THE MAHARAJA SAYAJIRAO UNIVERSITY OF BARODA

सत्यं शिवं सुन्दरम्

ORACLE

SQL

A DBMS PROJECT

SPIRITUAL SPHERE

SUBJECT TEACHER: ANJALI G JIVANI

◈ PROJECT BY:

NAME	SEAT NO.	PRN
JANKI H. THAKKAR	453074	8021074876
NANDINI Y. PARMAR	453037	8021006769
KEYA D. SHAH	453066	8021074727
HARSHVARDHAN S. SHINDE	453069	8021079203

INDEX

What is Spiritual Sphere?

Temple Management System in Database Management System refers to the design and implementation of a database system to manage the various activities and operations of a temple. This project involves creating a system to store, organize and manage information related to the temple's resources, events, staff and members. This system is designed in such a way that it meets the specific needs of the temple and its community. It includes various kinds of managements like

- Employee Management: The employee management feature in a temple management system helps in maintaining a database of members, which includes their personal details and contact information.

- Event Management: Event Management is an important feature of a temple management system, which helps in managing various events and ceremonies organized by the temple.

- Donation Management: A temple management system also helps in managing donations and contributions from members and other donors.

And Many more....

So let's Explore...

Assumptions that we've made

- All booking tables are dynamic in nature.

- Combo booking is not taken into consideration. It will be only available in temple at respective timings.

- All ac and non ac rooms have same prices accordingly.

- The fests starting at respective dates, having a schedule of days, would have same starting and ending time every day.

Cardinality

1. Emp-Dept m:1
2. Donation-Dept m:1
3.Timings-Dept 1:m
4. Aarti-Aarti_Booking 1:m
5. Prasad-Prasad_Booking 1:m
6.Accomodation-Rooms_booking 1:m
7.Accomodation-Rooms_price m:1
8. Trustee-Dept m:1
9. Combo_prasad-prasad m:n
10. Dept-Is A- Aarti, Prasad, Darshan, Accommodation
11. Fest_info-Fest 1:m

• Description of Normalised Tables

1. DEPT (dept_id, dname, d_budget, start_time, end_time)
- Normalized Form: 3rd NF
- Primary key: dept_id
- Various departments are inherited from the dept entity using 'IS A' relationship.

2. EMP (emp_id, dept_id, e_name, sal, mob_no, e_mail, job, hiredate)
- Normalized Form: 3rd NF
- Primary Key: emp_id
- Foreign Key: dept_id refers to Dept table's dept_id
- Employees work in various departments.

3. DARSHAN (d_type, start_time, end_time, dept_id)
- Normalized Form: 3rd NF
- Primary Key: d_type
- Foreign Key: dept_id refers to dept table dept_id
- d_type includes various darshans like 'Sarva, VIP etc.'

4. AARTI (a_type, start_time, end_time, pid, dept_id)
- Normalized Form: 3rd NF
- Primary Key: a_type
- Foreign Keys: pid & dept_id
- Pid refers to emp table emp_id
- Dept_id refers to dept table dept_id
- A_type includes various aartis like 'Mangla, Shringar, Rajbhog etc.'

5. PRASHAD (name, price, dept_id)
- Normalized Form: 3rd NF
- Primary Key: name
- Foreign Key: dept_id which refers to the dept table dept_id
- name stores different types of prashad like 'Ladoo, Mysore Pak etc.'

6. COMBO (pr_comboid, price)
- Normalized Form: 3rd NF
- Primary Key: pr_comboid

7. FEST (f_id, f_name, dept_id, total_days)
- Normalized Form: 3rd NF
- Primary Key: f_id
- Foreign Key: dept_id which refers to the dept table dept_id

8. FEST_INFO (f_id, start_date, budget, start_time, end_time, mgr)
- Normalized Form: 3rd NF
- Primary Key: f_id & start_date
- This table is a weak entity as it depends on fest table
- Here start_date is a discriminator and with mixing of fest table primary key it becomes unique
- Foreign Key: f_id & mgr
- f_id refers to Fest table f_id
- mgr refers to Emp table emp_id
- Every year the budget of respective fest might be changed or not

9. TRUSTEE (t_id, t_name, t_mob, t_mail, dept_id)
- Normalized Form: 3rd NF
- Primary Key: t_id
- Foreign Key: dept_id
- One department is governed by many employees

10. ACCOMODATION (r_id, r_type, capacity, dept_id)
- Normalized Form: 3rd NF
- Primary Key: r_id
- Foreign Key: r_type & dept_id
- r_type refers to room_price table r_type
- dept_id refers to the dept table dept_id

11. DESCRIBE_COMBO (pr_comboid, name, quantity)
- Normalized Form: 3rd NF
- Primary Key: pr_comboid & name
- This is a many to many relationships' entity
- Foreign Key: pr_comboid & name
- Pr_comboid refers to combo table pr_comboid
- name refers to prashad table name

12. ROOM_PRICE (r_type, price)
- Normalized Form: 3rd NF
- Primary Key: r_type

13. AARTI_BOOKING (ab_id, a_type, booked_by, amt, date)
- Normalized Form: 3rd NF
- Primary Key: ab_id
- Foreign Key: a_type
- a_type refers to aarti table a_type

