1. LIST FEATURES IMPLEMENTED

All business requirements were successfully implemented as below:

Requirement 1 a and b

Fully implemented using the following:

PROCEDURES

1. **Get_customer_details** Retrieves customer details given customer_id,surname,email or phone.
2. **New_customer** - Creates new customer

FUNCTIONS

1. **Check_customer_by_id**- checks existence of a customer based on id provided.
2. **Check_customer_by_surname**- checks existence of a customer based on surname.
3. **Check_customer_by_email**- checks existence of a customer based on email.
4. **Check_customer_by_phone**- checks existence of a customer based on phone number.

Requirement 1 c and d

Fully implemented using the following:

PROCEDURES

1. **Add_booking_by_id** Makes new booking based on customer_id entered.
2. **Add_booking_by_surname** Makes new booking based on customer surname if id is forgotten.
3. **Add_booking_by_email** Makes new booking based on customer_email if id is forgotten.
4. **Add_booking_by_phone** Makes new booking based on customer_phone number if id is forgotten.
5. **Check_start_date** Checks if startdate is equal to current date.

Requirement 2

Fully implemented using the following:

PROCEDURES

1. **Cancel_booking** Cancels open/confirmed bookings

Requirement 3

Fully implemented using the following:

PROCEDURES

1. **Add_new_vehicle** Creates a new vehicle into the database by adding vehicle details in vehicle, manufacturer and model tables.

Requirement 4

Fully implemented using the following:

TRIGGER

1. **Secure_booking_statuscode** Trigger enforces business rule only open/confirmed bookings can be cancelled .

Requirement 5

Fully implemented using the following:

FUNCTIONS

1. **daily_bookings_report-** prints report on bookings as per date provided.

VIEWS

1. **Available_vehicles view**-Selects list of vehicles from vehicles and booking tables.
2. **Booking_view**- shows booking list- used as report source

Requirement 6

Fully implemented. Appropriate error handling incorporated in the all the code.

2. IMPLEMENTATION

a. Overall Design

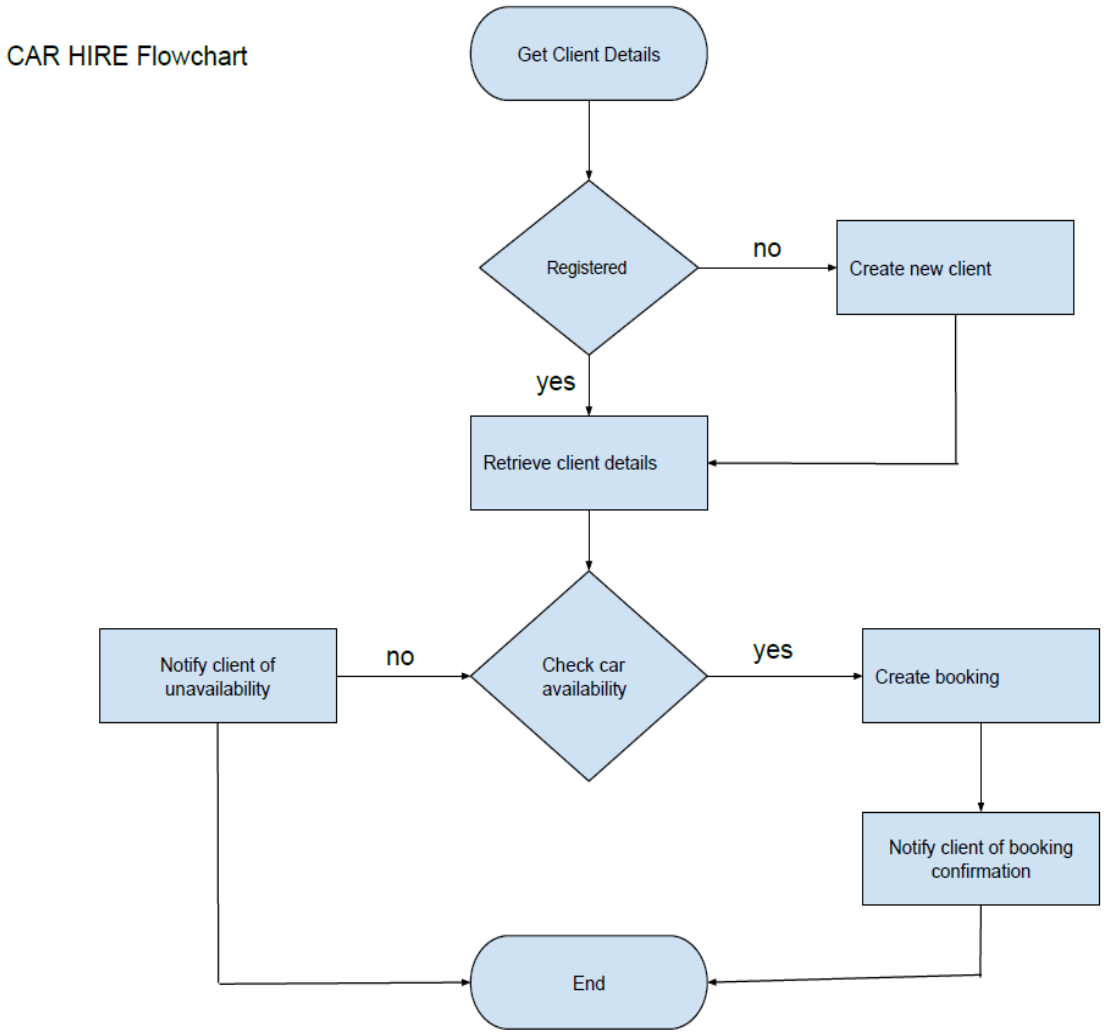


Figure 1

Structure

The implantation structure used was to have one CAR_HIRE package that holds procedures and functions required to run the application. Views were used to consolidate data for report generation and triggers to compliment functionality such as enforce business rule of cancelling only open/confirmed bookings and inserting a record when it does not exist in customers table.

