Database Management Project

Gupta, Ayush
NUID:
001817303

Title: Database Design for Online Hotel Booking Management Database

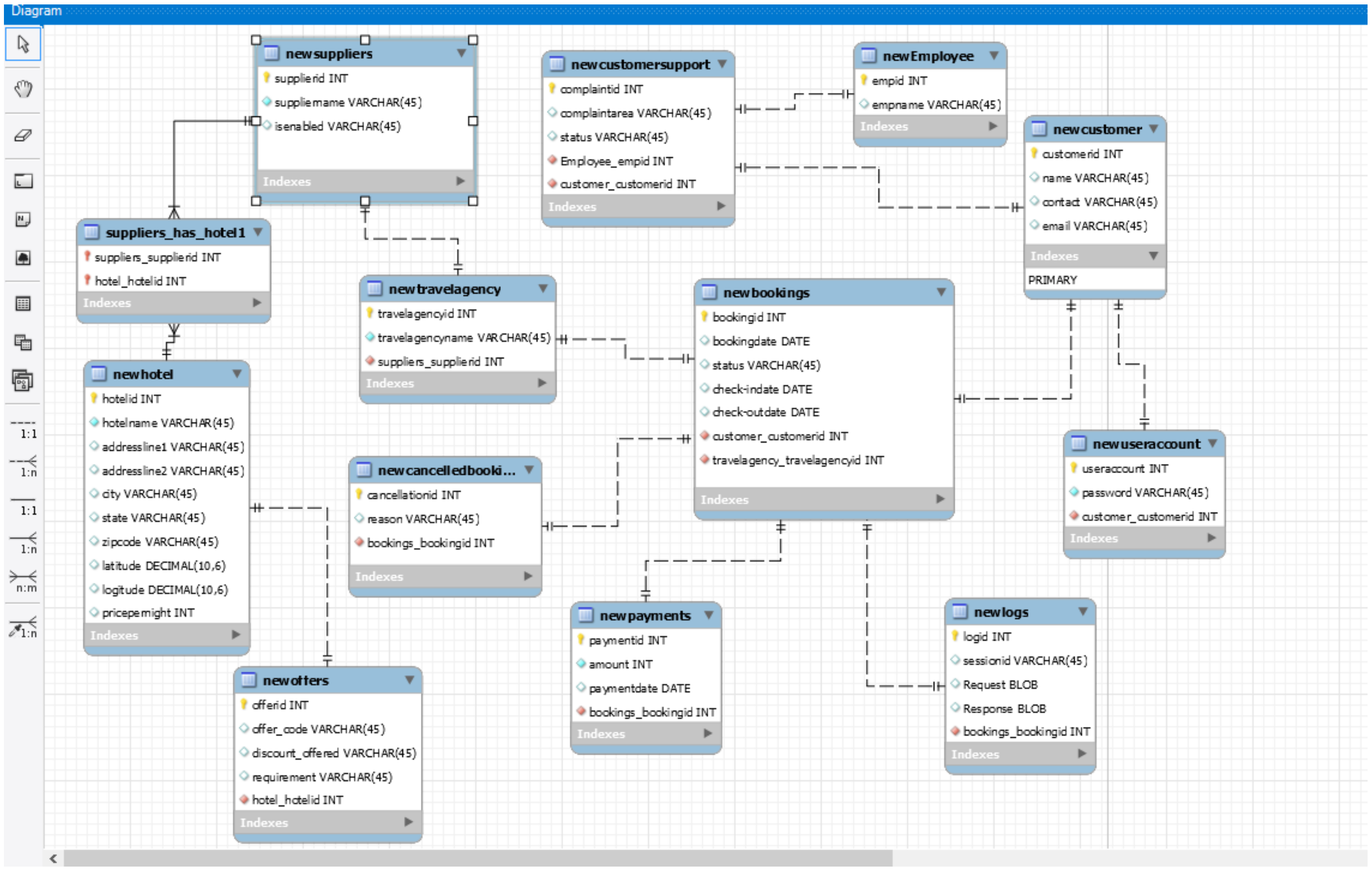
SUMMARY

There are thousands of travel bookings happening every day online. The Online travel booking market is expected to grow exponentially by 2020. The online travel industry should be competent enough to handle such huge amount of data along with maintaining their employee and customers.

SCOPE

The Database model I have designed is scalable and intelligent to meet the needs of today's dynamic market environment. Database uses have far exceeded the typical data storage needs and should be intelligent enough to manipulate the data run-time avoiding mismanagement and unnecessary revenue losses. I have designed to fit the needs of B2B model where multiple organizations come together for achieving success for each other.