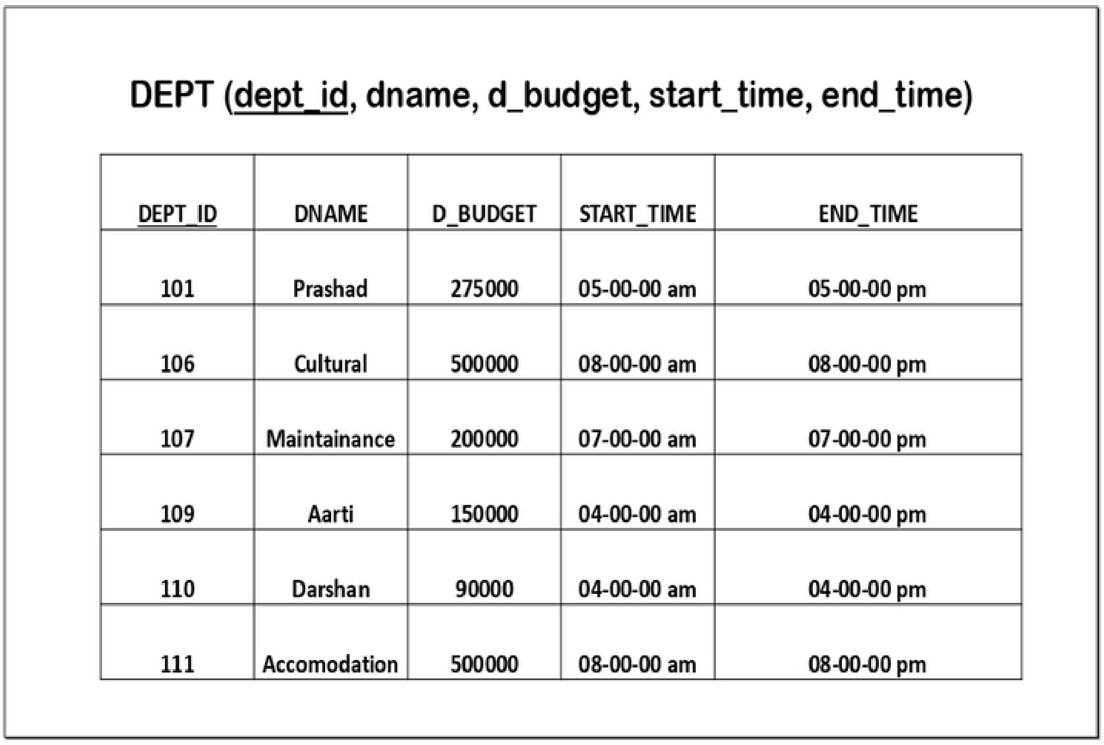
14. PRASAD_BOOKING (pb_id, name, booked_by, quantity, amt, date)
- Normalized Form: 3rd NF
- Primary Key: pb_id
- Foreign Key: name refers to table prashad name

15. ROOM_BOOKING (rb_id, booked_by, date, amt, room_type)
- Normalized Form: 3rd NF
- Primary Key: rb_id
- Foreign Key: room_type which refers to the table accommodation room_type

16. DETAILS (ab_id, member_names)
- Normalized Form: 3rd NF
- Primary Key: ab_id & member_names
- It is a weak entity
- Member_names is a discriminator therefore and with the mixing of
aarti_booking primary key it becomes unique

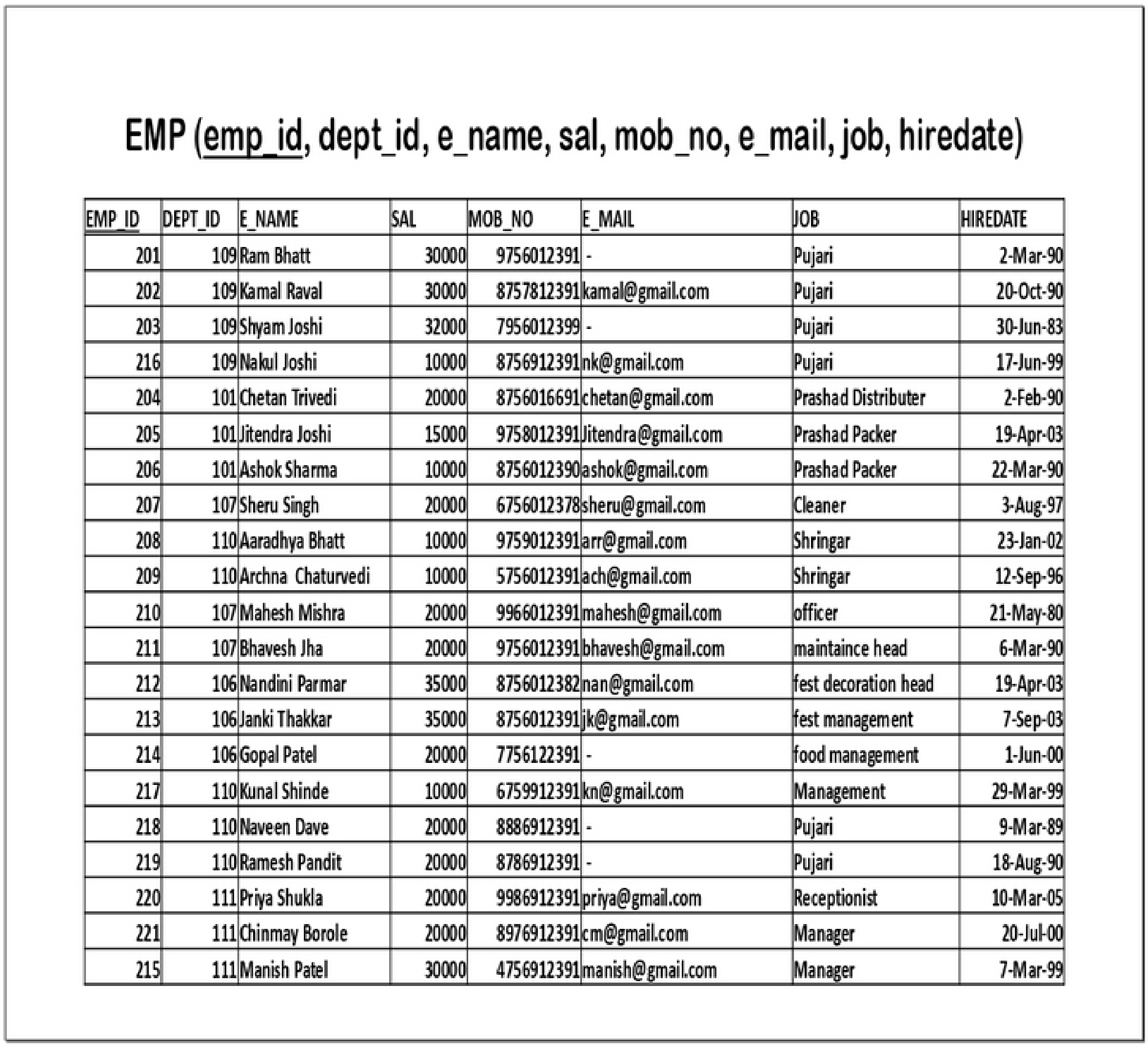
17. DONATION (d_id, donor_name, amt, dept_id, d_date)
- Normalized Form: 3rd NF
- Primary Key: d_id
- Foreign Key: dept_id which refers to the dept table dept_id

