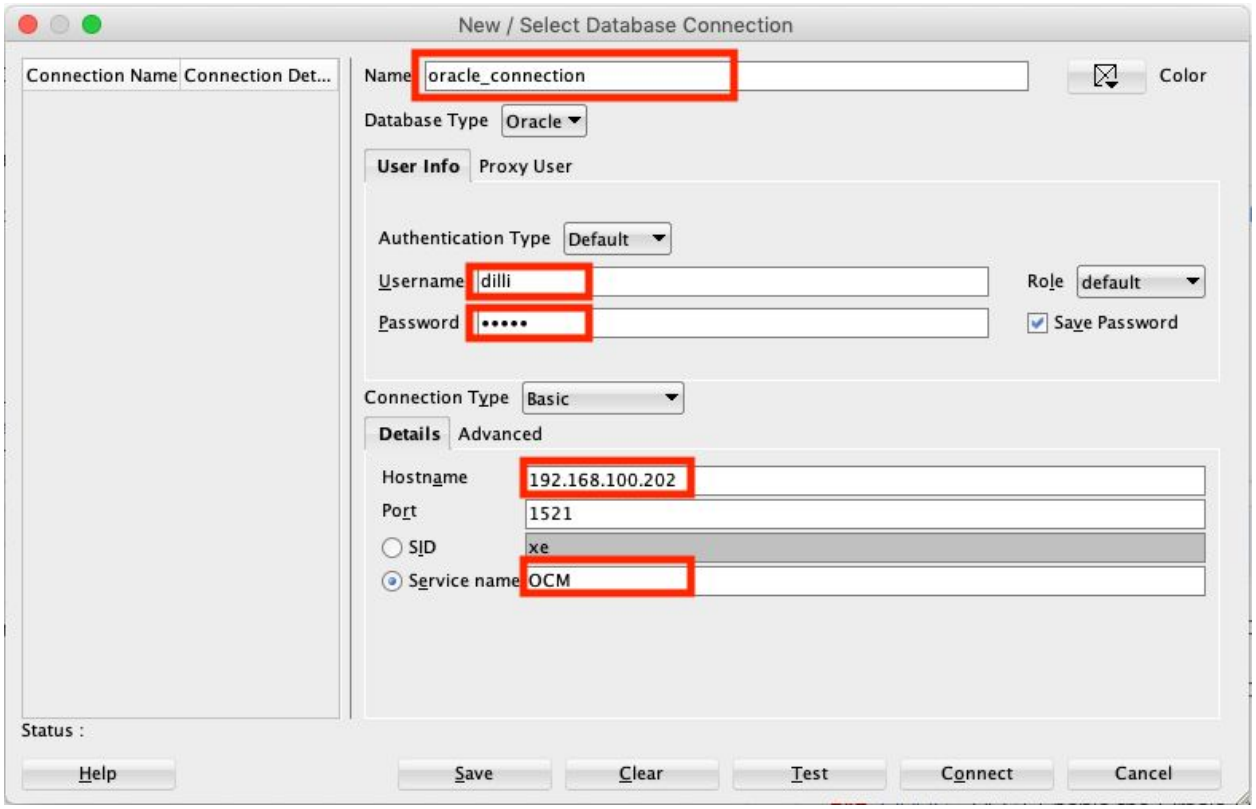
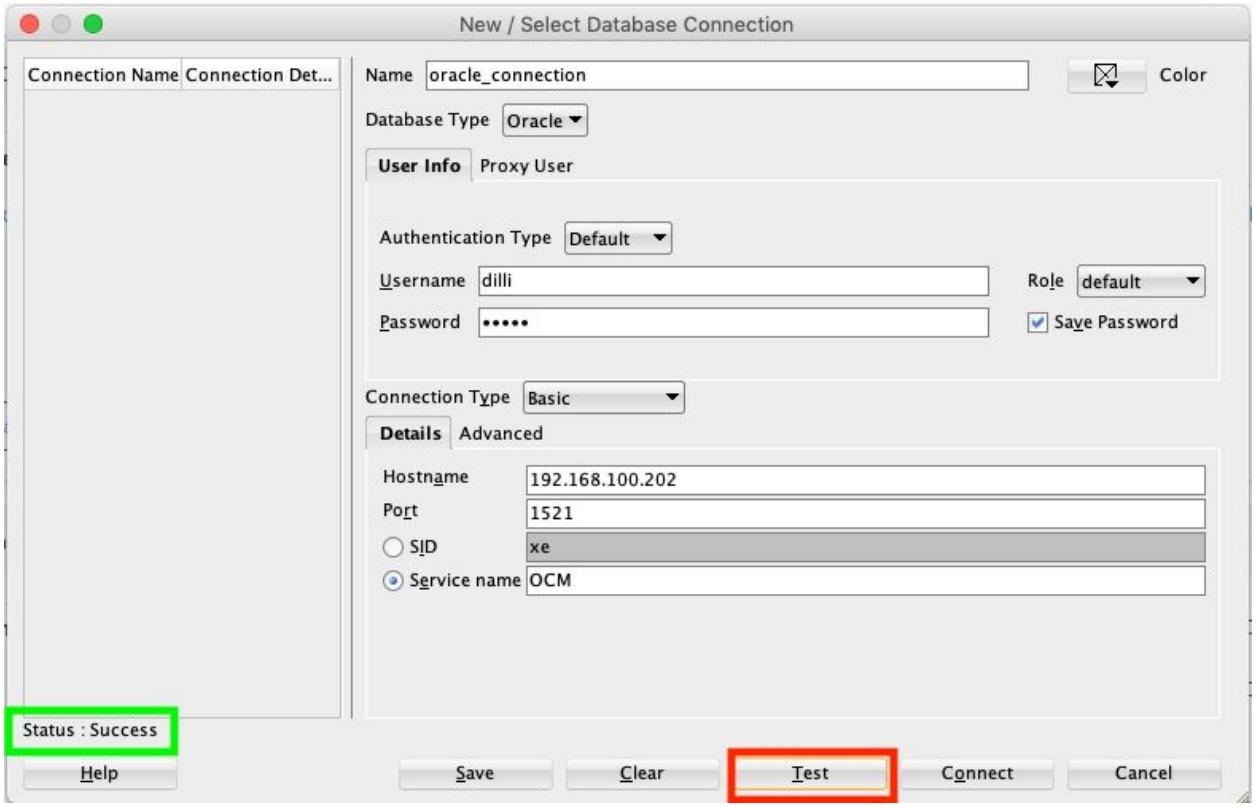
Using SQL Developer