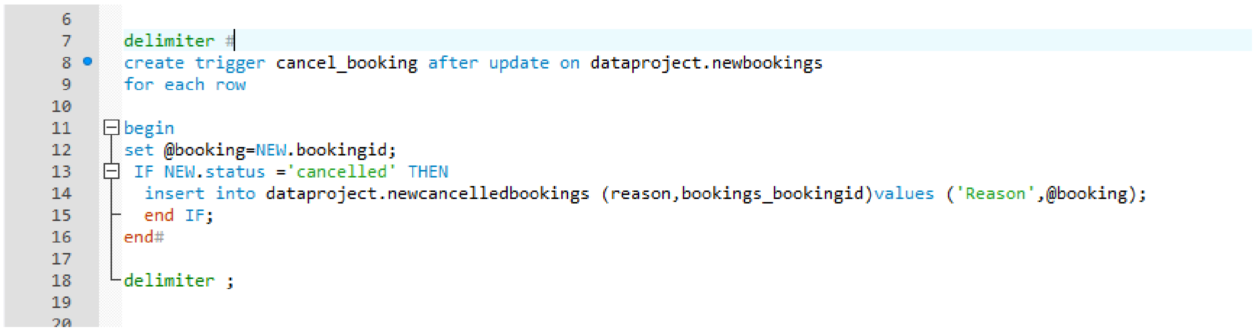
E-R DIAGRAM



Triggers/Stored Procedures/Views/Events:

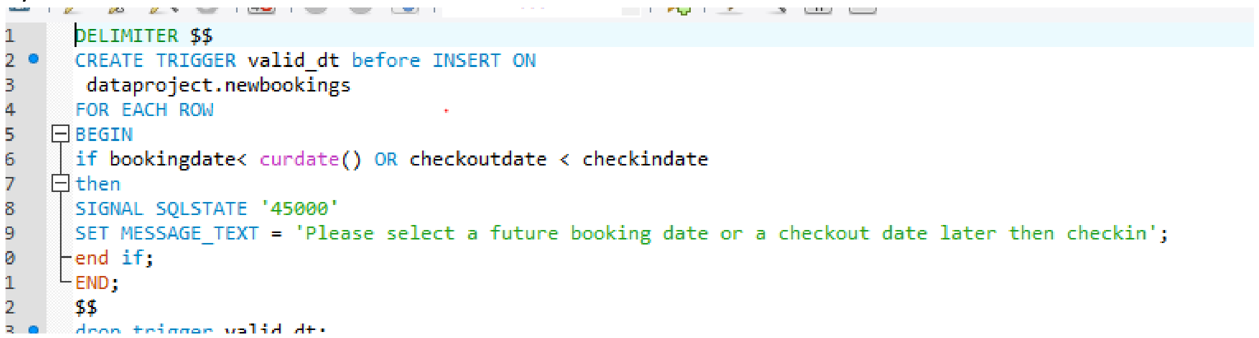
TRIGGERS:-

1) cancel_booking : Whenever a booking status gets updated to cancelled this trigger on bookings table adds that booking to the cancelled bookings table.
   Syntax:



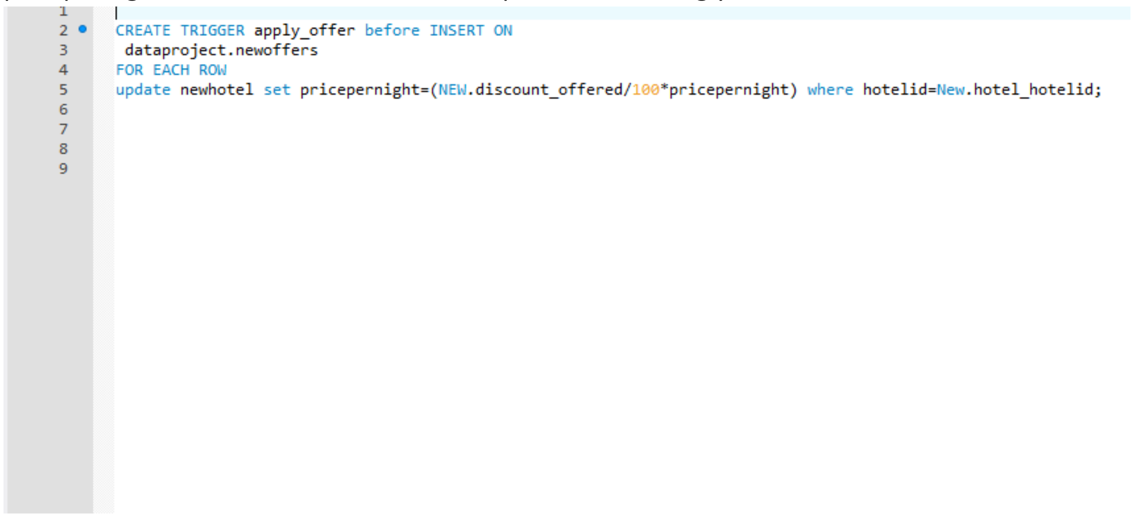
```
delimiter #
create trigger cancel_booking after update on dataproject.newbookings
for each row

begin
set @booking=NEW.bookingid;
 IF NEW.status ='cancelled' THEN
  insert into dataproject.newcancelledbookings (reason,bookings_bookingid)values ('Reason',@booking);
  end IF;
end#

delimiter ;
```

