Group Members:

TU856

Stephen Moore - D21125383, Laura Andrews – D21125370

Domain Types

Outline here the domain types you created, their names, types, the reason you included them and an example how you used them e.g.

Domain Type Name	Type Definition	Reason Introduced	Example
Address	VARCHAR (50)	For attributes that require someone abode or address	employAddress in entity Employee
Age	NUMERIC (3)	For attributes which represent the years someone has been alive	custAge in entity Customer
DOB	DATE (10)	Date of when said person was born	employDOB in entity Employee
Email	VARCHAR (50)	For attributes that required email addresses	tracingEmail in entity Tracing
Identifier	NUMERIC (7)	This is for attributes so the information can be differentiated	tableID in entity Table
Location	VARCHAR (20)	This is for attributes where you need to put the place where something is	tableLocatArea in entity Table
Name	VARCHAR (30)	Used for attributes that require someone's identity	custName in entity Customer
PeopleAmount	NUMERIC (2)	Used for attribute to see amount of people at table	bookingPeopleAmount in entity Booking
PhoneNumber	NUMERIC (15)	Used for attribute to take in contact numbers	tracePhoneNum in entity Tracing

StartDate	DATE (10)	Used for the start of peoples starting date	employStartDate in entity Employee
Style	VARCHAR (20)	Used for the way that the customer will book	bookingStyle in entity Booking
Time	NUMERIC (10)	Used for attributes that need to represent time	overTimeMin in entity Tracing

Major Decisions

Have a section per decision. This can include decisions such as where to place particular attributes, whether to have an entity, what type of relationship to use. Anything you had to discuss you should include in this section

Decision 1

The first major decision we looked at starting the assignment was the number of entities we were going to have this was because there was a section at the bottom of the case study that had an attribute that could be added to its own table or used in an already made table this was the amount of time stayed after the two hours. This attribute in our eyes could have been included in the customer section or made its own entity and in the end, we decided to make it its own table because it was not an absolute that the customer would stay longer than the allowed time every time and may just end up being no used very often and was easier to see in its own section.

Decision 2

The second major decision that had to be made once we had decided on the amount tables based on the case study was what relationship each table would have. We decided this is the relationships each table would have:

Restaurant 1:M Employee

The reason we chose this relationship for these entities was because there can be many employees in one restaurant.

Employee M:1 Table

This is because there can be many employees to one table as maybe the someone who takes the order than a different person who takes the food to the table.

Table 1:M Tracing

Reason this is a 1 to many relationships is because you can have one table for many guests or people being traced.

Table 1:1 Customer

This relationship is because there is one table to one customer

Customer 1:1 Booking

This is because there is one booking for one customer

Customer 1:1 Overstay

This is because there is one overstay fine to one customer that stays more than the allotted time

Decision 3

A major decision we also had to make after deciding all the tables and the relationships were the attributes to go into each table and after looking at the case study and seeing what we needed we decided on

Restaurant:

restaurantID,

restaurantAddress,

restaurantPhoneNum

this is because the restaurant must have an identification number to make it unique from the others, an address to show where the restaurant is and a phonenum for contact reasons.

Customer:

customerID,

customerName,

customerAddress,

customerEmail,

customerAge,

This is the next set of attributes chosen because the customer needs an identifier to make them different from the rest, a name to identify them by, address, email just in case they need to be contacted and age to make sure they can legally book the restaurant.

Booking:

bookingID,

bookingType,

bookingTimeHour

bookingPeopleAmount,

Booking attributes were picked so they can differentiate the bookings, see what type of booking it is walk-in, online etc., and finally to make sure that you know how many people are coming.

Table:

tableID,

tableLocatArea,

tableType

The table attributes were decided on because they need to be able to see what table is being served, where the table is situated by the window etc. and finally what type of table is it a 2-seater, a 4-seater etc.

Tracing:

tracingID,

tracingName,

tracingEmail,

tracingPhoneNum

We decided on these attributes due to the fact they need to identify each contact tracing participant, they need the required information for contact tracing the name, email, and phone number.

