American International University-Bangladesh

Final Term Report

Project Title: Tour Guide Management System

Course Name: ADVANCE DATABASE MANAGEMENT SYSTEM

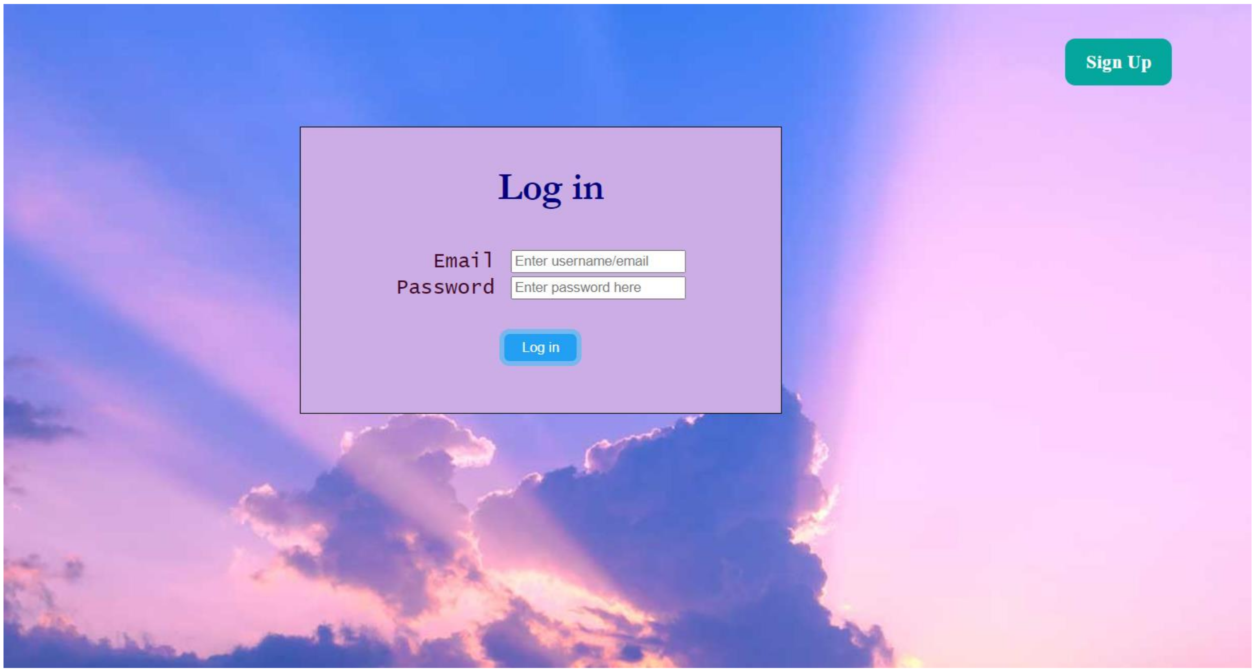
Course Teacher: Rezwan Ahmed

Semester: Summer 21-22

Name	ID	Section
Anik Sen	**20-42138-1**	**C**
Abdullah Yousuf Noman	**20-42130-1**	**C**
Musfiqur Rahman	**20-42558-1**	**C**
Ananna Rashid Kumkum	**20-43589-1**	**C**

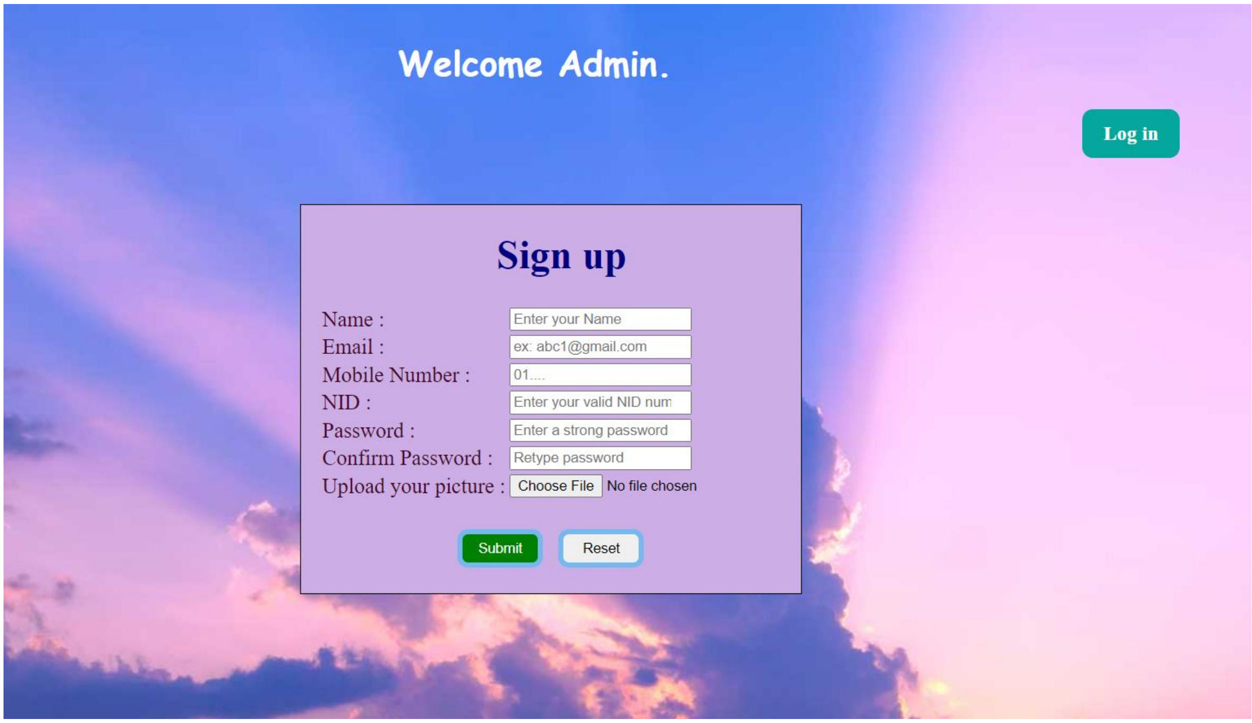
Table of Contents

- ❖ **We created the admin pannel page where an admin can log in to the system, if he doesn’t have an account he can easily create a new account from sign up option.**



Log in

Email Enter username/email
Password Enter password here

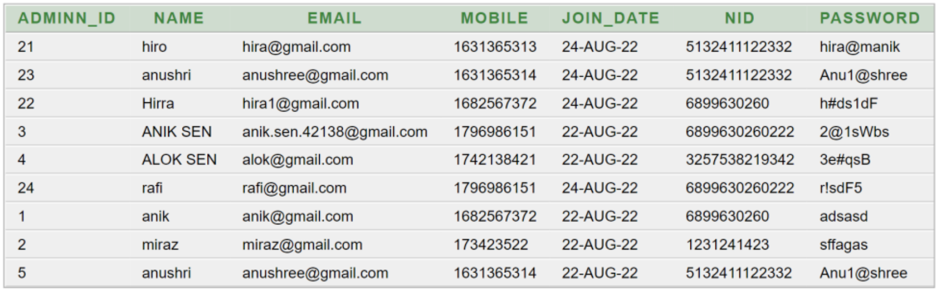


Welcome Admin.

