1. Client Description

Our client, *NorthGate*, is a hotel chain brand whose hotels are located in various cities across the United States. The goal of this project is to design a database for *NorthGate* to satisfy the data storage, management, and query needs in its daily business operations.

As a hotel, the fundamental goal of *NorthGate* is to provide a comfortable stay experience for its customers. *NorthGate* provides check-in and check-out services, which are the most basic services of all hotels. Besides, *NorthGate* needs to record and manage its own staff and keep records of historical room prices for its data analysis needs. At the same time, *NorthGate* also provides a variety of other services, including dining, spa, laundry, and shuttle, which can be ordered by customers who have already checked in at the hotel.

Customers of *NorthGate* can book rooms online, and customers with different point levels can enjoy different discounts when booking rooms. During the stay, customers can purchase various services provided by the hotel by placing service orders. After checking out, they can rate their stay experience.

The employees of *NorthGate* are divided into three categories, full-time, part-time and others, each employee can only work in exactly one hotel of *NorthGate*. Also, *NorthGate* requires each employee to be responsible for at least one guest room.

To support the business operation of *NorthGate*, we have designed a database for it. In our database, we have designed 13 entities including *Hotel*, *Room*, *Customer*, *Customer Type*, *Room Order*, *Historical Price*, *Service* and its subclasses, *Service Order*, and *Employee* and its subclasses. Also, we have 12 relationships connecting these entities. Finally, all these entities and relationships are converted into 22 tables.

When building our database, we make the following assumptions:

1. When booking rooms online, if multiple rooms are booked in one order, the check-in and check-out dates for all the rooms must be the same. Reserving rooms with different check-in and check-out dates requires placing multiple orders.
2. Each service order can only be associated with one room. Different rooms need to place separate service orders to order services.
3. The database only records employees' work emails, and each employee has only one work email. However, a customer can have multiple email addresses.
4. The hotel offers only four types of services, including dining, spa, laundry, and shuttle. Therefore, the relationship between the superclass *Service* and its four subclasses is disjoint and total.
5. There are three types of employees in *NorthGate*, including full-time, part-time and others, and the relationship between the superclass *Employee* and its three subclasses is disjoint and total.