Employee:

employID,

employLocatBranch,

employName,

employAddress,

employPhoneNum,

employEmail,

employDOB,

employStartDate

The decision for these attributes is based on the fact that the management will need to be able to make sure they can tell employees apart(employID), they will need to no which branch they work at(employLocatBranch), They will need the name, Address, phone number, email just as basic information to contact their staff with. The Date of birth will ensure that the employee is of legal age to work in the restaurant and the start date is to see how long they have been with the company.

Overstay:

overstayId

overstayTimeMin

These attributes were decided on by the fact that if the customer stays over the limit of 2 hours they will be charged per extra minute and they need the identifier to differentiate each case and the time, so they know how much to charge them based on how long they stayed over.

In conclusion these decisions on the attributes were made in conjunction with looking at the case study and thinking out what would fit in which entity/table and why it would fix there and work.

Decision 4

We had to decide what size and logical type to make each domain as if we made them too short, they couldn't be used too long and it's not a sharp fit and if the type is wrong, they cannot be used for their intended purpose.

For Address we chose varchar and size of 50 this was because varchar allows for letters and a size of 50 because some addresses can be very long.

For Age we chose numeric with size of 3 because age is more efficiently displayed as a number and the size accounts for those lucky enough to live up to and over 100.

DOB for this we chose Date as the type with a size of 10. The type of date was chosen as it is the best way to take in date related attributed.

Email is type varchar with a size 50. Varchar is so you can enter numbers, letters, and symbols and 50 as the email has a long email identifier.

Identifier type Numeric with length 7. This was chosen as number as it is most efficient this way as you can start from 1 and work up and the length accounts for up to a billion ids.

Location is type Varchar with size 20. Varchar is best for this as locations location are identified usually using letters and the size accounts for longer named locations.

Name is type varchar and size 20. The varchar accounts for that name are all letters.

PeopleAmount is type Numeric size 2. This is because amount is bets as numeric and size 2 as they will never take more than 99 people.

PhoneNumber is type NUMERIC with size 15. Type numeric as phone numbers are all numbers and size 15 as 15 is the maximum length for a phone number.

StartDate is type date with size 10. The type is useful in this case because it helps as were looking for date.