Package structure

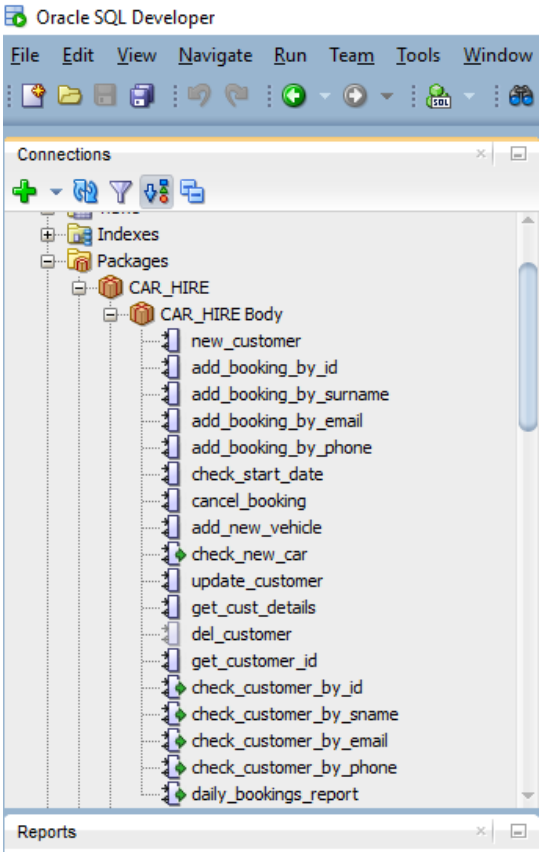


Figure 2

Views

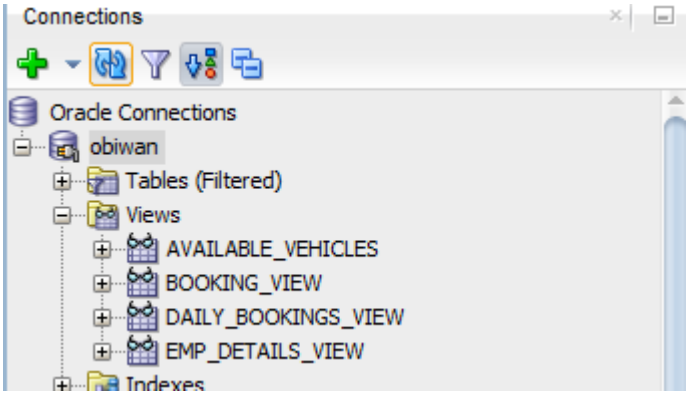


Figure 3

Triggers

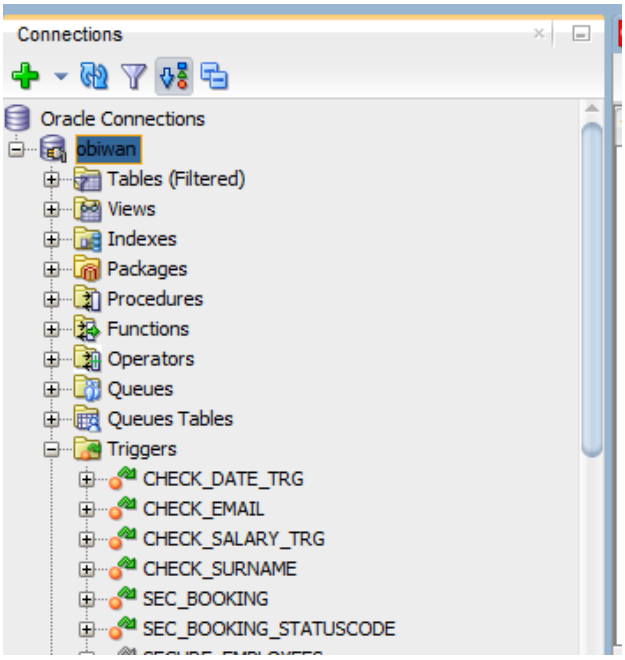


Figure 4

b. Detailed Development

1. The system should be able to recognise if the customer is new or existing based on the information provided by the customer.

   a. If a new customer is submitting a car hire request, they must be registered by the system as a part of the booking process.

checking for existing customer

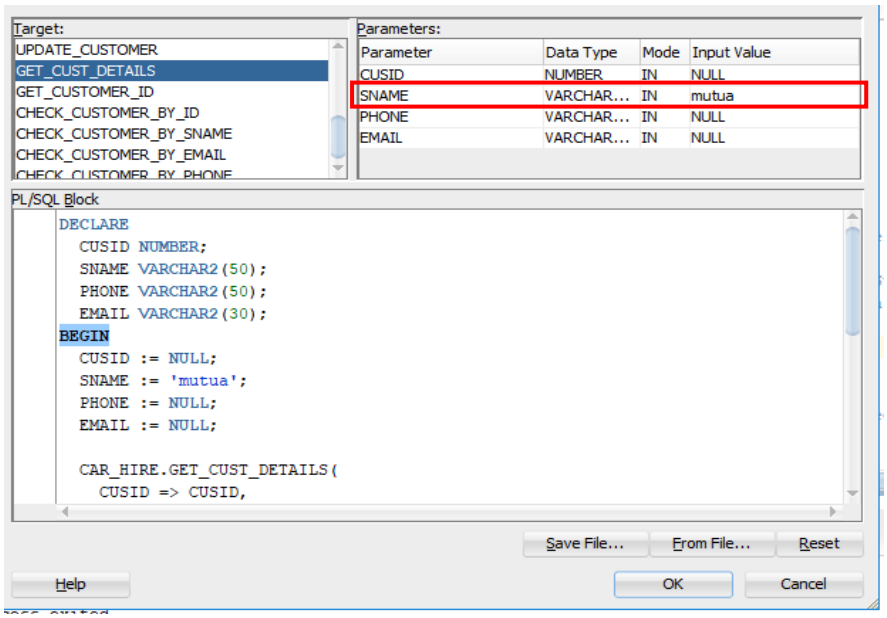


Figure 5

results

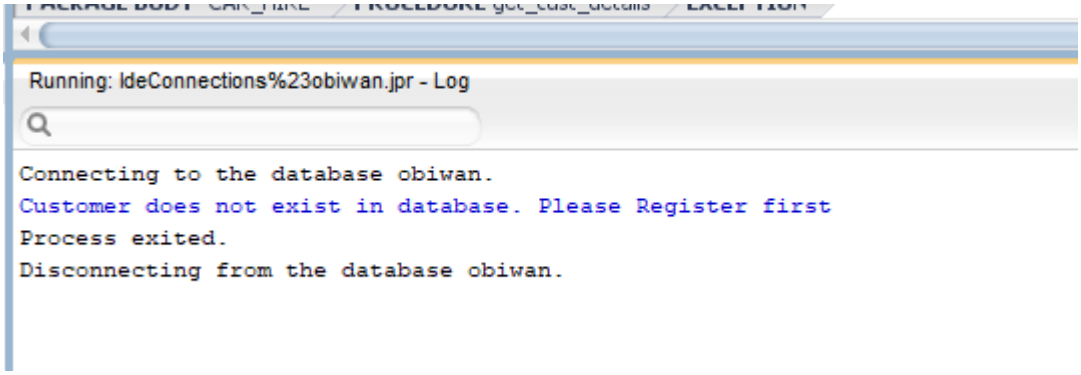


Figure 6

b. If an existing customer is submitting a car hire request their details should be taken from the customer table.

