```
SET search_path = 'Project Hotel';

/*reserved roomIDs*/
CREATE OR REPLACE FUNCTION roomIDoccu (DBEG timestamp, DEND timestamp, HID int)
RETURNS TABLE (roomID int) AS $$
	SELECT roomID FROM "Project Hotel".ReservationCurrent
	WHERE (HID = hotelID OR HID = -1) AND (DEND >= dateBeg AND DBEG < dateEnd)
$$ LANGUAGE SQL;

/*available (unreserved) roomIDs*/
CREATE OR REPLACE FUNCTION roomIDfree (DBEG timestamp, DEND timestamp, HID int)
RETURNS TABLE (roomID int) AS $$
	SELECT roomID FROM "Project Hotel".Room
	WHERE (HID = hotelID OR HID = -1) AND roomID NOT IN (SELECT roomIDoccu(DBEG, DEND, HID))
$$ LANGUAGE SQL;

/*make resCur into resPast*/
CREATE OR REPLACE FUNCTION construct_respast (UID int, BNUM int) RETURNS int AS $$
DECLARE
	resInfo record;
	roomInfo record;
BEGIN
	SELECT roomID, hotelID, isRented, dateBeg, dateEnd, occupants INTO resInfo FROM "Project Hotel".ReservationCurrent
	WHERE (UID = userID AND BNUM = bookingNum);
	SELECT capacity, extendable, view, amenities, price INTO roomInfo FROM "Project Hotel".Room
	WHERE (roomID = resInfo.roomID);

	INSERT INTO "Project Hotel".ReservationPast(userID, bookingNum, hotelID, roomID, isRented, dateBeg, dateEnd,
	occupants, capacity, extendable, view, amenities, price)
	VALUES (
		UID,BNUM,
		resInfo.hotelID, resInfo.roomID, resInfo.isRented, resInfo.dateBeg, resInfo.dateEnd, resInfo.occupants,
		roomInfo.capacity, roomInfo.extendable, roomInfo.view, roomInfo.amenities, roomInfo.price
	);
	RETURN 1;
END; $$ LANGUAGE plpgsql;

/*find expired reservations, make resPast, then remove resCur*/
CREATE OR REPLACE FUNCTION check_rescur_expiry() RETURNS int AS $$
DECLARE
	UID int;
	BNUM int;
BEGIN
	FOR UID,BNUM IN
		SELECT userID,bookingNum FROM "Project Hotel".ReservationCurrent
		WHERE (dateEnd < CURRENT_TIMESTAMP) AND (paymentDue <= 0)
	LOOP
		PERFORM construct_respast(UID,BNUM);
		DELETE FROM "Project Hotel".ReservationCurrent WHERE (UID = userID AND BNUM = bookingNum);
	END LOOP;
	RETURN 2;
END; $$ LANGUAGE plpgsql;
```

```
SET search_path = 'Project Hotel';

/*Update hotel and room counts*/

CREATE OR REPLACE FUNCTION hotelcount_add() RETURNS trigger AS $$BEGIN
UPDATE "Project Hotel".Brand SET hotelCount=hotelCount+1 WHERE (NEW.brandName = brandName); RETURN NEW; END;$$ LANGUAGE plpgsql;
CREATE OR REPLACE FUNCTION hotelcount_del() RETURNS trigger AS $$BEGIN
UPDATE "Project Hotel".Brand SET hotelCount=hotelCount-1 WHERE (OLD.brandName = brandName); RETURN OLD; END;$$ LANGUAGE plpgsql;

DROP TRIGGER IF EXISTS trigger_hotelcount_add ON "Project Hotel".Hotel;
DROP TRIGGER IF EXISTS trigger_hotelcount_del ON "Project Hotel".Hotel;
CREATE TRIGGER trigger_hotelcount_add AFTER INSERT ON "Project Hotel".Hotel FOR EACH ROW EXECUTE PROCEDURE hotelcount_add();
CREATE TRIGGER trigger_hotelcount_del AFTER DELETE ON "Project Hotel".Hotel FOR EACH ROW EXECUTE PROCEDURE hotelcount_del();

CREATE OR REPLACE FUNCTION roomcount_add()  RETURNS trigger AS $$BEGIN
UPDATE "Project Hotel".Hotel SET roomCount=roomCount+1 WHERE (NEW.hotelID = hotelID); RETURN NEW; END;$$ LANGUAGE plpgsql;
CREATE OR REPLACE FUNCTION roomcount_del()  RETURNS trigger AS $$BEGIN
UPDATE "Project Hotel".Hotel SET roomCount=roomCount-1 WHERE (OLD.hotelID = hotelID); RETURN OLD; END;$$ LANGUAGE plpgsql;

DROP TRIGGER IF EXISTS trigger_roomcount_add ON "Project Hotel".Room;
DROP TRIGGER IF EXISTS trigger_roomcount_del ON "Project Hotel".Room;
CREATE TRIGGER trigger_roomcount_add  AFTER INSERT ON "Project Hotel".Room FOR EACH ROW EXECUTE PROCEDURE roomcount_add();
CREATE TRIGGER trigger_roomcount_del  AFTER DELETE ON "Project Hotel".Room FOR EACH ROW EXECUTE PROCEDURE roomcount_del();

/*check for conflicts, then complete resCur*/
CREATE OR REPLACE FUNCTION rescur_pre() RETURNS trigger AS $$
DECLARE
	cost numeric(10,2);
	dt float8;
BEGIN
	IF (NEW.roomID IN (SELECT roomIDoccu( NEW.dateBeg, NEW.dateEnd, NEW.hotelID))) THEN
		RAISE EXCEPTION 'Room is already booked during the time interval specified';
	END IF;
	NEW.bookingNum = (SELECT bookingNum FROM "Project Hotel".Customer WHERE (NEW.userID = userID));

	SELECT price INTO cost FROM "Project Hotel".Room WHERE (NEW.roomID = roomID);
	dt = DATE_PART('day', NEW.dateEnd - NEW.dateBeg) * 24 + DATE_PART('hour', NEW.dateEnd - NEW.dateBeg);
	NEW.paymentDue = ROUND( dt / 24 * cost::float8 * 100)/100;
	RETURN NEW;
END;$$ LANGUAGE plpgsql;

/*increment bookingNum*/
CREATE OR REPLACE FUNCTION rescur_pst() RETURNS trigger AS $$BEGIN
	UPDATE "Project Hotel".Customer SET bookingNum=bookingNum+1 WHERE (NEW.userID = userID);
	RETURN NEW;
END;$$ LANGUAGE plpgsql;

DROP TRIGGER IF EXISTS trigger_rescur_pre ON "Project Hotel".ReservationCurrent;
DROP TRIGGER IF EXISTS trigger_rescur_pst ON "Project Hotel".ReservationCurrent;

CREATE TRIGGER trigger_rescur_pre BEFORE INSERT ON "Project Hotel".ReservationCurrent
	FOR EACH ROW EXECUTE PROCEDURE "Project Hotel".rescur_pre();
CREATE TRIGGER trigger_rescur_pst AFTER  INSERT ON "Project Hotel".ReservationCurrent
	FOR EACH ROW EXECUTE PROCEDURE "Project Hotel".rescur_pst();
```