**DATA BASE & MANAGEMENT SYSTEM**

**(CSX-224)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**Dr. B R AMBEDKAR NATIONAL INSTITUTE OF TECHNOLOGY**

**JALANDHAR – 144011, PUNJAB (INDIA)**

**Submitted To: Submitted by:**

**Ms.** **KIRTI**

**Assistant Professor Chinmay Choudhary**

**RollNumber:15103032**

**Dept. of CSE**

1

**INTRODUCTION TO SOFTWARE(ORACLE 10G EXPRESS EDITION)**

1. **FEATURES:-**

Over the last 27 years, Oracle has made tremendous improvements in its core database product. Now, that product is not only the world's most reliable and performant database, but also part of a complete software infrastructure for enterprise computing. With each new release comes a sometimes dizzying display of new capabilities and features, sometimes leaving developers, IT managers, and even seasoned DBAs wondering which new features will benefit them most.

With the introduction of Oracle Database 10g, DBAs will have in their hands one of the most profound new releases ever from Oracle. So, DBAs who take the time to understand the proper application of new Oracle technology to their everyday jobs will enjoy many time-saving, and ultimately, money-saving new capabilities.

Oracle Database 10g offers many new tools that help DBAs work more efficiently (and perhaps more enjoyably), freeing them for more strategic, creative endeavorsnot to mention their nights and weekends. Oracle Database 10g really is that big of a deal for DBAs.

1. **ADVANTAGES:-**
2. **Grouping Transactions:-**

The ability to group several transactions into the same batch for processing sets Oracle apart from its competitors. Microsoft's version of SQL is limited to executing each transaction individually in a sequential order. This gives Oracle SQL far greater scalability over sequential versions of SQL, which are limited to only expanding vertically. Vertical expansion can be significantly more expensive than horizontal expansion. Expanding vertically requires the addition of server memory, hard drive capacity and ventilation equipment. Oracle can expand horizontally by clustering transactions for more efficient processing.

1. **Improved Performance:-**

In addition to batch processing of transactions, Oracle offers other methods of improving your database's performance. You can use multiple servers to work on the same database with the Real Application Cluster feature. This can significantly increase your processing power for only the price of an additional server. Oracle SQL also gives you more options to fine-tune the operation of your database to suit the capabilities of your server.

1. **Versatility:-**

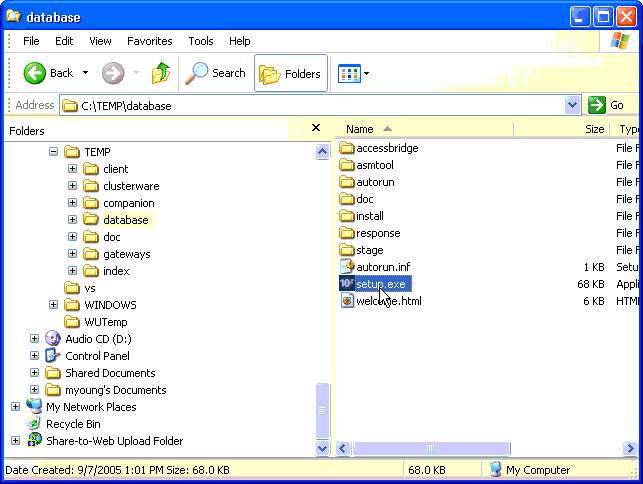
Oracle SQL gives you the flexibility of choosing to run your database in any operating system. Dedicated languages are only compatible with operating systems from the same manufacturer. For example, you can only run Microsoft SQL Server on a Windows-based machine. In comparison, you can install Oracle SQL on a Unix server and benefit from the reliability of Unix while keeping the standardization of SQL. Unix is less vulnerable to many common computer viruses, which keeps your information secure. Oracle SQL is also backward-compatible, so you have the option of upgrading in the future without losing any data.

1. **INSTALLATION STEPS**

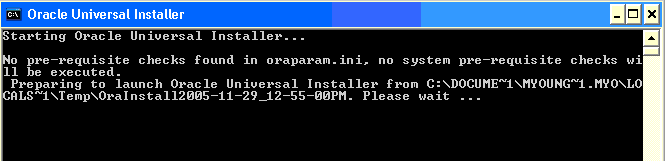
**Installing Oracle Database 10g on Windows**

To install the Oracle software, you must use the Oracle Universal installer.

