TOPIC :HOTEL MANAGEMENT

GROUP MEMBERS	ROLL NO
MAYUR PATIL	1934203
VIVEK MORE	1934220
ANSHUL NATHE	1934255
JISHA LADE	1934242

INTRODUCTION TO HOTEL MANGEMENT

The **Hotel Management System**, referred to as **HMS**, is an application that will help users better utilize rooms used by UP employees and other guests. HMS helps users manage guest flows by affording them the ability to easily check UP guests in, check them out, and generate stay reports, among other things. This help text will outline some of the most common processes you will perform in HMS. It examines the following HMS topic, processes, and features:

- **Elements of the HMS Dashboard**
- **Checking in Guests**
  -
  - Editing incoming guest information
  - Checking in Guests
  - Processing Overflows
  - Processing Walks
  - Undoing a Check In
- **Checking Guests Out**
  -
  - Editing information of checked in guests
  - Checking guests out
  - Undoing a Check Out
- **Working with Managed Stays**
  - Transferring an Employee

- Performing a Billing Review
- Viewing Billing Adjustments
- Creating a Billing Adjustments

SCOPE OF HOTEL MANAGEMENT

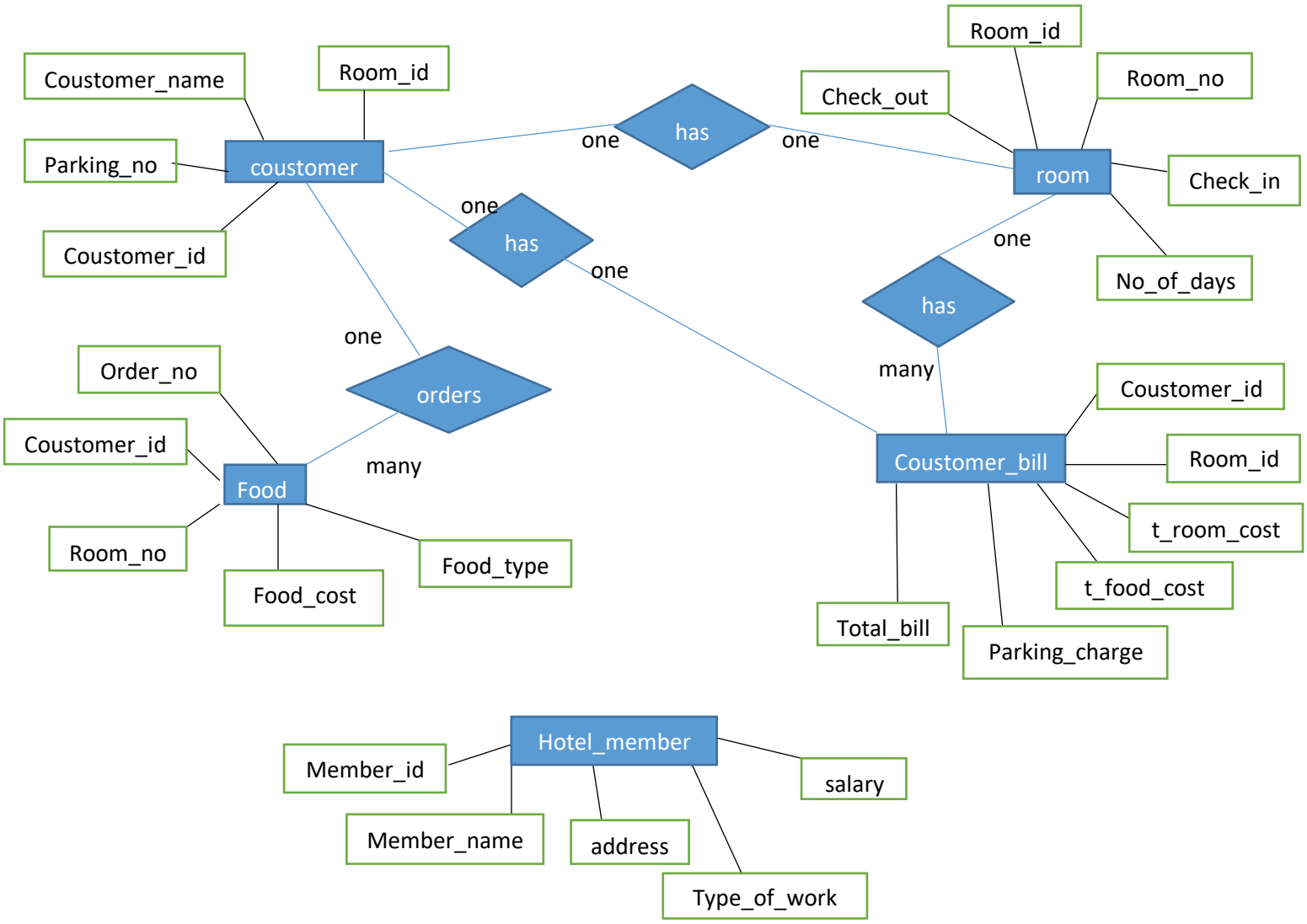
The Hotel management is one of the highly job oriented field; it covers a wide range of services including food service, accommodation and catering.