1. Open SQL Developer and click on + icon at upper left.

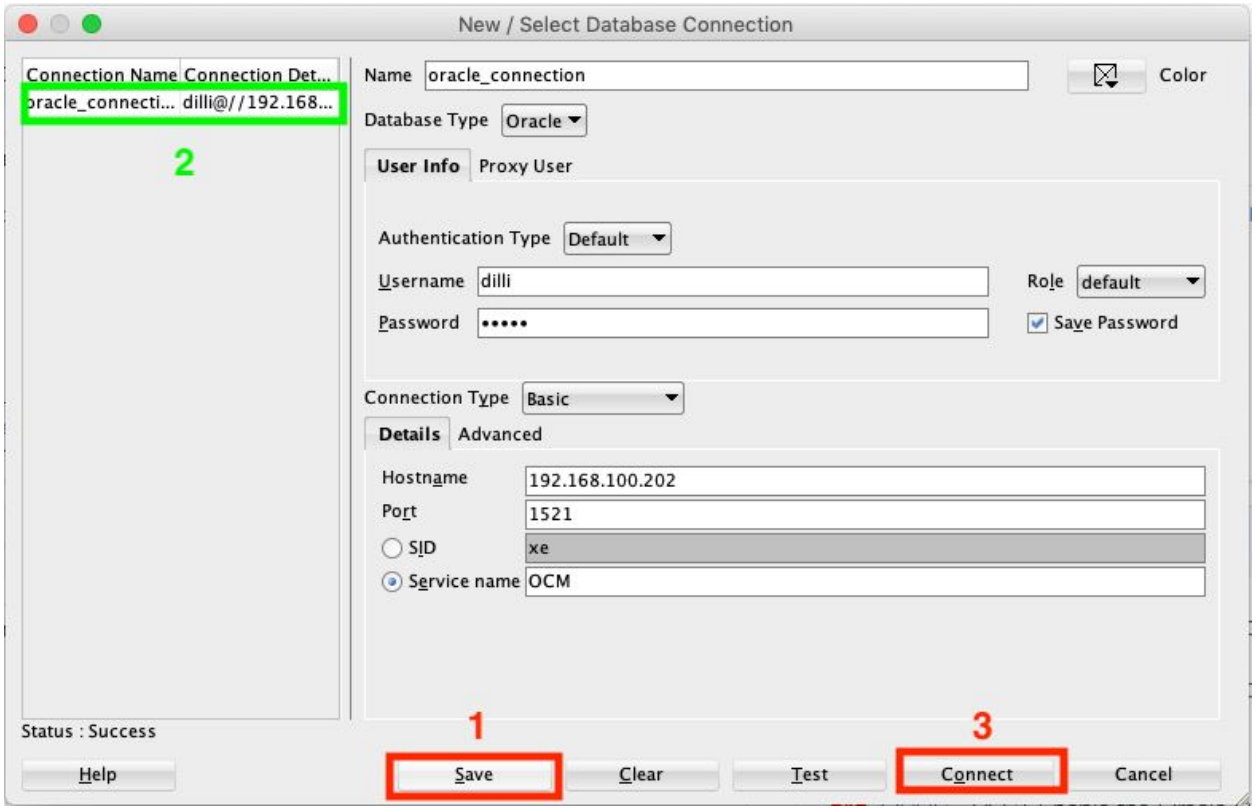
2. On new database connection window provide connection name, username, password, hostname and service name.