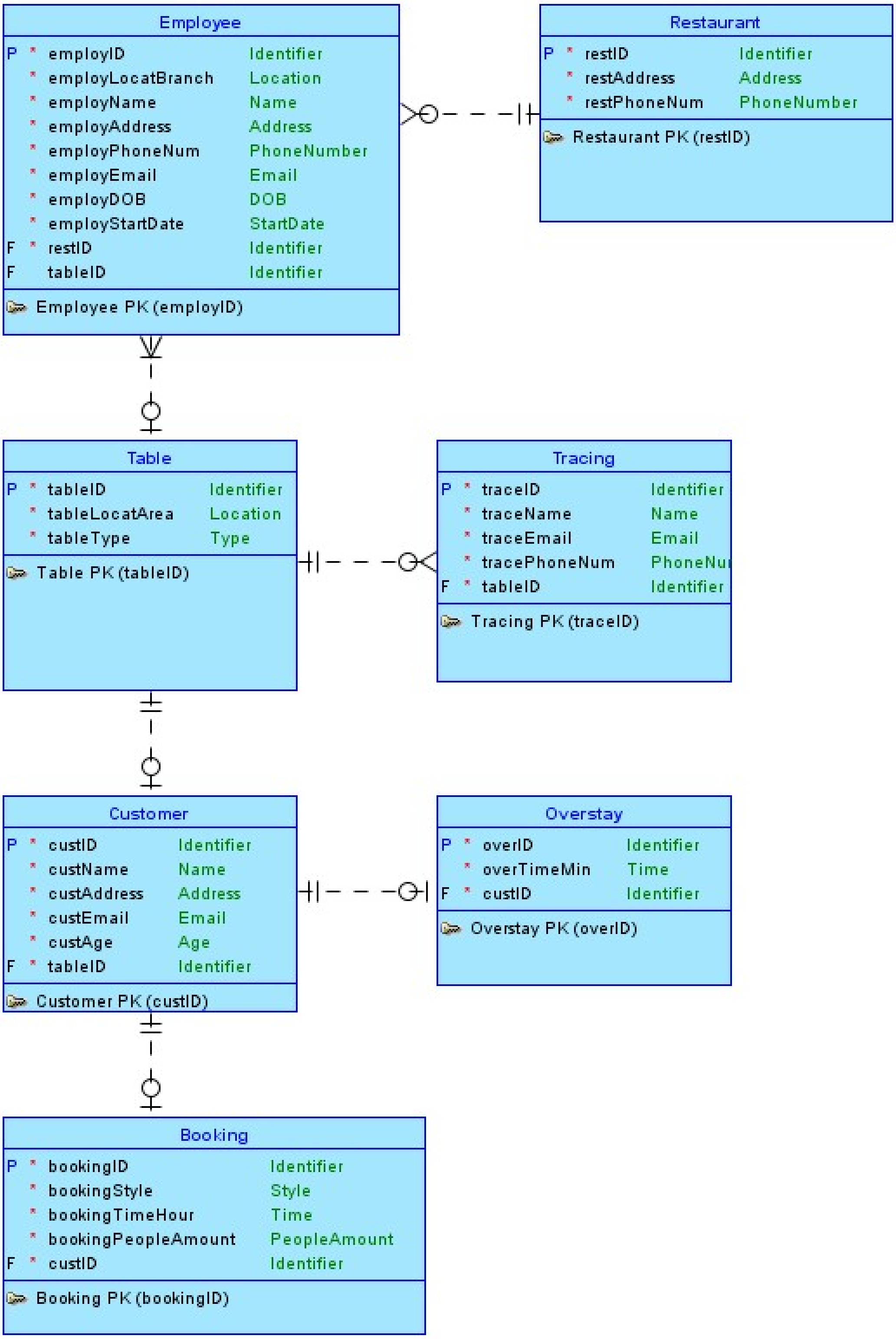
Style is type VARCHAR and size 20. Varchar is for the fact that the style contains usually letters, but this accounts for if symbols or numbers need to be used and the size assures that the style should always fit.

Time is type NUMERIC size 3. The reason for type numeric is because we're not taking actual time but some numbers that represent time in the attributes. The size is accounting for the max amount of time they should need.

To conclude we made a lot of decisions in how we decided to give each domain there type and size as it varied depending on what the domain was being used for.

Present ERD

Screenshot of Logical Model



Employee			
P	*	employID	Identifier
	*	employLocatBranch	Location
	*	employName	Name
	*	employAddress	Address
	*	employPhoneNum	PhoneNumber
	*	employEmail	Email
	*	employDOB	DOB
	*	employStartDate	StartDate
F	*	restID	Identifier
F		tableID	Identifier
		Employee PK (employID)	

Restaurant			
P	*	restID	Identifier
	*	restAddress	Address
	*	restPhoneNum	PhoneNumber
		Restaurant PK (restID)	

Table			
P	*	tableID	Identifier
	*	tableLocatArea	Location
	*	tableType	Type
		Table PK (tableID)	