Checking for existing customer

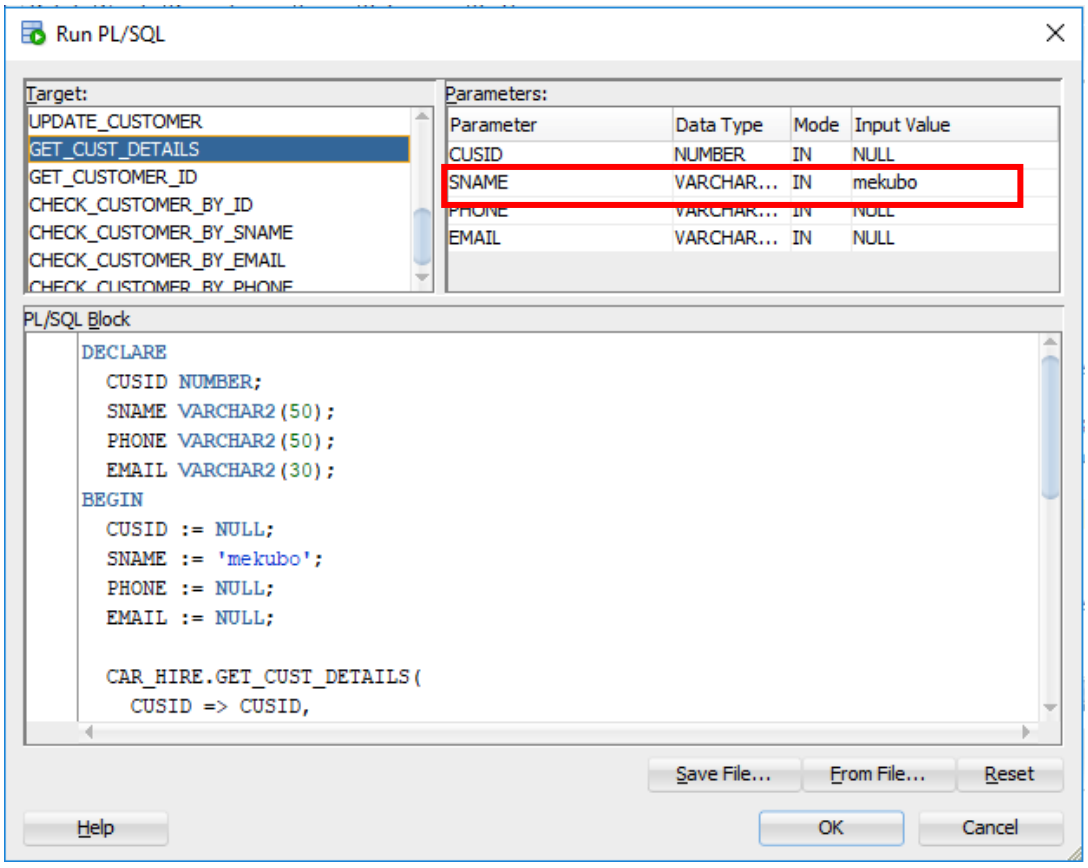


Figure 7

results

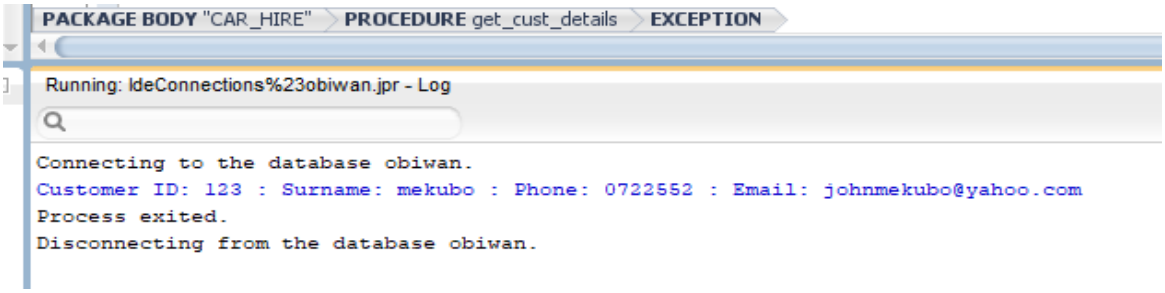


Figure 8

The process should use the existing sequence book_id_seq to generate a new booking id automatically. After the booking has been made it should also print on the screen an appropriate confirmation message including the customer id, the booking id as well as car hiring details (simulation of sending booking confirmation to the customer via email).

On receiving a car hire enquiry, the company's process is required to check the availability of requested cars and vans for dates specified:

c. If there are vehicles of a requested category available, the customer's details are recorded (if not stored already) and a new booking is made and one of the available cars (using registration number) is assigned to the booking. An appropriate confirmation message is displayed on the screen. Customers are not required to pay at this stage and will pay for the vehicle at the time of pick up.