2. EER Model

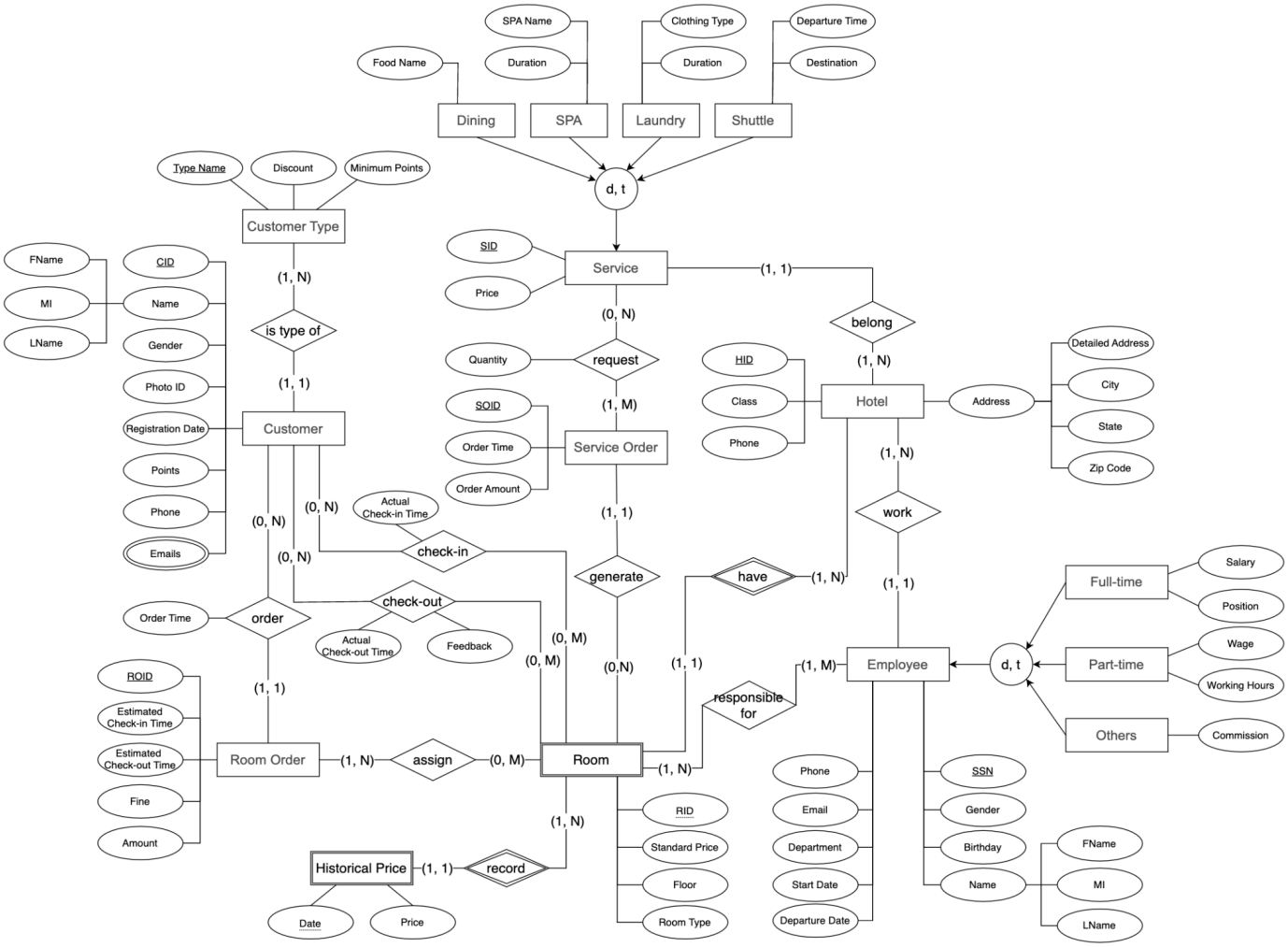


Figure 1. *EER Model*

3. Relational Design (Schema)

1. CUSTOMER_TYPE (Type_Name, Discount, Minimum_Points)
2. CUSTOMER (CID, FName, MI, LName, Gender, Registration_Date, Photo_ID, Points, Phone, Type_Name[1])
3. HOTEL (HID, Class, Phone, Detailed_Address, City, State, Zipcode)
4. ROOM (RID, HID[3], Standard_Price, Floor, Room_Type)
5. ROOM_ORDER (ROID, Estimated_Check_in_Time, Estimated_Check_out_Time, Fine, Amount, CID[2], Order_Time)
6. HISTORICAL_PRICE (Date, RID[4], HID[4], Price)
7. SERVICE (SID, Price, HID[3])
8. DINING (SID[7], Food_Name)
9. SPA (SID[7], SPA_Name, SDuration)
10. LAUNDRY (SID[7], Clothing_Type, LDuration)

11. SHUTTLE (SID[7], Departure_Time, Destination)
12. SERVICE_ORDER (SOID, Order_Time, Order_Amount, RID[4], HID[4])
13. EMPLOYEE (SSN, Gender, Birthday, EFName, EMI, ELName, EPhone, EEmail, Department, Start_Date, Departure_Date, HID[3])
14. FULL_TIME (SSN[13], Salary, Position)
15. PART_TIME (SSN[13], Wage, Working_Hours)
16. OTHERS (SSN[13], Commission)
17. ASSIGN (ROID[5], RID[4], HID[4])
18. CHECKIN (CID[2], RID[4], HID[4], Actual_Checkin_Time)
19. CHECKOUT (CID[2], RID[4], HID[4], Acutual_Checkout_Time, Feedback)
20. REQUEST (SOID[12], SID[7], Quantity)
21. RESPONSIBLE_FOR (SSN[13], RID[4], HID[4])
22. EMAILS (CID[2], Email)