TABLES



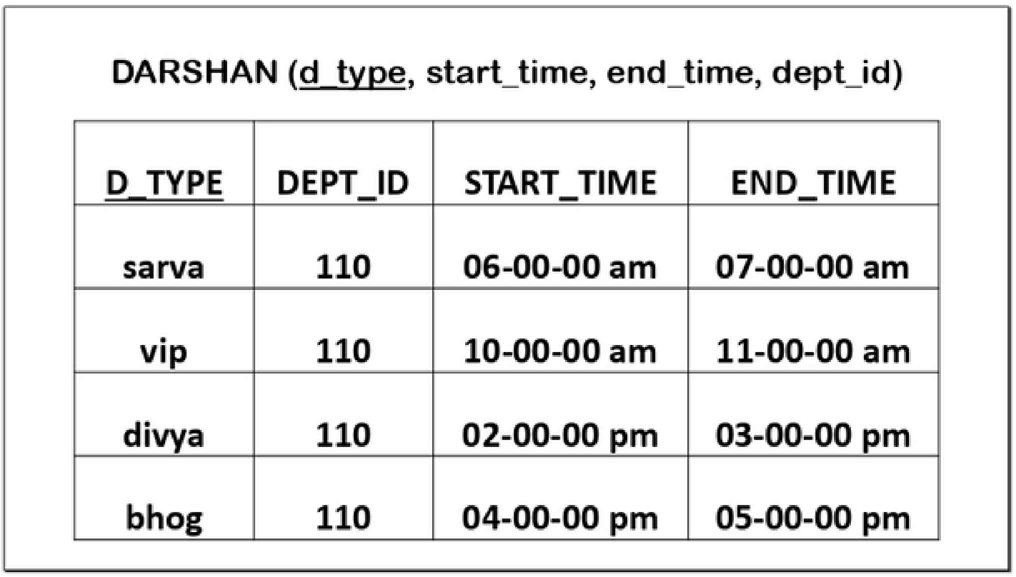
DEPT (dept_id, dname, d_budget, start_time, end_time)

DEPT_ID	DNAME	D_BUDGET	START_TIME	END_TIME
101	Prashad	275000	05-00-00 am	05-00-00 pm
106	Cultural	500000	08-00-00 am	08-00-00 pm
107	Maintainance	200000	07-00-00 am	07-00-00 pm
109	Aarti	150000	04-00-00 am	04-00-00 pm
110	Darshan	90000	04-00-00 am	04-00-00 pm
111	Accomodation	500000	08-00-00 am	08-00-00 pm



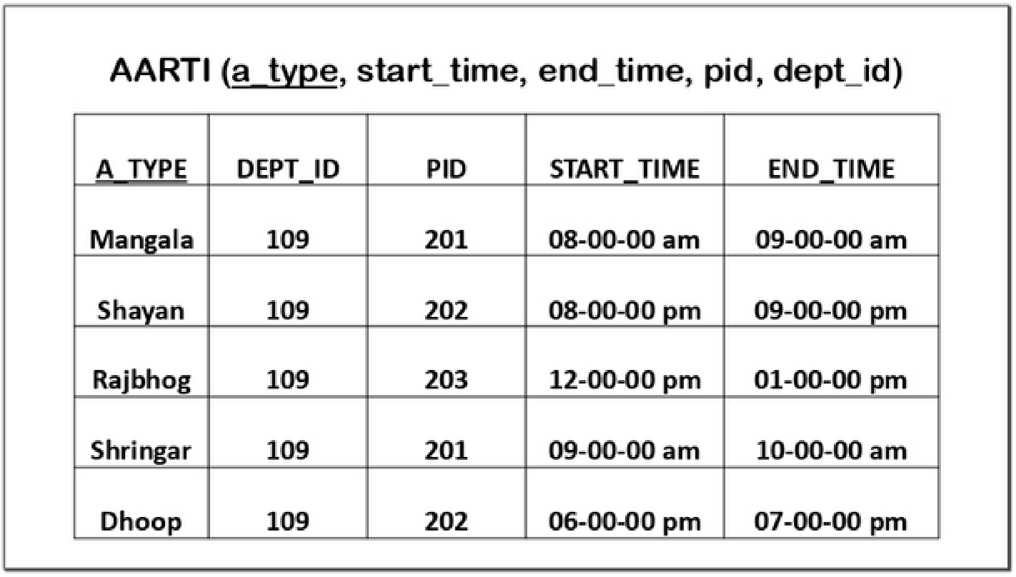
EMP (emp_id, dept_id, e_name, sal, mob_no, e_mail, job, hiredate)