Sign up

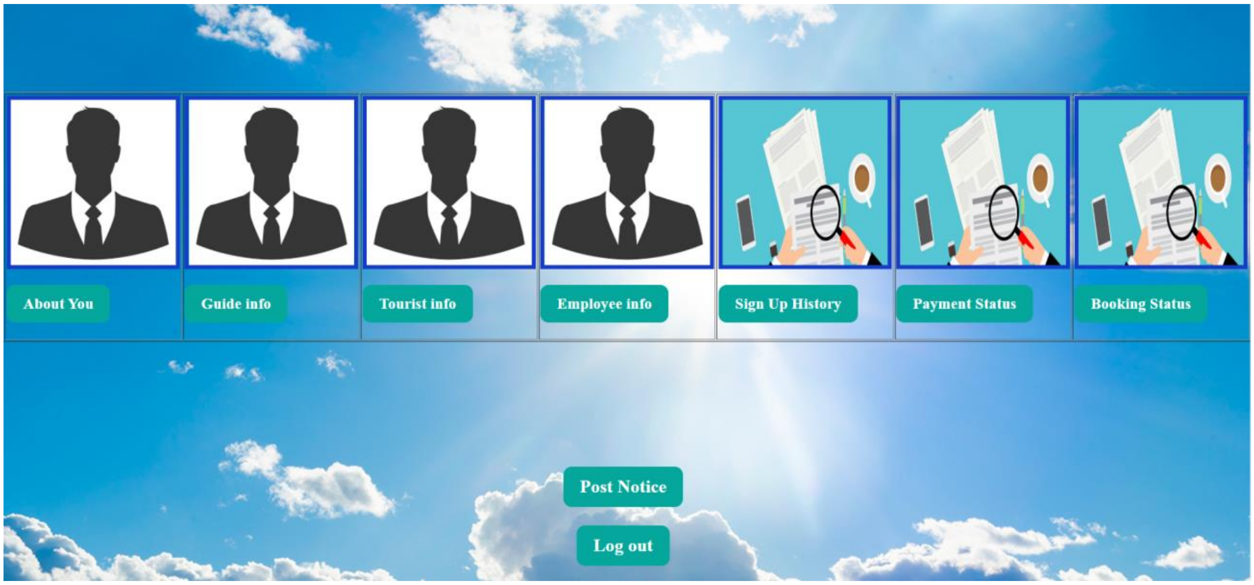
Name : Enter your Name
Email : ex: abc1@gmail.com
Mobile Number : 01....
NID : Enter your valid NID num
Password : Enter a strong password
Confirm Password : Retype password
Upload your picture : Choose File No file chosen

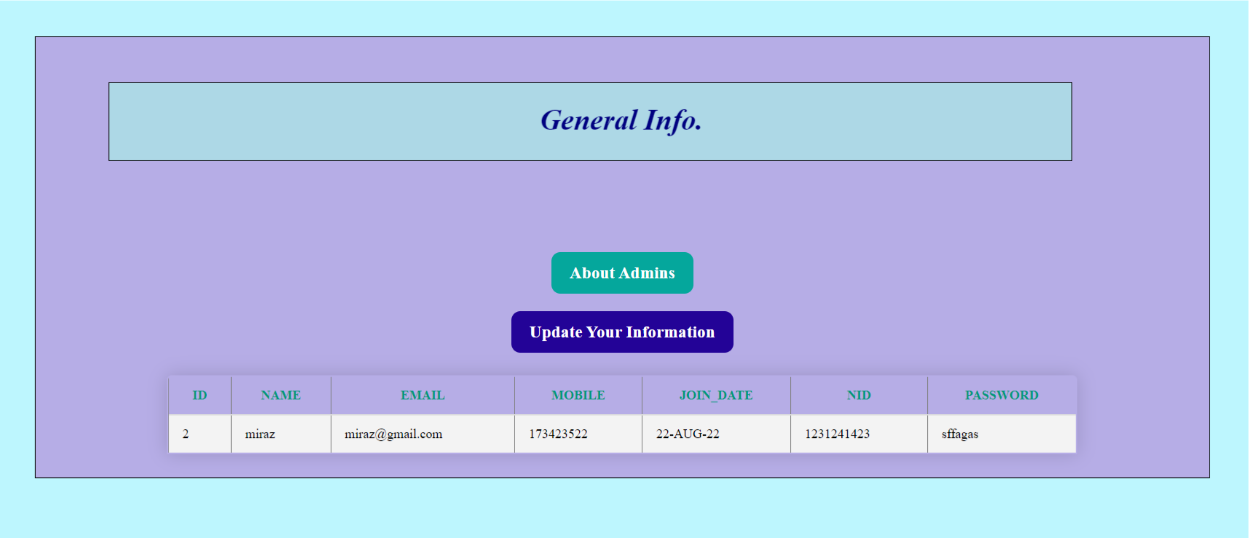
- Admin's database where data stored after sign up.



ADMINN_ID	NAME	EMAIL	MOBILE	JOIN_DATE	NID	PASSWORD
21	hiro	hira@gmail.com	1631365313	24-AUG-22	5132411122332	hira@manik
23	anushri	anushree@gmail.com	1631365314	24-AUG-22	5132411122332	Anu1@shree
22	Hirra	hira1@gmail.com	1682567372	24-AUG-22	6899630260	h#ds1dF
3	ANIK SEN	anik.sen.42138@gmail.com	1796986151	22-AUG-22	6899630260222	2@1sWbs
4	ALOK SEN	alok@gmail.com	1742138421	22-AUG-22	3257538219342	3e#qsB
24	rafi	rafi@gmail.com	1796986151	24-AUG-22	6899630260222	r!sdF5
1	anik	anik@gmail.com	1682567372	22-AUG-22	6899630260	adsasd
2	miraz	miraz@gmail.com	173423522	22-AUG-22	1231241423	sffagas
5	anushri	anushree@gmail.com	1631365314	22-AUG-22	5132411122332	Anu1@shree

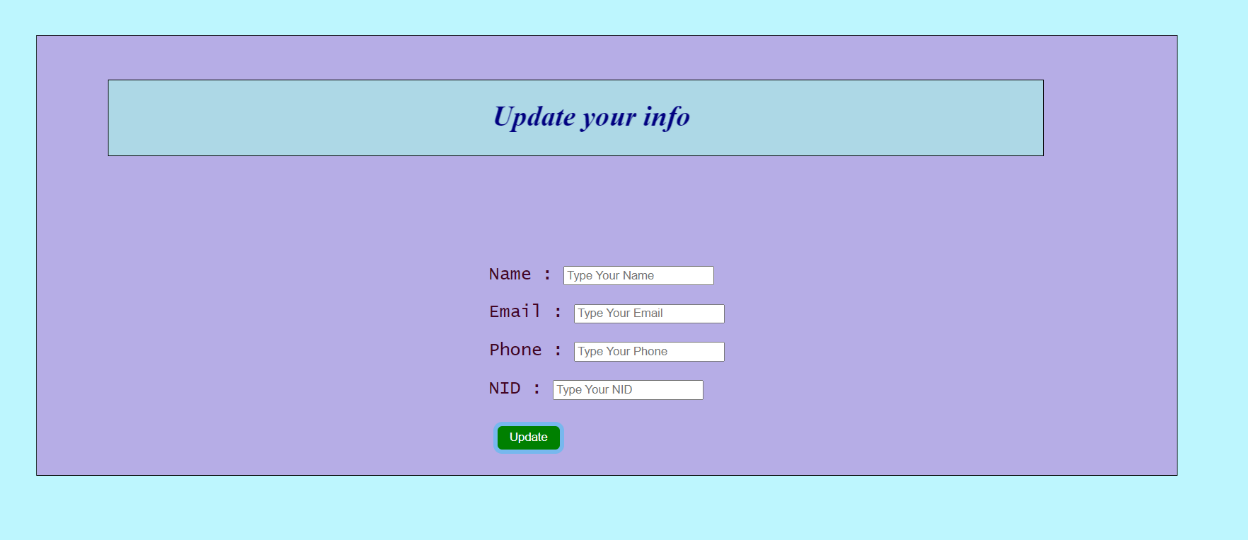
❖ Interfaces to communicate with database.





General Info.

ID	NAME	EMAIL	MOBILE	JOIN_DATE	NID	PASSWORD
2	miraz	miraz@gmail.com	173423522	22-AUG-22	1231241423	sffagas



