**ORACLE ASSIGNMENT**

**Assignment 1 – Database Creation and Querying Exercises**

**Section 1- Creating Tables:**

**TABLES TO CREATE: department, student, staff, subject, mark – Assume that we create tables and data with the description requirement and discard all the previous version of table structure and data**

**1-department (department\_id number PK, department\_name varchar(30), department\_block\_number number)**

DROP TABLE sys.department;

Create table sys.department (

department\_id number Primary Key,

department\_name varchar(30),

department\_block\_number number);

**2-student (student\_id number, student\_name varchar(30), address varchar (40), city varchar(30), department\_id number FK)**

DROP TABLE sys.student;

Create table sys.student (

student\_id number Primary key,

student\_name varchar(30),

address varchar(40),

city varchar(30),

department\_id number,

CONSTRAINT department\_id\_FK

FOREIGN KEY (department\_id)

REFERENCES sys.department (department\_id) );

**3-staff (staff\_id number, staff\_name varchar(30), department\_id number FK)**

DROP TABLE sys.staff;

Create table sys.staff (

staff\_id number Primary key,

staff\_name varchar(30),

department\_id number,

CONSTRAINT dep\_staff\_FK

FOREIGN KEY (department\_id)

REFERENCES sys.department (department\_id) );

**4- subject (subject\_id number PK, student\_name varchar (30), student\_code varchar(10), staff\_id number FK)**

Table subject has the wrong column description for subject\_name, subject\_code. Double check with the excel spreadsheet data.

DROP TABLE sys.subject;

Create table sys.subject (

subject\_id number Primary key,

subject\_name varchar(30),

subject\_code varchar(10),

staff\_id number,

CONSTRAINT staff\_id\_FK

FOREIGN KEY (staff\_id)

REFERENCES sys.staff (staff\_id));

**5- mark (value number, subject\_id number PK FK, student\_id number PK FK)**

DROP TABLE sys.mark;

Create table sys.mark (

value number,

subject\_id number,

student\_id number,

CONSTRAINT mark\_PK

PRIMARY KEY (subject\_id, student\_id),

CONSTRAINT mark\_sub\_FK

FOREIGN KEY (subject\_id)

REFERENCES sys.subject (subject\_id),

CONSTRAINT mark\_student\_FK

FOREIGN KEY (student\_id)

REFERENCES sys.student (student\_id));

**6-Add a constraint by writing a query to add a not null constraint to the column staff\_name in the staff table.**

Alter table staff

Alter column staff\_name Not null;

**7-Add a column by writing a query named emailid of type varchar (20) to the student table.**

Alter table student

Add column emailid varchar(20);

8. **Modify the size of the type of field emailid on the student table by writing a query to change it to varchar(50);**

Alter table student

Alter column emailid varchar(50);

**9-Remove the emailid column on the student table by writing a query.**

Alter table student

drop column emailid;

**Section 2-**

**Inserting Into Tables**

**10 - Download the following excel sheet in LMS under Documents -> Week 3 -> Oracle hands on:**

* **DEPARTMENT**
* **STAFF**
* **MARK**
* **SUBJECT**
* **STUDENT**

**Download the excel sheets from LMS into local drive**

**11 - Insert those excel sheets to their corresponding tables**

**While importing data from excel spreadsheets to their corresponding tables, there are errors to tables subject and mark. Temporarily disable constraints of table subject & mark to import the data**

Alter table subject

DISABLE constraint subject\_id;

Alter table mark

(DISABLE constraint mark\_pk,

DISABLE Constraint mark\_sub\_fk,

DISABLE Constraint mark\_student\_fk);

**Import data of tables: department, staff, student, subject, mark**

**Enable the constraints of table subject and mark:**

Alter table subject

ENABLE constraint subject\_id;

Alter table mark

(ENABLE constraint mark\_pk,

ENABLE Constraint mark\_sub\_fk,

ENABLE Constraint mark\_student\_fk);

**Section 3-**

**Updating Records**

**12 - Update a record by writing a query to update the subject\_name in the subject table from sales to Computer Science and subject\_code from 1842 to 1919.**

