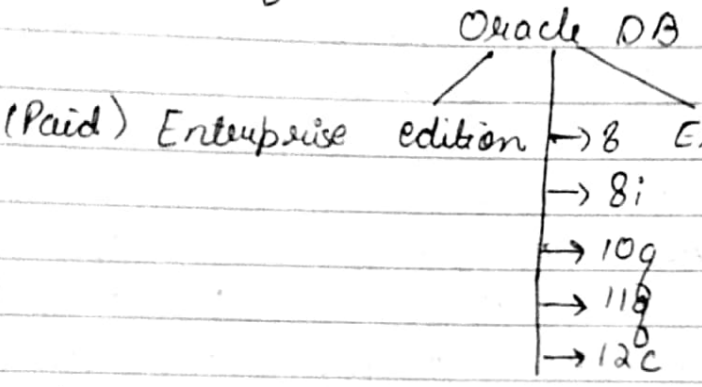
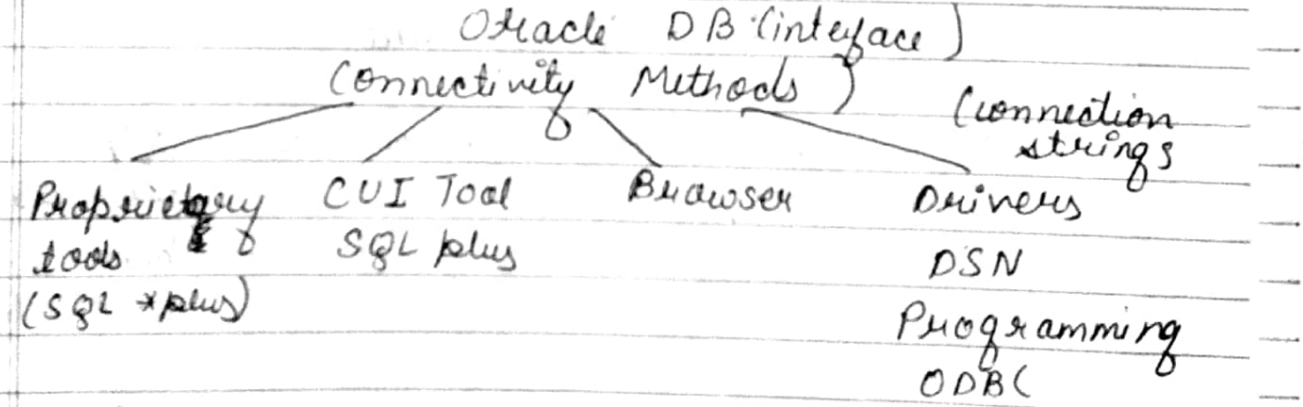
Practical

Dt. _____ B+
Pg. _____

Oracle database is one of the most powerful and popular DBMS used by thousands of companies as a backend software to manage their database. There are different versions and variations of Oracle



When we install Oracle a database server is created in the host machine to which other clients can connect using diff methods as shown below:



Drivers
(connection string)
programming
JDBC
ODBC
DSN

clear screen

Dt.
Pg.
B+

Oracle is a complex software & it requires lot of RAM and disc space. When Oracle is started and an instance is created in RAM which is also known as database engine through which we can connect to the data server. As Oracle is RDMS all its data is kept in the form of tables.

In order to connect to Oracle DBS we need a login name and permission to login. This login name is actually a user created by some powerful user. When we install Oracle some default users are created such as System (Manager)
SYS (Most powerful DBA)
HR (~~proactive~~) (practise Table)
SCOTT (Sample user)

To connect the command is — sqlplus
from OS
sql> — Connect
in Oracle

~~Enter~~ Every default user in oracle has got some fix pre defined tables that are used for server management and performance tuning besides practise. To see current user the command is SQL> show user OR
SQL> select user from dual;

Dt.
Pg.
B+

To see the list of tables — select * from tab
select * from cat;
select * from_tables;

19, 11

alter user hr account unlock;

disconnect - to disconnect from current user.

