UNIX Coursework 2 Report
Data Management COMP1204

University of Southampton

Suyash Datt Dubey

sdd1n17

ID: 29533414

1 The Relational Model and Entity-Relationship Diagramming

1.1 EX1

- R1(ReviewID: Integer, HotelID: Integer, Overall rating: Double, Avg. Price: String, URL: String, Author: String, Content: String, Date: String, No. Reader: Integer, No. Helpful: Integer, Overall: Integer, Value: Integer, Rooms: Integer, Location: Integer, Cleanliness: Integer, Check in/ front desk: Integer, Service: Integer, Business service: Integer)

- **Note:** This is assuming that an author can write multiple reviews of a hotel, but not on the same day.

- **Note:** Underlined attributes form the primary key.

1.2 EX2

- HotelID, ReviewID $\longrightarrow$ Overall rating, Avg. Price, URL, Author, Content, Date, No. Reader, No. Helpful, Overall, Value, Rooms, Location, Cleanliness, Check in / front desk, Service, Business service

- HotelID, Author, Date $\longrightarrow$ ReviewID, Overall rating, Avg. Price, URL, Content, Date, No. Reader, No. Helpful, Overall, Value, Rooms, Location, Cleanliness, Check in / front desk, Service, Business service

- HotelID $\longrightarrow$ Overall rating, Avg. Price, URL

1.3 EX3

- 1NF Already in 1NF

- 2NF

  - R1(ReviewID, HotelID, Author, Content, Date, No. Reader, No. Helpful, Overall, Value, Rooms, Location, Cleanliness, Check in / front desk, Service, Business Service)
  - R2(HotelID, Overall rating, Avg. Price, URL)

- **Note:** Since no transitive dependencies exist, the 3NF form and BCNF form are the same as the 2NF form.

1.4 EX4

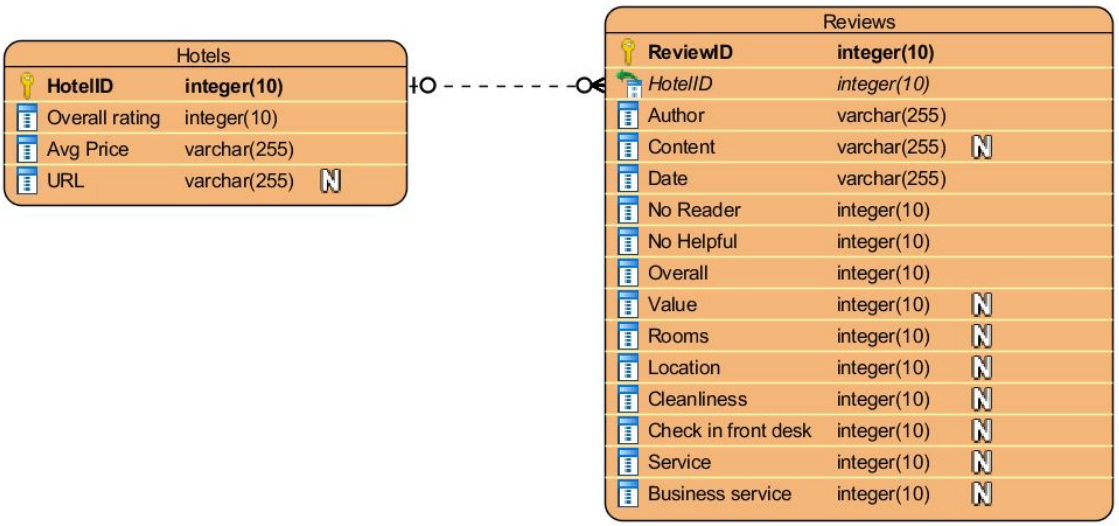

Figure 1: ERD Model of Normalised Relations

2 Relational Algebra

2.1 EX5

$$\sigma_{Author="UserID"}(Reviews)$$

2.2 EX6

$$\pi_{Author,\ ReviewCount}(\sigma_{ReviewCount>2}({}_{Author}\gamma_{count(ReviewID)} \rightarrow {}_{ReviewCount}(Reviews)))$$