4. Table Structure in MySQL

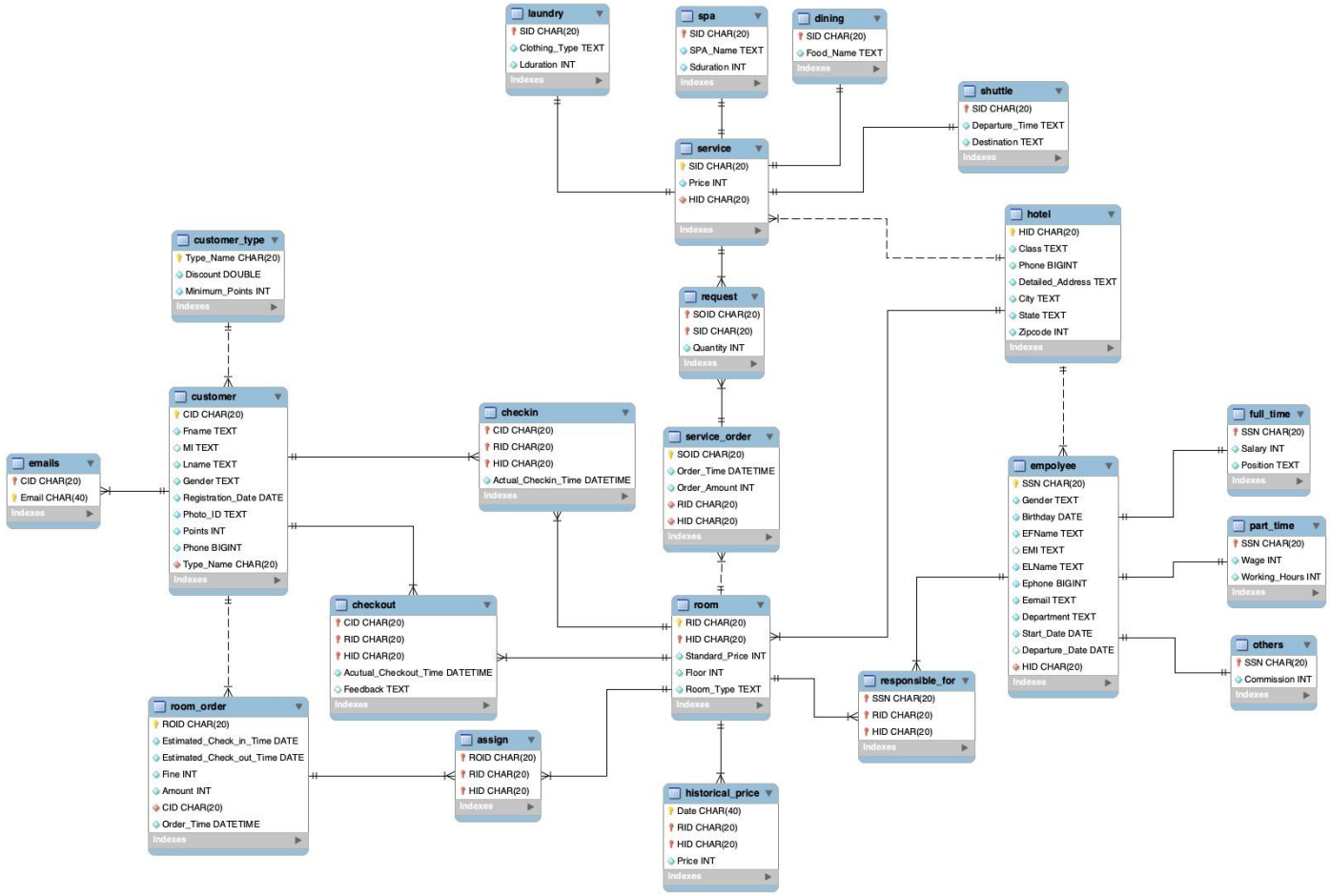


Figure 2. *Screenshot of the Table Structure in MySQL*

Here are some of the tables and data we have implemented in MySQL:

TABLE 1. *Checkout*

CID	RID	HID	Acutual_Checkout_Time	Feedback
20220101001	r1301	h9001	2022/1/3	Good!
20220101002	r1302	h9001	2022/1/2	
20220101003	r1303	h9001	2022/1/2	
20220101004	r1304	h9001	2022/1/4	Tasty food. I like it.
20220101005	r1305	h9001	2022/1/2	
20220101006	r1401	h9002	2022/1/2	
20220102001	r1402	h9002	2022/1/3	
20220102002	r2403	h9002	2022/1/3	
20220101003	r2404	h9002	2022/1/3	Great service.
20220101004	r2405	h9002	2022/1/3	