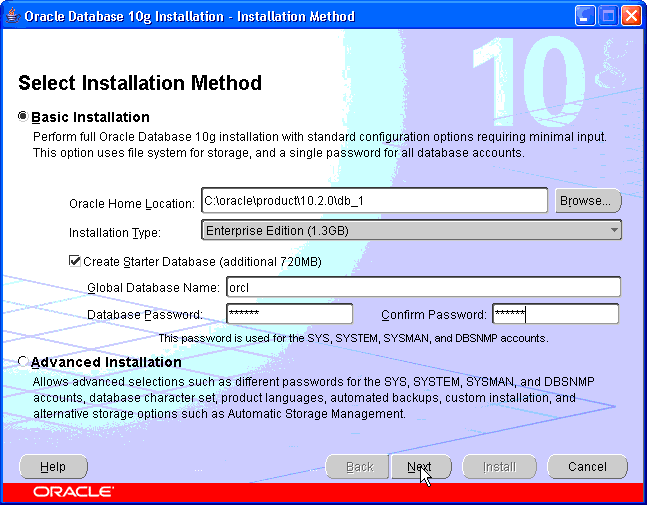
1. For this installation you need either the DVDs or a downloaded version of the DVDs. In this tutorial, you install from the downloaded version. From the directory where the DVD files were unzipped, double-click setup.exe.



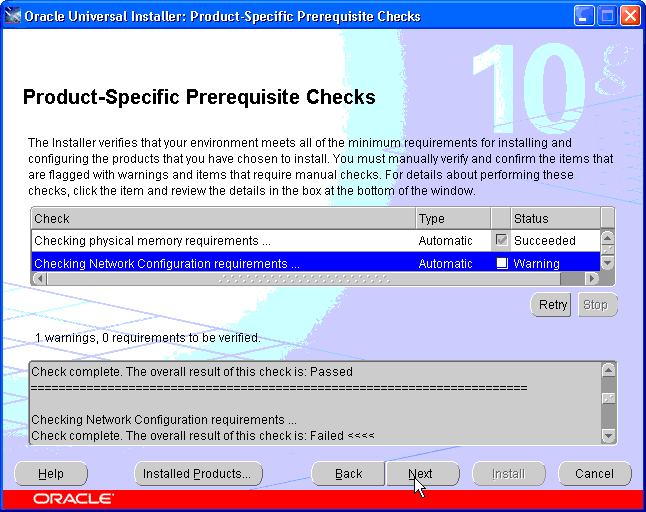
1. The Oracle Universal Installer starts.



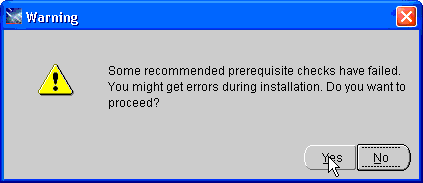
1. You will perform a basic installation with a starter database. Enter orcl for the Global Database Name and oracle for the Database Password and Confirm Password. Then click **Next**.



1. The installer now verifies that the system meets all the minimum requirements for installing and configuring the chosen product. Please correct any reported errors (warnings are OK) before continuing. When the check successfully completes (with or without warnings), click **Next**.



1. If you received any warnings, you can proceed. Click **Yes**.



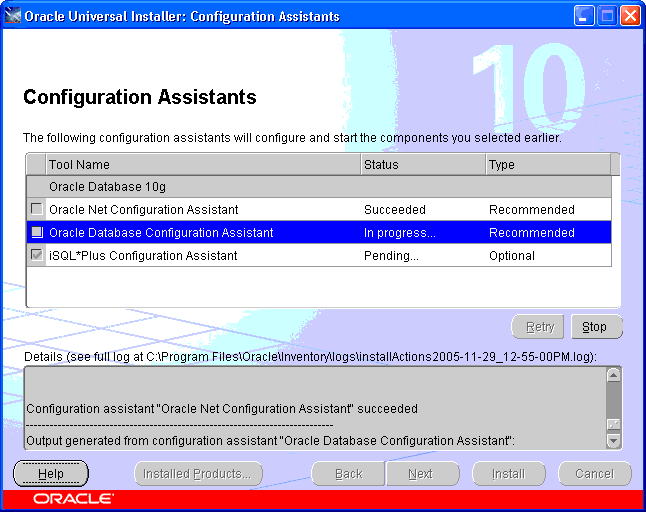
1. Review the Summary window to verify what is to be installed. Then, click **Install**