2.3 EX7

$$\pi_{HotelID}(\sigma_{ReviewCount>10}({}_{HotelID}\gamma_{count(ReviewID)} \rightarrow {}_{ReviewCount}(Reviews)))$$

2.4 EX8

$$(\pi_{HotelID}(\sigma_{Overallrating>3}(Hotels))) \bowtie (\pi_{HotelID}(\sigma_{AvgCleanliness\geq 4.5}({}_{HotelID}\gamma_{avg(Cleanliness)} \rightarrow {}_{AvgCleanliness}(Reviews))))$$

3 SQL Queries

3.1 EX9

```
CREATE TABLE HotelReviews(
HotelID integer NOT NULL,
ReviewID integer NOT NULL,
OverallRating double NOT NULL,
AvgPrice varchar NOT NULL,
URL varchar,
Author varchar NOT NULL,
Content varchar,
Date varchar NOT NULL,
NoReader integer NOT NULL,
NoHelpful integer NOT NULL,
Overall integer NOT NULL,
Value integer,
Rooms integer,
Location integer,
Cleanliness integer,
CheckIn integer,
Service integer,
BusinessService integer,
PRIMARY KEY (HotelID, ReviewID));
```

3.2 EX10

Bash script used is in the appendix. It takes a long time to run on the given dataset.

3.3 EX11

```
CREATE TABLE Hotels(
HotelID integer PRIMARY KEY,
Overallrating double,
AvgPrice varchar,
URL varchar);
```

```
CREATE TABLE Reviews(
ReviewID integer PRIMARY KEY,
Author varchar,
Content varchar,
Date varchar,
NoReader integer,
NoHelpful integer,
Overall integer,
Value integer,
Rooms integer,
Location integer,
Cleanliness integer,
CheckIn integer,
Service integer,
BusinessService integer,
HotelID integer,
FOREIGN KEY(HotelID) REFERENCES Hotels(HotelID));
```

3.4 EX12

```
1 INSERT INTO Hotels SELECT HotelID, Overallrating, AvgPrice, URL FROM
      HotelReviews;
2
3 INSERT INTO Reviews SELECT ReviewID, HotelID, Author, Content, Date,
      NoReader, NoHelpful, Overall, Value, Rooms, Location,
     Cleanliness, CheckIn, Service, BusinessService FROM HotelReviews;
4
5 DROP TABLE HotelReviews;
```

3.5 EX13

```
1 CREATE INDEX ReviewsHotel ON Reviews(HotelID);
2 CREATE INDEX ReviewDate ON Reviews(HotelID, Date);
3 CREATE INDEX Author ON Reviews(Author);
4 CREATE INDEX PriceAndRating ON Hotels(AvgPrice, Overallrating);
5 CREATE INDEX Helpful ON Reviews(HotelID, NoHelpful);
```

Reasoning:

1. Users should be able to view the reviews of a particular hotel.
2. Users should be able to view how recently a review has been written for a particular hotel.
3. Users should be able to view other reviews an Author has written.
4. Users would find it helpful to view the Average Price and Overall Rating of hotels so they can choose a hotel suitable for them.
5. Users should be able to see how many other users have found certain reviews of a hotel helpful.

3.6 EX14

```
1 SELECT * FROM Reviews WHERE Author="UserID";
2 SELECT Author, COUNT(DISTINCT(HotelID)) FROM Reviews GROUP BY Author
      HAVING COUNT(Author)>2;
3 SELECT DISTINCT(HotelID) FROM Reviews GROUP BY HotelID HAVING COUNT(
     ReviewID)>10;
4 SELECT Hotels.HotelID FROM Reviews, Hotels WHERE Hotels.HotelID=
     Reviews.HotelID AND Hotels.OverallRating>3 GROUP BY Reviews.
     HotelID HAVING AVG(Reviews.Cleanliness)>=4.5;
```

3.7 EX15

- For some reviews, the URL field is empty.
- Writing the bash script was quite time-consuming. The insertion part took some time to figure out.
- Many of the ratings for attributes are null or -1.
- The entries in the date field were not in the same format as the date data type in SQL.
- Converting Relational Algebra to SQL Queries was not straightforward.
- Dataset provided is extensive, and therefore it took quite some time to run the script and queries on it. It also made error correction a more prolonged process as well.