**Update subject**

**Set subject\_name = ‘Computer Science’,**

**subject\_code = ‘1919’**

**where subject\_name = ‘Sales’ and subject\_code = ‘1842’;**

**Section 4-**

**Deleting Records**

**13 - Delete the row from the subject table where subject name is Accounting by writing the appropriate query.**

delete from subject

where subject\_name = 'Accounting';

**Section 5-**

**Basic Selection of Records**

**14 -** **Display the names of the department in the college by writing the appropriate query. Please note that these must be displayed in ascending order.**

Select department\_name

from department

order by department\_name ASC;

**15- Display the names of the departments where departments block number is between 3 and 10 by writing the appropriate query.**

Select department\_name

from department

where department\_block\_number between 3 and 10

order by department\_name ASC;

**16- Display the names of all the students in the college by writing the appropriate query. Please note these must be displayed in ascending order.**

Select student\_name

from student

order by student\_name ASC;

**Section 6-**

**Selecting Single Rows**

**17- Display the names of the students who are from Chicago, Taylor and San Jose. Please note these must be displayed in ascending order of their respective id.**

Select student\_name

From student

Where city in ('Chicago','Taylor','San Jose')

order by student\_id ASC;

**18-** **Writing the correct query, display the address and city of the students table give the alias as Address\_Student.**

Select address as Address\_Student, city

From student;

**19- Display all of the student’s names whose names are of 6 characters in length by writing the correct query.**

Select student\_name

From student

Where length(student\_name) = 6;

**Section 7-**

**Selecting Groups**

**20- Display the blocknumber and number of departments in each block by writing the correct query that is ordered by block id. Make sure it is displayed as count (department\_name)**

select department\_block\_number, department\_id, count(department\_name)

from department

group by department\_block\_number, department\_id

order by department\_block\_number, department\_id;

**21-** **Display the number of students in the college by writing the correct query and give an alias as stud\_count.**

select count(\*) as stud\_count from student;

**Section 8-**

**SQL Joins**

**22-** **Display the names of the department and the student count in each department by writing the correct query. The student count in each department must be in ascending order based on the department name and an alias of student\_count for the student count.**

select department\_name, count(student\_id) as student\_count

from department d left join student s

on d.department\_id = s.department\_id

group by d.department\_name

order by department\_name ASC;

**23-** **Display the Student\_Name from STUDENT and the Subject\_name from SUBJECT where the Subject\_code from SUBJECT is greater than 1600.**

select m.student\_name, m.subject\_name

from mark m left join student s

on m.student\_id = s.student\_id

left join subject sub

on m.subject\_id = sub.subject\_id

where subject\_code > '1600';

**24-** **Display the Stundent\_Name from STUDENTS and the Subject\_name from SUBJECT where the value on MARK table is less 3.**

select student\_name, subject\_name

from mark m left join student s

on m.student\_id = s.student\_id

left join subject sub

on m.subject\_id = sub.subject\_id

where value < 3;

**Section 9 –**

**Selecting Sub-Queries**

**25- Display the block number in which the maximum number of departments is located by writing the correct sub-query**.

select top 1 department\_block\_number

from (

select department\_block\_number, count(department\_id)

from department

group by department\_block\_number

order by count(department\_id) DESC) x;

**26- Display the names of the staff who are not handling any subjects by ascending order using the correct sub-query.**

select staff\_name

from staff

where staff\_id not IN

(select staff\_id from subject);

**Section 10-**

**Functions**

**27-** **Write a function that takes department\_id as the input and returns the department\_name.**

**Use the function name below:**

**Function name: find\_dept\_name**

create or replace function

find\_dept\_name (in\_dept\_id number)

return varchar2

as

l\_dname department.department\_name%type;

begin

select department\_name into l\_dname

from department

where department\_id = in\_dept\_id;

return l\_dname;

end;

exec dbms\_output.put\_line (find\_dept\_name (10));

**28-** **Write a function that takes department id as the input and returns the block number.**

**Use the function name below:**

**Function name:  find\_dept\_block**

create or replace function

find\_dept\_block (in\_dept\_id number)

return number

as

l\_dblock department.department\_block\_number%type;

begin

select department\_block\_number into l\_dblock

from department

where department\_id = in\_dept\_id;

return l\_dblock;

end;

exec dbms\_output.put\_line (find\_dept\_block (10));

