

# IS 420 Project

## Group 10

Name & Procedures:

April Raimond 7,8,9,10,15

James Thang 5,6,16,17,18,19

Robert Galvan 3,4, 20,21,22,23

Ray Rasolofonera 1,2,11,12,13,14

(Ray Rasolofonera, MEMBER 1)

1. Add a new hotel: Create a new hotel with appropriate information about the hotel as input parameters

```
--Create a function that Insert Hotel  
create or replace procedure Insert_hotel (
```

```
--Create variable that has the same datatype as the Column
```

```
input_id in hotel.hotel_id%type,
```

```
input_name in hotel.hotel_name%type,
```

```
input_address in hotel.address%type,
```

```
input_city in hotel.city%type,
```

```
input_state in hotel.state%type,
```

```
input_zip in hotel.zip%type,
```

```
input_rate in hotel.hotel_rate%type,
```

```
input_phone_number in hotel.phone_number%type,
```

```
input_is_active in hotel.is_active%type) -- Check if the
```

```
Is
```

```
--Start insertions
```

```
BEGIN
```

```
INSERT INTO HOTEL(hotel_id, hotel_name, address, city, state,zip,
```

```
Hotel_rate, phone_number, is_active)
```

```
VALUES
```

```
(input_id, input_name, input_address,
```

```
input_city, input_state, input_zip,
```

```
input_rate, input_phone_number,
```

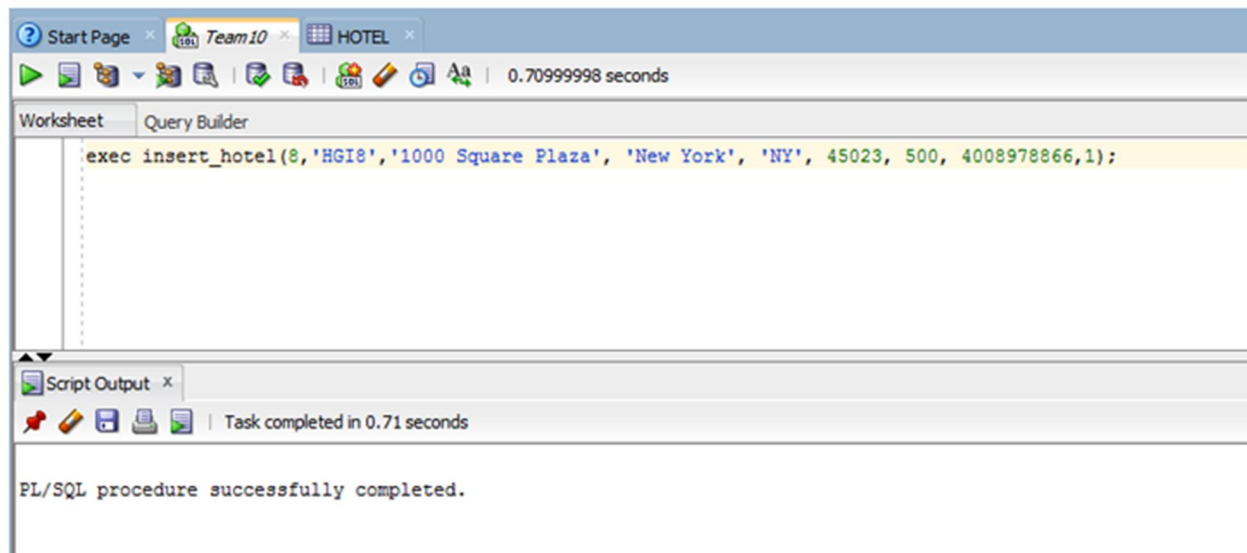
```
input_is_active);
```

```
End;
```

```
--Function call
```

```
exec insert_hotel(8,'HGI8','1000 Square Plaza', 'New York', 'NY', 45023, 500, 4008978866,1);
```

```
--Delete from Hotel where Hotel_ID = 8; This will be used for the presentation
```



(Ray Rasolofonera, MEMBER 1)

2) Find a hotel: Provide as input the address of the hotel and return its hotel ID

--Create a function that will get given address as input and display its ID

create or replace procedure get\_hotel\_ID(input\_address in varchar2, input\_state in varchar2,  
input\_zip in number) is

store\_hotel\_id number;

begin

-- Get the given address

select hotel\_id into store\_hotel\_id from hotel where address=input\_address and state=input\_state  
and zip=input\_zip;

dbms\_output.put\_line('Hotel ID: ' || store\_hotel\_id);

Exception -- In case of not finding the hotel

when too\_many\_rows then

dbms\_output.put\_line('Many hotels were found');

when no\_data\_found then

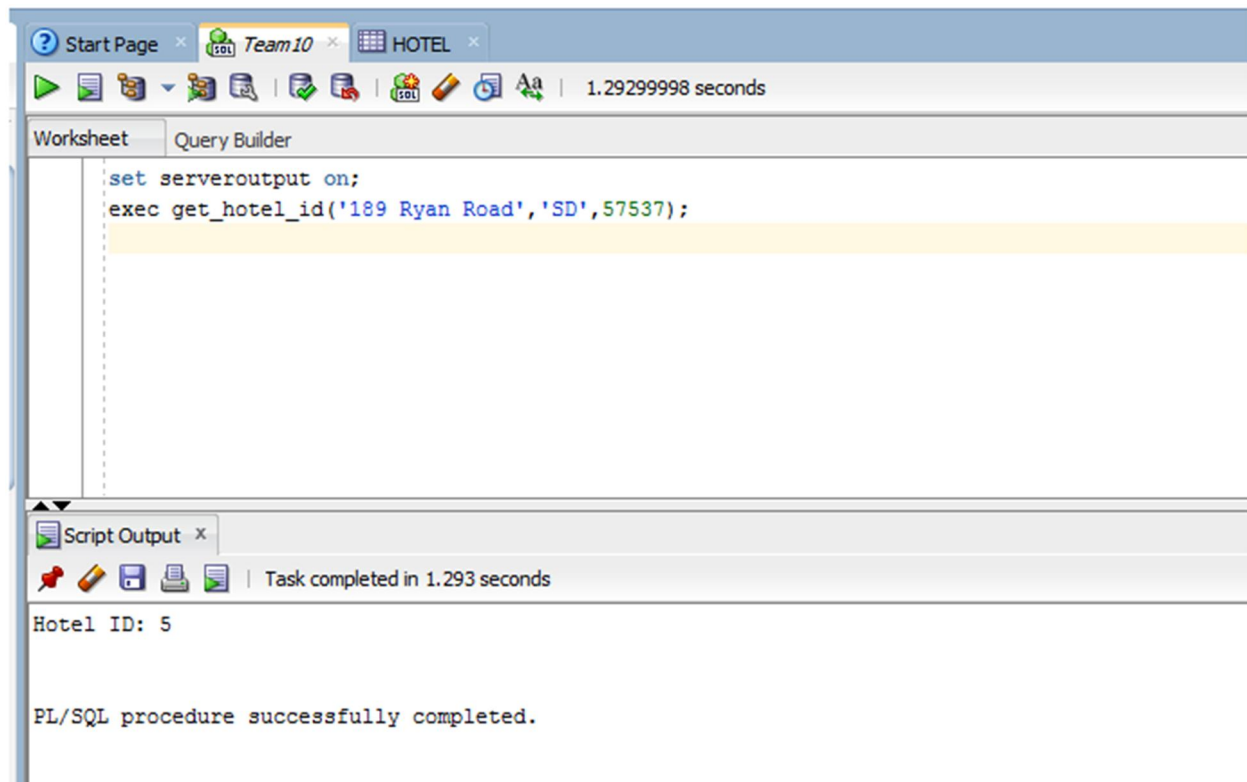
dbms\_output.put\_line('No hotel found.');

End;

-- Function call

set serveroutput on;

exec get\_hotel\_id('189 Ryan Road','SD',57537);



(April Raimond, Member 2)

7) Make a reservation: Input parameters: Hotel ID, guest's name, start date, end date, room type, date of reservation, etc. Output: reservation ID (this is called confirmation code in real-life).