The major job fields in the hospitality sector include Hotels, resorts, fast food chains, restaurants, etc. A hotel management professional can be employed in any of the above mentioned fields. The hotel management field is always been a hot career options among the students, with the globalization more and more hotel industries as expanding their business to the global markets. This has created a huge demand of hotel management professionals.

NEED FOR COMPUTERIZATION

Need for Computerization in Hotels: Computerisation in hotels is a basic necessity. All the star category hotels are computerized. Now all the hotels are going for PMS.PMS is a software which connects all the departments which becomes easily accessible which helps in preparing bills is food and beverage outlet, maintain a standard recipe in food production department, Inventory in stores, to track KOT in restaurants, to avoid misunderstanding among employees. To avoid these problems in mind, Property management system is necessary in Hotels. Hotel – Property Management software: Property management software is integrated software which has applications to manage entire operational and nonoperational departments of a hotel. Although the components differ.

ER-DIGRAM



```
postgres=# \c  hotel_management;
You are now connected to database "hotel_management" as user "postgres".


postgres=# \d
          List of relations
 Schema |      Name      | Type  |  Owner
--------+----------------+-------+----------
 public | basic_info     | view  | postgres
 public | coustomer      | table | postgres
 public | coustomer_bill | table | postgres
 public | food           | table | postgres
 public | hotel_member   | table | postgres
```

```
 public | room          | table | postgres
public | staff         | view  | postgres
(7 rows)
```

#Description of tables

```
postgres=# \d coustomer;
                  Table "public.coustomer"
    Column     |        Type         | Collation | Nullable | Default
---------------+---------------------+-----------+----------+---------
 coustomer_id  | character varying(20) |           | not null |
 room_id       | character varying(10) |           |          |
 coustomer_name | character(30)        |           |          |
 parking_no    | character varying(10) |           |          |
Indexes:
   "coustomer_pkey" PRIMARY KEY, btree (coustomer_id)
Foreign-key constraints:
   "coustomer_room_id_fkey" FOREIGN KEY (room_id) REFERENCES room(room_id) ON UPDATE CASCADE ON DELETE
CASCADE
Referenced by:
   TABLE "coustomer_bill" CONSTRAINT "coustomer_bill_coustomer_id_fkey" FOREIGN KEY (coustomer_id) REFERENCES
coustomer
(coustomer_id) ON UPDATE CASCADE ON DELETE CASCADE
   TABLE "food" CONSTRAINT "food_coustomer_id_fkey" FOREIGN KEY (coustomer_id) REFERENCES
coustomer(coustomer_id) ON UP
DATE CASCADE ON DELETE CASCADE
```

```
postgres=# \d coustomer_bill;
                 Table "public.coustomer_bill"
    Column     |        Type         | Collation | Nullable | Default
---------------+---------------------+-----------+----------+---------
 coustomer_id   | character varying(20) |           |          |
 room_id       | character varying(10) |           |          |
 t_room_cost    | integer             |           |          |
 t_food_cost    | integer             |           |          |
 parking_charge | integer             |           |          |
 payment_type   | character(10)       |           |          |
 total_bill     | integer             |           |          |
Foreign-key constraints:
   "coustomer_bill_coustomer_id_fkey" FOREIGN KEY (coustomer_id) REFERENCES coustomer(coustomer_id) ON
UPDATE CASCADE O
N DELETE CASCADE
   "coustomer_bill_room_id_fkey" FOREIGN KEY (room_id) REFERENCES room(room_id) ON UPDATE CASCADE ON
DELETE CASCADE
```

```
postgres=# \d food;
                    Table "public.food"
   Column    |        Type         | Collation | Nullable | Default
-------------+---------------------+-----------+----------+---------
 order_no     | character varying(10) |           | not null |
 coustomer_id | character varying(20) |           |          |
 room_no      | integer               |           |          |
 food_cost    | integer               |           |          |
 food_type    | character varying     |           |          |
Indexes:
   "food_pkey" PRIMARY KEY, btree (order_no)
Foreign-key constraints:
   "food_coustomer_id_fkey" FOREIGN KEY (coustomer_id) REFERENCES coustomer(coustomer_id) ON UPDATE
CASCADE ON DELETE CASCADE


postgres=# \d room;
                    Table "public.room"
  Column   |        Type         | Collation | Nullable | Default
-----------+---------------------+-----------+----------+---------
 room_id    | character varying(10) |           | not null |
 room_no    | integer               |           | not null |
 no_of_days | integer               |           |          |
 check_in   | character(30)         |           | not null |
 check_out  | date                  |           |          |
Indexes:
   "room_pkey" PRIMARY KEY, btree (room_id)
Referenced by:
   TABLE "coustomer_bill" CONSTRAINT "coustomer_bill_room_id_fkey" FOREIGN KEY (room_id) REFERENCES
room(room_id) ON UP
DATE CASCADE ON DELETE CASCADE
   TABLE "coustomer" CONSTRAINT "coustomer_room_id_fkey" FOREIGN KEY (room_id) REFERENCES room(room_id)
ON UPDATE CASCADE ON DELETE CASCADE


postgres=# \d hotel_member;
                  Table "public.hotel_member"
   Column    |        Type         | Collation | Nullable | Default
-------------+---------------------+-----------+----------+---------
 member_id    | character varying(20) |           | not null |
 member_name  | character(30)         |           |          |
 address      | character varying(50) |           |          |
 type_of_work | character(20)         |           |          |
 salary       | double precision      |           |          |
```