<http://www.oracle.com/webfolder/technetwork/tutorials/obe/db/10g/r2/prod/install/wininst/images/install06.gif>

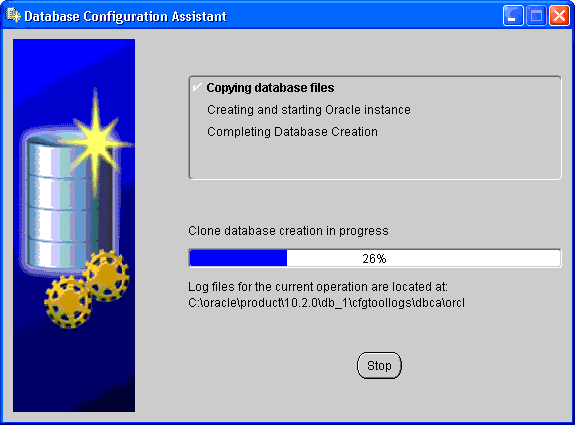
1. The progress window appears

<http://www.oracle.com/webfolder/technetwork/tutorials/obe/db/10g/r2/prod/install/wininst/images/install07.gif>

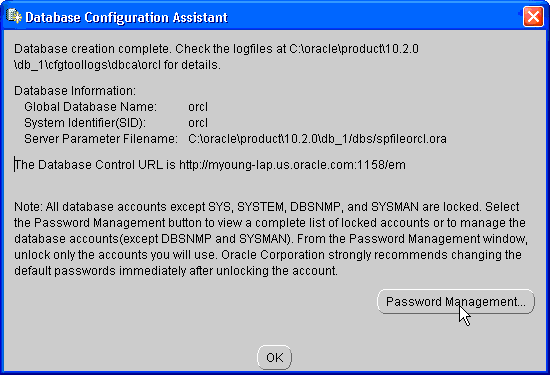
1. The Configuration Assistants window appears



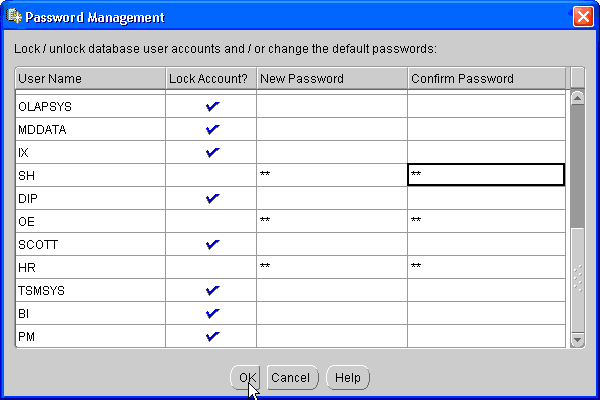
1. Your database is now being created



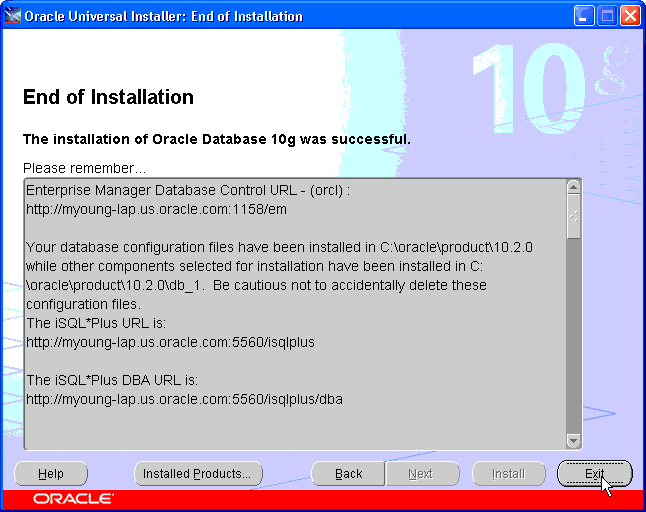
1. When the database has been created, you can unlock the users you want to use. Click **Password Management**



1. Unlock **SH**, **OE** and **HR** users by clicking on the check mark in the Lock Account? column. Enter the same name as the user in the New Password and Confirm Password fields. For example, to unlock **SH** user, enter SH in the New Password and Confirm Password fields. Then, click **OK.**