NOTE: Only one guest per reservation. However, the same guest can make multiple reservations.

create or replace procedure create\_reservation

```
(  
  hotel_id_input in hotel.hotel_id%type,  
  guest_name_input in customer.customer_name%type,  
  start_date_input in reservation.res_start_date%type,  
  res_end_input in reservation.res_end_date%type,  
  room_type_input in room.room_type%type,  
  new_cus_phone in customer.customer_phone_num%type,  
  new_cus_address in customer.address%type,  
  new_cus_city in customer.city%type,  
  new_cus_state in customer.state%type,  
  new_cus_zip_code in customer.zip%type,  
  req_services in customer.customer_services_type%type,  
  new_cus_cc in customer.credit_card%type,  
  new_cus_services_date in customer.customer_services_date%type  
)
```

is

out\_res\_id number;

BEGIN

out\_res\_id:=res\_num\_seq.nextval;

INSERT INTO

CUSTOMER(customer\_id,customer\_name,address,city,state,zip,customer\_phone\_num,credit\_card,customer\_services\_date,  
customer\_services\_type)

VALUES(customer\_num\_seq.nextval,guest\_name\_input,new\_cus\_address,new\_cus\_city,new\_cus\_state,new\_cus\_zip\_code,

new\_cus\_phone,new\_cus\_cc,new\_cus\_services\_date,req\_services);

DBMS\_OUTPUT.PUT\_LINE('The reservation id is: '||res\_num\_seq.nextval);

END;

set serveroutput on;

exec create\_reservation(6, 'Andrew Bloom', Date '2018-01-07', Date '2018-05-07', 'suite',  
'4439860424', '3201 Patterson Ave',  
'Baltimore', 'MD',21224,'L',4322491103220499,Date '2018-01-07');

Procedure CREATE\_RESERVATION compiled

Error starting at line : 35 in command -

BEGIN create\_reservation(6, 'Andrew Bloom', Date '2018-01-07', Date '2018-05-07', 'suite', '4439860424', '3201 Patterson Ave',; END;

Error report -

ORA-06550: line 1, column 127:

PLS-00103: Encountered the symbol ";" when expecting one of the following:

```
( - + case mod new not null <an identifier>
<a double-quoted delimited-identifier> <a bind variable>
continue avg count current exists max min prior sql stddev
sum variance execute forall merge time timestamp interval
date <a string literal with character set specification>
<a number> <a single-quoted SQL string> pipe
<an alternatively-quoted string literal with character set specification>
<an alternatively
```

06550. 00000 - "line %s, column %s:\n%s"

\*Cause: Usually a PL/SQL compilation error.

\*Action:

Error starting at line : 36 in command -

'Baltimore', 'MD',21224,'L',4322491103220499,Date '2018-01-07')

Error report -

Unknown Command

(April Raimond, Member 2)

8) Find a reservation: Input is guest's name and date, hotel ID. Output is reservation ID

```
create or replace procedure find_reservation
(guest_name_input customer.customer_name%type,
date_input reservation.res_start_date%type,
hotel_id_input hotel.hotel_id%type)
is
    find_id number;
    out_res_id number;

begin
    select customer_id into find_id from customer where guest_name_input =
customer.customer_name;
    select reservation_id into out_res_id from reservation where find_id = customer_id;
    dbms_output.put_line('Reservation id: ' || out_res_id);
EXCEPTION
when no_data_found then
    dbms_output.put_line('None found');
End;

set serveroutput on;
exec find_reservation('William Wing',DATE '2018-12-06',2);
```

Worksheet    Query Builder

```
--April Raimond
/** Find a reservation: Input is guest's name and date, hotel ID. Output is reservation ID
**/

create or replace procedure find_reservation
(guest_name_input customer.customer_name%type,
date_input reservation.res_start_date%type,
hotel_id_input hotel.hotel_id%type)
is
find_id number;
out_res_id number;

begin
select customer_id into find_id from customer where guest_name_input = customer.customer_name;
select reservation_id into out_res_id from reservation where find_id = customer_id;
dbms_output.put_line('Reservation id: ' || out_res_id);
EXCEPTION
when no_data_found then
dbms_output.put_line('None found');
End;

set serveroutput on;
exec find_reservation('William Wing',DATE '2018-12-06',2);
```

Script Output x

Task completed in 0.222 seconds

Procedure FIND\_RESERVATION compiled

Reservation id: 3

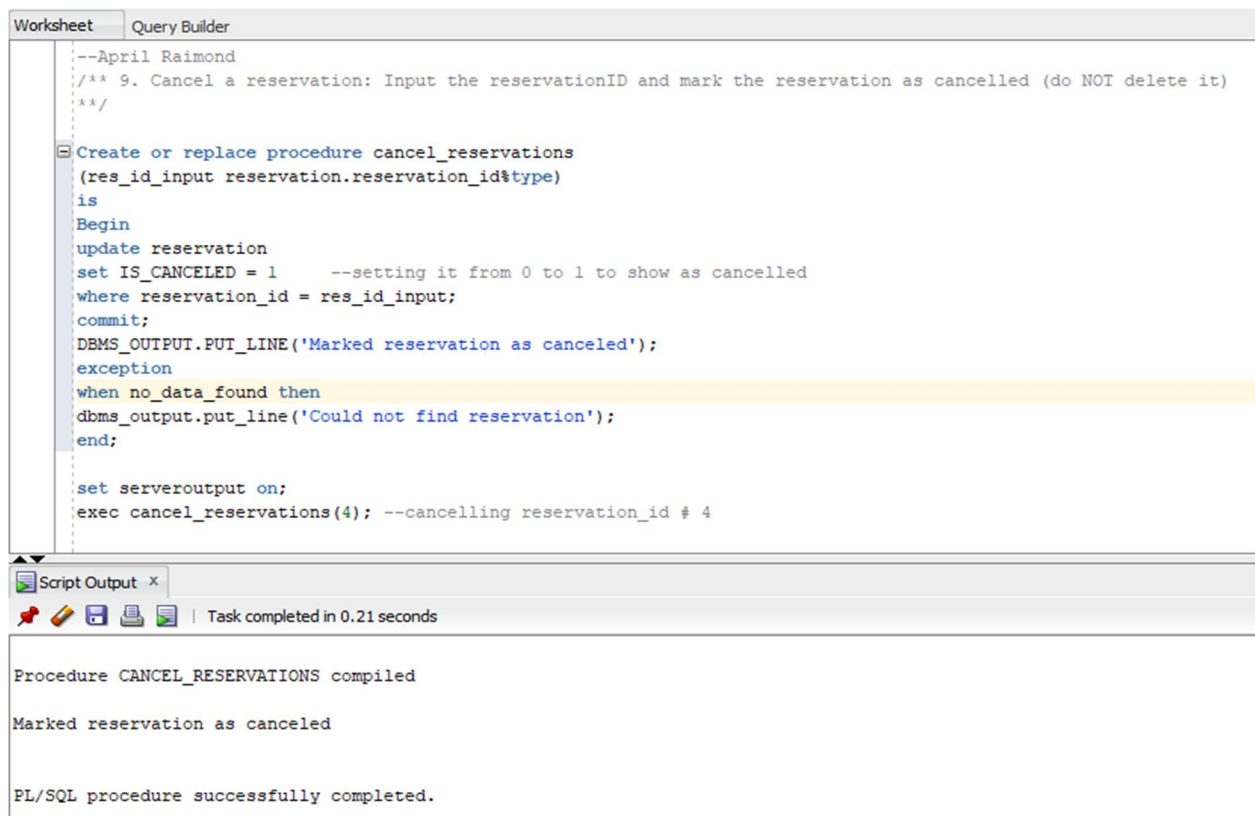
PL/SQL procedure successfully completed.

(April Raimond, Member 2)

9) Cancel a reservation: Input the reservationID and mark the reservation as cancelled (do NOT delete it)

```
Create or replace procedure cancel_reservations
(res_id_input reservation.reservation_id%type)
is
Begin
update reservation
set IS_CANCELED = 1    --setting it from 0 to 1 to show as cancelled
where reservation_id = res_id_input;
commit;
DBMS_OUTPUT.PUT_LINE('Marked reservation as canceled');
exception
when no_data_found then
dbms_output.put_line('Could not find reservation');
end;

set serveroutput on;
exec cancel_reservations(4); --cancelling reservation_id # 4
```



The screenshot displays the SQL Developer environment. The top pane, titled 'Worksheet' and 'Query Builder', contains a PL/SQL script. The script includes a comment for April Raimond, a procedure definition for canceling a reservation, and an execution command. The bottom pane, titled 'Script Output', shows the results of the script execution, including the compilation of the procedure, the output message 'Marked reservation as canceled', and a confirmation that the PL/SQL procedure was successfully completed.

```
--April Raimond
/** 9. Cancel a reservation: Input the reservationID and mark the reservation as cancelled (do NOT delete it)
**/

Create or replace procedure cancel_reservations
(res_id_input reservation.reservation_id%type)
is
Begin
update reservation
set IS_CANCELED = 1    --setting it from 0 to 1 to show as cancelled
where reservation_id = res_id_input;
commit;
DBMS_OUTPUT.PUT_LINE('Marked reservation as canceled');
exception
when no_data_found then
dbms_output.put_line('Could not find reservation');
end;

set serveroutput on;
exec cancel_reservations(4); --cancelling reservation_id # 4
```

Script Output x

Task completed in 0.21 seconds

Procedure CANCEL\_RESERVATIONS compiled

Marked reservation as canceled

PL/SQL procedure successfully completed.