2) valid_dt: This trigger is used for validation of dates such that booking date needs to be greater than the current date and the check-in date cannot be later than the checkout date.
   Syntax:



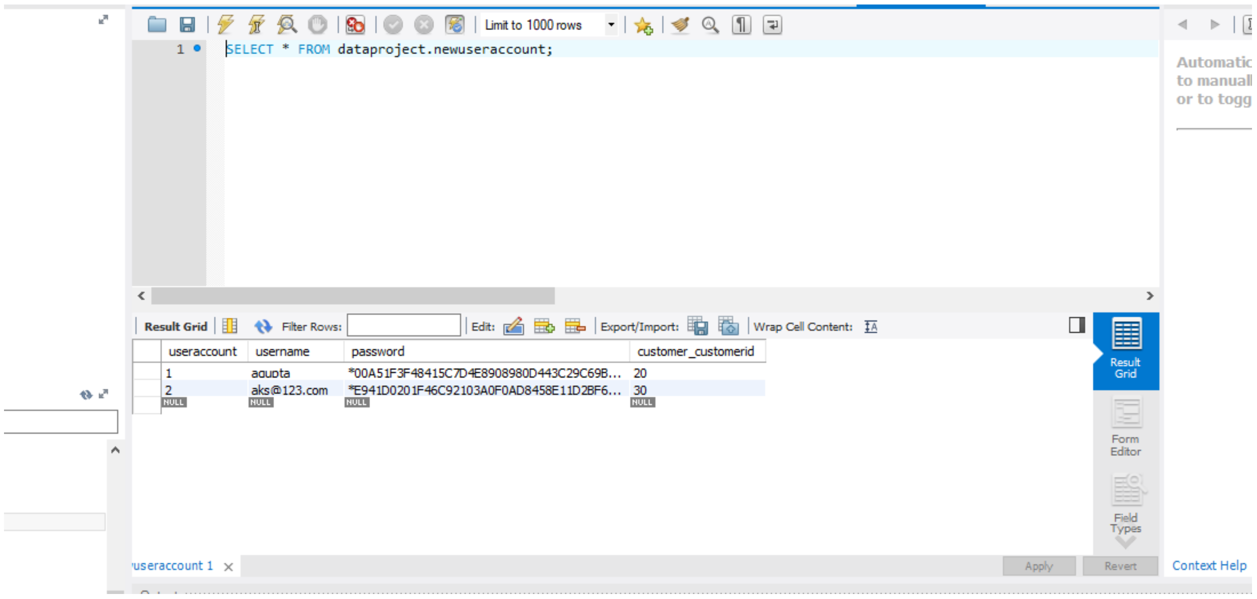
```
DELIMITER $$
CREATE TRIGGER valid_dt before INSERT ON
 dataproject.newbookings
FOR EACH ROW
BEGIN
if bookingdate< curdate() OR checkoutdate < checkindate
then
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = 'Please select a future booking date or a checkout date later then checkin';
end if;
END;
$$
drop trigger valid_dt;
```

3) apply_offer: Trigger is used to update new offers on the hotels. If a new offer is inserted the pricepernight of the hotel is revised and updated accordingly



```

CREATE TRIGGER apply_offer before INSERT ON
 dataproject.newoffers
FOR EACH ROW
update newhotel set pricepernight=(NEW.discount_offered/100*pricepernight) where hotelid=New.hotel_hotelid;
```

4) sec_pass: Trigger is used to store the user password as a hash and not the actual password increasing the security

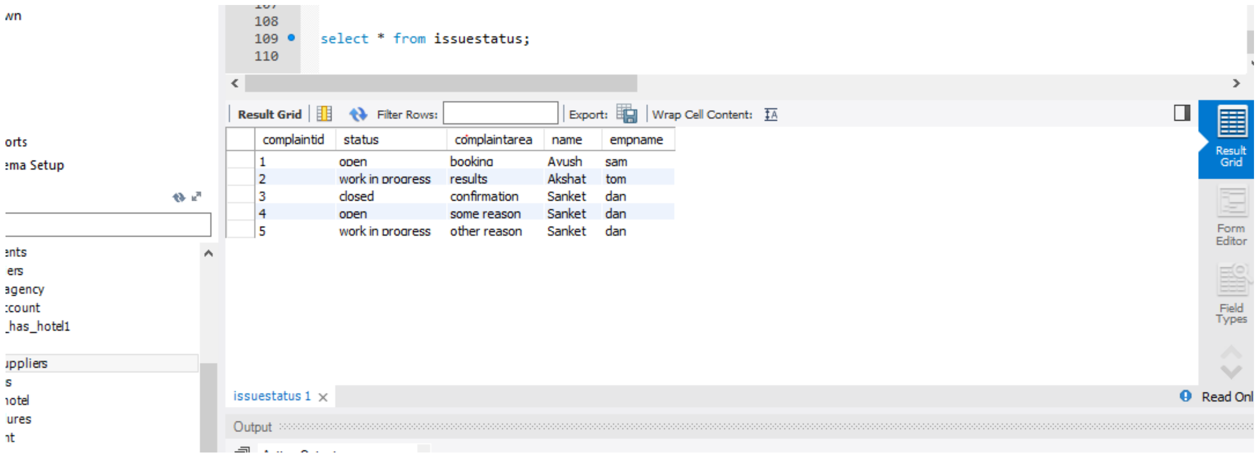


```
SELECT * FROM dataproject.newuseraccount;
```