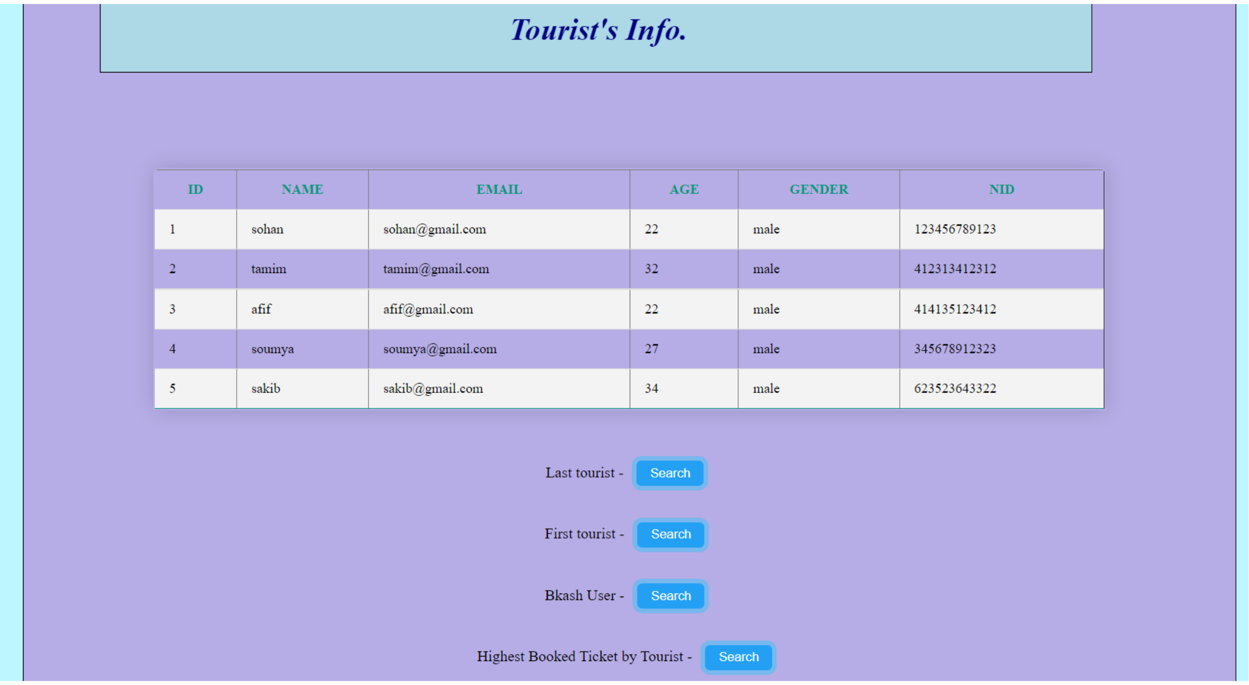
Update your info

Name : Type Your Name

Email : Type Your Email

Phone : Type Your Phone

NID : Type Your NID



Tourist's Info.

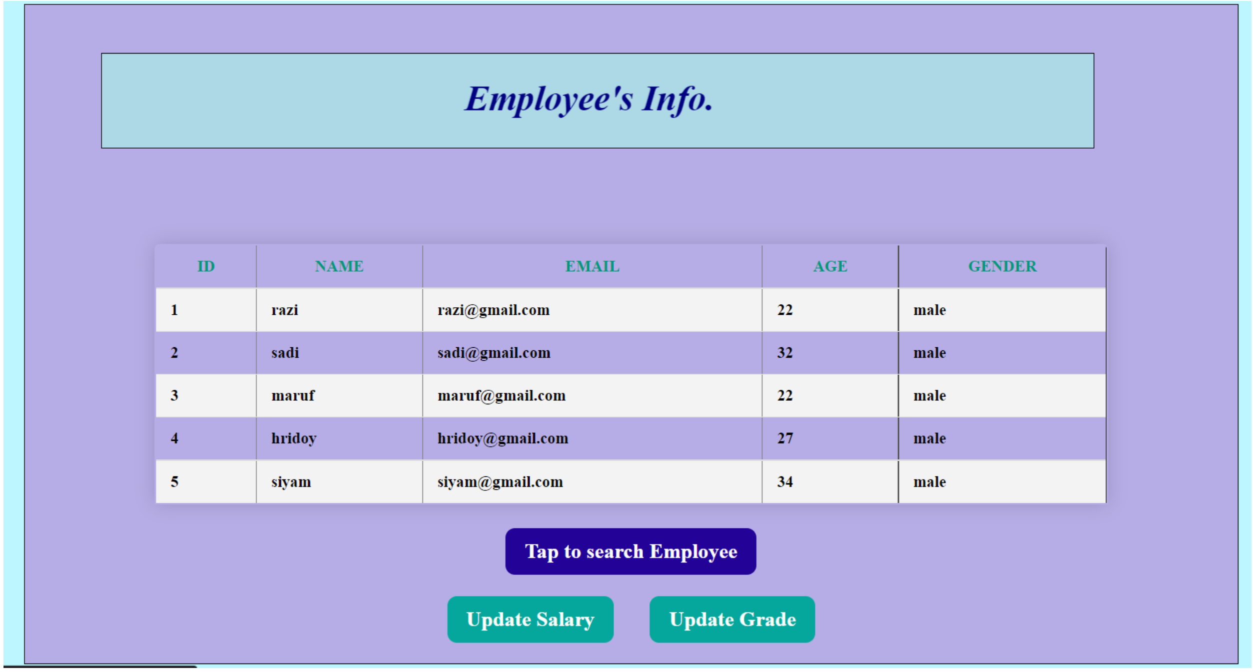
ID	NAME	EMAIL	AGE	GENDER	NID
1	sohan	sohan@gmail.com	22	male	123456789123
2	tamim	tamim@gmail.com	32	male	412313412312
3	afif	afif@gmail.com	22	male	414135123412
4	soumya	soumya@gmail.com	27	male	345678912323
5	sakib	sakib@gmail.com	34	male	623523643322

Last tourist -

First tourist -

Bkash User -

Highest Booked Ticket by Tourist -



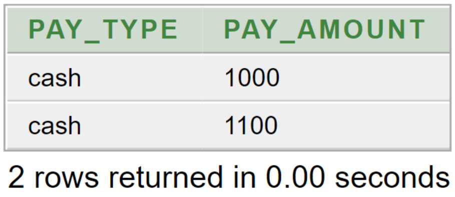
Employee's Info.

ID	NAME	EMAIL	AGE	GENDER
1	razi	razi@gmail.com	22	male
2	sadi	sadi@gmail.com	32	male
3	maruf	maruf@gmail.com	22	male
4	hridoy	hridoy@gmail.com	27	male
5	siyam	siyam@gmail.com	34	male

❖ Different type of searching and advance searching. Some of them are attached from system.

- **Display the payment method and amount considering the highest quantity of ticket.**

select pay_type, pay_amount from payment where quantity = (select max(quantity) from payment)

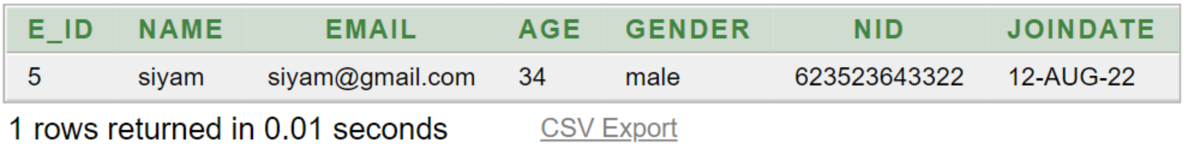


PAY_TYPE	PAY_AMOUNT
cash	1000
cash	1100

2 rows returned in 0.00 seconds

- **Display the employee details who joined after 'maruf' and age is more than 28**

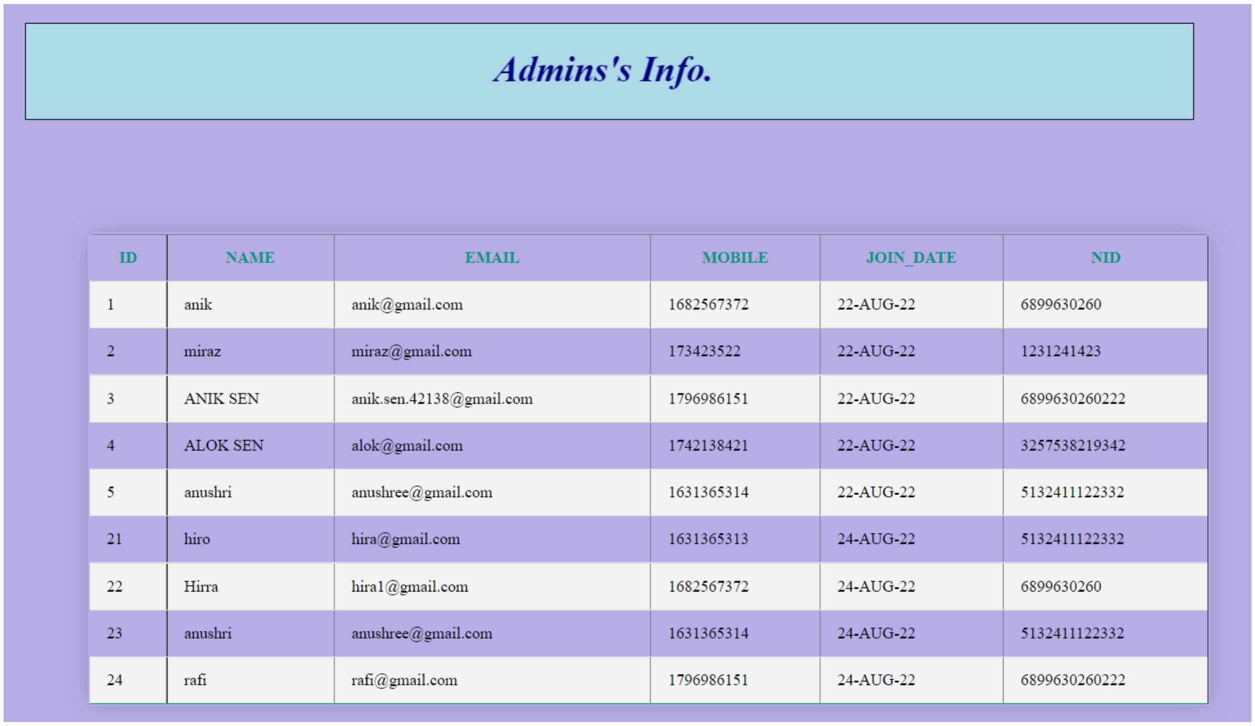
  select e.e_id, e.name, e.email, e.age, e.gender, e.nid, e.joindate from emp e,emp m where m.name='maruf' and m.e_id<e.e_id and e.age>28;