(April Raimond, Member 2)

10) Show Cancellations: Print all canceled reservations in the hotel management system. Show reservation ID, hotel name, location, guest name, room type, dates.

Set Serveroutput On;

Select Reservation\_ID, Hotel\_Name, Guest\_Name, Room\_Type, Res\_Start\_Date,

Res\_End\_Date

from Reservation

Where is\_canceled = 1;

The screenshot shows a 'Query Builder' window with a SQL query and a 'Script Output' window showing the results of the query.

**Query:**

```
--April Raimond
/** Show Cancellations: Print all canceled reservations in the hotel management system. Show reservation ID, hotel name,
location, guest name, room type, dates **/

Set Serveroutput On;
Select Reservation_ID, Hotel_Name, Guest_Name, Room_Type, Res_Start_Date, Res_End_Date
from Reservation
Where is_canceled = 1;
```

**Script Output:** Task completed in 0.296 seconds

| RESERVATION_ID | HOTEL_NAME | GUEST_NAME   | ROOM_TYPE  | RES_START | RES_END_D |
|----------------|------------|--------------|------------|-----------|-----------|
| 3              | HGI1       | William Wing | single     | 12-APR-18 | 12-JUN-18 |
| 4              | HGI4       | Calvin Chao  | conference | 12-JUN-18 | 17-AUG-18 |

(Ray Rasolofonera, MEMBER 3)

11) Change a reservation Date: Input the reservation ID and change reservation start and end date, if there is availability in the same room type for the new date interval

--Create the Name of the Procedure

create or replace procedure change\_res\_date(res\_Id in number, res\_s\_date in date, res\_e\_date in date) as

room\_type\_temp varchar2(30);

hotel\_id\_temp number;

is\_ava number:=1;

begin

select room\_type into room\_type\_temp from RESERVATION where res\_Id = reservation\_id;

select hotel\_id into hotel\_id\_temp from RESERVATION where res\_Id = reservation\_id;

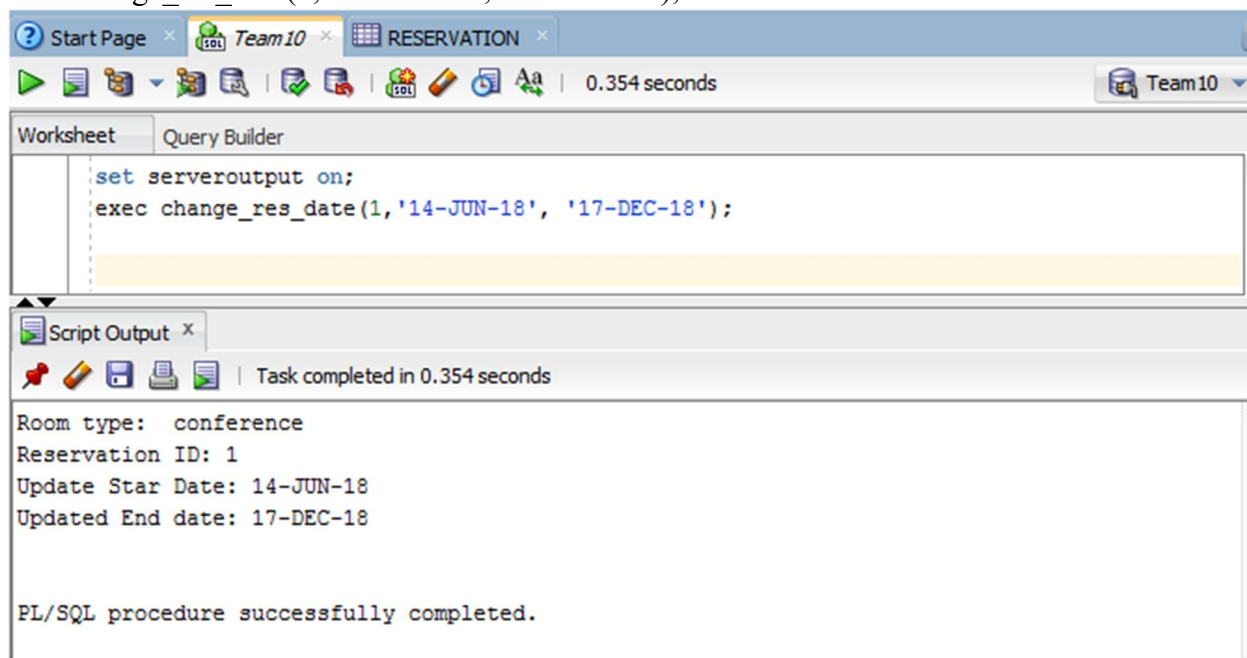
dbms\_output.put\_line('Room type: ' || room\_type\_temp);

```

for o in (select * from room where hotel_id_temp = hotel_id and room_type_temp = room_type)
LOOP -- Loop through all the rooms
if(o.ROOM_AVAILABLE = 0) then --Check if Room status is 0 or Available
is_ava :=0;
end if;
end loop;
if(is_ava = 0) then --If the room is available then allow Date Change
update reservation
set res_start_date = res_s_date, res_end_date = res_e_date
where reservation_ID = res_Id;
commit;
dbms_output.put_line('Reservation ID: '|| res_Id);
dbms_output.put_line('Update Star Date: '||res_s_date);
dbms_output.put_line('Updated End date: ' || res_e_date);
else -- If Room is not available then
dbms_output.put_line('No rooms available with the same type');
end if;
EXCEPTION -- If the Room type is the same then say that No room available
when NO_DATA_FOUND then
dbms_output.put_line('No rooms available with the same type');
end;

set serveroutput on;
exec change_res_date(1, '14-JUN-18', '17-DEC-18');

```



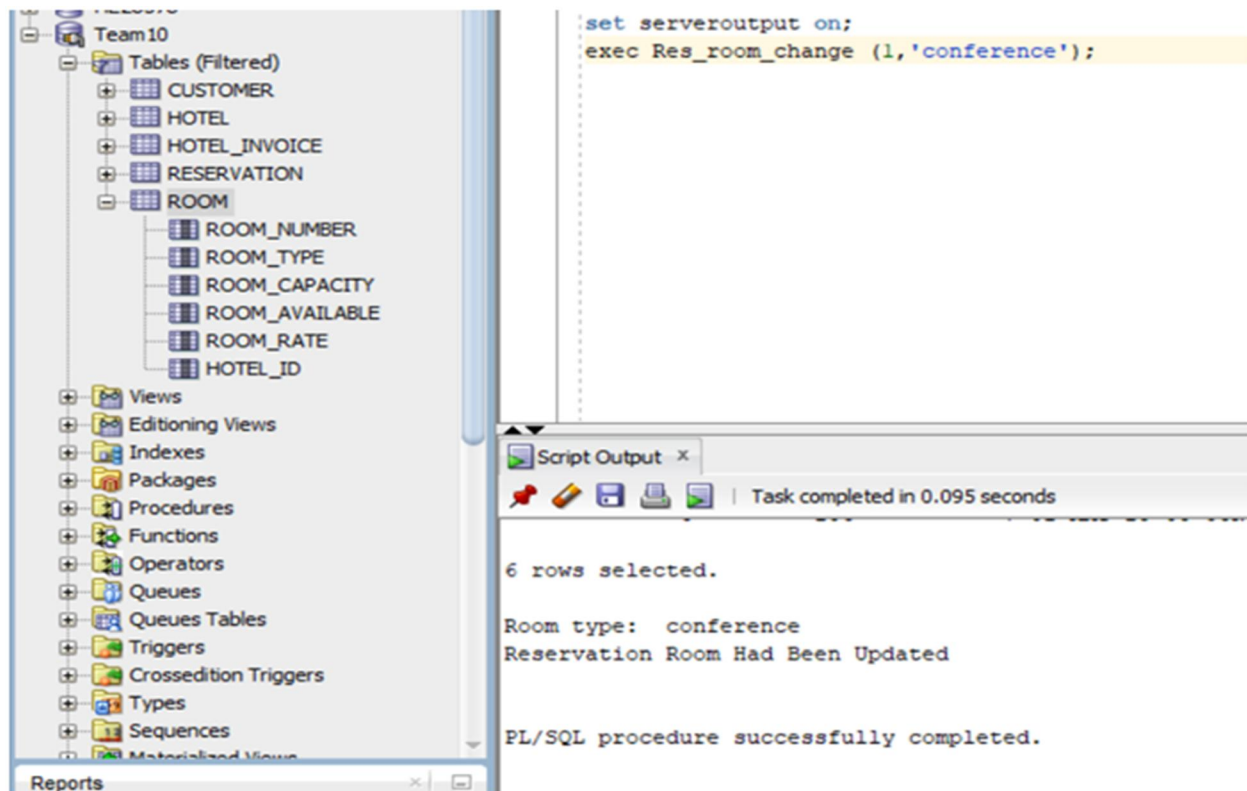
(Ray Rasolofonera, MEMBER 3)