EMP_ID	DEPT_ID	E_NAME	SAL	MOB_NO	E_MAIL	JOB	HIREDATE
201	109	Ram Bhatt	30000	9756012391	-	Pujari	2-Mar-90
202	109	Kamal Raval	30000	8757812391	kamal@gmail.com	Pujari	20-Oct-90
203	109	Shyam Joshi	32000	7956012399	-	Pujari	30-Jun-83
216	109	Nakul Joshi	10000	8756912391	nk@gmail.com	Pujari	17-Jun-99
204	101	Chetan Trivedi	20000	8756016691	chetan@gmail.com	Prashad Distributer	2-Feb-90
205	101	Jitendra Joshi	15000	9758012391	Jitendra@gmail.com	Prashad Packer	19-Apr-03
206	101	Ashok Sharma	10000	8756012390	ashok@gmail.com	Prashad Packer	22-Mar-90
207	107	Sheru Singh	20000	6756012378	sheru@gmail.com	Cleaner	3-Aug-97
208	110	Aaradhya Bhatt	10000	9759012391	arr@gmail.com	Shringar	23-Jan-02
209	110	Archna Chaturvedi	10000	5756012391	ach@gmail.com	Shringar	12-Sep-96
210	107	Mahesh Mishra	20000	9966012391	mahesh@gmail.com	officer	21-May-80
211	107	Bhavesh Jha	20000	9756012391	bhavesh@gmail.com	maintaince head	6-Mar-90
212	106	Nandini Parmar	35000	8756012382	nan@gmail.com	fest decoration head	19-Apr-03
213	106	Janki Thakkar	35000	8756012391	jk@gmail.com	fest management	7-Sep-03
214	106	Gopal Patel	20000	7756122391	-	food management	1-Jun-00
217	110	Kunal Shinde	10000	6759912391	kn@gmail.com	Management	29-Mar-99
218	110	Naveen Dave	20000	8886912391	-	Pujari	9-Mar-89
219	110	Ramesh Pandit	20000	8786912391	-	Pujari	18-Aug-90
220	111	Priya Shukla	20000	9986912391	priya@gmail.com	Receptionist	10-Mar-05
221	111	Chinmay Borole	20000	8976912391	cm@gmail.com	Manager	20-Jul-00
215	111	Manish Patel	30000	4756912391	manish@gmail.com	Manager	7-Mar-99



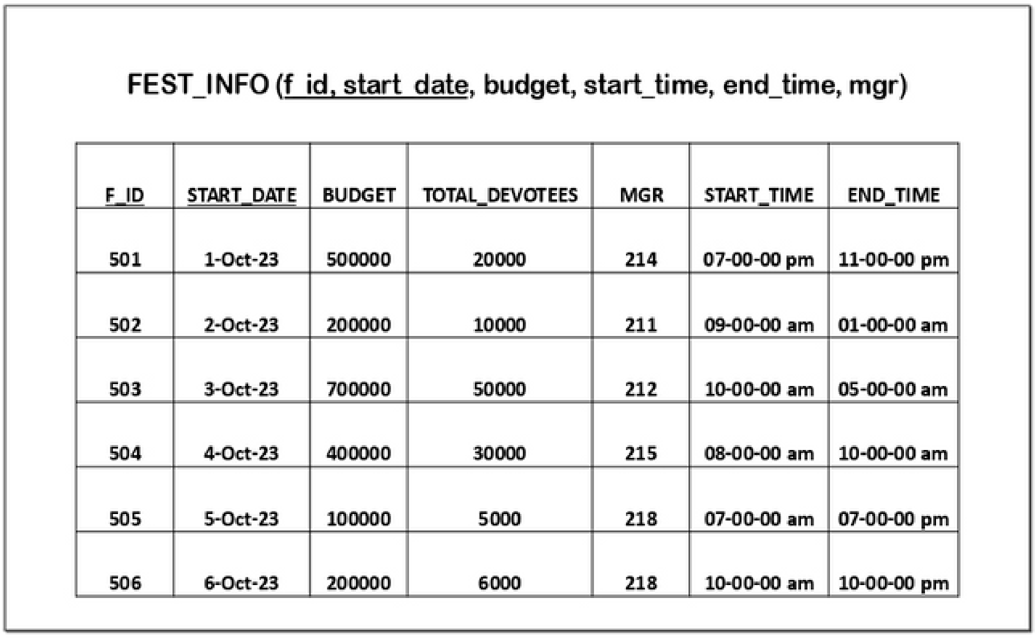
DARSHAN (d_type, start_time, end_time, dept_id)

D_TYPE	DEPT_ID	START_TIME	END_TIME
sarva	110	06-00-00 am	07-00-00 am
vip	110	10-00-00 am	11-00-00 am
divya	110	02-00-00 pm	03-00-00 pm
bhog	110	04-00-00 pm	05-00-00 pm



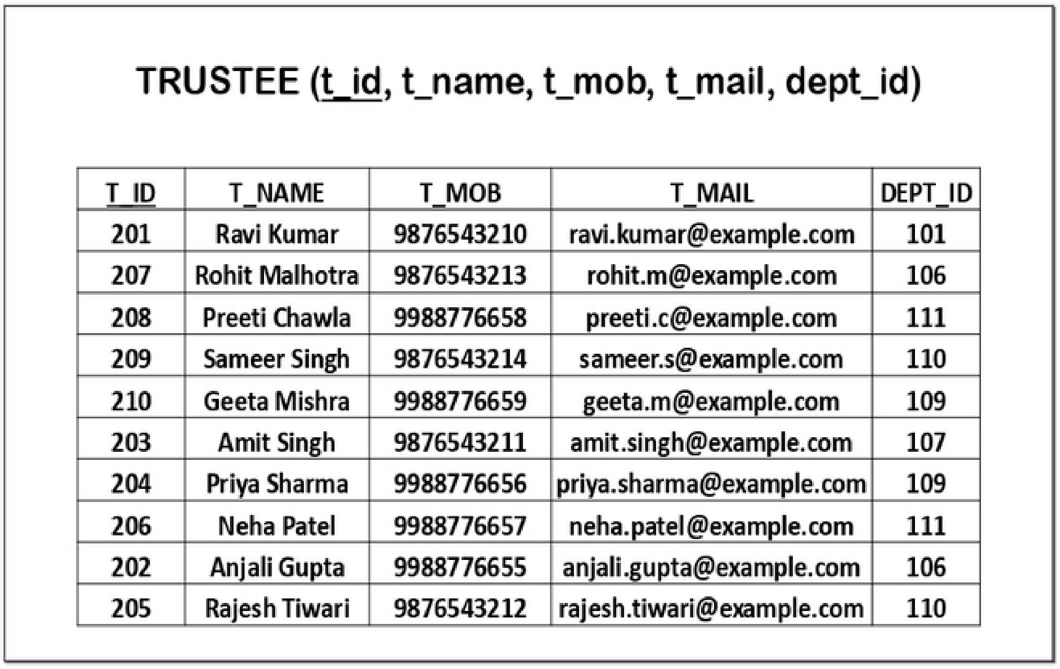
AARTI (a_type, start_time, end_time, pid, dept_id)

