**Job Interview Question Portal**

Subject : Advanced Database Management System

Name : Sanika Sambhaji Bhagat

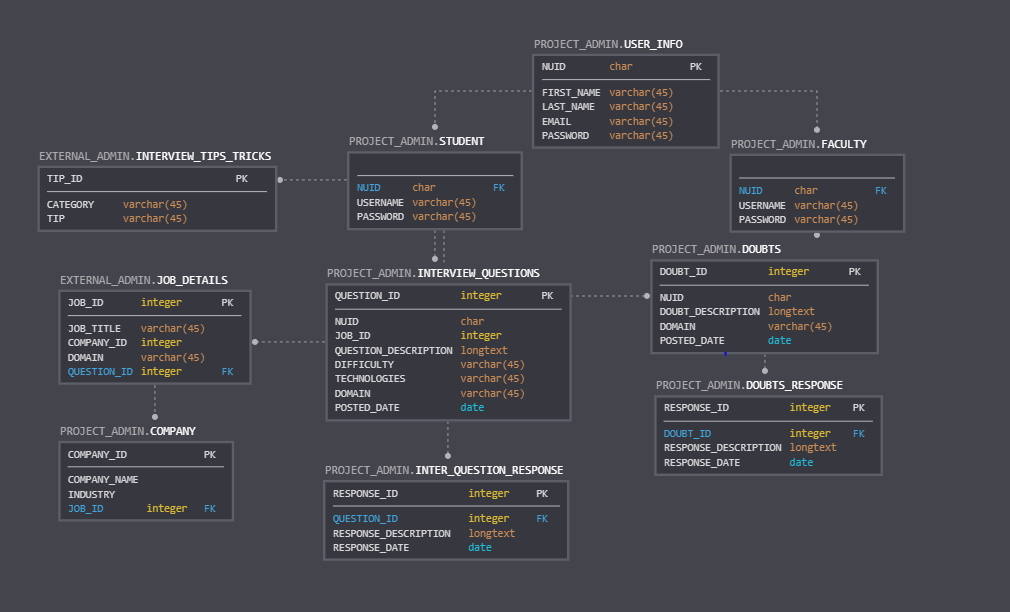
**Purpose**

All the students looking for a co-op need help to prepare during their interviews. They need some sample interview questions asked in the company, or maybe some sample questions about the position that they are interviewing to get an idea about how the interview will be. Also, some students may be having doubts while studying for their interviews which they might want to ask their professors. The interview Question portal is an application where the students and the faculty can register to so that they can help each other. The students can look at the interview questions posted by other students or even ask doubts to the professor which are registered on the portal.

* The interview question portal will help the students to get all the necessary help that they need for their interviews.
* This portal can be used by the students:

1. To post some interview questions that are worthy to be shared with other students
2. View some interview questions to prepare well for the interview
3. Post doubts which can be answered by the subject faculty
4. View some last minute interview tips and tricks

**ER Diagram**



**Use Cases**

* Post Interview Questions
* Post Doubts
* View Interview Questions
* View Interview Questions posted by user
* View Doubts posted by User
* View Responses to Interview Questions
* View Responses to Doubts
* Post Response to Interview Questions
* View Interview Tips and Tricks
* Post Responses to Doubts
* View Doubts