Creating user in Oracle

{conn system/oracle}

conn system/oracle
show user
create user secA61 identified by abcd;
conn
Enter user-name : secA61
Enter password : abcd.
ORA-01045: user ~~SYS~~ secA61 lacks create session privilege; login ~~design~~ denied.
Error:
Warning: You are no longer connected to ORACLE.

Without login permission you cannot login

conn system
Enter password : abcd.
connected
grant resource, connect to secA61;
grant succeeded.

Dt. Pg. B+

SQL> show user
user is "SCOTT"
SQL> select * from tab;
no rows selected

Q Visit Oracle.com, click on menu and select download and trials. In this section select db option & note down diff versions of oracle and MySQL database. Click on all database downloads

The SQL command can be broadly classified into following categories:
DDL (data definition language)
DML (data manipulation language)
DCL (data control language)

Data definition language (DDL)
All the commands related to any of the following comes under DDL.
(i) creation of table or schema definition
(ii) modification in table design
(iii) addition of a column
(iv) removal of column
(v) change of size of column
(vi) removal of table
(vii) creation of user
(viii) removal of user

Dt. Pg. B+

Creating a table:-

SQL> create table student (regno number, name varchar2(10), address varchar2(15));
Table created

DML (data manipulation language)
The queries related to following comes under DML
(i) insertion of data into table
(ii) view / selection of data from table
(iii) modification of data values
(iv) deletion of records / rows

eg: to view table data select * from student
no rows selected

DCL (data control language)
The SQL statements related to following comes under DCL
(i) grant of permission
(ii) revoking of permission
(iii) commit & transaction
(iv) rollback of transmission.

viewing the structure of the table
the structure of the table is also known as table schema if it includes field names & data types

Dt. Pg. B+

SQL> describe student;

Name	NULL?	TYPE
Rollno		Number
Name		varchar2(10)
Address		varchar2(25)

different ways to add contents into table

(1) Inserting data into all columns

SQL> insert into student values (56, 'Amit', 'Dehradun');
1 row created

SQL> Select * from student

Name	Rollno	Address
Amit	56	Dehradun

SQL> commit;
commit complete

when you commit the data all DML changes done till now are made permanent and updated in hard disk

(2) Inserting data in any (selected) column.

There are diff ways to insert data in selected columns.

SQL> Insert into Student1(Rollno, city) values (40, DDN);
(name of table)

e-COPY

Dt. Pg. B+

(3) Inserting values in selected columns with different order

SQL> insert into student2(Balance, name) values (888, 'ABCDEF');

(4) Inserting values in selected columns without specifying columns

SQL> insert into student1 values(77, NULL, 'HRD', NULL);

(5) Inserting values using substitution method.

SQL> insert into student1 values (&serial, '&NAME', '&LOCA', &money);

value for serial: 77
" " naam: kapil
" " location: Delhi
" " money: 66

old 1: insert into student1 values (&serial, '&naam', '&loca', &money)
new 1: insert into " " (77, 'kapil', 'Delhi', 66)
1 row created

SQL> /
Enter values for serial: - -

(if you don't want to enter
numerical → NULL
varchar → ↵

e-COPY

Dt. B+
Pg.

Environment Variables

Oracle provides no. of environment variables for formatting the output screen.

To see the list of oracle environment variables the command is

SQL> show all

To change pagesize

↳ SQL> set pagesize 10

To see current page size

↳ SQL> show all

To change line size

↳ SQL> set linesize 30

SQL> set heading off

SQL> set heading on

The changes in environment variables are valid only during session

8.02.16

Modification of data

" " " also comes under DML. There are diff. ways to change table values.

(i) Changing all the values of a column:

SQL> update student set name = "anything"

Dt. B+
Pg.

(ii) Removing all the values of a column

SQL> Update student set name = NULL;

(iii) Changing selected values

SQL> update student set name = 'ABCD' where Roll no = 1;

(iv) Changing multiple values

SQL> Update student set name = 'ABCD' where roll no = 1, 3, 5;

error at line 1;

SQL> Update student set name = 'abcd' where roll no = 1 OR 3;

⟹ error.

SQL> update student set name = 'abcd' where rollno = 1 or roll no = 3;

2 rows updated.