12). Change a reservation RoomType: Input the reservation ID and change reservation room type if there is availability for that room type during the reservation's date interval

-- Create the Name of the Procedure

```
create or replace procedure Res_room_change(reservation_ID_input
reservation.reservation_ID%type, room_type_input in varchar) as
room_type_temp varchar2(30);
hotel_id_temp number;
is_ava number:=1;
begin
select room_type into room_type_temp from RESERVATION where reservation_ID_input =
reservation_id;
select hotel_id into hotel_id_temp from RESERVATION where reservation_ID_input =
reservation_id;
dbms_output.put_line('Room type: ' || room_type_temp);
for o in (select * from room where hotel_id_temp = hotel_id and room_type_temp = room_type)
LOOP-- Loop through all the rooms
if(o.ROOM_AVAILABLE = 0) then
is_ava :=0;
end if;
end loop;
if(is_ava = 0) then --If the room is available then allow Room Type Change
update reservation
set room_type = room_type_input
where reservation_id = reservation_ID_input;
commit;
dbms_output.put_line('Reservation Room Had Been Updated');
end if;
Exception-- If the Reservation ID is not found therefore room type does not exist
when no_data_found then
dbms_output.put_line('No reservation Found');
end;
```

```
set serveroutput on;
exec Res_room_change (1,'conference');
```



(Ray Rasolofonera, MEMBER 3)

13) Show single hotel reservations: Given a hotel ID, show all reservations for that hotel

--Create the Name of the Function Procedure

create or replace procedure show\_single\_Hot\_Res(Hotel\_input\_ID HOTEL.HOTEL\_ID%type)  
as

cursor Hotel\_Res\_Cursor is

select \* from reservation

where Hotel\_ID = Hotel\_input\_ID;

hotelRow Hotel\_Res\_Cursor%rowtype;

begin

open Hotel\_Res\_Cursor;

loop -- Loop through all the Hotel\_Res\_Cursor

fetch Hotel\_Res\_Cursor into hotelRow;

exit when Hotel\_Res\_Cursor%notfound;

dbms\_output.put\_line('Hotel ID: ' || hotelRow.Hotel\_id);

dbms\_output.put\_line('Guest Name: ' || hotelRow.Guest\_name);

dbms\_output.put\_line('Reservation ID: ' || hotelRow.Reservation\_id);

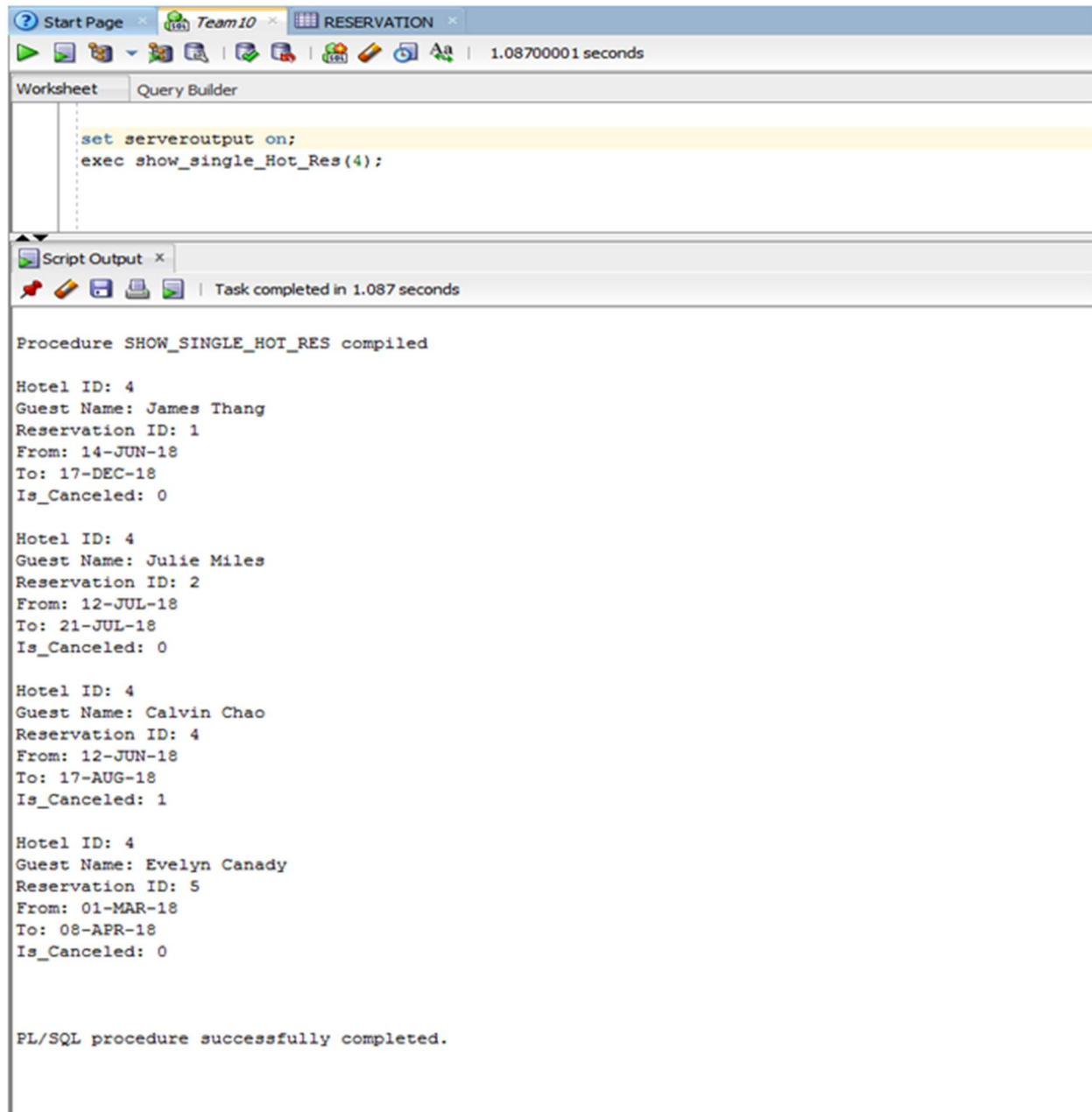
dbms\_output.put\_line('From: ' || hotelRow.res\_start\_date);

dbms\_output.put\_line('To: ' || hotelRow.res\_end\_date);

dbms\_output.put\_line('Is\_Canceled: ' || hotelRow.is\_canceled);

```
dbms_output.put_line(' ');
end loop;
close Hotel_Res_Cursor;

exception -- If the Hotel ID is not Found
when NO_DATA_FOUND then
dbms_output.put_line('Wrong Hotel ID');
End;
set serveroutput on;
exec show_single_Hot_Res(4);
```



The screenshot shows the Oracle SQL Developer interface. The top toolbar includes icons for Start Page, Team10, and a tab for 'RESERVATION'. The 'Worksheet' tab is active, displaying a PL/SQL script. The 'Script Output' tab is also open, showing the execution results. The script executed is:

```
set serveroutput on;
exec show_single_Hot_Res(4);
```

The output indicates that the procedure 'SHOW\_SINGLE\_HOT\_RES' was compiled successfully. It then displays the details of four reservations for Hotel ID 4:

```
Hotel ID: 4
Guest Name: James Thang
Reservation ID: 1
From: 14-JUN-18
To: 17-DEC-18
Is_Canceled: 0

Hotel ID: 4
Guest Name: Julie Miles
Reservation ID: 2
From: 12-JUL-18
To: 21-JUL-18
Is_Canceled: 0

Hotel ID: 4
Guest Name: Calvin Chao
Reservation ID: 4
From: 12-JUN-18
To: 17-AUG-18
Is_Canceled: 1

Hotel ID: 4
Guest Name: Evelyn Canady
Reservation ID: 5
From: 01-MAR-18
To: 08-APR-18
Is_Canceled: 0
```