Booking by customer id

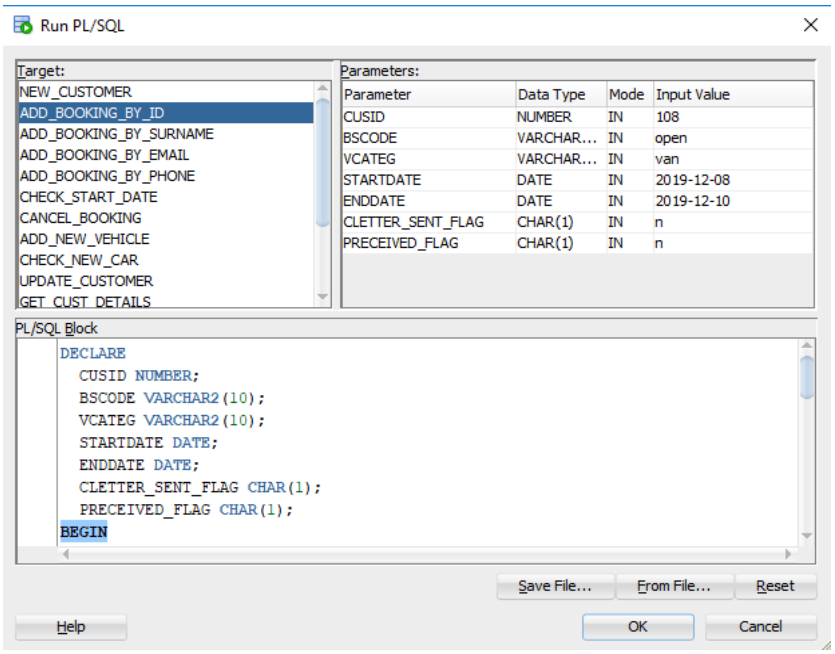


Figure 9

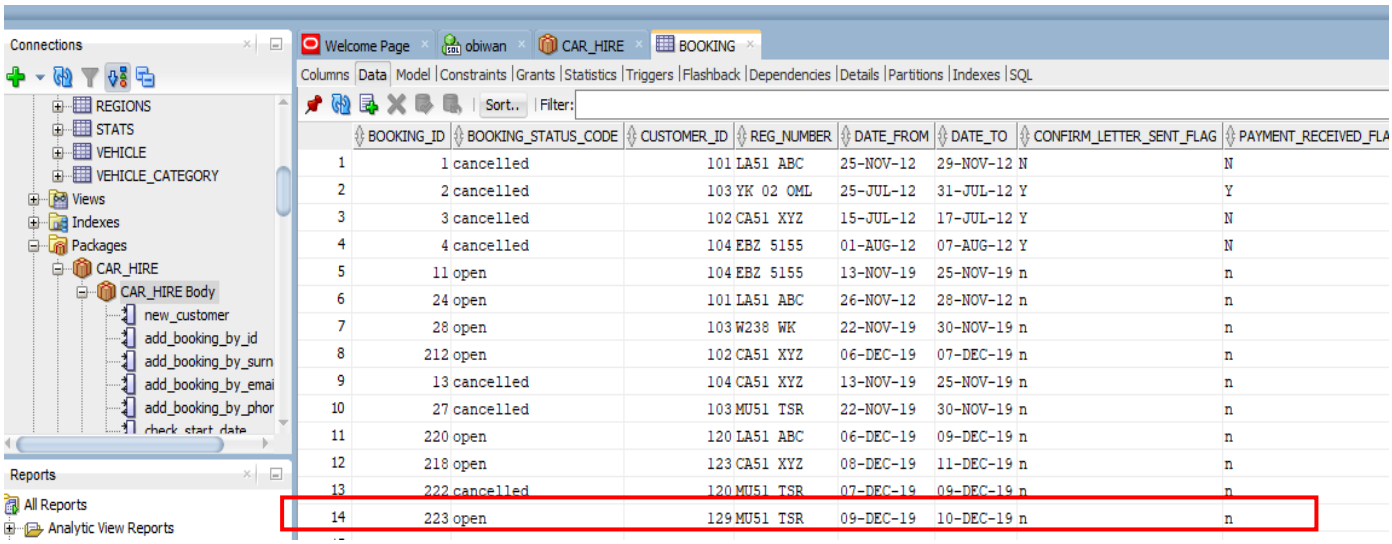
Results

```
Car Registration Number W238 WK is available
Please note that booking is made 1 day from booking date(Today)
Car Registration Number W238 WK has been booked from 09-DEC-19 To: 10-DEC-19
```

Figure 10

3. A customer should be able to cancel an existing booking. Cancelling the highlighted record

Record before cancellation



	BOOKING_ID	BOOKING_STATUS_CODE	CUSTOMER_ID	REG_NUMBER	DATE_FROM	DATE_TO	CONFIRM_LETTER_SENT_FLAG	PAYMENT_RECEIVED_FLA
1	1	cancelled	101	LA51 ABC	25-NOV-12	29-NOV-12	N	N
2	2	cancelled	103	YK 02 OML	25-JUL-12	31-JUL-12	Y	Y
3	3	cancelled	102	CA51 XYZ	15-JUL-12	17-JUL-12	Y	N
4	4	cancelled	104	EBZ 5155	01-AUG-12	07-AUG-12	Y	N
5	11	open	104	EBZ 5155	13-NOV-19	25-NOV-19	n	n
6	24	open	101	LA51 ABC	26-NOV-12	28-NOV-12	n	n
7	28	open	103	W238 WK	22-NOV-19	30-NOV-19	n	n
8	212	open	102	CA51 XYZ	06-DEC-19	07-DEC-19	n	n
9	13	cancelled	104	CA51 XYZ	13-NOV-19	25-NOV-19	n	n
10	27	cancelled	103	MU51 TSR	22-NOV-19	30-NOV-19	n	n
11	220	open	120	LA51 ABC	06-DEC-19	09-DEC-19	n	n
12	218	open	123	CA51 XYZ	08-DEC-19	11-DEC-19	n	n
13	222	cancelled	120	MU51 TSR	07-DEC-19	09-DEC-19	n	n
14	223	open	129	MU51 TSR	09-DEC-19	10-DEC-19	n	n