E_ID	NAME	EMAIL	AGE	GENDER	NID	JOINDATE
5	siyam	siyam@gmail.com	34	male	623523643322	12-AUG-22

1 rows returned in 0.01 seconds CSV Export

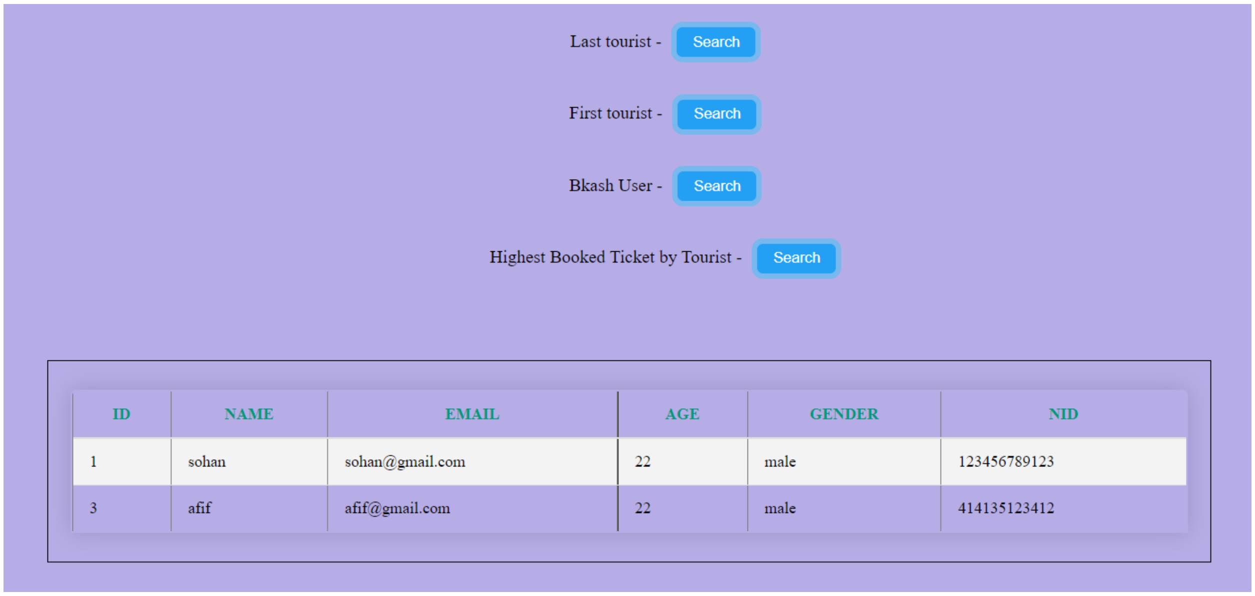
- **Display all the admins info order by admin id.**



Admins's Info.

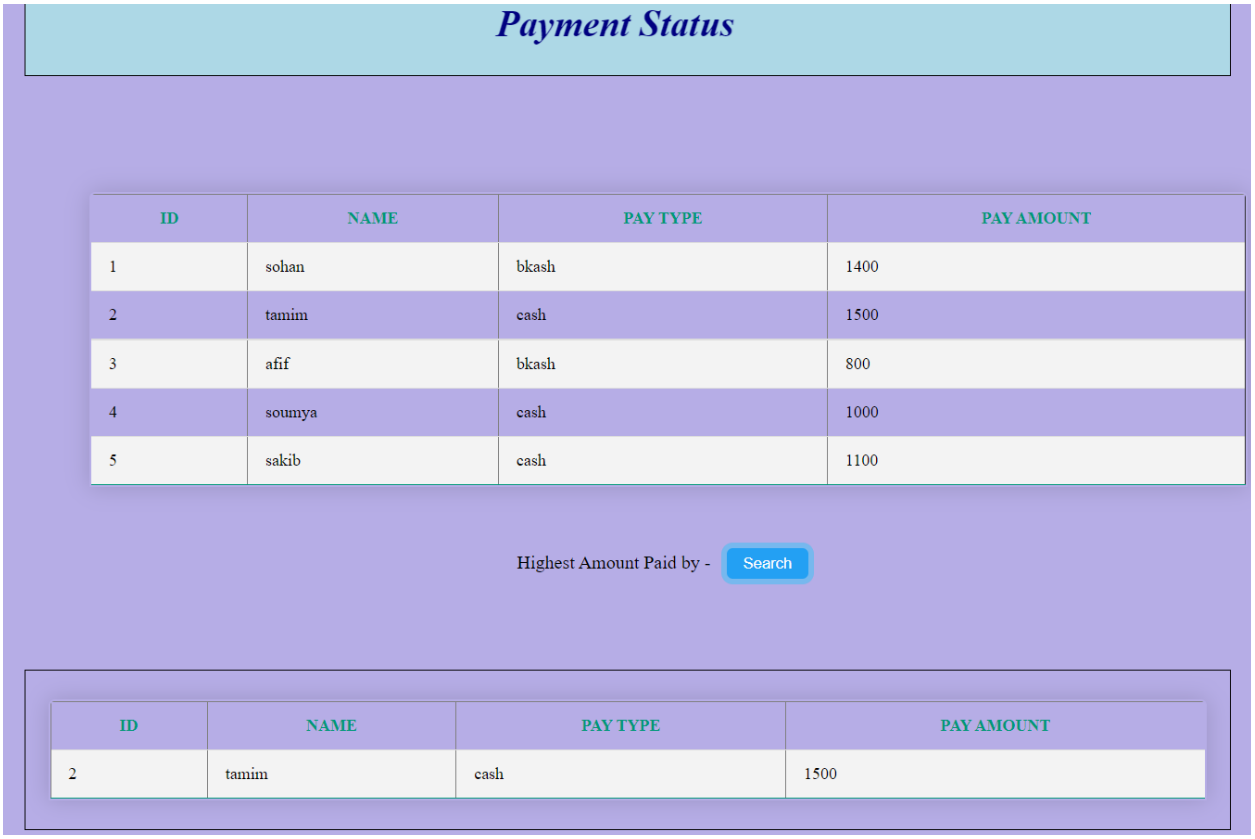
ID	NAME	EMAIL	MOBILE	JOIN_DATE	NID
1	anik	anik@gmail.com	1682567372	22-AUG-22	6899630260
2	miraz	miraz@gmail.com	173423522	22-AUG-22	1231241423
3	ANIK SEN	anik.sen.42138@gmail.com	1796986151	22-AUG-22	6899630260222
4	ALOK SEN	alok@gmail.com	1742138421	22-AUG-22	3257538219342
5	anushri	anushree@gmail.com	1631365314	22-AUG-22	5132411122332
21	hiro	hira@gmail.com	1631365313	24-AUG-22	5132411122332
22	Hirra	hira1@gmail.com	1682567372	24-AUG-22	6899630260
23	anushri	anushree@gmail.com	1631365314	24-AUG-22	5132411122332
24	rafi	rafi@gmail.com	1796986151	24-AUG-22	6899630260222

-



ID	NAME	EMAIL	AGE	GENDER	NID
1	sohan	sohan@gmail.com	22	male	123456789123
3	afif	afif@gmail.com	22	male	414135123412