4 Conclusions

- My basic approach was to separate all the fields of the reviews and store them in arrays. Using the length of any one of these arrays, create a loop to insert elements into the table.
- Writing the bash script was quite time-consuming. The insertion part took some time to figure out.
- Dataset provided is extensive, and therefore it took quite some time to run the script and queries on it. This also made fixing bugs a more prolonged process.

5 Appendix

```
#!/bin/bash
>hotelreviews.sql
sqlite3 hotels.db <<EOF

DROP TABLE IF EXISTS HotelReviews;

CREATE TABLE HotelReviews(
HotelID integer NOT NULL,
ReviewID integer NOT NULL,
OverallRating double NOT NULL,
AvgPrice varchar NOT NULL,
URL varchar,
Author varchar NOT NULL,
Content varchar,
Date varchar NOT NULL,
NoReader integer NOT NULL,
NoHelpful integer NOT NULL,
Overall integer NOT NULL,
Value integer,
Rooms integer,
Location integer,
Cleanliness integer,
CheckIn integer,
Service integer,
BusinessService integer,
PRIMARY KEY (HotelID, ReviewID));

EOF


for file in $1/*;
do
    HotelID=$(basename $file | sed -e 's/\.dat//g' -e 's/.*_//g')
    OverallRating=$(grep "Overall Rating" $file  | cut -c17-)
    AvgPrice=$(grep "Avg. Price" $file | cut -c13-)
    URL=$(grep \<URL\> $file | cut -c6-)
    Author=($(grep \<Author\> $file | cut -c9- | sed -r 's/ //g'))
    Content=($(grep \<Content\> $file | cut -c10- | sed -r 's/ //g')
       )
    Date=($(grep \<Date\> $file | cut -c7- | sed -r 's/ //g'))
    Reader=($(grep "No. Reader" $file | cut -c13-))
    Helpful=($(grep "No. Helpful" $file | cut -c14-))
    Overall=($(grep \<Overall\> $file | cut -c10-))
    Value=($(grep \<Value\> $file | cut -c8-))
    Rooms=($(grep \<Rooms\> $file | cut -c8-))
```

```
45      Location=($(grep \<Location\> $file | cut -c11-))
46      Clean=($(grep \<Cleanliness\> $file | cut -c14-))
47      Checkin=($(grep "<Check in / front desk>" $file | cut -c24-))
48      Service=($(grep \<Service\> $file | cut -c10-))
49      Bservice=($(grep "Business service" $file | cut -c19-))
50
51      length=${#Author[@]}
52      for((i = 0; i < $length; i++));
53      do
54          echo "INSERT INTO HotelReviews (HotelID, OverallRating,
                AvgPrice, URL, ReviewID, Author, Content, Date, NoReader,
                 NoHelpful, Overall, Value, Rooms, Location, Cleanliness,
                 CheckIn, Service, BusinessService) VALUES (\"$HotelID\",
                 \"$OverallRating\", \"$AvgPrice\", \"$URL\", \"$i\", \"$
                {Author[i]}\, \"${Content[i]}\", \"${Date[i]}\", \"${
                Reader[i]}\", \"${Helpful[i]}\", \"${Overall[i]}\", \"${
                Value[i]}\", \"${Rooms[i]}\", \"${Location[i]}\", \"${
                Clean[i]}\", \"${Checkin[i]}\", \"${Service[i]}\", \"${
                Bservice[i]}\");" >> hotelreviews.sql
55      done
56  done
```