Figure 11

results

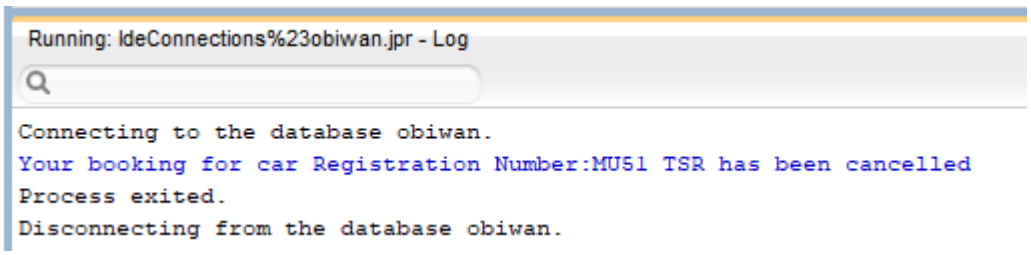


Figure 12

Record after cancellation

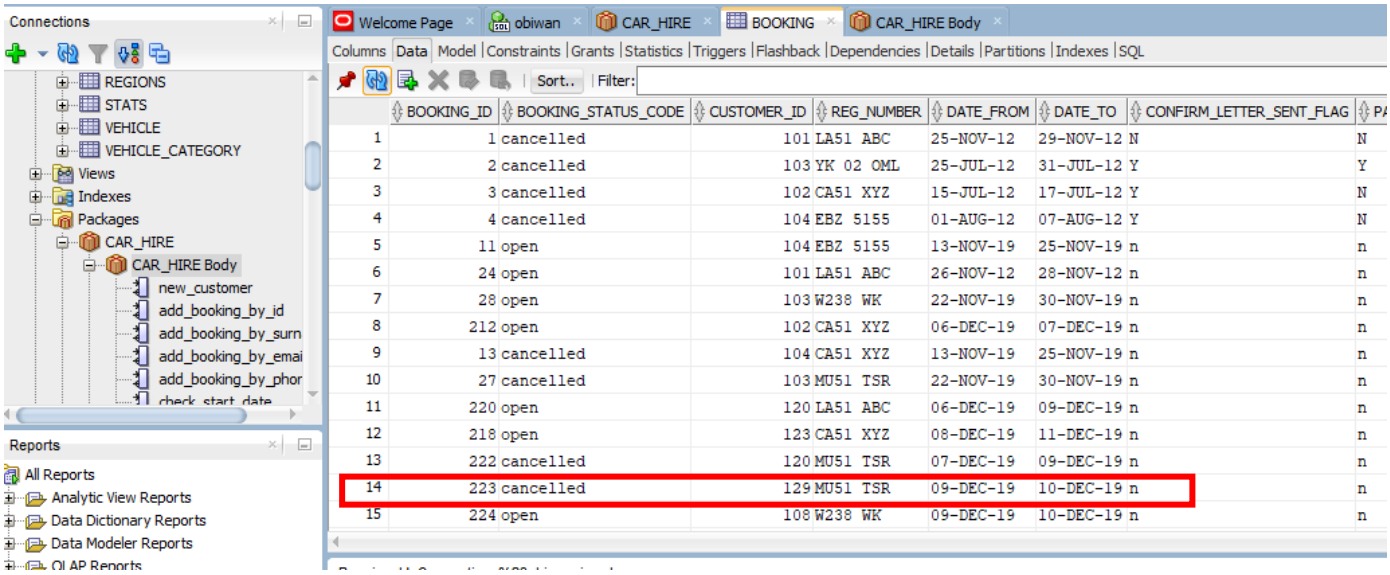


Figure 13

3. The system should be able to record all necessary information about new cars added to the company fleet.

When a new vehicle is created, apart from the vehicle, model and manufacturer tables are updated with necessary details of the same vehicle.

capture vehicle details screen

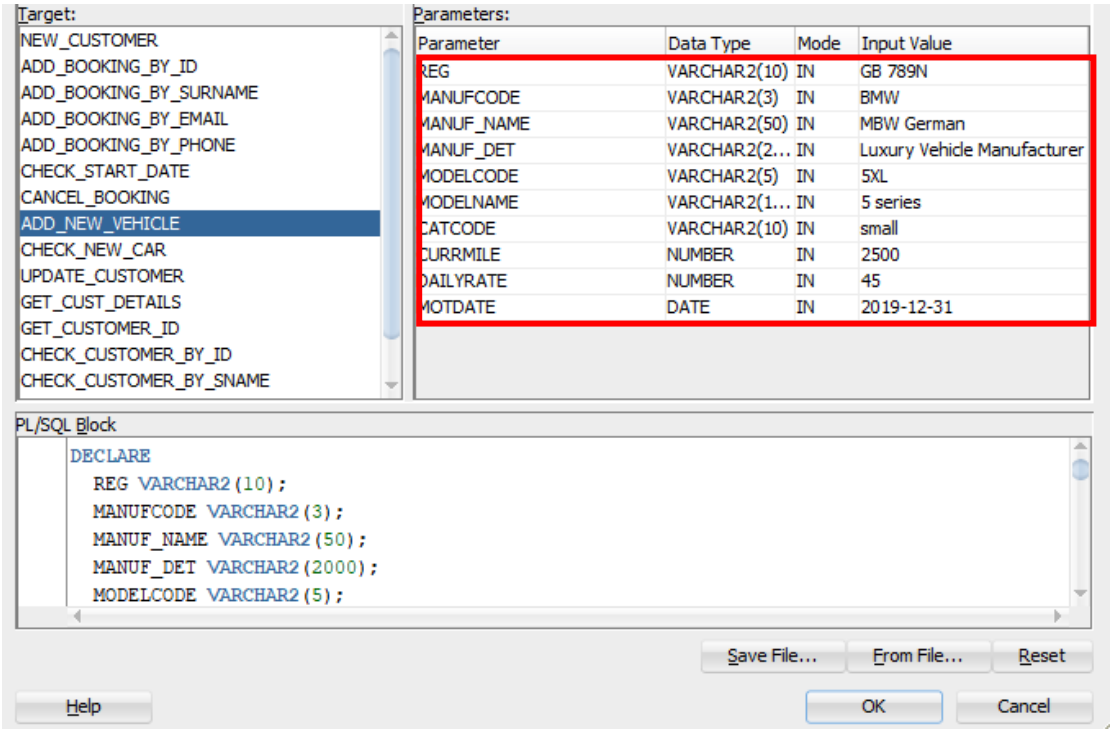


Figure 14

Results

```
Running: IdeConnections%23obiwan.jpr - Log
Connecting to the database obiwan.
Vehicle Registration Number GB 789N details added into manufacturer table
Vehicle Registration Number GB 789N details inserted into model table
Vehicle Registration Number GB 789N details inserted into vehicle table
Process exited.
Disconnecting from the database obiwan.
```