useraccount	username	password	customer_customerid
1	agupta	*00A51F3F48415C7D4E8908980D443C29C69B...	20
2	aks@123.com	*E941D0201F46C92103A0F0AD8458E11D2BF6...	30
NULL	NULL	NULL	NULL

VIEWS:

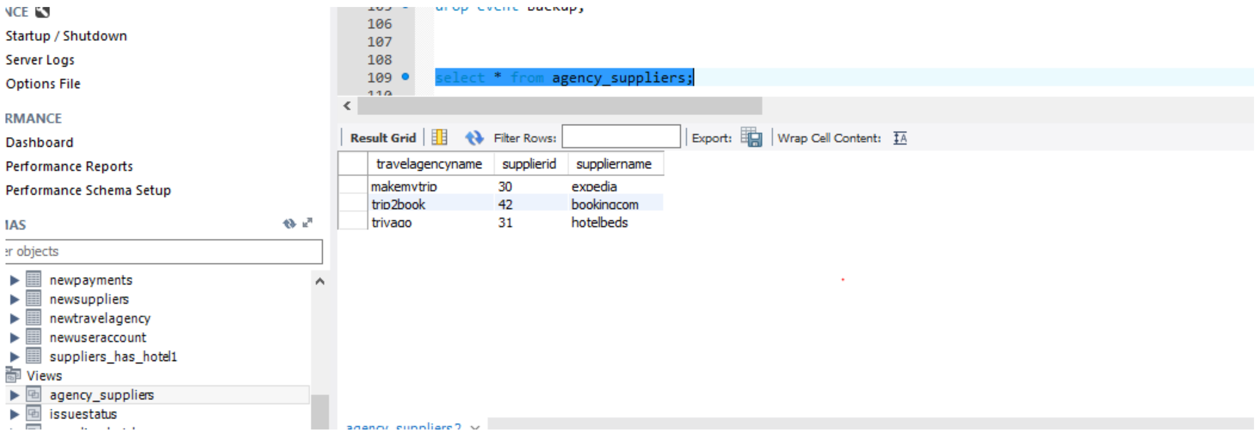
1) issuestatus: This view returns the insights for the issues reported by the customer which includes the complaint status, employee assigned to it and the customer who reported the issue.



```
select * from issuestatus;
```

complaintid	status	complaintarea	name	empname
1	open	booking	Ayush	sam
2	work in progress	results	Akshat	tom
3	closed	confirmation	Sanket	dan
4	open	some reason	Sanket	dan
5	work in progress	other reason	Sanket	dan

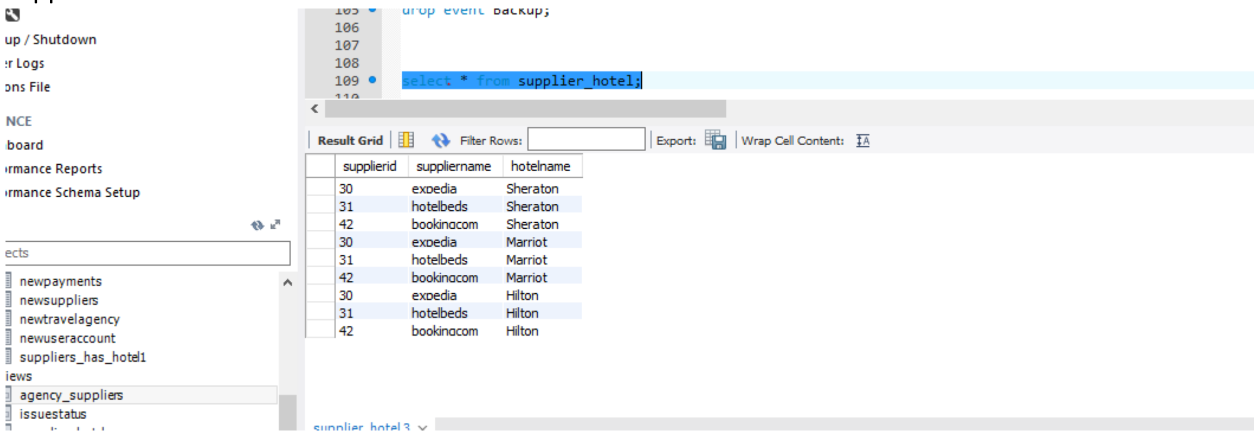
2) agency_suppliers: This view returns the travel agency along with the particular supplier they are using.



```
select * from agency_suppliers;
```