1. Click **Exit**.

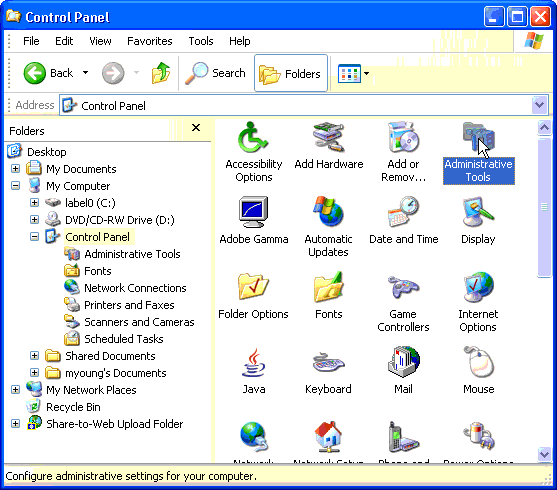


**Post Installation Tasks**

For additional functionality within DB Control, functionality such as starting and stopping the database, an operating system user must belong to the ORA\_DBA group. An existing user can be used or a new user can be created. This topic shows you how to create the operating system user oracle and make the user part of the ORA\_DBA group:

**Note**: The following steps outlined were done in a Windows XP environment. The steps will be slightly different in a Windows NT or 2000 environment.

|  |  |
| --- | --- |
| **1** | **Start** **> Settings >** **Control Panel**. Select **Administrative Tools**. |

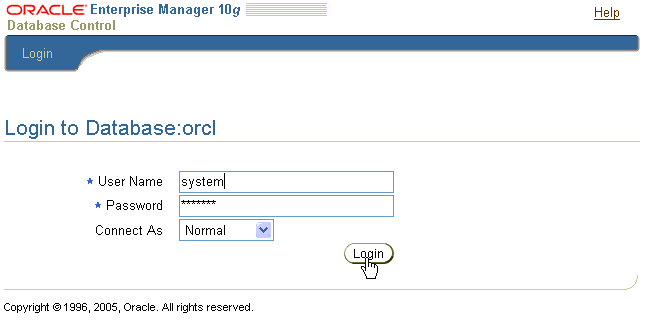


2**.Select** **Computer Management**

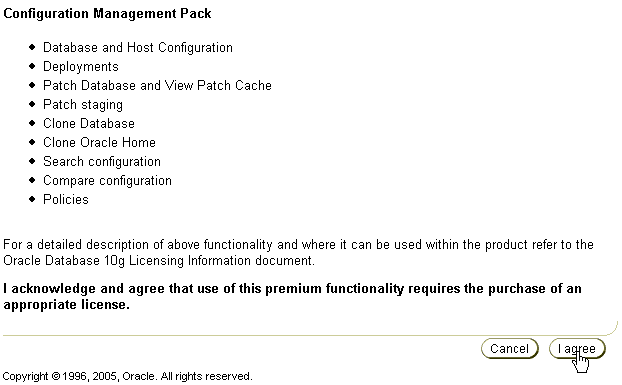
**Testing Your Installation**

To test that your installation completed successfully, perform the following:

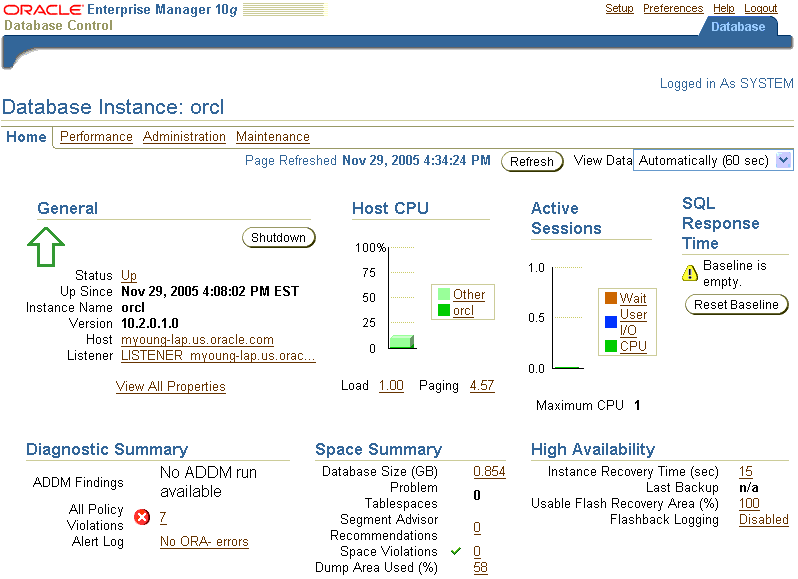
|  |
| --- |
| When you click exit from the previous section, a browser opens and the Database Control URL is already launched. If this does not happen, open a browser and enter the following URL:  1.**Error! Hyperlink reference not valid.**  Enter system as the User Name and oracle as the Password and click **Login**. |