Indexes:
"hotel_member_pkey" PRIMARY KEY, btree (member_id)

#insertion into tables

postgres=# select * from coustomer;

coustomer_id	room_id	coustomer_name	parking_no
B112261	A101	Vivek More	MH02DE2345
B112262	A102	Vishal More	MH02TH2628
B112264	A104	Great Khali	MH34hf4637
B112265	A105	Rey Mysterio	MH02HJ3829
B112266	A106	Charlotte Flair	MH02KJ3728
B112267	A107	Asuka	MH02SK8657
B112268	A108	Shasha Banks	MH02Dk8789
B112271	A111	Otis	HD98JH7678
B112272	A112	Heavy Machinery	HD98JH7667
B112274	A114	Big E	Dg02HG7656
B112277	A117	Undertaker	GH02GH4567
B112281	A121	Scallet Jonasson	SD24HJ6278
B112282	A122	Black Widow	SD24HJ6343
B112286	A126	Iron Man	DG35FG5656
B112289	A129	Joseph	Mh27728jj
B112288	A128	nick	nh01hg1678
B112287	A127	Fury	nh01hg1678
B112285	A125	Marvel	nh01hg1678
B112284	A124	Virat Kholi	nh01hg1678
B112283	A123	Steve Smith	nh01hg1678
B112280	A120	Captain America	nh01hg1678
B112278	A118	HHH	nh01hg1678
B112279	A119	Steve Austin	nh01hg1678
B112290	A130	Thaliava	nh01hg1678
B112263	A103	john cena	nh01hg1678
B112269	A109	Beky Lynch	nh01hg1678
B112270	A110	Shorty G	nh01hg1678
B112273	A113	The New Day	nh01hg1678
B112275	A115	Kofy Kingston	nh01hg1678
B112276	A116	The Street Profit	nh01hg1678

(30 rows)

postgres=# select* from food;