Figure 15

Updated manufacturer table

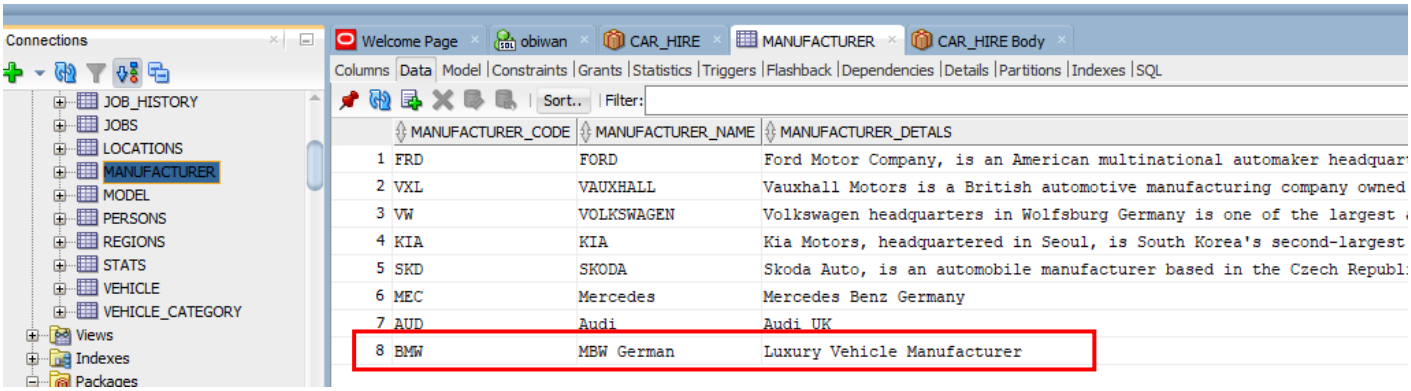


Figure 16

Updated model table

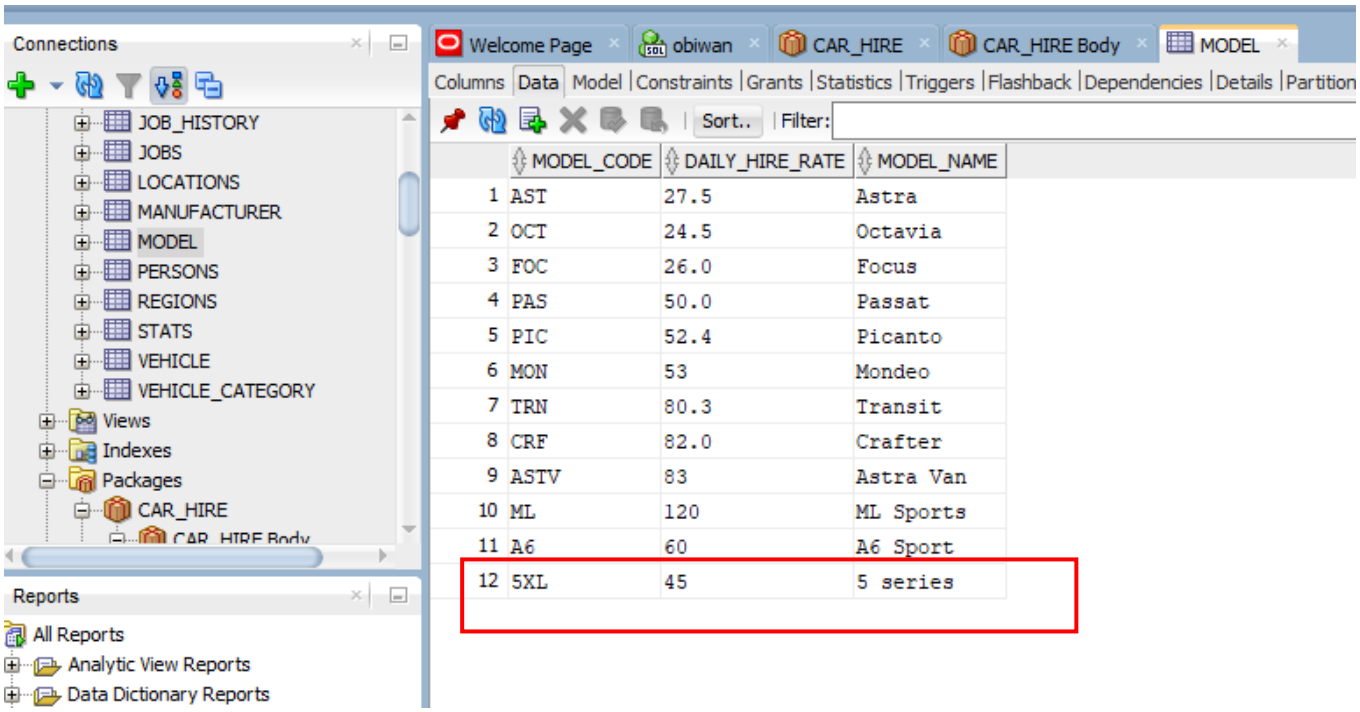


Figure 17

4. The company has a business rule: ‘If the status of the booking is ‘open’, or ‘confirmed’ it can be changed, but the bookings with the ‘cancelled’ cannot be changed’. You must design and implement the best way to enforce this business rule.

This section was implemented using a trigger to check status of sec_booking_statuscode column in booking table. If the status is “open” or “confirmed”, the record may be edited otherwise record is locked for edit status is” cancelled”.