3. Click on Test and check if it is a success or not.

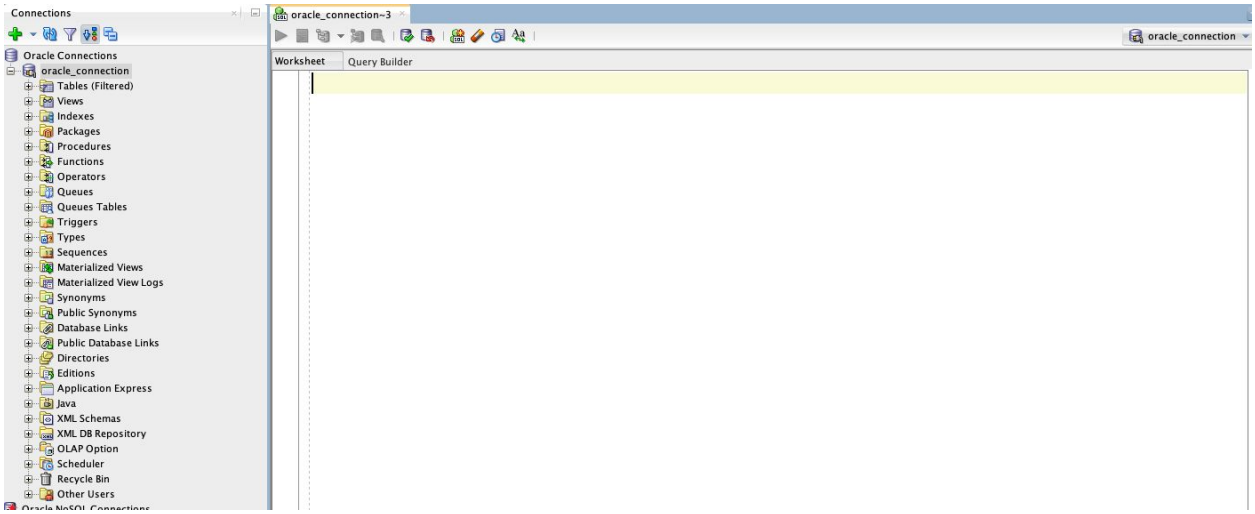


4. Click on Save, You will notice connection Name and connection details on left panel. Click on connect.



5. Connection window will be popped up and provide user/pass and click on OK.

6. Workspace panel will be displayed once you login to the database.



DDL SQL Commands

```
-- tbl_course
CREATE TABLE tbl_course(
        course_id number constraint pk_cid primary key,
        course_name varchar2(32) not null,
        course_duration number not null,
        course_fee number(7,2) not null,
        pass_mark number(2) not null
);

comment on table tbl_course is 'Table to store course information';

-- tbl_teacher
CREATE TABLE tbl_teacher(
        teacher_id number constraint pk_tid primary key,
        teacher_name varchar2(32) not null
);

comment on table tbl_teacher is 'Table to store teacher information';
```

```
-- tbl_student
CREATE TABLE tbl_student(
	student_id number constraint pk_sid primary key,
	student_name varchar2(32) not null
);

comment on table tbl_student is 'Table to store student information';

-- tbl_course_enrollment
CREATE TABLE tbl_course_enrollment(
	course_id number,
	teacher_id number,
	student_id number,
	start_date date,
	completion_date date,
	constraint fk_cid_enroll foreign key(course_id) references tbl_course(course_id),
	constraint fk_tid_enroll foreign key(teacher_id) references tbl_teacher(teacher_id),
	constraint fk_sid_enroll foreign key(student_id) references tbl_student(student_id)
);

comment on table tbl_course_enrollment is 'Table to store student enrolled for particular course';


-- tbl_exam
CREATE TABLE tbl_exam(
	course_id number,
	student_id number,
	scored_mark number(3,1),
	constraint fk_cid_exam foreign key(course_id) references tbl_course(course_id),
	constraint fk_sid_exam foreign key(student_id) references tbl_student(student_id)
);


comment on table tbl_exam is 'Table to store student exam information';

-- tbl_payment
CREATE TABLE tbl_payment(
   student_id number,
   paid_date date not null,
   paid_amount number(7,2) not null,
   constraint fk_sid_payment foreign key(student_id) references tbl_student(student_id)
);
```

```
comment on table tbl_payment is 'Table to store student payment information';

-- tbl_fee
CREATE TABLE tbl_fee(