**2.** The Licensing window appears. Scroll to the bottom and click **I agree.**



**3.** The Database Control Home Page appears. Your Installation was successful.



11

**LAB EXEPERIMENT NO.1**

**AIM:**TO STUDY THE BASIC DDL & DML COMMNADS.

DATA DEFINITION LANGUAGE

A data definition language (DDL) is a computer language used to create and modify the structure of database objects in a database. These database objects include views, schemas, tables, indexes, etc.   
This term is also known as data description language in some contexts, as it describes the fields and records in a database table.

Various commands of DDL are as follows:

**CREATE TABLE COMMAND:**

This command is used to creates a new table, a view of a table, or other object in database.

**SYNTAX:** CREATE TABLE tablename (Column 1 datatype(size),Column 2 datatype(size). . . . . .);

**EXAMPLE:**

create table cse\_students (name char(25),rollno int ,branch char(25), contact int)

Capture.PNG

**DROP COMMAND:**

This command willdeletes an entire table, a view of a table or other object in the database.

**SYNTAX** :for deleting whole table:DROP TABLE tablename;

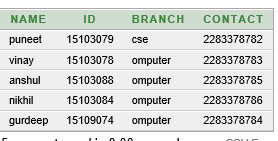
**EXAMPLE**: drop table nit\_students

table not present.PNG

**SYNTAX** :for deleting particular column in a table:

ALTER TABLE tablename DROP COLUMN columnname;

**EXAMPLE:** alter table students drop column address



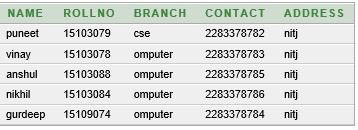
**RENAME COMMAND:**

This command is used to change the name of table or to change the name of a particular column.

**SYNTAX**: for renaming table:

RENAME old\_tablename to new\_tablename;

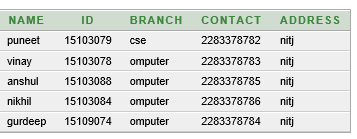
**EXAMPLE:** rename cse\_students to students



**SYNTAX** :for renaming column:

ALTER TABLE tablenameRENAME COLUMNold\_columnname to new\_columnname;

**EXAMPLE:** alter table students rename column rollno to id



**ALTER Table Command:**

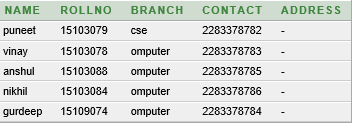
This command is used tomodifies an existing database object, such as a table.

ADDING New Columns:

**SYNTAX:**ALTER TABLE table\_nameADD(Column\_name1 datatype(size), . . . . . . .);

**EXAMPLE:** alter table cse\_students add(address varchar(30))

select \* from cse\_students



**MODIFYING Existing Table:**

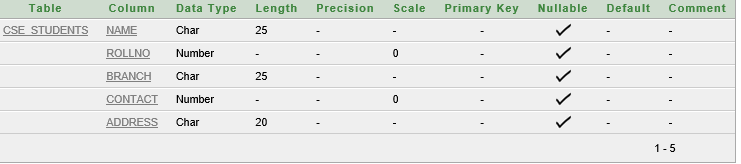
This command is used to change the datatype and size of the column of table.

**SYNTAX:**ALTER TABLE table\_name MODIFY (column\_name1 newdatatype (new\_size);

**EXAMPLE**: alter table cse\_students modify (address char(20))

select \* from cse\_students

desc cse\_students



**TRUNCATE Command:**

This command is used to remove all records from a table, including all spaces allocated for the records are removed.