sec_booking_statuscode trigger

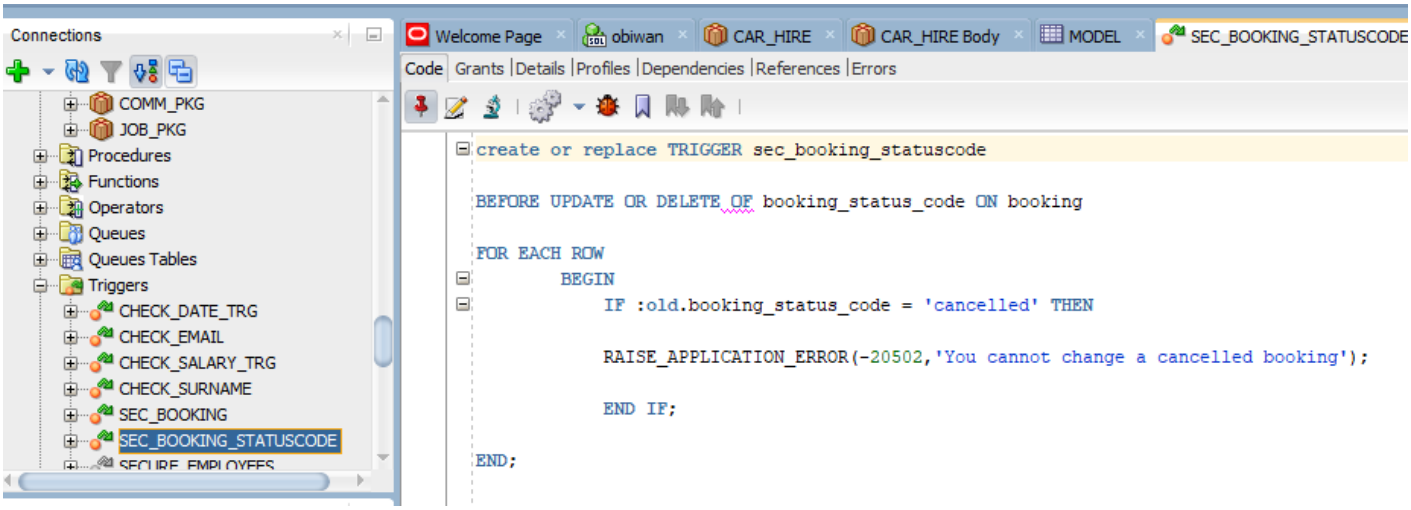


Figure 18

changing highlighted record status to open

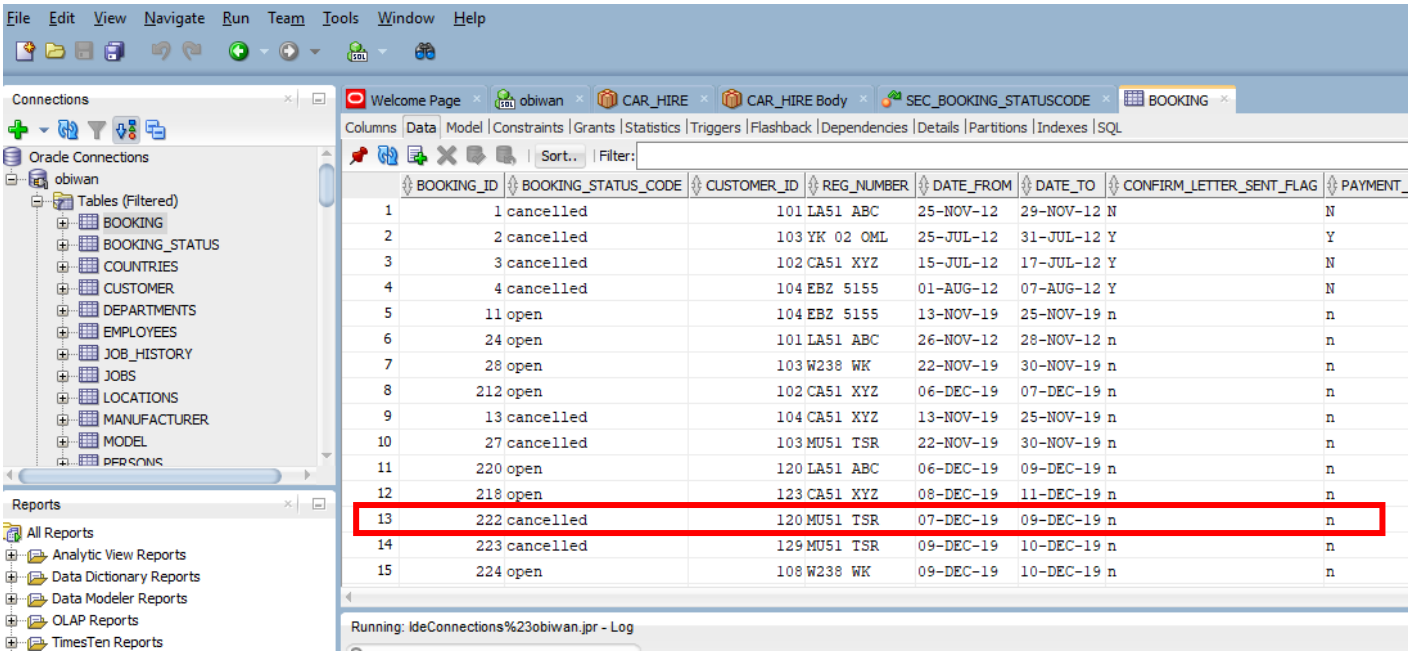


Figure 19

Error thrown after trying to edit record

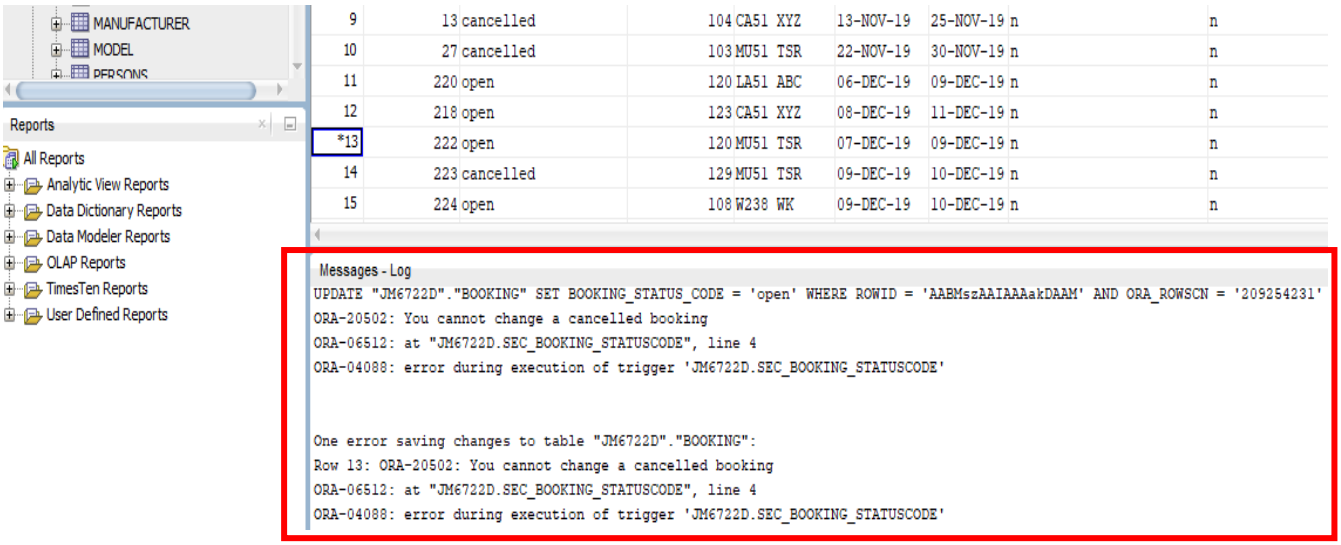


Figure 20

5. A company's clerk should be able to print a report showing all bookings for any day.

Sample report generated.

This requirement was implemented using function **daily_bookings_report.** When executed, it displays a prompt for user to enter date and on clicking apply the report is displayed.

business rule.

Enter date prompt

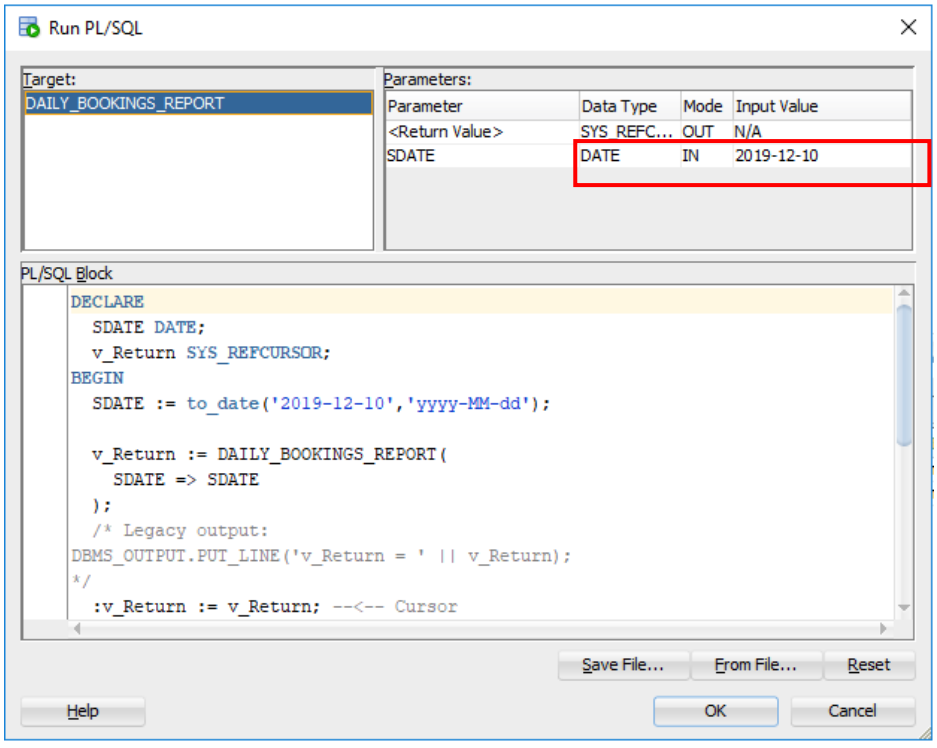


Figure 21

daily_bookings_report- Note that you have to click on output variables tab to see the report.