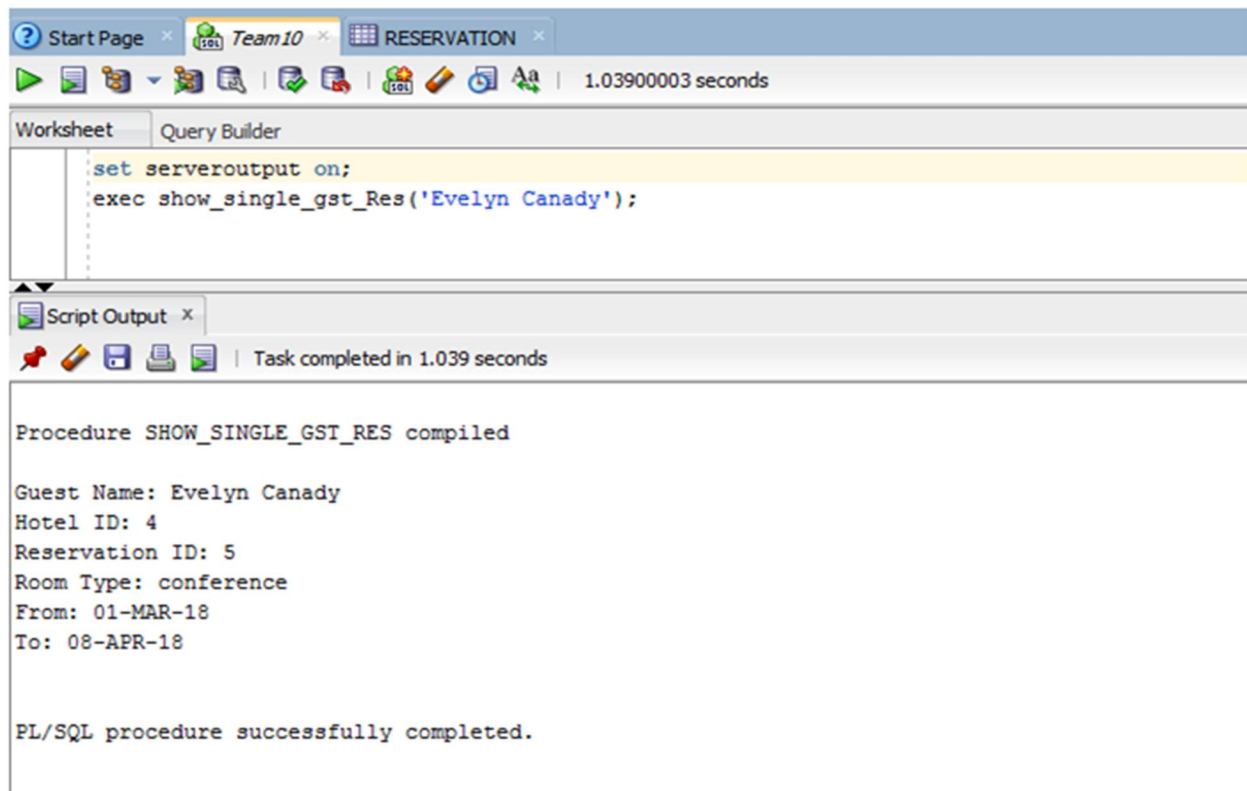
The final line of the output states: 'PL/SQL procedure successfully completed.'

(Ray Rasolofonera, MEMBER 3)

14) Show single guest reservations: Given a guest name, find all reservations under that name

```
--Create the Name of the Function Procedure
create or replace procedure show_single_gst_Res(guest_name_input
customer.customer_name%type) as
cursor Hotel_Res_Cursor is
select * from reservation
where guest_name= guest_name_input;
hotelRow Hotel_Res_Cursor%rowtype;
begin
open Hotel_Res_Cursor;
Loop --Loop through all the Hotel_Res_Cursor
fetch Hotel_Res_Cursor into hotelRow;
exit when Hotel_Res_Cursor%notfound;
dbms_output.put_line('Guest Name: ' || hotelRow.Guest_name);
dbms_output.put_line('Hotel ID: ' || hotelRow.Hotel_id);
dbms_output.put_line('Reservation ID: ' || hotelRow.Reservation_id);
dbms_output.put_line('Room Type: ' || hotelRow.Room_type);
dbms_output.put_line('From: ' || hotelRow.res_start_date);
dbms_output.put_line('To: ' || hotelRow.res_end_date);
end loop;
close Hotel_Res_Cursor;
exception --When the guest name is not Found
when NO_DATA_FOUND then
dbms_output.put_line('Wrong Guest Name');
end;

set serveroutput on;
exec show_single_gst_Res('Evelyn Canady');
```



```
set serveroutput on;
exec show_single_gst_Res('Evelyn Canady');
```

Script Output x

Task completed in 1.039 seconds

Procedure SHOW\_SINGLE\_GST\_RES compiled

Guest Name: Evelyn Canady  
Hotel ID: 4  
Reservation ID: 5  
Room Type: conference  
From: 01-MAR-18  
To: 08-APR-18

PL/SQL procedure successfully completed.

(Robert, Member 1)

3)Robert - Sell existing hotel: Sell a hotel by providing its hotel ID. Mark it as sold, do not delete the record.

--this fuction sets a Hotel as sold

--Args: Hotel\_id

create or replace procedure sell\_hotel(Hotel\_id\_input in number) as

begin

update Hotel

set is\_active = 1 where hotel\_id = Hotel\_id\_input;

dbms\_output.put\_line(Hotel\_id\_input || ' has been sold.');

commit;

Exception

--will print cannot find hotel if no hotel with that ID exists

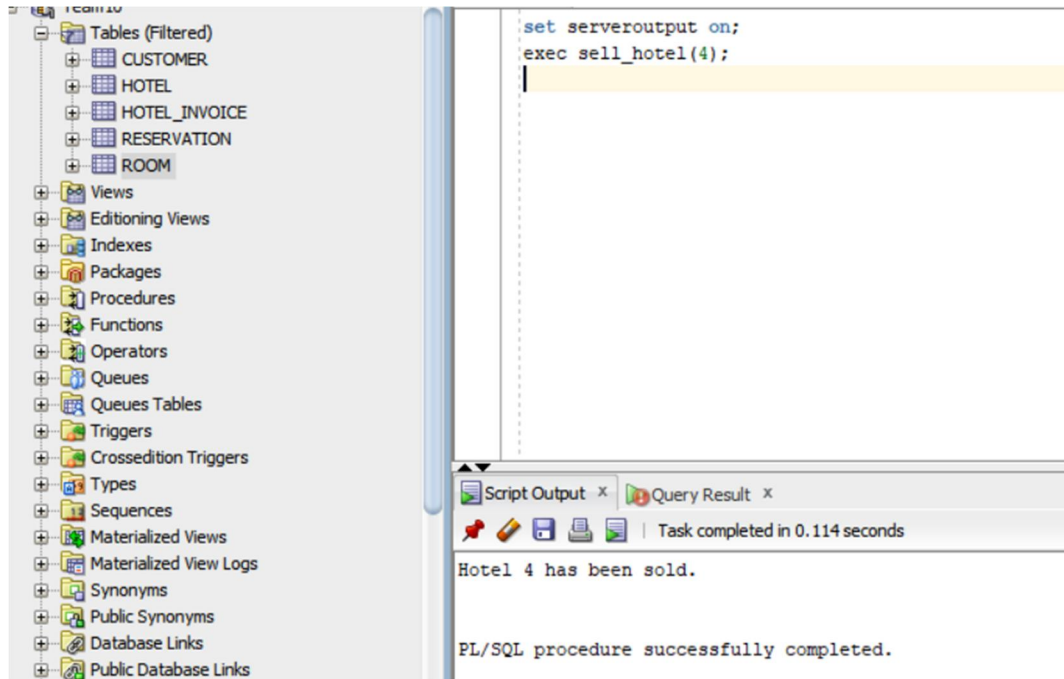
when no\_data\_found then

dbms\_output.put\_line('Cannot find that Hotel');

End;

```
set serveroutput on;
```

```
exec sell_hotel(4);
```



(Robert, Member 1)

4) Robert - Display hotel info: Given a hotel ID, display all information about that hotel

--shows all information about hotel based on the hotel ID

--arg: Hotel ID

create or replace procedure show\_hotel\_details(Hotel\_id\_input in number) is

cursor hotel\_cursor is

select \*

from HOTEL

where hotel\_id = hotel\_id\_input;

hotelRow hotel\_cursor%rowtype;