order_no	coustomer_id	room_no	food_cost	food_type

```
          +           +         +          +
 Z61      | B112261   |    101 |      100 | pasta
 Z62      | B112262   |    102 |      100 | French Fries
 Z63      | B112263   |    103 |      100 | Ice Cream
 Z64      | B112263   |    103 |      100 | Bread
 Z65      | B112263   |    103 |      100 | Fried Rice
 Z66      | B112263   |    103 |      100 | Pan Cake
 Z67      | B112264   |    103 |      100 | burger
 Z68      | B112265   |    105 |      100 | burger
 Z69      | B112266   |    106 |      100 | pizza
 Z70      | B112266   |    106 |      100 | pumpkin pie
 Z71      | B112267   |    107 |      100 | Apple pie
 Z72      | B112268   |    108 |      100 | Bagel
 Z73      | B112269   |    109 |      100 | Bagel
 Z74      | B112270   |    110 |      100 | muffins
 Z75      | B112278   |    106 |      100 | muffins
 Z76      | B112279   |    107 |      100 | Cheese cake
 Z77      | B112280   |    108 |      100 | Chetos
 Z78      | B112281   |    109 |      100 | Nachos
 Z79      | B112282   |    110 |      100 | Chimichango
 Z80      | B112283   |    101 |      100 | Chimichango
 Z81      | B112284   |    102 |      100 | Salsa
 Z82      | B112285   |    103 |      100 | Broccoli
 Z83      | B112286   |    104 |      100 | Chocolate covered Stawberries
 Z84      | B112287   |    105 |      100 | Steak
 Z85      | B112288   |    106 |      100 | Beaf
 Z86      | B112289   |    107 |      100 | Hot Dog
 Z87      | B112290   |    108 |      100 | biscuits and Gravy
(27 rows)


postgres=# select * from room;
 room_id | room_no | no_of_days |          check_in          | check_out
         +         +            +                            +
 A101    |    101 |          5 | 2020-1-5                   | 2020-01-10
 A102    |    102 |          3 | 2020-1-5                   | 2020-01-08
 A103    |    103 |          2 | 2020-1-5                   | 2020-01-07
 A104    |    104 |          6 | 2020-1-6                   | 2020-01-12
 A105    |    105 |          4 | 2020-1-7                   | 2020-01-11
 A106    |    106 |          1 | 2020-1-10                   | 2020-01-11
 A107    |    107 |          2 | 2020-1-15                   | 2020-01-17
 A108    |    108 |          3 | 2020-1-16                   | 2020-01-19
 A109    |    109 |          2 | 2020-1-17                   | 2020-01-19
 A110    |    110 |          1 | 2020-1-18                   | 2020-01-19
 A111    |    103 |          1 | 2020-1-10                   | 2020-01-11
 A112    |    103 |          5 | 2020-1-13                   | 2020-01-18
 A113    |    101 |          3 | 2020-1-21                   | 2020-01-24
 A114    |    102 |          6 | 2020-1-22                   | 2020-01-29
```

```
 A115   |    103 |        5 | 2020-1-24              | 2020-01-29
 A116   |    104 |        4 | 2020-1-27              | 2020-01-31
 A117   |    105 |        2 | 2020-1-27              | 2020-01-29
 A118   |    106 |        1 | 2020-1-29              | 2020-01-30
 A119   |    107 |        6 | 2020-1-30              | 2020-02-05
 A120   |    108 |        5 | 2020-1-30              | 2020-01-31
 A121   |    109 |        3 | 2020-2-31              | 2020-02-04
 A122   |    110 |        2 | 2020-2-2               | 2020-02-04
 A123   |    101 |        2 | 2020-2-4               | 2020-02-06
 A124   |    102 |        2 | 2020-2-5               | 2020-02-07
 A125   |    103 |        1 | 2020-2-6               | 2020-02-07
 A126   |    104 |        3 | 2020-2-9               | 2020-02-12
 A127   |    105 |        7 | 2020-2-10              |
 A128   |    106 |        5 | 2020-2-13              |
 A129   |    107 |        3 | 2020-2-13              |
 A130   |    108 |        4 | 2020-2-15              |
(30 rows)


postgres=# select * from coustomer_bill;
 coustomer_id | room_id | t_room_cost | t_food_cost | parking_charge | total_bill
--------------+---------+-------------+-------------+----------------+------------
 B112290      | A130    |        3200 |         150 |            600 |       3950
 B112264      | A104    |        4800 |         150 |            900 |       5850
 B112261      | A101    |        4000 |         150 |            750 |       4900
 B112265      | A105    |        3200 |         150 |            600 |       3950
 B112266      | A106    |         800 |         300 |            150 |       1250
 B112267      | A107    |        1600 |         150 |            300 |       2050
 B112268      | A108    |        2400 |         150 |            450 |       3000
 B112285      | A125    |         800 |         150 |            150 |       1100
 B112284      | A124    |        1600 |         150 |            300 |       2050
 B112269      | A109    |        1600 |         150 |            300 |       2050
 B112270      | A110    |         800 |         150 |            150 |       1100
 B112271      | A111    |         800 |           0 |            150 |        950
 B112272      | A112    |        4000 |           0 |            750 |       4750
 B112273      | A113    |        2400 |           0 |            450 |       2850
 B112274      | A114    |        4800 |           0 |            900 |       5700
 B112275      | A115    |        4000 |           0 |            750 |       4750
 B112276      | A116    |        3200 |           0 |            600 |       3800
 B112277      | A117    |        1600 |           0 |            300 |       1900
 B112278      | A118    |         800 |         150 |            150 |       1100
 B112279      | A119    |        4800 |         150 |            900 |       5850
 B112287      | A127    |        5600 |         150 |           1050 |       6800
 B112262      | A102    |        2400 |         150 |            450 |       3000
 B112280      | A120    |        4000 |         150 |            750 |       4900
 B112286      | A126    |        2400 |         150 |            450 |       3000
 B112288      | A128    |        4000 |         150 |            750 |       4900
 B112281      | A121    |        2400 |         150 |            450 |       3000
```

```
 B112289    | A129  |      2400 |       150 |         450 |      3000
 B112263    | A103  |      1600 |       600 |         300 |      2500
 B112282    | A122  |      1600 |       150 |         300 |      2050
 B112283    | A123  |      1600 |       150 |         300 |      2050
(30 rows)


postgres=# select * from hotel_member;
 member_id |        member_name         |          address          |     type_of_work     | salary
-----------+----------------------------+---------------------------+----------------------+--------
 M1231     | Anshul Nathe               | hyderabad 3421            | CEO                  | 100000
 M1232     | Jisha lade                 | maharastra manapa 41092   | Data Handler         | 80000
 M1233     | Sandeep Mundhe             | maharastra Alandi Road 411015 | waiter           | 50000
 M1234     | Mayur Patil                | dongri mumbai 23490       | waiter               |  50000
 M1235     | Mausom                     | Afganistan babu l2920     | Cleaning             | 50000
 M1236     | Vivek                      | Maharastra pune 411015    | Supervisor           | 50000
 M1237     | Namdev                     | kolhapur 93020            | cook                 | 60000
 M1238     | singh                      | punjab bale bale 78929    | cook                 | 60000
 M1239     | Arman                      | Delhi Sevhydyjbft 76878   | Laundry              | 60000
 M1240     | Malik                      | Delhi vghfh 6878          | dish washer          | 60000
 M1241     | taker                      | bhopal                    | washer               |  5000
 M1242     | edge                       | chicago                   | washer               | 5000
 M1243     | dwane                      | antartico                 | washer               | 5000
 M1244     | dwane                      | america                   | hdosjd               | 5000
 M1245     | char                       | america                   | hdosjd               | 5000
 M1246     | bellas                     | bpi america               | hunfjd               | 5000
 M1247     | bellas                     | bpi                       | hunfjd               | 4000
 M1248     | bellas                     | ranchi                    | slater               | 4000
 M1249     | gonja                      | hind                      | slater               | 4000
 M1250     | hhhq                       | asia                      | slater               | 4000
 M1251     | lotte                      | asia ameri ca             | slater               |  4000
 M1252     | megis                      | asia ameri fsjf00 ca      | watcher              |  4000
M1253      | ricky                      | fjjsj jfks ameri fsjf00 ca | watcher             | 4000
M1254      | cena                       | fsjf00 ca                 | watcher              | 4000
M1255      | vanndy cena                | fsjf00 ca                 | watcher              | 4000
M1256      | tiple hhh cena             | fsjf00 ca                 | watcher              | 4000
M1257      | twims                      | mp copjde joun            | watcher              | 4000
(27 rows)
```