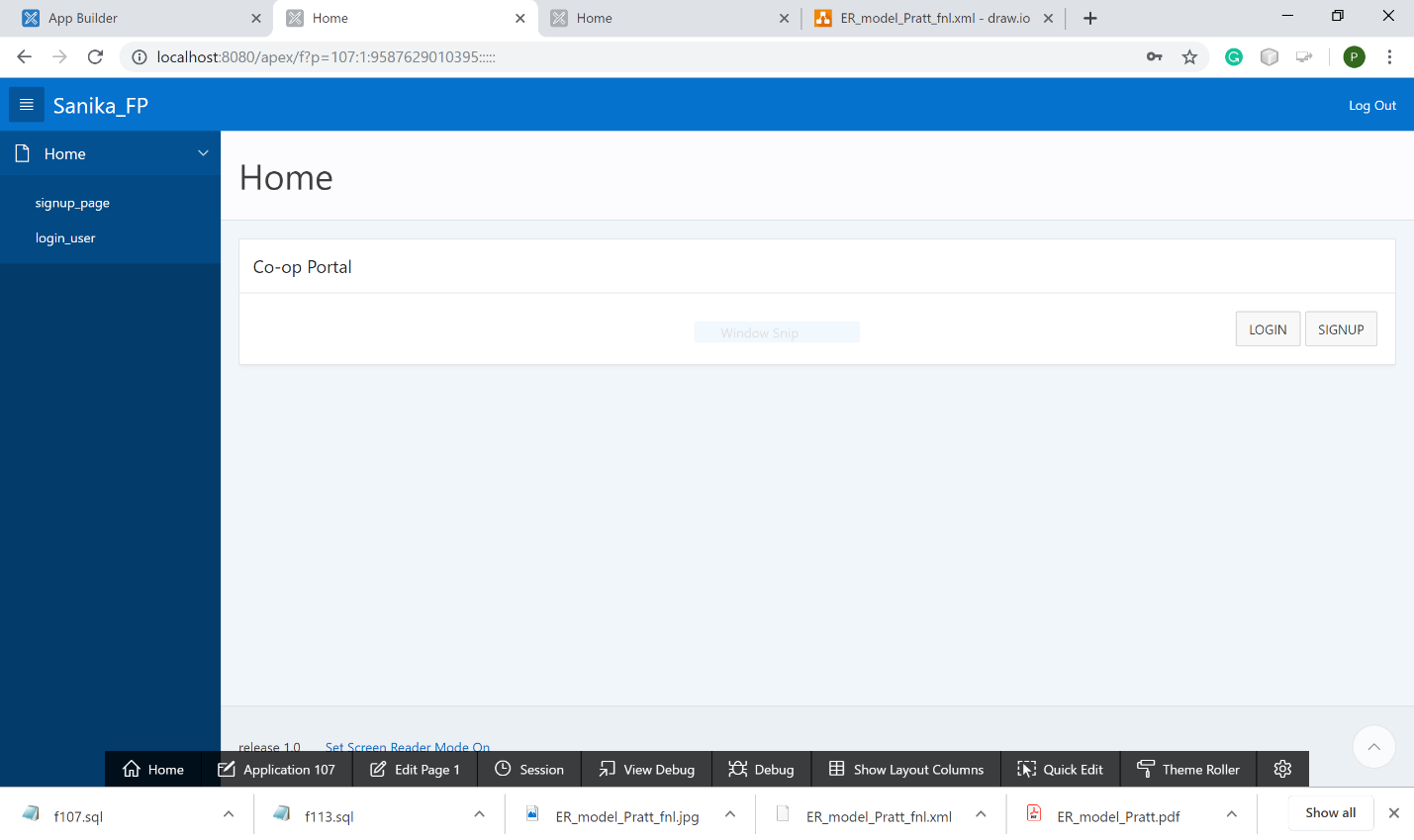
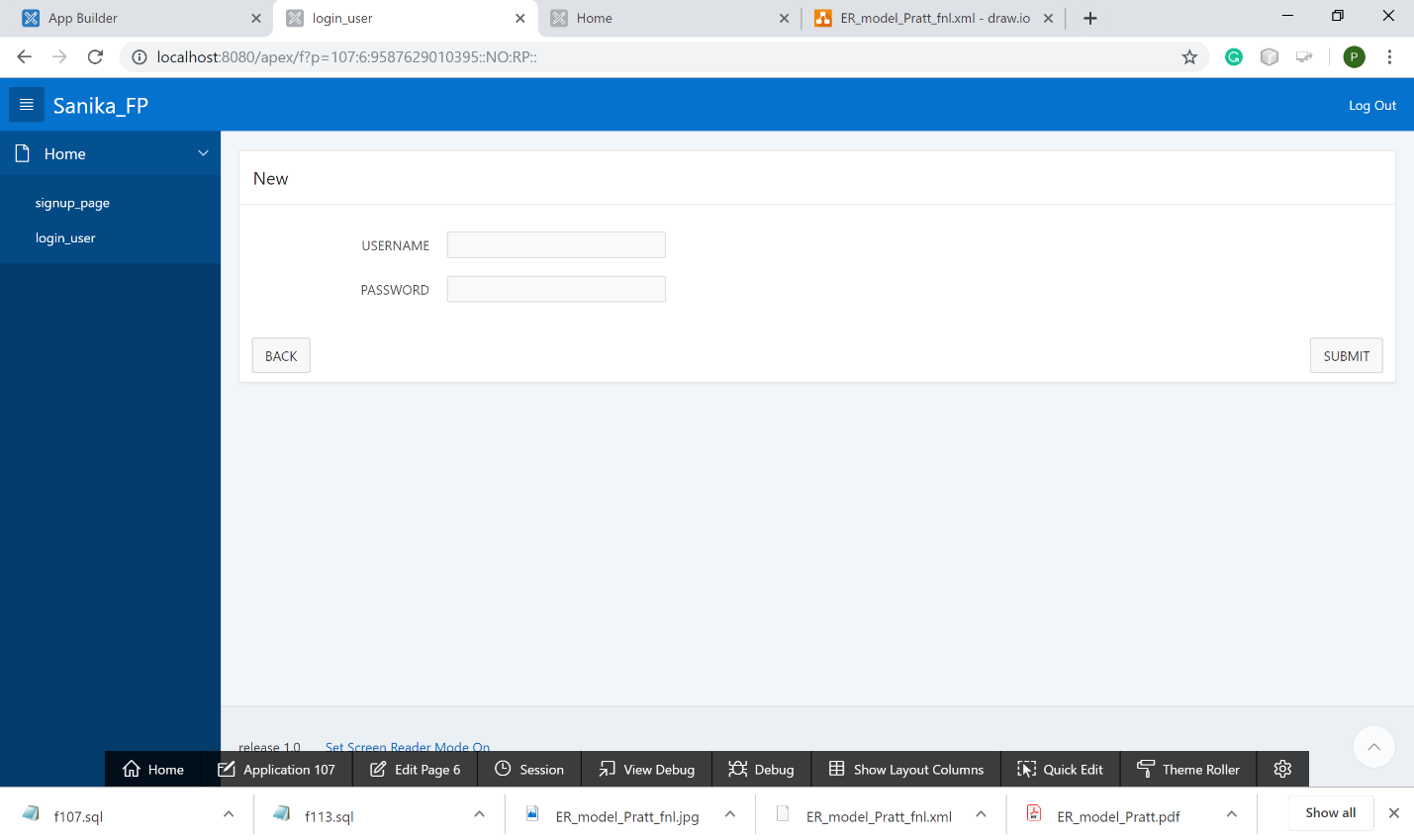
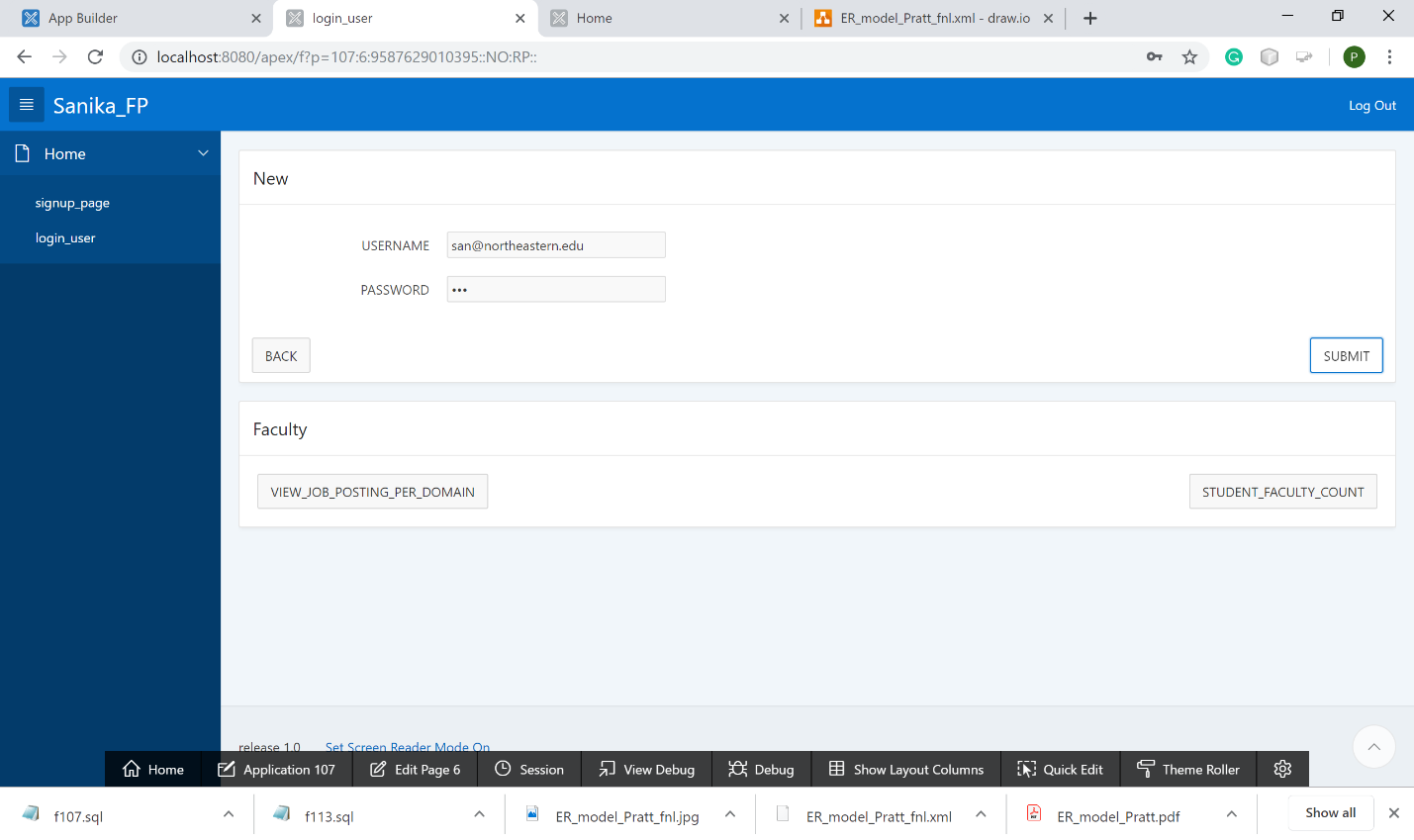
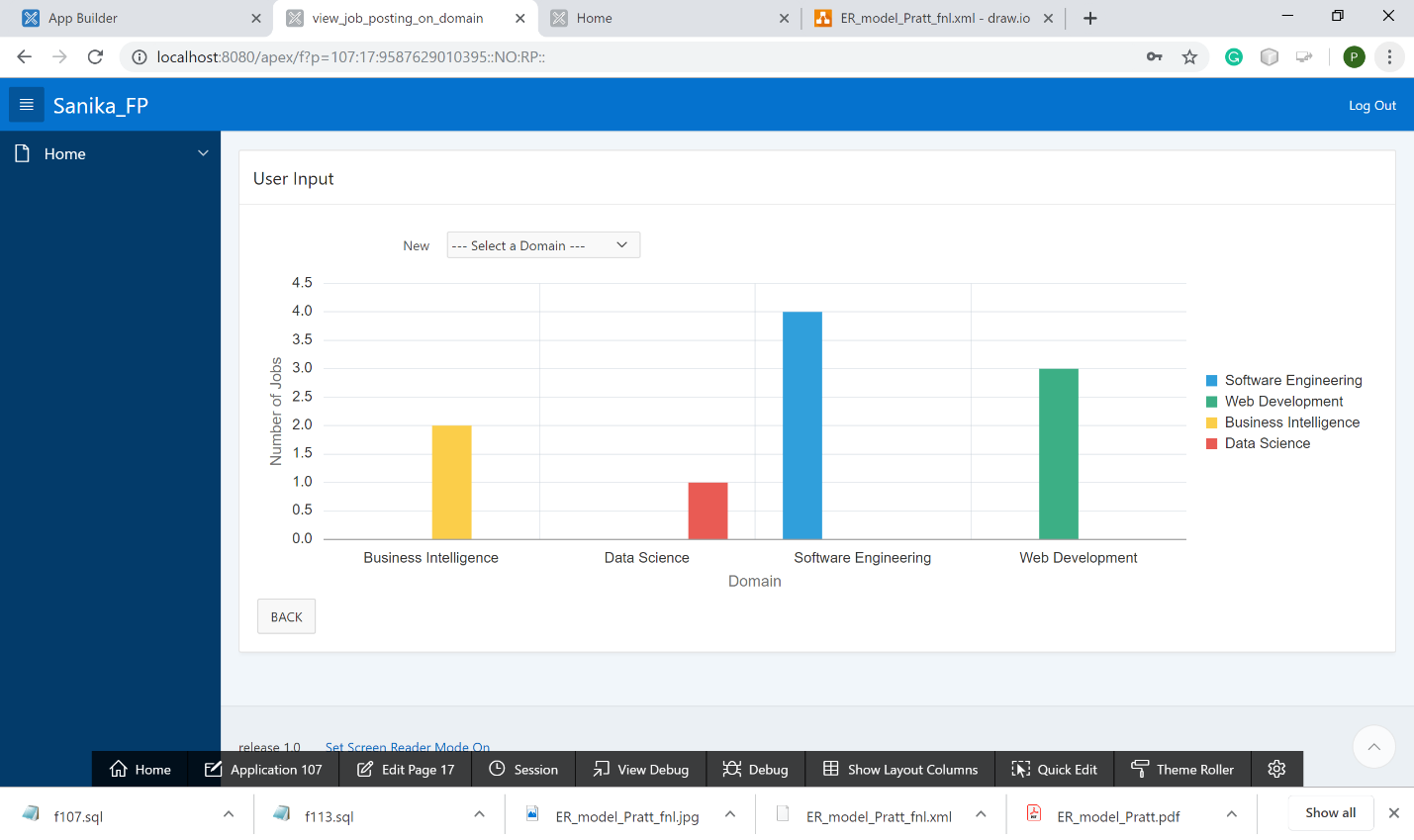
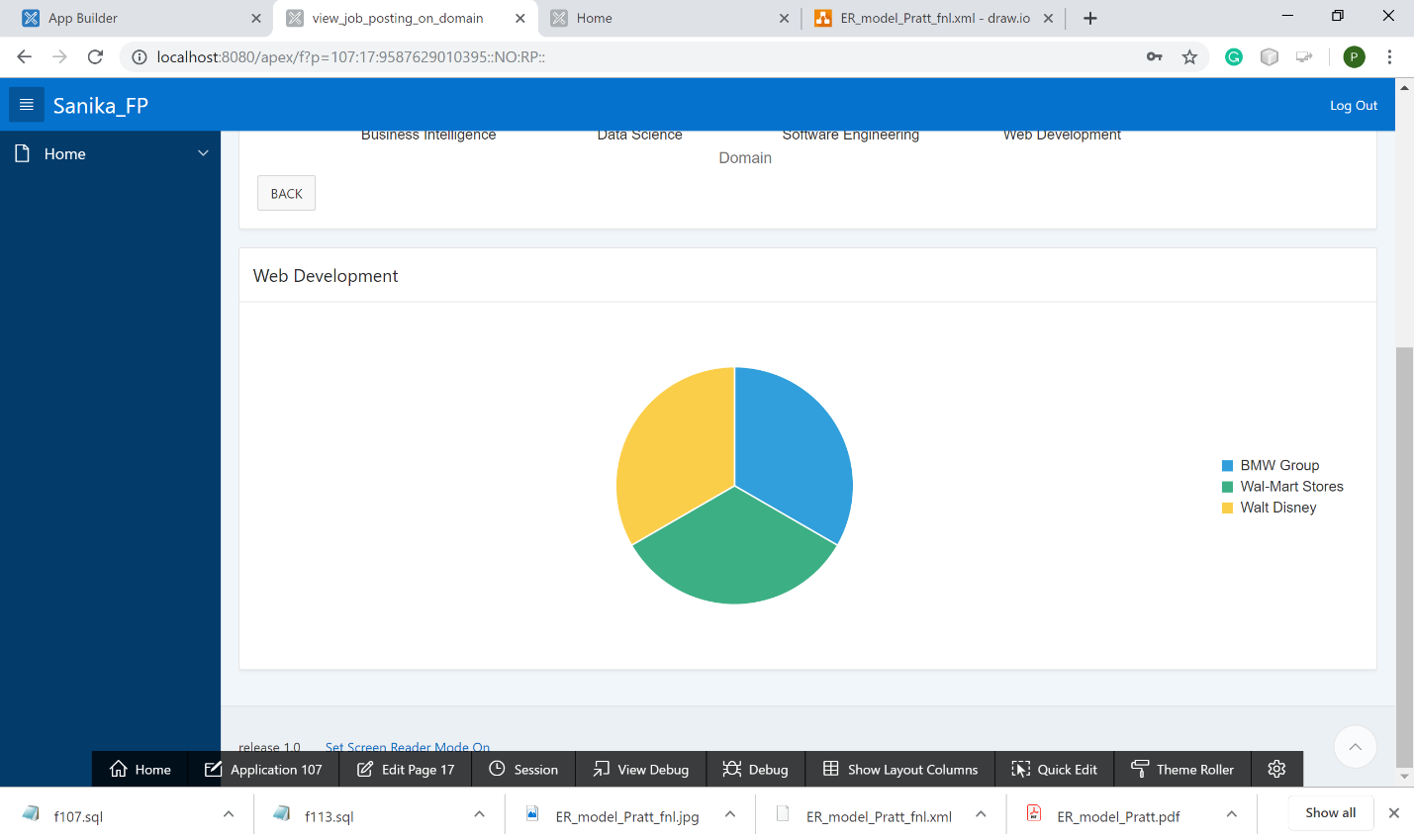