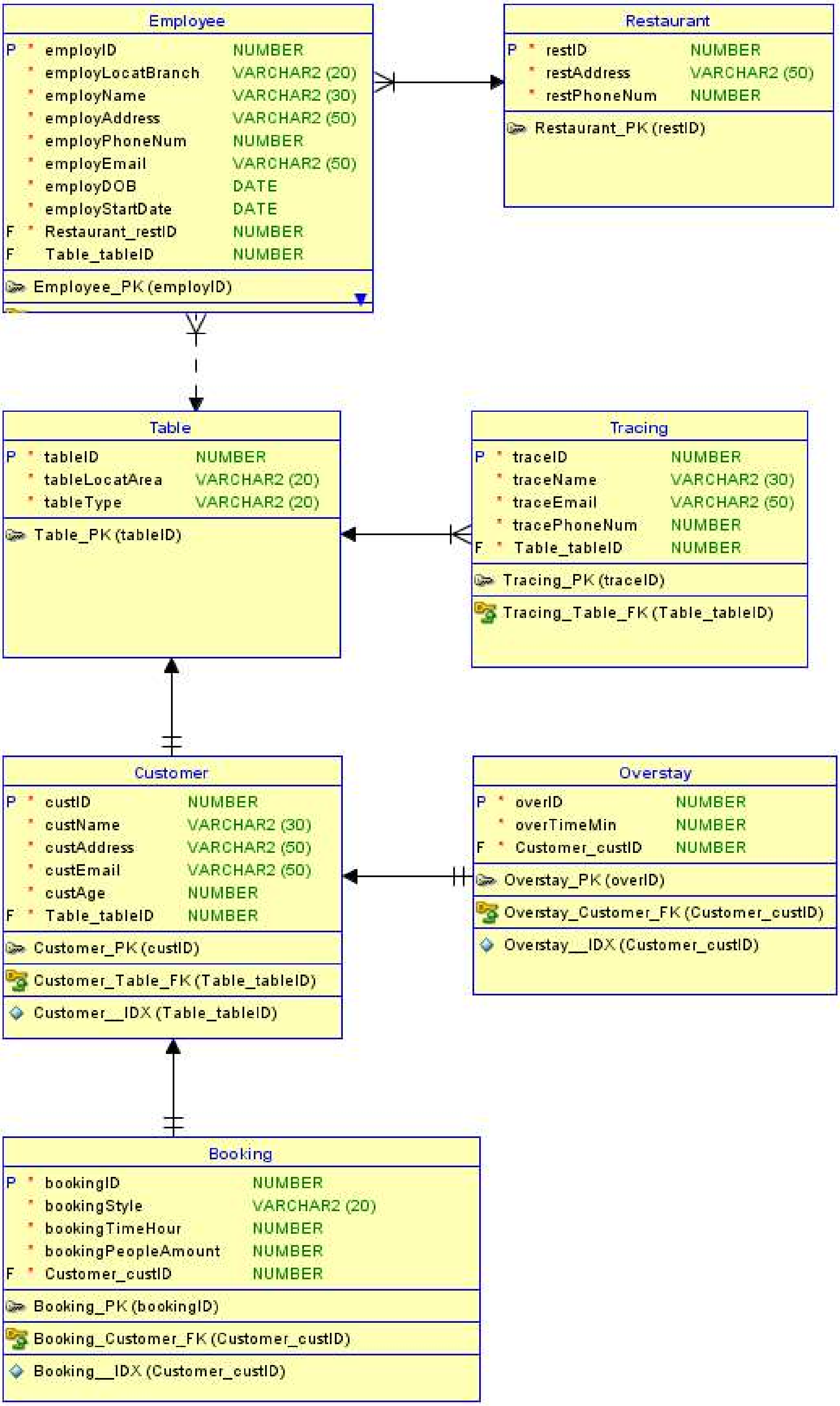
Tracing			
P	*	traceID	Identifier
	*	traceName	Name
	*	traceEmail	Email
	*	tracePhoneNum	PhoneNu
F	*	tableID	Identifier
		Tracing PK (traceID)	

Customer			
P	*	custID	Identifier
	*	custName	Name
	*	custAddress	Address
	*	custEmail	Email
	*	custAge	Age
F	*	tableID	Identifier
		Customer PK (custID)	

Overstay			
P	*	overID	Identifier
	*	overTimeMin	Time
F	*	custID	Identifier
		Overstay PK (overID)	

Booking			
P	*	bookingID	Identifier
	*	bookingStyle	Style
	*	bookingTimeHour	Time
	*	bookingPeopleAmount	PeopleAmount
F	*	custID	Identifier
		Booking PK (bookingID)	

Screenshot of Physical Model



Employee

P	*	employID	NUMBER
	*	employLocatBranch	VARCHAR2 (20)
	*	employName	VARCHAR2 (30)
	*	employAddress	VARCHAR2 (50)
	*	employPhoneNum	NUMBER
	*	employEmail	VARCHAR2 (50)
	*	employDOB	DATE
	*	employStartDate	DATE
F	*	Restaurant_restID	NUMBER
F		Table_tableID	NUMBER