---

creating views

View-1

```
postgres=# create view staff as select member_id,member_name ,salary from hotel_member;
CREATE VIEW
```

```
postgres=# \d staff
                View "public.staff"
  Column   |        Type         | Collation | Nullable | Default
-----------+---------------------+-----------+----------+---------
 member_id   | character varying(20) |           |          |
 member_name | character(30)         |           |          |
 salary      | double precision      |           |          |


postgres=# select * from staff;
 member_id |         member_name          | salary
-----------+------------------------------+--------
 M1231     | Anshul Nathe                 | 100000
 M1232     | Jisha lade                   | 80000
 M1233     | Sandeep Mundhe               | 50000
 M1234     | Mayur Patil                  |  50000
 M1235     | Mausom                       | 50000
 M1236     | Vivek                        | 50000
 M1237     | Namdev                       | 60000
 M1238     | singh                        |  60000
 M1239     | Arman                        |  60000
 M1240     | Malik                        |  60000
 M1241     | taker                        |   5000
 M1242     | edge                         |   5000
 M1243     | dwane                        |   5000
 M1244     | dwane                        |   5000
 M1245     | char                         |   5000
 M1246     | bellas                       |   5000
 M1247     | bellas                       |   4000
 M1248     | bellas                       |   4000
 M1249     | gonja                        |   4000
 M1250     | hhhq                         |   4000
 M1251     | lotte                        |  4000
 M1252     | megis                        |  4000
 M1253     | ricky                        |  4000
 M1254     | cena                         | 4000
 M1255     | vanndy cena                  |  4000
 M1256     | tiple hhh cena               |  4000
M1257      | twims                        |   4000
(27 rows)
```