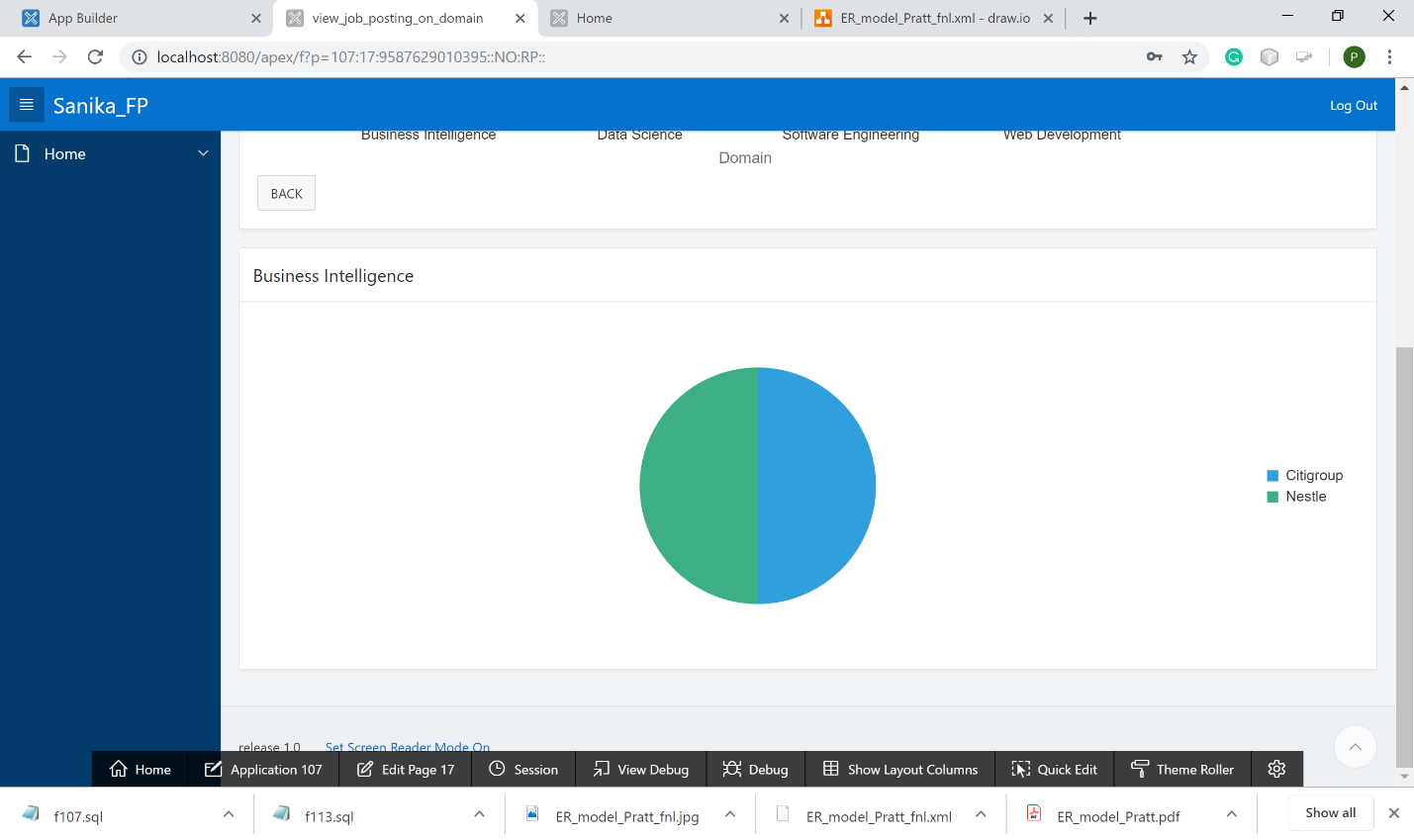
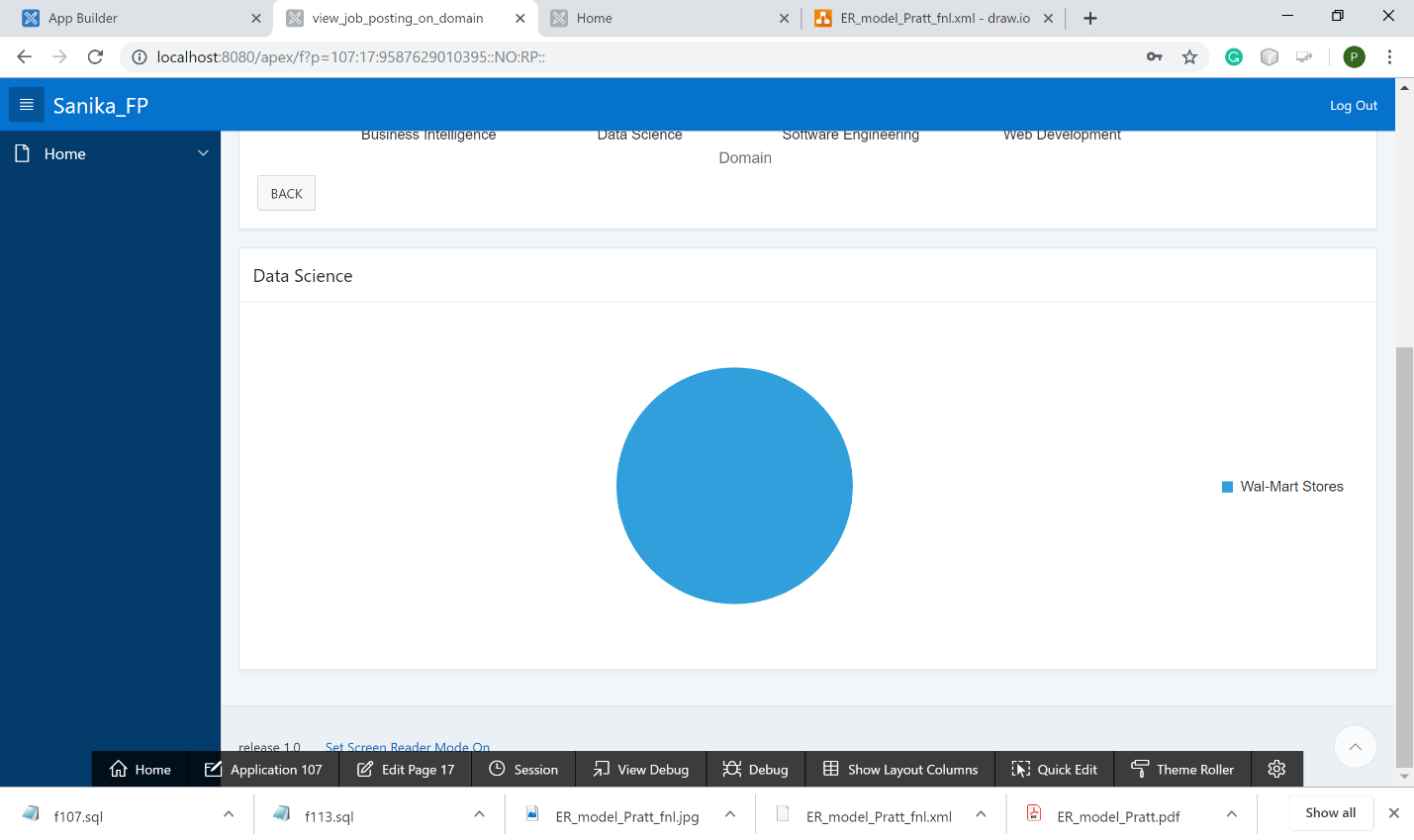
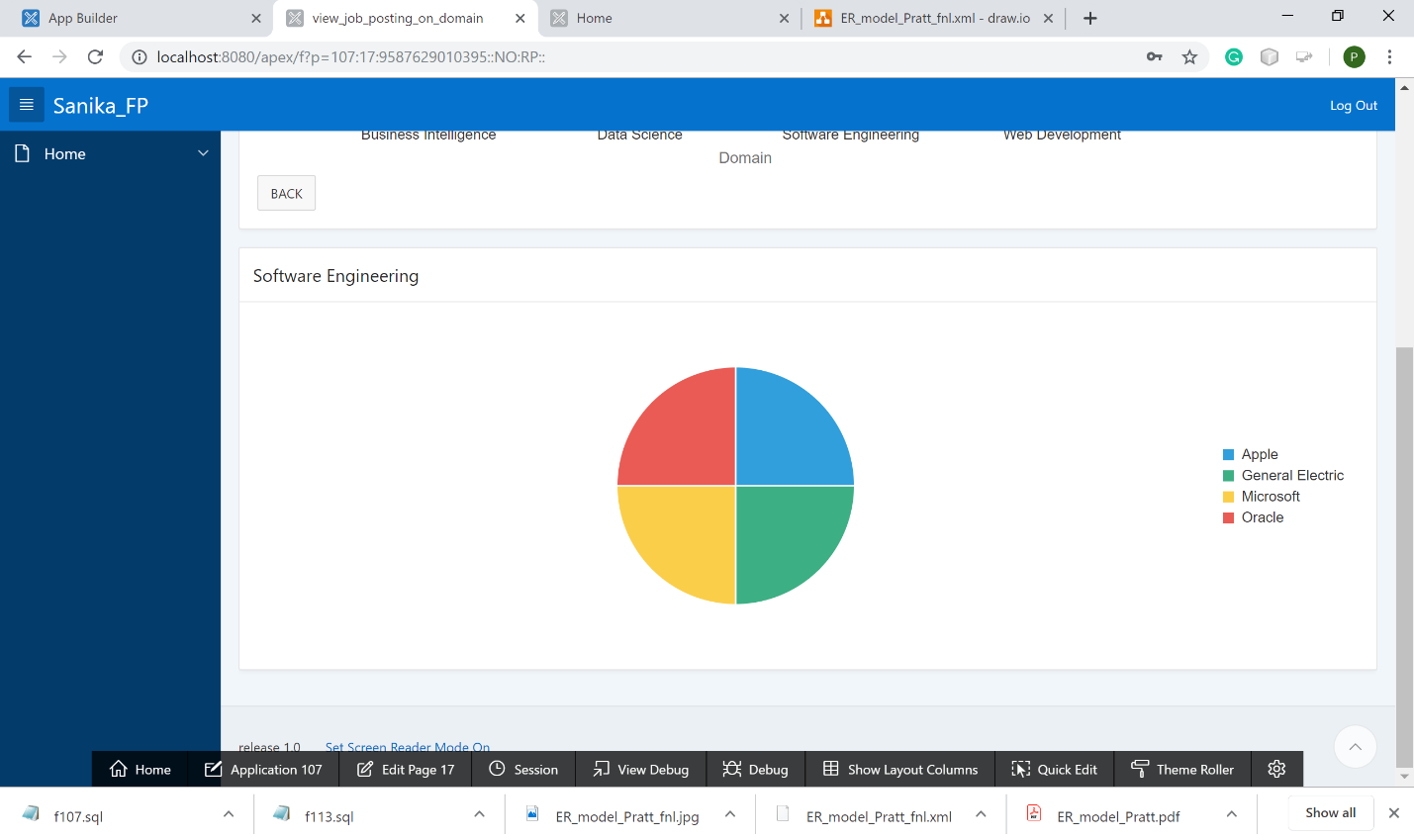
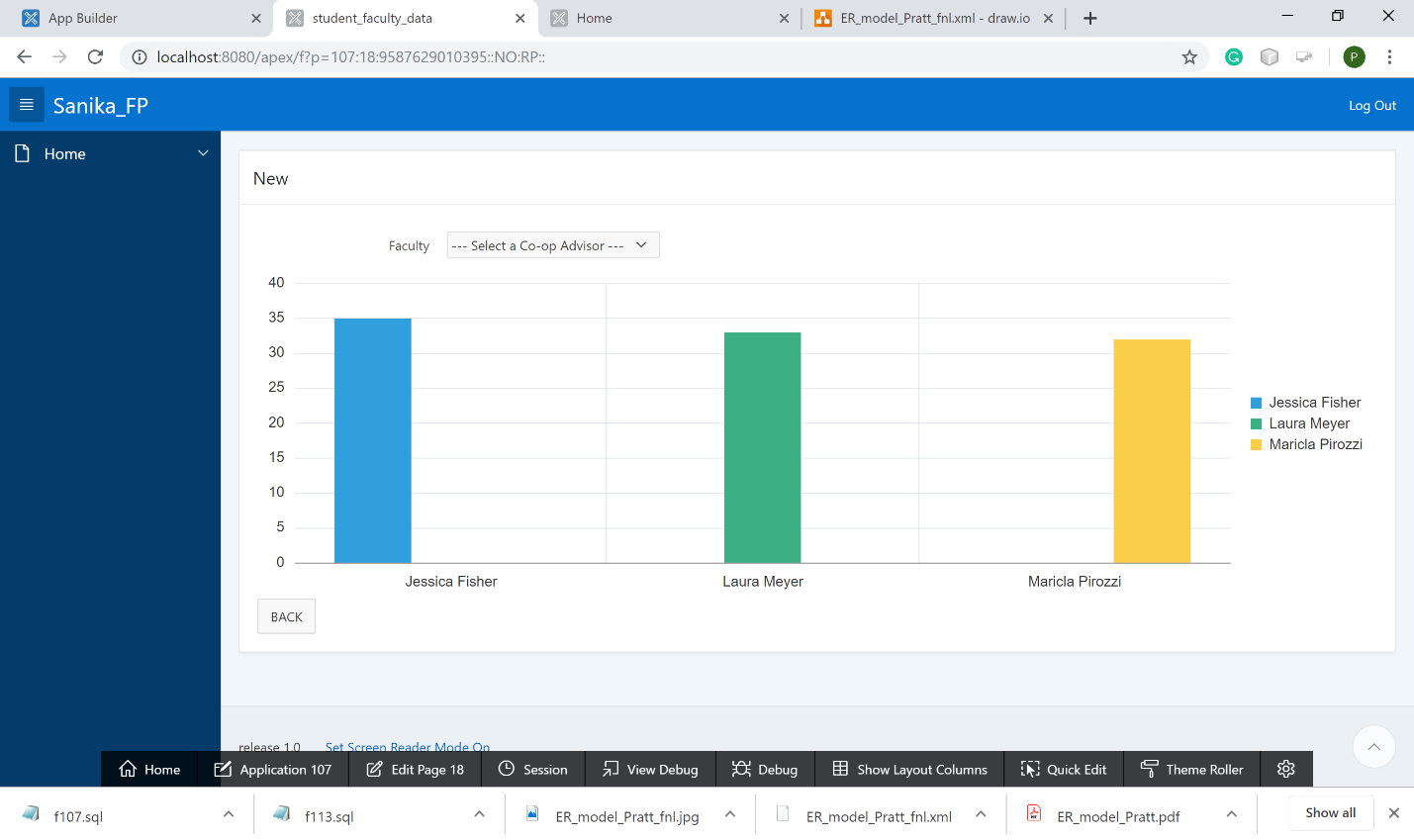
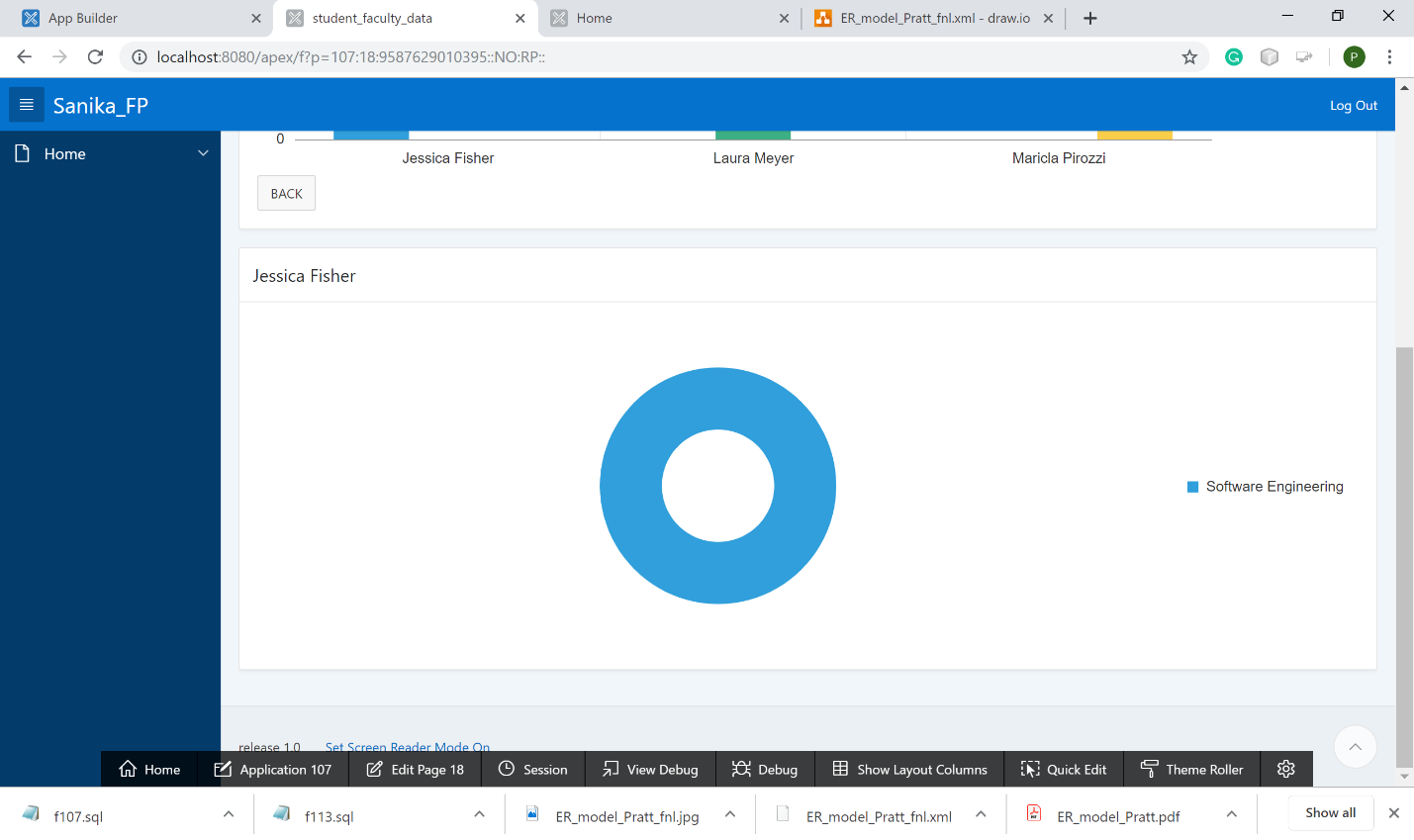