-



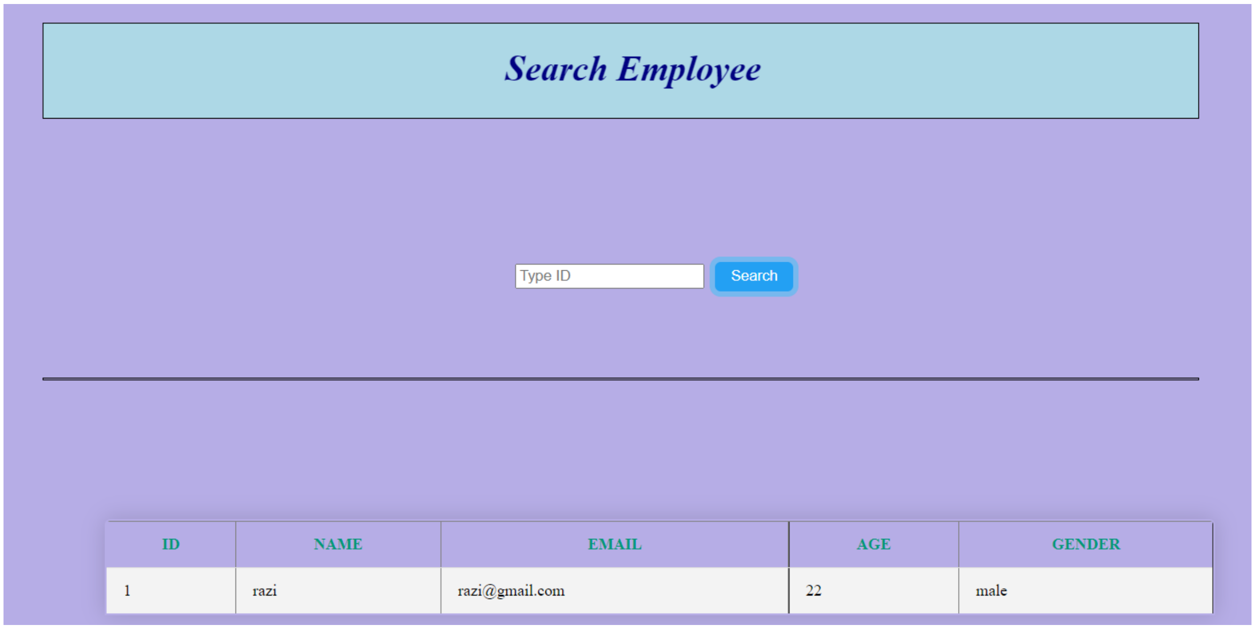
Payment Status

ID	NAME	PAY TYPE	PAY AMOUNT
1	sohan	bkash	1400
2	tamim	cash	1500
3	afif	bkash	800
4	soumya	cash	1000
5	sakib	cash	1100

Highest Amount Paid by - Search

ID	NAME	PAY TYPE	PAY AMOUNT
2	tamim	cash	1500

-

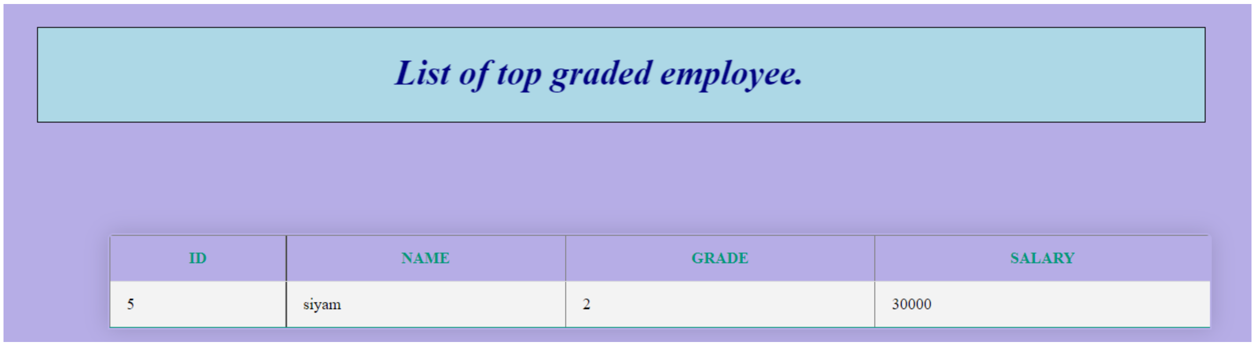


Search Employee

Type ID Search

ID	NAME	EMAIL	AGE	GENDER
1	razi	razi@gmail.com	22	male

-



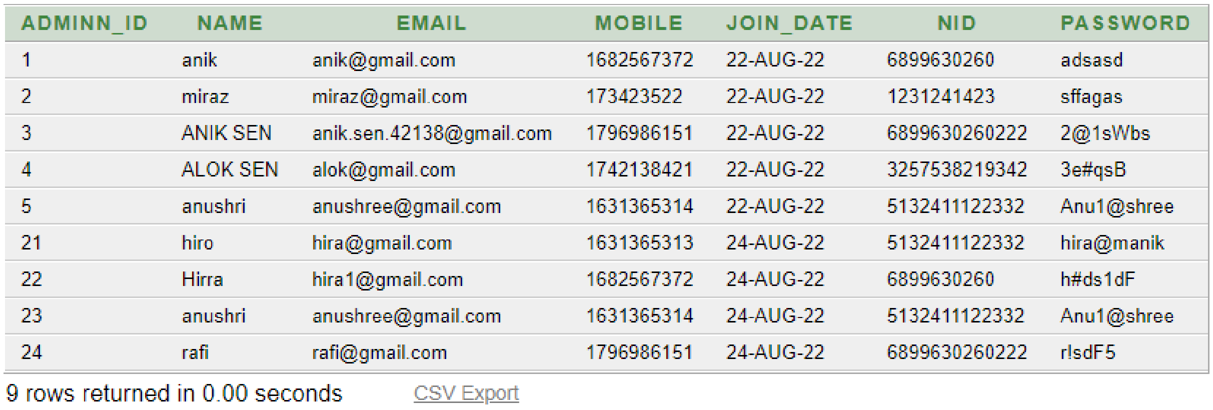
List of top graded employee.

ID	NAME	GRADE	SALARY
5	siyam	2	30000

❖ Tables using sequence to maintain primary key

Table : adminn

```
create table adminn (
 adminn_id number(30),
 name varchar(30) not null,
 email varchar(30) not null,
 mobile number(11) not null,
 join_date date not null,
 nid number(13) not null,
 password varchar(20) not null,
 primary key (adminn_id));


CREATE SEQUENCE adminn_id
 START WITH     1
 INCREMENT BY   1
```

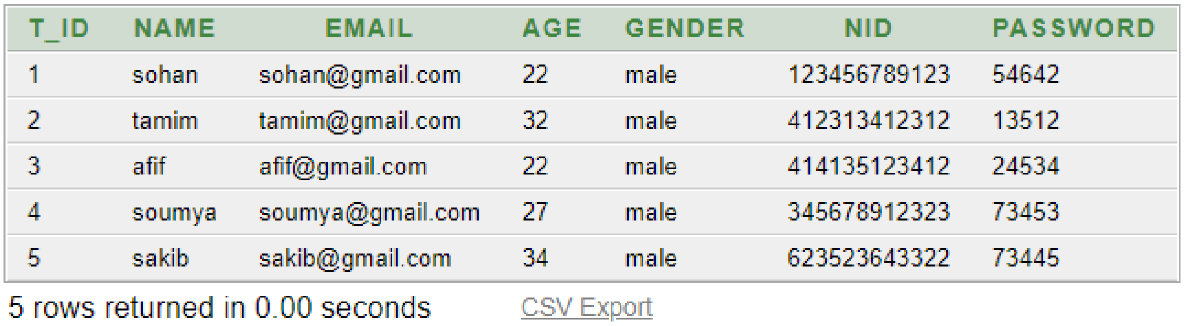


ADMINN_ID	NAME	EMAIL	MOBILE	JOIN_DATE	NID	PASSWORD
1	anik	anik@gmail.com	1682567372	22-AUG-22	6899630260	adsasd
2	miraz	miraz@gmail.com	173423522	22-AUG-22	1231241423	sffagas
3	ANIK SEN	anik.sen.42138@gmail.com	1796986151	22-AUG-22	6899630260222	2@1sWbs
4	ALOK SEN	alok@gmail.com	1742138421	22-AUG-22	3257538219342	3e#qsB
5	anushri	anushree@gmail.com	1631365314	22-AUG-22	5132411122332	Anu1@shree
21	hiro	hira@gmail.com	1631365313	24-AUG-22	5132411122332	hira@manik
22	Hirra	hira1@gmail.com	1682567372	24-AUG-22	6899630260	h#ds1dF
23	anushri	anushree@gmail.com	1631365314	24-AUG-22	5132411122332	Anu1@shree
24	rafi	rafi@gmail.com	1796986151	24-AUG-22	6899630260222	r!sdF5