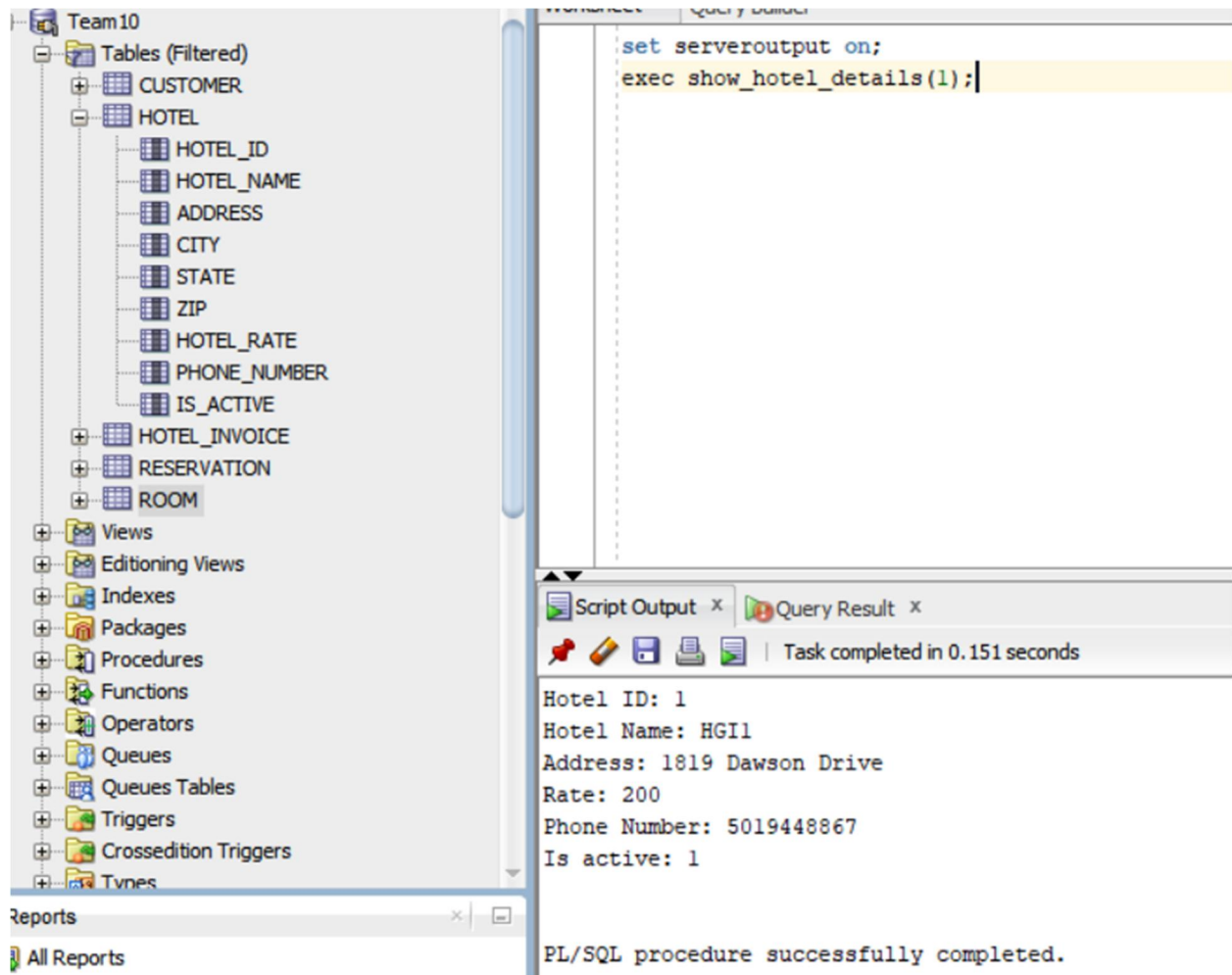
begin

open hotel\_cursor;

loop



```
fetch
hotel_cursor into hotelRow;
exit when hotel_cursor%notfound;
dbms_output.put_line('ID: ' || hotelRow.hotel_id);
dbms_output.put_line('Name: ' || hotelRow.hotel_name);
dbms_output.put_line('Address: ' || hotelRow.address);
dbms_output.put_line('Rate: ' || hotelRow.hotel_rate);
dbms_output.put_line('Number: ' || hotelRow.phone_number);
dbms_output.put_line('Is active: ' || hotelRow.is_active);
end loop;
close hotel_cursor;
end;
set serveroutput on;
exec show_hotel_details(1);
```



(Robert, Member 5)

20) Robert - show\_reservations , display all rooms reserved in all hotels.

```

create or replace procedure show_reservation is
cursor reservation_cursor is
select room_number
from reservation;
reservationRow reservation_cursor%rowtype;
begin
dbms_output.put_line('All reserved rooms: ');
open reservation_cursor;
loop
fetch
reservation_cursor into reservationRow;
exit when reservation_cursor%notfound;
dbms_output.put_line('Room: ' || reservationRow.room_number);

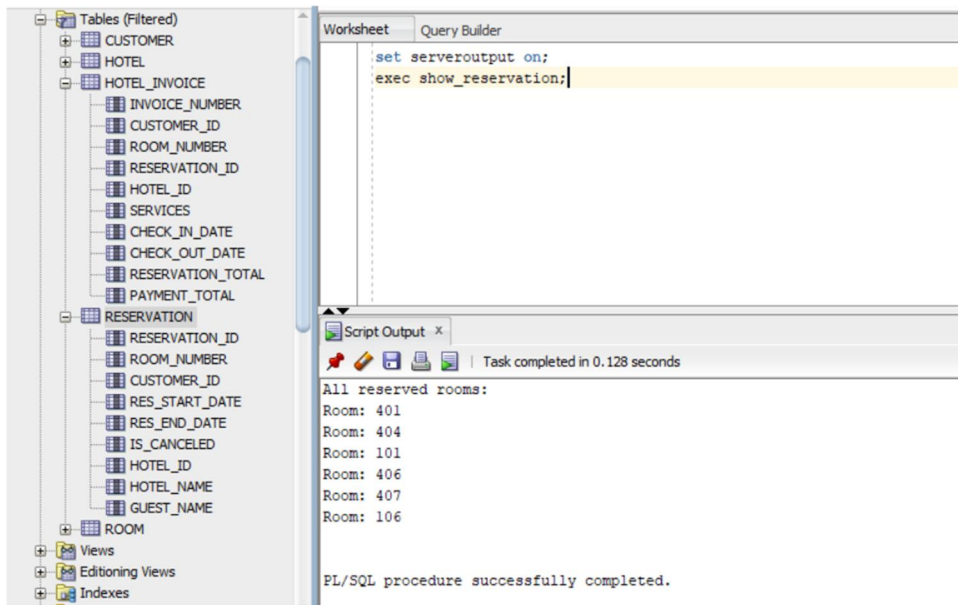
```

```

end loop;
close reservation_cursor;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        dbms_output.put_line('No rows found in reservation');
end;

set serveroutput on;
exec show_reservation;

```



(Robert, Member 5)

21) Robert - Show available rooms by type: Given a hotel ID, display the count of all available rooms by room type

```

create or replace procedure ava_rooms_by_type(Hotel_id_input in number) is
    cursor room_cursor is
        select room_type
        from room
        where hotel_id = hotel_id_input
        and ROOM_AVAILABLE = 0;
    roomRow room_cursor%rowtype;
    suite_count number:=0;
    conference_count number:=0;
    single_count number:=0;

```

```

begin
open room_cursor;
loop
fetch
room_cursor into roomRow;
exit when room_cursor%notfound;
if (roomRow.room_type = 'suite') then
suite_count:= suite_count+1;
elsif (roomRow.room_type = 'single') then
single_count :=single_count+1;
elsif (roomRow.room_type = 'conference') then
conference_count :=conference_count+1;
end if;
end loop;
dbms_output.put_line('Suite Room Type Count Available: ' || suite_count);
dbms_output.put_line('Single Room Type Count Available: ' || single_count);
dbms_output.put_line('Conference Room Type Count Available: ' || conference_count);
close room_cursor;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
dbms_output.put_line('No rooms found in given hotel id');
end;

set serveroutput on;
exec ava_rooms_by_type(1);

```

The screenshot shows the Oracle SQL Developer interface. On the left, the 'Tables (Filtered)' pane displays a hierarchical view of the database schema, including tables like CUSTOMER, HOTEL, HOTEL\_INVOICE, and RESERVATION. The central 'Query Builder' pane contains the following SQL code:

```
set serveroutput on;
exec ava_rooms_by_type(1);
```

On the right, the 'Script Output' pane shows the results of the execution:

```
Suite Room Type Count Available: 2
Single Room Type Count Available: 1
Conference Room Type Count Available: 1
```

Below the output, a message states: 'PL/SQL procedure successfully completed.'

0 means room is available

| ROOM_NUMBER | ROOM_TYPE  | ROOM_CAPACITY | ROOM_AVAILABLE | ROOM_RATE | HOTEL_ID |
|-------------|------------|---------------|----------------|-----------|----------|
| 101         | single     | 4             | 0              | 100       | 1        |
| 102         | single     | 4             | 1              | 100       | 1        |
| 105         | suite      | 10            | 0              | 400       | 1        |
| 106         | suite      | 10            | 0              | 400       | 1        |
| 107         | conference | 90            | 0              | 700       | 1        |
| 108         | conference | 90            | 1              | 700       | 1        |

6 rows selected.