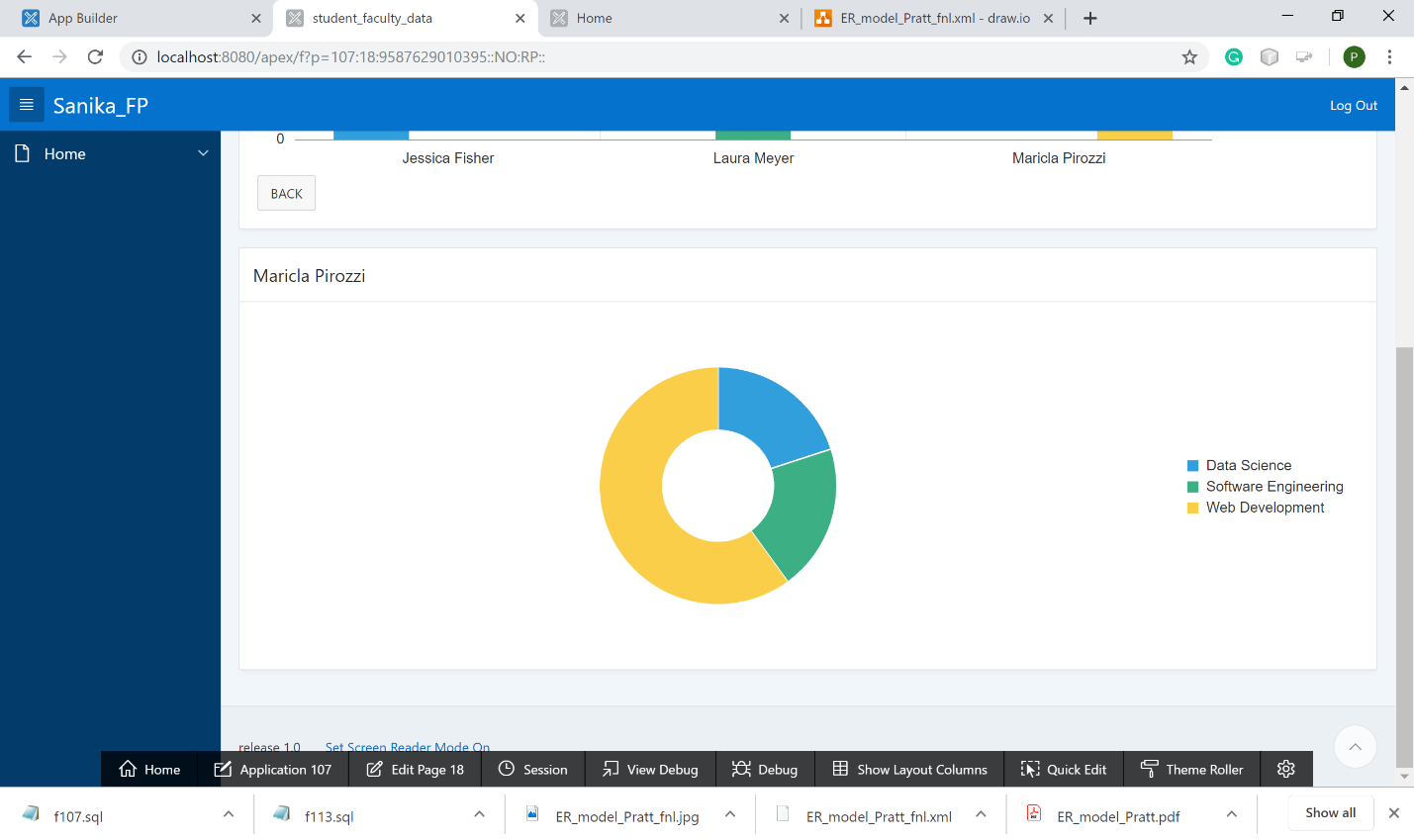
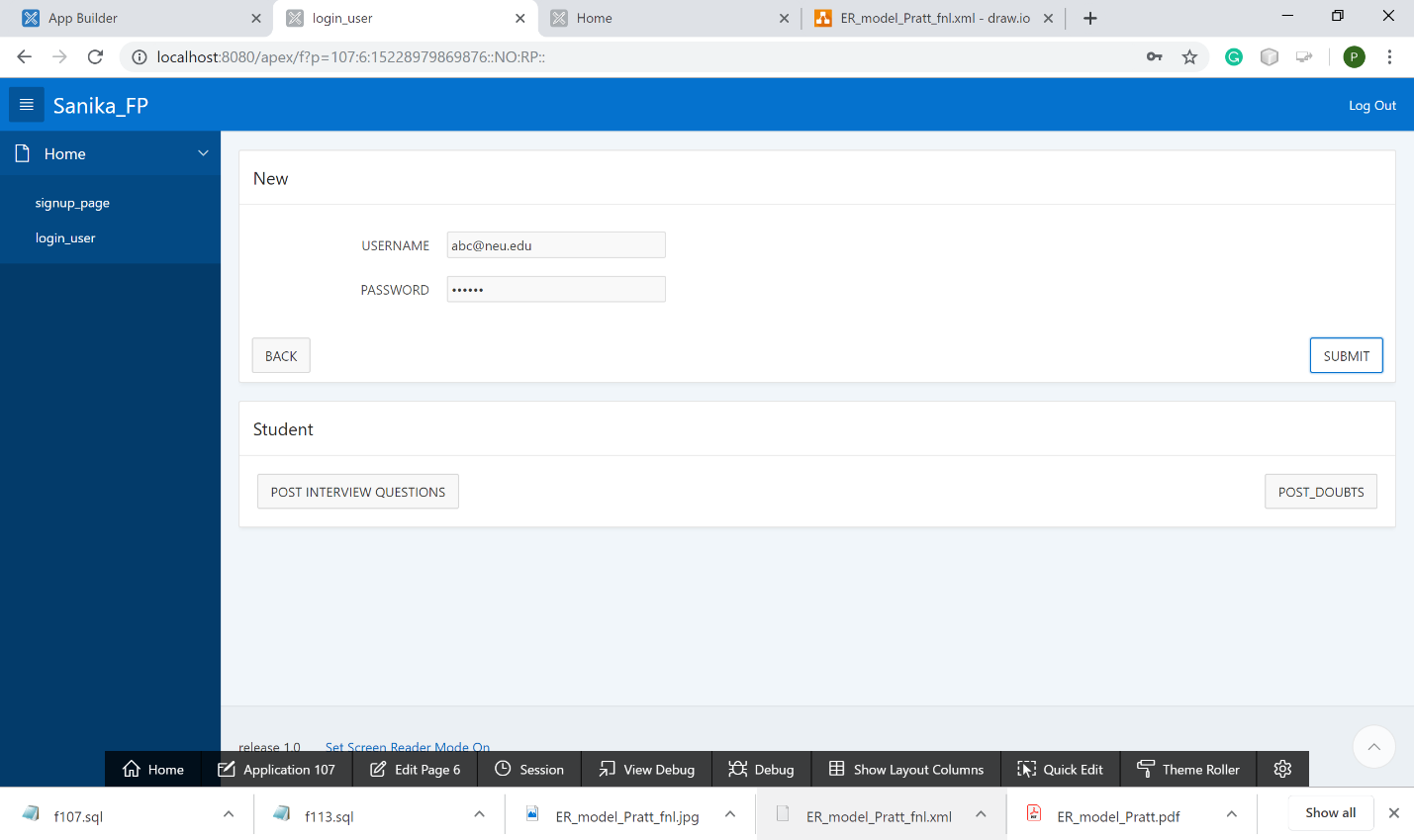
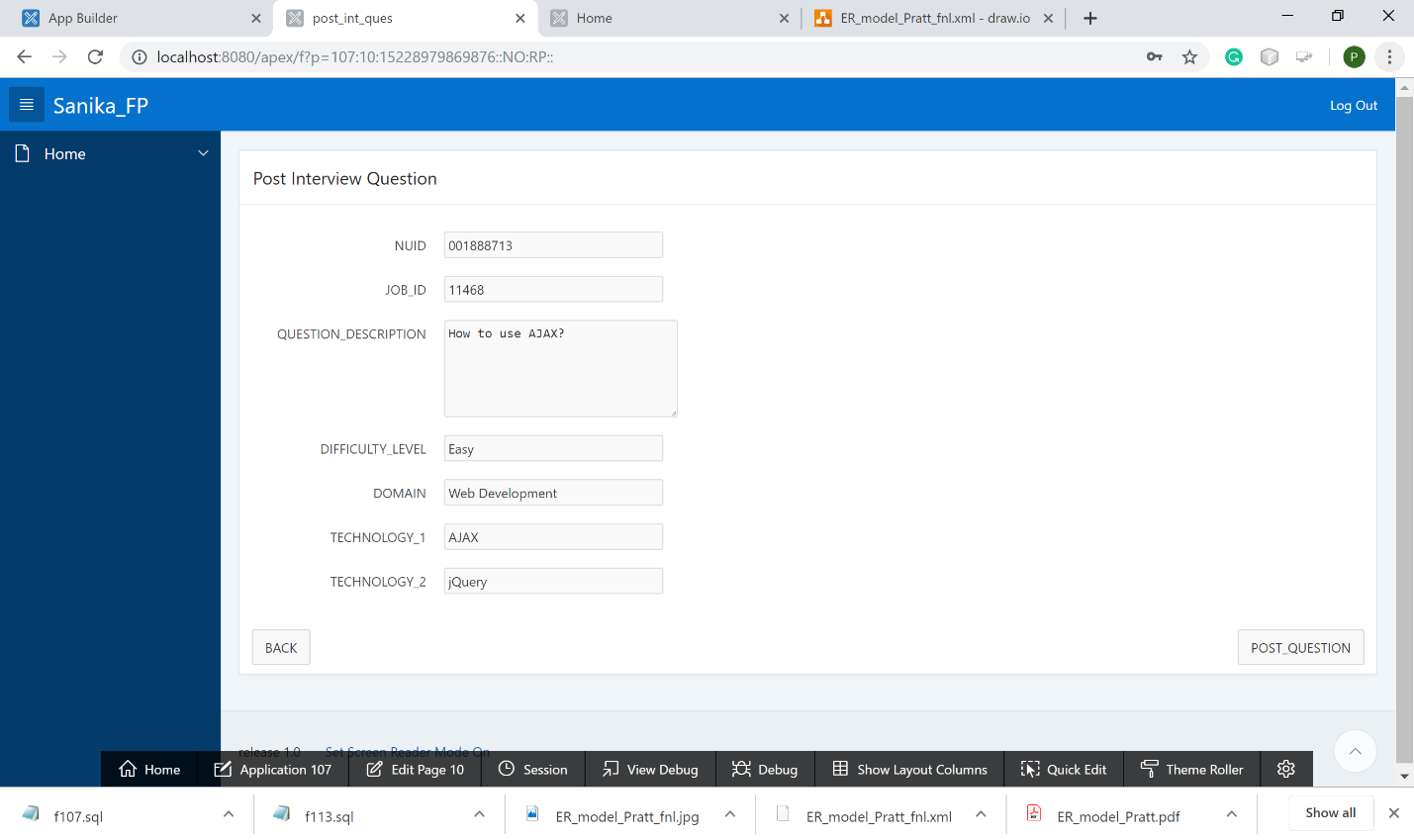
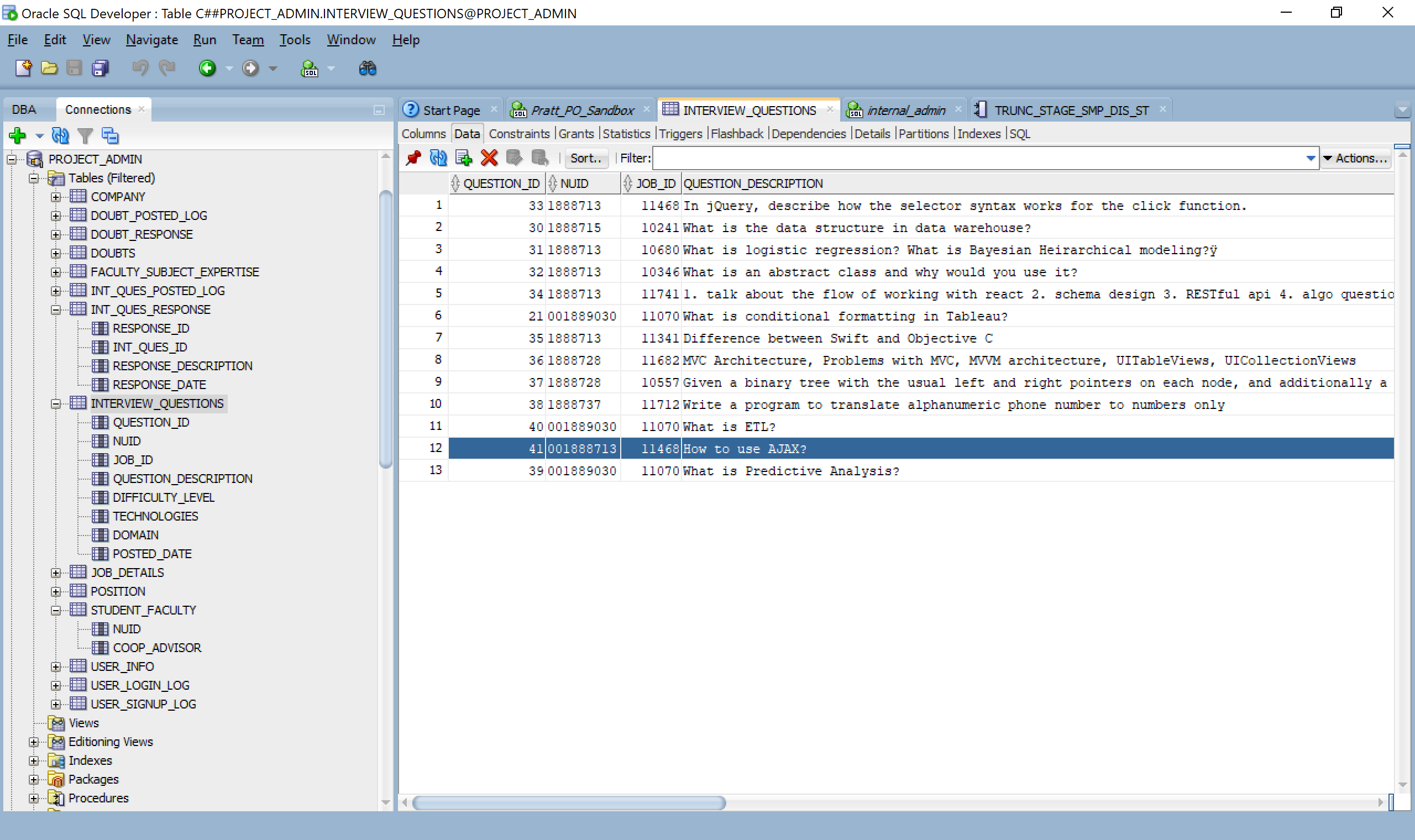
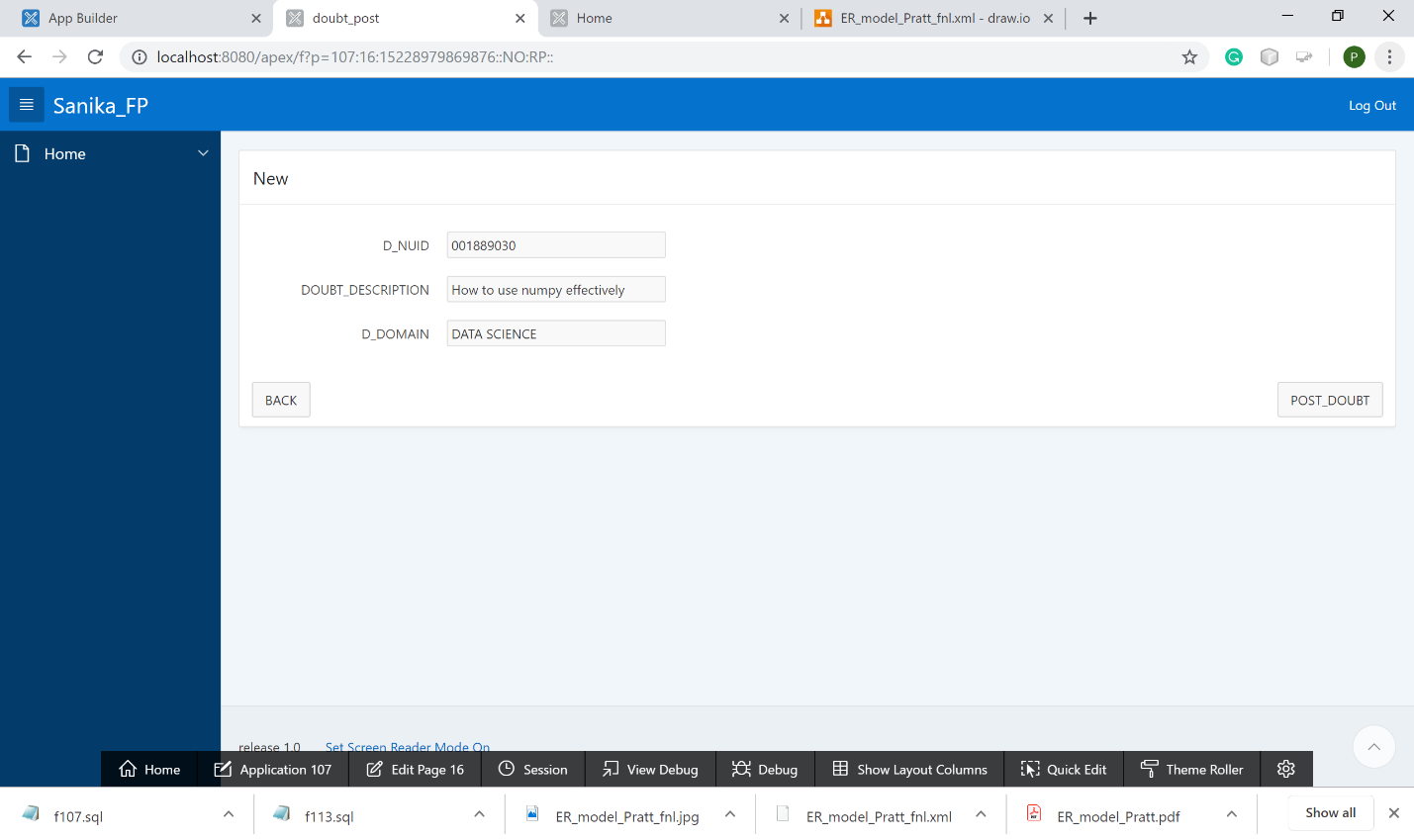