travelagencyname	supplierid	suppliername
makemytrip	30	expedia
trip2book	42	bookingcom
trivago	31	hotelbeds

3) This view returns the hotels available with each supplier such that a travel agency can see which supplier to connect to.

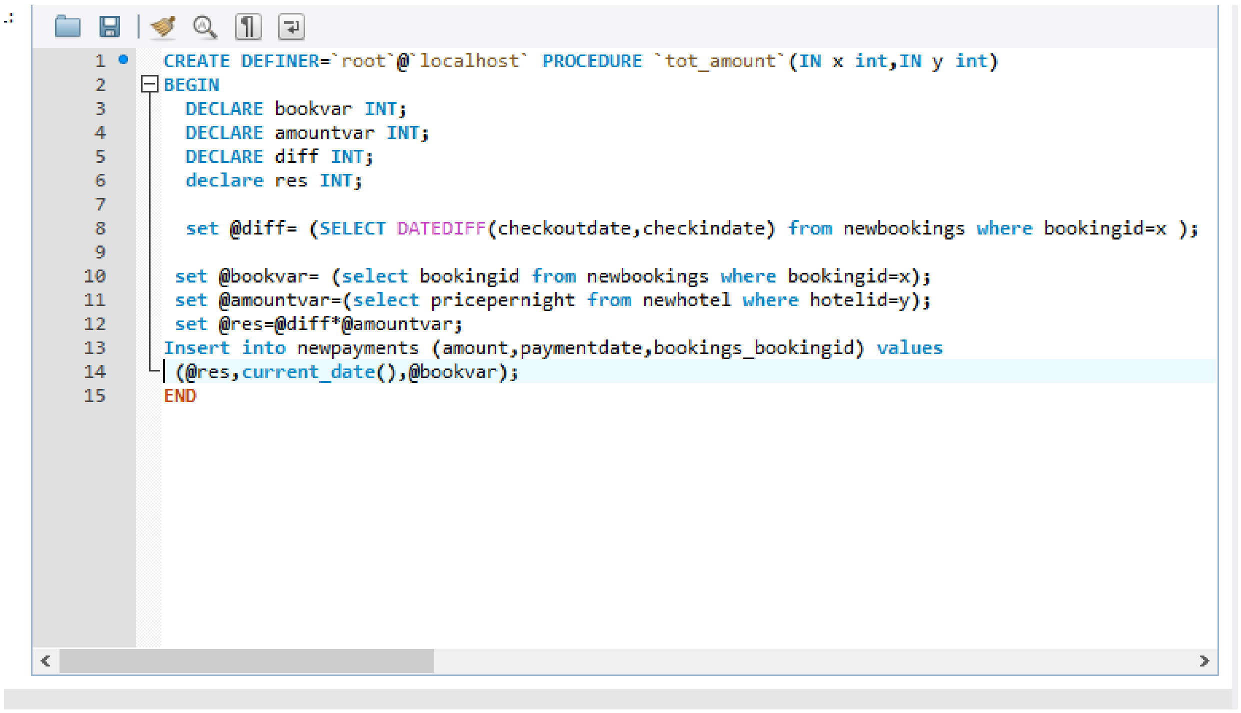


```
select * from supplier_hotel;
```

supplierid	suppliername	hotelname
30	expedia	Sheraton
31	hotelbeds	Sheraton
42	bookingcom	Sheraton
30	expedia	Marriot
31	hotelbeds	Marriot
42	bookingcom	Marriot
30	expedia	Hilton
31	hotelbeds	Hilton
42	bookingcom	Hilton