(Robert, Member 5)

22)Robert - Room Checkout Report: Input: ReservationID

```
create or replace procedure checkout_report(reservation_id_input in number) is
cursor reservation_cursor is
select *
from HOTEL_INVOICE
where reservation_id_input = RESERVATION_ID;
hotel_invoice_row reservation_cursor%rowtype;
customer_name_var varchar(50);
customer_id_save number;
room_num_save number;
room_rate_save number;
begin
dbms_output.put_line('Check out report: ');
select customer_id into customer_id_save from HOTEL_INVOICE where reservation_id_input
= RESERVATION_ID;
select customer_name into customer_name_var from customer where customer_id_save =
customer_id;
select room_number into room_num_save from HOTEL_INVOICE where reservation_id_input
= RESERVATION_ID;
select room_rate into room_rate_save from room where room_number = room_num_save;

dbms_output.put_line('Name: ' || customer_name_var);
open reservation_cursor;
loop
fetch
reservation_cursor into hotel_invoice_row;
exit when reservation_cursor%notfound;
dbms_output.put_line('Room Number: ' || hotel_invoice_row.room_number);
dbms_output.put_line('Rate: ' || room_rate_save);
dbms_output.put_line('Services: ' || hotel_invoice_row.services);
dbms_output.put_line('Amount to be paid: ' || hotel_invoice_row.payment_total);
end loop;
close reservation_cursor;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
dbms_output.put_line('Could not find reservation ID in hotel invoice');
end;
```

```
set SERVEROUTPUT ON;  
exec checkout_report(1);
```

The screenshot displays the Oracle SQL Developer environment. On the left, the 'Connections' pane shows a connection to 'HE28578' under the 'Team10' user. The 'Tables (Filtered)' list includes CUSTOMER, HOTEL, and HOTEL\_INVOICE. The HOTEL\_INVOICE table is expanded, showing columns: INVOICE\_NUMBER, CUSTOMER\_ID, ROOM\_NUMBER, RESERVATION\_ID, HOTEL\_ID, SERVICES, CHECK\_IN\_DATE, CHECK\_OUT\_DATE, RESERVATION\_TOTAL, and PAYMENT\_TOTAL. The 'Script Output' window at the bottom right shows the execution of the 'checkout\_report' procedure, which completed successfully in 0.118 seconds. The output text is as follows:

```
Check out report:  
Name: James Thang  
Room Number: 401  
Rate: 100  
Services: R  
Amount to be paid: 120  
  
PL/SQL procedure successfully completed.
```

(Robert, Member 5)

23) Robert - Total income by state. Use a given state find income generated by it.

```
create or replace procedure income_by_state(state_input in varchar2) is
cursor reservation_cursor is
select *
from HOTEL_INVOICE;
hotel_id_save number;
total_sum number:=0;
state_temp varchar2(30);
hotel_invoice_row reservation_cursor%rowtype;
begin
dbms_output.put_line('Income for state of ' || state_input);
open reservation_cursor;
loop
fetch
reservation_cursor into hotel_invoice_row;
exit when reservation_cursor%notfound;
select state into state_temp from hotel where hotel_invoice_row.hotel_id = hotel_id;
if (state_temp = state_input) then
total_sum:= total_sum+hotel_invoice_row.payment_total;
end if;
end loop;
dbms_output.put_line('Total: ' || total_sum);
close reservation_cursor;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
dbms_output.put_line('Could not find any income for given state');
end;

set SERVEROUTPUT ON;
exec income_by_state('MD');
```



HE28578  
Team10

Tables (Filtered)

- CUSTOMER
- HOTEL
  - HOTEL\_ID
  - HOTEL\_NAME
  - ADDRESS
  - CITY
  - STATE
  - ZIP
  - HOTEL\_RATE
  - PHONE\_NUMBER
  - IS\_ACTIVE
- HOTEL\_INVOICE
  - INVOICE\_NUMBER
  - CUSTOMER\_ID
  - ROOM\_NUMBER
  - RESERVATION\_ID
  - HOTEL\_ID
  - SERVICES
  - CHECK\_IN\_DATE
  - CHECK\_OUT\_DATE
  - RESERVATION\_TOTAL
  - PAYMENT\_TOTAL
- RESERVATION

worksheet Query Builder

```
set SERVEROUTPUT ON;  
exec income_by_state('CA');
```

Script Output x

Task completed in 0.103 seconds

Income for state of CA  
Total: 1980

PL/SQL procedure successfully completed.

(James Thang, Member 1)

5) JAMES THANG - SoldHotels: Print all sold hotel information. Show hotel ID, location, etc.

--Comment –

This procedure does not take in any parameters. It creates a cursor that points to hotel table and it prints out all the columns in the row if the column isActive is set to 0. This means the hotel has been sold.

Create or replace procedure soldHotel

is

Cursor hotel\_cursor is

Select hotel\_id, hotel\_name, address, city, state, zip, hotel\_rate, phone\_number from hotel where  
is\_active = 1;

hotelRow hotel\_cursor%rowtype;

Begin

Open hotel\_cursor;

Loop

Fetch

Hotel\_cursor into hotelRow;

Exit when hotel\_cursor%notfound;

dbms\_output.put\_line('Id: ' || hotelRow.hotel\_id);

dbms\_output.put\_line('Name: ' || hotelRow.hotel\_name);

dbms\_output.put\_line('Address: ' || hotelRow.address);

dbms\_output.put\_line('City: ' || hotelRow.city);

dbms\_output.put\_line('State: ' || hotelRow.state);

dbms\_output.put\_line('Zip: ' || hotelRow.zip);

dbms\_output.put\_line('Rate: ' || hotelRow.hotel\_rate);

dbms\_output.put\_line('Number: ' || hotelRow.phone\_number);

end loop;

end;

set serveroutput on;

exec soldhotel;

Team10

- Tables (Filtered)
  - CUSTOMER
  - HOTEL
    - HOTEL\_ID
    - HOTEL\_NAME
    - ADDRESS
    - CITY
    - STATE
    - ZIP
    - HOTEL\_RATE
    - PHONE\_NUMBER
    - IS\_ACTIVE
  - HOTEL\_INVOICE
  - RESERVATION
  - ROOM
- Views
- Editing Views
- Indexes
- Packages
- Procedures
- Functions
- Operators
- Queues
- Queues Tables
- Triggers
- Crossedition Triggers
- Tunes

Query Builder

```
set serveroutput on;  
exec soldhotel;
```

Script Output x Query Result x

Task completed in 0.16 seconds

hotel id: 5  
hotel name: HGI5  
hotel address: 189 Ryan Road  
hotel city: Hayes  
hotel state: SD  
hotel zip: 57537  
hotel hotel rate: 200  
hotel phone number: 6055673502  
hotel id: 8  
hotel name: HGI8  
hotel address: 1000 Square Plaza

(James Thang, Member 1)

6) JAMES THANG - ReportHotelsInState: Given a state, display information of all hotels in that particular state.

---Comment---

This procedure takes in a parameter “state” such as MD, and prints out all hotels and information about it based on it’s state.

create or replace procedure show\_hotel\_by\_state(Hotel\_state\_input Hotel.state%type) is

cursor hotel\_cursor\_state is

select \*

from HOTEL

where state = hotel\_state\_input;

hotelRow hotel\_cursor\_state%rowtype;

begin

open hotel\_cursor\_state;

loop

fetch

hotel\_cursor\_state into hotelRow;

exit when hotel\_cursor\_state%notfound;

dbms\_output.put\_line('ID: ' || hotelRow.hotel\_id);

dbms\_output.put\_line('Name: ' || hotelRow.hotel\_name);

dbms\_output.put\_line('Address: ' || hotelRow.address);

dbms\_output.put\_line('Rate: ' || hotelRow.hotel\_rate);

dbms\_output.put\_line('Number: ' || hotelRow.phone\_number);

dbms\_output.put\_line('Active: ' || hotelRow.is\_active);

end loop;

close hotel\_cursor\_state;

end;

set serveroutput on;

exec show\_hotel\_by\_state('MD');

The screenshot displays the Oracle SQL Developer environment. On the left, the 'Team10' schema is expanded, showing a 'Tables (Filtered)' folder with the following tables: CUSTOMER, HOTEL, HOTEL\_INVOICE, RESERVATION, and ROOM. The 'HOTEL' table is selected, revealing its columns: HOTEL\_ID, HOTEL\_NAME, ADDRESS, CITY, STATE, ZIP, HOTEL\_RATE, PHONE\_NUMBER, and IS\_ACTIVE. The main window shows a SQL query in the 'Query Builder' tab:

```
set serveroutput on;  
exec show_hotel_by_state('MD');
```

Below the query editor, the 'Script Output' and 'Query Result' tabs are visible. The 'Query Result' tab shows the output of the executed query:

Task completed in 0.104 seconds

|               |                    |
|---------------|--------------------|
| Hotel ID:     | 6                  |
| Hotel Name:   | HGI6               |
| Address:      | 1000 Hilltop Plaza |
| Rate:         | 500                |
| Phone Number: | 3018765544         |
| Is active:    | 0                  |

PL/SQL procedure successfully completed.