TABLE 2. *Customer*

CID	Fname	MI	Lname	Gender	Registration_Date	Photo_ID	Points	Phone	Type_Name
20220101001	William	David	Smith	M	2022-01-01	x2967318	8	9253284290	normal
20220101002	Nancy		Lee	F	2022-01-01	x3720476	39	6463446932	normal
20220101003	Benjamin		Green	M	2022-01-01	y2937162	635	5106392402	vvip
20220101004	Noah		Phillips	M	2022-01-01	y2937297	62	9206755845	normal
20220101005	Robert		Brown	M	2022-01-01	x2937244	340	6699478833	vip
20220101006	Liam	Louise	Smith	M	2022-01-01	y3013735	66	7148663956	normal
20220102001	Amy	Rose	Johnson	F	2022-01-01	y6428732	289	5102864208	vip
20220102002	Oliver		Thompson	M	2022-01-02	x9348276	95	5107293722	normal
20220102003	Patricia	Mary	Jones	F	2022-01-02	y2019373	24	9258475562	normal
20220102004	James		Hill	M	2022-01-02	y8342121	189	5109374766	vip

TABLE 3. *Employee*

SSN	Gender	Birthday	EFName	EMI	ELName	Ephone	Eemail	Department	Start_Date	Departure_Date	HID
325293752	F	1970/12/3	Barbara		Johnson	5109283756	barbarajohnson@gmail.com	housekeeping	2021/12/1		h9001
936286152	F	1969/8/28	Linda		Smith	6692648269	lindasmith@gmail.com	housekeeping	2021/12/1		h9001
282338254	M	1977/3/19	Charles		Wilson	8052847764	charleswilson@gmail.com	housekeeping	2021/12/3		h9003
837529374	M	1983/7/24	Robert	John	Brown	9253728335	robertjohnbrown@gmail.com	food	2020/11/27	2021/12/21	h9004
372638304	F	1990/10/18	Hannah		Chen	4152382733	hannahchen@gmail.com	food	2021/10/30		h9004
659483602	M	1992/2/14	Richard		Smith	5108726637	richardsmith@gmail.com	guest services	2021/11/13		h9006
538273649	F	1988/8/22	Victoria		Jones	9253745529	victoriajones@gmail.com	guest services	2021/12/20		h9007
479204749	M	1985/11/5	Michael	James	Caddel	5301729283	michaeljames@gmail.com	marketing	2021/11/30		h9008
827462999	F	1993/12/9	Lucy		Miller	7602846283	luchmiller@gmail.com	accounting	2021/12/8		h9008
938272766	F	1989/5/27	Emma		Lee	5103846281	emmalee@gmail.com	accounting	2021/12/11		h9008

TABLE 4. *Full_Time*

SSN	Salary	Position
325293752	31000	housekeeper
936286152	31000	housekeeper
837529374	65000	chef
372638304	85000	kitchen manager
538273649	63000	receptionist
294713976	59000	security
827462999	65000	junior accountant
938272766	80000	senior accountant
836194722	31000	housekeeper
237492749	65000	junior accountant

TABLE 5. *Hotel*

HID	Class	Phone	Detailed_Address	City	State	Zipcode
h9001	economy	5102342837	8925 Telegraph Ave	San Fransico	CA	94108
h9002	deluxe	9252736258	3912 13th St	Los Angeles	CA	90008
h9003	standard	5103827364	11 Washington St	San Jose	CA	94089
h9004	superior	4152837466	87 Airport Access Rd	San Diego	CA	91932
h9005	deluxe	7183628354	1983 Powell St	New York	NY	10005
h9006	superior	9294725193	4819 Coliseum Wy	Buffalo	NY	14213
h9007	superior	7133928368	6652 Edes St	Dallas	TX	75032
h9008	economy	9706726233	3802 Shellmound Ave	Denver	CO	80202
h9009	economy	3865826376	565 Hegenberger Rd	Miami	FL	33125
h9010	economy	5032837622	1695 Broadway	Portland	OR	97201