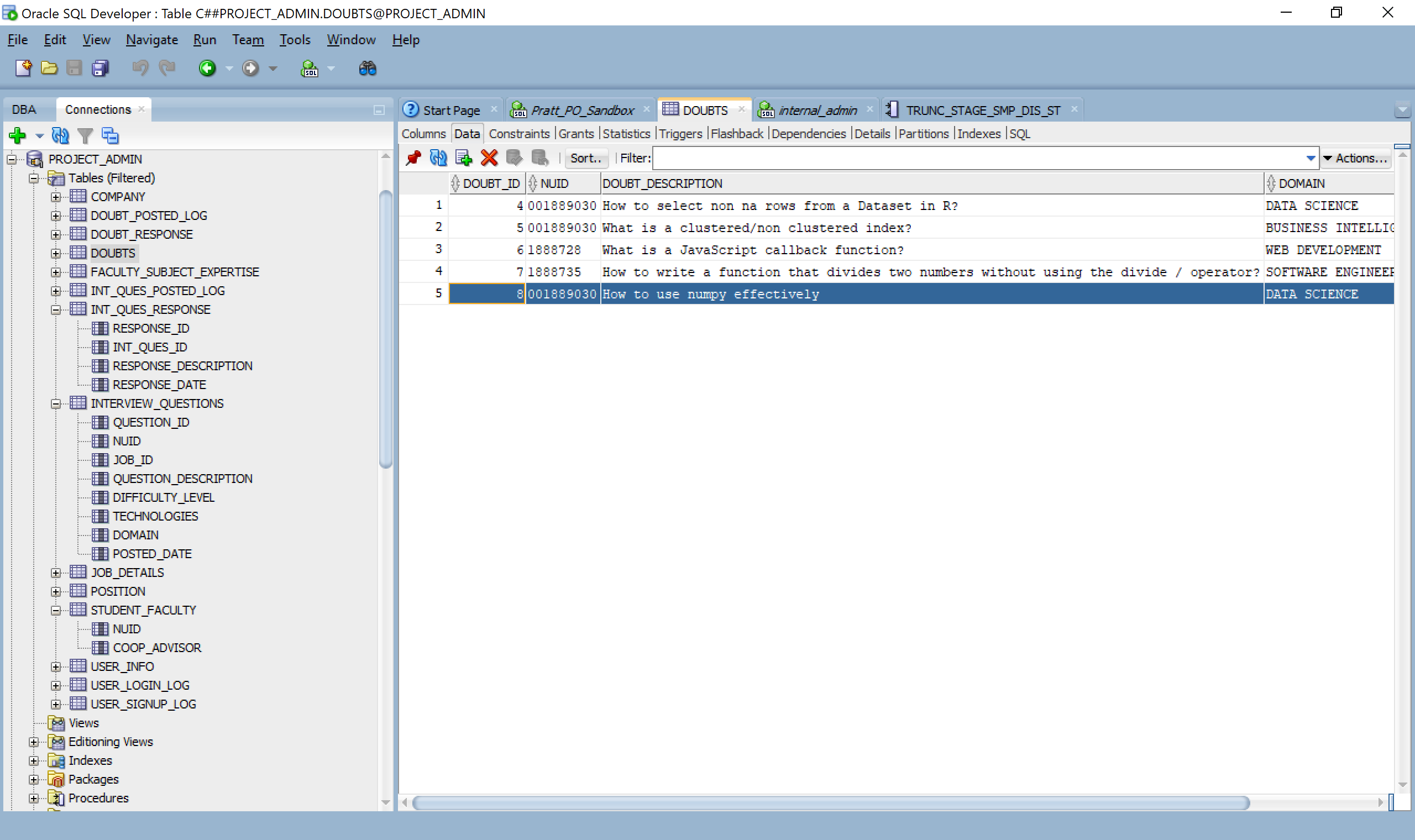
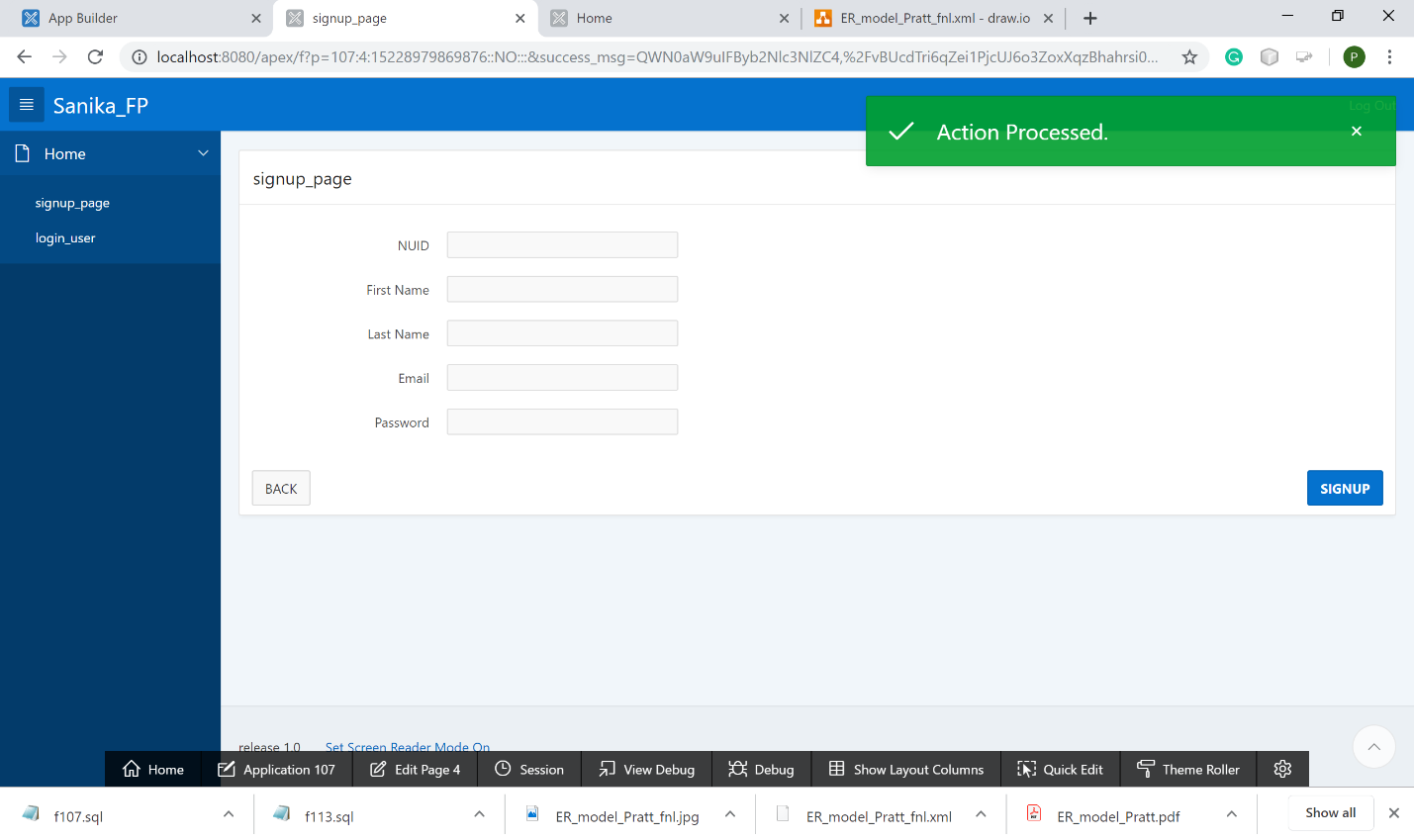
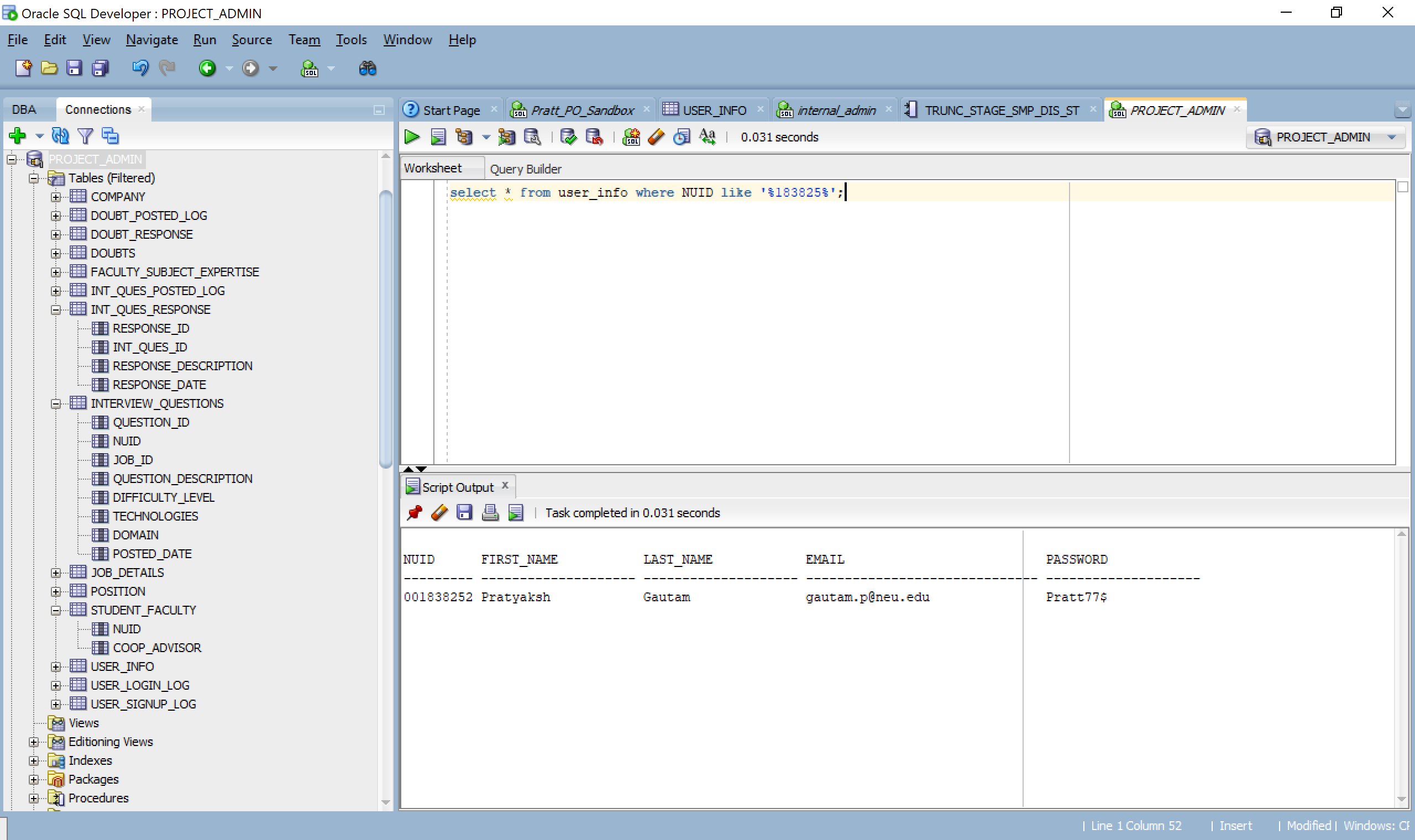
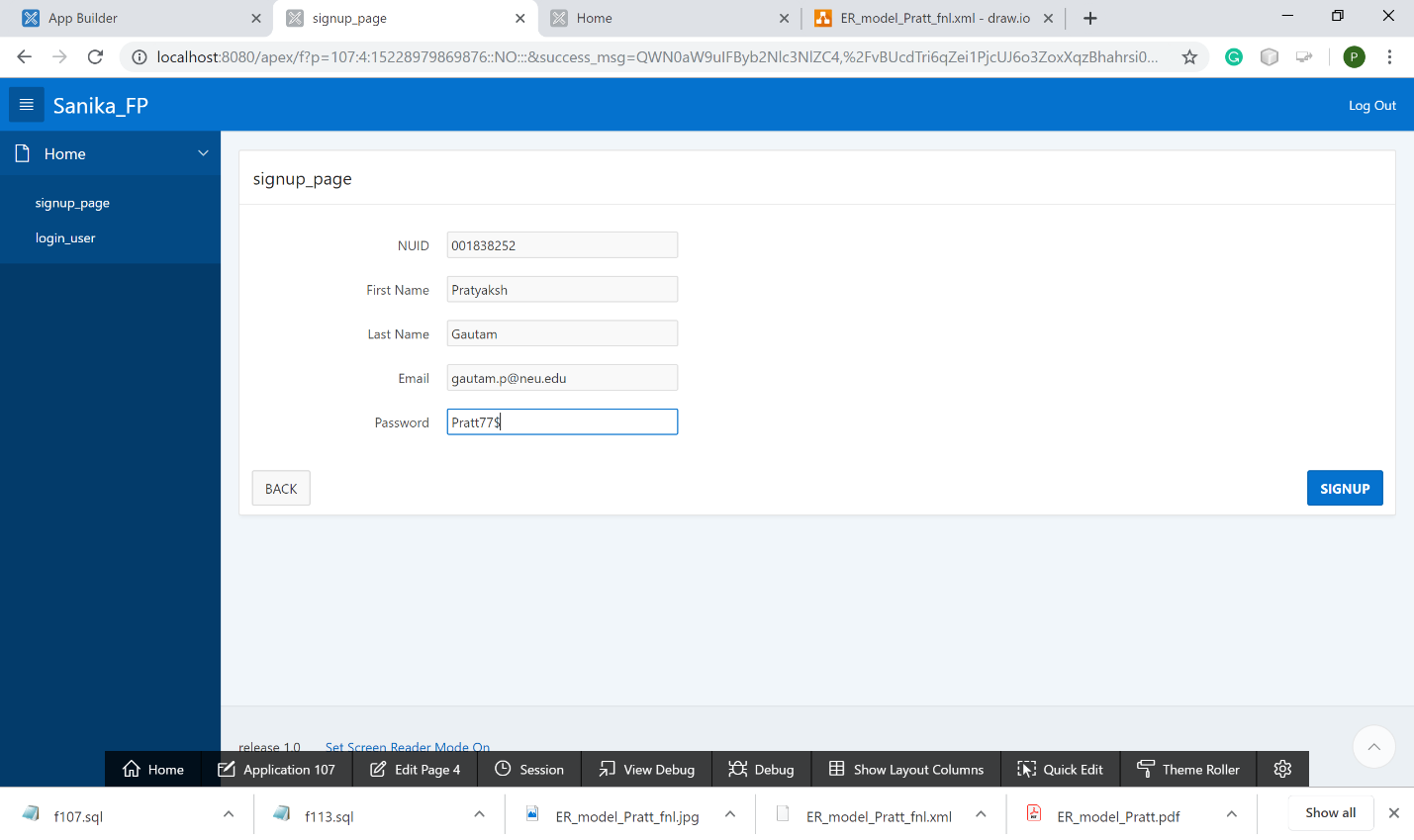
**Features Used**