(iv)

SQL> update student set name = 'Aany' where rollno = 1 and name = 'kapil';

0 rows update

(v) Changing multiple values without OR operator

SQL> Update student set name = 'Any' where rollno in (4, 5, 6, 7, 8)

3 rows updated

Dt.
Pg. B+

Deleting Table values

Table values can be deleted using multiple ways.

(i) deleting all records

SQL> delete * from sample1;
error at line 1
(delete does not work in delete)

SQL> delete from sample1;
6 rows deleted. (DML query)
However the table is not removed.
* Delete command can be rolled back

(ii) deleting all records permanently.
If you don't want to roll back deleted records. this command be used;

SQL> truncate table SAMPLE1;
(DDL command)
no rows selected
* Truncate cannot be rolled back
* table is not deleted
* Delete is DML, truncate is DDL
* Delete can be rolled back, truncate cannot.
* Delete can be conditioned, but truncate cannot.

SQL> Delete from sample1 where rollno>5 OR name='Kapil';
2 rows deleted.

Dt.
Pg. B+

Creating dublicate copy of a table.

We can create copy of a table in diff ways such as.

(i) creating exact copy
SQL> create table student2 as select * from student

(ii) creating a dublicate copy with selected records.
SQL> Create table student as select * from student where ROLL no>5;

(iii) creating duplicate copy of a table with diff column order.
SQL> create table student as select city, name, balance, rollno, from student;

(iv) creating duplicate copy with diff column names
SQL> create table student3(serial no, naam, shehar, money) as select rollno, name, city, balance from student

(v) creating duplicate copy without any records
SQL> create table student as select * from student where 1>2;
* it is not necessary to give column name after "where"

(vi) using filter in select clause/query
We can apply diff. type of condition & filters in select query.
These filters can also be used in update & delete queries as well.

% ↓ overall _ ↓ one

Dt. Pg. B+

(i) display all records
SQL> select * from student

(ii) display selected records
SQL> select * from student where rollno > 10;

SQL> select * from student rollno > 10 or city in ('KANPUR', 'DELHI');
no rows selected
(case sensitive)

(iii) pattern matching
to display the names of those students whose name starts with K
SQL> select * from student where name like 'K%';

SQL> select * from student where name = 'A%';
100% ↓ equality
('A%')

pattern matching works only with 'like' operator

(iv) display records of those students whose name ends with 't'
SQL> select * from student where name like '%t';

name of those who have 2nd letter as 'a'

e-COPY

Dt. Pg. B+

SQL> select * from student where name like '_a%';

details of those who do not belong to delhi
SQL> select * student where city not in('Delhi');
8 rows selected
OR
SQL> select * from student where city <> 'Delhi';
(space)
OR
SQL> selod * from student where city != 'Delhi';

SQL> update student set email = 'ABC@gmail.com' where rollno = 1;
SQL> alter table student drop column name;
SQL> truncate table student;
SQL> drop table student;

Oracle Constraints

SQL provides diff types of constraints to restrict or control data & values. The constraints are:
- Primary key
- Foreign key
- Unique key
- Not null
- Check
- Default parameter.

e-COPY

Primary key: The primary key constraint allows unique and distinct values in a column it also helps in ~~removing~~ preventing duplicate rows.

Some properties of primary keys are

1.) There can be only 1 primary key in a table.
2.) Any type of column can be made primary key
3.) A table can be created without primary key also
4.) A PK column does not allow duplicate values
5.) A PK column does not allow null or ~~activalues~~ empty values
6.) A PK ~~column~~ can be made up of either a single column or multiple columns
7.) A single column PK is known as atomic key or simple key
8.) A multiple column PK is known as composite key or complex key.

Different ways of creating primary key.

(1.) Creating PK during table creation at column level

SQL> create table sample(Rollno number primary key, name varchar2(10));

Table created.

SQL> Desc sample;

Name	Null?	Type
Rollno	Not Null	Number
Name		varchar2(10)

SQL> Insert into sample values(3, 'Amit');

1 row created

SQL> /

error. The above example creates atomic primary key i.e. single column PK

(2.) Creating PK during table creation at table level or row level.