TABLE 6. *Room*

RID	HID	Standard_Price	Floor	Room_Type
r1301	h9001	239	3	standard
r1302	h9001	239	3	standard
r1303	h9001	239	3	standard
r1304	h9001	329	3	deluxe
r1305	h9001	239	3	standard
r1401	h9002	239	4	standard
r1402	h9002	239	4	standard
r2403	h9002	329	4	deluxe
r2404	h9002	329	4	deluxe
r2405	h9002	329	4	deluxe

TABLE 7. *Room_Order*

ROID	Estimated_Check_in_Time	Estimated_Check_out_Time	Fine	Amount	CID	Order_Time
ro1301	2022-01-01	2022-01-03	0	478	20220101001	12/19/2021-14:18
ro1302	2022-01-01	2022-01-02	0	239	20220101002	12/22/2021-09:17
ro1303	2022-01-01	2022-01-02	50	289	20220101003	12/26/2021-12:36
ro1304	2022-01-01	2022-01-04	0	987	20220101004	12/30/2021-18:07
ro1305	2022-01-01	2022-01-02	0	239	20220101005	12/20/2021-01:46
ro1401	2022-01-01	2022-01-02	0	239	20220101006	12/24/2021-23:14
ro1402	2022-01-02	2022-01-03	0	239	20220102001	12/28/2021-19:58
ro2403	2022-01-02	2022-01-03	100	429	20220102002	12/20/2021-16:36
ro2404	2022-01-02	2022-01-03	0	329	20220101003	01/01/2022-01:53
ro2405	2022-01-02	2022-01-03	30	359	20220101004	12/18/2021-11:52

TABLE 8. *Shuttle*

SID	Departure_Time	Destination
10006	1:00	airport
10007	2:00	airport
10033	3:00	airport
10034	4:00	airport
10035	5:00	airport
10036	6:00	airport
10037	7:00	airport
10038	8:00	airport
10039	9:00	airport
10040	10:00	airport

TABLE 9. *Spa*

SID	SPA_Name	Sduration
10008	signature 60	60
10009	signature 80	80
10010	signature 100	100
10011	anti ageing	90
10012	ayurvedic	80
10013	medical	80
10014	healing	80
10015	detox	90
10016	thalassotherapy	80
10017	mineral springs	60

TABLE 10. *Laundry*

SID	Clothing_Type	Lduration
10004	regular	70
10005	dry cleaning	120
10025	fluff	80
10026	colors	60
10027	whites	60
10028	cotton	60
10029	fabric	60
10030	silk	100
10031	commercial	90
10032	special	100

5. Interesting Queries and Data Modelling

5.1 Analytical Question-1

The number of rooms is limited in one hotel, and the price is different for the same room, so, how to make more profit with a limited room? That's an interesting question for a hotel owner. If the hotel only has one room left in the next 7 days, and suppose that the customer can bid for the hotel room, then what is the best way to arrange the schedule to make more profit? To solve this problem, we need to get the customers' bids for the room, however, we don't know the future bids, so we will focus on the price in the past. Assume that one type of room in one hotel shares the same price, and the actual price they paid is the same as their bids. Suppose a customer checked in on 2022-01-01 and checked out on 2022-01-03 and paid $479 when he left, this means he bid $479 for the room.

SQL Query to extract the data from the database:

```
SELECT
     RID,
     Amount,
     Estimated_Check_in_Time,
     Estimated_Check_out_Time,
From ROOM_ORDER RO
JOIN ROOM R ON RO.ROID = R.RID
WHERE R.Room_type = 'standard'
```

From this query, we can get the check in time, check out time, and amount for standard rooms in one hotel. (Note: Amount stands for how much the customer paid when they checked out which is the same as what they bid.) Assume 2022-01-01 is day 1, and we can draw a table that indicates the check in, check out date, and amount:

TABLE 11. *Received Bids and the Date*

Check in Date	Check out date	Customer Number	Amount
1	2	1	259
1	5	2	759
2	4	3	259
4	5	4	159
4	6	5	659
5	6	6	359