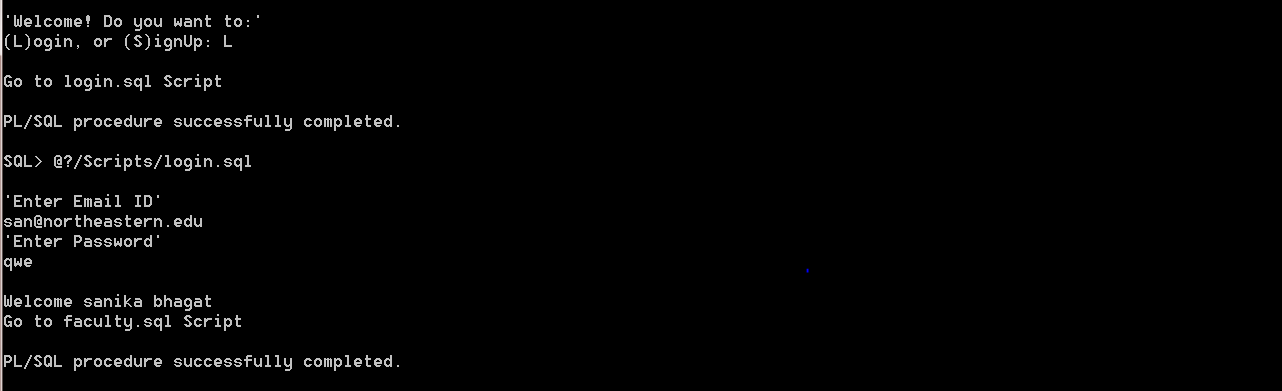
* **User input (‘&’)** - The user is asked for input in all the scripts so that the application works according to the user input. **Conditional functions and loops** have been used for the application to work based on the user input.
* **Pragma exceptions, pre-defined exceptions** – The exceptions are handled in every possible case where there was a chance of the exception being implemented. Check contraints exception, foreign key exceptions, no\_data\_found exception, value\_error exception are some of the exceptions to name a few
* **Triggers on log on schema, DML to populate log tables** - For any use cases where the data is being inserted into the table (Signup, post interview questions, post doubts, post response) or login, triggers have been implemented so that the information is stored in the log tables
* **Conditional statements** – Conditional statements and loops are used for writing a lot of cases so that the application works based totally on the user input.
* **Procedures and Functions** – Procedures and Functions were extensively used throughout the application for almost all the use cases such as View Interview Questions, View Interview Questions posted by user, View Doubts posted by User, View Responses to Interview Questions, View Responses to Doubts
* **Regexp** – Regexp is used to handle some cases such as email validation so that the user cannot enter anything as the email.
* **Collections (Varray, Record)** – Collections have been used where complex datatypes were needed such as while posting interview questions, the list of technologies were stored in a varray.
* **Cursors** – Cursors were used to fetch records depending upon the user input.
* **External Tables** – External tables were used to store some tables that were already populated but whose data was needed for some
* **Data Loading by SQL Loader** – Some
* **PDB and CDB** – PDB and CDB were created and the external tables and the tables loaded from SQL loader were stored in the PDBs.
* **Views** – Views were used to join get a view of joining multiple tables and then querying it
* **APEX Application** - Apart from the command line scripts, an APEX application has been implemented to show a working model of the Interview Questions Portal.
* **Creating users and granting privileges**