View-2

```
postgres=# create view basic_info as select room.room_id,room.room_no,coustomer.coustomer_id,coustomer_name
from coustomer,room where coustomer.room_id=room.room_id;
CREATE VIEW
```

```
postgres=# \d basic_info
                View "public.basic_info"
    Column     |        Type         | Collation | Nullable | Default
---------------+---------------------+-----------+----------+---------
 room_id       | character varying(10) |           |          |
 room_no       | integer               |           |          |
 coustomer_id  | character varying(20) |           |          |
 coustomer_name | character(30)        |           |          |

postgres=# select * from basic_info;
 room_id | room_no | coustomer_id |        coustomer_name
---------+---------+--------------+------------------------------
 A101    |     101 | B112261      | Vivek More
 A102    |     102 | B112262      | Vishal More
 A104    |     104 | B112264      | Great Khali
 A105    |     105 | B112265      | Rey Mysterio
 A106    |     106 | B112266      | Charlotte Flair
 A107    |     107 | B112267      | Asuka
 A108    |     108 | B112268      | Shasha Banks
 A111    |     103 | B112271      | Otis
 A112    |     103 | B112272      | Heavy Machinery
 A114    |     102 | B112274      | Big E
 A117    |     105 | B112277      | Undertaker
 A121    |     109 | B112281      | Scallet Jonasson
 A122    |     110 | B112282      | Black Widow
 A126    |     104 | B112286      | Iron Man
 A129    |     107 | B112289      | Joseph
 A128    |     106 | B112288      | nick
 A127    |     105 | B112287      | Fury
 A125    |     103 | B112285      | Marvel
 A124    |     102 | B112284      | Virat Kholi
 A123    |     101 | B112283      | Steve Smith
 A120    |     108 | B112280      | Captain America
 A118    |     106 | B112278      | HHH
 A119    |     107 | B112279      | Steve Austin
 A130    |     108 | B112290      | Thaliava
 A103    |     103 | B112263      | john cena
 A109    |     109 | B112269      | Beky Lynch
 A110    |     110 | B112270      | Shorty G
 A113    |     101 | B112273      | The New Day
 A115    |     103 | B112275      | Kofy Kingston
 A116    |     104 | B112276      | The Street Profit
(30 rows)
```

Creating functions

* Function -1

```
postgres=# CREATE OR REPLACE FUNCTION PASTE(droom_id varchar) returns integer as'
postgres'#  declare
postgres'# dno_of_days integer;
postgres'# dcoustomer_id varchar;
postgres'# dparking_no varchar;
postgres'#  begin
postgres'#  select no_of_days into dno_of_days from room where droom_id=room_id;
postgres'#  select coustomer_id into dcoustomer_id from coustomer where droom_id=room_id;
postgres'# select parking_no into dparking_no from coustomer where droom_id=room_id;
postgres'#  update coustomer_bill set coustomer_id=dcoustomer_id where room_id=droom_id;
postgres'#  update coustomer_bill set t_room_cost=dno_of_days*800 where room_id=droom_id;
postgres'#  if  dparking_no= null then
postgres'# update coustomer_bill set parking_charge=0 where room_id=droom_id;
postgres'#  else
postgres'#      update coustomer_bill set parking_charge=dno_of_days*150 where room_id=droom_id;
postgres'#  end if;
postgres'# return 1;
postgres'#  end;
postgres'#  'LANGUAGE 'plpgsql' ;
CREATE FUNCTION
```

*Working of fuction-1

```
postgres=# select paste('A101');
 paste
-------
     1
(1 row)
```

