CIS395-1900 Spring 2022 Project 1 – Sample Solution by Prof Jose Ramon Santos

(Please do not copy)

Enhancement to the hoteldb schema

1. Enhance the primary keys of each base tables (relations) to utilize automatic generation of the primary keys.

```
-- SQL to fix the primary keys as auto_increment.

alter table booking modify column hotelNo int;

alter table booking modify column roomNo int;

alter table booking modify column guestNo int;

-- auto_increment must be changed to int datatypes

alter table hotel modify column hotelNo int auto_increment;

alter table guest modify column guestNo int auto_increment;

alter table booking add column bookingNo int primary key auto_increment;
```

2. Enhance the tables guest, hotel which store address information. Include real-world address fields as applicable.

```
alter table hotel add column stateCd char(2);

alter table hotel add column zip varchar(20);

alter table hotel add column countryCd varchar(200);

alter table guest add column stateCd char(2);

alter table guest add column zip varchar(20);

alter table guest add column countryCd varchar(200);
```

State and Country names can be coded in a codeName table where the abbreviation and codetype can be the primary key that can identify a row to retrieve a name column that saves the full name of the state or country. Other code to name translations can use this table by grouping using a value in the codetype column.

3. The current fields of the booking table uses a primary key that may not be sufficient for ensuring uniqueness of the primary key. If a current booking needs to change a guest's start date, the old booking start date will be an additional record related to the guest's old booking information, and a new booking record with a new start date will represent the new booking information. This is due to the

primary key of the existing design for booking table as (guestNo, hotelNo, roomNo, startDate). Enhance the booking table to use a unique column as primary key. This column can be similar to the other entities and can utilize automatic generation of values. Apply a uniqueness constraint to the booking table using the columns: guestNo, hotelNo, roomNo, startDate. Lastly, add a datetime column to the booking table to keep a record of the latest insert or update of a row in the booking table.

3. Booking table needs uniqueness constraint on columns: guestNo, hotelNo, roomNo, startDate.

alter table booking add column bookingNo int primary key auto_increment;

alter table booking add unique index booking_idx (guestNo, hotelNo, roomNo, dateFrom);

Booking table needs lstModTS column of timestamp datatype.

alter table booking add column lastModTS timestamp default current_timestamp;

4. In a short paragraph, describe a process to determine if a new booking will have an intersection of rows with a previous booking of the same guestNo, hotelNo, and roomNo.

A solution to determine if a guest making a new booking has overlapping days with a prior booking is by enhancing the hoteldb schema to add a calendar of days of the years in scope. SQL can be used to determing if dates from one booking row intersect with dates of another booking row by counting the rows for each date within each booking record. Or the SQL can be selecting using the guestNo for similar results.

Check two booking records if days overlap:

```
use hoteldb;

select t1.tdate, count(*)

from tcalendar t1

,    booking b1

where (t1.tdate >= b1.dateFrom

and    t1.tdate <= b1.dateTo)

and b1.guestNo = 1

group by t1.tdate

having count(*) > 1
```

```
order by t1.tdate;
```

To setup the dates table, I listed the steps below:

4.1. Create a table with full year dates tcalendar. One primary key, the date column datatype.

```
use hoteldb;
create table tcalendar (tdate date primary key);
```

4.2. Insert dates for the active years with booking data.

```
use hoteldb;
insert into tcalendar
select date_add(str_to_date(12/31/19','%m/%d/%y'),interval 1 day);
insert into tcalendar
select date_add((select max(tdate) from tcalendar where year(tdate) = 2020),interval 1 day);
```

4.3. Check if there is an intersection between two booking data for a guest.

```
-- SQL to list the dates of a guest's booking data
use hoteldb;
select t1.tdate
from tcalendar t1
,    booking b1
where (t1.tdate >= b1.dateFrom
and    t1.tdate <= b1.dateTo)
and b1.guestNo = 1;

-- SQL to insert test data that has an overlap.
test an overlap for guestno = 1
insert into BOOKING values(1,1,'2021-09-08','2021-09-13',1);
```

```
-- SQL to list dates that are overlapping between two booking information.
use hoteldb;
select t1.tdate, count(*)
from tcalendar t1
,    booking b1
where (t1.tdate >= b1.dateFrom
and    t1.tdate <= b1.dateTo)
and b1.guestNo = 1
group by 1
having count(*) > 1
order by 1;
```

5. In a short paragraph, describe how we use SQL to avoid hard-coding tax rates in the SQL. This relates to item 2, where address information can be improved. Assume that tax rates are different for each state and country, design a table that can have state column and country column, and have a tax rate. INSERT records and write SQL to calculate cost of stay using the tax rate from the table.