**Highlight:**

Apart from the command line scripts, an APEX application has been implemented to show a working model of the Interview Questions Portal.

The Screenshots are posted below:







**APPENDIX:**

1. Welcome.sql

set echo off

set serveroutput on

set verify off

set define '&'

prompt

prompt 'Welcome! Do you want to:'

accept tab prompt '(L)ogin, or (S)ignUp: '

prompt

declare

selection varchar2(1) := upper(substr('&tab',1,1));

begin

if selection = 'L' then

dbms\_output.put\_line('Go to login.sql Script');

-- Here the login.sql script should come

elsif selection = 'S' then

dbms\_output.put\_line('Go to signup.sql Script');

-- Here the script of signup.sql should come

else

dbms\_output.put\_line('Please enter valid options');

end if;

end;

/

1. Login.sql

set echo off

set serveroutput on

set verify off

set define '&'

prompt

prompt 'Enter Email ID'

accept email char

prompt 'Enter Password'

accept password char

prompt

DECLARE

user\_record USER\_INFO%rowtype;

row\_count number;

BEGIN

select \* into user\_record from USER\_INFO where EMAIL='&email' and PASSWORD='&password';

row\_count:=SQL%ROWCOUNT;

if (row\_count=1) then

dbms\_output.put\_line('Welcome '||user\_record.FIRST\_NAME||' '||user\_record.LAST\_NAME);

if(instr(user\_record.EMAIL,'northeastern.edu') > 0) then

dbms\_output.put\_line('Go to faculty.sql Script');

-- Here the script of faculty.sql should come

else

dbms\_output.put\_line('Go to student.sql Script');

-- Here the script of student.sql should come

end if;

else

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

end if;

EXCEPTION

when NO\_DATA\_FOUND then

dbms\_output.put\_line('Please enter valid credentials');

END;

/

1. Signup.sql

set echo off

set serveroutput on

set verify off

set define '&'

prompt

prompt 'Enter NUID'

accept nuid char

prompt 'Enter First Name'

accept first\_name char

prompt 'Enter Last Name'

accept last\_name char

prompt 'Enter Email ID'

accept email char

prompt 'Enter Password'

accept password char

prompt 'Confirm Password'

accept confirm\_password char

prompt

DECLARE

user\_record USER\_INFO%rowtype;

var1 number;

BEGIN

if REGEXP\_LIKE ('&email','^[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}$') then

if (instr('&email','northeastern.edu') > 0 or instr('&email','neu.edu') > 0) then

if('&password'='&confirm\_password') then

INSERT into USER\_INFO values ('&nuid','&first\_name','&last\_name','&email','&password');

var1 := SQL%ROWCOUNT;

if (var1 = 1) then

commit;

dbms\_output.put\_line('Number of Rows affected: ' || var1);

else

rollback;

end if;

else

dbms\_output.put\_line('The Passwords should match');

end if;

else

dbms\_output.put\_line('The Email ID should be university Email ID');

end if;

else

dbms\_output.put\_line('Please enter valid email');

end if;

END;

/

1. Faculty.sql

set echo off

set serveroutput on

set verify off

set define '&'

prompt

prompt 'Do you want to:'

accept tab prompt '(D)oubts Posted for you, (P)ost Responses to Doubts'

prompt

declare

selection varchar2(1) := upper(substr('&tab',1,1));

begin

if selection = 'D' then

dbms\_output.put\_line('Go to view\_doubt.sql Script');

-- Here the view\_doubt.sql script should come

elsif selection = 'P' then

dbms\_output.put\_line('Go to call\_func\_doubt\_response.sql Script');

-- Here the script of call\_func\_doubt\_response.sql should come

else

dbms\_output.put\_line('Please enter valid options');

end if;

end;

/

1. Student.sql

set echo off

set serveroutput on

set verify off

set define '&'

prompt

prompt 'Do you want to:'

accept tab prompt '(P)ost (V)iew Int Ques, (R)esp Vi(E)w to Ques, (A)sk Doubt, (I)nt Ques by you, (T)ips'

prompt

declare

selection varchar2(1) := upper(substr('&tab',1,1));

begin

if selection = 'P' then

dbms\_output.put\_line('Go to post\_interview\_question.sql Script');

-- Here the post\_interview\_question.sql script should come

elsif selection = 'V' then

dbms\_output.put\_line('Go to view\_interview\_questions.sql Script');

-- Here the script of view\_interview\_questions.sql should come

elsif selection = 'R' then

dbms\_output.put\_line('Go to call\_proc\_int\_ques\_response.sql Script');

-- Here the question\_response.sql script should come

elsif selection = 'A' then

dbms\_output.put\_line('Go to post\_doubt.sql Script');

-- Here the script of post\_doubt.sql should come

elsif selection = 'I' then

dbms\_output.put\_line('Go to call\_proc\_int\_ques\_by\_user.sql Script');

-- Here the script of call\_proc\_int\_ques\_by\_user.sql should come

elsif selection = 'T' then

dbms\_output.put\_line('Go to interview\_tips\_tricks.sql Script');

-- Here the script of interview\_tips\_tricks.sql should come

elsif selection = 'E' then

dbms\_output.put\_line('Go to call\_proc\_view\_response\_int\_question.sql Script');

-- Here the script of call\_proc\_view\_response\_int\_question.sql should come

else

dbms\_output.put\_line('Please enter valid options');

end if;

end;

/

1. post\_interview\_question.sql

set echo off

set serveroutput on

set verify off

set define '&'

prompt

prompt 'Enter NUID'

accept nuid char

prompt 'Enter JOB ID'

accept job\_id number

prompt 'Enter Interview Question'

accept question char

prompt 'Enter Difficulty Level'

accept difficulty char

prompt 'Enter the Technology 1'

accept technology\_list\_1 char

prompt 'Enter the Technology 2'

accept technology\_list\_2 char

prompt

DECLARE

var1 number;

domain varchar2(40);

CHECK\_CONSTRAINT\_VIOLATED EXCEPTION;

PRAGMA EXCEPTION\_INIT(CHECK\_CONSTRAINT\_VIOLATED, -2290);

FOREIGN\_KEY\_CONSTRAINT\_VIOLATED EXCEPTION;

PRAGMA EXCEPTION\_INIT(FOREIGN\_KEY\_CONSTRAINT\_VIOLATED, -2291);

BEGIN

SELECT DOMAIN into domain FROM JOB\_DETAILS where job\_id = '&job\_id' ;

INSERT into INTERVIEW\_QUESTIONS(NUID,JOB\_ID,QUESTION\_DESCRIPTION,DIFFICULTY\_LEVEL,TECHNOLOGIES,DOMAIN,POSTED\_DATE) values ('&nuid',&job\_id,'&question','&difficulty',technology\_list\_type(technology\_type('&technology\_list\_1'),technology\_type('&technology\_list\_2')),domain,SYSDATE);

var1 := SQL%ROWCOUNT;

if (var1 = 1) then

commit;

dbms\_output.put\_line('Number of Rows affected: ' || var1);

else

rollback;

end if;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

dbms\_output.put\_line('Please enter valid details');

WHEN CHECK\_CONSTRAINT\_VIOLATED THEN

dbms\_output.put\_line('Insert failed due to check constraint violation');

WHEN FOREIGN\_KEY\_CONSTRAINT\_VIOLATED THEN

dbms\_output.put\_line('Insert failed due to foreign key constraint violation');

END;

/

1. post\_doubt.sql

set echo off

set serveroutput on

set verify off

set define '&'

prompt

prompt 'Enter NUID'

accept nuid char

prompt 'Enter Doubt'

accept doubt char

prompt 'Enter Domain'

accept domain char

prompt

DECLARE

var1 number;

FOREIGN\_KEY\_CONSTRAINT\_VIOLATED EXCEPTION;

PRAGMA EXCEPTION\_INIT(FOREIGN\_KEY\_CONSTRAINT\_VIOLATED, -2291);

CHECK\_CONSTRAINT\_VIOLATED EXCEPTION;

PRAGMA EXCEPTION\_INIT(CHECK\_CONSTRAINT\_VIOLATED, -2290);

BEGIN

INSERT into DOUBTS(NUID,DOUBT\_DESCRIPTION,DOMAIN, POSTED\_DATE) values ('&nuid','&doubt',upper('&domain'),SYSDATE);

var1 := SQL%ROWCOUNT;

if (var1 = 1) then

commit;

dbms\_output.put\_line('Number of Rows affected: ' || var1);

else

rollback;

end if;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

dbms\_output.put\_line('Please enter valid details');

WHEN FOREIGN\_KEY\_CONSTRAINT\_VIOLATED THEN

dbms\_output.put\_line('Insert failed due to foreign key constraint violation');

WHEN CHECK\_CONSTRAINT\_VIOLATED THEN

dbms\_output.put\_line('Insert failed due to Invalid Domain');

END;

/

Procedure Script:

------------------------------------------------------------------------------------------------------------------------

1. View Interview Question based on Domain:

------------------------------------------------------------------------------------------------------------------------