A_TYPE	DEPT_ID	PID	START_TIME	END_TIME
Mangala	109	201	08-00-00 am	09-00-00 am
Shayan	109	202	08-00-00 pm	09-00-00 pm
Rajbhog	109	203	12-00-00 pm	01-00-00 pm
Shringar	109	201	09-00-00 am	10-00-00 am
Dhoop	109	202	06-00-00 pm	07-00-00 pm



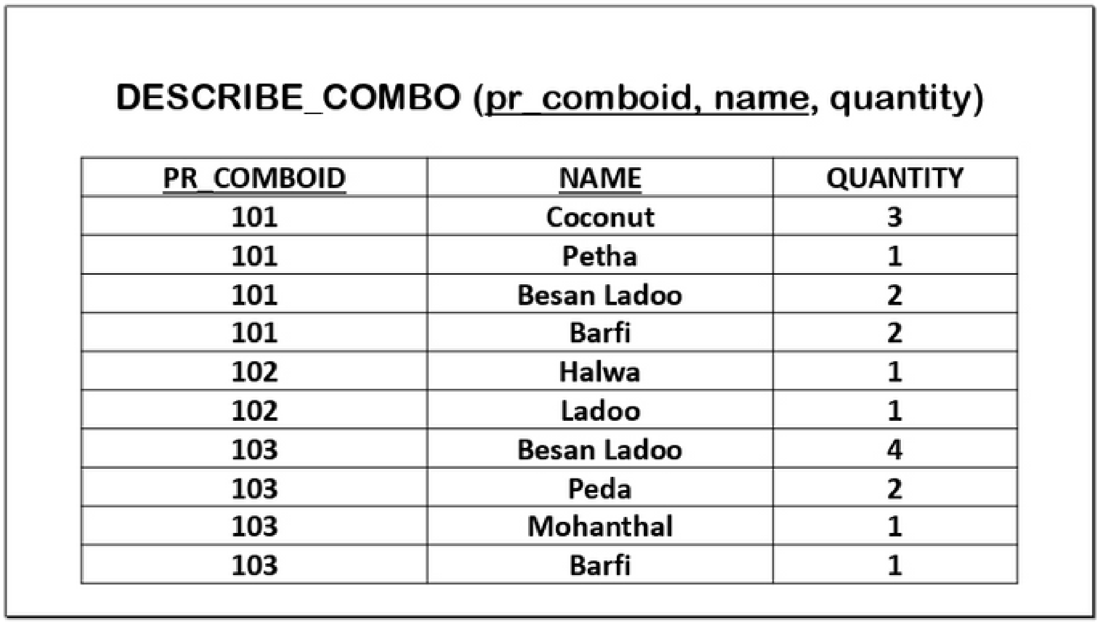
FEST_INFO (f_id, start_date, budget, start_time, end_time, mgr)

F_ID	START_DATE	BUDGET	TOTAL_DEVOTEES	MGR	START_TIME	END_TIME
501	1-Oct-23	500000	20000	214	07-00-00 pm	11-00-00 pm
502	2-Oct-23	200000	10000	211	09-00-00 am	01-00-00 am
503	3-Oct-23	700000	50000	212	10-00-00 am	05-00-00 am
504	4-Oct-23	400000	30000	215	08-00-00 am	10-00-00 am
505	5-Oct-23	100000	5000	218	07-00-00 am	07-00-00 pm
506	6-Oct-23	200000	6000	218	10-00-00 am	10-00-00 pm



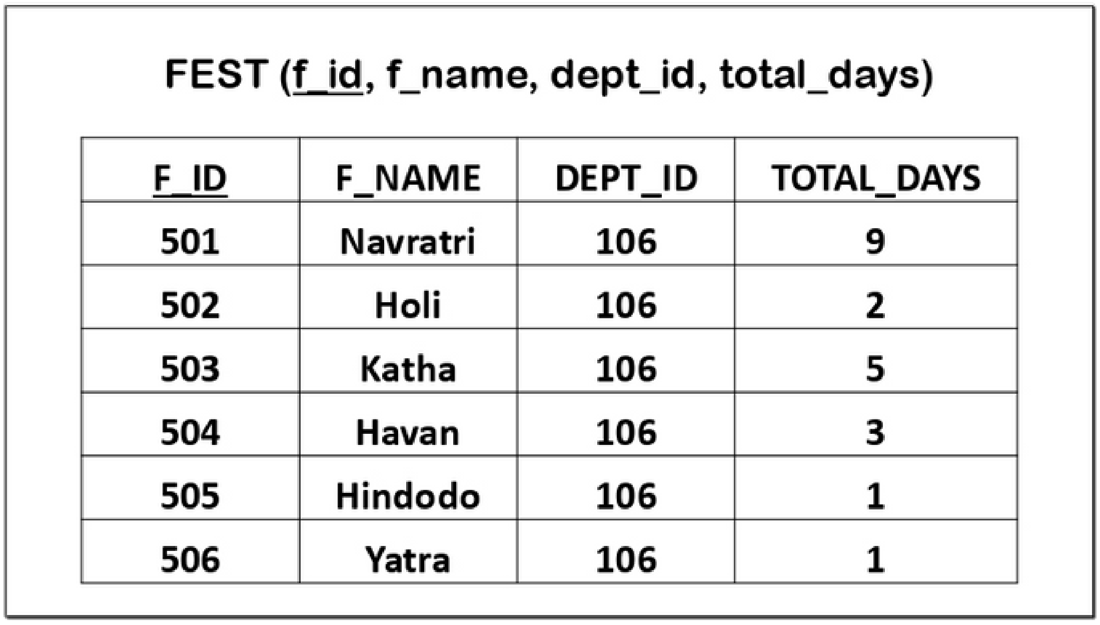
TRUSTEE (t_id, t_name, t_mob, t_mail, dept_id)