	student_id number,
	due_amount number(7,2) not null,
	due_date date not null,
	constraint fk_sid_fee foreign key(student_id) references tbl_student(student_id)
);

comment on table tbl_fee is 'Table to store students due fee information';

--Trig_fee_deduct
Create or replace trigger trig_fee_deduct
After insert on tbl_payment
For each row
Begin
Update tbl_fee set due_amount=due_amount - :new.paid_amount where student_id=:new.student_id;
End;
/


--Trig_fee_Add
create or replace trigger trig_fee_add
before insert on tbl_course_enrollment
For each row
Declare
PRAGMA AUTONOMOUS_TRANSACTION;
v_fee number(7,2);
v_duration number;
Begin
Select course_fee,course_duration into v_fee,v_duration from tbl_course where course_id=:new.course_id;
merge into tbl_fee f
using (select :new.student_id student_id from dual) s
on(s.student_id=f.student_id)
when matched then
Update set due_amount=due_amount + v_fee
when not matched then
```

```
insert (student_id,due_amount,due_date)
values(:new.student_id,v_fee,sysdate+3);
:new.completion_date:=:new.start_date + v_duration;
commit;
End;
/
```

```
-- seq_teacher
Create sequence seq_teacher start with 1;

-- seq_course
Create sequence seq_course start with 1;

-- seq_student
Create sequence seq_student start with 1;

-- proc_due_date
create or replace procedure proc_due_date
as
cursor cur is select student_id, student_name,due_amount from tbl_fee join tbl_student using(student_id) where due_date >= sysdate;
begin
for c in cur
loop
   dbms_output.put_line('Student Name: ' || c.student_name || ' Due Amount: ' || c.due_amount);
end loop;
end;
/


-- func_result
Create or replace function func_result(v_student_id number,v_course_id number)
Return varchar2
as
v_marks number(3);
v_pass_marks number(2);
```

```
v_result varchar2(4);
Begin
Select scored_mark into v_marks from tbl_exam where course_id=v_course_id and student_id=v_student_id;
Select pass_mark into v_pass_marks from tbl_course where course_id=v_course_id;
If v_marks >=v_pass_marks then
v_result:='PASS';
else
        v_result:='FAIL';
End if;
Return v_result;
End;
/

-- vw_coruse_enrolled
Create view vw_course_enrolled as
select s.student_name,c.course_name,e.start_date
from tbl_student s join tbl_course_enrollment e
using(student_id)
join tbl_course c
using(course_id);

select * from vw_course_enrolled;
```