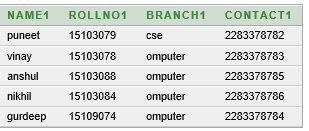
**SYNTAX:**TRUNCATE TABLE table\_name;

**EXAMPLE:**Creating table from another Table:

**SYNYAX:**CREATE TABLE newTable\_name (column\_nmae1, column\_name2) AS SELECT column\_name1,column\_name2 from table\_name;

**EXAMPLE:** create table nit\_students (name1,rollno1,branch1,contact1) as select name,rollno,branch,contact from cse\_students

select \* from nit\_students



**DATA MANIPULATION LANGUAGE**

A data manipulation language is a language that enables users to access or manipulate data as organized by the appropriate data model. The type of access are:

Retrieval of information stored in the database

Insertion of new information into the database

Deletion of information from the database

Modification of information stored in the database

Various commands under DML are:

**INSERT:** This command is used to insert data into rows of table.

**SYNTAX:** INSERT INTO table\_name(column\_name1,column\_name2,. . . .) VALUES ( Expresssion1,Expression2,. . . .)

**EXAMPLE:** insert into cse\_students (name,rollno ,branch, contact) values ('puneet',15103079,'cse',2283378782)

select\* from cse\_students

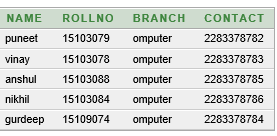
select without condition.PNG

**UPDATE Command:** The UPDATE command is used to change or modify data values in a table and UPDATE command can Update all the rows from the table or a set of rows from the table.

**SYNTAX**: for updating all rows of column:

UPDATE table\_name SET column\_name=Expression;

**EXAMPLE:** update cse\_students set branch='omputer'



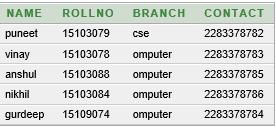
**SYNTAX for updating particular row:**

UPDATE table\_name SET column\_name=Expression WHERE condition(column\_name)=Expression;

**EXAMPLE:**

update cse\_students set branch='cse' where rollno=15103079

select \* from cse\_students



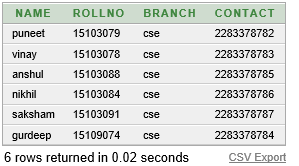
**SELECT Command:**

Once data has been inserted into a table, the next most logical operation would be to view what has been inserted. The SELECT SQL verb is used to achieve this.

**SYNTAX** : to view whole table:

SELECT \* FROM table\_name;

**EXAMPLE:** select \* from nit\_students



**SYNTAX**: to view particular rows:

SELECT \* FROM table\_name WHERE column\_name=Expression;

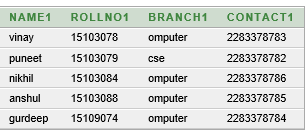
**EXAMPLE:** select \* from cse\_students where name='puneet'

select with condition.PNG

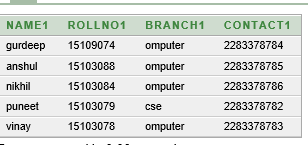
**Sorting Data**:The Rows retrieved from the table will be sorted in either**Ascending(ASC)**or**Descending(DESC)**order depending on the condition specified in select statement, the Keyword has used **ORDER BY.**

**SYNTAX:** SELECT \* FROM table\_name ORDER BY column\_name ASC/DESC;

**EXAMPLE:** select \* from nit\_students order by rollno1 asc



select \* from nit\_students order by rollno1 desc



**DELETE Command:**The DELETE command can remove all the rows from the table or a set of rows from the table.

**SYNTAX** :for deleting all rows from table:

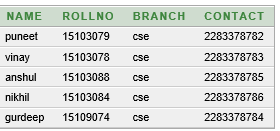
DELETE FROM table\_name;

**EXAMPLE:** delete from cse\_students

**SYNTAX**: for deleting particular row:

DELETE FROM table\_name WHERE column\_name=Expression;

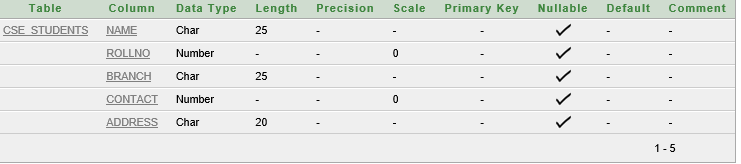
**EXAMPLE:** delete from cse\_students where rollno=15103091



**DESCRIBE:**This is an Oracle **command** used to **describe** the structure of objects within a given database.

**SYNTAX:** DESC table\_name;

**EXAMPLE:** desc cse\_students



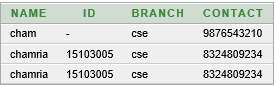
**LAB EXEPERIMENT NO.2**

**AIM:** use of more database commands

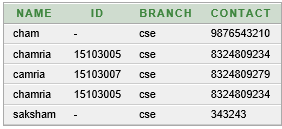
**LIKE-PREDIACTE:**This is used for pattern matching

SYNTAX: select \* from table\_name where name like ‘ba\_%’ or ‘ba%’