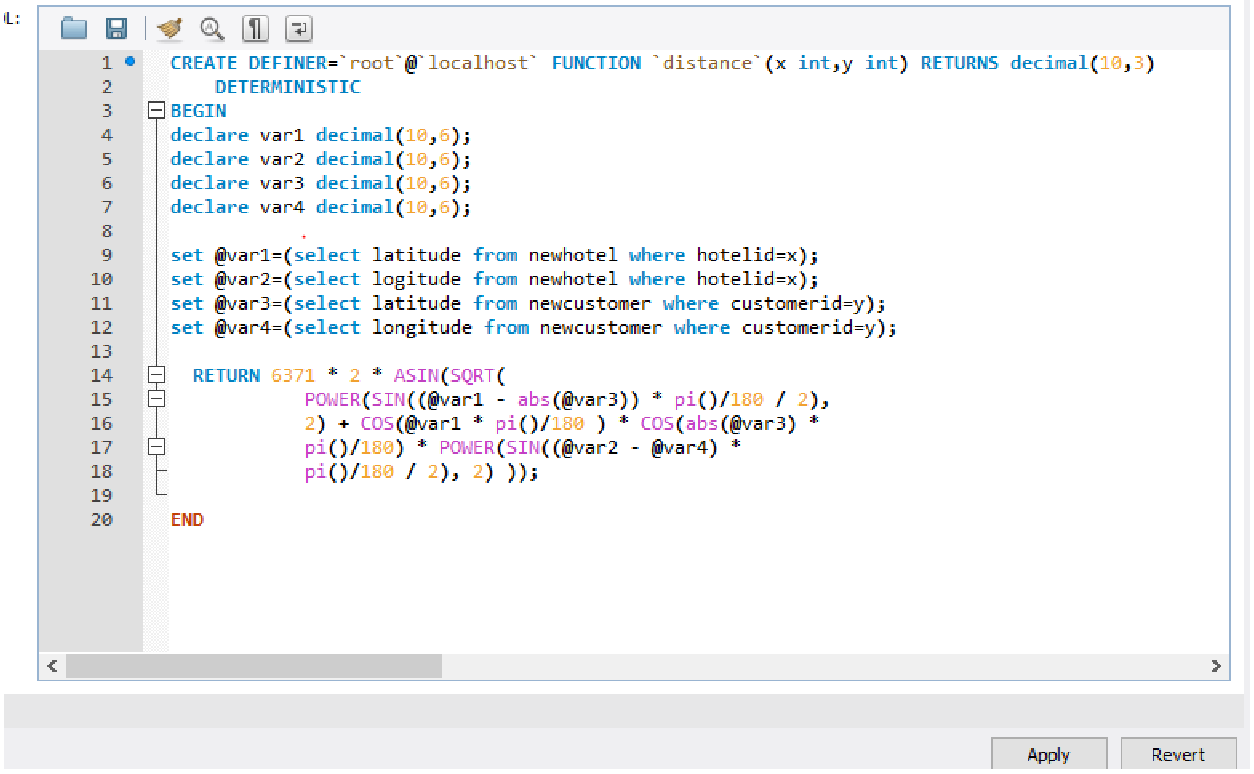
STORED PROCEDURE:

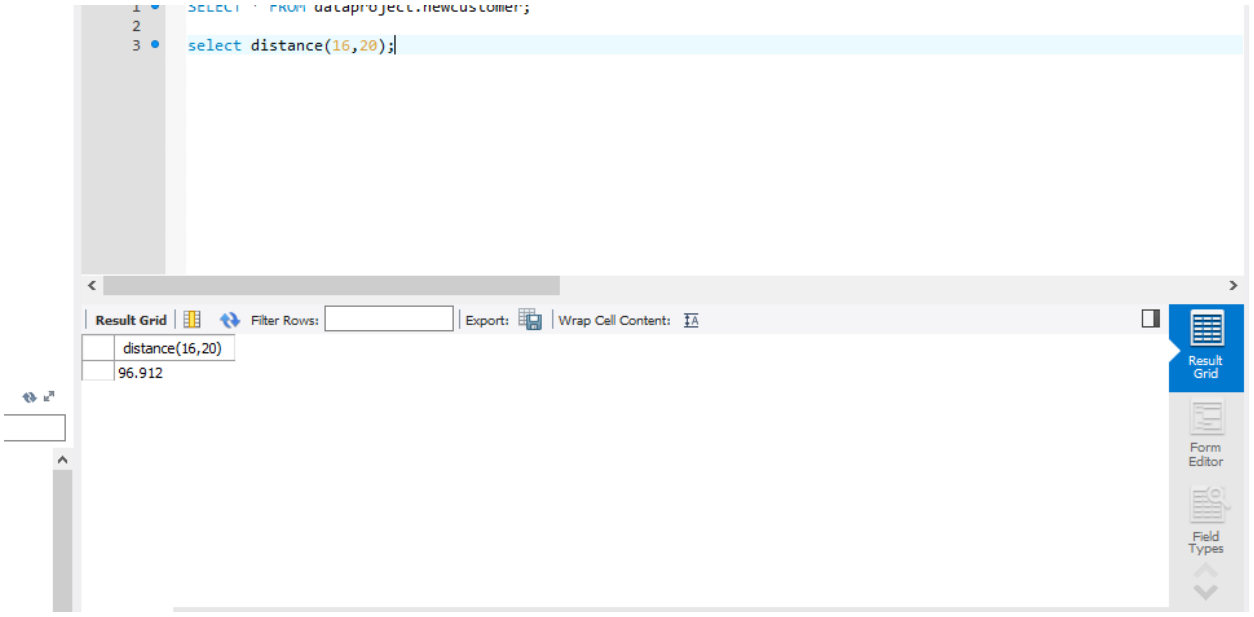
tot_amount – This procedure takes the input as booking id and hotelid. By providing the inputs it calculates the total booking amount based on the hotel pricepernight and the duration of stay and stores the amount in the payment table.


```
CREATE DEFINER=`root`@`localhost` PROCEDURE `tot_amount`(IN x int,IN y int)
BEGIN
  DECLARE bookvar INT;
  DECLARE amountvar INT;
  DECLARE diff INT;
  declare res INT;

  set @diff= (SELECT DATEDIFF(checkoutdate,checkindate) from newbookings where bookingid=x );

 set @bookvar= (select bookingid from newbookings where bookingid=x);
 set @amountvar=(select pricepernight from newhotel where hotelid=y);
 set @res=@diff*@amountvar;
Insert into newpayments (amount,paymentdate,bookings_bookingid) values
 (@res,current_date(),@bookvar);
END
```