Employee_PK (employID)

Restaurant

P	*	restID	NUMBER
	*	restAddress	VARCHAR2 (50)
	*	restPhoneNum	NUMBER

Restaurant_PK (restID)

Table

P	*	tableID	NUMBER
	*	tableLocatArea	VARCHAR2 (20)
	*	tableType	VARCHAR2 (20)

Table_PK (tableID)

Tracing

P	*	traceID	NUMBER
	*	traceName	VARCHAR2 (30)
	*	traceEmail	VARCHAR2 (50)
	*	tracePhoneNum	NUMBER
F	*	Table_tableID	NUMBER

Tracing_PK (traceID)

Tracing_Table_FK (Table_tableID)

Customer

P	*	custID	NUMBER
	*	custName	VARCHAR2 (30)
	*	custAddress	VARCHAR2 (50)
	*	custEmail	VARCHAR2 (50)
	*	custAge	NUMBER
F	*	Table_tableID	NUMBER

Customer_PK (custID)

Customer_Table_FK (Table_tableID)

Customer__IDX (Table_tableID)

Overstay

P	*	overID	NUMBER
	*	overTimeMin	NUMBER
F	*	Customer_custID	NUMBER

Overstay_PK (overID)

Overstay_Customer_FK (Customer_custID)

Overstay__IDX (Customer_custID)

Booking

P	*	bookingID	NUMBER
	*	bookingStyle	VARCHAR2 (20)
	*	bookingTimeHour	NUMBER
	*	bookingPeopleAmount	NUMBER
F	*	Customer_custID	NUMBER

Booking_PK (bookingID)

Booking_Customer_FK (Customer_custID)

Booking__IDX (Customer_custID)

Screenshot of details that you have incorporated into your model such as unique or check constraints.

Constraints

This constraint was to stop a booking for a party of more than 6 because it breaks the rules

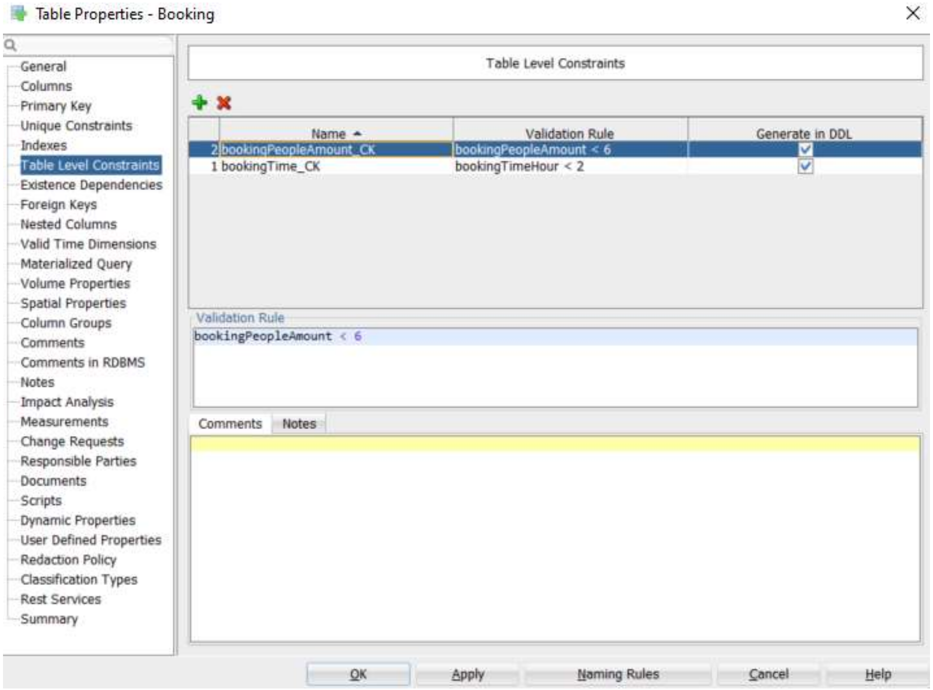


Table Properties - Booking

Table Level Constraints

	Name	Validation Rule	Generate in DDL
2	bookingPeopleAmount_CK	bookingPeopleAmount < 6	☑
1	bookingTime_CK	bookingTimeHour < 2	☑

Validation Rule

```
bookingPeopleAmount < 6
```

This Constraint is to make sure they can't book a slot for more than 2 hours as it is against the rules set in place.