- **Model Formulation:**

We can use the shortest path problem model to solve this problem, and we will use AMPL to solve it. For shortest path problem:

$$Min \sum_{(i,j)\in E} c_{ij} x_{ij}$$

$$s.t.\ \sum x_{ij} - \sum x_{ki} = b_i$$

c_{ij}: $distance\ from\ i\ to\ j$

x_{ij}: $units\ of\ flow\ (binary)$

b_i: $1\ at\ origin,\ \ -1\ at\ destination$

Then, we can solve this problem using AMPL. The optimal values are shown below:

```
ampl: display _objname, _obj, _varname, _var;
:    _objname    _obj  _varname _var    :=
1   neg_profit   -1177   x12      1
2         .        .     x23      0
3         .        .     x15      0
4         .        .     x24      1
5         .        .     x34      0
6         .        .     x45      0
7         .        .     x46      1
8         .        .     x56      0
```

Figure 3. *Optimal Values of Decision Variables*

- **Model interpret:**

From the result, we can see that if the hotel only has one room left, the maximum profit for the hotel is $1177, and the hotel should take the bid from customer 1, customer 3, and customer 5. The result makes sense because in this way the room won't be empty for the whole week and the hotel is most profitable. The model is useful for the hotel, because if they have a similar situation again in the future, they can come up with a plan to maximize the profit. However, our model has limitations: First of all, the data set is small, since we only have the data for 6 days. Second, we based on past data, so we know that the customer showed up, however, in real life the customers may not show up, and we may have a penalty for that. These are still questions that need future consideration.

5.2 Analytical Question-2

Hotel's price is changing slightly everyday, however, whether the changes are following a seasonality pattern needs to be verified by implementing the **local regression**. In this query, We are interested in answering the question of whether the hotel's price is changing by seasonality. This analysis helps our clients to understand the cyclical changes in hotels' prices. Meanwhile, our clients can not only **predict the next period trend** but also **reproduce** the period length of different room types (or hotels of different class) by adjusting the grouping method of SQL statements, and formulate the corresponding strategy for each period length.

SQL Query to extract the data from the database:

```
SELECT HID as Hotel_ID,
      DATE(year(Date) || "-" || month(Date)
                      || "-" || "01") as Date_Time,
        AVG(Price) as AVG_Price,
FROM HISTORICAL_PRICE
GROUP BY  HID, -- can be reproduced to another dimension
          month(Date),
```

```
        year(Date)
HAVING year(Date) >= 2013 -- We only use the past 10-years data
```

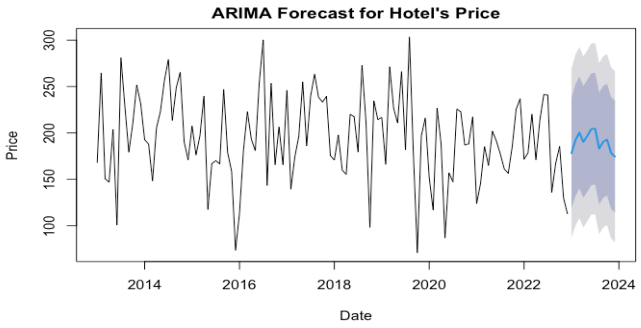
In this data query, we are trying to derive the average price of each hotel on each month.

- **Models Formulation and Visualization:**

To deal with the time-series data, the most classical model is ARIMA(p, d, q), where p is the number of the degree of autoregressive, d is the number of nonseasonal differences for stationary, and q is the number of lagged errors during the prediction . The general forecasting ARIMA models can be written as the following way, where theta represent the moving average parameter.

$$y_t = \mu + \phi_1 y_{t-1} + \ldots + \phi_p y_{p-1} - \Theta_1 e_{t-1} - \ldots - \Theta_p e_{t-p}$$

where ARIMA(1,0,0) can be writes as $y_t = \mu + \phi_1 y_{t-1}$, known as the first-order autoregressive model, providing the insight of time periods that are only 1-period apart. Implement the method directly by calling the R-function `auto.arima()` and deploy the prediction result:




```
ARIMA(0,0,1)(0,0,1)[12] with non-zero mean

Coefficients:
          ma1     sma1      mean
      0.1796  0.2182  194.0691
s.e.  0.0985  0.1031    5.8838

sigma^2 = 2140:  log likelihood = -629.17
AIC=1266.34   AICc=1266.68   BIC=1277.49
```

Figure 4. *ARIMA Prediction Confidence Interval*

Figure 5. *ARIMA Prediction Model*

Using the R build-in function `stl()`, we can directly derive the seasonal decomposition of the prices.