Solution:

5.1 Create a State Tax Table

```
use hoteldb;
create table stateTax
(stateCd varchar(2) primary key,
 taxRate dec(12,4) default 0);
```

5.2 Insert rows to stateTax table

5.3 Modify cost of stay by joining the stateTax table for related rows based on the stateCd.

CodeName Table:

Additional tables like codeName table:

```
use hoteldb;
create table codeName
(code varchar(20) not null
,codeType varchar(20) not null
,name varchar(60)
,primary key (code, codeType));

use hoteldb;
insert into codeName (code, codeType, name) values ('NJ','STATE','New Jersey');
insert into codeName (code, codeType, name) values ('NY','STATE','New York');
insert into codeName (code, codeType, name) values ('CA','STATE','California');
insert into codeName (code, codeType, name) values ('IL','STATE','Illinois');
insert into codeName (code, codeType, name) values ('PA','STATE','Pennsylvania');
insert into codeName (code, codeType, name) values ('AZ','STATE','Arizona');
insert into codeName (code, codeType, name) values ('CO','STATE','Colorado');
insert into codeName (code, codeType, name) values ('CT','STATE','Connecticut');
insert into codeName (code, codeType, name) values ('MA','STATE','Massachusetts');
insert into codeName (code, codeType, name) values ('VA','STATE','Virginia');
insert into codeName (code, codeType, name) values ('DC','STATE','District of Columbia');
insert into codeName (code, codeType, name) values ('FL','STATE','Florida');
insert into codeName (code, codeType, name) values ('GA','STATE','Giorgia');
insert into codeName (code, codeType, name) values ('SC','STATE','South Carolina');
insert into codeName (code, codeType, name) values ('NC','STATE','North Carolina');
insert into codeName (code, codeType, name) values ('DE','STATE','Delaware');
insert into codeName (code, codeType, name) values ('WV','STATE','West Virginia');
insert into codeName (code, codeType, name) values ('OR','STATE','Oregon');
insert into codeName (code, codeType, name) values ('WA','STATE','Washington');
```

```
insert into codeName (code, codeType, name) values ('MN','STATE','Minnesota');
insert into codeName (code, codeType, name) values ('ME','STATE','Maine');
insert into codeName (code, codeType, name) values ('VT','STATE','Vermont');
insert into codeName (code, codeType, name) values ('IN','STATE','Indiana');
insert into codeName (code, codeType, name) values ('MO','STATE','Missouri');
insert into codeName (code, codeType, name) values ('OK','STATE','Oklahoma');
insert into codeName (code, codeType, name) values ('TN','STATE','Tennessee');
insert into codeName (code, codeType, name) values ('AL','STATE','Alabama');
insert into codeName (code, codeType, name) values ('OH','STATE','Ohio');
insert into codeName (code, codeType, name) values ('TX','STATE','Texas');
insert into codeName (code, codeType, name) values ('NM','STATE','New Mexico');
insert into codeName (code, codeType, name) values ('NV','STATE','Nevada');
insert into codeName (code, codeType, name) values ('MT','STATE','Montana');
insert into codeName (code, codeType, name) values ('UT','STATE','Utah');
insert into codeName (code, codeType, name) values ('HI','STATE','Hawaii');
insert into codeName (code, codeType, name) values ('AK','STATE','Alaska');
insert into codeName (code, codeType, name) values ('AR','STATE','Arkansas');
insert into codeName (code, codeType, name) values ('ND','STATE','North Dakota');
insert into codeName (code, codeType, name) values ('SD','STATE','South Dakota');
insert into codeName (code, codeType, name) values ('ID','STATE','Idaho');
insert into codeName (code, codeType, name) values ('WY','STATE','Wyoming');
insert into codeName (code, codeType, name) values ('IO','STATE','IOWA');
insert into codeName (code, codeType, name) values ('KS','STATE','Kansas');
insert into codeName (code, codeType, name) values ('WI','STATE','Wisconsin');
insert into codeName (code, codeType, name) values ('KN','STATE','Kentucky');
insert into codeName (code, codeType, name) values ('NH','STATE','New Hampshire');
insert into codeName (code, codeType, name) values ('LA','STATE','Louisiana');
insert into codeName (code, codeType, name) values ('MD','STATE','Maryland');
insert into codeName (code, codeType, name) values ('MS','STATE','Mississippi');
```

```
insert into codeName (code, codeType, name) values ('NB','STATE','Nebraska');
insert into codeName (code, codeType, name) values ('RI','STATE','Rhode Island');
```

Using the codeName table to decode stateCd and CountryCd.

```
select g.name
,      g.address
,      g.city
,      c.name
from  guest g
,     codeName c
where  g.stateCd = c.code
and    c.type = 'STATE';
```