(James Thang, Member 4)

16) JAMES THANG - Add a service to a reservation: Input: ReservationID, specific service. Add the service to the reservation for a particular date. Multiple services are allowed on a reservation for the same date.

It takes in reservation ID and it adds a service to the reservation.

create or replace Procedure addService(res\_id in number, service  
customer.customer\_services\_type%type)

IS

p\_service\_type customer.customer\_services\_type%type;  
p\_customer\_id reservation.customer\_id%type;

p\_service\_total integer := 0;

rCustomer integer;

pCustomer integer;

lCustomer integer;

BEGIN

select customer\_id into p\_customer\_id  
from RESERVATION  
where RESERVATION\_ID=res\_id;

Update CUSTOMER --updates the customer\_service\_type column  
SET customer\_services\_type = service  
where CUSTOMER\_ID=p\_customer\_id;

select customer\_services\_type into p\_service\_type  
from CUSTOMER  
where CUSTOMER\_ID=p\_customer\_id;

rCustomer := INSTR(p\_service\_type, 'R');

pCustomer := INSTR(p\_service\_type, 'P');

lCustomer := INSTR(p\_service\_type, 'L');

if rCustomer !=0 THEN p\_service\_total := p\_service\_total + 20;  
end if;

```
if pCustomer !=0 THEN p_service_total := p_service_total + 5;  
end if;
```

```
if lCustomer !=0 THEN p_service_total := p_service_total + 10;  
end if;
```

```
Update CUSTOMER --updates the service total  
SET customer_service_total = p_service_total  
where CUSTOMER_ID=p_customer_id;
```

```
end;
```

CUSTOMER Start Page is420\_group\_proj ADDSERVICE 0.11 seconds

Worksheet Query Builder

```
exec addService(2,'R');
commit;
```

Script Output x

Task completed in 0.11 seconds

PL/SQL procedure successfully completed.

Commit complete.

PL/SQL procedure successfully completed.

Commit complete.

| CUSTOMER_PHONE_NUM | CREDIT_CARD      | CUSTOMER_SERVICES_DATE | CUSTOMER_SERVICES_TYPE | CUSTOMER_SERVICE_TOTAL |
|--------------------|------------------|------------------------|------------------------|------------------------|
| 1 15467735         | 4556684216666496 | 17-DEC-15              | R                      | 20                     |
| 2 23148055         | 7087441234568527 | 15-JAN-17              | L                      | 10                     |
| 3 177437856        | 4589321589564585 | 27-FEB-17              | LR                     | 25                     |
| 4 1067255812       | 8756852564584548 | 23-APR-18              | LRP                    | 35                     |
| 5 09926285         | 1325458957565786 | 10-JAN-18              | RP                     | 25                     |
| 6 12224444         | 4444555566678    | 20-APR-18              | L                      | (null)                 |



(James Thang, Member 4)

17. JAMES THANG - Reservation Services Report: Input the reservation ID and display all services on this reservation. Print “no services for this reservation” if none exists

-- Input the reservation ID and display all services on this reservation.

create or replace

Procedure serviceReport(res\_id in number)

IS

```
p_service_type customer.customer_services_type%type;  
p_service_date customer.customer_services_date%type;  
p_customer_id reservation.customer_id%type;
```

begin

```
select customer_id into p_customer_id  
from RESERVATION  
where RESERVATION_ID=res_id;
```

```
select customer_services_date, customer_services_type into  
p_service_date, p_service_type  
from CUSTOMER  
where CUSTOMER_ID=p_customer_id;
```

--Print the information to the screen.

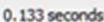









```
dbms_output.put_line('Reservation ID: ' || res_id);  
dbms_output.put_line('Services: ' || p_service_type);  
dbms_output.put_line('Service Type: ' || p_service_date);  
dbms_output.put_line('L=Laundry, R=Restaurant, P= Pay Per View');
```

exception --Throw an exception if no services for this reservation exists

when no\_data\_found then



```
dbms_output.put_line('No services for this reservation');
```

End;



Worksheet    Query Builder

```
set serveroutput on size 30000;
exec serviceReport(1);
```



Script Output    Task completed in 0.133 seconds

incorrect. The input data did not contain a number where a number was required by the format model.

\*Action: Fix the input data or the date format model to make sure the elements match in number and type. Then retry the operation.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Reservation ID: 1  
Services: LRP  
Service Type: 23-APR-18  
L=Laundry, R=Restaurant, P= Pay Per View

PL/SQL procedure successfully completed.

(James Thang, Member 4)

18) JAMES THANG - Show Specific Service Report: Input the service name, and display information on all reservations that have this service in all hotels.

Inputs the service name and it displays all reservation that has the service

create or replace Procedure spec\_resReport(service customer.customer\_services\_type%type )

IS

```
CURSOR custRow_cursor is
select customer_id,customer_services_type from customer;
```

```
customerRow custRow_cursor%rowtype;
```

```
CURSOR resRow_cursor is
select reservation_id, room_number, customer_id, res_start_date, res_end_date,
is_canceled from reservation;
```

```
reservationRow resRow_cursor%rowtype;
```

BEGIN

```
for customerRow in custRow_cursor
```

```
loop
```

```
if ( instr(customerRow.customer_services_type, service) > 0 ) then
```

```
for reservationRow in resRow_cursor
```

```
loop
```

```
if (customerRow.customer_id = reservationRow.customer_id) then
```

```
dbms_output.put_line('Res ID: ' || reservationRow.reservation_id || ' Room Number:' ||
```

```
reservationRow.room_number
```

```
|| ' Customer ID: ' || reservationRow.customer_id || ' Res Start: ' ||
```

```
reservationRow.Res_start_date||
```

```
' Res End:' || reservationRow.res_end_date
```

```
|| ' Res Canceled:' || reservationRow.is_canceled);
```

```
end if;
```

```
end loop;
```

```
end if;
```

```
end loop;
```

END;00

The screenshot displays the Oracle SQL Developer environment. The top toolbar includes icons for running queries, saving, and undo/redo, along with a timer showing 0.15899999 seconds. The main workspace is divided into a 'Worksheet' and a 'Query Builder' tab. The 'Worksheet' tab contains the SQL command: `exec spec_resreport('L');`. Below the workspace, the 'Script Output' window shows the execution results. It indicates that the task was completed in 0.159 seconds and that the PL/SQL procedure was successfully completed. The output also displays a table of reservation data.

|           |                 |                |                      |                   |                |
|-----------|-----------------|----------------|----------------------|-------------------|----------------|
| Res ID: 2 | Room Number:404 | Customer ID: 2 | Res Start: 12-JUL-18 | Res End:21-JUL-18 | Res Canceled:0 |
| Res ID: 3 | Room Number:101 | Customer ID: 3 | Res Start: 14-FEB-18 | Res End:25-FEB-18 | Res Canceled:1 |
| Res ID: 4 | Room Number:406 | Customer ID: 4 | Res Start: 12-JUN-18 | Res End:17-AUG-18 | Res Canceled:1 |
| Res ID: 6 | Room Number:106 | Customer ID: 7 | Res Start: 01-MAY-18 | Res End:08-JUN-18 | Res Canceled:0 |

PL/SQL procedure successfully completed.

(James Thang, Member 4)

19) JAMES THANG - Total Services Income Report: Given a hotelID, calculate and display income from all services in all reservations in that hotel.

Calculates the Total Services Income of the Hotel

create or replace Procedure getIncome(hotel\_id hotel\_invoice.hotel\_id%type )

IS

```
CURSOR invoice_cursor is
select hotel_id,services,reservation_total from hotel_invoice;
```

```
invoiceRow invoice_cursor%rowtype;
totalCost integer;
```

BEGIN

```
totalCost := 0;
```

```
for invoiceRow in invoice_cursor
```

```
loop
```

```
    if ( invoiceRow.hotel_id = hotel_id) then
```

```
totalCost := totalCost + invoiceRow.reservation_total;
```

```
end if;
```

```
end loop;
```

```
dbms_output.put_line('Hotel ID: ' || hotel_id || ' Income:' || totalCost);
```

END;

Worksheet

Query Builder

```
SET SERVEROUTPUT ON  
exec getIncome(4);  
exec getIncome(1);
```

Script Output x

Task completed in 0.149 seconds

PL/SQL procedure successfully completed.

Hotel ID: 4    Income:1980

PL/SQL procedure successfully completed.

Hotel ID: 1    Income:535

PL/SQL procedure successfully completed.