9 rows returned in 0.00 seconds   CSV Export

Table : tourist_info

```
create table tourist_info(
t_id number(20) not null,
name VARCHAR(50) not null,
email varchar(20) not null,
age number(30) not null,
gender varchar(20),
nid number(20) not null,
password varchar(8) not null,
primary key (t_id));
CREATE SEQUENCE t_id
START WITH 1
INCREMENT BY 1
```

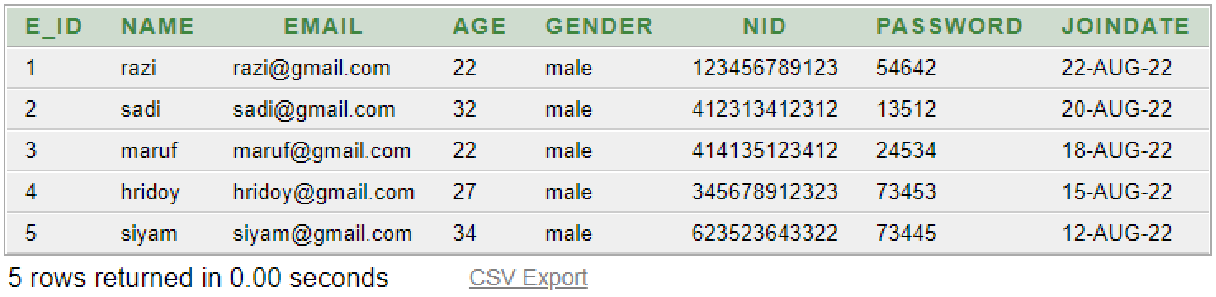


T_ID	NAME	EMAIL	AGE	GENDER	NID	PASSWORD
1	sohan	sohan@gmail.com	22	male	123456789123	54642
2	tamim	tamim@gmail.com	32	male	412313412312	13512
3	afif	afif@gmail.com	22	male	414135123412	24534
4	soumya	soumya@gmail.com	27	male	345678912323	73453
5	sakib	sakib@gmail.com	34	male	623523643322	73445

5 rows returned in 0.00 seconds CSV Export

Table : emp

```
create table emp(
e_id number(20) not null,
name VARCHAR(50) not null,
email varchar(20) not null,
age number(30) not null,
gender varchar(20),
nid number(20) not null,
password varchar(8) not null,
joindate date not null,
primary key (e_id));

CREATE SEQUENCE e_id
START WITH 1
INCREMENT BY 1
```

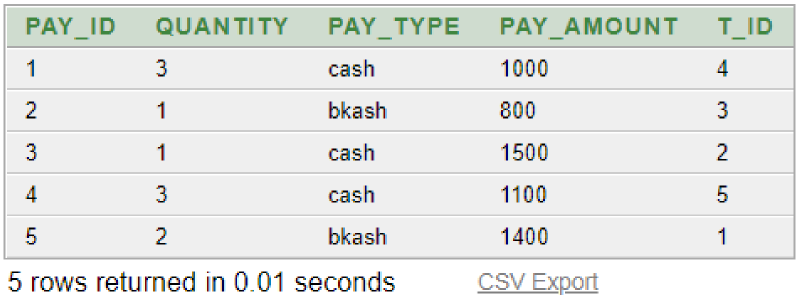


E_ID	NAME	EMAIL	AGE	GENDER	NID	PASSWORD	JOINDATE
1	razi	razi@gmail.com	22	male	123456789123	54642	22-AUG-22
2	sadi	sadi@gmail.com	32	male	412313412312	13512	20-AUG-22
3	maruf	maruf@gmail.com	22	male	414135123412	24534	18-AUG-22
4	hridoy	hridoy@gmail.com	27	male	345678912323	73453	15-AUG-22
5	siyam	siyam@gmail.com	34	male	623523643322	73445	12-AUG-22

5 rows returned in 0.00 seconds CSV Export

Table : payment

```
create table payment(
pay_id number(20) not null,
quantity number(20) not null,
pay_type varchar(20) not null,
pay_amount number(20) not null,
t_id number(20) not null,
primary key (pay_id),
FOREIGN KEY (t_id) REFERENCES tourist(t_id)
);

CREATE SEQUENCE pay_id
START WITH 1
INCREMENT BY 1
```

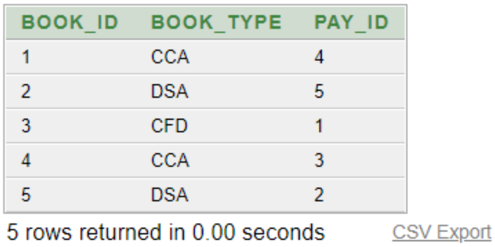


PAY_ID	QUANTITY	PAY_TYPE	PAY_AMOUNT	T_ID
1	3	cash	1000	4
2	1	bkash	800	3
3	1	cash	1500	2
4	3	cash	1100	5
5	2	bkash	1400	1

5 rows returned in 0.01 seconds CSV Export

Table : booking

```
create table booking(
book_id number(20) not null,
book_type VARCHAR(20) not null,
pay_id number(20) not null,
primary key (book_id),
FOREIGN KEY (pay_id) REFERENCES payment(pay_id)
);

CREATE SEQUENCE book_id
START WITH 1
INCREMENT BY 1
```



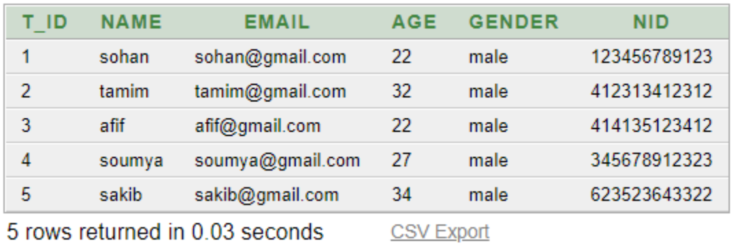
BOOK_ID	BOOK_TYPE	PAY_ID
1	CCA	4
2	DSA	5
3	CFD	1
4	CCA	3
5	DSA	2

5 rows returned in 0.00 seconds CSV Export

❖ View

- CREATE OR REPLACE VIEW emp_detail_view AS

```
SELECT e_id, NAME, email, age, gender
FROM emp
```

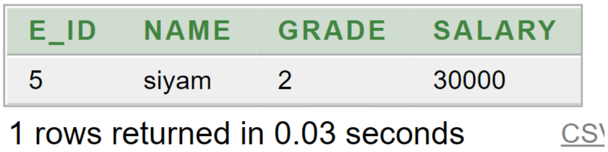


T_ID	NAME	EMAIL	AGE	GENDER	NID
1	sohan	sohan@gmail.com	22	male	123456789123
2	tamim	tamim@gmail.com	32	male	412313412312
3	afif	afif@gmail.com	22	male	414135123412
4	soumya	soumya@gmail.com	27	male	345678912323
5	sakib	sakib@gmail.com	34	male	623523643322

5 rows returned in 0.03 seconds CSV Export

- CREATE or replace VIEW Hpaid_emp_view AS

```
select  e.e_id, e.name, d.grade, g.salary
from sal_detail d, salgrade g, emp e
where d.e_id = e.e_id and d.grade = g.grade and g.grade<=2
```

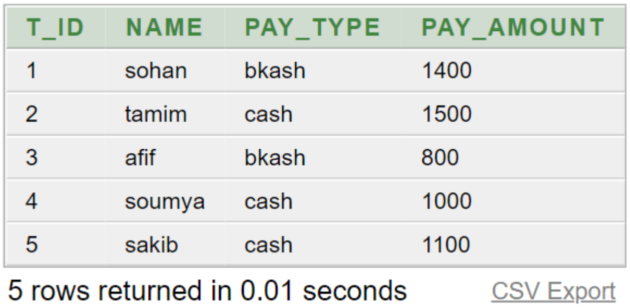


E_ID	NAME	GRADE	SALARY
5	siyam	2	30000

1 rows returned in 0.03 seconds CS

- CREATE or Replace VIEW payment_by_user AS

```
SELECT t.t_id, t.NAME, p.pay_type, p.pay_amount
FROM tourist_info t, payment p where t.t_id = p.t_id order by t.t_id
```