*Function-2

```
postgres=# create or replace function total_bill(dcoustomer_id varchar) returns integer as'
postgres'#  declare
postgres'# dtotal_dish integer;
postgres'# dt_food_cost integer;
postgres'# dt_room_cost integer;
postgres'# dparking_charge integer;
postgres'#  begin
postgres'#  select count(*) from food into dtotal_dish where coustomer_id=dcoustomer_id;
postgres'# update coustomer_bill set t_food_cost=dtotal_dish*150 where coustomer_id=dcoustomer_id;
postgres'# select t_food_cost from coustomer_bill into dt_food_cost where coustomer_id=dcoustomer_id;
```

```
postgres'# select t_room_cost from coustomer_bill into dt_room_cost where coustomer_id=dcoustomer_id;
postgres'# select parking_charge from coustomer_bill into dparking_charge where coustomer_id=dcoustomer_id;
postgres'# update coustomer_bill set total_bill=dt_room_cost +dt_food_cost +dparking_charge where
coustomer_id=dcoustom
er_id;
postgres'# return total_bill from coustomer_bill where coustomer_id=dcoustomer_id;
postgres'#  end;
postgres'#  'LANGUAGE 'plpgsql' ;
CREATE FUNCTION
```

*working of function-2

```
postgres=# select total_bill('B112261');
 total_bill
--------
     4900
(1 row)
```

---

creating trigger

```
postgres=#  create or replace  function discount() returns trigger as $$
postgres$#  begin
postgres$# if new.total_bill>4500 then
postgres$# raise exception 'your bill is more than 4500, u got discount of 10%';
postgres$#  end if;
postgres$# return new;
postgres$#  end;
postgres$#  $$ language plpgsql;
CREATE FUNCTION


postgres=# create trigger tg1 before insert on coustomer_bill for each row execute procedure t1();
CREATE TRIGGER
```

*Working of trigger

```
postgres=# select total_bill('B112275');
 total_bill
--------
     4750
ERROR: your bill is more than 4500, u got discount of 10%
CONTEXT:  PL/pgSQL function t1() line 4 at RAISE
```

#creating cursor

```
postgres=# CREATE OR REPLACE FUNCTION cursor1() RETURNS int AS ' DECLARE
postgres'# Coustomer_name coustomer.coustomer_name%type;
postgres'# Room_no room.room_no%type;
postgres'# c1 cursor FOR SELECT coustomer.coustomer_name,room.room_no from coustomer,room where
coustomer.room_id=room.r
oom_id;
postgres'# BEGIN
postgres'# OPEN c1;
postgres'# LOOP
postgres'# FETCH c1 into coustomer_name,room_no;
postgres'# EXIT WHEN NOT FOUND;
postgres'# RAISE NOTICE ''coustomer_name=%,room_no=%'',coustomer_name,room_no;
postgres'# END LOOP;
postgres'# CLOSE c1;
postgres'# RETURN 1;
postgres'# END;
postgres'# 'LANGUAGE 'plpgsql';
CREATE FUNCTION
```

*Working of cursor

```
postgres=# select * from cursor1();
NOTICE:  coustomer_name=Vivek More              ,room_no=101
NOTICE:  coustomer_name=Vishal More             ,room_no=102
NOTICE:  coustomer_name=Great Khali             ,room_no=104
NOTICE:  coustomer_name=Rey Mysterio            ,room_no=105
NOTICE:  coustomer_name=Charlotte Flair         ,room_no=106
NOTICE:  coustomer_name=Asuka                   ,room_no=107
NOTICE:  coustomer_name=Shasha Banks            ,room_no=108
NOTICE:  coustomer_name=Otis                    ,room_no=103
NOTICE:  coustomer_name=Heavy Machinery         ,room_no=103
NOTICE:  coustomer_name=Big E                   ,room_no=102
NOTICE:  coustomer_name=Undertaker              ,room_no=105
NOTICE:  coustomer_name=Scallet Jonasson        ,room_no=109
NOTICE:  coustomer_name=Black Widow             ,room_no=110
NOTICE:  coustomer_name=Iron Man                ,room_no=104
NOTICE:  coustomer_name=Thaliava                ,room_no=108
NOTICE:  coustomer_name=john cena               ,room_no=103
NOTICE:  coustomer_name=Beky Lynch              ,room_no=109
NOTICE:  coustomer_name=Shorty G                ,room_no=110
NOTICE:  coustomer_name=The New Day             ,room_no=101
NOTICE:  coustomer_name=Kofy Kingston           ,room_no=103
NOTICE:  coustomer_name=The Street Profit       ,room_no=104
NOTICE:  coustomer_name=HHH                     ,room_no=106
NOTICE:  coustomer_name=Steve Austin            ,room_no=107
NOTICE:  coustomer_name=Captain America         ,room_no=108
```

```
NOTICE:  coustomer_name=Steve Smith              ,room_no=101
NOTICE:  coustomer_name=Virat Kholi              ,room_no=102
NOTICE:  coustomer_name=Marvel                   ,room_no=103
NOTICE:  coustomer_name=Fury                    ,room_no=105
NOTICE:  coustomer_name=nick                    ,room_no=106
NOTICE:  coustomer_name=Joseph                   ,room_no=107
 cursor1
------
     1
```