**29- Write a function that takes the staff id as the input and returns the staff name.**

**Use the function name below:**

**Function name:  find\_staff\_name**

create or replace function

find\_staff\_name (in\_staff\_id number)

return varchar2

as

l\_staffname staff.staff\_name%type;

begin

select staff\_name into l\_staffname

from staff

where staff\_id = in\_staff\_id;

return l\_staffname;

end;

exec dbms\_output.put\_line (find\_staff\_name (100));

**Section 11-**

**Triggers**

**30- Create a trigger with the name 'trigger\_department\_af\_update' which will display “DEPARTMENTS table has been updated” after an attempt to update the DEPARTMENTS has been made.**

**Trigger name: trigger\_department\_af\_update**

create or replace trigger trigger\_department\_af\_update

after update

on department

begin

if updating then

dbms\_output.put\_line('DEPARTMENTS table has been updated');

end if;

end;

Update department

Set department\_name = “IRM”

Where department\_id = 60;

**31- Create a trigger with the name ‘trigger\_department\_bf\_delete’ which will display “A row has been deleted from DEPARTMENT” before an attempt to delete a row is execute on DEPARTMENT.**

**Trigger name :   trigger\_department\_bf\_delete**

create or replace trigger trigger\_department\_bf\_delete

before delete

on department

begin

if deleting then

dbms\_output.put\_line(‘A row has been deleted from DEPARTMENT’);

end if;

end;

delete from department where department\_id = 220;

**Section 12-**

**Views and Index**

**32- Create an Index command that will reference all of the students names containing the letter ‘b’ on the Student table.**

CREATE INDEX stud\_name\_index

ON Student (student\_name)

Where student\_name like “%b%’;

**33- Create a view from the staff table that will display staff names.**

CREATE VIEW staff\_view AS

SELECT staff\_names

FROM staff;

**Section 13-**

**Cursors**

**36- Declare an explicit cursor using the STUDENT table to select of column. Fetch the rows using a loop and display each data retrieved.**

Declare

cursor s1 is

Select \* from student;

s\_rec s1%rowtype;

Begin

Open s1;

Loop

Fetch s1 into s\_rec;

dbms\_output.put\_line (‘student: ‘||s-rec);

Exit when s1%not found;

If s1%isopen then

dbms\_output.put\_line (‘Cursor opened ‘||s1%rowcount);

End if;

End loop;

Close s1;

If not s1%isopen then

dbms\_output.put\_line (‘Cursor closed’);

End if;

End;

**Section 14-**

**PACKAGES AND PROCEDURES**

**37 - Define a package give it the name of College. The definition should include a procedure give it the name of select\_departments. It should also include a function that takes a argument of type number and returns a variable of type VARCHAR, give it the name of select\_student.**

Create or replace package College

as

Procedure select\_departments (s\_dept\_name in varchar(30))

As

Begin

Select department\_name into s\_dept\_name

from department;

dbms\_output.put\_line (‘Department name: ‘||s\_dept\_name);

end select\_departments;

function select\_student (s\_stud\_id in number)

return varchar

as

s\_stud\_name student.student\_name%type;

begin

select student\_name into s\_stud\_name

from student

where student\_id = s\_stud\_id;

return s\_stud\_name;

end select\_student;

end College;

**38 - Implement the body of College department. the select\_departments procedure should display all column from DEPARTMENT. The select\_student function should take the id of the student and return that student name.**

Create or replace package College\_department

as

Procedure select\_departments

As

Begin

Declare

cursor d1 is

select \* from department;

d\_rec d1%rowtype;

Begin

Open d1;

Loop

Fetch d1 into d\_rec;

Exit when d1%notfound;

dep\_id := d\_rec.department\_id;

dep\_name := d\_rec.department\_name;

dep\_block\_num := d\_rec.department\_block\_number;

End loop;

Close d1;

End;

end select\_departments;

function select\_student (s\_stud\_id in number)

return varchar

as

s\_stud\_name student.student\_name%type;

begin

select student\_name into s\_stud\_name

from student

where student\_id = s\_stud\_id;

return s\_stud\_name;

end select\_student;

end College\_department;