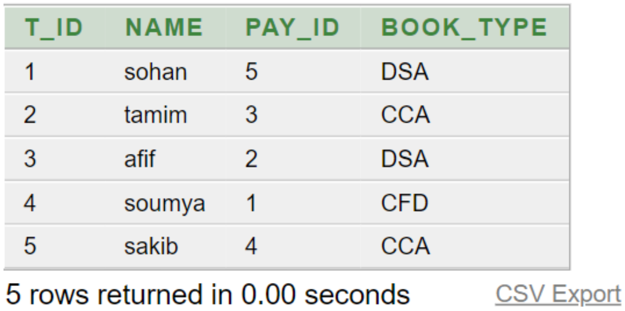


T_ID	NAME	PAY_TYPE	PAY_AMOUNT
1	sohan	bkash	1400
2	tamim	cash	1500
3	afif	bkash	800
4	soumya	cash	1000
5	sakib	cash	1100

5 rows returned in 0.01 seconds CSV Export

- CREATE or Replace VIEW booked_by_user AS

```
SELECT t.t_id, t.name, p.pay_id, b.book_type
FROM tourist_info t, payment p, booking b where t.t_id = p.t_id and p.pay_id = b.pay_id order by t.t_id
```

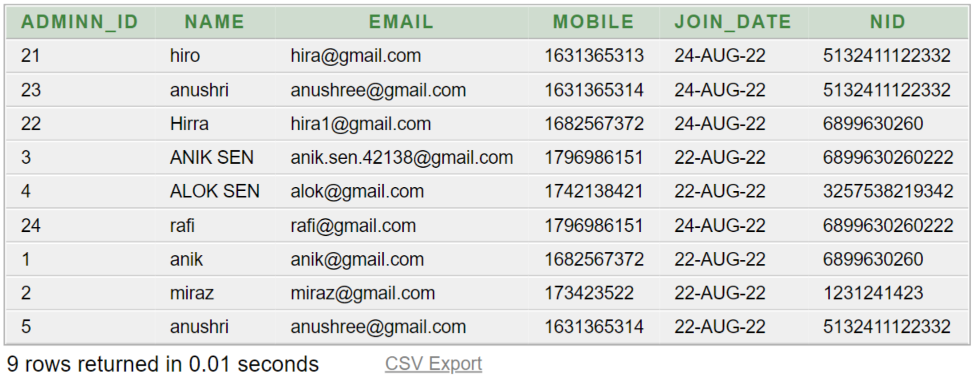


T_ID	NAME	PAY_ID	BOOK_TYPE
1	sohan	5	DSA
2	tamim	3	CCA
3	afif	2	DSA
4	soumya	1	CFD
5	sakib	4	CCA

5 rows returned in 0.00 seconds

- CREATE or replace VIEW admin_detail_view AS

SELECT adminn_id, NAME, email, mobile, join_date, nid
FROM adminn



ADMINN_ID	NAME	EMAIL	MOBILE	JOIN_DATE	NID
21	hiro	hira@gmail.com	1631365313	24-AUG-22	5132411122332
23	anushri	anushree@gmail.com	1631365314	24-AUG-22	5132411122332
22	Hirra	hira1@gmail.com	1682567372	24-AUG-22	6899630260
3	ANIK SEN	anik.sen.42138@gmail.com	1796986151	22-AUG-22	6899630260222
4	ALOK SEN	alok@gmail.com	1742138421	22-AUG-22	3257538219342
24	rafi	rafi@gmail.com	1796986151	24-AUG-22	6899630260222
1	anik	anik@gmail.com	1682567372	22-AUG-22	6899630260
2	miraz	miraz@gmail.com	173423522	22-AUG-22	1231241423
5	anushri	anushree@gmail.com	1631365314	22-AUG-22	5132411122332

9 rows returned in 0.01 seconds

❖ Procedures and Functions (Using Package and Exception handling)

- This package is used for admin’s registration and update admin’s info. There are two procedures under this package. We use some exception handling approach here

create or replace package pack_admin_add_update
as

```
procedure proc_add_admin(aname adminn.name%type, mail adminn.email%type, phone
adminn.mobile%type, nidd adminn.nid%type, pass adminn.password%type);
procedure proc_update_admin(aname adminn.name%type, mail adminn.email%type, phone
adminn.mobile%type, nidd adminn.nid%type, pass adminn.password%type);

end pack_admin_add_update;

create or replace package body pack_admin_add_update
is
        procedure proc_add_admin(aname adminn.name%type, mail adminn.email%type, phone
adminn.mobile%type, nidd adminn.nid%type, pass adminn.password%type)
         is
           begin
           insert into adminn
(ADMINN_ID,NAME,EMAIL,MOBILE,JOIN_DATE,NID,PASSWORD) values
(adminn_id.nextval, aname, mail, phone, sysdate, nidd, pass);
         end proc_add_admin;
        procedure proc_update_admin(aname adminn.name%type, mail adminn.email%type,
phone adminn.mobile%type, nidd adminn.nid%type, pass adminn.password%type)
         is
           begin
           update adminn set name = aname, email = mail, mobile =phone, nid=nidd where
password = pass;
         end proc_update_admin;
end pack_admin_add_update;
```

- This package is used for employee’s grade update, delete and insert purpose. There are three procedures and two functions under this package. We use some exception handling approach here

```
create or replace package emp_grade
as
procedure emp_grade_up(id emp.e_id%type, grd sal_detail.grade%type);
procedure emp_grade_delete(id emp.e_id%type);
procedure emp_grade_insert(id emp.e_id%type, grd sal_detail.grade%type);


end emp_grade;

create or replace package body emp_grade
is
function VALID_EMPNO(id emp.e_id%type)
return boolean
is
chk emp.e_id%type;
begin
```

```
select e_id into chk from emp where id = e_id;
return true;
exception
when no_data_found then
   return false;
end VALID_EMPNO;

function VALID_GRADE(grd salgrade.grade%type)
return boolean
is
chk salgrade.grade%type;
begin
select grade into chk from salgrade where grade = grd;
return true;
exception
when no_data_found then
   return false;
end VALID_GRADE;


         procedure emp_grade_up(id emp.e_id%type, grd sal_detail.grade%type)
         is
         begin
                  if (VALID_EMPNO(id)) then
                            if (VALID_GRADE(grd)) then
                               update sal_detail set grade=grd where e_id=id;
                               dbms_output.put_line('Grade Updated');

                            else
                               dbms_output.put_line('Invalid Grade');
                           end if;
                  else
                     dbms_output.put_line('Employee ID does not exist');
                  end if;

         end emp_grade_up;
         procedure emp_grade_delete(id emp.e_id%type)
         is
         begin
                  if (VALID_EMPNO(id)) then
                     delete from sal_detail where e_id=id;
                  else
                     dbms_output.put_line('Employee ID does not exist');
                  end if;
         end emp_grade_delete;
         procedure emp_grade_insert(id emp.e_id%type, grd sal_detail.grade%type)
```

```
        is
        begin
                if (VALID_GRADE(grd)) then
                   insert into sal_detail values(id, grd);
                else
                   dbms_output.put_line('Invalid Grade');
                end if;
        end emp_grade_insert;

end emp_grade;
```

- This package is used for employee’s salary update purpose. There is a procedure and a function under this package. We use some exception handling approach here

```
       create package pack_sal_update

as
procedure proc_sal_update(grad salgrade.grade%type, sal salgrade.salary%type);

end pack_sal_update;

create or replace package body pack_sal_update
is

function VALID_GRADE(grad salgrade.grade%type)
return boolean
is
chk salgrade.grade%type;
begin
select grade into chk from salgrade where grade = grad;
return true;
exception
when no_data_found then
   return false;
end VALID_GRADE;

        procedure proc_sal_update(grad salgrade.grade%type, sal salgrade.salary%type)
        is
        begin
                if (VALID_GRADE(grad)) then
                   update salgrade set salary = sal where grade = grad;
             dbms_output.put_line('Salary Updated.');
                else
                   dbms_output.put_line('This Grade does not exist');
                end if;
        end proc_sal_update;

end pack_sal_update;
```

- This package is used for search some tourist information. There is a procedure named proc_bkash_user . We use some exception handling approach here

