

# Database Management System Lab

## I Sem MCA

MCA-4141

2.5 hr/ week    1 Credit

---

Archana.H,  
Assistant Professor –Senior Scale,  
DSCA,MIT, Manipal

# Rules & Regulations

---

- Seating Arrangements
- Login Book Entry
- Mobiles in Class & Lab – Switch off mode.
- Lab Attendance- 75%
  - Totally 12 Labs including the Final Lab exam.
- Save your files and take backups regularly.

# ...Rules & Regulations

---

- Regular Internal lab evaluation is done as per concepts covered and convenience of time.
- Managing missing Lab.
  - Work extra time to finish missed lab and submit Lab Records.
  - HOD permission( genuine case) is required to claim missing Lab evaluation Marks.
  - Inform faculty well before –Personally/by email regarding Absence.

# Course Outcomes

---

## Course Outcomes (COs)

*At the end of this course, the student should be able to:*

		No. of Contact Hours
C01:	Create and modify database objects	6
C02:	Manipulate the data in the database	3
C03:	Design queries to retrieve the data from the database	9
C04:	Perform database operations by integrating procedural language constructs	9
C05:	Design stored programs	6

# Course Plan

Week	Topics to be covered	
1-2	SQL Basics – CREATE, ALTER, DROP	
3	Populate and manipulate the database using INSERT, UPDATE, DELETE	
4-6	SQL Simple and Advanced Queries - Quiz	
7	MID TERM EXAM , PL/SQL	
8	Cursors	
9	Exception Handling	
10	Triggers	
11	Procedures, Functions, Packages - Quiz	
12	END TERM PRACTICAL EXAMINATION	

# Lab Evaluation

---

## Total Internal Evaluation – 60 Marks

2 Evaluations \* 20 Marks + Midterm Exam – 20 Marks – 1 Hour (Write-Up & Execution)

Evaluations: Observation Book - 6 Marks	}	<b>20 Marks</b>
Execution marks - 7 Marks		
Quiz - 7 Marks		

## Lab End Semester Exam – 40 Marks

Pattern: 2 Questions Covering SQL , PL/SQL concepts - 3 Hours , Write-up & Execution

# References

---

1. Ivan Bayross, “SQL, PL/SQL-The Programming Language of ORACLE”, 4<sup>th</sup> Edition, BPB Publications
2. Satish Asnani, “Oracle Database 11g”, PHI, 2010.
3. Scott Urman, “ORACLE – PL/SQL Programming”, Oracle Press.

# Working in Lab

---

- Switch on the Computer
- Select Windows Booting (Default)
  - User Name: mca
  - Password: mca
- Create a Folder: D:\OraclePrg\240970xxx
  - Create Subfolders – PRACTICE, WEEK1, WEEK2, ...



# Starting Oracle SQL

---

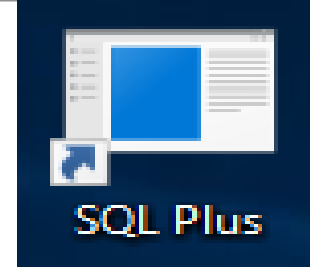
Double Click - SQL icon present on the Desktop.

Enter user-name: **mcaxxx@mcaorcl**

Enter password: **mcaxxx**

You will get the following SQL Prompt

**SQL>**



# CREATE TABLE

Create a Table ***Student*** to store information about *Student ID, Name, Department Name* (where he has joined for course), *Total Credits*. Assume data type and size as below-

```
create table student (  
    ID          varchar(5),  
    name        varchar(20),  
    dept_name   varchar(20),  
    tot_cred    numeric(3,0) );
```

Simple Syntax:

```
CREATE TABLE table_name (  
    Column_name1 DataType(size),  
    Column_name2 DataType(size),  
    ..... ,  
    Column_nameN DataType(size), );
```

# How to View Structure of a Table

---

To find out- column name, datatype, size and some constraints.

```
SQL> DESCRIBE Student;
```

**Syntax:**

```
DESCRIBE table_name;
```

# How to insert Record into a Table

---

Add new rows to a table by using the `INSERT` statement:

```
sql> INSERT INTO Student  
VALUES( 1001, 'Rajesh','Comp.Sc',35);
```

- With this syntax, only one row is inserted at a time.