USER DEFINED FUNCTION:

Distance(): This function takes the hotel and customers location from customer and hotel table. The location is taken in the form of latitude and longitude. By passing the hotel id and customer id it returns the distance between them in kilometers.


```
CREATE DEFINER=`root`@`localhost` FUNCTION `distance`(x int,y int) RETURNS decimal(10,3)
    DETERMINISTIC
BEGIN
declare var1 decimal(10,6);
declare var2 decimal(10,6);
declare var3 decimal(10,6);
declare var4 decimal(10,6);

set @var1=(select latitude from newhotel where hotelid=x);
set @var2=(select logitude from newhotel where hotelid=x);
set @var3=(select latitude from newcustomer where customerid=y);
set @var4=(select longitude from newcustomer where customerid=y);

 RETURN 6371 * 2 * ASIN(SQRT(
           POWER(SIN((@var1 - abs(@var3)) * pi()/180 / 2),
           2) + COS(@var1 * pi()/180 ) * COS(abs(@var3) *
           pi()/180) * POWER(SIN((@var2 - @var4) *
           pi()/180 / 2), 2) ));

END
```



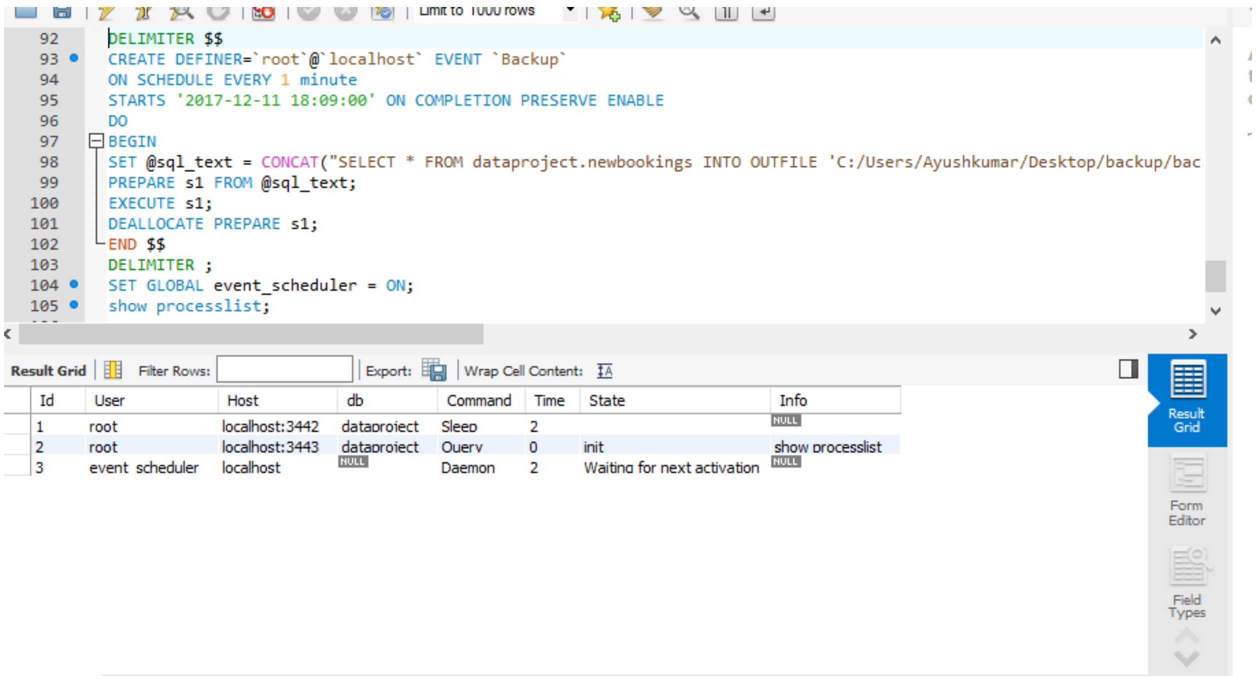
```
SELECT * FROM dataproject.newcustomer;

select distance(16,20);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

distance(16,20)
96.912

EVENTS:

Backup: The backup event is a job that can be configured to be run at schedule intervals and take incremental backups. It is better to automate the backup process rather than the manual work of taking the backup. The event writes the data to an out file of the system so that in case Database crash the sensitive data can be recovered also this can be used for sending bills, invoices to third parties or customers.