creataing exception

```
postgres=# create or replace function checking(dcoustomer_id in varchar) returns integer as'
postgres'# declare
postgres'# dcheck_out date;
postgres'# begin
postgres'# select check_out into dcheck_out from room,coustomer where coustomer.coustomer_id=dcoustomer_id &&
coustomer.
room_id=room.room_id;
postgres'# return 1;
postgres'# exception
postgres'# when NO_DATA_FOUND then
postgres'# return 0;
postgres'# end;
postgres'# 'language 'plpgsql';
CREATE FUNCTION
```

*Working of exception

```
postgres=#select checking('B112290');
checking
-----
   0
    (1  row)
```

Queries

1] Calculate the total bill of coustomer having coustomer_id 'B112263' .

```
postgres=# select total_bill('B112263');
 total_bill
--------
     2500
```

(1 row)

```
postgres=# select * from coustomer_bill where coustomer_id='B112263';
 coustomer_id | room_id | t_room_cost | t_food_cost | parking_charge | total_bill
--------------+---------+-------------+-------------+----------------+------------
 B112263      | A103    |        1600 |         600 |            300 |       2500
(1 row)
```

2] check whether coustomer having coustomer_id 'B112288' have check_out.

```
postgres=#select checking('B112288');
checking
----------
        0
```

```
postgres=# select * from coustomer where coustomer_id='B112288' ;
 coustomer_id | room_id |        coustomer_name        | parking_no
--------------+---------+------------------------------+------------
B112288      | A128    | nick                         | nh01hg1678
(1 row)
```

```
postgres=# select * from room where room_id='A128';
 room_id | room_no | no_of_days |          check_in          | check_out
---------+---------+------------+----------------------------+-----------
 A128    |     106 |          5 | 2020-2-13                  |
(1 row)
```

3] select names of hotel members their work and their salary of those members whose salary is more than 10,000.

```
postgres=# select member_name,type_of_work,salary from hotel_member where salary>10000;
        member_name         |     type_of_work     | salary
----------------------------+----------------------+--------
 Anshul Nathe               | CEO                  | 100000
 Jisha lade                 | Data Handler         |  80000
 Sandeep Mundhe             | waiter               |  50000
 Mayur Patil                | waiter               |  50000
 Mausom                     | Cleaning             |  50000
 Vivek                      | Supervisor           |  50000
 Namdev                     | cook                 |  60000
 singh                      | cook                 |  60000
 Arman                      | Laundry              |  60000
 Malik                      | dish washer          |  60000
(10 rows)
```

4] find coustomer,their ids and no of days living in the hotel

```
 postgres=# select room.no_of_days,coustomer.coustomer_name,coustomer.coustomer_id from coustomer,room
where coustomer.ro
om_id=room.room_id;
 no_of_days |        coustomer_name        | coustomer_id
------------+------------------------------+--------------
          5 | Vivek More                   | B112261
          3 | Vishal More                  | B112262
          6 | Great Khali                  | B112264
          4 | Rey Mysterio                 | B112265
          1 | Charlotte Flair              | B112266
          2 | Asuka                        | B112267
          3 | Shasha Banks                 | B112268
          1 | Otis                         | B112271
          5 | Heavy Machinery              | B112272
          6 | Big E                        | B112274
          2 | Undertaker                   | B112277
          3 | Scallet Jonasson             | B112281
          2 | Black Widow                  | B112282
          3 | Iron Man                     | B112286
          3 | Joseph                       | B112289
          5 | nick                         | B112288
          7 | Fury                         | B112287
          1 | Marvel                       | B112285
          2 | Virat Kholi                  | B112284
          2 | Steve Smith                  | B112283
          5 | Captain America              | B112280
          1 | HHH                          | B112278
          6 | Steve Austin                 | B112279
          4 | Thaliava                     | B112290
          2 | john cena                    | B112263
          2 | Beky Lynch                   | B112269
          1 | Shorty G                     | B112270
          3 | The New Day                  | B112273
          5 | Kofy Kingston                | B112275
          4 | The Street Profit            | B112276
(30 rows)
```