A. Create a Procedure

------------------------------------------------------------------------------------------------------------------------

create or replace procedure view\_interview\_questions(i\_domain in varchar2, int\_ques\_refcursor out sys\_refcursor)

is

begin

open int\_ques\_refcursor for select QUESTION\_ID, JOB\_ID, QUESTION\_DESCRIPTION, DIFFICULTY\_LEVEL, POSTED\_DATE from INTERVIEW\_QUESTIONS where DOMAIN = i\_domain

order by POSTED\_DATE DESC;

end;

B. Executing the Procedure

------------------------------------------------------------------------------------------------------------------------

DECLARE

type iq\_out\_record is record(

i\_question\_id interview\_questions.question\_id%type,

i\_job\_id interview\_questions.job\_id%type,

i\_question\_description interview\_questions.question\_description%type,

i\_difficulty\_level interview\_questions.difficulty\_level%type,

i\_posted\_date interview\_questions.posted\_date%type

);

iq\_out iq\_out\_record;

i\_domain interview\_questions.domain%type;

i\_int\_ques\_refcursor SYS\_REFCURSOR;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('QUESTION ID JOB\_ID QUESTION DESCRIPTION DIFFICULTY LEVEL POSTED DATE');

DBMS\_OUTPUT.PUT\_LINE('-----------------------------------------------------------------------------------------------------------------');

view\_interview\_questions('&i\_domain', i\_int\_ques\_refcursor);

FETCH i\_int\_ques\_refcursor INTO iq\_out;

while i\_int\_ques\_refcursor%FOUND

loop

DBMS\_OUTPUT.PUT\_LINE(iq\_out.i\_question\_id || ' ' || iq\_out.i\_job\_id || ' ' || iq\_out.i\_question\_description || ' '|| iq\_out.i\_difficulty\_level || ' '||iq\_out.i\_posted\_date);

FETCH i\_int\_ques\_refcursor INTO iq\_out;

end loop;

if i\_int\_ques\_refcursor%ROWCOUNT=0 THEN

dbms\_output.put\_line('Please enter a valid Domain Name');

CLOSE i\_int\_ques\_refcursor;

end if;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

dbms\_output.put\_line('No records');

END;

/

------------------------------------------------------------------------------------------------------------------------

2. View Interview Questions Posted by User:

------------------------------------------------------------------------------------------------------------------------

A: Create a Procedure

------------------------------------------------------------------------------------------------------------------------

create or replace procedure int\_ques\_by\_user(i\_nuid in char, int\_ques\_user\_refcursor out sys\_refcursor)

is

begin

open int\_ques\_user\_refcursor for select QUESTION\_ID, JOB\_ID, QUESTION\_DESCRIPTION, DIFFICULTY\_LEVEL, POSTED\_DATE from INTERVIEW\_QUESTIONS where NUID = i\_nuid

order by POSTED\_DATE DESC;

end;

B: Executing the Procedure

------------------------------------------------------------------------------------------------------------------------

DECLARE

type iq\_out\_record is record(

i\_question\_id interview\_questions.question\_id%type,

i\_job\_id interview\_questions.job\_id%type,

i\_question\_description interview\_questions.question\_description%type,

i\_difficulty\_level interview\_questions.difficulty\_level%type,

i\_posted\_date interview\_questions.posted\_date%type

);

iq\_out iq\_out\_record;

i\_nuid interview\_questions.nuid%type;

i\_int\_ques\_user\_refcursor SYS\_REFCURSOR;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('QUESTION ID JOB\_ID QUESTION DESCRIPTION DIFFICULTY LEVEL POSTED DATE');

DBMS\_OUTPUT.PUT\_LINE('-----------------------------------------------------------------------------------------------------------------');

int\_ques\_by\_user('&i\_nuid', i\_int\_ques\_user\_refcursor);

FETCH i\_int\_ques\_user\_refcursor INTO iq\_out;

while i\_int\_ques\_user\_refcursor%FOUND

loop

DBMS\_OUTPUT.PUT\_LINE(iq\_out.i\_question\_id || ' ' || iq\_out.i\_job\_id || ' ' || iq\_out.i\_question\_description || ' '|| iq\_out.i\_difficulty\_level || ' '||iq\_out.i\_posted\_date);

FETCH i\_int\_ques\_user\_refcursor INTO iq\_out;

end loop;

if i\_int\_ques\_user\_refcursor%ROWCOUNT=0 THEN

dbms\_output.put\_line('Please enter valid NUID');

CLOSE i\_int\_ques\_user\_refcursor;

end if;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

dbms\_output.put\_line('No records');

END;

/

------------------------------------------------------------------------------------------------------------------------

3. View Doubts Posted by User:

------------------------------------------------------------------------------------------------------------------------

A: Create a Procedure

------------------------------------------------------------------------------------------------------------------------

create or replace procedure doubts\_by\_user(i\_nuid in char, doubts\_user\_refcursor out sys\_refcursor)

is

begin

open doubts\_user\_refcursor for select DOUBT\_ID, DOUBT\_DESCRIPTION, DOMAIN, POSTED\_DATE from DOUBTS where NUID = i\_nuid

order by POSTED\_DATE DESC;

end;

B. Executing the PROCEDURE

------------------------------------------------------------------------------------------------------------------------

DECLARE

type d\_out\_record is record(

i\_doubt\_id doubts.doubt\_id%type,

i\_doubt\_description doubts.doubt\_description%type,

i\_domain doubts.domain%type,

i\_posted\_date doubts.posted\_date%type

);

d\_out d\_out\_record;

i\_nuid interview\_questions.nuid%type;

i\_doubts\_user\_refcursor SYS\_REFCURSOR;

BEGIN

DBMS\_OUTPUT.PUT\_LINE('DOUBT ID DOUBT DESCRIPTION DOMAIN POSTED DATE');

DBMS\_OUTPUT.PUT\_LINE('-----------------------------------------------------------------------------------------------------------------');

doubts\_by\_user('&i\_nuid', i\_doubts\_user\_refcursor);

FETCH i\_doubts\_user\_refcursor INTO d\_out;

while i\_doubts\_user\_refcursor%FOUND

loop

DBMS\_OUTPUT.PUT\_LINE(d\_out.i\_doubt\_id || ' ' || d\_out.i\_doubt\_description || ' ' || d\_out.i\_domain || ' '||d\_out.i\_posted\_date);

FETCH i\_doubts\_user\_refcursor INTO d\_out;

end loop;

if i\_doubts\_user\_refcursor%ROWCOUNT=0 THEN

dbms\_output.put\_line('Please enter valid NUID');

CLOSE i\_doubts\_user\_refcursor;

end if;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

dbms\_output.put\_line('No records');

END;

/

------------------------------------------------------------------------------------------------------------------------

4. Response to Interview Question Posted:

------------------------------------------------------------------------------------------------------------------------

A: Create a Procedure

------------------------------------------------------------------------------------------------------------------------

create or replace PROCEDURE INT\_QUES\_RESPONSE\_FUNC(INT\_QUES IN NUMBER, RESPONSE IN LONG, ROWS\_AFFECTED OUT NUMBER)

AS

SQL\_STMT VARCHAR2(500);

sysdate\_var date;

BEGIN

sysdate\_var := SYSDATE;

SQL\_STMT := 'INSERT INTO INT\_QUES\_RESPONSE(INT\_QUES\_ID, RESPONSE\_DESCRIPTION,RESPONSE\_DATE) VALUES (:1,:2,:3)';

EXECUTE IMMEDIATE SQL\_STMT USING INT\_QUES, RESPONSE, sysdate\_var;

ROWS\_AFFECTED := SQL%ROWCOUNT;

if (ROWS\_AFFECTED = 1) then

commit;

else

rollback;

end if;

END;

B. Executing the PROCEDURE

------------------------------------------------------------------------------------------------------------------------

set serveroutput on

DECLARE

i\_response\_description int\_ques\_response.response\_description%type;

i\_question\_id interview\_questions.question\_id%type;

rows\_affected number;

BEGIN

INT\_QUES\_RESPONSE\_FUNC(&i\_question\_id, '&i\_response\_description',rows\_affected);

dbms\_output.put\_line('Number of Rows Affected: ' || to\_char(rows\_affected));

END;

/

------------------------------------------------------------------------------------------------------------------------

Trigger Scripts:

---------------------------------------------------------------------------------------------------------------

1. User Login Trigger

---------------------------------------------------------------------------------------------------------------

create or replace TRIGGER USER\_LOGIN\_TRIGGER

AFTER LOGON

ON C##PROJECT\_ADMIN.SCHEMA

BEGIN

INSERT INTO USER\_LOGIN\_LOG (USERNAME, LOGIN\_DATE) VALUES (USER, SYSDATE);

COMMIT;

END;

---------------------------------------------------------------------------------------------------------------

2. User Signup Trigger

create or replace TRIGGER USER\_SIGNUP\_TRIGGER

AFTER INSERT ON USER\_INFO

FOR EACH ROW

BEGIN

if (instr(:new.EMAIL,'northeastern.edu')>0) then

INSERT INTO USER\_SIGNUP\_LOG values (:new.email, 'Faculty',SYSDATE);

else

INSERT INTO USER\_SIGNUP\_LOG values (:new.email, 'Student',SYSDATE);

end if;

END USER\_SIGNUP\_TRIGGER;

/

---------------------------------------------------------------------------------------------------------------

3. Insert Interview Questions Trigger

create or replace TRIGGER INT\_QUES\_POSTED\_TRIGGER

AFTER INSERT

OR DELETE

ON INTERVIEW\_QUESTIONS

FOR EACH ROW

BEGIN

CASE

WHEN INSERTING THEN

INSERT INTO INT\_QUES\_POSTED\_LOG values (:new.QUESTION\_ID, :new.NUID,:new.POSTED\_DATE);

WHEN DELETING THEN

DELETE FROM INT\_QUES\_POSTED\_LOG WHERE QUESTION\_ID = :new.QUESTION\_ID;

END CASE;

END INT\_QUES\_POSTED\_TRIGGER;

---------------------------------------------------------------------------------------------------------------

4. Insert Interview Questions Trigger

create or replace TRIGGER DOUBT\_POSTED\_TRIGGER

AFTER INSERT

OR DELETE

ON DOUBTS

FOR EACH ROW

BEGIN

CASE

WHEN INSERTING THEN

INSERT INTO DOUBT\_POSTED\_LOG values (:new.DOUBT\_ID, :new.NUID,:new.POSTED\_DATE);

WHEN DELETING THEN

DELETE FROM DOUBT\_POSTED\_LOG WHERE DOUBT\_ID = :new.DOUBT\_ID;

END CASE;

END DOUBT\_POSTED\_TRIGGER;

Trigger Tables:

---------------------------------------------------------------------------------------------------------------

1. Login Trigger Table

---------------------------------------------------------------------------------------------------------------

CREATE TABLE USER\_LOGIN\_LOG(

USERNAME VARCHAR2(100),

LOGIN\_DATE DATE );

---------------------------------------------------------------------------------------------------------------

---------------------------------------------------------------------------------------------------------------

2. Signup Trigger Table

---------------------------------------------------------------------------------------------------------------

CREATE TABLE USER\_SIGNUP\_LOG(

USERNAME VARCHAR2(30),

DESIGNATION VARCHAR2(30),

SIGNUP\_DATE DATE );

---------------------------------------------------------------------------------------------------------------

---------------------------------------------------------------------------------------------------------------

3. Interview Question Posted Table

---------------------------------------------------------------------------------------------------------------

CREATE TABLE INT\_QUES\_POSTED\_LOG(

QUESTION\_ID NUMBER,

NUID NUMBER,

POSTED\_DATE DATE );

---------------------------------------------------------------------------------------------------------------

---------------------------------------------------------------------------------------------------------------

4. Doubt Posted Table

---------------------------------------------------------------------------------------------------------------

CREATE TABLE DOUBT\_POSTED\_LOG(

DOUBT\_ID NUMBER,

NUID NUMBER,

POSTED\_DATE DATE );

---------------------------------------------------------------------------------------------------------------

VArray Scripts:

---------------------------------------------------------------------------------------------------------------

1. Technology List VArray

---------------------------------------------------------------------------------------------------------------

create or replace TYPE TECHNOLOGY\_LIST\_TYPE AS VARRAY(5) OF TECHNOLOGY\_TYPE;

create or replace TYPE TECHNOLOGY\_TYPE AS OBJECT (object\_value VARCHAR2(200));

---------------------------------------------------------------------------------------------------------------