EXAMPLE\_1: select \* from students where name like 'ch%'

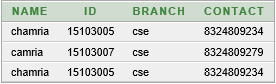


EXAMPLE\_2: select \* from students where name like '\_h%'or name like '\_a%'



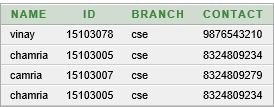
**IN**: This is used for for given a tuple for an entities whose values of any particular attribute is from given values

SYNTAX:select \* from table\_name where city IN(jalandhar,chandigarh)

EXAMPLE: select \* from students where contact IN(8324809279,8324809234)

**BETWEEN:** This is used for for given a tuple for an entities whose values of any particular attribute is from a given range

SYNTAX:select \* from table\_name where rollno BETWEEN 10 and 20 /\*both included\*/

EXAMPLE: select \* from students where id between 15103005 and 15103079

**GROUP FUNCION(aggregate fxn):**

**SYNTAX:**select group\_fxn(attribute\_value) from table\_name

EXAMPLE\_1: select AVG(id) from students

avg.PNG

EXAMPLE\_2: select MIN(id) from students

min.PNG

EXAMPLE\_3: select MAX(id) from students

max.PNG

EXAMPLE\_4: select SUM(id) from students

sum.PNG

EXAMPLE\_5: select count(\*) from students

count2.JPG

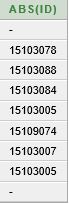
EXAMPLE\_6: select count(id) from students

count.PNG

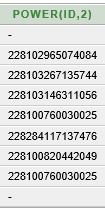
**NUMERIC FUNCTION**:

SYNATX:select numeric\_fxn(values) from table\_name

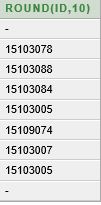
EXAMPLE\_1: select ABS(id) from students



EXAMPLE\_2: select power(id,2) from students



EXAMPLE\_3: select round(id,10) from students



EXAMPLE\_4: select sqrt(id) from students



**DUAL:**Dual is small work table which consist of only one row and column and contain the value of X into that column

SYNTAX:select 3\*5 from dual/\*we can perform arthematic operation direct on it\*/

dual.JPG

EXAMPLE:select \* from dual

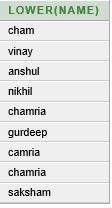
dummy.JPG

**LAB EXEPERIMENT NO.4**

**AIM:** To learn the use of string function

**LOWER\_FUNCTION:**

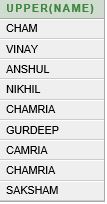
SYNTAX:select lower(attribute) from table\_name  
EXAMPLE: select lower(name) from students



**UPPER\_FUNCTION():**

SYNATX: select upper(attribute) from table\_name

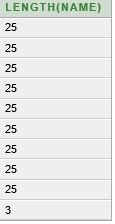
EXAMPLE: select upper(name) from students



**LENGTH\_FUNCTION():**

**SYNTAX:**select length(attribute) from table\_name

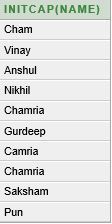
EXAMPLE: select length(name) from students



**INITCAP\_FUNCTION():**convert first letter in uppercase

**SYNTAX:**select INITCAP(attribute) from table\_name

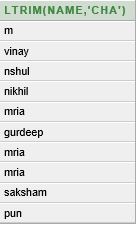
EXAMPLE: select initcap (name) from students



**LTRIM\_FUNCTION():**

**SYNTAX:**select LTRIM(attribute,trimming\_character) from table\_name

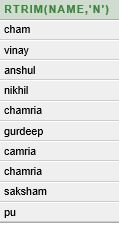
EXAMPLE: select LTRIM(name,'cha') from students



**RTRIM\_FUNCION():**

**SYNTAX:**select RTRIM(attribute,trimming\_character) from table\_name

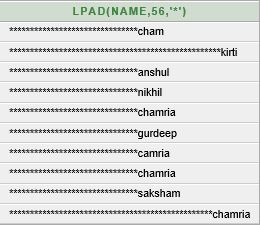
EXAMPLE: select RTRIM(name,'n') from students



**LPAD\_FUNCTION():**

**SYNTAX:**select LPAD(attribute,size\_of\_padding,character\_fill\_padding)

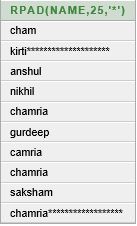
EXAMPLE: select LPAD(name,25,'\*') from students