```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` EVENT `Backup`
ON SCHEDULE EVERY 1 minute
STARTS '2017-12-11 18:09:00' ON COMPLETION PRESERVE ENABLE
DO
BEGIN
SET @sql_text = CONCAT("SELECT * FROM dataproject.newbookings INTO OUTFILE 'C:/Users/Ayushkumar/Desktop/backup/bac
PREPARE s1 FROM @sql_text;
EXECUTE s1;
DEALLOCATE PREPARE s1;
END $$
DELIMITER ;
SET GLOBAL event_scheduler = ON;
show processlist;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Id	User	Host	db	Command	Time	State	Info
1	root	localhost:3442	dataproject	Sleep	2		NULL
2	root	localhost:3443	dataproject	Query	0	init	show processlist
3	event_scheduler	localhost	NULL	Daemon	2	Waiting for next activation	NULL

OTHER FUNCTIONS/FEATURES USED:

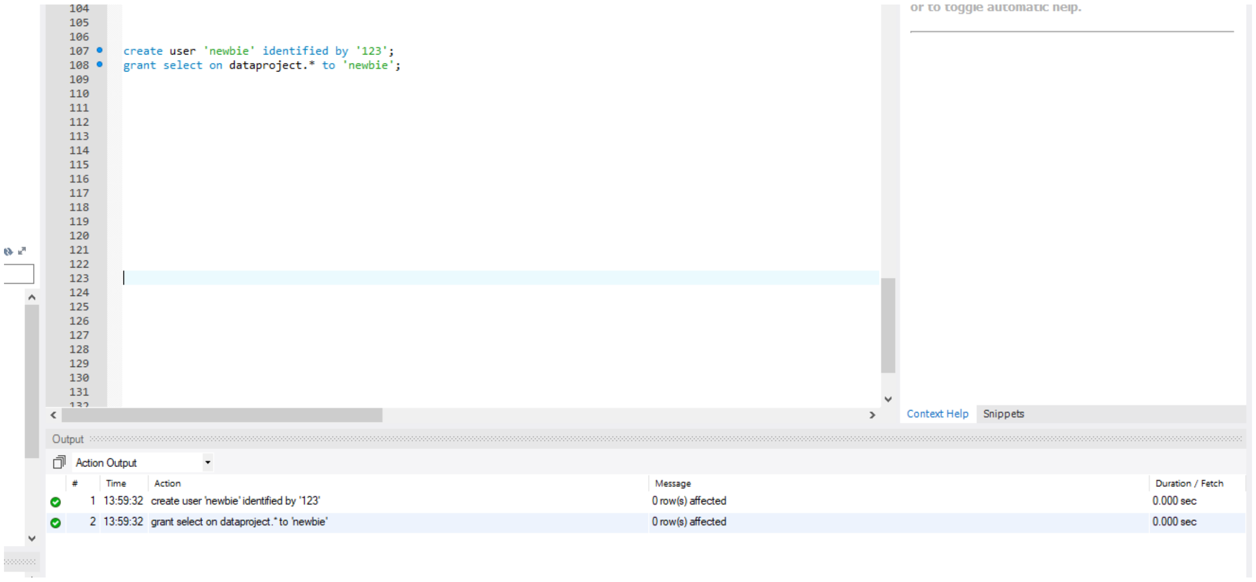
DATEDIFF() – Used this function to calculate the difference between checkin and checkout dates to calculate the amount in total_amount stored procedure.

PASSWORD()- Used this for hashing the passwords of the new customer sign ups.

BLOB- Used this data type to store hotel images and session xml logs.

ENUM- Used this data type to predefine the status of the complaints filed to (‘open’, ’work in progress’ , ’closed’)

PRIVILIGES:- created a new user newbie and granted only select functionality on the database.



```
create user 'newbie' identified by '123';
grant select on dataproject.* to 'newbie';
```

```
#  Time      Action                                        Message            Duration / Fetch
1  13:59:32  create user 'newbie' identified by '123'      0 row(s) affected  0.000 sec
2  13:59:32  grant select on dataproject.* to 'newbie'     0 row(s) affected  0.000 sec
```