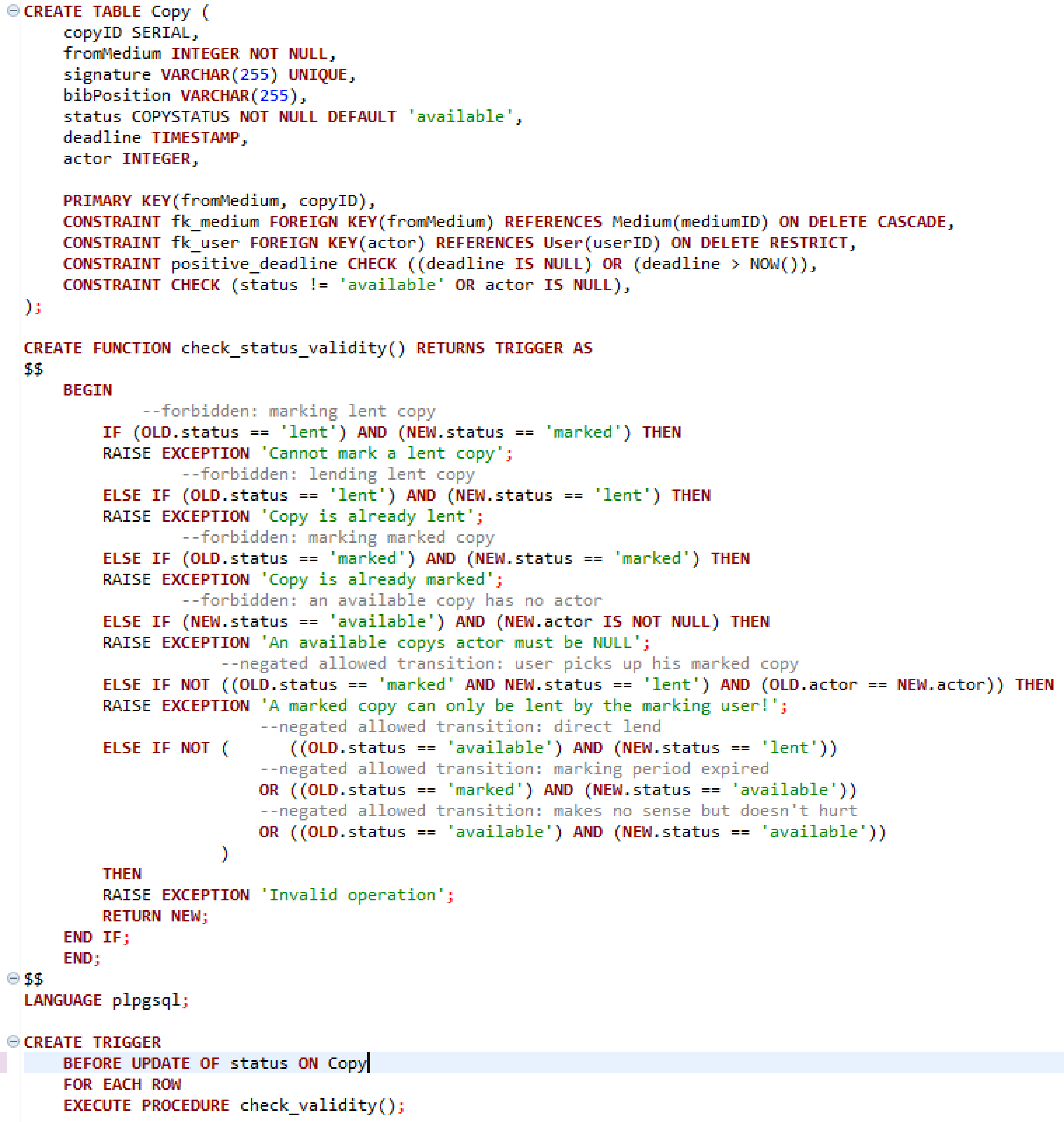

```sql
CREATE TABLE Copy (
    copyID SERIAL,
    fromMedium INTEGER NOT NULL,
    signature VARCHAR(255) UNIQUE,
    bibPosition VARCHAR(255),
    status COPYSTATUS NOT NULL DEFAULT 'available',
    deadline TIMESTAMP,
    actor INTEGER,

    PRIMARY KEY(fromMedium, copyID),
    CONSTRAINT fk_medium FOREIGN KEY(fromMedium) REFERENCES Medium(mediumID) ON DELETE CASCADE,
    CONSTRAINT fk_user FOREIGN KEY(actor) REFERENCES User(userID) ON DELETE RESTRICT,
    CONSTRAINT positive_deadline CHECK ((deadline IS NULL) OR (deadline > NOW()),
    CONSTRAINT CHECK (status != 'available' OR actor IS NULL),
);

CREATE FUNCTION check_status_validity() RETURNS TRIGGER AS
$$
    BEGIN
            --forbidden: marking lent copy
        IF (OLD.status == 'lent') AND (NEW.status == 'marked') THEN
        RAISE EXCEPTION 'Cannot mark a lent copy';
                --forbidden: lending lent copy
        ELSE IF (OLD.status == 'lent') AND (NEW.status == 'lent') THEN
        RAISE EXCEPTION 'Copy is already lent';
                --forbidden: marking marked copy
        ELSE IF (OLD.status == 'marked') AND (NEW.status == 'marked') THEN
        RAISE EXCEPTION 'Copy is already marked';
                --forbidden: an available copy has no actor
        ELSE IF (NEW.status == 'available') AND (NEW.actor IS NOT NULL) THEN
        RAISE EXCEPTION 'An available copys actor must be NULL';
                    --negated allowed transition: user picks up his marked copy
        ELSE IF NOT ((OLD.status == 'marked' AND NEW.status == 'lent') AND (OLD.actor == NEW.actor)) THEN
        RAISE EXCEPTION 'A marked copy can only be lent by the marking user!';
                        --negated allowed transition: direct lend
        ELSE IF NOT (      ((OLD.status == 'available') AND (NEW.status == 'lent'))
                        --negated allowed transition: marking period expired
                        OR ((OLD.status == 'marked') AND (NEW.status == 'available'))
                        --negated allowed transition: makes no sense but doesn't hurt
                        OR ((OLD.status == 'available') AND (NEW.status == 'available'))
                    )
        THEN
        RAISE EXCEPTION 'Invalid operation';
        RETURN NEW;
    END IF;
    END;
$$
LANGUAGE plpgsql;

CREATE TRIGGER
    BEFORE UPDATE OF status ON Copy
    FOR EACH ROW
    EXECUTE PROCEDURE check_validity();
```