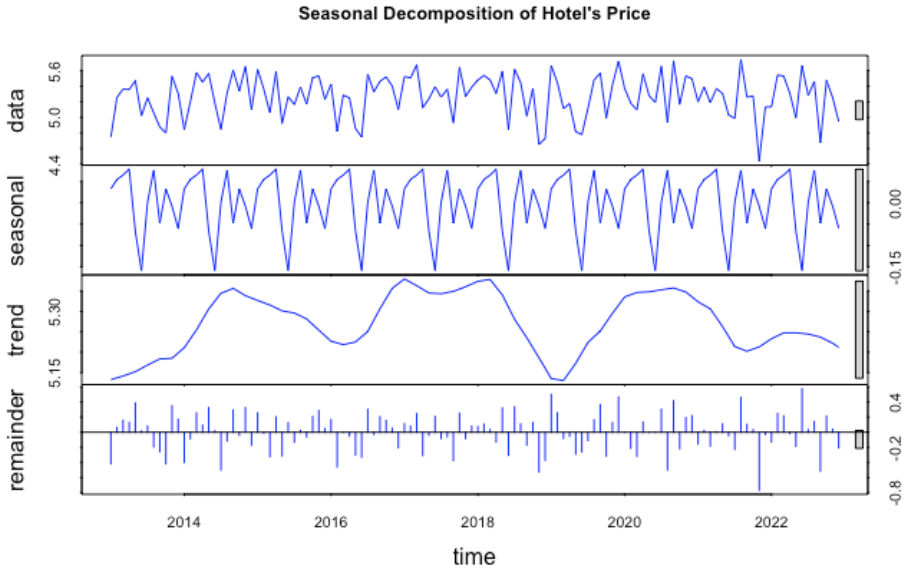


Figure 6. *Seasonal Decomposition of Hotel's Price*

Reference: Hartmann, K., Krois, J., Waske, B. (2018): E-Learning Project SOGA: Statistics and Geospatial Data Analysis. Department of Earth Sciences, Freie Universitaet Berlin.

- **Result Analysis:**

Since the data are randomly generated based on a normal distribution, we can see that the ARIMA model does not predict well. Although the predicted values are within the confidence interval, the range of the confidence interval is unusually wide-spread due to the randomness of the data. In reality, hotel room prices should show regular fluctuations in residual values (remainders) through seasonal decomposition. If this is the case, we can improve the accuracy of the ARIMA model by removing the seasonal effect. As mentioned earlier, with the ARIMA model and seasonal decomposition, the customer can roughly know the cycle length and the expected change in the next cycle. For our clients, they can adjust their publicity strategies in time to lock in certain trends in uncertain changes. In addition, clients can modify SQL to reproduce data with another dimension, such as classes of a hotel, types of rooms, to derive different cycle lengths. These average prices from different dimensions can enable our clients to formulate marketing strategies in an accurate manner.

5.3 Analytical Question-3

The hotel industry is essentially a service industry, which would require many employees in various functional departments to complete the hotel services. One of the most prominent issues is to arrange staff for room services, as staff in other functional departments are rather fixed, insensitive of how many guests checked in that day. However, for room services staff in general, we may face over-staff or short of staff situations in peak seasons or in recession. Therefore, as an analyst, we now focus on how to minimize labor cost associated with the hotel management with respect to room services by recruiting the minimum labors required.

Here, we will make more assumptions. The employees must take two days off when working for 5 days according to the state laws. For hotel rooms, we would require one staff per three rooms. We will first write the MySQL query to extract the data for the staff number required for each workday and then input data into AMPL to find the minimum number of employees required for this integer programming question.