DML Insert commands.

```
insert into tbl_course(course_id,course_name,course_duration,course_fee,pass_mark)
values(SEQ_COURSE.nextval,'ORACLE',90,7000,60);
insert into tbl_course(course_id,course_name,course_duration,course_fee,pass_mark)
values(SEQ_COURSE.nextval,'JAVA',120,9000,50);
insert into tbl_course(course_id,course_name,course_duration,course_fee,pass_mark)
values(SEQ_COURSE.nextval,'C/C++',45,6000,50);

select * from tbl_course;
commit;



insert into tbl_teacher(teacher_id,teacher_name)
values(SEQ_TEACHER.nextval,'Ram Shrestha');
insert into tbl_teacher(teacher_id,teacher_name)
values(SEQ_TEACHER.nextval,'Laxman Pandey');

select * from tbl_teacher;
commit;


insert into tbl_student(student_id,student_name)
values(SEQ_STUDENT.nextval,'Raju Bajracharya');
insert into tbl_student(student_id,student_name)
values(SEQ_STUDENT.nextval,'Krishna Gurung');

select * from tbl_student;
commit;

insert into tbl_course_enrollment(course_id,teacher_id,student_id,start_date,completion_date)
values(1,1,1,sysdate,null);

insert into tbl_course_enrollment(course_id,teacher_id,student_id,start_date,completion_date)
values(2,1,2,sysdate,null);
select * from tbl_course_enrollment;
commit;
select * from tbl_fee;
```

```
insert into tbl_payment(student_id,paid_date,paid_amount)
values(1,sysdate,1000);
insert into tbl_payment(student_id,paid_date,paid_amount)
values(2,sysdate,2000);

select * from tbl_payment;
select * from tbl_fee;

commit;

insert into tbl_exam(course_id,student_id,scored_mark)
values(1,1,50);
insert into tbl_exam(course_id,student_id,scored_mark)
values(2,2,50);
commit;

select * from tbl_exam;

select func_result(2,2) from dual;
```

SQL Join

```
select s.student_name,c.course_name, e.scored_mark,func_result(e.student_id,e.course_id)
from tbl_student s join tbl_exam e
on(s.student_id=e.student_id)
join tbl_course c
on (e.course_id=c.course_id);
```

Create user commands.

```
create user dilli identified by dilli
Quota unlimited on users;
grant connect, resource to dilli;
grant create view, create synonym, create trigger to dilli;
grant DATAPUMP_EXP_FULL_DATABASE, DATAPUMP_IMP_FULL_DATABASE to dilli;
Grant select any table to dilli;
Grant exp_full_database, imp_full_database to dilli;


Create directory dir_iims as '/home/oracle/iims';
Grant read,write on directory dir_iims to public;


drop table tbl_course_enrollment cascade constraint purge;
drop table tbl_exam cascade constraint purge;
drop table tbl_fee cascade constraint purge;
drop table tbl_payment cascade constraint purge;
drop table tbl_student cascade constraint purge;
drop table tbl_teacher cascade constraint purge;
drop table tbl_course cascade constraint purge;
```