SQL> create table sample(rollno number, name varchar2(10), city varchar2(10), Primary key(Rollno));

(3.) Creating PK after table was creation (on existing table) We can add primary key in existing table only when the current column values do not conflict with PK definition

SQL> alter table student add primary key (rollno);

Composite Primary key

SQL> Create table sample(rollno number PK, name varchar2(10) PK, city varchar2(10));

→ Error

Dt. Pg. B+

SQL> create table sample (orderno number, itemno number, qty number, price number, Primary key (orderno, itemno));

→ Table created.

(1) In case of composite ~~key~~ PK single prime columns cannot accept null values

22/2/18 **Deleting Primary Key**

The primary key can be removed using alter table ORA. command and no data is removed.

SQL> alter table sample drop primary key;

Unique constraint - The unique constraint does not allow duplicate values.

There can be many unique columns in a single table

The unique constraint is not visible in the structure.

SQL> create table sample(rollno number primary key, name varchar2(10) unique);

table created

SQL> desc table

Name	NULL	Type
Rollno	Not Null	Number
Name		varchar2

Dt. Pg. B+

Unique column accept NULL(even multiple) ~~null~~ values because NULL cannot be compared. There is no question of equality.

Not NULL constraint - It allows duplicate values.

There can be only one unique column in a single table.

The not null constraint is visible in the structure

~~There can~~ It does not accept NULL values

SQL> Create table sample(rollno Primary key, name varchar2(10) unique, age number not null, city varchar2(10) unique, phno number NOT NULL);

Table created

SQL> Desc Table.

Name	NULL?	Type.
Rollno	Not Null	Number.
Name		varchar2(10)
Age	Not Null	number.
City		varchar2(10)
Phno	Not Nul	number

Dt. B+
Pg.

We can apply multiple constraints on a single column as well without using & , (comma) operator.

logically primary key = unique + not null

SQL> create table sample (adhar number primary key, pan number unique not null, passport unique not null);
Table created.

SQL> Desc table.

Name	Null?	Type.
adhar	Not Null	Number
pan	Not Null	Number
passport	Not Null	Number

Check constraint
Check constraint is used to specify the range of values for a column.

SQL> create table sample (rno number primary key, age number check (age>17));
↳ And age >15
Table created

you can also make age unique & not null.

e-COPY

Dt. B+
Pg.

Check constraint is not displayed in structure.

SQL> ~~create table sample (roll number primary key, name varchar2(15) check (name = 'aa'));~~

SQL> create table sample(roll number primary key name varchar2(10) check (name like 'a%'))

Default Parameter
Default is not a constraint it only helps in faster data entry

SQL> create table sample (roll number, name varchar2(10), city varchar2(10) default 'ddn')
Table created.

SQL> insert into sample values(1, 'Amit', default);
1 row created

SQL> insert into sample(rollno, name) values(3, 'Rapii')
1 row created.

In both the rows dehradun will be entered by default

[U can add unique default not null check all together
[PR me check ~~table~~ kr sakte hain]

e-COPY

Dt. Pg. B+

Foreign Key

A foreign key attribute of a table is dependent on another column for its values in other words if a column is a foreign key then it can have only those values that are present in its parent column.

Foreign key attribute is also known as referencial integrity constraint and primary key is known as entity integrity constraint.

A table can have many foreign keys

A column can be parent key for multiple foreign key columns

Foreign key can be both atomic or composite.

Foreign key sometimes is also known as child key.

Both parent key and child columns can be in same table or in two different tables

Foreign key can be applied at both column level & table level

Dt. Pg. B+

Foreign key helps in maintaining integrity of data

A foreign key can refer only to a column which is either primary key or unique.

column level

SQL> create table marks (rollno number references student, hindi number)

table created (if rollno in student is PK)

SQL> insert into marks(3,44);
table created (3 is in table student)

SQL> insert into marks(33,55)
error (33 is not in table student)

SQL> drop table student
error.
unique/primary key in table referenced by foreign key.

* Student table is known as parent table, marks table is known as child table.