**Simple Syntax:**

```
INSERT INTO table VALUES(value1 , value2...);
```

# How to insert Record into a Table

---

Add new rows to a table by using the `INSERT` statement:

```
SQL> INSERT INTO Student(ID,Name,dept_name,tot_cred)
```

```
VALUES( 1002, 'Raj',Physics',35);
```

```
SQL> INSERT INTO Student values('&ID','&Name','&dept_name','&tot_cred);
```

**Another way to save and execute multiple Insert commands into Notepad**

```
SQL> INSERT ALL
```

```
    INTO Student VALUES ( '1003', 'Ajith', 'Info.Sc', 36)
```

```
    INTO Student VALUES ( '1004', 'Aakanksha', 'Data Sc.', 26)
```

```
SELECT * FROM dual;
```

# Save, Edit & Execute SQL commands

---

## Save

- Open a **Notepad** From Windows.
- Type the command in the Notepad.

**Example:** INSERT INTO Student VALUES ( 1006, 'Raj', 'Comp.Sc', 30);

INSERT INTO Student VALUES( 1007, 'Ravi',NULL,28);

- Save the notepad file with name **stud\_ins.sql** in the folder D:\OraclePrg\220970xxx \Practice

# Save, Edit & Execute SQL commands

---

## Edit

- Open a **Notepad** From Windows.
- Open the file **stud\_ins.sql** from the folder D:\OraclePrg\220970xxx \Practice
- Modify the data in the SQL

**Example:** INSERT INTO Student VALUES ( 1003, 'Ajith', 'Info.Sc', 36);

- Save the notepad file with name **stud\_ins.sql** or **different name** in the folder D:\OraclePrg\220970xxx \Practice

# Save, Edit & Execute SQL commands

---

## Execute

- Go to SQL Prompt and Type as below-
- **SQL> @ D:\OraclePrg\220970xxx \Practice\ stud\_ins.sql**
- Press Enter. **OR**
- **SQL> START D:\OraclePrg\220970xxx \Practice\ stud\_ins.sql**
- Press Enter.



# Retrieving Data Stored

---

Display the Records stored in the table Student.

```
SQL> SELECT * FROM student;
```

**Syntax:** SELECT \* FROM *table\_name*;

```
SQL> SELECT ID,DEPT_NAME FROM Student;
```

**Syntax:** SELECT col\_name1,col\_name2  
FROM *table\_name*;

```
SQL> SELECT Id, Dept_name FROM Student  
WHERE tot_cred<30;
```

**Syntax:** SELECT col\_name1,col\_name2  
FROM *table\_name*  
WHERE col\_namex\_condition;

# Using Spool

---

- To take backup spool file to be created.

```
SQL> spool D:\OraclePrg\220970xxx \Practice\lab1.txt;
```

```
SQL> desc student;
```

```
SQL> Insert into Student Values(.....);
```

```
-----
```

```
-----
```

```
-----
```

```
SQL> spool off;
```

# Using Spool

---

- To add the content to the same spool file

```
SQL> spool D:\OraclePrg\220970xxx \Practice\lab1.txt append;
```

```
SQL>SELECT * FROM STUDENT;
```

```
SQL> spool off;
```

# SQL Constraints

---

SQL Constraints are rules used to limit the type of data that can go into a table, to maintain the accuracy and integrity of the data inside table.

Constraints can be divided into following two types,

**Column level constraints** : limits only column data

**Table level constraints** : limits whole table data

# SQL Constraints

---

Constraints are used to make sure that the integrity of data is maintained in the database. Following are the most used constraints that can be applied to a table.

- NOT NULL
- UNIQUE
- PRIMARY KEY
- FOREIGN KEY
- CHECK
- DEFAULT

# Exercise

---

Create the following tables

TEMP

Attributes	DataType	Size	Constraint
<u>EMPCODE</u>	Number	3	Primary Key
NAME	Varchar	15	
Salary	Number	7,1	
Deptno	Varchar	3	F.Key References TDEPT

TDEPT

Attributes	DataType	Size	Constraint
<u>Dno</u>	Varchar	3	Primary Key
Dname	Varchar	15	
Zone	Varchar	10	

# CREATE TABLE EMP,DEPT

---

```
create table DEPT (  
    DNO          varchar2(3) Primary Key,  
    Dname        varchar2(15) NOT NULL,  
    Zone         varchar2(10) Default 'North');
```

```
create table EMP(  
    EMPCODE      Number(3) Primary Key,  
    Name         varchar2(15),  
  
    DeptNo       varchar2(3) references Dept(Dno),  
    Salary       numeric(7,1));
```

# Exercise

Insert following records into the tables.