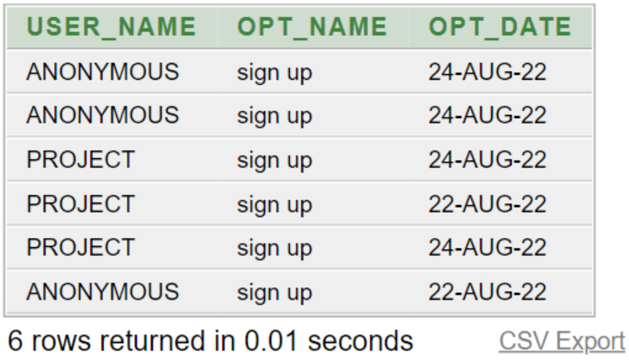
```
create package pack_under_tourist
as
procedure proc_bkash_user(id OUT tourist_info.t_id%type, tname OUT
tourist_info.name%type, temail OUT tourist_info.email%type, tage OUT tourist_info.age%type,
tgender OUT tourist_info.gender%type, tnid OUT tourist_info.nid%type);

end pack_under_tourist;

create or replace package body pack_under_tourist
is

        procedure proc_bkash_user(id OUT tourist_info.t_id%type, tname OUT
tourist_info.name%type, temail OUT tourist_info.email%type, tage OUT tourist_info.age%type,
tgender OUT tourist_info.gender%type, tnid OUT tourist_info.nid%type)
       is
        cursor c1
          is
           select t_id, name, email, age, gender, nid from tourist_info where t_id in (select t_id
from payment where pay_type = 'bkash');
          i number(25);
            begin
             for i in c1 loop
             id := i.t_id;
             tname := i.name;
             temail := i.email;
             tage := i.age;
             tgender := i.gender;
             tnid := i.nid;
          dbms_output.put_line(id||'   '|| tname ||'   '|| temail ||'   '|| tage ||'   '|| tgender||'   '|| tnid);
            end loop;

       end proc_bkash_user;

end pack_under_tourist;

declare
 id tourist_info.t_id%type;
 tname tourist_info.name%type;
 temail tourist_info.email%type;
 tage tourist_info.age%type;
 tgender tourist_info.gender%type;
 tnid tourist_info.nid%type;
begin
pack_under_tourist.proc_bkash_user(id, tname , temail , tage , tgender, tnid);
end;
```

❖ Triggers

- This trigger is used to keep the log of admin registration.

➢ create table admin_reg_log (user_name varchar(20), opt_name varchar(20), opt_date date);

```
create or replace trigger admin_reg_log
after insert on adminn
declare
opt varchar(30);
begin
        opt := 'sign up';
insert into admin_reg_log values(user,opt,sysdate);
end;
```



USER_NAME	OPT_NAME	OPT_DATE
ANONYMOUS	sign up	24-AUG-22
ANONYMOUS	sign up	24-AUG-22
PROJECT	sign up	24-AUG-22
PROJECT	sign up	22-AUG-22
PROJECT	sign up	24-AUG-22
ANONYMOUS	sign up	22-AUG-22

6 rows returned in 0.01 seconds   CSV Export

- This trigger is used to keep secure the employee registration process.

- create or replace trigger emp_reg_secure

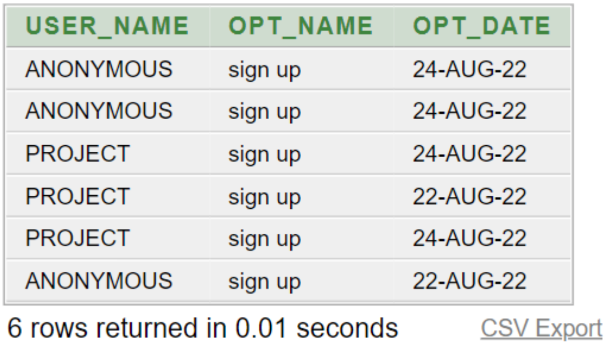
```
 before insert on emp
declare
```

```
begin
if to_char(sysdate,'HH24') not between '8' and '17' or to_char(sysdate,'DAY') in('FRIDAY',
'SATURDAY') then
raise_application_error(-20202, 'Please come in regular working working hour');
END IF;
End;
```

- This trigger is used to keep the log of employee registration.

- create table emp_reg_log (user_name varchar(20), opt_name varchar(20), opt_date date);

```
create or replace trigger emp_reg_log
after insert on emp
declare
opt varchar(30);
begin
        opt := 'sign up';
insert into emp_reg_log values(user,opt,sysdate);
end;
```



USER_NAME	OPT_NAME	OPT_DATE
ANONYMOUS	sign up	24-AUG-22
ANONYMOUS	sign up	24-AUG-22
PROJECT	sign up	24-AUG-22
PROJECT	sign up	22-AUG-22
PROJECT	sign up	24-AUG-22
ANONYMOUS	sign up	22-AUG-22

6 rows returned in 0.01 seconds CSV Export

- This trigger is used to keep the log of employee’s grade update.

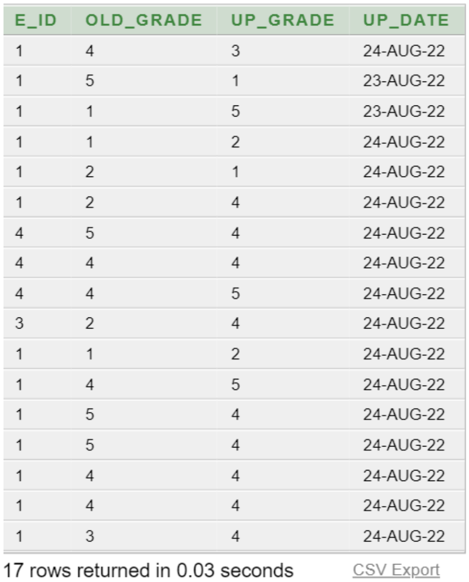
- create table emp_grade_uplog (e_id number(20), old_grade number(20), up_grade number(20), up_date date);

```
create or replace trigger emp_grade_up
after update of grade on sal_detail
for each row
begin
insert into emp_grade_uplog values (:old.e_id, :old.grade, :new.grade, sysdate);
end;
```

- create or replace trigger sal_alart

```
 before update on salgrade
 for each row
```

```
begin
if :new.salary < :old.salary then
raise_application_error(-20202, 'Warning: Updated salary must be greater then previous salary.');
END IF;
End;
```



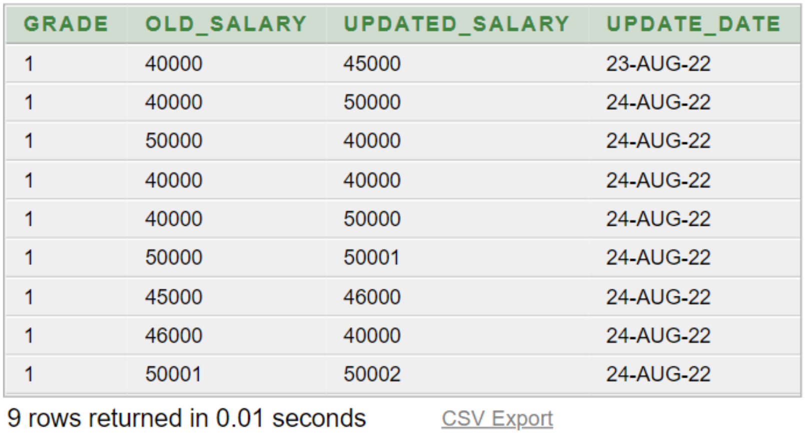
E_ID	OLD_GRADE	UP_GRADE	UP_DATE
1	4	3	24-AUG-22
1	5	1	23-AUG-22
1	1	5	23-AUG-22
1	1	2	24-AUG-22
1	2	1	24-AUG-22
1	2	4	24-AUG-22
4	5	4	24-AUG-22
4	4	4	24-AUG-22
4	4	5	24-AUG-22
3	2	4	24-AUG-22
1	1	2	24-AUG-22
1	4	5	24-AUG-22
1	5	4	24-AUG-22
1	5	4	24-AUG-22
1	4	4	24-AUG-22
1	4	4	24-AUG-22
1	3	4	24-AUG-22

17 rows returned in 0.03 seconds CSV Export

This trigger is used to keep the log of employee’s salary update.

- create table emp_sal_uplog (grade number(20), old_salary number(20), updated_salary number(20), update_date date);

```
create or replace trigger emp_sal_up
after update of SALARY on salgrade
for each row
begin
insert into emp_sal_uplog values (:old.grade, :old.salary, :new.salary, sysdate);
end;
```



GRADE	OLD_SALARY	UPDATED_SALARY	UPDATE_DATE
1	40000	45000	23-AUG-22
1	40000	50000	24-AUG-22
1	50000	40000	24-AUG-22
1	40000	40000	24-AUG-22
1	40000	50000	24-AUG-22
1	50000	50001	24-AUG-22
1	45000	46000	24-AUG-22
1	46000	40000	24-AUG-22
1	50001	50002	24-AUG-22

9 rows returned in 0.01 seconds CSV Export