SQL Query to extract the data from the database:

```
SELECT
case when “2022-01-01” BETWEEN Estimated_Check_in_Time AND Estimated_Check_in_Time
Then count(ROID) as Monday
case when “2022-01-02” BETWEEN Estimated_Check_in_Time AND Estimated_Check_in_Time
Then count(ROID) as Tuesday
case when “2022-01-03” BETWEEN Estimated_Check_in_Time AND Estimated_Check_in_Time
Then count(ROID) as Wednesday
case when “2022-01-04” BETWEEN Estimated_Check_in_Time AND Estimated_Check_in_Time
Then count(ROID) as Thursday
case when “2022-01-05” BETWEEN Estimated_Check_in_Time AND Estimated_Check_in_Time
Then count(ROID) as Friday
case when “2022-01-06” BETWEEN Estimated_Check_in_Time AND Estimated_Check_in_Time
Then count(ROID) as Saturday
case when “2022-01-07” BETWEEN Estimated_Check_in_Time AND Estimated_Check_in_Time
Then count(ROID) as Sunday
From ROOM_ORDER
```

TABLE 12. *Extract Data from One-week Period*

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
12	18	17	26	33	50	44

- **Model Formulation:**

We can view it as an integer linear programming problem where the number of staff required cannot be fraction.

$$Min \sum_{i=1}^{7} x_i$$

$$s.t. \sum_{j=i}^{i+4} x_j >= R_i, \quad \forall \text{i}$$

$$\sum x_i >= 0, \forall \text{i}$$

R_i: number of staff required on day i

X_i: number of staff hired on day i

Then, we can solve the problem using AMPL. The optimal values are shown below:

```
ampl: display _var, _varname, _obj,_objname;
: _var _varname _obj          _objname           :=
1    0   Day1     50   Number_of_Staff_Required
2    9   Day2      .              .
3   29   Day3      .              .
4    3   Day4      .              .
5    9   Day5      .              .
6    0   Day6      .              .
7    0   Day7      .              .
;
```

Figure 7. *Optimal Values of Decision Variables*

- **Result Analysis:**

This suggested that according to the optimal results, we will just need to hire a total of 50 staff for this period of time. Specifically, 9 staff on Day 2; 29 staff on Day 3; 3 staff on Day 4; 9 staff on Day 5 and 0 staff for Day 1,6,7.

These results can be generalizable for any period of time. However, we may want to use the average results of one month or a season to calculate the number of staff required. Due to the limited data we have here, this serves as an example of how to minimize the labor cost of hotel room staff, which is really sensitive to the number of guests checked in on one specific day to avoid over-staff or shortage of staff.

Further analysis can be implemented to decide how many part-time or full-time employees we will need to have based on the joined table with the EMPLOYEES Entity.

6. Normalization Analysis

6.1 Analysis part-1

For the schema of *NorthGate* hotel, we've created a relation for each multi-valued attribute so the schema is in the first normal form (1NF).

For example, the attribute *Email* in entity *Customer* is a multi-valued attribute. Instead of:

CUSTOMER (CID, FName, MI, LName ,Gender, Registration_Date, Photo_ID, Points, Phone, Type_Name[1], Email),

in the schema, we have changed it into :

2. CUSTOMER (CID, FName, MI, LName ,Gender, Registration_Date, Photo_ID, Points, Phone, Type_Name[1])

22. EMAILS (CID[2], Email)

6.2 Analysis part-2

Since the second normal form (2NF) disallows partial dependencies for non-prime attributes, the only relation that might violate 2NF is listed as follows:

4. ROOM (RID, HID[3], Standard_Price, Floor, Room_Type)

As we can see, the attribute *Floor* only depends on *RID*. The relation could be modified into:

4a. ROOM (RID, HID[3], Standard_Price, Room_Type)

4b. ROOMFLOOR (RID, Floor)

6.2 Analysis part-3

The third normal form (3NF) disallows transitive dependencies for non-prime attributes. Nevertheless, in the following relation, we find out that *HID* → *City* and *City* → *State*:

3. HOTEL (HID, Class, Phone, Detailed_Address, City, State, Zipcode)

Thus, it can be changed into:

3a. HOTELCITY (City, State)

3b. HOTEL (HID, Class, Phone, Detailed_Address, City[3a], Zipcode)