**RPAD\_FUNCTION():**

**SYNTAX:**select RPAD(attribute,size\_of\_padding,character\_fill\_padding)

EXAMPLE: select RPAD(name,25,'\*') from students



**SUBSTR\_FUNCTION():**

**SYNTAX:**select SUBSTR(attribute\_name,start\_character\_number.,number of character) from table\_name

EXAMPLE**:** select SUBSTR(name,2,4) from students



**LAB EXEPERIMENT NO.6**

**AIM:** use of various type of key

**PRIMARY KEY:**

**TYPE\_1:**

**SYNTAX:** CREATE TABLE tablename (Column 1 datatype(size) primary key,Column 2 datatype(size). . . . . .);

**EXAMPLE:** create table employees (emp\_id int primary key,name varchar(20),designation varchar(30))

insert into employees (emp\_id,name,designation) values (123,'puneet','manager')

select \* from employees

keyinsert.JPG

insert into employees (emp\_id,name,designation) values (123,'nikhil','submanager')

violet.JPG

insert into employees (emp\_id,name,designation) values (124,NULL,'submanager')

select \* from employees



insert into employees (emp\_id,name,designation) values (NULL,'nikhil','submanager')

nullpri.JPG

**TYPE\_2:**form any primary key after declare all the attributes

**SYNTAX:** CREATE TABLE tablename (Column 1 datatype(size) ,Column 2 datatype(size). . . . . .,constraint xyz primary key(attribute\_name)/primary key(attribute\_name));

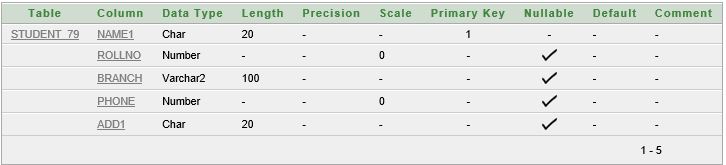
**EXAMPLE:** create table employees\_1 (emp\_id int ,name varchar(20),designation varchar(30),constraint xyz primary key(emp\_id))

123.JPG

**TYPE\_3:**make primary key of previously exist table

**SYNTAX:** alter table table\_name add primary key (attribute)

**EXAMPLE:** alter table student\_79 add primary key (name1)



**DROP PRIMARY KEY():**

**SYNTAX:**alter table table\_name drop primary key

**EXAMPLE:** alter table student\_79 drop primary key

dropprimary.JPG

**UNIQUE KEY:**

**TYPE\_1:**

**SYNTAX:** CREATE TABLE tablename (Column 1 datatype(size) unique,Column 2 datatype(size) unique. . . . . .);

**EXAMPLE:** create table employees\_6 (emp\_id int unique,name varchar(20) unique,designation varchar(30))

insert into employees\_6 (emp\_id,name,designation) values (123,'puneet','manager')

select \* from employees\_6

keyinsert.JPG

insert into employees\_6 (emp\_id,name,designation) values (123,'nikhil','submanager')

violet.JPG

insert into employees\_6 (emp\_id,name,designation) values (124,NULL,'submanager')

select \* from employees\_6



insert into employees\_6 (emp\_id,name,designation) values (null,'anshul','submanager')

select \* from employees\_6



**TYPE\_2:**form any unique key after declare all the attributes

**SYNTAX:** CREATE TABLE tablename (Column 1 datatype(size) ,Column 2 datatype(size). . . . . .,constraint xyz unique(attribute\_name)/unique(attribute\_name));

**EXAMPLE:** create table employees\_7 (emp\_id int ,name varchar(20),designation varchar(30),constraint asd unique(emp\_id)/unique(emp\_id))

123.JPG

**TYPE\_3:**make unique key of previously exist table

**SYNTAX:** alter table table\_name add unique(attribute)

**EXAMPLE:** alter table student\_79 add unique (name1)

ater.JPG

**DROP UNIQUE KEY():**

**SYNTAX:**alter table table\_name drop unique (unique\_key \_name)

**EXAMPLE:** alter table student\_79 drop unique(name1)

dropprimary.JPG

**FOREIGN\_KEY():**

**TYPE\_1:**

**SYNTAX:** CREATE TABLE child\_tablename (Column 1 datatype(size) primary key,Column 2 datatype(size). . . . . .,foreign key (column\_name),references parent\_table\_anme);

**EXAMPLE:** create table employees (emp\_id int primary key,name varchar(20),designation varchar(30))