T_ID	T_NAME	T_MOB	T_MAIL	DEPT_ID
201	Ravi Kumar	9876543210	ravi.kumar@example.com	101
207	Rohit Malhotra	9876543213	rohit.m@example.com	106
208	Preeti Chawla	9988776658	preeti.c@example.com	111
209	Sameer Singh	9876543214	sameer.s@example.com	110
210	Geeta Mishra	9988776659	geeta.m@example.com	109
203	Amit Singh	9876543211	amit.singh@example.com	107
204	Priya Sharma	9988776656	priya.sharma@example.com	109
206	Neha Patel	9988776657	neha.patel@example.com	111
202	Anjali Gupta	9988776655	anjali.gupta@example.com	106
205	Rajesh Tiwari	9876543212	rajesh.tiwari@example.com	110



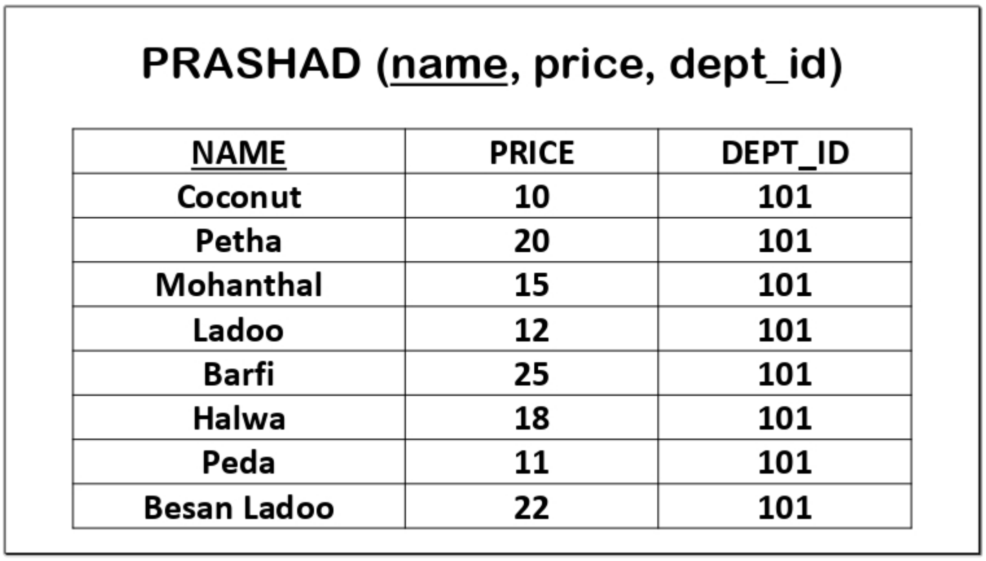
DESCRIBE_COMBO (pr_comboid, name, quantity)

PR_COMBOID	NAME	QUANTITY
101	Coconut	3
101	Petha	1
101	Besan Ladoo	2
101	Barfi	2
102	Halwa	1
102	Ladoo	1
103	Besan Ladoo	4
103	Peda	2
103	Mohanthal	1
103	Barfi	1



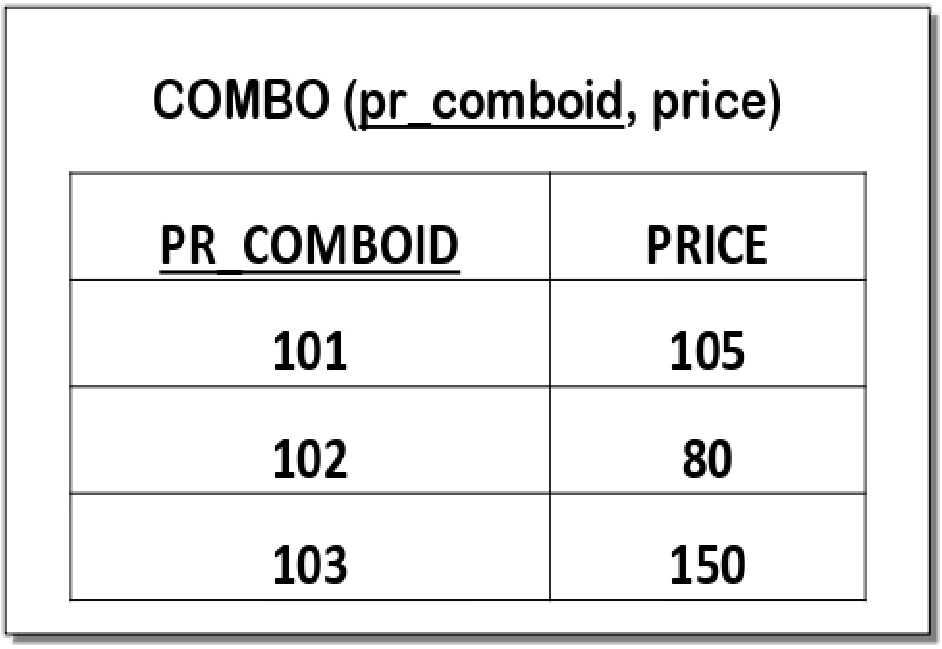
FEST (f_id, f_name, dept_id, total_days)

F_ID	F_NAME	DEPT_ID	TOTAL_DAYS
501	Navratri	106	9
502	Holi	106	2
503	Katha	106	5
504	Havan	106	3
505	Hindodo	106	1
506	Yatra	106	1