**EMP**

F.key

<u>EMPCODE</u>	NAME	Deptno	Salary
100	RAJESH	D1	100000
101	RAVI	D2	120000
102	VIJAY	D1	100000
108	AJAY	D3	140000
110	BHASKAR	D2	120000

**DEPT**

<u>Dno</u>	Dname	Zone
D1	Operations	North
D2	Finance	West
D3	Sales	South
	Human	
D5	Resource	WEST

Simple Syntax:

```
INSERT INTO table VALUES(value1 , value2...);
```



Create the table MARKS (RegNo, Mark1, Mark2,Total, Grade)

Impose the constraints that-

---

*all marks must be in the range 0 to 100*

*Grade must be – A+,A,B,C,D,E,F*

Create table Marks (regno number(3),

Mark1 number(3) check(mark1>=0 AND mark1<=100),

mark2 number(3) check (mark2 between 0 AND 100),

Total number(3),

Grade char(2) Check( Grade IN ('A+', 'A', 'B', 'C', 'D', 'E', 'F'));

# Try yourself

---

What happens if invalid data entered to some columns in the MARKS table ?

```
insert into Marks values(100,108,78,null, 'A');
```

We get constraint Violation message as ORA-xxxxxx ...some text

ORA-02290: check constraint (MCAxxx.SYS\_C0010461) violated

Similarly try

```
insert into Marks values(100,98,78,null, 'Z');
```

Use **CONSTRAINT *name\_of\_constraint*** along with constraint

---

Create table Marks (regno number(3), Mark1 number(3) **CONSTRAINT Range\_Mark1** check(mark1>=0 AND mark1<=100), mark2 number(3) **CONSTRAINT Range\_Mark2** check (mark2 between 0 AND 100), Total number(3), Grade char(2) **CONSTRAINT Range\_GRADE** Check( Grade IN ('A+', 'A', 'B', 'C', 'D', 'E', 'F')));

Try inserting invalid records again and check the error message displayed



# Create the table COURSE (CourseID, Sem, Credits).

Impose the constraints that—

---

Course ID must start with MCA— example MCA2262 etc.

Sem must be 1 to 8 only

```
Create table Course (CourseId Varchar (10) CONSTRAINT  
CourseId_Prefix Check( CourseId LIKE 'MCA%'), Sem  
number(1) CONSTRAINT valid_semester Check(Sem  
between 1 AND 8), Credits Number(2)) ;
```

Try inserting —

Insert into Course values('ABC123',4,36):

is a wildcard character which ignores 1 character(accepts exactly one any character).

---

Example:

Create a table PRODUCT(ProdID, Name, Price) to store products of JOI company. Product numbers must be of the form JOI- followed by any 5 characters(use 5 under score character).

```
CREATE TABLE PRODUCT(ProdID varchar2(9) PRIMARY KEY CHECK  
(ProdID LIKE 'JOI-____'),  
Name varchar(10),  
Price Number(5));
```

Create the table STUD (Regno, Name, Phone, email).

Impose the constraints that-

---

Primary Key on RegNo,

Phone is Unique

Email is Unique

```
CREATE TABLE STUD(Regno Number(3) PRIMARY KEY,  
Name varchar2(10), Phone number(10) UNIQUE,  
Email varchar2(20) CONSTRAINT UNQ_Email UNIQUE);
```

*Note: Unique columns do not accept duplicate values , but can accept null values*

# Unique

---

- Create table DEPT ( Dno Varchar2(3) UNIQUE ,Dname varchar2(10));
- Create table EMP( Empno Number(3) Primary key, Name varchar2(10), Deptno varchar2(3) References Dept(Dno));

# DEFAULT

The DEFAULT constraint is used to provide a default value for a column.

---

**Example:**

```
CREATE TABLE Persons (  
    ID Number(3) NOT NULL,  
    LastName varchar(10) NOT NULL, FirstName varchar(10),  
    Age Number(2), City varchar(15) DEFAULT 'Manipal' );
```

**Example:**

```
INSERT INTO Persons(ID,Lastname) VALUES(100,'AAA');
```

Inserts value to ID=100 , Lastname=AAA , **FirstName= NULL** , **Age=NULL** & City takes value **Manipal** automatically assigned though City value is not specified in the INSERT command.