* The parent table cannot be dropped before child table

alter table column add primary key (rno);

Table level

SQL> create table marks (rollno number, hindi number, foreign key (rollno) references student);

the parent key and the child can both be in the same table.

SQL> create table sample (Ecode number primary key, ename varchar2(10), Mgr number references sample);

table created

* foreign key can contain NULL and duplicate values

SQL> insert into sample values (1, 'Amit', NULL);
1 row created

SQL> insert into sample values (2, 'kapil', 1);
1 row created

SQL> insert into val sample values (5, 'kapil', 9);
error

* A table can have multiple foreign keys from different tables.

e-COPY

SQL> create table multi (serial number references student, regis number references sample, amount number);
table created

* we can apply multiple constraints in foreign key.

SQL> create table marks (roll number primary key references student (rollno), hindi number);
optional but mandatory to refer unique key
table created

SQL> desc table

22/3/18

Handling Data in Oracle

Data is a complex data type in oracle & it stores various information not visible to the user in general

The default date format of oracle is:-
DD-MON-RR
earlier it was DD-MON-YY

Eg: SQL> create table sample (rollno number, DOB date, DOJ date);
Table created.

SQL> desc sample;

e-COPY

Dt. Pg. B+

Name	NULL!	Type
Rollno		Number
DOB		date
DOJ		date

We can enter either system date in date column, or user specific date.

SQL> insert into sample values (1, '5-Dec-67', sysdate);
1 row created

SQL> insert into sample values (3, '5-Dec-1650', '5-Jan-1455');

Rollno	DOB	DOJ
1	5-Dec-67	22-Mar-18
3	5-Dec-50	5-Jan-55

* To view full date to_char function is used

SQL> select rollno, DOB, To_char (DOJ, 'dd month yyyy');
from sample;

Rollno	DOB	To_char
		5 December 1967
		25 december 1650

e-COPY

Dt. Pg. B+

These were the various methods to view the data

Whenever you save a date value, time is also saved along with time zone, time zone offset and NLS - date format

To view time →
SQL> select rollno, to_char (DOB, 'dd mm yy hh mi ss') from sample;

e-COPY

21/04/18

Grouping

Grouping - When we use a function on a table it either returns a single result for all the rows

for eg- consider the following table:

Course2

Courseccode	Coursename	Coursefees	Coursesemester	Depts
101	MCA	80000	6	MCA
102	MBA	50000	4	MGMT
103	BTech	45000	8	ENGG
104	BCA	45000	6	MCA
105	BSCIT	25000	6	Allied
106	MSCIT	35000	6	Allied
107	BBA	35000	6	MGMT
108	OCP	30000	2	MCA
109	PGDM	25000	3	MGMT
110	CCNA	45000	3	ENGG

10 rows selected

Display all course name in small letters

SQL> select lower(coursename) from course2;

Uppercase

SQL> select upper(coursename) from course2;

To count

SQL> select count(*) from course2;

COUNT(*)
10

display total coursefees of all courses.

SQL> select sum(coursefees) from course2;

display department wise total coursefees of all courses where dept='MCA';

(Avoidable method)

SQL> select sum(coursefees) from course where dept='MCA';

The above method is not suitable.

∴ We will use 'group by' class

The group by class creates internal group in RAM and then applies function on each group.

for eg-

SQL> select sum(coursefees) from course2 group by dept;

MCA

101	MCA	80000	6	MCA
104	BCA	45000	6	MCA
108	OCP	3000	2	MCA
		128000		

MGMT

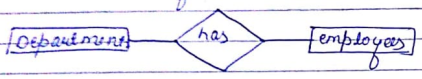
102	MBA
107	BBA
109	PGDM

Dt. Pg. B+

22/4/19

In RDBMS the data is always spread accross multiple table & we cannot view complete information from one table

For eg- A company has no. of departments and in each department there are many employees. Draw its E-R diagram & convert it into tables.



Suppose, following table is created :-

Ecode	Ename	dcode	dname	dlocation
1	A	10	Sales	delhi
2	B	10	Sales	delhi
3	A	20	HR	Mumbai
4	C	20	HR	Mumbai

As we can see that the above single table is wrong & it has anomalies.