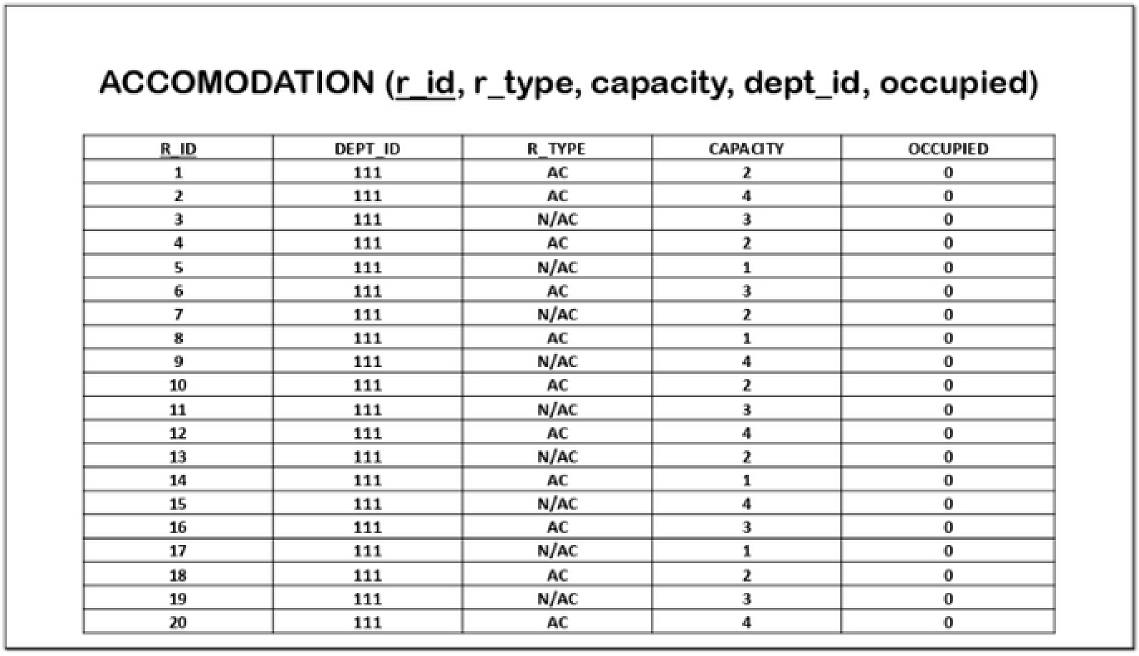
PRASHAD (name, price, dept_id)

NAME	PRICE	DEPT_ID
Coconut	10	101
Petha	20	101
Mohanthal	15	101
Ladoo	12	101
Barfi	25	101
Halwa	18	101
Peda	11	101
Besan Ladoo	22	101



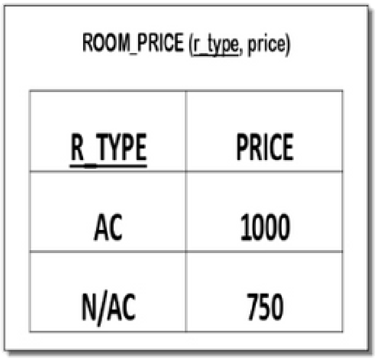
COMBO (pr_comboid, price)

PR_COMBOID	PRICE
101	105
102	80
103	150



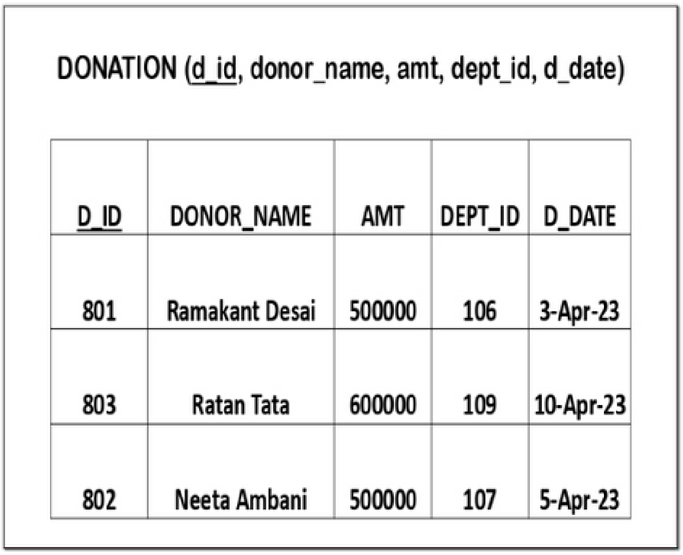
ACCOMODATION (r_id, r_type, capacity, dept_id, occupied)

R_ID	DEPT_ID	R_TYPE	CAPACITY	OCCUPIED
1	111	AC	2	0
2	111	AC	4	0
3	111	N/AC	3	0
4	111	AC	2	0
5	111	N/AC	1	0
6	111	AC	3	0
7	111	N/AC	2	0
8	111	AC	1	0
9	111	N/AC	4	0
10	111	AC	2	0
11	111	N/AC	3	0
12	111	AC	4	0
13	111	N/AC	2	0
14	111	AC	1	0
15	111	N/AC	4	0
16	111	AC	3	0
17	111	N/AC	1	0
18	111	AC	2	0
19	111	N/AC	3	0
20	111	AC	4	0



ROOM_PRICE (r_type, price)

R_TYPE	PRICE
AC	1000
N/AC	750



DONATION (d_id, donor_name, amt, dept_id, d_date)

D_ID	DONOR_NAME	AMT	DEPT_ID	D_DATE
801	Ramakant Desai	500000	106	3-Apr-23
803	Ratan Tata	600000	109	10-Apr-23
802	Neeta Ambani	500000	107	5-Apr-23

FUNCTION, PROCEDURE AND TRIGGER

◈ A package which contains functions to calculate room amount and prasad price:

```
create or replace package calculate_amt as
  function room_amt (r ACCOMODATION.r_id%type,d
ROOM_BOOKING.no_of_days%type) return room_booking.amount%type;
  function get_price (n PRASHAD.name%type,quantity
prashad_booking.quantity%type)  return number;
end calculate_amt;

create or replace package body calculate_amt as
function room_amt (r ACCOMODATION.r_id%type,d
ROOM_BOOKING.no_of_days%type) return room_booking.amount%type
is
p number(4);
begin
select price into p from ROOM_PRICE where r_type=(select r_type from
ACCOMODATION where r_id=r);
return p*d;
end room_amt;

function get_price (n PRASHAD.name%type,quantity
prashad_booking.quantity%type)  return number
   is
a prashad.price%type;
begin
select price into a from prashad where name = n;
return a*quantity;
end get_price;

end calculate_amt;
```