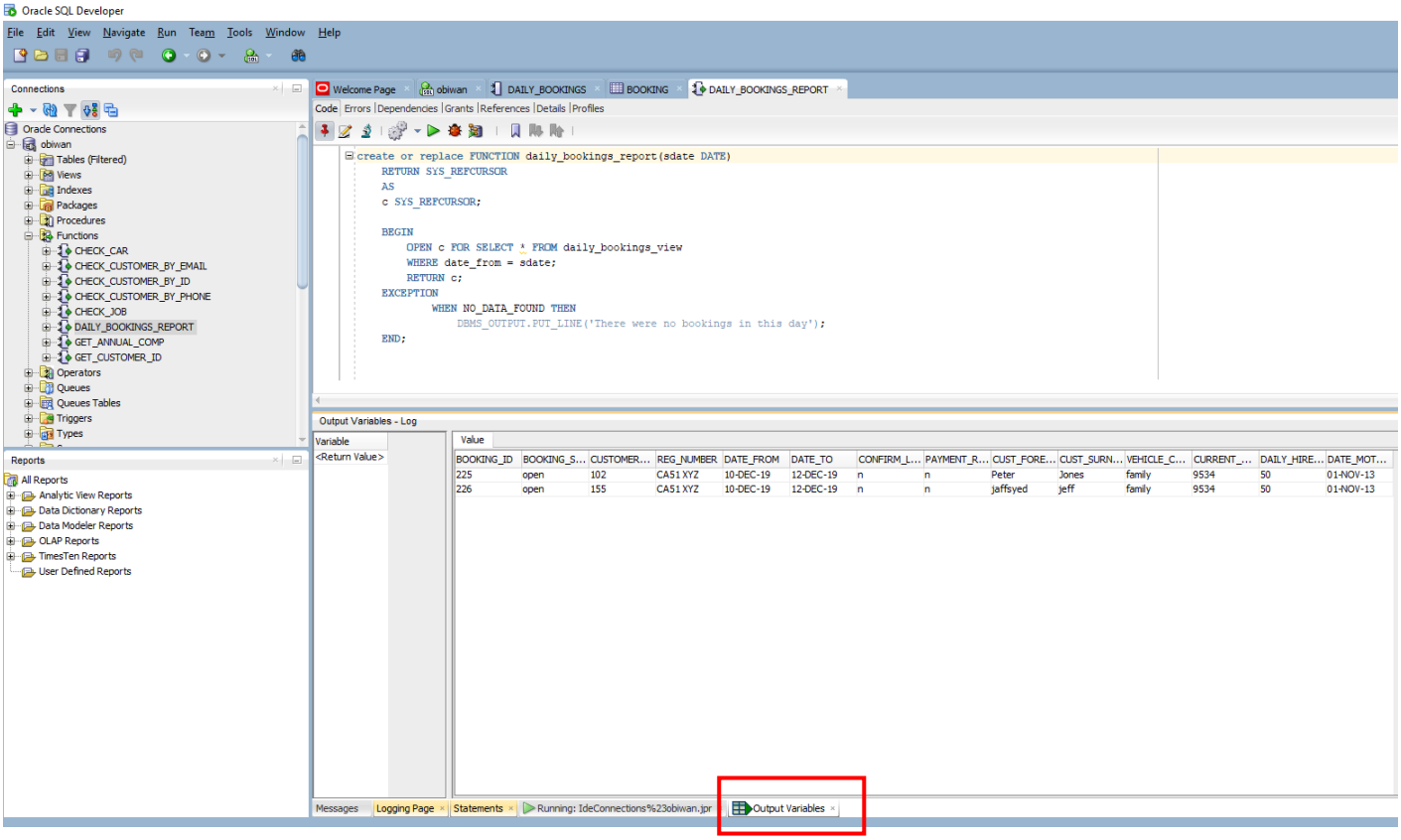


Figure 22

6. All your PL/SQL subprograms should include appropriate error handling.

Appropriate error handling incorporated in the code to capture exceptions that may occur.

Error Handling

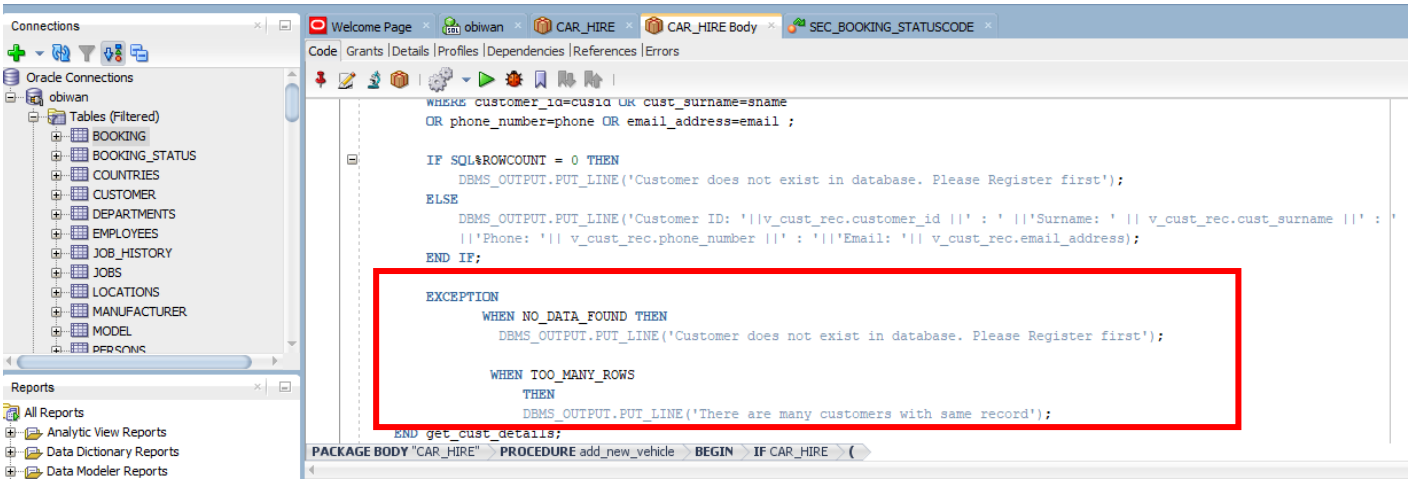


Figure 23