For eg → wherever department no. 10 appears its name 'sales' & location 'delhi' is repeated which is redundancy anomaly. Therefore the correct solution is to design following 2 tables

e-COPY

Dt. Pg. B+

Employee

PK ← Ecode	Ename	FK → dcode
1	A	10
2	B	10
3	A	20
4	C	20

Department

dcode	dname	dlocation
10	Sales	delhi
20	HR	Mumbai
30	Accounts	HRD

Subquery

A subquery is a query which is written inside another query. The output of subquery is not displayed on the screen rather it is passed to the outer query.

For eg → Refer to the same course table & answer the following queries

Display the highest coursefees

SQL> select max(coursefees) from course;

MAX(COURSEFEES)
80000

Display second highest coursefees

SQL> select max(coursefees) from course where courseCode != 101;

e-COPY

Dt. Pg. B+

SQL> select max(coursefees) from course where coursefees <> 80000

The above query will give correct answer but the method is wrong.

⟹ select max(coursefees) from course where coursefees <> max(coursefees)

This query appears to be correct but using a function directly on the RHS of a relational operator is not allowed

Correct

∴ The correct method is to use a subquery.

Correct → select max(coursefees) from course where coursefees <> (select max(coursefees) from course);

Another method

select max(coursefees) from course where coursefees not in (select max(coursefees) from course);

e-COPY

Dt. Pg. B+

A subquery can be nested several times

Display third highest coursefees

Wrong → SQL> select max(coursefees) from course where coursefees not in(select max(coursefees) from course where coursefees not in (select max(coursefees) from course));

The above query will give wrong o/p as 80000 Similarly if you want to access nth highest value nested subquery will not work.

e-COPY

24/4/18

* Subquery can display data only from one table but data might be related from 2 or more tables. However, a join can display data from single table as well as multiple tables. This means that some of the queries can be answered by both subquery and join. And some of the queries can only be answered by join

* Register

Rollno	Coursecode
4	103
2	101
3	101

* Student —

Rollno	Name	City	Balance
1	amitabh	Kanpur	500
2	rohit	Kanpur	
3	kamal	Kanpur	500
4	mudit	Kanpur	500
5	shashwat	"	
6	kumar	"	500
7	rohit	"	500
8	akash	"	
9	A sharma	Delhi	500

e-COPY

* Course —

Code	Course	Fees	Semester
101	MCA	80000	6
102	MBA	50000	4
103	B.Tech	4500	8
104	BCA	30000	6
105	BscIT	25000	6
106	Msc IT	35000	6

display the names of all students
SQL> select name from student;

display the names of those students who are not having any balance
SQL> select name from student where balance = null;
⇒ 0 rows selected
The o/p is incorrect as null is not compared to null.

The correct query is —
SQL> select name from student where balance is null;

display the details of those students who are registered in any course.

(wrong method ↓)
(1.) SQL> select * from student where rollno = 2 or rollno = 3 or rollno = 4;
(2.) SQL> select * from student where rollno in (2, 3, 4);

e-COPY

Dt. B+
Pg.

(correct method by using subquery ↓)
SQL> select * from studtbl where rollno in (select rollno from register);

while using subquery you should be careful in relation operators.
For eg - consider the following query

SQL> select * from student where rollno = (select rollno from register);

the above query is syntactically correct but it gives runtime error.

SQL> select * from student where rollno = (select rollno from register where courseccode = 103);

the above query will work so it is better to always use 'in' operator rather than '=' operator

SQL> select * from student where roll no in (select rollno...);

We can write a subquery within another subquery and so on. However the data will be displayed from outermost query. and you need to have atleast 1 common attribute between 2 tables within subquery.

E.G)

Dt. B+
Pg.

display the details of those students who are registered in MCA course
SQL> Select * from student where rollno in (select rollno from register where courseccode ~~from course~~ in (select courseccode from course where course = 'MCA'));

display the details of those courses in which the student of kanpur are registered.
SQL> select * from course where courseccode in (select courseccode from register where rollno in (select rollno from student where city = 'kanpur'));

display total no. of registration in each course
SQL> select courseccode, count(*) from register group by courseccode;