◈ A package which contain procedure to display Aarti, prasad and room details:

```
create or replace package detail
as
 procedure get_details (a in number,b in number);
end  detail;
/

create or replace package body detail
    as
 procedure p_aarti_details(a in out details.ab_id%type)
is
    cursor c1 is select * from details order by ab_id;
begin
    dbms_output.put_line('Name of family members');
    for r1 in c1
    loop
     if(r1.ab_id = a)
     then
      dbms_output.put_line(r1.member_names);
  end if;
 end loop;
end p_aarti_details;

 procedure get_details
(a in number,b in number)
as
    rb ROOM_BOOKING%rowtype;
 ab AARTI_BOOKING%rowtype;
 pb PRASHAD_BOOKING%rowtype;
begin
if(a=1) then
    select * into rb from ROOM_BOOKING where rb_id=b;
dbms_output.put_line('Booking id '||rb.rb_id);
dbms_output.put_line('booked by '||rb.booked_by);
dbms_output.put_line('amt :'||rb.amount);
dbms_output.put_line('on date :'||rb.rb_date);

elsif(a=2) then
select * into ab from AARTI_BOOKING where ab_id=b;
dbms_output.put_line('Booking id '||ab.ab_id);
dbms_output.put_line('Type of aarti '||ab.a_type);
dbms_output.put_line('booked by '||ab.booked_by);
dbms_output.put_line('amt :'||ab.amt);
dbms_output.put_line('on date :'||ab.b_date);
 p_aarti_details(ab.ab_id);

elsif(a=3) then
select * into pb from PRASHAD_BOOKING where pb_id=b;
dbms_output.put_line('Token No: '||pb.pb_id);
dbms_output.put_line('booked_by '||pb.booked_by);
dbms_output.put_line('name,quantity : '||pb.name||'
'||pb.quantity);
dbms_output.put_line('amt : '||pb.amount);
end if;
end get_details;

end detail;
/
```

◈A function to calculate the total donation amount received by a particular department:

```
create or replace function get_donation_amount
(d DEPT.dept_id%type)
return number
is
cursor c1 is select * from DONATION where dept_id=d;
amt number(10,2);
begin
 amt:=0;
for r1 in c1
loop
amt:=amt+r1.amt;
end loop;
return amt;
end get_donation_amount;
```

◈Triggers to show the details of Aarti and Prasad Booking details as the booking is done:

```
create or replace trigger t_aarti
after insert on aarti_booking
for each row
begin
 dbms_output.put_line('Booking id '||:new.ab_id);
   dbms_output.put_line('Type of aarti '||:new.a_type);
      dbms_output.put_line('booked by '||:new.booked_by);
      dbms_output.put_line('amt :'||:new.amt);
      dbms_output.put_line('on date :'||:new.b_date);
end t_aarti;
/

create or replace trigger t_prashad
after insert on prashad_booking
for each row
begin
dbms_output.put_line('Token No: '||:new.pb_id);
dbms_output.put_line('booked_by '||:new.booked_by);
   dbms_output.put_line('name,quantity : '||:new.name||'
'||:new.quantity);
   dbms_output.put_line('amt : '||:new.amount);
end t_prashad;
/
```

◈Package for getting Aarti and Darshan timings using views:

```
create or replace view v_aarti (type,start_time,end_time) as select
a_type,to_char(start_time,'hh-mi-ss am'),to_char(end_time,'hh-mi-ss am')
from AARTI with read only;
create or replace view v_darshan (type,start_time,end_time) as select d_type
,to_char(start_time,'hh-mi-ss am'),to_char(end_time,'hh-mi-ss am')  from
DARSHAN with read only;


create or replace package view_timings
as
procedure aarti_timings(c1 in out sys_refcursor);
procedure darshan_timings(c1 in out sys_refcursor);
end view_timings;

create or replace  package body view_timings
as
procedure aarti_timings(c1 in out sys_refcursor)
as
begin
open c1 for
select * from v_aarti;
end aarti_timings;
procedure darshan_timings(c1 in out sys_refcursor)
as
begin
open c1 for
select * from v_darshan;
end darshan_timings;
end view_timings;
/
```

◈A Procedure to display details about a fest in next 5 days:

```
create or replace procedure view_festdetails
is
r1 FEST_INFO%rowtype;
name FEST.f_name%type;
day1 FEST.total_days%type;
temp number(3);
cursor c1 is select * from FEST_INFO;
begin
for r1 in c1
loop
   select total_days into day1 from FEST where f_id=r1.f_id;
   temp:=round(r1.start_date-sysdate+day1);
   if temp<=5 and temp>0 then
   select f_name into name from FEST where f_id=r1.f_id;

   dbms_output.put_line('NAME : '||name||' START DATE :
'||r1.start_date||' NUMBER OF DAYS : '||day1||' START TIME :
'||to_char(r1.start_time,'hh-mi-ss am')||' END TIME :
'||to_char(r1.end_time,'hh-mi-ss am'));
end if;
end loop;
end view_festdetails;
```

◈A Trigger for Room booking using a function to check room vacancy:

```
create or replace trigger t_room
   before insert on room_booking
   for each row
   begin
   if(check1(:new.room_id)) then
 dbms_output.put_line('Booking id '||:new.rb_id);
   dbms_output.put_line('Room no '||:new.room_id);
   dbms_output.put_line('booked by '||:new.booked_by);
   dbms_output.put_line('amt
:'||calculate_amt.room_amt(:new.room_id,:new.no_of_days));
   dbms_output.put_line('on date :'||:new.rb_date);
else
   dbms_output.put_line('booked');
   raise_application_error(-20000,'booked already');
end if;
end t_room;
/
create or replace function check1
 (r ACCOMODATION.r_id%type)
return boolean
as
cursor c1 is select * from ROOM_BOOKING where room_id=r order by rb_date desc;
r1 ROOM_BOOKING%rowtype;
begin
open c1;
   fetch c1 into r1;
   loop
exit when c1%rowcount=2 or c1%notfound;
if(round(sysdate-r1.rb_date-r1.no_of_days)>0) then
return true;
else return false;
end if;
end loop;
close c1;
return true;
exception
   when no_data_found then return true;
when others then return true;
end check1;
/
```

◈A view to display details of vacant rooms:

```
create or replace view room_vacancy as select
a.r_id,a.r_type,a.capacity,rp.price from
ACCOMODATION a,ROOM_PRICE rp where r_id not in
(select distinct(room_id) from room_booking where
round(sysdate-rb_date-no_of_days)<0) and
a.r_type=rp.r_type order by r_id
```