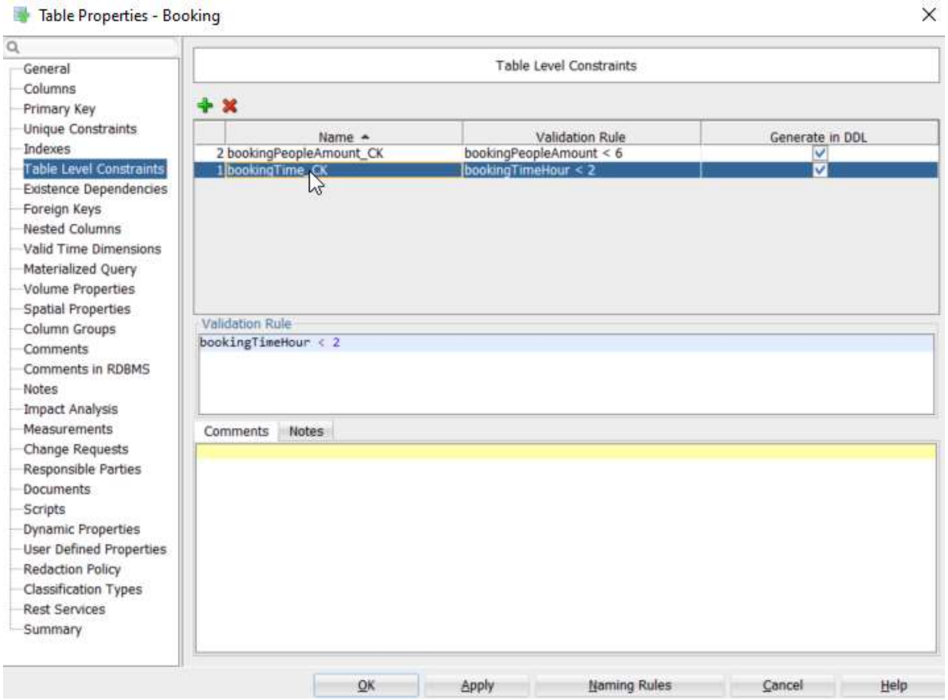


Table Properties - Booking

Table Level Constraints

	Name	Validation Rule	Generate in DDL
2	bookingPeopleAmount_CK	bookingPeopleAmount < 6	☑
1	bookingTime_CK	bookingTimeHour < 2	☑

Validation Rule

```
bookingTimeHour < 2
```

Constraint to make sure the customer is older than 18 for insurance reasons.