# ..FOREIGN KEY- UPDATE/DELETE Restrictions

EMP- DEPTNO Foreign Key			DEPARTMENT - DNO Primary Key		
EMPNO	NAME	DEPTNO	DNO	NAME	BUDGET
100	Raj	D1	D1	MCA	128999
101	Krishna	D2	D2	CompSc	124456
102	Manoj	D1	D3	Mech	123562
103	Ravi	D3			
104	Shriivas				

- Similarly,
- UPDATE EMP SET DEPTNO='D5' WHERE EMPNO=100;  
is **Rejected**.
- UPDATE EMP SET DEPTNO='D3' WHERE EMPNO=100;  
is **Accepted**.
- DELETE FROM DEPARTMENT WHERE DNO= 'D1';  
is **Rejected**

To execute above DELETE command, execute in following Order

- 1<sup>st</sup> Delete from **Child** Table(EMP) and then 2<sup>nd</sup> Delete from **Parent**(DEPARTMENT)
  - This Deletion process can be **automated** by using Clause **ON DELETE CASCADE / ON DELETE SET NULL** while creating Child Table
- Similarly Altering Structure of DNO or Dropping DNO is **Rejected**.

# ..FOREIGN KEY- ON DELETE CASCADE/ON DELETE SET NULL

- A foreign key with cascade delete means that if a record in the parent table is deleted, then the corresponding records in the child table will automatically be deleted. This is called a cascade delete in Oracle.

- **Parent(Master) Table:**

- **CREATE TABLE Department ( Dno varchar(2) PRIMARY KEY, Name varchar(10), Budget Number(9) );**

- **Child(Detail) Table**

- **CREATE TABLE Emp ( Empno number(3) PRIMARY KEY, Name varchar(10), Deptno varchar(2) REFERENCES Department ON DELETE CASCADE );**

**Any Delete operation on the table Department(Parent) first deletes dependent records in the EMP(child) table automatically. Thus Delete operation restriction on Foreign key constraint is get resolved automatically.**

# ..FOREIGN KEY- ON DELETE CASCADE/ON DELETE SET NULL

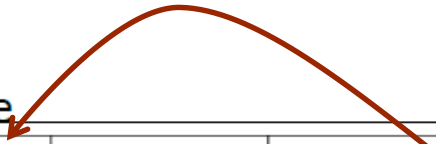
- A foreign key with “**ON DELETE SET NULL** ” means that if a record in the parent table is deleted, then the corresponding records in the child table will have the **foreign key fields set to null**. The records in the child table will **not be deleted**.
- **Parent(Master) Table:**
  - **CREATE TABLE Department**(Dno varchar(2) PRIMARY KEY, Name varchar(10), Budget Number(9));
- **Child(Detail) Table**
  - **CREATE TABLE Emp**( Empno number(3) PRIMARY KEY, Name varchar(10), Deptno varchar(2) **REFERENCES Department ON DELETE SET NULL** );

## ..FOREIGN KEY- ON DELETE CASCADE/ON DELETE SET NULL

- When a record is deleted from Department(Parent) table it will not delete dependent records in the EMP(child) table instead puts NULL values to corresponding foreign key column/s.
- Thus removes dependency of corresponding records in the child table on table records being deleted in the Parent table.
- Thus Delete operation restriction on Foreign key is get resolved automatically.

# ..FOREIGN KEY - Recursive Relationship

EMP table



EMPNO	ENAME	MGRNO
100		103
101		100
103		104
104		104
105		

MGRNO is the Employee number of Manger. Employee with EMPno 103 is the Manger for Employee with Empno 100. Therefore MGRNO is Foreign Key Referencing EMPNO

## Example:

```
CREATE TABLE EMP ( Empno number(3) PRIMARY KEY, Ename  
Varchar2(10), MGRNO number(3) );
```

Note: Referential Integrity constraint on MGRNO can be defined using  
**Alter Table** command **after creating EMP table**

**OR**

```
CREATE TABLE EMP( Empno number(3) PRIMARY KEY, Ename  
Varchar2(10), MGRNO number(3) REFERENCES EMP );
```

# The FLASHBACK TABLE Statement

```
DROP TABLE emp2;
```

```
SELECT original_name, operation,  
droptime, FROM recyclebin;
```

```
FLASHBACK TABLE emp2 TO BEFORE DROP;
```

DROP TABLE ...PURGE

DROP TABLE dept80 PURGE;

# Data Dictionary for Constraints

- **SELECT CONSTRAINT\_NAME,CONSTRAINT\_TYPE, TABLE\_NAME  
FROM USER\_CONSTRAINTS;**
- **SELECT CONSTRAINT\_NAME,CONSTRAINT\_TYPE  
FROM USER\_CONSTRAINTS  
WHERE TABLE\_NAME='EMP';**
- **SELECT CONSTRAINT\_NAME,COLUMN\_NAME  
FROM USER\_CONS\_COLUMNS  
WHERE TABLE\_NAME='EMP';**