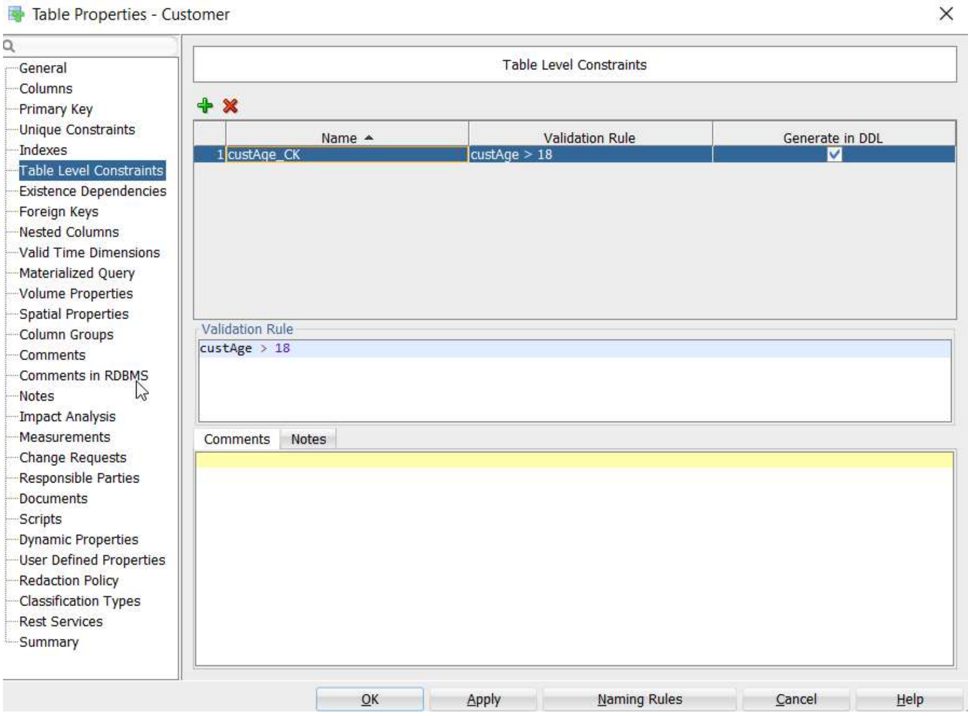


Table Level Constraints

	Name ▲	Validation Rule	Generate in DDL
1	custAge_CK	custAge > 18	☑

Validation Rule

```
custAge > 18
```

Constraint to makes sure the employees email end with @burgershack.com

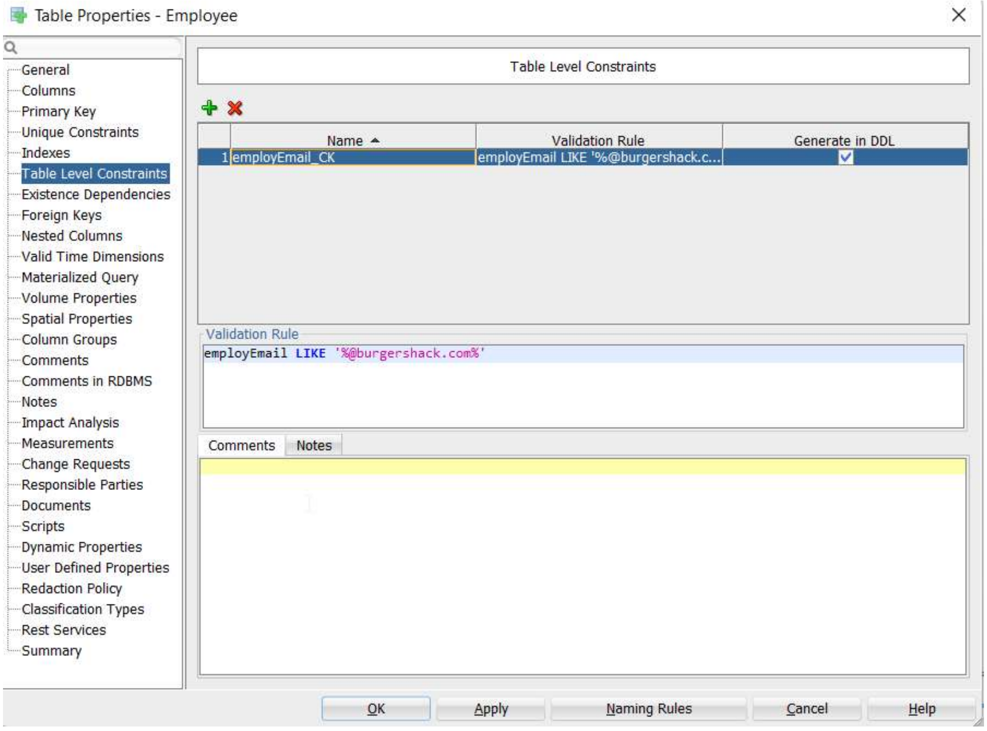


Table Level Constraints

	Name ▲	Validation Rule	Generate in DDL
1	employEmail_CK	employEmail LIKE '%@burgershack.c...	☑

Validation Rule

```
employEmail LIKE '%@burgershack.com%'
```